REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DE BATNA2
FACULTE DE TECHNOLOGIE
DEPARTEMENT GENIE INDUSTRIEL

THESE

Présentée au Laboratoire d'Automatique et Productique
En vue de l'obtention du diplôme de

DOCTORAT EN SCIENCES

Spécialité

Génie Industriel

Par

Ouahiba CHOUHAL

Magister en Informatique de l'université de Tébessa

Titre

Contribution à la surveillance des systèmes de production par les Systèmes Multi-Agents Collectifs

Soutenue le: 08/03/2018

Devant le jury:

Président	Kinza Nadia MOUSS	*Professeur*	Université de Batna 2
Rapporteur	Leila Hayet MOUSS	*Professeur*	Université de Batna 2
Examinateur	Okba KAZAR	*Professeur*	Université de Biskra
Examinateur	Mohammed BEN MOHAMMED	*Professeur*	Université de Constantine 2
Examinateur	Abdelkamel TARI	*Professeur*	Université de Bejaia
Examinateur	El kamel MERAH	*MC-A*	Université de Khenchela

Année universitaire: 2017-2018

Remerciements

Tout d'abord, je remercie Dieu de tout puissant de m'avoir donné le courage et la patience durant toutes ces années d'études.

Je tiens à exprimer ma profonde gratitude et ma reconnaissance envers ma Directrice de thèse, **Mme. Leila Hayet MOUSS**, Professeur à l'université de Batna 2 et directrice de Laboratoire d'Automatique et Productique « LAP » de m'avoir accueilli au sein de son équipe 3S « Surveillance – Supervision- Sûreté », pour la confiance qu'elle m'a prodiguée, pour ses encouragements continus, sa disponibilité, pour le suivi et la direction de mon travail, ainsi pour ses conseils judicieux.

Par ailleurs, je remercie vivement **Mlle. N.K. MOUSS**, professeur à l'université de Batna 2, qui ma fait l'honneur de présider le jury de ma thèse.

Je tiens également à présenter mes sincères remerciements à **Mr. M. BENMOHAMED,** Professeur à l'université de Constantine 2, **Mr. A. TARI**, Professeur à l'université de Bejaia, **Mr. O. KAZAR** professeur à l'université de Biskra et, **Mr. E. MERAH**, Maitre de Conférences à l'université de Khenchela pour leur participation à l'évaluation scientifique de ce travail.

Je n'oublie évidement pas mes collègues du LAP.

Dédicaces

A ma Mère

A mon père

A mon mari

A mes filles ***Amina, meriam*** *et* ***manel***

A Mon fils ***Chouki Haydar***

A mes frères et mes sœurs

A tous ceux qui me sont chers.

Je dédie ce modeste travail

Ouahiba

Résumé

Compte tenu de la complexité des systèmes de production actuels et dans le but d'améliorer leur productivité et augmenter leur fiabilité, l'objectif de cette thèse réside dans la conception d'une architecture distribuée pour la surveillance des systèmes de production. Cette architecture est réalisé à l'aide de la technologie à base d'agents coopérants et la technologie SOA à base des services web. La Surveillance Distribuée à base d'Agents Coopérants proposée et baptisée SDAC utilise à la fois les techniques des communautés de recherche FDI et DX est supportée par une application informatique AI-SDAC est couplé avec les concepts du SOA pour l'interopérabilité dans une architecture distribuée coopérante et interopérable SOA-SDAC permettant la surveillance multi-sites. Le système de la clinkérisation de la cimenterie d'Ain Touta a été choisi comme champ d'application privilégie.

Mots clés : Système de production, Surveillance coopérative, Diagnostic à base de cohérence, Surveillance distribuée à base d'agents, Surveillance distribuée à base de services web, SOA, Système Multi Agent.

Abstract

Given the complexity of current production systems and in order to improve their productivity and increase their reliability, the objective of this thesis is to design a distributed architecture for the monitoring of production systems. This architecture is realized with the help of cooperating agent technology and SOA based on web services. The distributed monitoring based on the cooperating agents proposed and named SDAC uses both the FDI and DX research community techniques is supported by an AI-SDAC computer application and coupled with SOA concepts for interoperability in a cooperative distributed architecture and Interoperable SOA-SDAC for multi-site monitoring. The clinkering system of the Ain Touta cement plant has been chosen as the preferred field of application.

Keywords: Manufacturing system, Cooperative monitoring, Diagnosis based on coherence, Distributed agent-based monitoring, Distributed monitoring based on web services, SOA, Multi-Agent System.

ملخـــــص

نظرالكون أنظم الإنتاج الحالية جد معقدة ومن أجل تحسين إنتاجيتها وزيادة موثوقيتها،فإن الهدفالأساسي من هذه الأطروحة هو تصميم بنية موزعة لرصد و مراقبة هذه النظم.هذه البنية انجزت بمساعدة تكنولوجيا العملاءالمتعاونون و تكنولوجياSAO التي تعتمد على الخدمات الموجه على الإنترنت.المراقبة الموزعة على أساس العملاء المتعاونين وتعمد SDACتستخدم كلا من تقنيات FDIو DXو تعتمد على البرنامج المعلوماتي -AI SDACإلى جانب مفاهيمSAO من اجل التوافقية للتشغيل المتبادل مشكلة بذلك بنية موزعة متعاونة قابلة للتشغيل المتبادل التي تسمح بمراقبة مواقع متعددة. وقد تم اختيار نظام مصنع الإسمنت عين التوتة كنطاق للتجارب.

الكلمات المفتاحية: نظام الإنتاج، مراقبة موزعة ،تصحيح الاعطاب بناء على الاتساق، مراقبة موزعة بواسطةالعملاء،مراقبة موزعة بواسطة الخدمات الموجه على الإنترنت، SAO، نظام متعدد العملاء.

Table des matières

Liste des figures
Tableaux
et
Abréviation

Introduction générale

Chapitre 1: Surveillance distribuée des systèmes complexes

Chapitre 2: Modes d'interaction dans les systèmes multi agents

Chapitre 3 : Approche méthodologique pour la Surveillance Distribuée à base d'Agents Coopérants: SDAC

Chapitre 4: Application Informatique support à la Surveillance Distribuée à base d'Agents Coopérants: AI-SDAC

Chapitre 5: Architecture interopérable à base de services web pour la surveillance multi-sites

Chapitre 2: Modes d'interaction dans les systèmes multi agents

Chapitre 3 : Approche méthodologique pour la Surveillance Distribuée à base d'Agents Coopérants: SDAC

Chapitre 4: Application Informatique support à la Surveillance Distribuée à base d'Agents Coopérants: AI-SDAC

AI-SDAC	Application Informatique support à la Surveillance Distribuée à base d'Agents Coopérants
FDI	Fault Detection and Isolation
FIPA-ACL	FIPA Agent Communication Language
IA	Intelligence Artificielle
IAD	Intelligence Artificielle Distribuée
SdP	Système de Production
SDAC	Surveillance Distribuée à base d'Agents Coopérants
SMA	Systèmes Multi-Agents
SOA	Service Oriented Architecture
SOA-SDAC	Service Oriented Architecture- Surveillance Distribuée à base d'Agents Coopérants
SOAP	Simple Object Access Protocol
TIC	Technologies d'Information et de Communication
UDDI	Universal Description Discovery and Integration
UML	Unified Modeling Language
WSDL	Web Services Description Language
XML	eXtensible Markup Language

Introduction générale

Avec les évolutions technologiques, informatiques et organisationnelles, les systèmes industriels sont de plus en plus complexes. Afin d'assurer leur bon fonctionnement, la surveillance industrielle tient une position stratégique dans leur pilotage en fournissant l'état de fonctionnement des différents équipements industriels.

La surveillance classique de ce type de système, par un seul et unique organe, à partir d'une vue globale du système physique contenant la description de ce dernier dans sa totalité, pose évidemment des problèmes de fiabilité dû à la complexité du système physique à diagnostiquer et à l'architecture figée et peu robuste du système de surveillance.

Dans ce contexte, la distribution tend à doter la surveillance de caractéristiques leur permettant de palier aux carences existantes au niveau des architectures conventionnelles (centralisée, hiérarchique…). La conséquence la plus visible est une amélioration de la disponibilité et de la fiabilité du système de surveillance, d'où une augmentation des performances et des gains de l'entreprise

Le travail que nous présentons dans le cadre de cette recherche se propose d'apporter une contribution au domaine de la surveillance distribuée des systèmes complexes et plus particulièrement au niveau de la fonction diagnostic. Il s'intéresse à la mise en place d'un système de surveillance en mesure d'analyser un flot continu d'alarmes reçues par un centre de supervision et d'en donner une interprétation plus compréhensible pour l'opérateur.

L'idée est d'adopter une approche distribuée en décomposant les problèmes de la surveillance en une multitude de problèmes élémentaires et construisant des entités autonomes qui pouvant, en coopérant, participer à la construction d'une solution globale. Nous parlons alors d'une surveillance collective. Cette approche s'appuie sur les techniques de diagnostic dites à base de modèles et sur le paradigme multi-agents où chaque agent a une tâche à réaliser et les interactions entre les agents permettent la réalisation d'un but global.

Aussi, face à la complexité des systèmes de production actuels et dans le but d'améliorer leur productivité et d'augmenter leur fiabilité, les systèmes de surveillance expriment un grand besoin d'ouverture et de coopération à l'échelle mondiale. Ils ont besoin de s'allier à d'autres systèmes de compétences complémentaires afin de coopérer avec, et de renforcer l'exploitation sûre des systèmes qui ne sont pas à la portée d'un seul

système de surveillance. Cependant, pour coopérer, les systèmes de surveillance doivent adopter de nouveaux schémas de comportement, modifier éventuellement leur organisation et à s'ouvrir davantage à leur environnement et s'organiser en réseaux. Pour faire face à ces nouvelles exigences, nous nous intéressons aux architectures-orientées services (SOA) et les services web pour assurer l'interopérabilité des systèmes de surveillance multi-sites.

Problématique

Dans l'approche distribuée que nous proposons, chaque entité a une vue partielle ou locale du système à surveiller et particulièrement à diagnostiquer. Par conséquent, pour construire une solution globale, logique et cohérente il va falloir rassembler les solutions partielles. Donc, il faudrait que les entités coopèrent entre elles afin de partager leurs solutions et faire part de leurs problèmes et coordonner leurs activités.

L'un des défis de ce travail est de se poser les questions suivantes :

- Quels sont les algorithmes et protocoles à mettre en place pour la communication, la coopération et la coordination pour ces entités de surveillance coopérants ?
- Comment doter ce système de surveillance distribuée d'une capacité interopérabilité i.e. capacité de communication et de coopération avec d'autres systèmes de surveillance distants et d'accéder à leurs fonctionnalités via le réseau mondial internet ?

Dans notre travail de recherche, nous avons constaté que les questions concernant la surveillance collective multi-sites sont plus nombreuses que les réponses. Nous abordons donc cette recherche avec quelques questions comme :

- Quel type d'architecture pour la modélisation et la conception des systèmes de surveillance distribuée ?
- Quel type d'architecture pour gérer l'ouverture et la coopération des systèmes de surveillance distribuée à l'échelle mondiale indépendamment de toutes spécificités de ces systèmes, i.e. technologies, normes et standards, plates formes et logiciels, etc. ?
- Quelle architecture logicielle pour supporter la surveillance distribuée coopérant et interopérable?

- Quelles sont les technologies et les outils logiciels facilitant l'intégration et l'implémentation de la distribution dans telle type de systèmes ?

Contributions

Dans le cadre de cette thèse nous tentons à répondre à toutes ces questions avec comme objectif principal la conception d'une architecture distribuée interopérable comme support de la surveillance collective permettant la réduction de la complexité, la tolérance aux fautes, la réduction des coûts, l'amélioration des caractéristiques tout en maintenant la robustesse et la flexibilité.

La solution que nous proposons repose sur une approche d'intégration des concepts des services web et des agents logiciels en une entité cohérente qui tente de dépasser la faiblesse de chaque technologie, tout en renforçant leurs avantages individuels, afin de concevoir et d'implémenter une telle architecture logicielle.

Afin de gérer la distribution, nous proposons de développer un modèle multi-agent que nous avons baptisé SDAC (Surveillance Distribuée à base d'Agents Coopérants).

SDAC concerne les deux phases indissociables de la surveillance à savoir la détection et le diagnostic pour lesquelles il existe une large panoplie de méthodes et de techniques proposées par différentes communautés de recherche (FDI, DX). Nous avons choisi d'adopter une approche qui vise à utiliser conjointement les méthodes de surveillance issues de la communauté FDI pour la phase de détection, et les méthodes développées par la communauté DX pour l'étape de diagnostic afin de pouvoir tirer profit des avantages de chacune et avoir ainsi une certaine complémentarité.

Dans notre travail, nous nous sommes intéressés spécialement à la distribution de l'étape de diagnostic en utilisant la méthode dite du diagnostic à base de cohérence pour l'élaboration des diagnostics locaux. Ainsi, SDAC permet une coopération entre les agents de diagnostic locaux responsables du calcul des diagnostics locaux en garantissant la cohérence de leurs résultats pour établir le diagnostic global sans aucun superviseur. Pour illustrer SDAC, une application informatique AI- SDAC (Application Informatique support à la Surveillance Distribuée à base d'Agents Coopérants) est alors développée pour un cas d'étude réel. Le système de la clinkérisation de la cimenterie d'Ain Touta a été choisi comme champ d'application.

Cependant, le modèle SDAC présente certaines limites comme le manque d'interopérabilité car il ne permet pas une communication et une collaboration indépendantes des technologies, normes et standards, plates formes et logiciels des systèmes de surveillance distants. Pour assurer cette interopérabilité, notre modèle est alors couplé avec les concepts du modèle de référence pour l'interopérabilité SOA dans une architecture distribuée coopérante et interopérable SOA-SDAC permettant une collaboration machine-machine pour la surveillance distribuée, multi-sites. Cette architecture a été illustrée sur le même cas d'étude dont l'objectif est de montrer la faisabilité de la stratégie adoptée.

Organisation du manuscrit

Le manuscrit est organisé en deux parties. La première présente les fondements théoriques sur lesquelles nos travaux sont basés est organisée en deux chapitres. La deuxième présente notre contribution est quant à elle organisée en trois chapitres.

Après quelques définitions sur les concepts de la surveillance et l'intérêt d'une architecture distribuée de la surveillance, le premier chapitre est consacré à un état de l'art sur les méthodes de surveillance des processus industriels que ce soit des méthodes à base de modèle ou celles issues de l'IA. Ce chapitre traite la problématique de surveillance dans les systèmes complexes et distribués et leur mise en œuvre sur la base des technologies multi agent et service web en présentant quelques travaux de la littérature.

Le deuxième chapitre présente une synthèse sur l'interaction telle qu'elle est perçue par la communauté de chercheurs en SMA. Nous aborderons tout d'abord le concept d'agent et de SMA. Nous axerons sur les formes d'interaction dans les SMAs comme la coopération, la coordination et la communication.

Le troisième chapitre, le cœur de notre travail, présente la solution proposée à base de Multi Agent pour la Surveillance Distribuée à base d'Agents Coopérant pour les systèmes complexes. Nous présentons les principales caractéristiques de SDAC et les modèles sous-jacents. Cinq modèles ont été développés pour déployer notre proposition. Il s'agit du modèle d'agent, tâche, expertise, coordination et conceptuel.

Le modèle d'expertise se taille la part du lion. Il modélise les connaissances, spécifie le diagnostic selon deux approches quantitative et qualitative et définit un algorithme de diagnostic distribué.

L’objectif principal du quatrième chapitre est de valider les aspects théoriques de SDAC développés en présentant une application informatique AI-SDAC qui implémente un SMA dans un procédé industriel de fabrication du ciment.

Le cinquième chapitre est réservé à la technologie service web pour l'interopérabilité des systèmes collaboratifs. Nous présentons l’architecture distribuée et interopérable SOA-SDAC pour la surveillance multi-sites qui intègre les concepts du modèle multi-agent SDAC et ceux du modèle SOA. Ceci est réalisé par la description de ces composantes et de son fonctionnement. La dernière partie de ce chapitre est consacré à la présentation d’une application industrielle qui implémente un producteur de services web. Ces services web regroupent un ou plusieurs savoir-faire dans le domaine de la surveillance.

La conclusion et les perspectives clôturent ce manuscrit en présentant un bilan du travail effectué et un ensemble de perspectives liées notamment à la poursuite de ce travail ainsi qu’aux nouveaux thèmes de recherche qui nous paraissent les plus pertinents.

La figure.1 résume les différents chapitres présentés ainsi que leur articulation. Nous remarquons que les fondements théoriques sur les SMA sont impliqués fortement pour la conception du modèle multi agent SDAC et dans la réalisation de la plate forme informatique qui le supporte: AI- SDAC. Les services web sont utilisés pour assurer l’interopérabilité multi site on proposant SOA-SDAC.

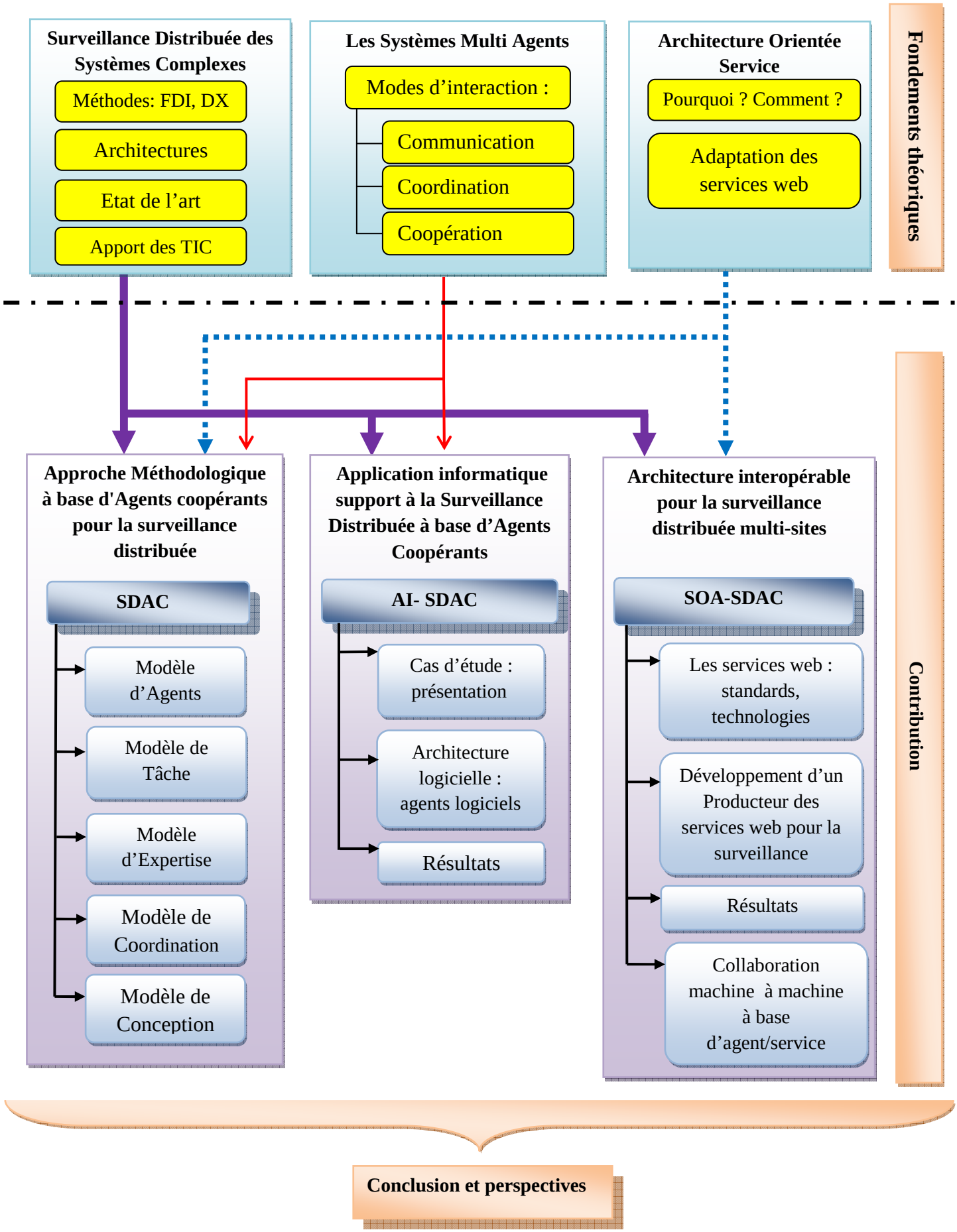


Figure 1. Organisation du mémoire.

Chapitre 1

Surveillance distribuée des systèmes complexes

Les systèmes industriels recouvrent de nombreuses formes. Aujourd'hui, ils sont complexes, de grande taille et géographiquement distribués. L'adoption d'une structure adaptée pour la surveillance est cruciale pour renforcer l'exploitation sûre de ces systèmes.

Ce chapitre a pour objectif de faire un état de l'art sur les méthodes et les approches utilisées dans le domaine de la surveillance distribuée.

1.1 Introduction

L'un des objectifs les plus importants de l'automatisation aujourd'hui concerne l'augmentation de la fiabilité, de la disponibilité donc de la sûreté de fonctionnement des processus technologiques. C'est la raison pour laquelle nous mettons en œuvre des systèmes de surveillance dont le but est d'être capable, à tout instant, de fournir l'état de fonctionnement des différents équipements constitutifs d'un processus technologique.

La surveillance n'améliore pas seulement les performances et la productivité d'un système complexe mais protège aussi les vies et les biens. Pour ces raisons, des approches de surveillance ont été largement étudiées dans la littérature. La plupart ont été développées pour des systèmes où l'information utilisée par la surveillance est centralisée ou hiérarchique. Or, la majorité des systèmes complexes (réseaux de communication, systèmes manufacturiers, systèmes de puissance, etc.) sont informationnellement distribués. Récemment, l'attention s'est portée sur la distribution de la surveillance de systèmes de natures diverses. L'architecture distribuée permet de résoudre les problèmes associés à ces architectures conventionnelles.

Comme notre travail se focalise sur la surveillance distribuée des systèmes et plus particulièrement sur la partie diagnostic distribué, un rappel de quelques définitions, un résumé des différentes approches, méthodes et architectures de la surveillance distribuée et diagnostic des défaillances dans un système industriel complexe sont présentés.

1.1 Notions fondamentales

Dans la littérature associée à notre domaine, plusieurs définitions parfois divergentes apparaissent. Les définitions proposées nous permettent d'exprimer notre point de vue sur la surveillance et le diagnostic de systèmes complexe. Ces définitions sont extraites de [Combacau 1991], [Zwingelstein 1995], [Basseville 1996], [Lefebvre 2000].

1) ***Système complexe :*** Un système complexe peut être défini comme un système mettant en œuvre différents systèmes de multiples fournisseurs qui interagissent entre eux dans un but commun et devant la plupart du temps, répondre à des critères de performances, de fiabilité et de sécurité prédéfinis [Murthy 2009]. Il est composé de multiples briques technologiques en interaction éventuelle avec des opérateurs ou des utilisateurs, ils prennent en compte de nombreuses informations pour réaliser une opération complexe de façon plus ou moins automatique. La complexité du système est généralement

transparente pour l'utilisateur final, mais ses défaillances peuvent avoir de lourdes conséquences humaines, sociales ou économiques.

2) ***composant industriel :*** Un composant industriel est un organe technologique qui forme une partie du processus industriel (réservoir, conduite, pompe ...).

3) ***Dégradation :*** Une dégradation représente une perte de performances d'une des fonctions assurées par un équipement.

4) ***Défaillance :*** Une défaillance est l'altération ou la cessation de l'aptitude d'un ensemble à accomplir sa ou ses fonctions requises avec les performances définies dans les spécifications techniques.

5) ***panne*****:** Une panne est l'inaptitude d'une entité (composant ou système) à assurer une fonction requise.

   Remarque : Si une défaillance, sur un plan temporel, correspond à une date, (instant t_2, figure 1.1), la panne représente une durée comprise entre la date d'occurrence de la défaillance et la date de fin de réparation.

6) ***Mode de fonctionnement*** : le mode de fonctionnement est l'un des états possibles d'un élément défaillant, pour une fonction exigée donnée. Un système présente généralement plusieurs modes de fonctionnement (figure 1.1) :

   - *Mode de fonctionnement nominal* est le mode où l'équipement ou le système industriel remplit sa mission dans les conditions de fonctionnement requises par le constructeur et avec les exigences attendues de l'exploitant.
   - *Mode de fonctionnement dégradé* correspond soit à l'accomplissement partiel de la mission, soit à l'accomplissement de celle-ci avec des performances moindre.
   - *Mode de défaillance* correspond à des mauvais fonctionnements du système, c'est-à-dire qu'il y a eu défaillance soit après dégradation soit défaillance brusque. Un mode de défaillance est caractérisé par les effets causés par cette défaillance. A chaque mode de défaillance, on associe une décision et une interprétation possible. Chaque équipement ou système peut posséder qu'un seul mode nominal, par contre, il possède plusieurs modes de défaillance.

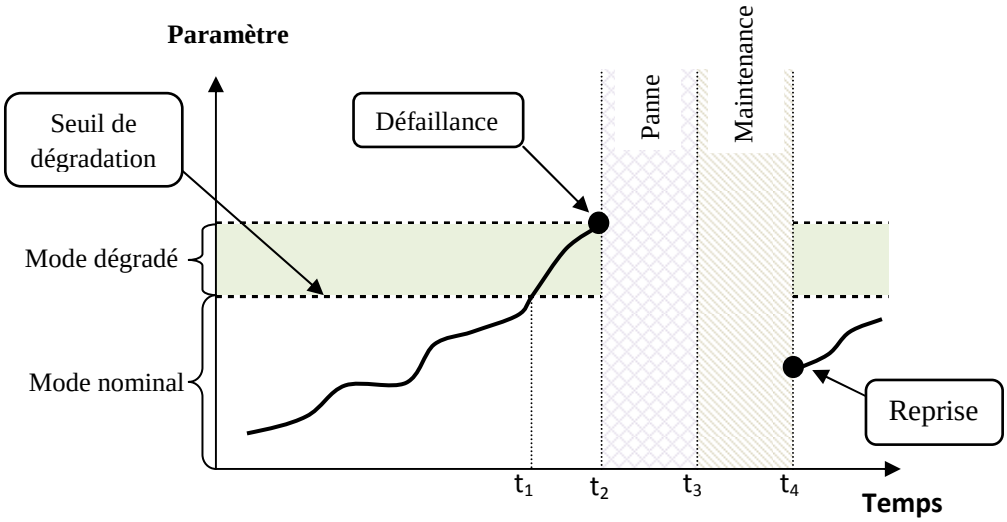


Figure 1.1 Modes de fonctionnement d'un système complexe.

7) ***maintenance*** c'est l'ensemble de toutes les actions techniques, administratives et de gestion, durant le cycle de vie d'un bien, destiné à le maintenir ou le rétablir dans un état dans lequel il peut accomplir une fonction requise.

8) ***surveillance*** c'est un dispositif passif, informationnel, qui analyse l'état du système et fournit des indicateurs [Lefebvre 2000]. La surveillance consiste notamment à détecter et classer les défaillances en observant l'évolution du système, puis à les diagnostiquer en localisant les éléments défaillants et en identifiant les causes premières (figure 1.2).

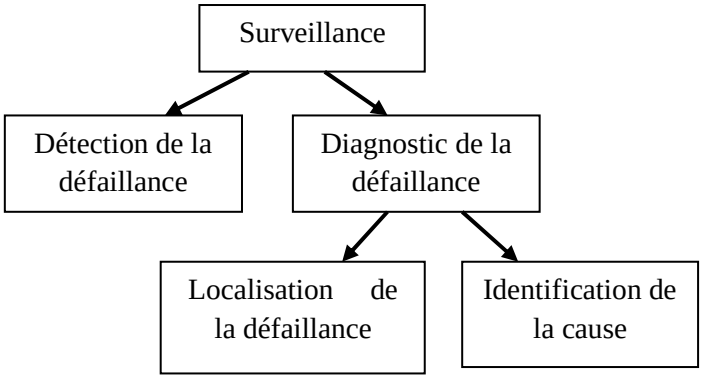


Figure 1.2 Architecture d'un système de surveillance.

Pour ***détecter*** les défaillances du système, il faut être capable de classer les situations observables comme étant normales ou anormales. Cette classification n'est pas triviale, étant donné le manque d'information qui caractérise généralement les situations anormales.

Une simplification communément adoptée consiste à considérer comme anormale toute situation qui n'est pas normale.

L'objectif du ***diagnostic*** est de rechercher les causes et de localiser les organes qui ont entraîné une observation particulière. Cette fonction se décompose en deux fonctions élémentaires :

- La ***localisation*** permet de déterminer le sous-ensemble fonctionnel défaillant.
- L'***identification*** consiste à déterminer les causes qui ont mené à une situation anormale. Ces causes peuvent être internes (sous-ensembles défaillants faisant partie de l'équipement), ou bien externes à l'équipement.

La section suivante sur les méthodes de la surveillance, ne présente pas un état de l'art exhaustif des méthodes de la surveillance existantes. Néanmoins, les techniques décrites sont les plus connues et les plus couramment utilisées. Le choix d'une méthode de surveillance dépend de la connaissance du système, de la présence de capteurs ou de modèles qui permettent de suivre l'état réel du système.

1.2 Méthodes de surveillance industrielle

La surveillance est la base d'une excellente sûreté de fonctionnement des processus technologiques. Elle constitue une interface entre l'opérateur et l'installation physique. Son rôle est de fournir les informations sur l'état de fonctionnement (correct ou erroné) des dispositifs surveillés ainsi que la validation des informations issues des capteurs et la localisation des composants défaillants.

La surveillance se réalise à l'aide d'algorithmes conçus pour traiter les données brutes (issues des automatismes et des opérateurs) et les données mesurées (issues des capteurs) dont l'objectif de produire des données dont l'efficacité a été prouvée par le système. Il s'agit de données valides. Nous présentons sur la figure 1.3, les techniques les plus courantes en surveillance d'équipements industriels.

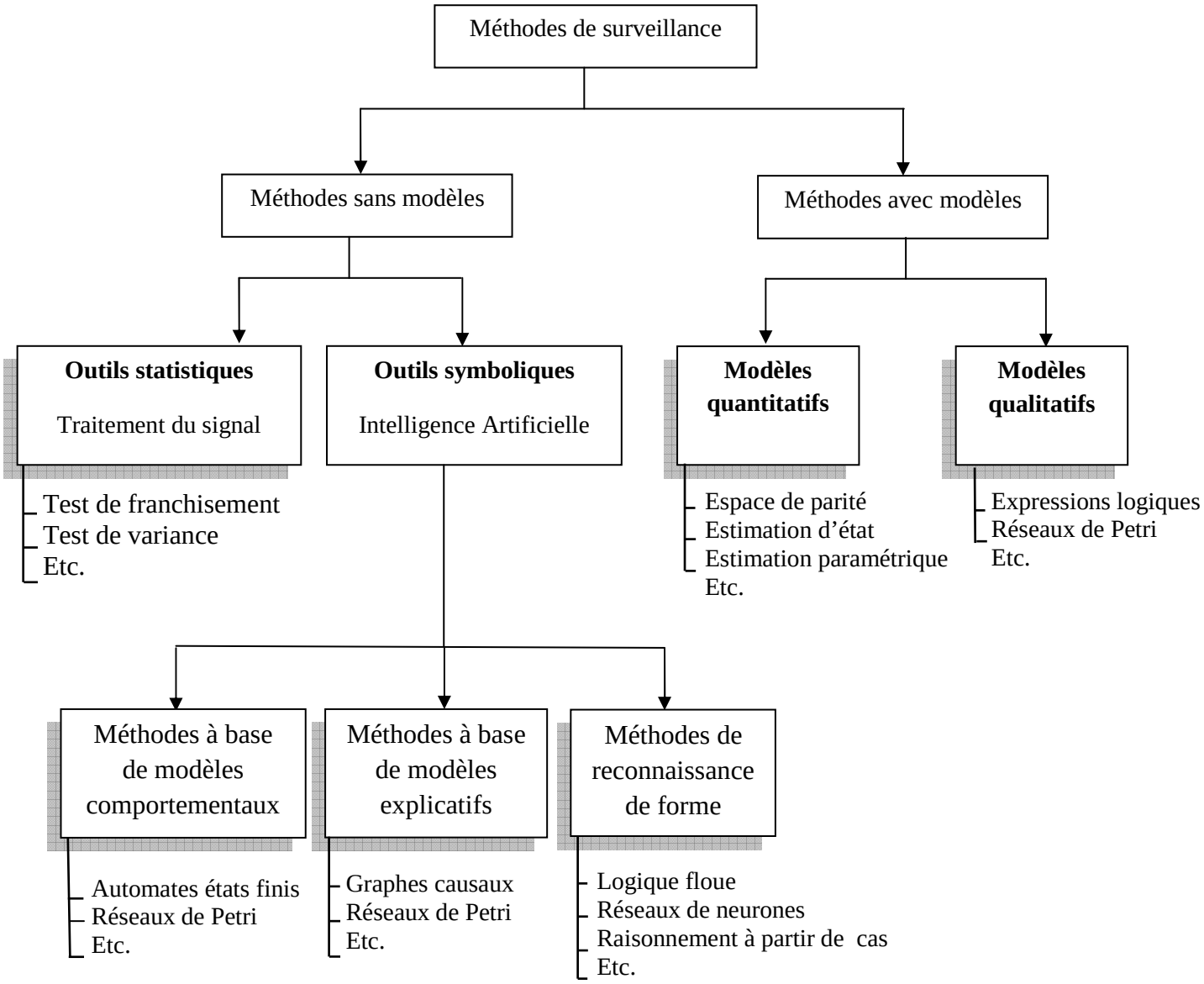


Figure 1.3 Classification des méthodes de surveillance industrielle.

L'existence d'un modèle formel ou mathématique de l'´equipement détermine la méthode de surveillance utilisée [Zemouri 2003]. La surveillance avec modèles se compose essentiellement de deux types : méthodes par modélisation quantitative, et les méthodes par modélisation qualitative. Mais, nombreuses sont les applications industrielles dont le modèle est difficile, voire impossible à obtenir suite à une complexité accrue. Pour ce type d'applications industrielles, les seules méthodes de surveillance opérationnelles sont celles sans modèles [Dubuisson 2001]. Ces méthodes se divisent en deux catégories: méthodes utilisant des outils statistiques et méthodes symboliques de l'Intelligence Artificielle.

1.2.1 Méthodes de surveillance sans modèles

1.2.1.1 Méthodes par outils statistiques

Les outils statistiques de détection de défaillances consistent à supposer que les signaux fournis par les capteurs possèdent certaines propriétés statistiques. On effectue alors quelques tests qui permettent de vérifier si ces propriétés sont présentes dans un échantillon des signaux mesurés. Parmi les tests les plus importants, nous mentionnons : le test de franchissement de seuils, le test de moyenne et le test de variance [Zemouri 2003].

1.2.1.2 Méthodes par outils symboliques

Ces méthodes s'appuient largement sur les techniques issues de l'IA et font appel à des connaissances symboliques, familières ou au moins partageables par l'opérateur [Basseville 1996]. L'utilisation de l'IA permet de pallier la complexité des systèmes à surveiller. En effet, elle peut se caractériser par la capacité de traiter : une grande quantité d'informations, des données non homogènes (numériques/symboliques), des données dépendant du contexte et de traiter des données incomplètes [Monnin 2004]. On distingue parmi les modèles symboliques :

- **Les méthodes à base de modèles comportementaux**: elles simulent le comportement du système, à partir d'une modélisation de son comportement. Elles regroupent notamment des outils tels que les réseaux de Petri et les automates d'états finis.
- **Les méthodes de reconnaissance de formes**: les mots clés qui caractérisent le mieux ces méthodes sont apprentissage/reconnaissance. On retrouve principalement des outils tels que les réseaux neuronaux, la logique floue, les réseaux neuro-flous, les systèmes expert et le raisonnement à partir de cas.
- **Les méthodes à base de modèles explicatifs**: elles fournissent une représentation de l'analyse causale des liens entre les défaillances, leurs causes et leurs effets observables. Les graphes d'influence, les graphes causaux, les graphes contextuels et la logique floue sont les outils les plus considérés.

1.2.2 Méthodes de surveillance à base de modèles

Ce travail s'intéresse particulièrement au diagnostic par modèles. Ainsi, cette section présente les approches de diagnostic à base de modèles. Ces approches reposent sur une connaissance physique profonde du système à diagnostiquer. Le système est représenté sous forme d'un ou plusieurs modèles qui décrivent la structure du système et son comportement nominal ou encore son comportement en présence de faute. La méthode de diagnostic s'appuie sur la comparaison du comportement réel observé sur le système physique avec le comportement prédit à l'aide de modèles (figure 1.4). La détection d'incohérences permet de conclure sur l'occurrence de faute dans le système. Un modèle de dysfonctionnement (modèle de faute) permet de localiser les fautes et éventuellement de les identifier.

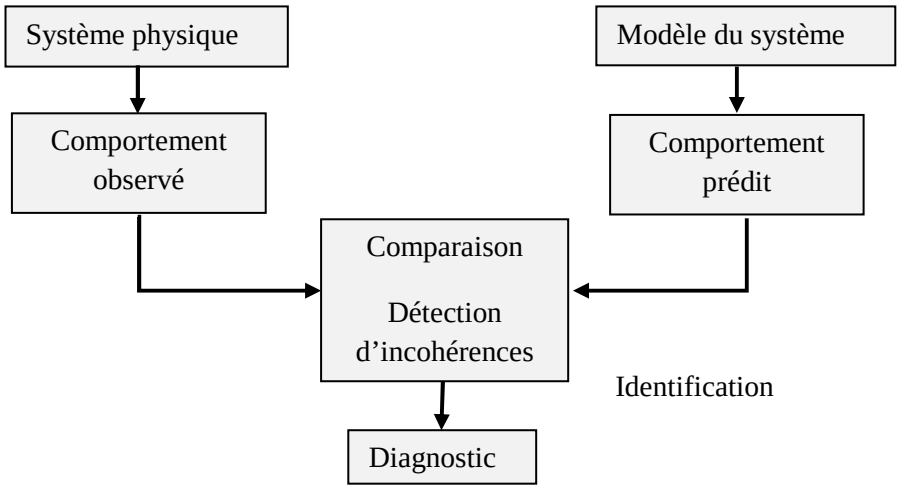


Figure 1.4 Diagnostic à base de modèles [Touaf 2005].

Deux principales approches peuvent se distinguer dans les méthodes de diagnostic à base de modèles. L'approche FDI issue de la communauté automatique utilise des modèles quantitatifs pour décrire le modèle de comportement du système. Cette approche a atteint un stade de réelle maturité et a produit de nombreux résultats [Patton 1991] [Frank 1996] [Iserman 1997]. L'approche DX, fondée sur une théorie logique du diagnostic, provient de la communauté de l'IA [Reiter 1987] [De Kleer 1987] [Hamscher 1992]. Elle utilise des modèles qualitatifs qui permettent de représenter de manière efficace les interactions entre composants ou systèmes.

1.2.2.1 L'approche FDI

L'approche FDI repose sur une connaissance approfondie du fonctionnement du système. Elle utilise des modèles de référence quantitatifs qui peuvent être obtenus à partir de lois fondamentales de la physique [Frank 1996].

Dans ce cadre, le diagnostic consiste à générer des indicateurs de présence de faute dans le système que l'on appelle des *résidus*. Un résidu correspond à une différence entre le comportement prédit par le modèle de référence et le comportement observé du système [Gertler 1998]. Le *modèle* de référence est un ensemble d'équations différentielles décrivant le comportement du système qui peut être linéaires ou non, à temps discret ou continu. Il peut également être un système hybride [Touaf 2005].

Une valeur non nulle d'un résidu est interprétée comme une modification anormale (ou une déviation inacceptable) d'une propriété ou d'un paramètre caractéristique du système modélisé. A cause des erreurs de modélisation et de la présence de bruit, les résidus ne sont jamais réellement nuls même s'il n'y a pas de faute dans le système. La décision d'une détection nécessite d'évaluer les expressions des résidus obtenus en utilisant les mesures du système afin de déterminer si la différence à laquelle ils sont associés est significative. Pour l'évaluation des résidus, des techniques de reconnaissance de formes, de logique floue, de seuillage sont utilisées [Basseville 1996]. Le principe d'une approche FDI basée sur le calcul des résidus est illustré sur la figure1.5.

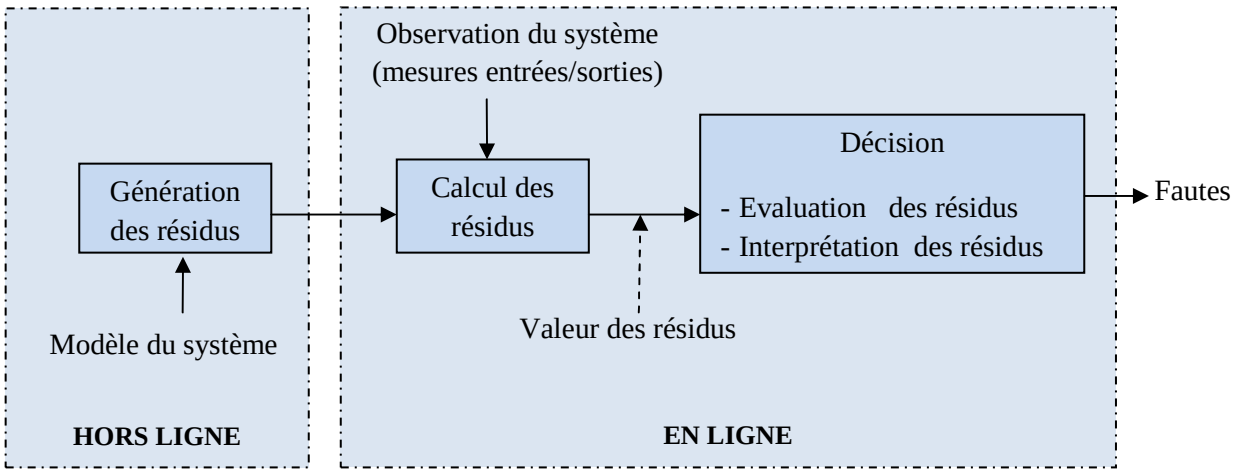


Figure 1.5 Approche FDI à base de résidus.

Pour obtenir les expressions analytiques des résidus, plusieurs techniques peuvent être utilisées [Iserman 1997] [Dubuisson 2001]. Parmi ces méthodes nous trouvons:

- **Espace de parité:** le principe de cette méthode est la vérification de la consistance existante entre les entrées et les sorties du système surveillé (figure 1.6).

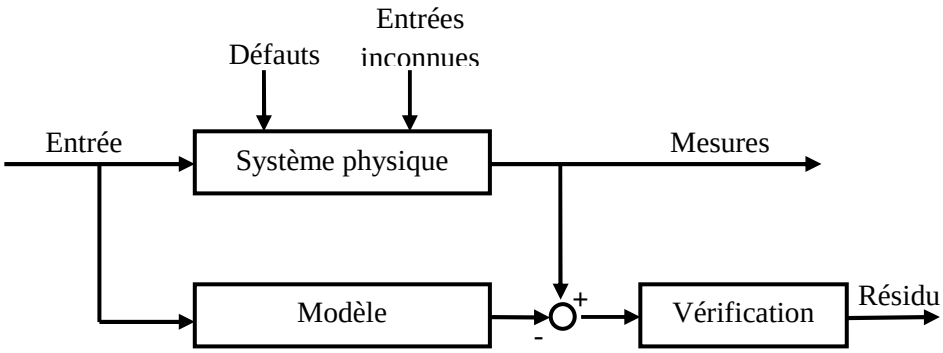


Figure 1.6 Approche de l'espace de parité dans un format entrée-sortie.

- **Estimation paramétrique:** elle consiste à estimer en continu des paramètres du procédé en utilisant les mesures d'entrée/sortie et en l'évaluation de la distance qui les sépare des valeurs de référence de l'état normal du procédé. La détection d'une faute se fait en comparant les paramètres estimés avec les paramètres nominaux qui caractérisent le comportement normal du système (figure 1.7).

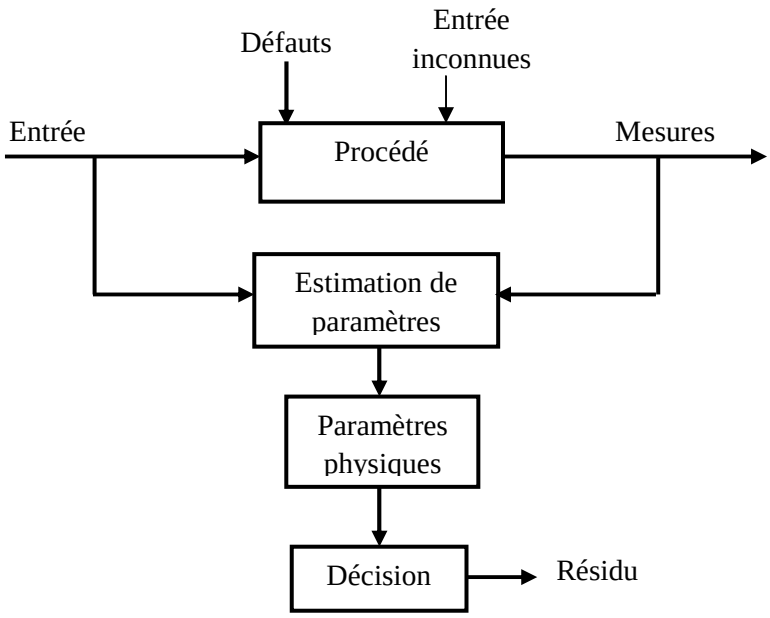


Figure 1.7 Estimation paramétrique pour la détection et le diagnostic de défauts.

- **Observateurs:** elles consistent à reconstruire à partir d'un modèle analytique et d'un ensemble d'observations partielles du système (entrées/sorties) les sorties non mesurables du système par une estimation de l'état du système. La présence d'un

résidu est évaluée en comparant les variables réelles et les variables estimées. Un vecteur de résidus est obtenu représentant la différence entre les mesures des sorties et les valeurs des sorties estimées.

Pour localiser les fautes, les résidus sont conçus pour que chacun soit sensible à un sous ensemble de fautes connues et insensible aux autres fautes [Issury 2011]. Lorsqu'une faute connue apparaît, la valeur de certains résidus est nulle ou proche de zéro tandis que d'autres résidus seront différents de zéro. L'ensemble des valeurs des différents résidus représente la signature de la faute. L'ensemble des signatures pour les différentes fautes connues pouvant apparaître dans le système est appelé la matrice de signatures (figure 1.8):

	f_1	f_2	...	f_f
r_1	0	1	...	0
r_2	1	1	...	1
⋮				
r_r	0	0	...	1

Figure 1.8 Matrice de signature d'une faute.

Dans l'exemple de matrice de signatures donné ci-dessus, le résidu associé à la relation de redondance analytique r_1 n'est pas affecté par la fautef_1, il est par contre affecté par la fautef_2. Le résidu associé à r_2 est quant à lui affecté par les deux fautes f_1 et f_2.

1.2.2.2 L'approche DX

L'approche DX est une approche qualitative basée sur la cohérence qui provient du domaine de l'IA. La technique du diagnostic de cohérence consiste à comparer le comportement réel du système observé et son comportement attendu tel qu'il peut être prédit grâce à des modèles de bon comportement. Elle repose sur la théorie logique du diagnostic introduite par [Reiter 1987] puis étendu et généralisé par [De Kleer 1987]. Les travaux les plus marquants sur ces approches de diagnostic à base de modèles sont regroupés dans [Hamscher 1992]. Le but du diagnostic logique est de déterminer les équipements (ou composants) du système dont le fonctionnement anormal peut expliquer les incohérences détectées entre les comportements prédits et les observations du système [De Kleer 1987]. Il s'agit d'un diagnostic abductif. Le principe de cette approche logique

basée sur la cohérence est synthétisé dans l'ouvrage [Dubuisson 2001] et est illustré sur la figure 1.9.

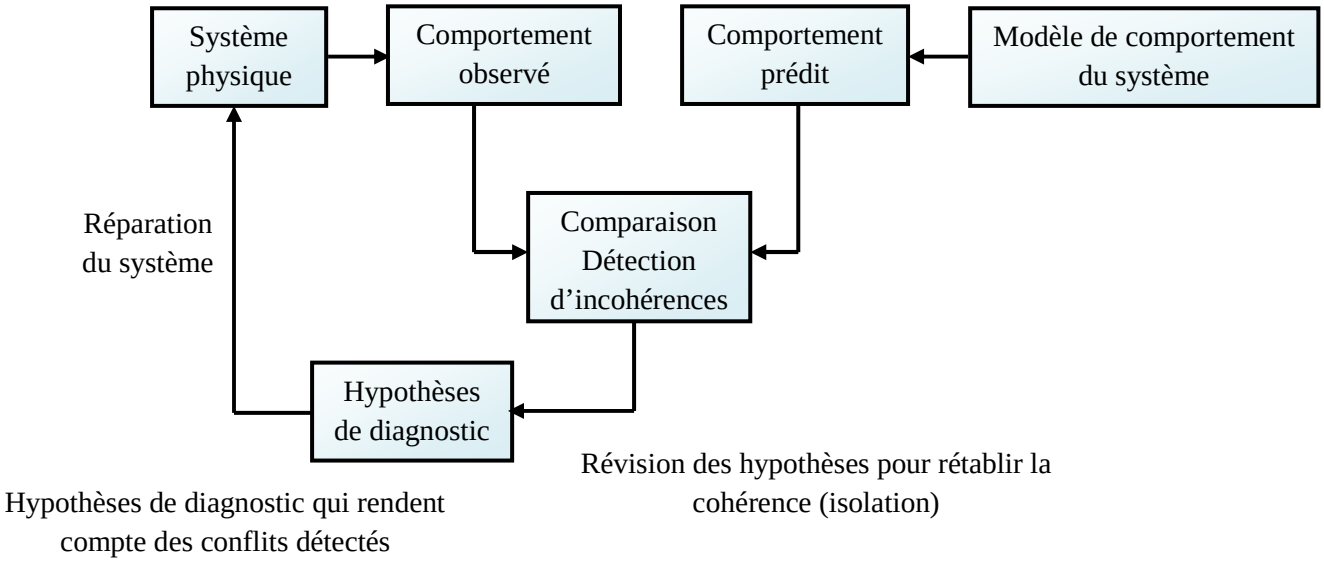


Figure 1.9 Approche logique basée sur la cohérence.

Le modèle utilisé par cette approche est qualitatif et décrit la structure du système et le comportement des composants du système. Le système est représenté par la paire$(SD, COMPS)$, dans la quelle :

- SD représente un ensemble de formules logiques du premier ordre qui décrivent la structure du système, c'est-à-dire l'ensemble des composants et leurs connexions, ainsi que le comportement des composants du système,
- $COMPS$ représente un ensemble fini de constantes qui représentent les composants du système.

Le modèle de comportement nominal du système est suffisant pour détecter des incohérences. Le prédicat Ab est utilisé pour indiquer qu'un composant ne se comporte pas correctement. La notation $Ab(C)$ signifie que le composant $C \in COMPS$ fonctionne anormalement, C est dit en faute. La notation $\neg Ab(C)$ veut donc dire au contraire que le composant C fonctionne correctement.

Un système observé est défini par le modèle$(SD, COMPS, OBS)$, où OBS est un ensemble de formules du premier ordre qui représentent les observations disponibles par les capteurs positionnés sur le système. C'est à partir de ces observations et du modèle de comportement nominal que des incohérences peuvent être détectées.

Le comportement d'un système est supposé anormal dès qu'une incohérence est détectée entre les observations et les prédictions faites à partir du modèle de comportement nominal du système. Les incohérences détectées ne se contentent pas de manifester la présence de fautes dans le système, mais renseignent également sur la localisation de ces dernières. Pour cela, il suffit d'utiliser les prédictions qui ont menées à ces incohérences. Si une prédiction a été faite en utilisant les modèles de comportement nominal des composants $\{c_1, \dots, c_n\} \in COMP$ et qu'elle entre en contradiction avec une observation, c'est donc que les composants $c_1, \dots, c_n$ ne peuvent être tous corrects et que l'un d'eux est nécessairement en faute. On dit que ces composants forment un conflit. [Reiter 1987]:

Un R-conflit pour un système $(SD, COMPS, OBS)$ *est un ensemble de composants* $\{c_1, \dots, c_k\} \subseteq COMP$ *tel que* $SD \cup OBS \cup \{\neg Ab(c_1), \dots, \neg Ab(c_k)\}$ *est incohérent. Un R-conflit minimal est un conflit n'incluant aucun autre R-conflit.*

Un conflit correspond donc à un ensemble de composants qui ne suivent plus le modèle de comportement nominal. La détection des conflits constitue la première phase du diagnostic à base de modèles.

La seconde phase du diagnostic consiste à générer des hypothèses sur les comportements des composants du système. Ces hypothèses rendent compte de tous les conflits, c'est-à-dire de toutes les incohérences détectées. Cela revient à changer l'hypothèse de fonctionnement correct de certains composants en une hypothèse de dysfonctionnement (de fonctionnement anormal), de manière à ce que toutes les contradictions disparaissent, i.e. qu'il n'y ait plus de conflit. Un ensemble de composants qui, cessant d'être supposés corrects, rétablit la cohérence avec les observations est précisément appelé un diagnostic. C'est le principe du diagnostic basé cohérence.

Le problème de diagnostic consiste à attribuer un mode de comportement (comportement normal ou anormal), représenté par Ab et ¬Ab, à chaque composant du système de manière à éliminer les conflits détectés. Un diagnostic est formellement défini de la manière suivante.

Un diagnostic pour un système observé $(SD, COMPS, OBS)$ *est un ensemble de composants* $\Delta \subseteq COMPS$ *tel que* $SD \cup OBS \cup \{\neg Ab(c), c \in COMPS \setminus \Delta\} \cup \{Ab(c), c \in \Delta\}$ *est cohérent.*

Pour un ensemble de composants donné, il ya généralement plusieurs diagnostics possibles. On ne s'intéresse en général qu'aux diagnostics minimaux. *Un diagnostic* Δ *est minimal s'il n'existe pas de diagnostic* Δ' *tel que* $\Delta' \subset \Delta$.

Reiter propose ainsi un algorithme « Diagnostic » pour déduire le diagnostic en utilisant les ensembles de conflits pour construire des arbres appelés hitting set (ensemble échantillon) [Reiter 1987]. Dans ces arbres, chaque chemin allant de la racine vers une feuille marquée par « √ » est un diagnostic.

Définition : considérons un ensemble de R-conflits minimaux C alors $H \subseteq COMPS$ est un ensemble échantillon si et seulement si $\forall c \in C, c \cap H \neq \emptyset$. Un ensemble échantillon est minimal s'il n'inclut aucun autre ensemble échantillon.

Définition : l'ensemble Δ est un diagnostic minimal pour le système observé $(DS, COMPS, OBS)$ si et seulement si Δ est un ensemble échantillon minimal de l'ensemble des R-conflits minimaux du système$(DS, COMPS, OBS)$.

Même si l'idée première du diagnostic à base de modèles est de se passer de connaissances sur les fautes et les dysfonctionnements du système, si de telles connaissances sont disponibles, elles peuvent aider à la localisation des fautes.

Au lieu de n'avoir que deux modes de comportement par composant correct et anormal, dont seul le premier est modélisé$(\neg Ab)$, plusieurs modes de dysfonctionnement correspondant aux fautes connues possibles seront modélisés [De Kleer 1989]. Pour des soucis de complétude, un mode inconnu dépourvu de tout modèle est toujours introduit et est censé regrouper tous les comportements de dysfonctionnement associés aux fautes non répertoriées. Un composant $c \in COMPS$ aura ainsi pour modes:$\{N(c), F1(c), \dots, Fm(c), I(c)$ avec N, le mode de fonctionnement normal (correct), $F1$,..., Fm, les modes de faute connus et I représentant le mode inconnu.

Un conflit devient dans ce cas une assignation de modes comportement aux certains composants qui est en contradiction avec les observations du système. Un diagnostic est alors une affectation de modes de comportement à tous les composants du système qui rétablit la cohérence avec les observations [De Kleer 1992]: $N(c_1) \wedge F1(c_2) \wedge \dots \wedge Fj(c_n)$.

1.2.2.3 Equivalence des deux approches: Bridge FDI-DX

Les communautés FDI et DX ont travaillé en parallèle sur le diagnostic à base de modèles durant de nombreuses années. Des travaux présentés dans [Cordier 2004] et

[Biswas 2004] permettent de comparer les deux approches et d'établir un pont entre les deux communautés.

Dans les deux approches FDI et DX, le diagnostic se base sur un modèle explicite du comportement normal du système. L'occurrence d'une faute est détectée à partir des incohérences entre le comportement observé et le comportement prédit à l'aide du modèle du système. L'isolation des fautes repose sur l'analyse des ensembles de composants impliqués dans chaque incohérence détectée.

La complexité croissante des processus industriels et leur mise en œuvre sur la base des technologies de l'information et de la communication ont pour conséquence essentielle la complexité de la tâche de surveillance basée sur les architectures conventionnelles. Les architectures distribuées de la surveillance étalées dans la section suivante, présentent des solutions adéquates aux problèmes posés par les autres architectures conventionnelles telle que la réduction de la complexité, la tolérance aux fautes, la réduction des coûts, l'amélioration des caractéristiques tout en maintenant la robustesse et la flexibilité, sont les avantages des architectures distribuées.

1.3 Architectures des systèmes de surveillance

Différents types d'architectures peuvent être considérées lors de la conception d'un système de la surveillance (figure 1.10). Nous présentons chaque architecture avec ses avantages et ses inconvénients [Zennir 2004].

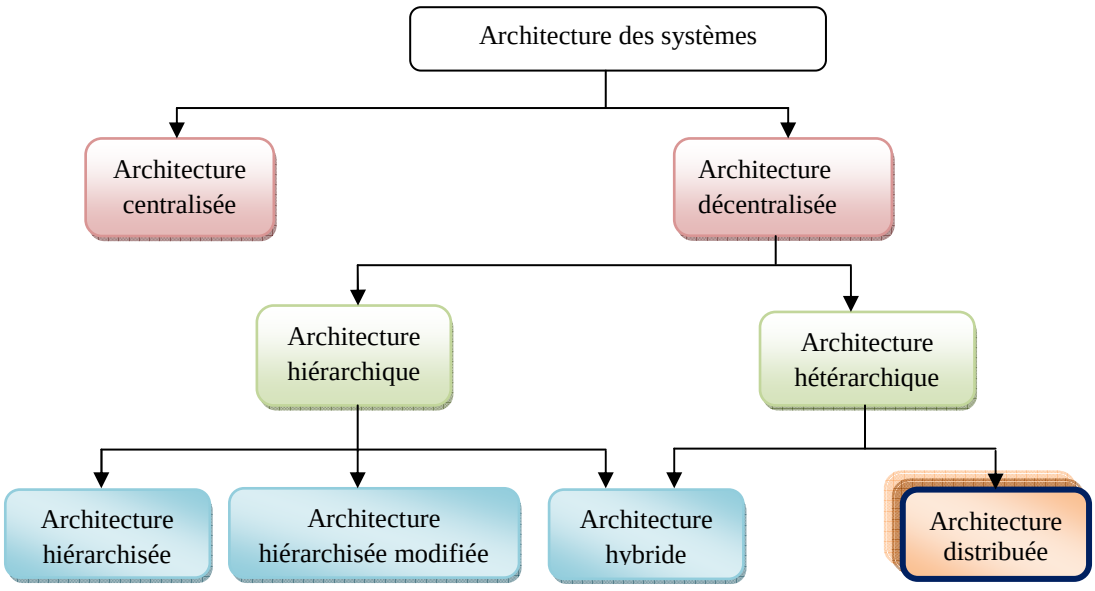


Figure 1.10 Différentes architectures des systèmes de la surveillance [Zennir 2004].

1.3.1 Architecture centralisée

Dans ce cas, une seule entité qui planifie, prend les décisions, pilote tous les mécanismes de coordination et maintient l'information globale sur l'ensemble des autres entités (figure 1.11). Cette structure présente certains avantages comme:

- l'entité centrale dispose d'une vue globale sur le système complet,
- les communications entre entités sont réduites,
- le nombre d'unités de contrôle, de moyens de traitement et de gestion de l'information est limité,
- possibilité d'optimiser de façon globale la gestion de l'information.

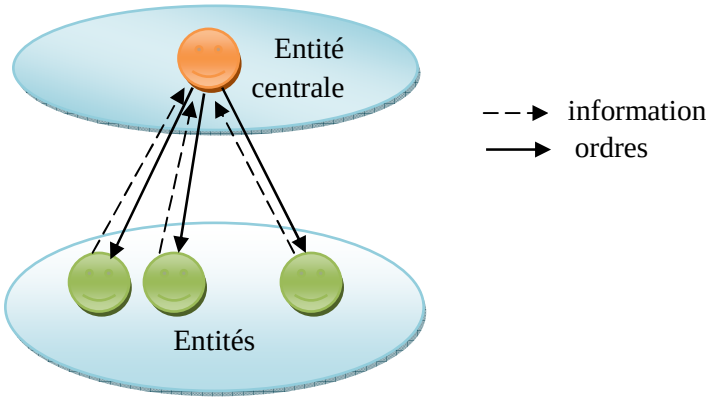


Figure 1.11 Architecture centralisée.

Néanmoins, cette architecture présente les inconvénients comme :

- le temps de réponse dépend de la dimension du système (i.e. lorsque le nombre d'entités augmente la vitesse des communications décroît),
- le système est peu robuste car il est sensible aux fautes de l'entité centrale,
- l'entité centrale doit disposer des informations globales à chaque instant, ce qui n'est pas toujours faisable,
- difficulté de faire évoluer le système à cause de non modularité.

Pour pallier ces inconvénients, les recherches se sont alors portées sur des architectures non centralisées.

1.3.2 Architecture décentralisée

Deux types d'architectures décentralisées sont distingués: hiérarchiques et hétérarchique.

1.3.2.1 Les architectures hiérarchiques

Elles sont inspirées des structures sociales. Ces architectures sont composées de plusieurs niveaux de sous-systèmes avec, au niveau supérieur, une centralisation locale comme le montre la figure 1.12 i.e. que chaque niveau dispose d'une entité centrale qui contrôle et coordonne les autres entités du même niveau. Les relations entre les entités d'un même niveau sont indépendantes des niveaux supérieurs. Cependant, il y a une relation de maître (niveau supérieur) à esclave (niveau inférieur) entre les niveaux.

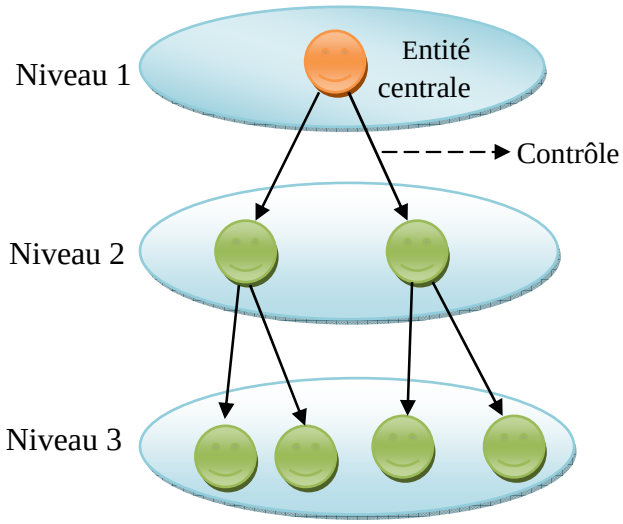


Figure 1.12 Architecture hiérarchique.

Il existe une autre forme d'architecture hiérarchique où les entités d'un même niveau peuvent se coordonner entre eux et communiquer. Elle est appelée architecture hiérarchique modifiée [Zennir 2004]. Parmi les avantages, de cette architecture décrite sur le schéma de la figure 1.13, les points suivants peuvent être relevés :

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

Les inconvénients sont dus aux aspects suivants [Reaidy 2003] :

- problèmes de transfert de partage des informations et de coordination entre les entités du même niveau,

- problème d'évolutivité : pour effectuer des modifications de structure, il faut refondre tout le système et mettre à jour les structures de plus haut niveau dans l'architecture,
- problème de perturbation imprévue, comme la panne d'une ressource qui rend la planification et l'ordonnancement pour le contrôleur de niveau élevé invalide,
- problème de robustesse dans le cas où le contrôleur central de haut niveau tombe en panne. Cette situation exige l'arrêt total du système.

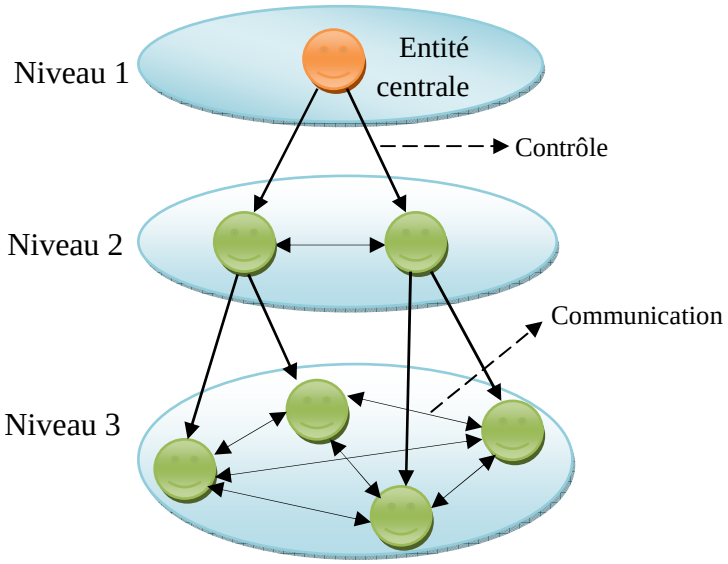


Figure 1.13 Architecture hiérarchique modifiée.

1.3.2.2 Les architectures hétérarchiques

Les architectures hétérarchiques sont une sorte d'opposition aux architectures hiérarchiques. En effet, il n'y a pas de relation maître/esclave. Les entités sont donc considérées au même niveau et ont tous les mêmes possibilités de participer à la prise de décision. Le terme hétérarchique est formé à partir de deux termes grecs : *heteros*(autre) et *Archhein*(commander), signifiant ainsi « commander par les autres » [Reaidy 2003]. L'architecture hétérarchique peut être une architecture décentralisée ou une architecture distribuée. Un exemple d'architecture distribuée est représenté sur la figure 1.14.

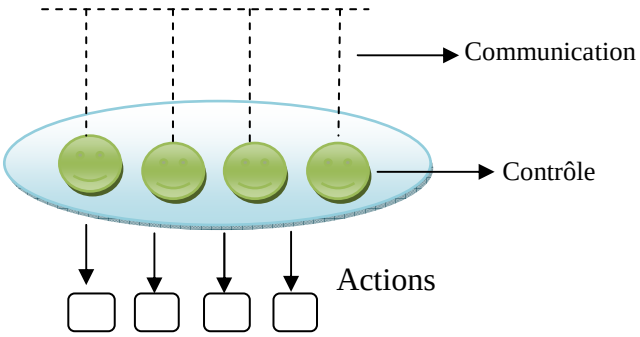


Figure 1.14 Architecture hétérarchique.

Les entités de cette architecture disposent des propriétés suivantes :

- elles ont le même droit d'accès aux ressources,
- la communication entre eux peut être fortement développée,
- les entités sont indépendantes pendant le fonctionnement,
- elles sont autonomes,
- pour atteindre l'objectif du système global, la collaboration et le travail collectif sont nécessaires.

Parmi les avantages de cette architecture nous pouvons citer :

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

Cette architecture présente aussi quelques inconvénients comme :

- la communication peut devenir très complexe,
- la coordination entre les entités est très importante et aussi complexe,
- la réalisation de l'objectif global est basé sur les objectifs locaux et est très difficile a assuré.
- les performances globales du système dépendent du choix des règles locales et des protocoles de négociation entre les entités.

1.3.3 Architecture hybride

Les deux structures hiérarchique et hétérarchique présentent des avantages et des inconvénients pour le diagnostic des SdP. Certains travaux de recherche ont essayé de préserver les avantages des deux structures en proposant une nouvelle structure hybride combinant les deux autres (figure 1.15).

Dans la structure hybride, les entités de contrôle de même niveau hiérarchique sont interconnectées via un même moyen de contrôle. Elles sont capables de communiquer et de coopérer pour satisfaire leurs objectifs locaux. Lors de perturbations, l'ensemble des entités de contrôle peut demander de l'aide à leur moyen de contrôle pour résoudre les problèmes détectés.

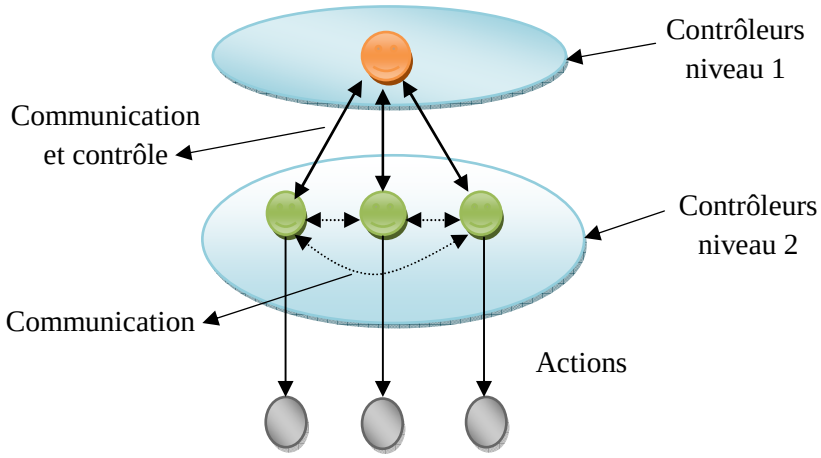


Figure 1.15 Architecture hybride.

Avantages et inconvénients :

- les avantages des deux structures (hiérarchique et hétérarchique) en même temps. Ainsi, elle combine la robustesse contre des perturbations, à travers les interactions locales entre les entités hétérarchiques, et l'optimisation globale et la prédiction à travers des entités hiérarchiques de contrôle.
- le problème est de trouver le bon compromis entre le contrôle au niveau hiérarchique et le degré d'autonomie attribué aux niveaux hétérarchiques. Le compromis cherché permet la stabilité et l'adaptation au changement dans un environnement complexe.

1.3.4 Architecture distribuée

L'architecture distribuée est choisi pour ce travail de recherche. Elle est envisagée particulièrement pour éviter les problèmes posés par l'architecture centralisée où un seul et unique organe est chargé d'établir la surveillance à partir d'une vue globale du système physique, ce qui pose évidemment des problèmes comme [Allem 2010]:

- *Problèmes liés à la complexité du système*: l'état du système est habituellement l'ensemble des états de ses composants. Pour évaluer l'état de ces systèmes, il est nécessaire d'en avoir une connaissance plus complète en engendrant une quantité d'informations qui pose des problèmes de charge de l'organe principal.
- *Problèmes liés à une architecture peu robuste*: un problème se produisant à l'intérieur d'un système centralisé de surveillance peut conduire à l'échec total du système. En revanche, dans un système distribué, la défaillance d'un module (par exemple de détection, ou de localisation) qui participe à la surveillance ne met pas en péril toute la procédure de la surveillance.
- *Problèmes liés à une architecture figée*: toute modification ou évolution structurelle du système nécessite une réécriture plus ou moins complète du programme de la surveillance.
- *Problèmes liés à la distribution des calculs*: en effet, toutes les tâches d'une procédure de surveillance sont exécutées sur un calculateur centralisé. Distribuer les différents calculs liés à la surveillance sur des machines différentes s'avère difficile dans une approche centralisée.

L'architecture distribuée permet de résoudre les problèmes complexes de la surveillance industrielle aux niveaux détection et diagnostic des pannes en les décomposant en une multitude de problèmes plus élémentaires en construisant des entités de surveillance communicantes.

Il existe dans la littérature beaucoup de travaux sur les architectures distribuées et ouvertes. Ces derniers ont subie une forte évolution principalement due à l'émergence des TIC intégrant les mécanismes éprouvés et utilisés dans le monde informatique tels que les nouveaux langages de programmation (JAVA, HTML, XML,…), les nouvelles technologies issues du monde des télécommunications telles que TCP/IP, WAP, Bluetooth, WiFi..., les plates formes pour le développement des système multi agents telle que Jade,

Swarm, MadKit … et les intergiciels (middleware) prenant en charge les fonctionnalités nécessaires à la communication entre éléments hétérogènes CORBA, RMI et les services web. Dans la section suivante nous présentons les travaux les plus répondus dans la littérature en motivant notre choix de technologies.

1.4 Les approches de la surveillance distribuée

Surveiller un système complexe dynamique qui peut être géographiquement distribué et de grande taille, entraîne la conception de plusieurs entités de surveillance autonomes communicantes.

Chacune d'elle collecte les informations relatives au fonctionnement du procédé à travers la réception d'une sous-séquence de la séquence d'événements générés par le procédé, traite ces données et communique éventuellement avec d'autres entités à travers l'envoi et la réception de messages d'événements. Elles déterminent l'état du procédé (normal ou défaillant) à partir de ses observations et de ses communications.

Les mécanismes utilisés pour la réalisation des fonctions de détection et de diagnostic se basent généralement sur des modèles de comportement local intégrant des protocoles de communication entre les différentes entités.

Il y a une distinction entre un système de surveillance sémantiquement distribué et un système de surveillance spatialement distribué [Froehlich 1996]. Les premiers référents à un groupe hétérogène de sous-systèmes, dans lequel chacun d'eux a sa propre vue du système. Ceci peut signifier que ces sous-systèmes sont dédiés à différents aspects du système ou encore utilisent différentes méthodes de surveillance. Les seconds référents à un groupe de sous-systèmes qui surveillent conjointement un système spatialement distribué avec des relations entre ses équipements ou ressources. Chaque sous-système a une information détaillée concernant une petite partie du système.

On s'intéresse à la surveillance des systèmes spatialement distribués en s'appuyant sur les agents et SMA qui offrent la capacité d'interaction (communication, coopérer, coordonner…) en vue de réaliser des tâches en commun. Comme on s'intéresse à la surveillance des systèmes sémantiquement distribués en s'appuyant sur les services web et SOA qui offrent une infrastructure bien définie et interopérable en assurant l'ouverture du système sur d'autres systèmes distants et l'échange de méthodes différentes.

Donc, dans ce travail ; nous nous intéressons à l'exploitation conjointe des technologies agent et service web. Les deux technologies assurant le développement d'applications de nature collaborative.

Les sections suivantes présentent un état de l'art non complet des travaux qui implémentent des architectures de surveillance et de diagnostic, exploitant en particulier les technologies agent et/ou service web.

1.4.1 Exploitation de la technologie agent

Plusieurs architectures de surveillance à base des SMA existent de nos jours:

- L'architecture proposée par [Leitão 2001] formée d'un ensemble d'agents autonomes, intelligents et coopératifs. Cette architecture a la particularité de présenter des mécanismes de surveillance intégrés dans tous les agents.
- [Fabre 2002] a conçu une architecture de surveillance composée de superviseurs locaux, un par équipement ou composant (figure 1.16). Ils se coordonnent pour définir une surveillance globale du système.

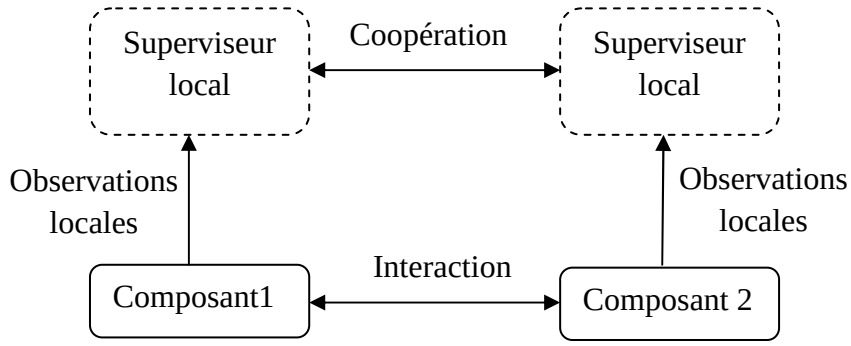


Figure 1.16 Un système distribué et une architecture de surveillance modulaire.

- La même approche a été utilisée pour la surveillance des réseaux de télécommunication composés de plusieurs sous-systèmes [Pencolé 1999]. Des systèmes de surveillance sont construits pour chacun de ces sous-systèmes. Chaque sous-système de surveillance effectue de manière locale la détection et le diagnostic. Toutes les informations de détection-diagnostic locales sont combinées, en utilisant l'historique des événements observés, afin de générer une détection et un diagnostic de niveau global.
- Dans l'architecture DIAMOND présentée sur la figure 1.17 mettant en œuvre le concept multi-agents, le système global de surveillance et de diagnostic M&D

(Monitoring & Diagnosis) est lui même composé de systèmes de M&D qui couvrent plusieurs systèmes de M&D plus petit. Le système global est donc découpé en plusieurs sous-ensembles, appelés domaines, surveillés par un système de M&D. Il possède son agent de surveillance qui génère des symptômes lorsque le domaine surveillé devient défaillant. Il possède également un agent facilitateur, un agent de résolution de conflits et un ou plusieurs agents de diagnostic. Ces derniers mettent en œuvre des techniques différentes pour diagnostiquer les composants du domaine surveillé. Les différentes méthodes implémentées par les agents de diagnostic permettent la détection des conflits dans les résultats produits. C'est l'agent de résolution de conflits qui va alors établir le diagnostic final. S'il ne parvient pas à produire une solution, il fait alors appel à l'agent facilitateur pour qu'il communique son problème à d'autres agents facilitateurs pour qu'ils le soumettent aux agents de résolution de conflits d'autres domaines de même niveau. Il est alors assez simple de définir une hiérarchie des différents agents implémentant l'architecture DIAMOND. Dans ce projet DIAMOND [Albert 2002], les agents de diagnostic inscrivent leurs diagnostics locaux sur un espace mémoire partagé nommé « tableau noir » (blackboard). Ces agents de diagnostic ne communiquent pas entre eux afin d'affiner leurs diagnostics locaux (gérer les conflits). C'est l'agent facilitateur qui est en charge de gérer les conflits.

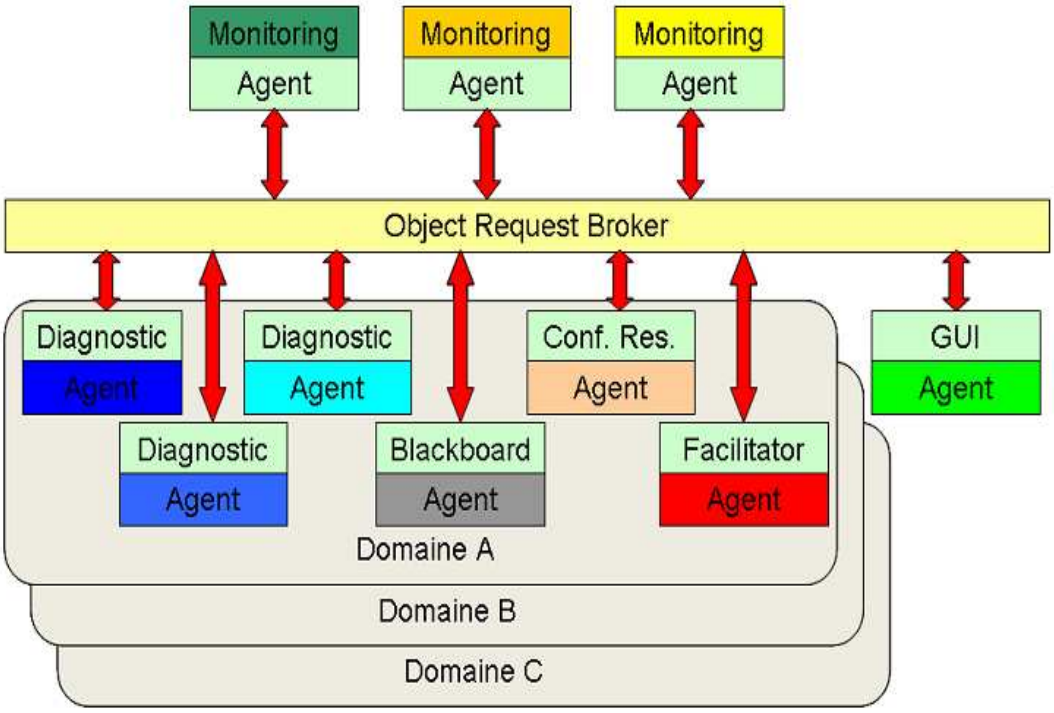


Figure 1.17 Architecture DIAMOND.

- L'architecture MAGIC [Touaf 2005] est composée de différents agents permettant d'aboutir au diagnostic d'un système. Chaque agent de diagnostic est alors totalement autonome et a la possibilité d'initier une communication avec les autres agents de diagnostic quand cela est nécessaire. La figure 1.18 représente les agents actifs en mode d'opération continue de cette architecture. Dans cette architecture, la couche de surveillance est la seule qui soit véritablement distribuée. Les D–Agents (Diagnostic Agents) implémentent des méthodes de génération de symptômes issues de domaines différents mais exploitant les mêmes mesures. Enfin, le DDA (Diagnostic Decision Agent) établit un état de santé global du système basé sur les symptômes générés et arbitrer ces diagnostics en fonction de la confiance qu'il accorde aux symptômes qui lui sont envoyés. Si un même symptôme est généré par tous les agents de surveillance, l'agent de diagnostic est sûr que la défaillance est présente. Par contre, si seulement un agent émet le symptôme et pas les autres, l'agent de diagnostic aura une confiance moindre quant au diagnostic qu'il va générer à partir de ce symptôme.

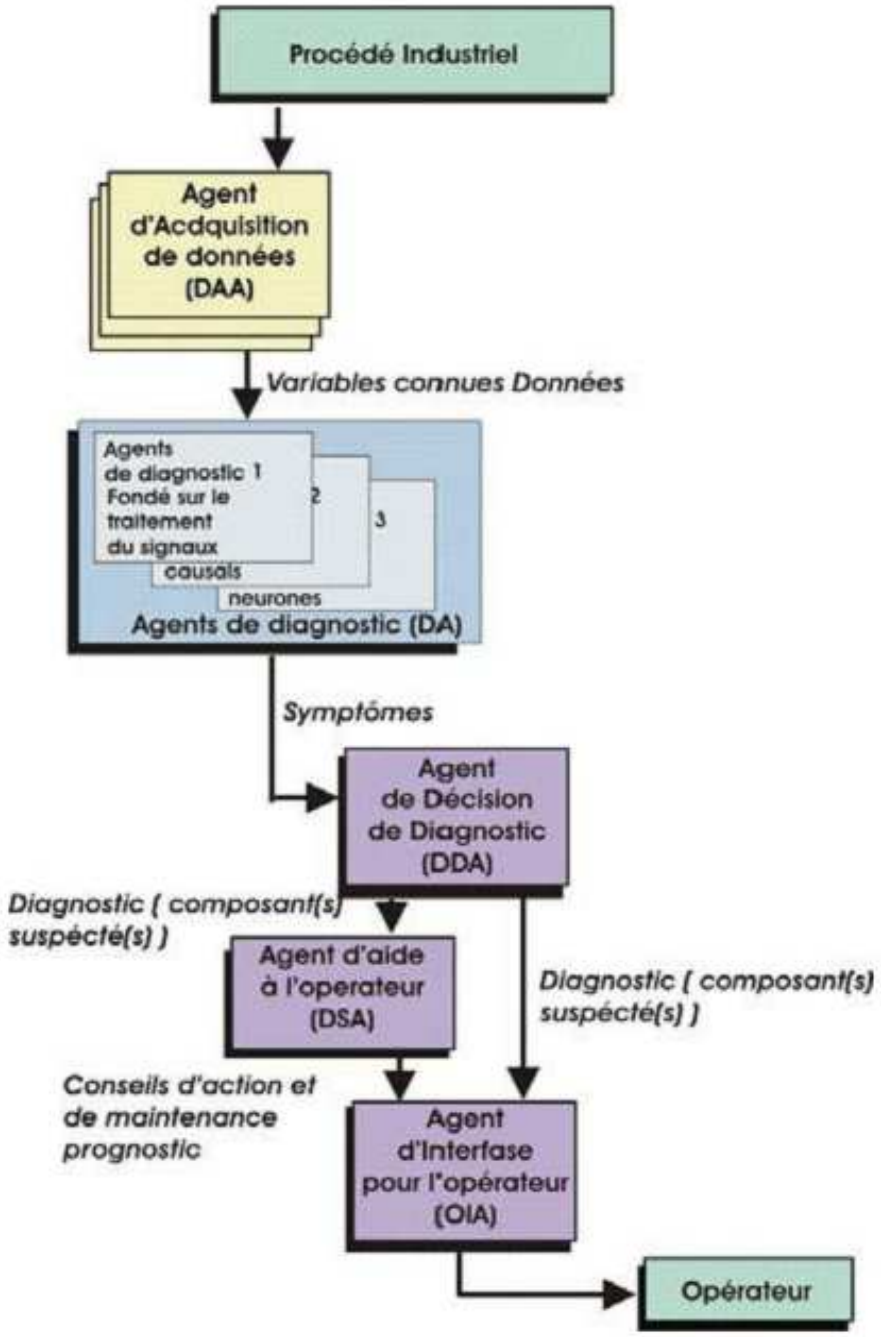


Figure 1.18 Agents actifs en mode d'opération continue de l'architecture MAGIC.

- Dans l'architecture WS-DIAMOND [Ardissono 2005], le système à diagnostiquer est découpé en plusieurs parties dont le diagnostic est établi par des agents de diagnostic locaux. Ces agents de diagnostic locaux n'ont aucune connaissance sur la globalité du système mais uniquement sur la partie qu'ils ont à diagnostiquer. Un superviseur de diagnostic détient quant à lui la connaissance des interactions entre chacune des parties du système. C'est à lui qu'incombe la tâche de définir un diagnostic global du système à

partir des diagnostics locaux. Cette technique est mise en œuvre par une architecture logicielle constituée exclusivement de services web communicant entre eux. Ces services web sont dotés, en plus d'une fonction de diagnostic, leur conférant une capacité à s'auto-diagnostiquer, d'une fonction leur permettant de se reconfigurer eux-mêmes. Le but recherché est d'assurer la fiabilité, la disponibilité et plus généralement, la qualité de service ou QoS (Quality of Services) des services web fournis.

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

La plupart de ces travaux sont des architectures décentralisées et non distribuées i.e. l'existence d'un agent superviseur est indispensable. On trouve aussi dans la littérature des travaux sur la surveillance à distance telle que [Pala 2014] [Resceanu 2008] [Shao 2013] [Tan 2011] [Wang 2007] et [Zhao 2009]. Cependant, la plupart de ceux-ci n'utilise pas les services web. La section suivante explique la pertinence de l'orientation service pour un système de la surveillance.

1.4.2 Pertinence de l'orientation service

Les conditions économiques actuelles du marché mondial poussent les entreprises à améliorer, d'une manière récurrente, leur compétitivité et à anticiper les évolutions par une stratégie d'envergure, à mobiliser leurs ressources humaines et techniques et à les adapter à l'évolution du contexte. En conséquence, le système de surveillance doit s'ouvrir à d'autres systèmes distants pour garantir par coopération la tâche de la surveillance. Pour se faire on doit trouver un moyen pour organiser et maitriser l'hétérogénéité de ces systèmes et

assurer leur interopérabilité multi-sites. Plusieurs définitions ont été proposées dans la littérature pour définir le terme « interopérabilité ». Pour notre travail; l'interopérabilité est la capacité de communiquer avec d'autres systèmes et d'accéder à leurs fonctionnalités [Vernadat 1996].

Alors, nous avons recours à l'architecture SOA mise en œuvre par la technologie services web. Elle est rapidement répandue et largement acceptée comme un style architectural pour simplifier le développement des logiciels et permettre l'interopérabilité entre les systèmes hétérogènes et distribués sur l'Internet. Des travaux intéressant sont présentés dans [Lee 2014] [El-Sharkawi 2013] [Rebeuf 2004] [Arpaia 2010] [Boukadi 2009] et [Ishak 2010].

Le principe d'une architecture orientée services pour notre travail consiste à structurer les fonctionnalités élémentaires contenues dans les applications de surveillance en un ensemble de services interopérables, standardisés qui exposent leur interface fonctionnelle et qui communiquent par messages. Ainsi, les avantages de l'orientation service pour le système de surveillance :

- L'approche service permet au système de surveillance d'acquérir une certaine flexibilité. En effet, dans la SOA les services interopérables, standardisés peuvent êtres rapidement combinés et réutilisés pour la construction de nouvelles fonctionnalités en réutilisant des services déjà existants. Cette rapidité de reconstruction des applications et des processus dote le système de surveillance, d'une réactivité considérable, qui permet en retour d'accélérer le temps de réponse de système de production surveillé vis-à-vis des changements de son environnement.
- L'approche service permet l'ouverture des systèmes de surveillance et la gestion de l'hétérogénéité de ces systèmes. Récemment, les services Web ont émergé pour proposer des solutions d'intégration des applications industrielles. Les services web constituent la technologie la plus importante pour mener une démarche SOA dans le domaine industriel. En effet, les systèmes de surveillance encapsulent les techniques automatiques par exemple, les techniques de détection et de localisation comme des services logiciels pour les rendre visibles sur Internet. Les services web sont accessibles à l'intérieur et à l'extérieur d'un système de surveillance. Par conséquent, ils permettent au système de surveillance d'intégrer ses applications hétérogènes ainsi

que ceux des centres de surveillance partenaires indépendamment des environnements techniques et des langages sur lesquels tournent ces applications.

- L'approche service permet les coopérations inter centres de surveillance. Le concept de services web offre aux systèmes de surveillance la possibilité d'échanger des services qui peuvent être automatisés au sein de leurs processus. Cette spécificité présente un atout considérable, favorisant ainsi la mise en place des coopérations inter systèmes de surveillance tout en assurant l'autonomie et la confidentialité des centres de surveillance.

1.4.3 Intégration d'agents et de services web

L'architecture orientée-services et les SMA sont deux technologies de plus en plus importantes pour la construction de systèmes logiciels. Les objectifs de ces deux architectures partagent quelques similitudes. A titre d'exemple, la création des systèmes distribués et flexibles composés d'entités faiblement liées qui interagissent les unes avec les autres. Malgré les similitudes, il existe des différences majeures dans leurs technologies. En effet, les services possèdent des standards pour la description des interfaces ainsi que des protocoles qui sont totalement différents des langages de communications des agents. Ainsi, ils ne peuvent pas interagir les uns avec les autres directement. Plusieurs travaux ont étudié ce problème et ont montré que l'intégration des agents et des services web produit des avantages intéressants [Buhler 2002], [Maamar 2003] [Starr 1996] [Matskin 2005] [Foukarakis 2007] [Ishak 2010], [Cheaib 2010]. Cependant, nous n'avons trouvé aucun travail dans le domaine de la surveillance industrielle. Ainsi, notre objectif est d'intégrer les agents logiciels et les services web, en une entité cohérente qui tente de dépasser la faiblesse de chaque technologie, tout en renforçant leurs avantages individuels. En effet, les services ont une infrastructure bien définie et interopérable, alors que les agents fournissent des capacités sociales et de l'intelligence pour les applications. En conséquence, l'intégration des agents et des services web améliorent l'adaptabilité, l'interopérabilité et l'ouverture du système.

1.5 Conclusion

Après avoir introduit les notions de base du domaine et les stratégies les plus importantes de la surveillance, nous avons présenté dans ce chapitre un aperçu général des différentes architectures de surveillance- diagnostic. Nous avons montré que les

architectures distribuées présentent des solutions adéquates aux problèmes posés par les autres architectures conventionnelles. La réduction de la complexité, la tolérance aux fautes, la réduction des coûts, l'amélioration des caractéristiques tout en maintenant la robustesse et la flexibilité, sont les avantages des architectures distribuées. Pour réaliser une telle architecture nous avons proposé de la concevoir en utilisant le paradigme multi-agent. Cependant, ce modèle manque des mécanismes de localisation et de mise en relation des différents sites ainsi que des mécanismes de gestion des hétérogénéités existantes entre les méthodes de diagnostic utilisés dans ces différents sites. Le modèle SOA apparaît comme une solution permettant de gérer cette hétérogénéité.

L'avènement de ces techniques a eu un impact significatif sur les architectures actuellement développées. Elles permettent des déploiements autres que centralisés des fonctions de la surveillance mais aussi de fiabiliser et d'augmenter la capacité de communication en termes de flux de données. En ce sens, les chapitres suivants sont consacré à la présentation de différentes techniques et mécanismes associés à la technologie agent, système multi-agent, à la technologie service web, architecture orientée service et la présentation de notre contribution dans ce contexte.

La technologie à base d'agent exposé en deuxième chapitre apporte un ensemble de moyens nécessaire pour réaliser des systèmes de surveillance distribuée, la contribution principale de ce travail de recherche. La surveillance distribuée à base d'agent est détaillé au troisième chapitre et l'application informatique qui la supporte au quatrième chapitre.

Ainsi, nos travaux de recherche ont eu pour objectif principal de développer une architecture interopérable qui assure la coopération interentreprises en se basant sur la technologie service web. Notre contribution dans ce domaine fait l'objet de cinquième chapitre.

Chapitre 2

Modes d’interaction dans les Systèmes Multi Agents

Les Systèmes Multi-Agents ont été développés dans le cadre de l'intelligence artificielle distribuée. L'intérêt qu'ils suscitent est lié à leur capacité d'aborder les problèmes complexes d'une manière distribuée et de proposer des solutions réactives et robustes.

Nous allons aborder dans ce chapitre les modes d’interactions et les concepts clés relatifs au domaine des SMA, qui nous paraissent les plus appropriés pour notre travail.

2.1 Introduction

L'émergence de l'intelligence artificielle distribuée (IAD) est associée au constat qu'une approche centralisée ne peut recouvrir la complexité de nombreux problèmes rencontrés en intelligence artificielle [Ferber 1995]. Ce constat a conduit à distribuer les capacités cognitives décisionnelles sur des processus de résolution en interaction. Les SMA constituent à cet égard un sous domaine de l'IAD fondé sur le concept d'agent possédant un certain nombre de propriétés particulières.

Les SMA possèdent les avantages traditionnels de la résolution distribuée et concurrente de problèmes comme la modularité, la vitesse (avec le parallélisme), et la fiabilité. Ils héritent aussi des bénéfices envisageables de l'IA comme le raisonnement symbolique (au niveau des connaissances), la facilité de maintenance, la réutilisation et la portabilité, mais surtout ils ont l'avantage de faire intervenir des schémas d'interaction sophistiqués.

Ce chapitre traite précisément un élément essentiel et fondateur des SMA: l'interaction. Son objectif réside dans la mise en relation dynamique de deux ou plusieurs agents par le biais d'une ou plusieurs actions réciproques et permet aux agents de participer à la réalisation d'un but global et d'éviter autant que possible les conflits. Elle peut se faire selon plusieurs manières suivant la situation à travers laquelle interagissent nos agents. Donc, après avoir présenté les définitions d'un agent et d'un SMA, nous allons présenter les différentes modes d'interaction entre ces agents.

2.2 Qu'est-ce qu'un agent ?

Dans la littérature, nous trouvons une multitude de définitions du concept agent. Elles présentent certaines similitudes et dépendent du type d'application pour laquelle est conçu l'agent. La plupart des travaux font référence à la définition fournie par Ferber [Ferber 1995] qui défini qu'un agent est une entité physique ou virtuelle qui:

- est capable d'agir dans un environnement (figure 2.1),
- peut communiquer directement avec d'autres agents,
- est mu par un ensemble de tendance (sous la forme d'objectifs individuels ou d'une fonction de satisfaction, voire de survie, qu'elle cherche à optimiser),
- possède des ressources propres,

- est capable de percevoir (mais de manière limitée) son environnement,
- ne dispose que d'une représentation partielle de cet environnement (et éventuellement aucune),
- possède des compétences et offre des services,
- peut éventuellement se « reproduire »,
- a un comportement qui tend à satisfaire ses objectifs, en tenant compte des ressources et des compétences dont elle dispose, et en fonction de sa perception, de ses représentations et des communications qu'elle reçoit.

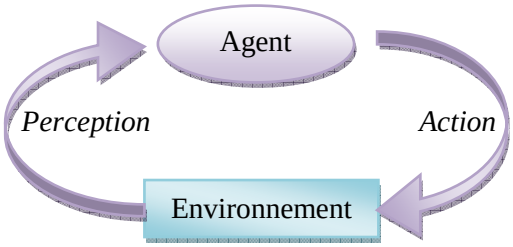


Figure 2.1 Relation Agent-Environnement.

D'autres définitions [Jennings 1998] mettent en évidence d'autres propriétés clés comme:

- les aptitudes sociales: elles désignent la capacité d'un agent à interagir avec les autres agents de façon coopérative ou compétitive pour atteindre ses objectifs.
- la pro-activité: elle désigne l'aptitude d'un agent à se fixer des buts pour atteindre ses objectifs sur sa propre initiative.

2.3 Qu'est-ce qu'un Système Multi-Agent ?

Un SMA est un ensemble d'agents en interaction afin de réaliser leurs buts ou d'accomplir leurs tâches [Wooldridge 1995]. La figure 2.2 présente l'aspect « social » interne au système, les interactions entre les agents qui composent le comportement global du système. Elles peuvent être directes par l'intermédiaire des communications, comme elles peuvent être indirectes via l'action et la perception de l'environnement. Les interactions peuvent être mises en œuvre dans un but de :

- coopération entre les agents, lorsqu'ils ont des buts communs,
- coordination, i.e. d'organisation pour éviter les conflits et tirer le maximum de profit de leurs interactions afin de réaliser leurs buts,
- compétition, lorsque les agents ont des buts concurrents.

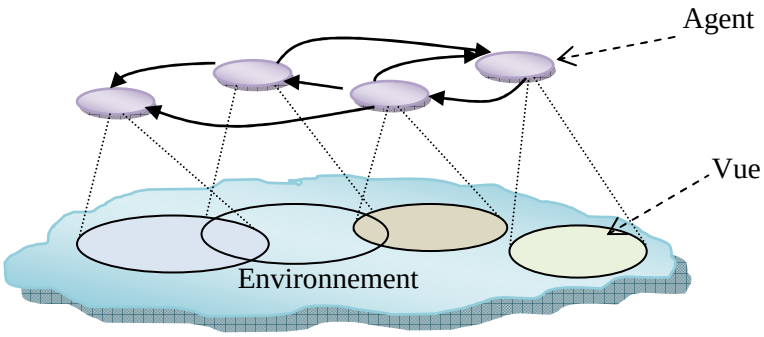


Figure 2.2 La vue sociale d'un SMA [Ferber 1995].

2.4 Différents modèles d'agents

Nous présentons dans ce qui suit, les principaux modèles d'agents classés par famille. Chaque famille se focalise sur une ou plusieurs caractéristiques que possède l'agent. Ainsi, nous distinguons :

2.4.1 Modèle d'agents réactifs

La structure des agents purement réactifs tend à la simplicité. Ce sont des agents qui fonctionnent selon un mode stimuli/réponse. Dès qu'ils perçoivent une modification de leur environnement, ils répondent par une action programmée. L'agent réactif ne possède pas une représentation complète de son environnement et n'est pas capable de tenir compte de ses actions passées. De ce fait, il ne peut avoir des capacités d'apprentissage. Ainsi, ils sont souvent considérés comme n'étant pas « intelligents » par leur nature. Mais ils peuvent être capables d'actions évoluées et coordonnées. C'est le cas d'une société de fourmis, étudiée par Alexis Drogoul [Drogoul 1999].

2.4.2 Modèle d'agents cognitifs

C'est le premier modèle d'agents qui a été proposé. Il est nommé aussi agent délibératif. Il se base sur un modèle qui provient d'une métaphore du modèle humain [Liu 2002]. Les agents cognitifs sont capables à eux seuls de réaliser des opérations relativement complexes. Généralement, ils coopèrent les uns avec les autres pour atteindre un but commun (résolution d'un problème, une tâche complexe, etc.). Ils possèdent un ensemble de représentations explicites (sur l'environnement, sur les autres agents et sur eux-mêmes) décrits dans une base de connaissances sur laquelle ils peuvent raisonner. Ils réagissent en fonction de leurs connaissances, leurs buts, de leurs échanges d'informations avec les autres agents et de la perception de l'environnement (figure 2.3). Ils sont dotés de

moyens et mécanismes de communication pour gérer les interactions avec d'autres agents (coopération, coordination et négociation). Le travail le plus représentatif de cette famille d'agent porte sur le modèle BDI (Believe Desire Intention) [Bratman 1988].

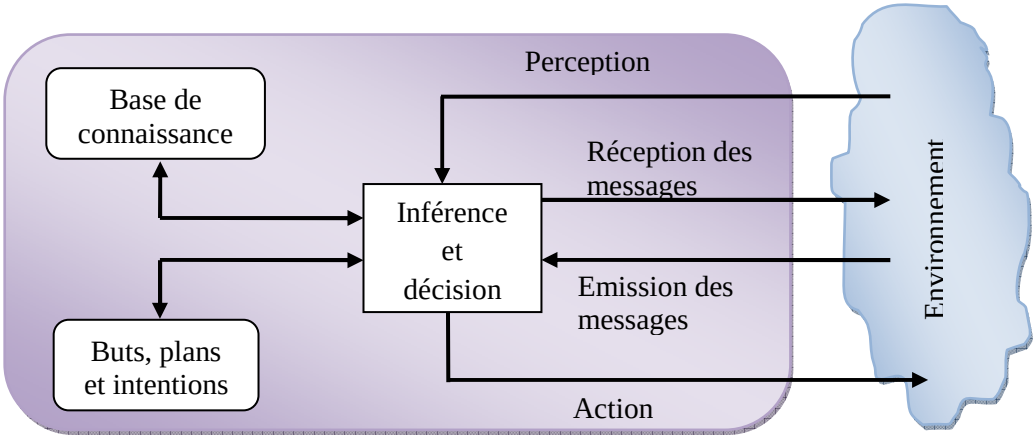


Figure 2.3 Modèle d'un agent cognitif.

A l'opposé des architectures réactives qui ont l'avantage de la simplicité, les agents cognitifs**,** sont plus intelligents et plus complexes. Le tableau 2.1 résume les différences entre les agents cognitifs et les agents réactifs.

Tableau 2.1 Différences entre agents cognitifs et agents réactifs.

Agent réactif	**Agent cognitif**
Fonctionnement Stimulus/Réponse	Agent complexe
Pas de représentation explicite	Représentation explicite de l'environnement
Pas de mémoire de son historique	Peut tenir compte de son passé
Grand nombre d'agents	Petit nombre d'agents

2.4.3 Modèle d'agents hybrides

Ce sont des agents ayant à la fois des capacités cognitives et réactives. Ils conjuguent la rapidité de réponse des agents réactifs ainsi que les capacités de raisonnement des agents cognitifs. Ce qui leur permet d'adapter leur comportement en temps réel à l'évolution de l'environnement. Dans ce type de modèle, les agents sont conçus comme étant composés de niveaux hiérarchiques qui interagissent entre eux. Chaque niveau gère un aspect du comportement de l'agent [O'Hare 1996].

2.5 Modes d'interaction dans les Systèmes Multi-Agents

Dans un SMA, les agents interagissent en vue de réaliser des tâches ou d'atteindre des buts. L'interaction a lieu, d'habitude dans un environnement commun où les agents ont diverses zones d'influence (figure 2.2). Ces zones peuvent être disjointes mais, dans la plupart des cas, elles se superposent. En interagissant dans un environnement partagé, les agents doivent coordonner leurs actions et avoir des mécanismes pour la résolution des conflits. L'interaction peut se faire selon plusieurs manières suivant la situation à travers laquelle interagissent les agents. Dans la littérature, quatre types d'interaction entre les agents sont distinguées : la communication, la coopération, la coordination et la négociation.

2.5.1 La communication

Les communications, dans les SMA, sont à la base des interactions et de l'organisation. Si les agents peuvent coopérer, coordonner leurs actions, réaliser des tâches en commun, c'est parce qu'ils communiquent. Les communications peuvent être sélectives sur un nombre restreint d'agents ou diffusées à l'ensemble des agents.

Il existe deux formes de communication, la communication par envoi de message et celle par partage d'information. Ces deux modèles ont été combinés pour donner naissance à un modèle hybride. Le principe est de concevoir des systèmes d'agents à base de tableau noir qui communiquent par le mode d'envoi de message [Baujard 1992].

2.5.1.1 La communication indirecte

Elle est adoptée par les agents réactifs. Elle se fait à travers l'environnement ou bien par le biais d'un tableau noir (figure 2.4). Dans une communication par environnement, les agents laissent des traces ou des signaux qui seront perçus par les autres agents. Dans ce genre de communication il n'y a pas de destinataire bien défini.

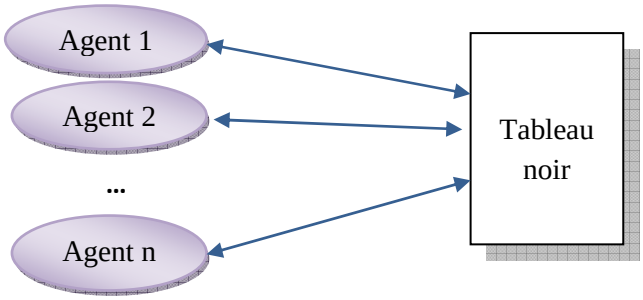


Figure 2.4 Communication par partage d'information.

2.5.1.2 La communication directe

Elle est spécifique aux agents cognitifs. Elle est intentionnelle et se fait par envoi de messages à un ou plusieurs destinataires (figure 2.5). La transmission se fait suivant deux modes :

- Transmission point à point : l'agent émetteur connaît et précise l'adresse de ou des agent(s) destinataire(s).
- Transmission par diffusion : le message est envoyé à tous les agents du système.

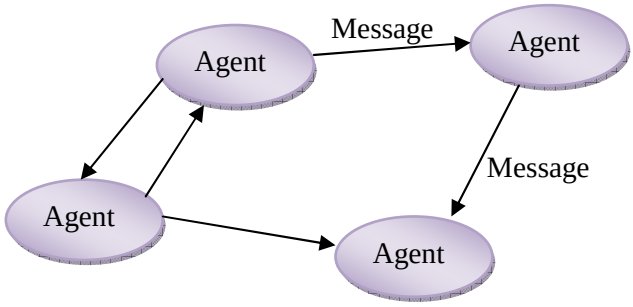


Figure 2.5 Communication par envoi de message.

Les systèmes utilisant la communication par envoi de message sont caractérisés par une distribution totale des connaissances, des résultats partiels et des méthodes utilisées pour aboutir à un résultat. Un agent ne peut donc manipuler que sa base de connaissances locales. Il peut cependant envoyer des messages aux autres agents qu'il connaît et sont appelés ses accointances. Ceci se fait d'une manière asynchrone et ce message peut être une requête, une information ou une suggestion, etc...

La communication directe se base sur trois éléments essentiels figure 2.6:

- *les langages de communication*: Ils permettent de structurer les messages échangés entre les agents. Les plus utilisés et connus sont KQML [Finin 1995] et FIPA ACL [FIPA 1999].
- *l'ontologie* : constitue une spécification ou une vue simplifiée et abstraite du domaine qui sera représenté. Elle sert à fournir un vocabulaire et une terminologie compréhensible par tous les agents. Cette sémantique sera régie par des règles et des contraintes qui permettront de définir un consensus sur le sens des termes.
- *les mécanismes de communication* : permettent de stocker, rechercher et adresser les messages aux agents. Ces mécanismes sont présents dans les plates-formes multi-agents comme JADE, JACK, MadKit, etc.

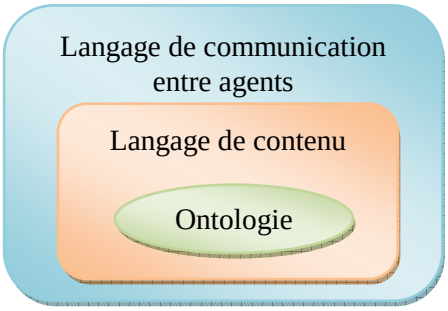


Figure 2.6 Modèle des langages de communication entre agents.

Dans la section suivante nous présentons le langage de communication FIPA ACL qui est intégré dans la plateforme JADE et utilisé dans ce travail.

2.5.1.3 Langage de communication: FIPA ACL

FIPA-ACL (FIPA Agent Communication Language) est un langage de communication et un protocole de haut niveau pour l'échange d'information proposé par l'organisation FIPA. Il est orienté messages et indépendant de la syntaxe du contenu et de l'ontologie du contenu des messages, indépendant aussi du mécanisme de transport (TCP/IP, e-mail etc.) et du langage utilisé pour coder le contenu des messages (Prolog, SQL etc.). Chaque message comprend trois couches :

- *La couche de contenu* qui spécifie le contenu réel du message d'agent ;
- *La couche de communication* qui décrit tous les paramètres de communication de bas niveau, par exemple, l'identificateur de l'agent émetteur et celui de l'agent récepteur, l'identificateur de la communication, etc.
- *La couche de message* : qui identifie le protocole de réseau, utilisé pour envoyer le message et de déterminer le performative, indiquant le type de ce message (par ex. une affirmation, une requête, une commande, etc.).

Il possède 21 actes communicatifs, exprimés par des performatives. Les actes communicatifs peuvent être primitifs ou composés. Les actes communicatifs primitifs sont définis de façon atomique, i.e. qu'ils ne sont pas définis à partir d'autres actes. En revanche, les actes communicatifs composés sont définis à partir d'autres actes. Les performatifs, peuvent être groupés selon leurs fonctionnalités de la façon suivante :

- passage d'information : *Inform, Inform-if, Inform-ref, Confirm, Disconfirm,*
- réquisition d'information : *Query-if, Query-ref, Subscribe,*
- négociation : *Accept-proposal, Cfp, Propose, Reject-proposal,*

- distribution de tâches (ou exécution d'une action): *Request, Request-when, Requestwhenever, Agree, Cancel, Refuse,*
- manipulation des erreurs : *Failure, Not-understood.*

Un message *FIPA-ACL* peut contenir une partie ou la totalité des éléments décrits dans le tableau 2.2. Les éléments nécessaires pour la transmission d'un message changent selon la situation. Si un agent ne reconnaît pas ou ne peut pas traiter un ou plusieurs éléments, alors il peut répondre avec le message *Not-understood*.

Tableau 2.2 Les éléments d'un message FIPA-ACL.

Elément	**Signification**
Performatif	le type de l'acte communicatif
Sender	l'émetteur du message
Receiver	le destinataire du message
reply-to	le participant à l'acte de communication
Content	le contenu du message (l'information transportée par le performatif)
Language	le langage dans lequel le contenu est représenté
Encoding	décrit le mode d'encodage du contenu du message
Ontology	le nom de l'ontologie utilisé pour donner un sens aux termes utilisés dans le contenu
Protocol	le nom du protocole d'interaction
conversation-id	l'identifiant de la conversation
reply-with	un identifiant du message, en vue d'une référence ultérieure
in-reply-to	il référence le message auquel l'agent est entrain de répondre (précisé par l'attribut reply-with dans le précédent message de l'émetteur)
reply-by	Un délai pour répondre au message

La figure 2.7 présente un exemple de message FIPA-ACL envoyé par l'agent A à l'agent B. L'agent *A* informe son interlocuteur que le prix d'un ordinateur *HP* est de 1500 euro. Le contenu du message est exprimé avec le langage *Prolog*. L'ontologie utilisée est celle des ordinateurs. Ce message fait partie d'une conversation ayant comme identifiant *conv01*. L'agent *B* est contraint par une durée de 10 minutes pour donner suite à ce message.

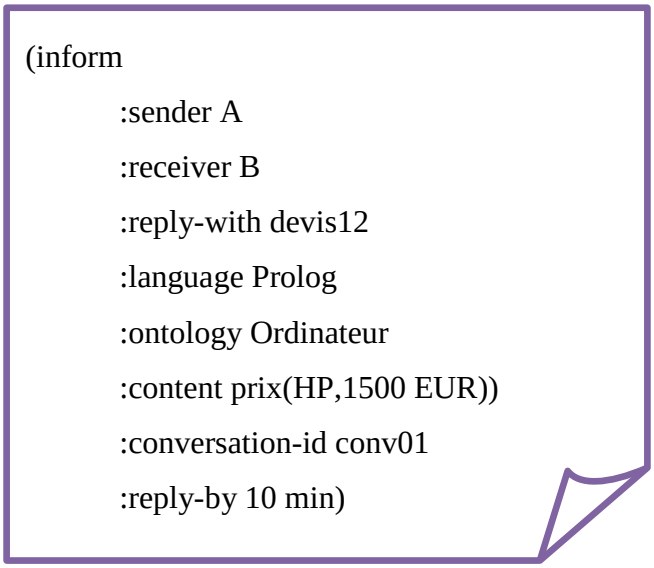


Figure 2.7 La structure d'un message FIPA ACL.

2.5.2 La coopération

La coopération se traduit par le fait qu'un ensemble d'agents travaillent ensemble pour satisfaire un but commun ou individuel. L'ajout ou la suppression d'un agent influe considérablement sur la performance du groupe. Le besoin de faire coopérer des agents, vient essentiellement du fait qu'un agent ne peut atteindre son objectif individuellement et a, par conséquence besoin de l'aide des autres agents du système.

La coopération peut être vue comme la détermination de « qui fait quoi », « quand », « où », « avec quel moyen », « de quelle manière » et « avec qui ». L'objectif de la coopération est d'améliorer le mode de travail des agents en termes d'augmentation:

- du taux de l'achèvement de la tâche à travers le parallélisme.
- de l'ensemble ou étendue des tâches réalisables en partageant des ressources
- de la probabilité de compléter des tâches en entreprenant d'autres tâches.
- et la diminution de l'intervention entre tâches en évitant des interactions malfaisantes.

Selon la structure de l'organisation pour les sociétés d'agents à savoir horizontale et verticale, on trouve deux manières de coopération entre les agents dans la première structure et trois modèles dans la deuxième structure. [Touaf 2005]

2.5.2.1 Coopération des agents dans l'organisation horizontale

Les agents de cette société sont au même niveau. Il n'y a pas d'agent maître qui joue le rôle du superviseur et des agents esclaves qui jouent le rôle des agents exécutants qui

traitent des sous tâches. Tous les agents peuvent communiquer entre eux (figure 2.8). Il existe deux modèles de coopération dans l'organisation horizontale:

- ***Coopération par partage de tâches*** : Le problème est distribué entre les différents agents. Les agents travaillent indépendamment les uns des autres. Chaque agent dispose de ressources et de compétences nécessaires pour accomplir la tâche qui lui a été assignée. Le contrôle est dirigé par les buts, et les agents sont représentés par les tâches qu'ils se sont engagés à exécuter.
- ***Coopération par partage des résultats :*** Les agents ne peuvent accomplir leurs tâches de manière indépendante. Ils sont appelés à se transmettre des résultats partiels. Le contrôle est dirigé par les buts, et les agents sont représentés par des sources de connaissances, la problématique réside dans la communication des résultats partiels.

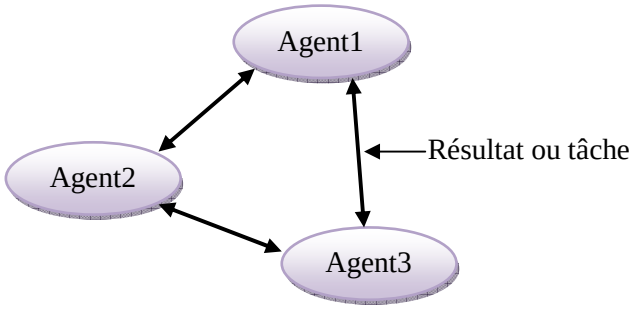


Figure 2.8 Coopération par partage de tâche ou de résultat.

2.5.2.2 Coopération des agents dans l'organisation verticale

A l'inverse de la structure précédente les agents sont structurés en plusieurs niveaux hiérarchiques. Dans une telle structure, l'agent reçoit le problème à résoudre, qui le décompose en sous problèmes auxquels il peut répondre localement, résoudre en coopérant avec les autres agents du même niveau que lui ou il les suivre aux agents du niveau inférieur dans la hiérarchie. On trouve trois modèles de coopération dans le cas ou il existe une hiérarchie entre les agents (figure 2.9):

- **Le mode *command*:** Dans ce mode, un agent superviseur décompose un problème en sous problèmes qu'il repartit entre les agents. Ceux-ci les résolvent et renvoient les solutions possibles à l'agent superviseur.
- **Le mode *appel d'offre*:** Dans ce mode, un agent superviseur décompose un problème en sous problèmes, dont il diffuse la liste aux agents. Chaque agent qui le souhaite

envoie une offre. L'agent superviseur choisi parmi celles-ci et distribue les sous problèmes. Le système fonctionne ensuite en mode commande.

- **Le mode *compétition*:** Il fonctionne comme dans le mode appel d'offre. Chaque agent résout un ou plusieurs sous problèmes et envoie les résultats correspondants à l'agent superviseur qui à son tour fait le tri.

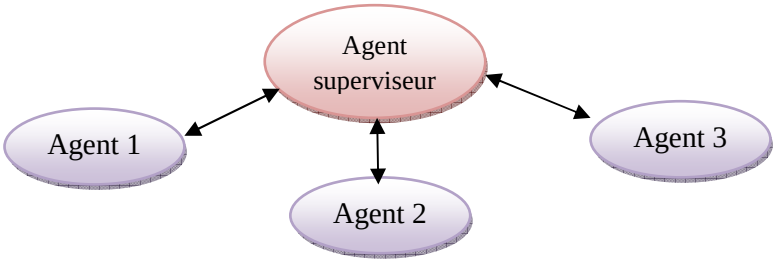


Figure 2.9 Coopération par commande, appel d'offres ou compétition.

2.5.3 La coordination

La coordination [Ciancarini 1996] est présente lorsqu'il existe une interdépendance dans les actions des agents, leurs buts ou même les ressources qu'ils utilisent afin d'éviter des problèmes dans le comportement global du système. En effet, la coordination met de l'ordre dans le processus global effectué par des agents. Elle consiste à synchroniser leurs activités ou à régler les conflits qui existent entre eux. Cette section présente les principales techniques de coordination dans les SMA notamment :

- ***la coordination par allocation de tâches:*** Cette technique consiste à répartir entre plusieurs agents une liste de sous-tâches résultant de la décomposition d'une tâche initiale.
- ***la coordination basée sur des structures organisationnelles:*** Cette technique s'attache à coordonner des agents à travers une structure organisationnelle qui offre un espace d'interaction et fixe des règles. La plus part des techniques organisationnelles (par exemple Aalaadin [Ferber 1998] et Gaia [Wooldridge 2000]) sont articulées autour les rôles intervenants dans l'organisation, les protocoles selon lesquels ces rôles interagissent et les mécanismes d'attribution de rôles aux agents.
- ***la coordination basée sur des protocoles d'interaction:*** Cette technique s'attache à coordonner les agents en définissant à priori la structure de leurs conversations. Ainsi, une conversation est considérée comme une occurrence d'un protocole d'interaction

(par exemple, les protocoles d'enchères, le «ContractNet» ou certains protocoles de négociation).

- ***la coordination par planification:*** L'objectif de la planification multi-agent est de coordonner les plans d'action de plusieurs agents afin d'éviter les conflits. En effet, un plan est un ensemble partiellement ordonné d'actions et d'interactions ayant un but, et pouvant être décomposé en plans partiels. Ces derniers peuvent également être fusionnés en un plan global après résolution de possibles conflits entre ces plans partiels et d'éventuelles optimisations.

2.5.4 La négociation

Dans le cas des SMA intelligents, la négociation est une composante de base de l'interaction surtout parce que les agents sont autonomes [Jennings 2000] et peuvent entrer en conflit (obtention de solutions différentes pour un même problème ou l'accès aux ressources). En cas de conflits les agents vont entrer dans un dialogue appelé *négociation* pour arriver ensemble à un compromis [David 1988], les auteurs affirment que: « *Par négociation, on entend une discussion dans laquelle des individus intéressés échangent des informations et arrivent à un accord en commun.* ». Selon cette définition, on peut identifier deux aspects essentiels de la négociation: la communication et la prise de décisions. Ainsi pour modéliser la négociation dans un logiciel multi-agent, il faut alors prendre en compte :

- Le langage de communication pour échanger les informations pendant la négociation.
- Le protocole de négociation pour structurer la négociation entre agents, le protocole réseau contractuel [Smith 1980] est le plus couramment utilisé.
- Le processus de décision utilisé par l'agent pour prendre des décisions pendant la négociation.

Le nombre de participants à la négociation peuvent aussi varier:

- ***Négociation un-à-un*****:** un agent négocie avec un autre, par exemple dans le cas où on essaie de négocier le prix d'achat d'une maison avec le représentant d'une agence immobilière.
- ***Négociation un-à-plusieurs*****:** un seul agent négocie avec plusieurs autres agents, par exemple les enchères où un agent veut vendre un objet

- ***Négociation plusieurs-à-plusieurs*:** plusieurs agents négocient avec plusieurs autres agents en même temps, par exemple, les participants à des enchères électroniques.

2.5.5 Protocoles d'interaction: réseau contractuel

Ce protocole a été un des premiers protocoles utilisés dans les SMA pour résoudre le problème d'allocation des tâches. Ce protocole est intéressant pour ce travail par ce qu'il est intégré dans JADE, la plateforme que nous avons choisi pour implémenter notre SMA.

Il est basé sur une métaphore organisationnelle, i.e. les agents coordonnent leurs tâches grâce à l'élaboration de contrats pour réaliser des objectifs spécifiques. Il comporte deux rôles essentiels: le gestionnaire et le contractant. Le premier décompose une activité en plusieurs tâches élémentaires et annonce chacune de ces tâches aux autres agents (contractants). Le second, quant à lui, évalue l'annonce par rapport aux ressources et compétences dont il dispose et envoi une proposition au gestionnaire. Ce dernier rassemble toutes les propositions qu'il a reçues, alloue la tâche à l'agent qui a fait la meilleure proposition et signe le contrat (figure 2.10).

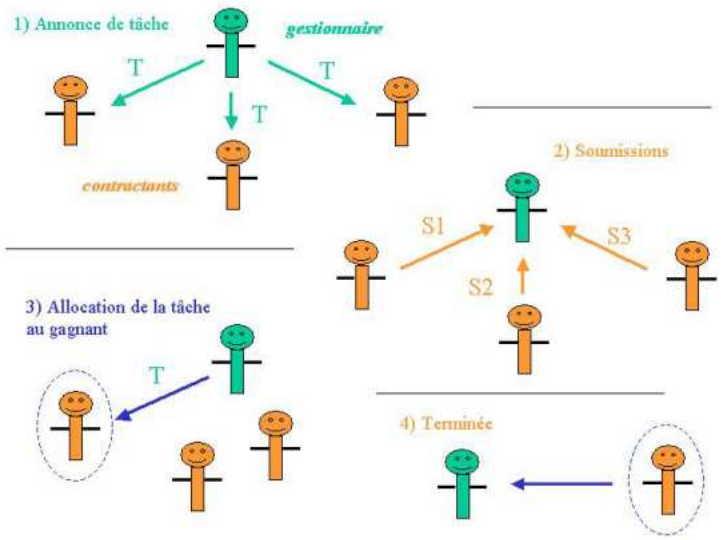


Figure 2.10 Etapes du protocole réseau contractuel [Ishak 2010].

Les rôles des agents ne sont pas spécifiés à l'avance. N'importe quel agent peut agir comme un gestionnaire et lancer un appel d'offres, n'importe quel autre agent peut être un contractant pour une annonce qu'on a faite. Cette flexibilité permet de nouvelles décompositions des tâches: un contractant pour une tâche spécifique peut agir comme un

gestionnaire en décomposant sa tâche et en annonçant les sous-tâches à d'autres agents. Les liens entre les gestionnaires et les contractants pour diverses tâches et sous-tâches forment un réseau contractuel hiérarchique qui permet la division des tâches et la synthèse des résultats.

2.6 Plateformes de développement de SMA

Les environnements de développement ou les plateformes multi-agents sont nécessaires pour renforcer le succès de la technologie multi-agents. Ces plates-formes proposent des fonctions de base pour la création des agents et la gestion des interactions entre ceux-ci. En conséquence, elles éliminent, dans la plupart des cas, la nécessité d'être familier avec les différents concepts théoriques des SMA et permettent aux développeurs de concevoir et de réaliser leurs applications facilement. Les plateformes les plus connues sont: Swarm, AgentBuilder, MadKit et JADE. Cependant, celles-ci ne proposent pas une solution facilitant l'utilisation des protocoles d'interaction, à l'exception de JADE. Dans notre travail nous allons analyser le problème en termes d'interactions entre les agents donc on va utiliser JADE. A notre connaissance elle est la seule plate-forme multi-agents qui propose une bibliothèque de protocoles d'interaction.

2.6.1 La plate-forme JADE

JADE (Java Agent Development Framework)[Caire 2003] est la plate-forme multi-agents choisi pour ce travail. Elle est implémentée entièrement en Java. Elle simplifie l'implémentation d'un SMA à travers un middleware répondant aux spécifications de FIPA. Une librairie de classes que les utilisateurs peuvent utiliser et étendre ainsi qu'un ensemble d'outils graphiques à savoir le « Sniffer Agent » ou encore le « Remote Monitoring Agent » permettent le débogage et l'administration du SMA à concevoir. JADE inclue tous les composants obligatoires qui contrôlent un SMA. Ces composants sont :

- Le DF "Director Facilitor" est un module qui fournit un service de pages jaunes à la plateforme.
- Le ACC "Agent Communication Channel" gère les différentes communications entre les agents.
- Le AMS "Agent Management System" supervise l'enregistrement des agents, leur authentification, leur accès et l'utilisation du système.

La communication entre agents est exécutée par messages FIPA ACL. La plateforme d'agent peut être répartie sur plusieurs serveurs. Une seule application Java, et donc une seule machine virtuelle de Java (JVM), est exécutée sur chaque serveur. Chaque JVM est un conteneur d'agents qui fournit un environnement complet pour l'exécution d'agent et permet à plusieurs agents de s'exécuter en parallèle sur le même serveur.

Les comportements « Behaviour » permettent aux agents d'agir dans un environnement. Un comportement est une tâche qu'un agent doit réaliser à un moment précis et selon des contraintes. Dans ce travail trois types de comportement (Behaviour) sont adoptés:

- **OneShot Behaviour***:* est un comportement qui s'exécute une seule fois et se termine.
- **CyclicBehaviour***:* est un comportement qui se répète. L'avantage de ce comportement dans l'instruction *Block().* Cette instruction permet de bloquer le comportement jusqu'à la validation d'une condition.
- **FSMBehaviour***:* ce type de comportement est défini par des états et des transitions. On dit que la transition est validée si la valeur de retour du FSMBehaviour est le poids de cette transition. Chaque état peut être un OneShotBehaviour ou CyclicBehaviour.

Compte tenu de touts ces caractéristiques et fonctions contrôlant un SMA, JADE à été choisi pour ce travail.

2.7 Conclusion

Nous avons présenté dans ce chapitre, les principales caractéristiques du concept agent et des SMA.

Nous avons mis l'accent dans ce chapitre sur le rôle déterminant de l'interaction dans les SMA, et les différentes formes d'interaction qu'elle peut prendre.

La notion d'interactions crée par les possibilités d'échanges et d'influences entre les agents est fortement liée à la notion de construction de stratégies de résolution coopérative des problèmes, offre un cadre de développement et d'analyse très intéressant à notre problématique. En effet, la modélisation d'une procédure de surveillance distribuée est une des applications possible des SMA. Une architecture logicielle orientée agent doit être aussi naturelle que possible i.e. que chaque agent

doit être identifié clairement par rapport à sa sémantique, son rôle et sa responsabilité.

Ainsi, le chapitre suivant est consacré à la présentation de notre modèle multi agents pour la surveillance distribuée en analysant le problème en termes d’interaction entre les agents.

Chapitre 3

Approche méthodologique pour la Surveillance Distribuée à base d'Agents Coopérants: SDAC

Dans ce chapitre nous proposons une approche méthodologique pour la Surveillance Distribuée à base des Agents Coopérants baptisée SDAC. Les concepts principaux pour étaler notre proposition sont regroupés dans cinq modèles. Ainsi, la conception et la modélisation de ces différentes modèles est l'une des contributions majeure de ce travail.

3.1 Introduction

Le premier chapitre nous a permis de mettre en évidence la problématique de la surveillance des systèmes complexes dans un contexte distribué. Nous avons vu l'intérêt d'avoir un système de surveillance distribuée par rapport à un système centralisé. Les SMA présentés dans le deuxième chapitre sont bien adaptés à la problématique. Ainsi, l'approche méthodologique pour la Surveillance Distribuée à base des Agents Coopérants (SDAC) a pour objectif de multiplier les entités intelligentes (agents) et de les doter de facultés de communication pour découvrir par coopération une solution globale au problème de la surveillance d'un système complexe. Le premier objectif de SDAC est de permettre de résoudre les problèmes complexes de la surveillance industrielle au niveau détection et diagnostic des défaillances en les décomposant en une multitude problèmes élémentaires et en construisant des entités autonomes qui pourront en coopérant participer à la construction d'une solution globale. Dans notre travail, nous nous sommes intéressés spécialement à la distribution de l'étape d'analyse diagnostic.

L'autre grand objectif est d'aboutir à des systèmes de surveillance modulaires et ouverts où le fait d'ajouter un agent ou de modifier la structure d'un système n'induit pas une re-conception d'une solution mais une converge automatiquement à la suite d'un processus d'apprentissage vers une nouvelle solution globale.

Il est question dans ce chapitre de présenter les principales caractéristiques de SDAC et les modèles sous-jacents.

3.2 Caractéristiques de SDAC

3.2.1 Hybridation de deux approches DX et FDI

L'adoption d'une structure distribuée pour la surveillance a engendré la distribution des différentes fonctions participant à la réalisation de la tâche de surveillance. La solution méthodologique que nous proposons pour SDAC réside dans l'utilisation à la fois des outils de calcul et des formalismes issus des techniques FDI et DX. Ainsi, la phase de détection est résolue par les outils de la communauté FDI tandis que le problème de diagnostic est résolu à l'aide des concepts de la communauté DX offrant ainsi une démarche méthodologique hybridée.

La distribution de l'étape de diagnostic est plus importante dans ce travail. Notre contribution concerne la mise en place de la fonction de diagnostic distribuée par l'adaptation de la méthode dite du diagnostic à base de cohérence pour l'élaboration des diagnostics locaux dans les différents agents de surveillance ainsi que les interactions nécessaires à la réalisation du but de diagnostic.

3.2.2 Appui à la distribution: des diagnostiqueurs locaux vers un diagnostic global

Pour éviter le problème de charge de l'organe principal d'une architecture centralisée et pouvoir diagnostiquer des systèmes fournissant des quantités d'informations supérieures, l'approche distribuée est adoptée [Pencolé 2005]. Elle permet de diminuer la charge due aux masses de données en la distribuant via des entités de diagnostic local. Chaque entité a une vue partielle ou locale du système à diagnostiquer. Ces entités locales de diagnostic doivent communiquer entre eux afin de tendre localement et globalement à un diagnostic cohérent.

Mettre en place une approche distribuée doit permettre d'améliorer la fiabilité du système. En effet, aucun organe n'est hiérarchiquement en charge d'établir un diagnostic global du système; ce qui implique qu'un diagnostic global existe toujours à moins que toutes les entités de diagnostic locales tombent en panne.

3.2.3 Exploitation des TIC

- **Exploitation de la technologie agent :**

Les SMA offrent un cadre intéressant à notre problématique. Ils permettent de modéliser les systèmes de surveillance complexes à l'aide d'une société d'agents coopérants. Ils fournissent des abstractions, des protocoles et des langages de haut niveau pour traiter la distribution, l'hétérogénéité et l'autonomie qui sont inhérentes à l'organisation coopérative.

- **Exploitation de la technologie services web**

La coopération des systèmes de surveillance interentreprises est importante afin d'augmenter les gains et d'améliorer la productivité. Pour coopérer efficacement ils doivent partager l'information. La technologie service web est retenue pour faire coopérer SDAC avec d'autres systèmes de surveillance distants en éliminant les difficultés d'accès à l'information et l'hétérogénéité de ceux-ci. Il s'agit d'une technologie permettant à des applications de

dialoguer à distance via Internet et ceci indépendamment des plateformes et des langages sur lesquelles elles reposent.

3.2.4 Inspiration des modèles de MAS-CommonKADS

Plusieurs méthodologies pour le développement des SMA ont été proposées. Elles constituent, soit une extension de méthodes orientées-objet, soit une extension de méthodes à base de connaissance. Certaines sont conçues pour un contexte particulier.

Le travail réalisé par [Picard 2004] nous a servi de base. Un comparatif complet de méthodes orientées agents a été effectué. Nous avons ainsi dressé le tableau filtre (tableau 3.1). En colonnes apparaissent les méthodes avec une couleur par catégorie de méthodes et en ligne les critères classés par catégorie.

Tableau 3.1 Comparatif de méthodologies orientées agent (adapté de [Picard 2004]).

		Gaia	Mase	Message	DESIRE	MAS- commonKads	Tropos	Aaladin
Concepts	Croyances	*	*	-	**	*	**	*
	Désirs	*	**	*	*	*	*	*
	Intentions	*	*	*	*	**	**	*
	Interaction	*	**	**	*	*	*	**
	Organisation	**	*	**	-	*	*	**
	Rôle	**	**	**	*	*	**	**
Processus développement	Modularité	**	*	**	*	*	*	*
	Analyse	**	**	**	*	*	**	**
	Conception	**	**	**	**	**	*	-
	Implémentation	-	*	*	*	*	*	**
Pragm -atique	Expertise requise	Oui	Non	Non	Oui	Oui	Oui	Non
	Langage spécification	Non	Non	Non	Oui	Non	Non	Oui

Notation : (**) pour les propriétés pleinement et explicitement prises en charge;

(*) pour les propriétés prises en charge de manière indirecte;

(−) pour des propriétés non prises en charge.

Afin de sélectionner la méthode la plus appropriée à notre situation nous avons réalisé une simulation sous Matlab (figure 3.1)

La lecture du tableau filtre et la simulation sous Matlab montrent que les agents déployés par la méthode MAS-CommonKADS sont à forte granularité (agents cognitifs) et propose des techniques de conception des protocoles dans le but de modéliser les agents et les interactions entre agents.

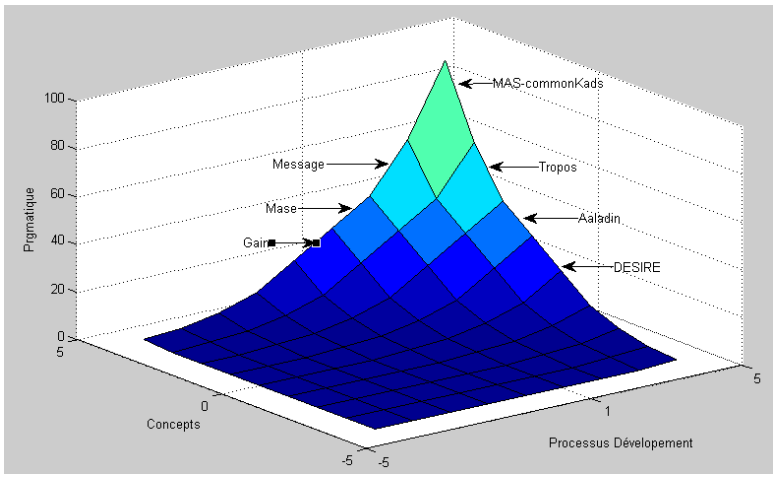


Figure 3.1 Les méthodologies orientées agent sous Matlab.

La méthode MAS-CommonKADS [Iglesias 1998] est orientée connaissance, elle constitue une extension de CommonKADS [Schreiber 1994] qui est vue actuellement comme un standard européen pour la modélisation des connaissances. Elle ajoute des techniques venant des méthodes orientées objet à des techniques de conception des protocoles dans le but de modéliser les agents et les interactions entre agents, convient à notre situation.

La méthodologie prévoit une ***phase de conceptualisation*** dédiée à recueillir les besoins de l'utilisateur et à obtenir une première description du système. Ensuite, la méthodologie définit les modèles (figure3.2) décrits ci-dessous pour ***l'analyse*** et la ***conception*** d'un système.

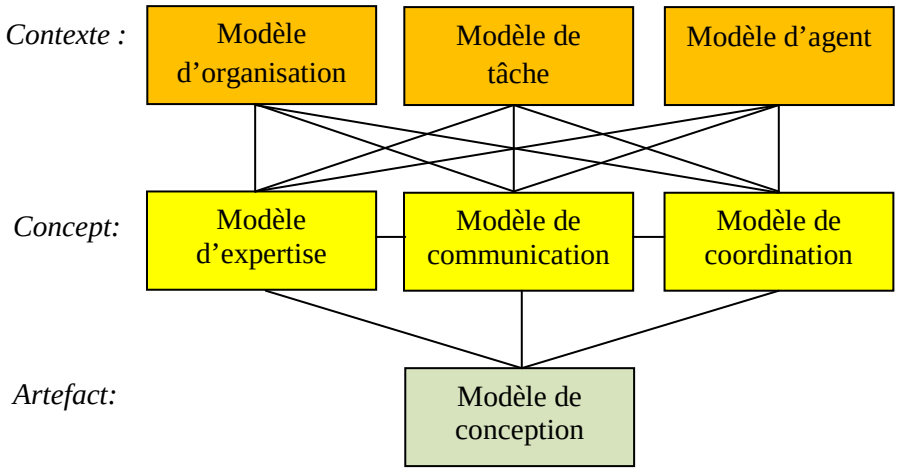


Figure 3.2 Les modèles de MAS-CommonKADS [Ghomari 2008].

Les modèles au nombre de sept, décrivent les caractéristiques principales des agents, leur organisation, les tâches devant être exécutées par ceux-ci. L'expertise nécessaire aux agents pour réaliser ses tâches, les interactions entre les agents pour se coordonner et les objets conceptuels issus des modèles précédents. Donc ces modèles sont très complets. Cependant, l'utilisation d'un spécimen textuel pour décrire les modèles et les redondances remarquées entre eux présentent pour nous une barrière pour l'application de ces modèles intégraux à notre travail.

Aussi, dans notre approche SDAC, nous allons adopter la décomposition en phase de conceptualisation et définition comme MAS-CommonKADS. Nous avons inspirer des modèles de celle-ci pour développer cinq modèles couvrant les étapes d'analyse et de conception de notre système et sans redondances entre eux. Ces modèles gardent les mêmes noms que ceux de MAS-CommonKADS, cependant leurs contenus sont adaptés à notre situation en se basant sur les travaux de [wooldridge 1995] et [Ferber 1995]. En effet, les concepts principaux pour analyser le problème de la surveillance distribuée en termes d'interactions entre les agents sont regroupés dans les modèles d'agent, d'expertise, de tâche, de coordination et de conception. Il est à noter que les modèles d'organisation et communication sont intégrés dans les autres modèles, en particulier, le modèle de la coordination. Aussi, ces modèles sont présentés en utilisant le formalisme UML pour modéliser le système à un haut niveau d'abstraction et de lisibilité.

Après la description de la phase de conceptualisation, nous allons détailler dans les sections suivantes les modèles de SDAC.

3.3 Phase de conceptualisation de SDAC

Cette phase permet de solliciter les besoins d'utilisateurs et avoir une description préliminaire du problème. La figure 3.3 montre le diagramme de *cas d'utilisation* qui donne une vision globale du comportement fonctionnel de SDAC et présente les utilisateurs potentiels ainsi que les différentes fonctions auxquelles il doit répondre. Deux acteurs externes en interaction avec SDAC :

- Après l'authentification, *l'expert* spécifie le modèle du bon fonctionnement du système complexe surveillé. Après l'acquisition des données, la fonction détection distribuée a pour rôle de détecter toute violation des spécifications du bon fonctionnement du système surveillé.
- Ce modèle et les résultats de détection sont utilisés par la fonction de diagnostic distribuée qui est le résultat d'assemblage de tous les diagnostics partiels des sous systèmes composant le système complexe.
- La fonction de diagnostic distribuée affiche à *l'opérateur* le mode de fonctionnement de système. Tous les états des composants de système (normal, défaillant, inconnu) et toutes les alarmes indiquant les problèmes possibles. En se basant sur ces résultats l'opérateur décide des maintenances nécessaires.
- La possibilité d'utiliser des services web fourni par des fournisseurs distants sera détaillée ultérieurement.

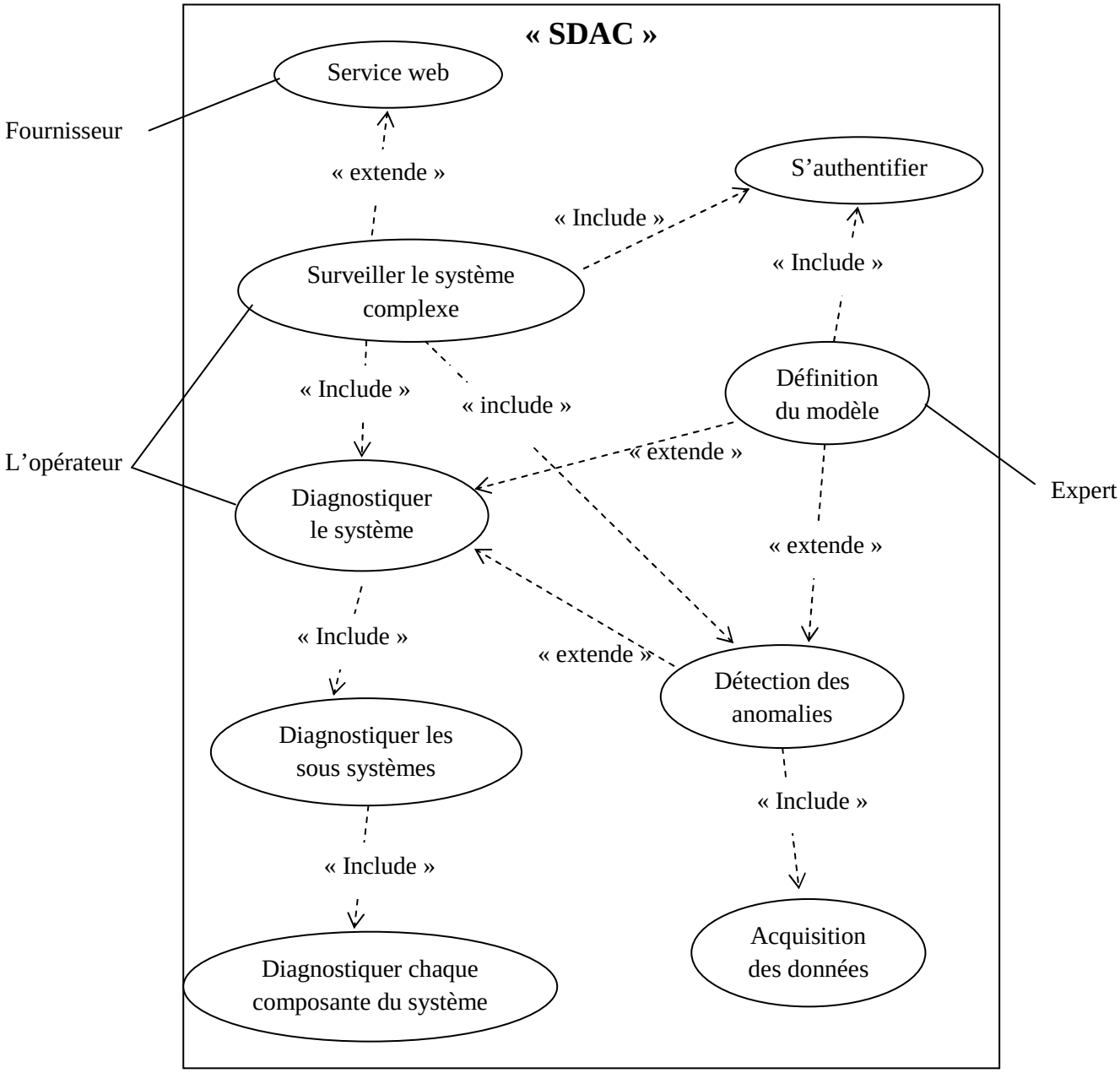


Figure 3.3 Diagramme de cas d'utilisation de SDAC.

3.4 Les modèles de SDAC

Pour SDAC, nous avons besoin de modèles décrivant les caractéristiques principales des agents, des tâches assurées par ceux-ci, des connaissances requises, des comportements des agents pour réaliser ces tâches et des interactions entre ces agents (figure 3.4).

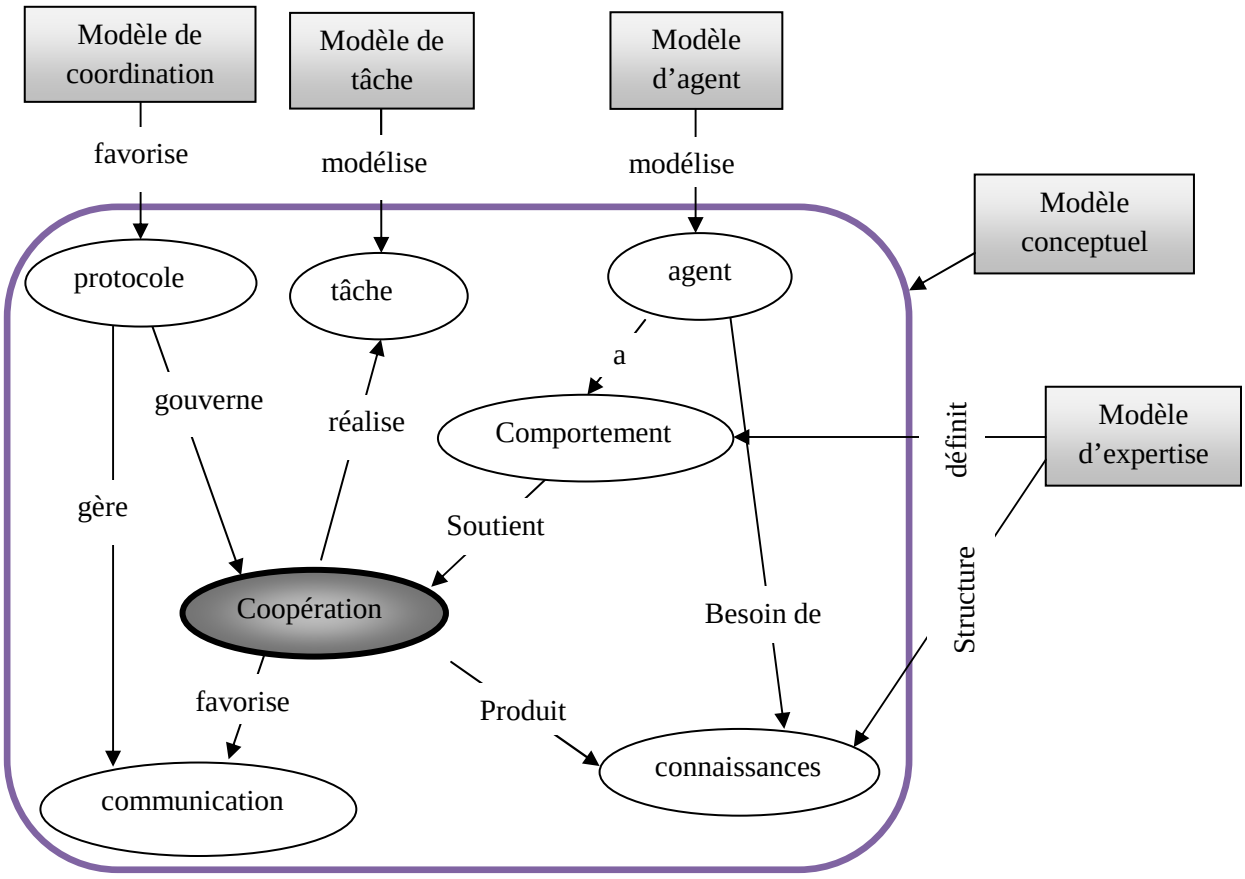


Figure 3.4 Synthèse des modèles de SDAC.

Ces modèles fournissent un moyen d'organiser la *coopération* inter-agents. En effet :

- *Le modèle d'agent:* il décrit les caractéristiques principales des agents de SDAC, y compris les habiletés et les tâches.
- *Le modèle d'expertise:* il décrit les connaissances et les comportements de ces agents.
- *Le modèle de tâche:* il décrit les tâches devant être exécutées par les agents et la décomposition des tâches.
- *Le modèle de coordination*: il décrit l'organisation de la société d'agents et les conversations entre ceux-ci.
- *Le modèle de conception*: il décrit les principaux objets issus des modèles précédents.

Les modèles sont repris de manière détaillée dans les sections suivantes:

3.4.1 Modèle d'Agent de SDAC

Chaque agent de SDAC construit ses propres structures opératoires, selon ses propres connaissances et modes de représentation du monde en fonction de ses capacités cognitives, de son histoire, de ses relations avec son environnement.

Selon [wooldridge 1995], le développement de tout SMA amène à s'interroger sur la distinction entre les comportements d'un agent indépendant des actions des autres agents d'un système et de ceux résultant d'interaction entre agents. Ces réflexions nous ont conduites à la décomposition du modèle d'agent en un modèle *individuel* et un modèle *social* (figure 3.5):

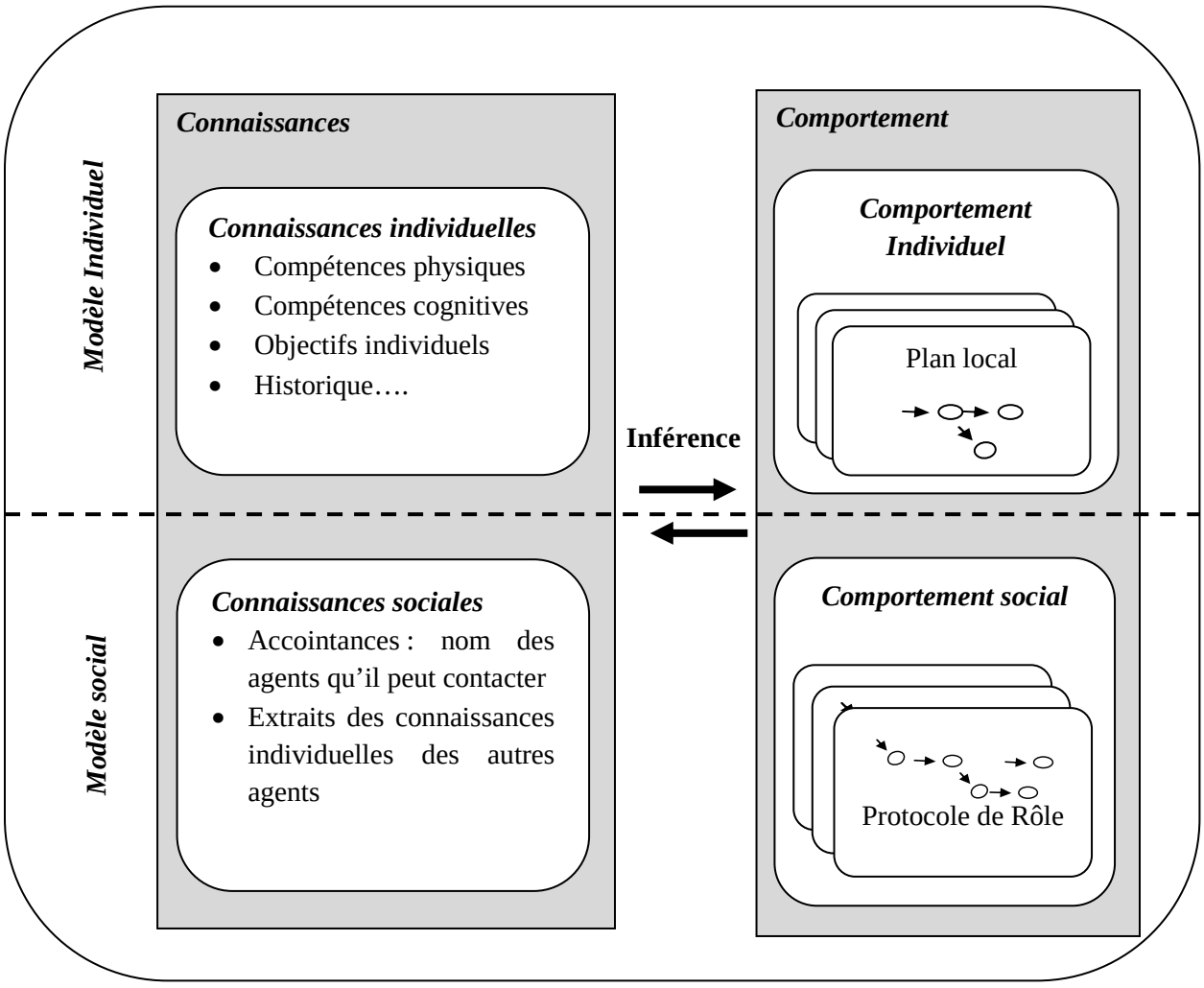


Figure 3.5 Modèle d'agent pour SDAC.

- ***Connaissances :*** les connaissances d'un agent de SDAC contiennent l'ensemble des informations détenues par un agent sur lui-même (connaissances individuelles) ou sur les autres (connaissances sociales) :
  - Les *connaissances individuelles* forgent son identité i.e. qu'elles identifient ce qu'il est capable de faire d'un point de vue technique (Compétences physiques) et d'un point de vue traitement de l'information (Compétences cognitives) et ce qu'il fait ou va faire (Objectifs individuels).
  - Les *connaissances sociales* caractérisent l'implication d'un agent dans le SMA. Elles portent d'abord sur les accointances d'un agent i.e. l'ensemble des agents dont il a connaissance de l'existence. Cette connaissance peut être complétée par des extraits des connaissances individuelles de certains de ces agents.
- ***Comportements*** : Alors que les connaissances décrivent ce que sait faire un agent de SDAC, les comportements détaillent le comment.
  - Les *comportements individuels* décrivent une séquence d'actions qu'un agent réalise dans un contexte donné. Le plan local décrit le comportement isolé d'un agent i.e. ne nécessitant pas d'interaction avec les autres agents.
  - Les *comportements sociaux* (protocole de rôles) se traduisent par des interactions entre au moins deux agents. Le protocole de rôle défini la tâche de chacun des agents durant leurs interactions au travers d'échanges de message formant ainsi une conversation.

La notion de plan utilisée ici pour décrire le comportement d'un agent est similaire à celle rencontrée dans les approches BDI [Bratman 1988]. Retenons qu'un plan spécifie une séquence d'actions élémentaires tels que l'envoi de message, des calculs divers, la mise à jour des connaissances, etc.

L' analyse préalable des principales caractéristiques des agents de SDAC et des rôles à jouer dans le cadre de l'organisation de la coopération entre agents de diagnostic locaux par rapport aux objectifs globaux nous a permis d'identifier deux types d'agents (figure 3.6) :

- les agents cognitifs qui ne sont autre que l'agent détecteur, de diagnostic local et d'évaluation
- les agents réactifs qui sont relatif à l'agent d'information, interface et de découverte.

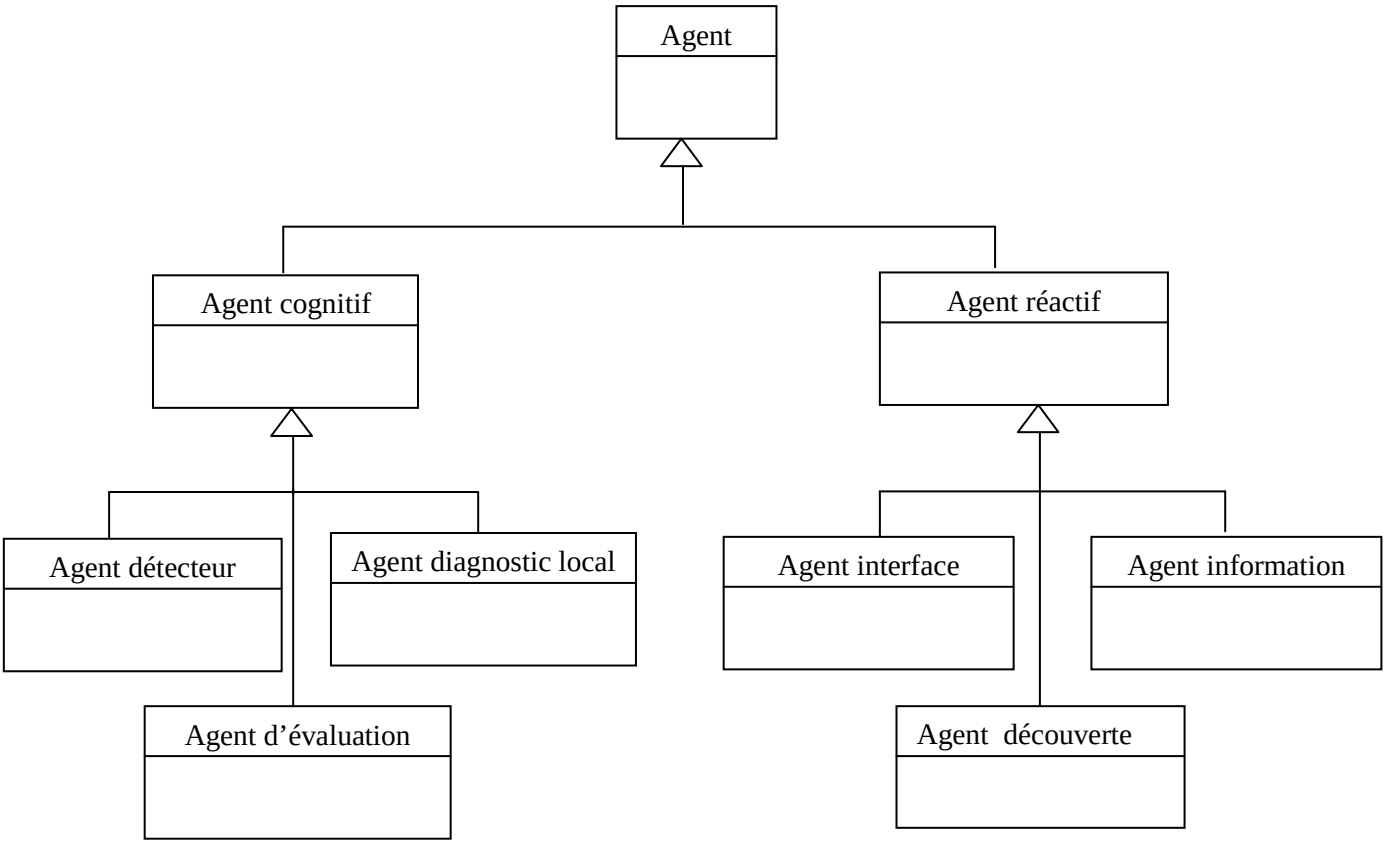


Figure 3.6 principaux agents pour SDAC.

En effet, les deux étapes de la surveillance à savoir détection et diagnostic sont réalisées par la coopération des agents détecteurs et de diagnostics locaux en servant des informations fournis par l'agent d'information et les services web fournis par l'agent de découverte. Après l'évaluation effectuée par l'agent d'évaluation, les résultats sont affichés par l'agent d'interface. Les architectures de ces six agents sont développées dans les sections suivantes :

Agent interface

Il permet à l'utilisateur d'interagir et d'afficher les informations présentées par le système de la surveillance et de visualiser les résultats, l'état du système fourni par l'agent d'évaluation ainsi que les différents symptômes et messages collectés sur le système et les conseils de maintenance. Comme il peut aider l'opérateur d'initialiser et d'interroger les différents agents du système.

- ***Agent de découverte***

Dans notre travail, les services web sont considérés comme des ressources exploités par les agents détecteurs et les agents de diagnostic locaux afin d'optimiser la détection et

le diagnostic des défauts. Ainsi l'agent de découverte prend en charge la découverte dynamique de services Web en se connectant à un annuaire de service (Universal, Description, Discovery and Integration). Il invoque ensuite le service approprié et permet l'intégration de services web et l'agent logiciel.

- ***Agent d'information***

Il gère les informations sur les composantes matérielles du système physique, y compris le planning de l'entretien, l'historique de la réparation, durée de vie et conditions de travail, etc. Cet agent (figure 3.7) minimise le temps de diagnostic en recherchant l'ancien diagnostic si les symptômes sont les mêmes dans la base de données et celle de connaissances. Ceci constitue un avantage capital.

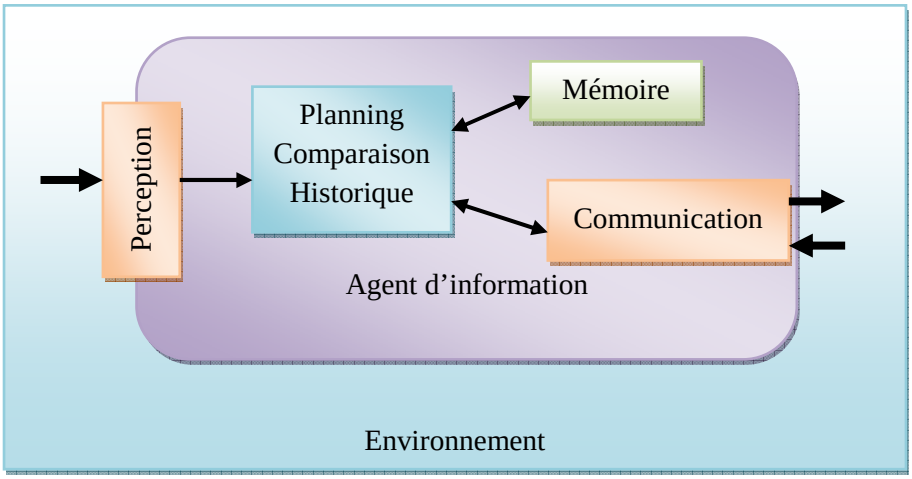


Figure 3.7 Architecture de l'agent d'information.

- ***Agent détecteur (AD)***

L'interface entre le processus et le système de surveillance est réalisé par les agents détecteurs (AD). Ils sont responsables de l'acquisition et le traitement des données du processus récupérées à partir des capteurs puis réaliser les prétraitements (filtrage, mise en format, validation, etc.) afin de les préparer à l'étape de génération des symptômes (figure 3.8). Chaque AD encapsule un algorithme de détection qui vérifie la consistance entre le comportement réel d'un système physique tel qu'il peut être observé par l'intermédiaire des capteurs par exemple et son comportement attendu tel qu'il peut être prédit grâce aux modèles de bon ou mauvais comportement. Les données et les connaissances ont ensuite envoyées à la base de données et de connaissances.

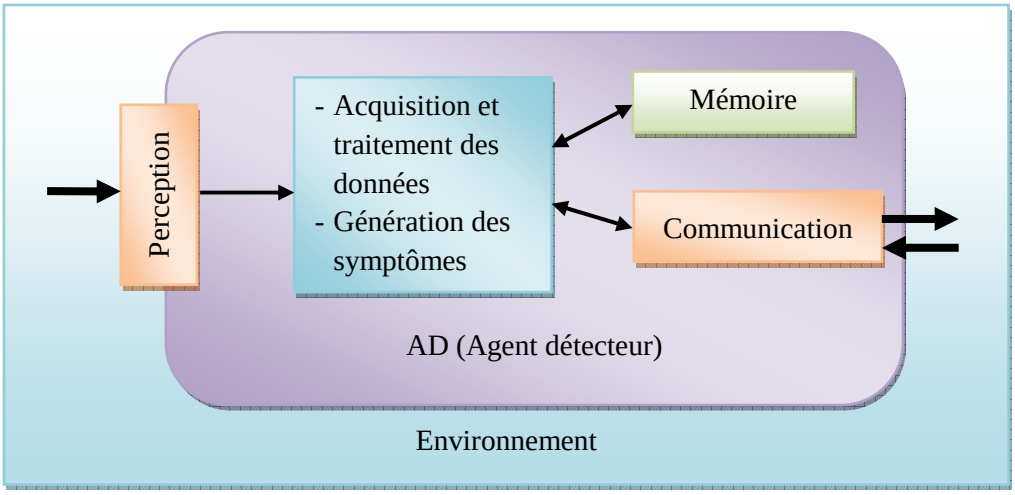


Figure 3.8 Architecture de l'agent détecteur.

- ***Agent diagnostic local (ADL)***

Il analyse les symptômes disponibles fournis par les ADs pour déterminer les états plausibles d'un composant ou sous système physique. La localisation permet de remonter à l'origine de l'anomalie et de localiser le (ou les) composant(s) défectueux. Dans le modèle que nous proposons, le raisonnement diagnostic est totalement distribué à travers plusieurs agents d'analyse selon une distribution sémantique des connaissances. Ainsi, chaque agent d'analyse a comme tâche de diagnostiquer un sous ensemble de composants du système et ce d'une manière autonome tout en gardant la possibilité d'avoir des agents d'analyse avec des connaissances dépendantes. Ceci implique une coopération des agents pour le calcul des diagnostics locaux. La figure 3.9 présente l'architecture d'un ADL.

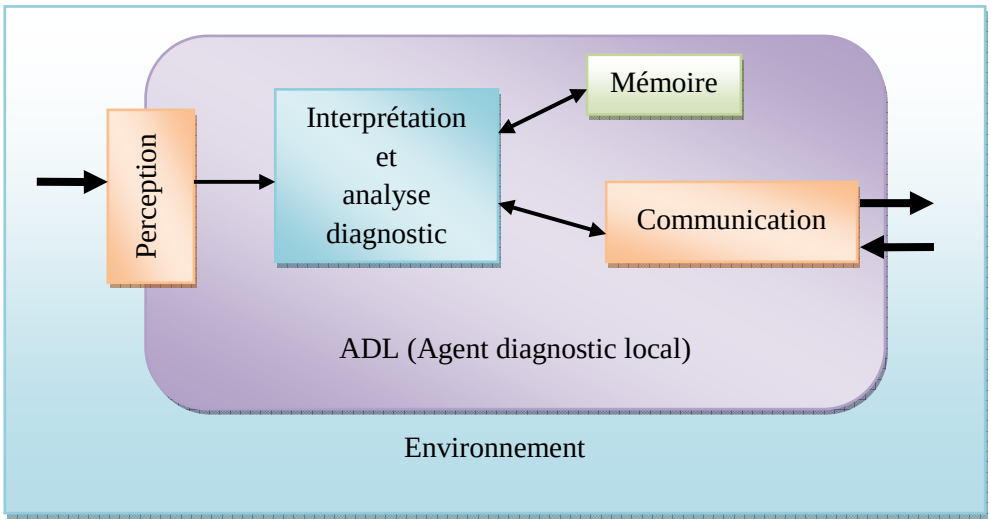


Figure 3.9 Architecture de l'agent de diagnostic local.

- ***Agent d'évaluation***

Il reçoit les différents diagnostics locaux fournis par les ADL et calcule le diagnostic global en fusionnant les diagnostics locaux en un seul diagnostic global (figure 3.10). Les résultats trouvés (diagnostic final) envoyés à la base de données et de connaissances pour une éventuelle utilisation en temps que connaissances expertes.

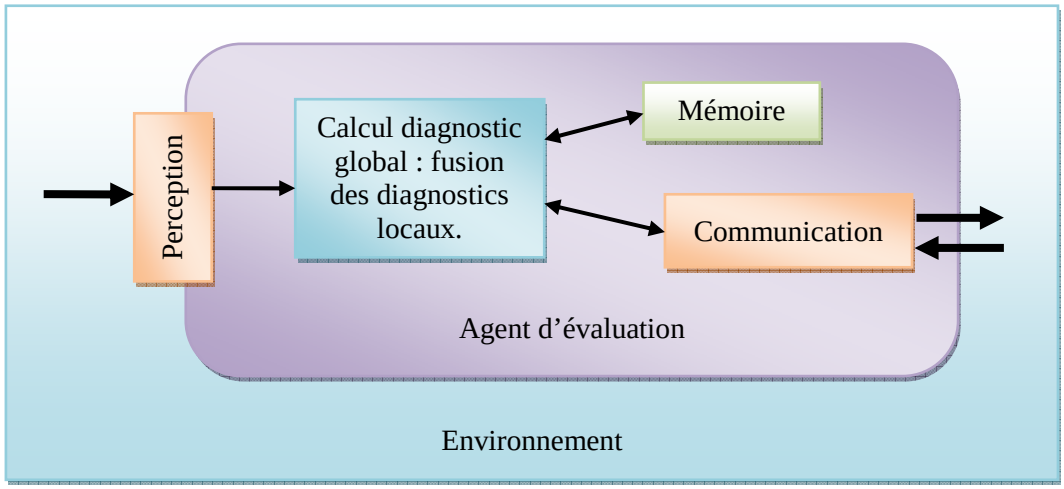


Figure 3.10 Architecture de l'agent d'évaluation.

La coopération entre les agents de SDAC consiste à décomposer les tâches de la surveillance distribuée en sous-tâches puis à les répartir entre les différents agents. Il existe plusieurs décompositions possibles. La section suivante est consacrée à la description de processus de décomposition en spécifiant un modèle de tâche.

3.4.2 Modèle de tâches de SDAC

Trois approches permettant de décomposer un problème, qualifié de complexe, en sous-problèmes ou sous-tâches [Ghomari 2008]:

- ***Approche orientée par les buts :***

Un but est un point de convergence que nous cherchons à atteindre par les activités, de résolution d'un problème. Un agent identifie le but global à atteindre. Ce but est considéré comme un ensemble fini de sous-buts qui correspondent chacun à une tâche directement affectable à un agent potentiel.

- ***Approche orientée par les objets d'information***

Le problème global est considéré ici, comme un ensemble d'objets d'information hétérogènes. Tout d'abord, un agent va regrouper ces objets d'information en paquets thématiques cohérents. Ensuite, dans chaque paquet, il identifie des sous-paquets permettant d'identifier des sous-tâches spécialisées utilisant des objets d'information spécialisés.

- ***Approche orientée par les compétences***

Selon cette approche, un agent envoie la description du problème global à tous les agents du système en leur demandant de définir ce qu'ils peuvent faire pour contribuer à la résolution du problème en question. Elle est utile pour résoudre des problèmes complètement nouveaux et conduit les agents à devoir coopérer très tôt dans le processus de décision.

Ces approches peuvent être utilisées séparément ou simultanément. La troisième approche semble la mieux adaptée à notre situation. En effet, l'allocation des tâches aux différents agents particulièrement les ADs et les ADLs est effectuée selon les compétences de chacun.

La situation de la surveillance distribuée qui nous intéresse fait ressortir deux catégories de tâches (figure 3.11) :

- Tâches coopérantes: elle est définie par le «quoi ?» (sa finalité) et le «qui ?» (agent responsable de sa réalisation). La manière dont la tâche est effectuée, le « comment ? » relève d'une démarche cognitive (expertise). Les agents de détection, les agents de diagnostic locaux et l'agent d'evaluation sont concernés par ce type de tâche.
- Tâches procédurales: ici le « comment » est de nature procédural. Ces tâches prescrites que nous appellerons de soutien. L'agent interface, d'information et de découverte sont concernés par ce type de tâche.

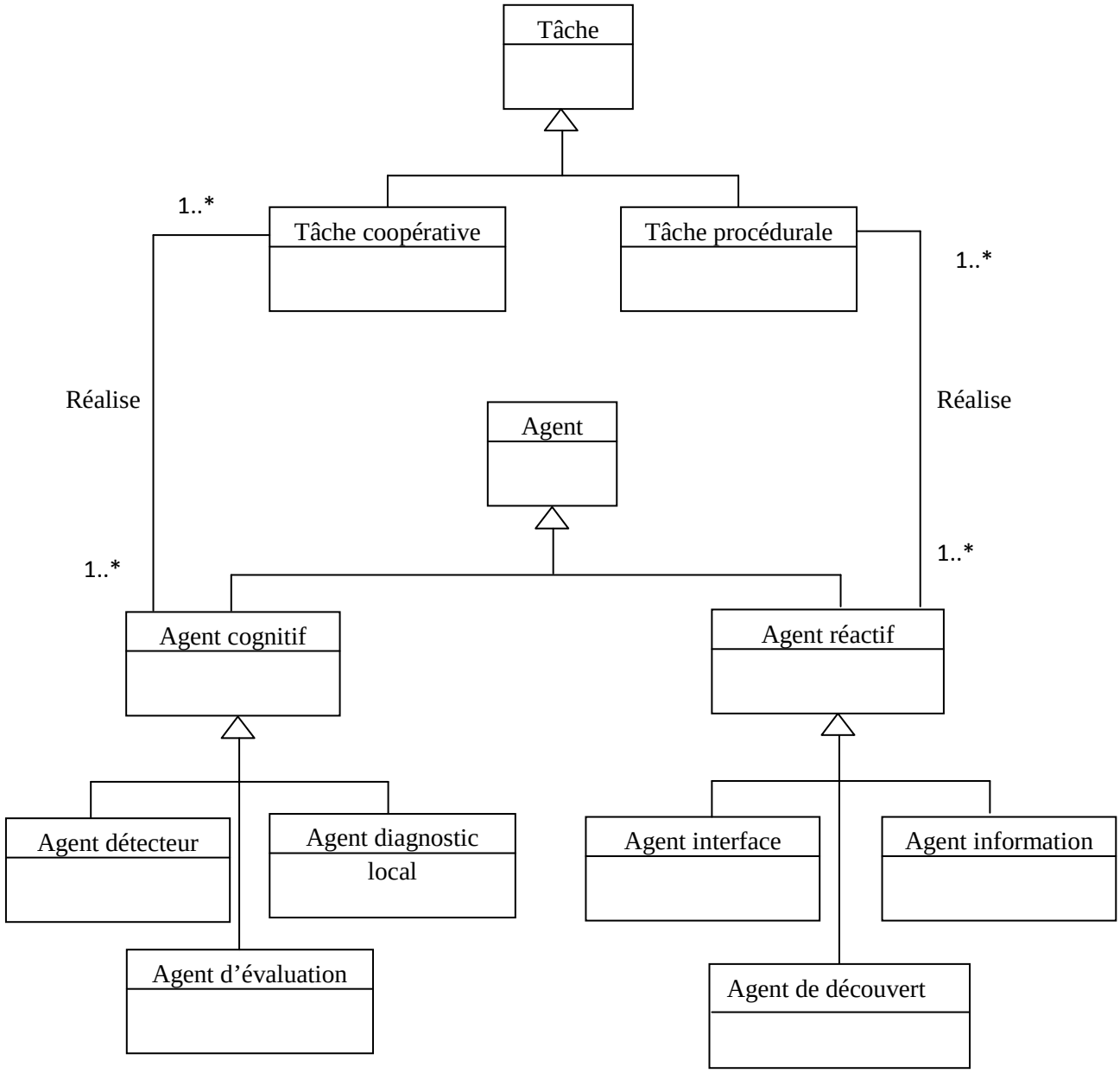
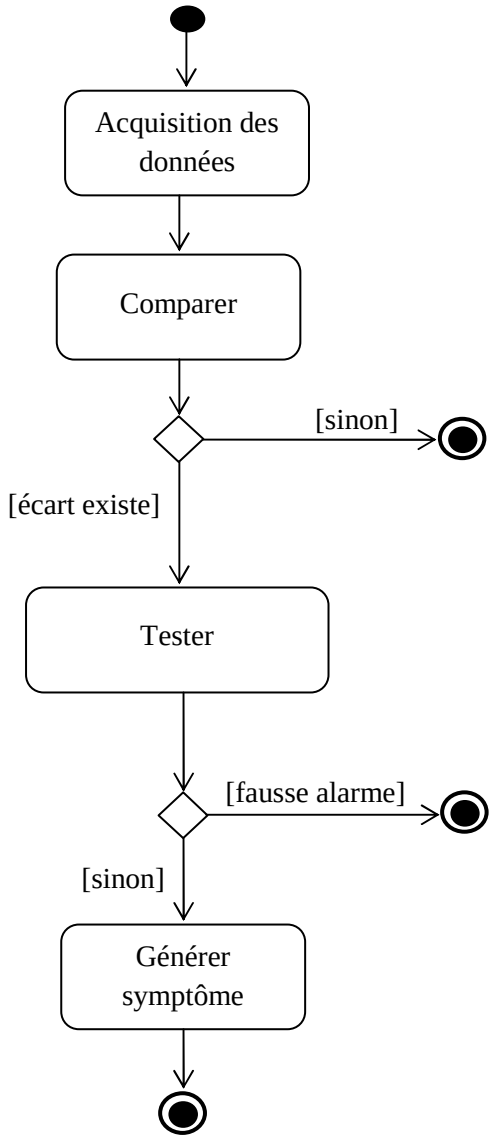


Figure 3.11 Diagramme de classes pour la surveillance distribuée.

La décomposition des tâches est faite selon les modèles de gestion de projets que sont WBS (Work Breakdown Structure), méthode de découpage hiérarchique arborescente du projet en composants élémentaires. L'allocation des tâches aux agents détecteurs et diagnostiqueurs est effectuée selon les compétences de chacun. La figure 3.12 représente le diagramme d'activités avec le Template textuel pour la tâche de la détection concernée par les agents de détection. Le Template textuel décrive les activités principales de celle-ci (nom, petite description, les ingrédients en entrés /sorties, etc.)



Activité « Comparer »

Objectif : comparer le comportement réel du système avec le modèle du système en fonctionnement normal

Description : vérifier si un ensemble d'informations représentatives de l'état d'un système physique est cohérent avec la connaissance d'un comportement donné qui peut être normal ou anormal. Le résultat de la comparaison produit un écart, appelé résidu.

Ingrédient : cette activité à besoin de données acquises via les capteurs et un modèle de bon fonctionnement prédéfini.

Activité « Tester »

Objectif : comparer le résidu à des seuils prédéfinis et fixes.

Description : le résidu résultant de la comparaison est comparé à des seuils fixés à priori. Si le seuil de la détection est trop petit, il peut y avoir de fausses alarmes sinon le test de détection permet de générer les symptômes.

Ingrédient : résidu et seuils.

Contraintes : aucune.

Exceptions : aucun.

Figure 3.12 Diagramme d'activités de la tâche détection avec le Template textuel des activités principales.

Les symptômes fournis par les agents de détection peuvent être regroupés dans une table de signatures qui servira de base pour les agents de diagnostic locaux. Le raisonnement diagnostic vise à déterminer l'ensemble des fonctions défaillantes et localiser leurs causes qui peuvent être un ensemble de composants en faute à partir d'un ensemble

de symptômes. Les figures 3.13 a) et b) illustrent respectivement le diagramme d'activité de la tâche diagnostic et le Template textuel décrivant ses activités principales.

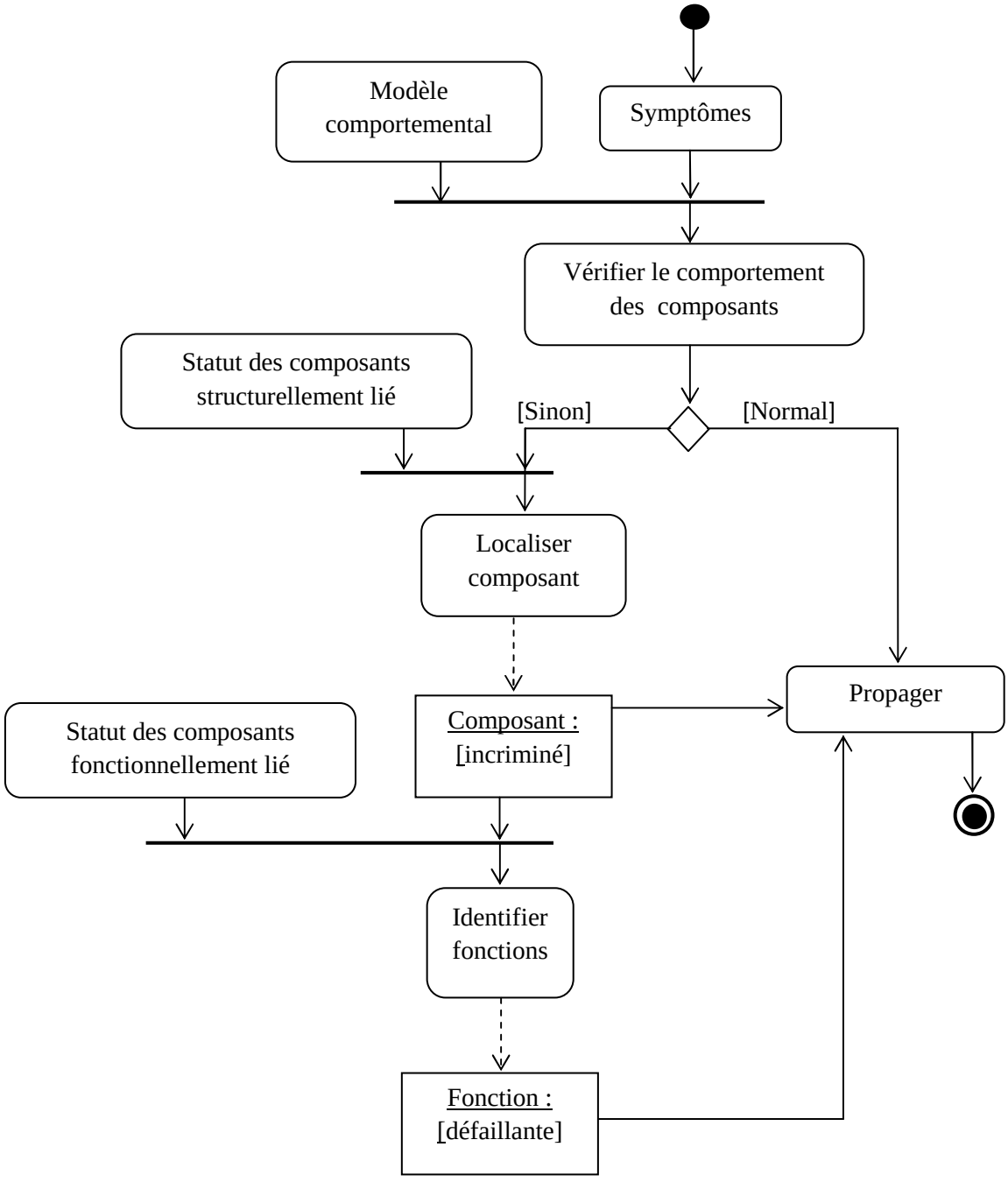


Figure 3.13 a) Diagramme d'activités de la tâche diagnostic.

Activité « Vérifier le comportement des composants »

Objectif : vérifier le comportement des composants selon un modèle comportemental

Description : afin de déterminer le comportement global du sous système, l'ADL vérifie la dynamique de ses composants selon le modèle comportemental qui peut décrire les différents modes de fonctionnement des composants. Nous nous intéressons aux modèles de mauvais fonctionnement et donc à la définition des relations entre défaillances et symptômes.

Ingrédient : symptômes, modèle comportemental

Activité « Localiser composant »

Objectif : localiser le composant incriminé entrainant une défaillance fonctionnelle

Description : le diagnostic vise à déterminer le composant en faute ayant entrainé une défaillance fonctionnelle à partir d'une surveillance au niveau des composants. La localisation du composant défaillant repose sur l'état du composant et le statut des composants structurellement liés.

Ingrédient : Etat de composant, statut des composants structurellement liés.

Activité « Identifier fonctions »

Objectif : identifier les fonctions défaillantes

Description : le sous système est structuré en plusieurs fonctions implémentées chacune par plusieurs composants. Un composant peut contribuer à plusieurs fonctions du sous système. Les composants défaillants entrainant la défaillance fonctionnelle.

Ingrédient : composant incriminé, statut des composants fonctionnellement liés.

Activité « Propager »

Objectif : déclarer les états des composants et celles des fonctions

Description : L'existence des liens structurels entre composants implémentant des fonctions différentes nécessite des échanges de messages entre les ADL. Ainsi, les résultats obtenus (exemple la défaillance d'un composant) doivent être propagés dans le système pour affiner le diagnostic (par coopération).

Ingrédient : statut du composant, statut de la fonction

Figure 3.13 b) Template textuel des activités principales de la tâche diagnostic de l'ADL.

En complément aux modèles d'agent et tâche déjà réalisés, nous cherchons par le modèle d'expertise de SDAC à décrire de manière détaillée les connaissances nécessaires aux agents SDAC et le raisonnement diagnostic à base de consistance proposée par [Reiter 1987] pour exécuter les tâches de la surveillance distribuée.

3.4.3 Modèle d'expertise de SDAC

Dans notre travail, nous proposons deux approches quantitative et qualitative pour la détection et le diagnostic. Nous nous intéressons à la méthode de Reiter qui s'appuie sur un modèle de la structure du système et du comportement de ses composants pour effectuer des évaluations sur les états du système. Pour ce faire, les ADLs doivent disposer des connaissances et des comportements suivants:

3.4.3.1 Représentation des connaissances

Trois types de connaissances sont nécessaires: *structurelle* qui porte sur la topologie du système qui décrit les liens entre les composants, *comportementale* décrivant le comportement de ses composants et *fonctionnelle* exprimant le rôle des composants d'un système permettant de faire le lien entre le comportement des composants du système à son objectif.

- **Représentation de la connaissance structurelle-fonctionnelle :**

La connaissance structurelle vise à représenter l'ensemble des composants implémentant le système ainsi que leurs interconnexions. Elle représente la base pour la connaissance fonctionnelle. La figure 3.14 illustre la connaissance structurelle-fonctionnelle.

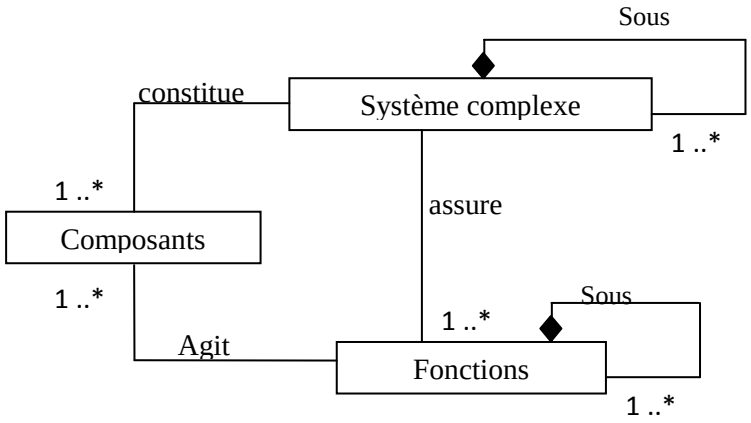


Figure 3.14 la connaissance structurelle-fonctionnelle.

Un système complexe (SC) est considéré comme étant composé de *n* sous-systèmes (SS) physiques. SC= $\{SS_1, SS_2, \ldots, SS_n\}$. Chaque SS_i est composé d'un ou de plusieurs composants. Ces composants peuvent contribués à l'implémentation d'une ou plusieurs fonctions. Le SS_i est défini par un graphe sans cycle où les nœuds représentent les composants et les arcs représentent les variables partagées et les variables privées. Les sous-systèmes sont statiques, physiquement séparés et n'ont pas de composants en commun i.e. $\forall i, j \in 1..n\ i \neq j\ SS_i \cap SS_j = \emptyset$ [Núnez 2005].

Lorsque le SC est considéré dans sa globalité, il apparaît comme une boîte grise ayant des variables *publics observables* (variables d'entrée / sortie système), des variables *privés* (utilisés uniquement par chaque sous système) et les variables *partagées* par les sous systèmes (variables non observables qui apparaissent dans plusieurs sous systèmes).

La figure 3.15 représente un exemple d'un SC décomposé en 3 sous systèmes différentes SS_1, SS_2 et SS_3 avec trois variables globales en entrée { $e_{2,1}$,$e_{1,1}$,$e_{1,2}$} et quatre variables globales en sortie{$s_{6,1}$, $s_{6,2}$, $s_{7,1}$, $s_{7,2}$}.

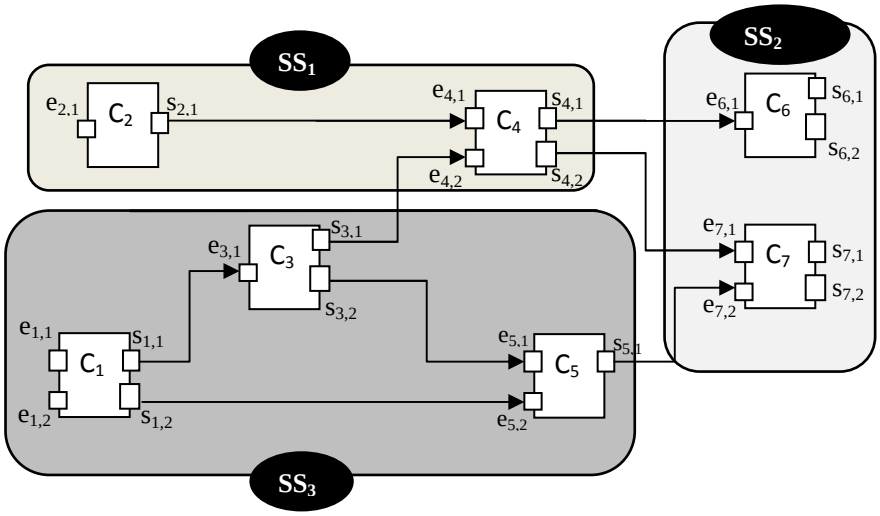


Figure 3.15 Exemple de décomposition d'un SC.

Au niveau SS, nous notons $F_{i,j}$ la fonction numéro i du SS_J avec j ∈ [1; n], n désignant le nombre total de fonctions du SS_J. Supposons que chaque SS_i de l'exemple implémente une seule fonction alors les relations suivantes sont obtenues:

$F_{1,1} : f(\mathrm{C2}, \mathrm{C4})$

$F_{2,1} : f(\mathrm{C6}, \mathrm{C7})$

$F_{3,1} : f(\mathrm{C1}, \mathrm{C3}, \mathrm{C5})$

Ainsi, la représentation structurelle du système qui met bien en évidence les interactions invariant dans le temps entre composants sont modélisées par des variables partagées (variables d'entrée, variables de sortie). La notation $s_{1.1} \rightarrow e_{3.1}$ signifie que $s_{1.1}$ est structurellement connecté à $e_{3.1}$. Cette notation repose bien évidemment sur les objectifs de conception des différents composants en assurant la fonction d'usage du SC.

Lorsque $s_{1.1} \rightarrow e_{3.1}$, la variable de sortie $s_{1.1}$ du composant C_1 impose sa valeur à la variable d'entrée $e_{3.1}$ du composant C_2 $e_{3.1} = s_{1.1}$.

Selon l'exemple de la figure 3.15 nous avons trois sous systèmes SS_1, SS_2 et SS_3 avec les variables suivantes :

- SS_1 avec la variable public $e_{2.1}$ et les variables partagées $\{e_{4.2}, s_{4.1}, s_{4.2}\}$.
- SS_2 avec les variable publics $\{s_{6.1}, s_{6.2}, s_{7.1}, s_{7.2}\}$et les variables partagées $\{e_{6.1}, e_{7.1}, e_{7.2}\}$.
- SS_3 avec les variable publics $\{e_{1.1}, e_{1.2}\}$et les variables partagées $\{s_{3.1}, s_{5.1}\}$.

Le reste des variables sont privées et utilisées uniquement par les sous systèmes. Ces derniers sont reliés entre eux par les variables partagées. Trois connexions structurelles notées STRU, pour cet exemple, sont obtenues :

STRU: $s_{3.1} = e_{4.2}$, $s_{4.2} = e_{7.1}$ et $s_{5.1} = e_{7.2}$

- **Représentation de la connaissance comportementale**

En s'appuyant sur la description structurelle et fonctionnelle des composants du système, il est possible de repérer trois types de modes de fonctionnement pour un composant: nominal, défaillant et anormal. Ces statuts peuvent être considérés comme des états qui doivent être identifiés par l'ADL par coopération avec les autres ADLs (figure 3.16).

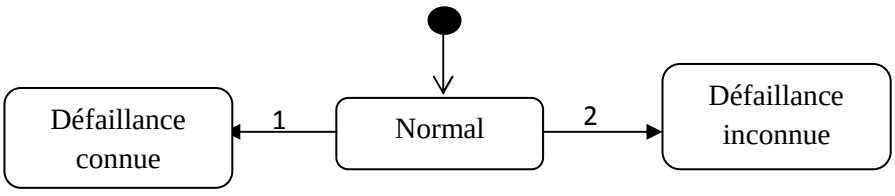


Figure 3.16 Statut d'un composant estimé par ADL.

Les flèches 1 et 2 correspondent à la situation où un symptôme incriminant un composant est reçu. Il y a alors deux situations. La première concerne le cas où le symptôme correspond à une situation de défaillance connue du composant, la seconde

concerne le cas où le symptôme incriminant le composant ne permet pas de statuer sur la nature de la défaillance du composant.

3.4.3.2 La détection

Nous proposons deux approches quantitative et qualitative pour la détection selon les positions des capteurs. Chaque composant a deux capteurs l'un à l'entrée et l'autre à la sortie. Les valeurs mesurées (observations) par les capteurs sont envoyées aux ADs. Chaque AD consulte sa base de connaissance et fait une comparaison entre les valeurs estimées dans un modèle et les valeurs observés avec une marge d'erreur tolérée. Le résultat de ce test est envoyé à l'ADL par un message indiquant l'état de l'entrée /sortie du sous système. Cependant, cette approche *quantitative* basée sur les capteurs en entrée et sortie pour chaque composant ne convient pas aux systèmes complexes comme le système de clinkérisation; notre étude de cas. Pour traiter un niveau élevé de complexité où l'utilisation des capteurs en entrée et sortie pour chaque composant est impossible, aussi pour minimiser le nombre des capteurs l'approche *qualitative* où les agents de détection examinent uniquement les sorties globales du système est plus adaptée.

Selon l'exemple de la figure 3.15, l'approche quantitative suppose l'existence de capteurs pour toutes entrées/sorties de chaque SSi et des ADs testent les valeurs récupérées comme suit :

SS_1: un AD pour chacune des entrées $\{e_{2,1}, e_{4,2}\}$ et chacune des sorties $\{ s_{4,1}, s_{4,2} \}$

SS_2 : un AD pour chacune des entrées $\{ e_{6,1}, e_{7,1}, e_{7,2} \}$ et chacune des sorties $\{ s_{6,1}, s_{6,2}, s_{7,1}, s_{7,2} \}$

SS_3 : un AD pour chacune des entrées $\{e_{1,1}, e_{1,2}\}$ et chacune des sorties $\{ s_{3,1}, s_{5,1}\}$

Cependant, l'approche qualitative suppose l'existence de capteurs pour les sorties globales du système complexe. Donc, quatre ADs testent les valeurs des sorties $\{s_{6,1}, s_{6,2}, s_{7,1}, s_{7,2}\}$.

3.4.3.3 Le diagnostic

Le raisonnement diagnostic est basé sur les résultants de la détection et selon l'approche utilisée. Ainsi, le SMA que nous proposons possède les caractéristiques suivantes:

- Un ADL associé à chaque sous-système.
- Tous ces ADLs participent au calcul du diagnostic (pas de superviseur).

- Chaque agent possède un modèle privé du sous-système (non visible par les autres diagnostiqueurs). Les connaissances incluses dans ce modèle décrivent la structure du sous système à diagnostiquer (connexions entre les composants) et son comportement (obtenu par composition à partir des comportements des différents composants du sous système).
- La connaissance sur le système est sémantiquement distribuée à travers les différents ADLs. C'est pour quoi ;
    - Les ADLs coopèrent pour le calcul des diagnostics locaux en garantissant la cohérence de leurs résultats pour établir le diagnostic global.
    - Chaque ADL doit savoir quelles sont les sous systèmes en entrées et sorties de son sous système associé. Il a besoin de connaître les voisins locaux. Ainsi, chaque ADL implémente une table privée pour enregistrer les relations entre ses variables publics et privées et les autres sous systèmes voisins i.e. les relations structurelles notés STRU.
    - Chaque ADL peut adopter sa propre stratégie de diagnostic et doit uniquement mettre en œuvre une interface de communication avec les autres ADLs.

- **Diagnostic par approche quantitative**

Le système est décomposé en SS. Nous supposons pour cette approche que chaque SS contient uniquement un seul composant avec une sortie et une entrée. Ainsi, l'ADL reçoit les deux messages concernant l'état de l'entrée et de la sortie de SS et fait un diagnostic selon l'algorithme suivant:

Algorithme diagnostic

Début

/* soit SSi le sous système où cet algorithme sera exécuté.

Si la sortie de SSi est correcte **alors** le SSi est en mode nominal

Sinon

Si l'entrée de SSi est correcte **alors** il y a donc un problème au niveau de SSi

Sinon envoi le message à ADLj de SSj relié à l'entrée de SSi

Finsi

Si la sortie de SSj est correcte **alors** il y a un problème de liaison entre les SSs

Sinon il conclu qu'il s'agit d'un problème ailleurs.

Finsi

Finsi

Fin

- **Diagnostic par approche qualitative**

En fonction des sorties globales du système, il est possible de savoir si l'ensemble du système fonctionne correctement. Dans cette approche, pour définir le comportement de chaque ADL, nous avons développé un algorithme de diagnostic distribué en se basant sur les travaux de [Biteus 2005] et [Núnez 2005]. Il est fondé sur l'utilisation des variables partagées entre les SSs, l'envoi de message entre les ADLs pour inférer les variables incorrectes et la génération des diagnostics minimaux. Il est divisé en trois étapes principales (figure 3.17):

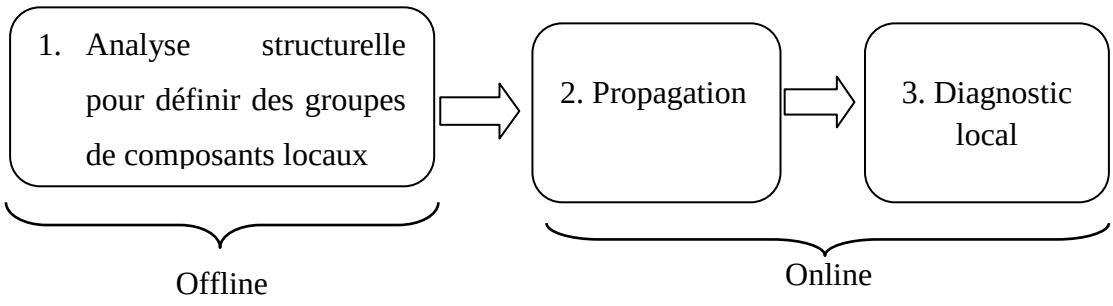


Figure 3.17 Les étapes d'algorithme de diagnostic distribué.

1) Définition des groupes de composants locaux

Formée par un sous-ensemble interne de composants liés par les variables privées ou partagées (dépendance structurelle), et non par variables privées avec les composants d'un groupe différent, cet ensemble est noté IS (internal subsets). Afin de clarifier, nous adoptons les notations utilisées dans [Núnez 2005]:

Entrée (c) sont les variables d'entrée pour le composant c.

Sortie (c) sont les variables de sortie pour le composant c.

Partagée (c) sont les variables partagées pour le composant c.

Privée (c) sont les variables privées pour le composant c.

Définition1 : soit $\{e_1,\ldots,e_q\}$ et $\{s_1,\ldots,s_r\}$ les variables globales respectivement d'entrées et de sorties, IS pour un SS_i est un ensemble de composants où:

- $\forall\, v \in$ *Entrée(c)* ($v \in \{e1, \ldots, eq\}$

$$\bigvee (v \in Privée\ (c) \wedge \exists\, c' \epsilon\, IS(c' \neq c \wedge v \in Sortie\ (c')))$$

$$\bigvee (v \in Partagée\ (c) \wedge \exists\, c' \epsilon\, IS(c' \neq c \wedge v \in Sortie\ (c')))$$

$$\bigvee (v \in partagée\ (c) \wedge v \in partagée(c') \mid c' \in SS_j \neq SS_i))$$

- $\forall\, v \in$ *sortie(c)* ($v \in \{s1, \ldots, sr\} \bigvee (v \in Privée\ (c) \wedge \exists\, c' \epsilon\, IS(c' \neq c \wedge v \in entrée\ (c'))) \bigvee (v \in Partagée\ (c))$

Pour l'ensemble des composants qui forment chaque sous-ensemble interne IS:

- $\forall\, v \in$ *Entrée(IS)* ($v \in Entrée(c) \wedge c \in IS$

$\wedge$ ($v \in \{e1, \ldots, eq\}$ $\vee\ v \in Partagée\ (c)$))

- $\forall\, v \in$ sortie*(IS)* ($v \in sortie(c) \wedge c \in IS \wedge$

($v \in \{s1, \ldots, sr\}$ $\vee\ v \in Partagée\ (c) \nexists\, c' \neq c \mid c' \in IS \wedge v \in Partagée\ (c')$))

Selon l'exemple de la figure 3.15, SS_1 avec IS = {C2, C4}, SS_2 avec IS = {C1, C3, C5} et SS_3 avec IS = {C6, C7}.

2) Propagation

Afin d'expliquer cette étape, de nouvelles définitions doivent être incluses:

Définition2: Variable Incorrecte Potentiel PIV (*Potential Incorrect Variable*) est une variable partagée liée à une sortie incorrecte.

Définition3: Variable Correcte CV (*Correct Variable*) est une variable partagée liée à une sortie correcte globale du système. Si une variable est liée à au moins une sortie correcte et une ou plusieurs sorties incorrectes simultanément, la variable est définie comme une variable incorrecte, puisque nous supposons que deux composants incorrects ne peuvent pas générer une sortie correcte.

Définition4: Variable Incorrecte IV (*Incorrect Variable*) soit *variables_partagées(IS_i)* l'ensemble des variables partagées en sortie de IS_i . Soit v une de ces variables partagées, et c une composante de *ISi*, on dit que v est une variable incorrecte si v est un PIV et $\forall\, IS_j : j \in 1 \ldots n \wedge i \neq j \wedge c \in IS_j\ (\nexists\, v' \in variables\_partagées(IS_i)\ v' est\ CV$

Dans cette étape de l'algorithme, un ensemble de messages est échangé entre les SS. Ces messages contiennent des informations sur le CVs et PIVs et la direction du trafic va des SSs liés aux sorties globales du système $\{s_1,\ldots,s_r\}$ vers les SSs liés aux entrées globales $\{e_1,\ldots,e_q\}$.

Par conséquent, la définition de CV est renforcée avec le nouveau concept. Un autre type de CV est une variable qui n'est pas incorrecte.

Les ADs envoient l'évaluation des sorties globales (corrects ou incorrects) aux ADLs des SSs associés à ces sorties. Ces ADLs vérifient ses variables et les marquent CVs ou IVs et envoient les états des variables partagées (CVs ou IVs selon la définition 4) et dans un message aux ADLs des SSs auxquels sont reliés à ses entrées. Après la réception de tous les messages concernant les sorties d'un SSi, l'ADLi exécute les mêmes actions : vérifie ses variables et envoi les messages. Alors, l'inférence est réalisé à partir de sorties globales vers les entrées globales du SC. Ces différents comportements de l'ADL sont décrits par les algorithmes suivants :

- ***Algorithme1 :*** Algorithme d'envoi de message aux ADLs des SSs voisins pour propager le marquage des variables partagées.

```
Algorithme envoi_ message
    Début
 /* soit SSi le sous système où cet algorithme sera exécuté.
 Pourchaque SSj relié aux entrées de SSi faire
      Si existe un sous ensemble de variables partagées Xi qui sont les entrées de
         SSi et les sorties de SSj et qui sont marquées comme PIVs
      alors Envoi le message (SSi, SSj, Xi) à ADLj de SSj
      Finsi
  Finpour
Fin
```

- ***Algorithme 2 :*** Algorithme de réception de message avec des CVs ou bien des messages avec PIVs. L'ADL exécute ensuite les deux algorithmes « vérifier_variables() » et « envoi_ message() »

Algorithme réception_message
Entrée : message (SSj, SSi, Xs)
début
Si Xs est vide **alors** Marquer tous les variables partages avec SSj comme CVs
Sinon
Marquer tous les variables partages avec SSj et n'appartienne pas à Xs comme CVs
Pourchaque variable v ∈ Xs **faire**
Si v n'est pas marquée CV **alors** marquer v comme PIV **finsi**
finpour
SI ADLi a reçu touts les messages des ADLs des SSs en relation de leurs sorties
alors
vérifier_variables()
envoi_ message()
finsi
finsi
fin

- ***Algorithme 3*** : Algorithme de marquage des variables comme CVs ou bien PIVs selon les ISs de systeme

Algorithme vérifier_variables()
Début
/* définition 4
Pourchaque variable partagée v marquée comme PIV **faire**
Si existe un composant c ∈ ISi et n'existe pas un autre ISj qui contient c avec la variable v' marquée comme CV et v'≠ v et v' ∈ sorties(ISj)
alors Marquer v comme IV
Sinon marquer v comme CV
finsi
Fin pour
Pourchaque variable de sortie v marquée comme IV **faire**
Marquer les variables en entries de ISs où v est un sortie comme PIV
Fin pour
Fin

3) Diagnostic local

Se basant sur les informations collectées durant la phase précédente, le diagnostic local est exécuté uniquement par les ADLs qui ont des VI. Le processus de diagnostic local comporte**:**

- **La génération des R-conflit :** La méthode basée sur le concept d'ensemble de conflit est à la base de la plupart des algorithmes mis en œuvre dans l'approche DX. Dans un R-conflit, au moins un des composants est défectueux par rapport aux observations, autrement dit, il est impossible que tous les composants du R-conflit se comportent normalement.
- **L'algorithme de hitting set pour le calcul des diagnostics minimaux :** En employant les R-conflits minimaux, il est possible de caractériser les diagnostics minimaux, et de fournir une base pour leur calcul. Cette caractérisation est basée sur la définition de l'ensemble minimal de candidats (ou Minimal Hitting Set).
- **La construction de la matrice de signature :** Pour simplifier la tâche de la génération des R-conflit et la construction de l'arbre de hitting set, nous utilisons une matrice de signature. Elle relie les variables partagées du SS avec les composants de l'ISs où chaque variable participe. Les lignes représentent les variables partagées Vi et les colonnes représentent les composants Ci (figure 3.18). Cette étape est réalisée une fois offline en stockant les informations précompilées. La matrice est analysé partiellement i.e. uniquement les lignes de variable marqué IV sont analysées. Les lignes correspondant aux IV représentent le hitting set.

	C1	**C2**	**…**	**Cn**
V1	1	0		0
V2	1	1		1
…				
Vm	0	1		0

Figure 3.18 Matrice de signature.

- **Diagnostic coopératif**

La figure 3.19 illustre le processus d'un diagnostic coopératif afin de savoir quels sont les composants défaillants. Il implique la prise en compte par un ADL, les contraintes des

autres agents soit par l'approche quantitative ou qualitative. Ceci est traduit par l'envoi et la réception des messages entre les ADLs. Cet échange permet à un ADL d'influer sur les connaissances d'un autre et présent le partage des résultats de calcul des diagnostics locaux entre eux.

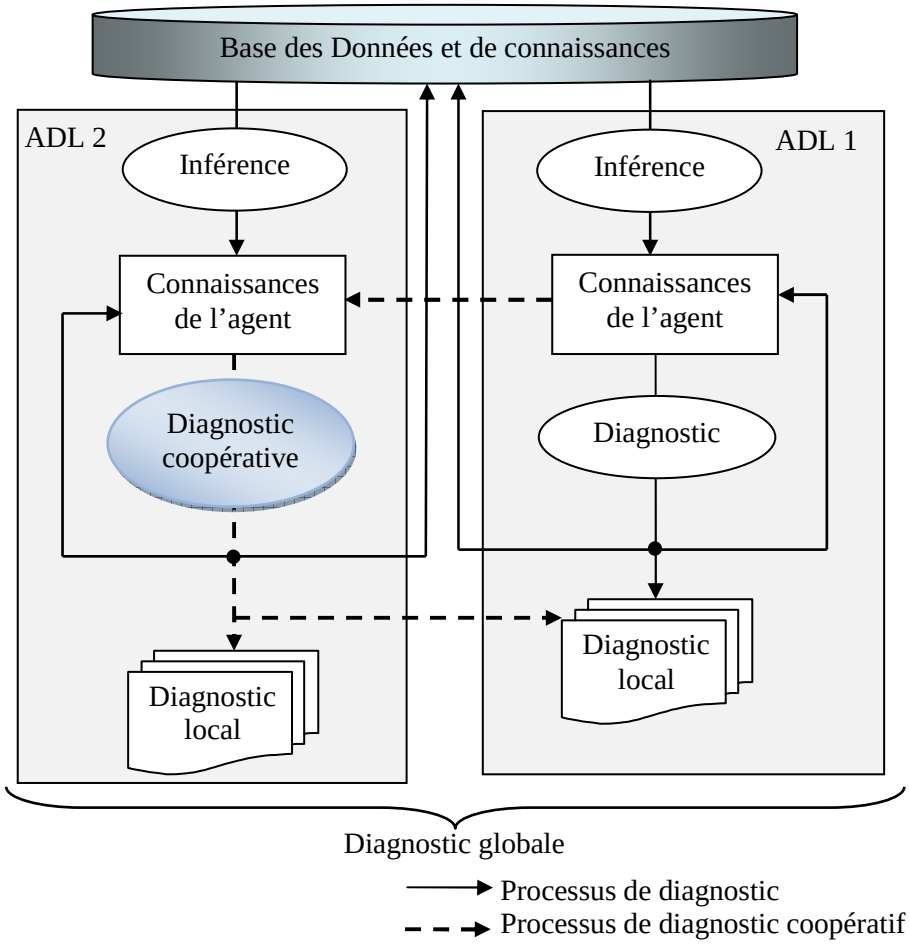


Figure 3.19 Calcul de diagnostic global à base d'agents coopératifs.

Pour montrer comment ce type de coopération peut être mis en œuvre pour développer une méthode de diagnostic coopératif, nous avons adapté la théorie de Reiter comme méthode de raisonnement diagnostic dans un contexte Multi-Agents:

Soit A={ADL_1, …, ADL_k} l'ensemble des agents de diagnostic locaux,

Soient SD_i la description du SS associé à l'agent ADL_i,

$COMPS_i$ ses composants et

OBS_i ses observations

Tel que :

$SD = \bigcup_{i=1}^{n} SD_i + STRU$, / STRU : connexion entre les SS.

$COMPS = \bigcup_{i=1}^{n} COMPS_i$,

$OBS = \bigcup_{i=1}^{n} OBS_i$.

Définition 1: $\Delta i \subseteq COMPS_i$ est un diagnostic associé à l'agent ADL_i pour $(SD_i, COMPS_i, OBS_i)$ ssi

$SD_i \cup OBS_i \cup \{\neg AB(c) \mid c \in COMPS_i - \Delta_i\} \cup \{AB(c) \mid c \in \Delta_j\} \cup \ldots \cup \{AB(c) \mid c \in \Delta_k\}$ est consistante ou plus concrètement

$SD_i \cup OBS_i \cup \{\neg AB(c) \mid c \in COMPS_i - \Delta_i\} \cup \Delta_j \ldots \cup \Delta_k$

Ainsi le diagnostic local Δ_i obtenu par l'agent ADL_i est calculé en tenant compte des résultats (diagnostics locaux) des autres agents ADL_j, …, ADL_k

Définition 2 : Soient Δ_1, Δ_2, Δ_3 …, Δ_n des diagnostics locaux associés respectivement aux agents ADL1, ADL2, ADL3, … , ADLn, alors $\Delta_g = \Delta_1 \cup \Delta_2 \cup \ldots \cup \Delta_n$ est un diagnostic global pour (SD, COMPS, OBS) tel que : $SD = \bigcup_{i=1}^{n} SD_i$, $COMPS = \bigcup_{i=1}^{n} COMPS_i$, $OBS = \bigcup_{i=1}^{n} OBS_i$.

Dans ce travail les diagnostics locaux Δi sont fusionnés par l'agent d'évaluation.

Comme nous l'avons déjà présenté, notre SMA pour SDAC se compose d'agents plus ou moins autonomes et hétérogènes, évoluant dans un environnement partagé. Ce type de conception induit de nouveaux problèmes liés directement à la vision partielle des agents, i.e. la poursuite d'objectifs locaux et le chevauchement des activités des agents. Raison pour laquelle, si nous voulons que la coopération entre agents ait un comportement global cohérent, il faut opter pour des mécanismes élaborés de coordination. Afin d'éviter tout conflit potentiel et favoriser la synergie des activités des agents, nous présentons ci après notre modèle de coordination.

3.4.4 Modèle de coordination de SDAC

La coordination est une question centrale pour les SMAs et la résolution de systèmes distribués. En effet, sans coordination un groupe d'agents peut dégénérer rapidement en une collection chaotique d'individus. Dans notre cas la coordination d'actions est nécessaire car aucun agent n'a suffisamment de compétence, de ressources et d'informations pour atteindre tout seul le but du système complet (la surveillance) ou son

propre but. Les agents ADLs ont besoin d'informations et de résultats que seul d'autres agents peuvent fournir. Par exemple, un agent ADLi qui surveille l'activité d'un procédé industriel en un point donné (SSi) aura besoin d'informations sur l'état de ce procédé en d'autres endroits (SSj).

Le modèle de coordination proposé décrit la structure organisationnelle de la société d'agent pour coordonner les comportements de ceux-ci dés la conception et limiter les conflits. Comme il utilise le Protocole d'interaction pour coordonner les différents agents, en particulier les ADLs pour les aides à atteindre un état global cohérent.

- **La structure organisationnelle**

Du point de vue de la coordination, et à un plus haut niveau d'abstraction, une organisation peut être vue comme un mécanisme de coordination qui se situe au niveau global, c'est-à-dire au niveau du réseau d'agents. Lors de la conception du SMA, les comportements des agents sont à un certain niveau d'abstraction coordonnés et réglementés par la structure qui les réunit. Dans ce type de structure, on retrouve les techniques de coordination par réglementation qui consistent à introduire des règles de comportement que les agents doivent respecter afin d'éviter des conflits potentiels. De ce fait, plusieurs conflits sont résolus a priori. L'intérêt généralement escompté de l'emploi de telles techniques est la réduction des coûts de communication entre agents et l'élimination de conflits potentiels dès la conception.

Dans la structure organisationnelle adaptée à notre situation de surveillance, la distribution des tâches est selon la dimension fonctionnelle qui consiste à répartir les rôles aux agents au sein de l'organisation en tenant compte de leurs aptitudes, de la nature des tâches qui leur incombent et des interactions possibles entre eux. La figure 3.20 illustre l'ensemble des agents impliqués durant la surveillance distribuée. Nous distingons trois niveaux: celui de l'utilisateur, l'application et enfin celui de base de données et de connaissances. Les agents SDAC sont au niveau application.

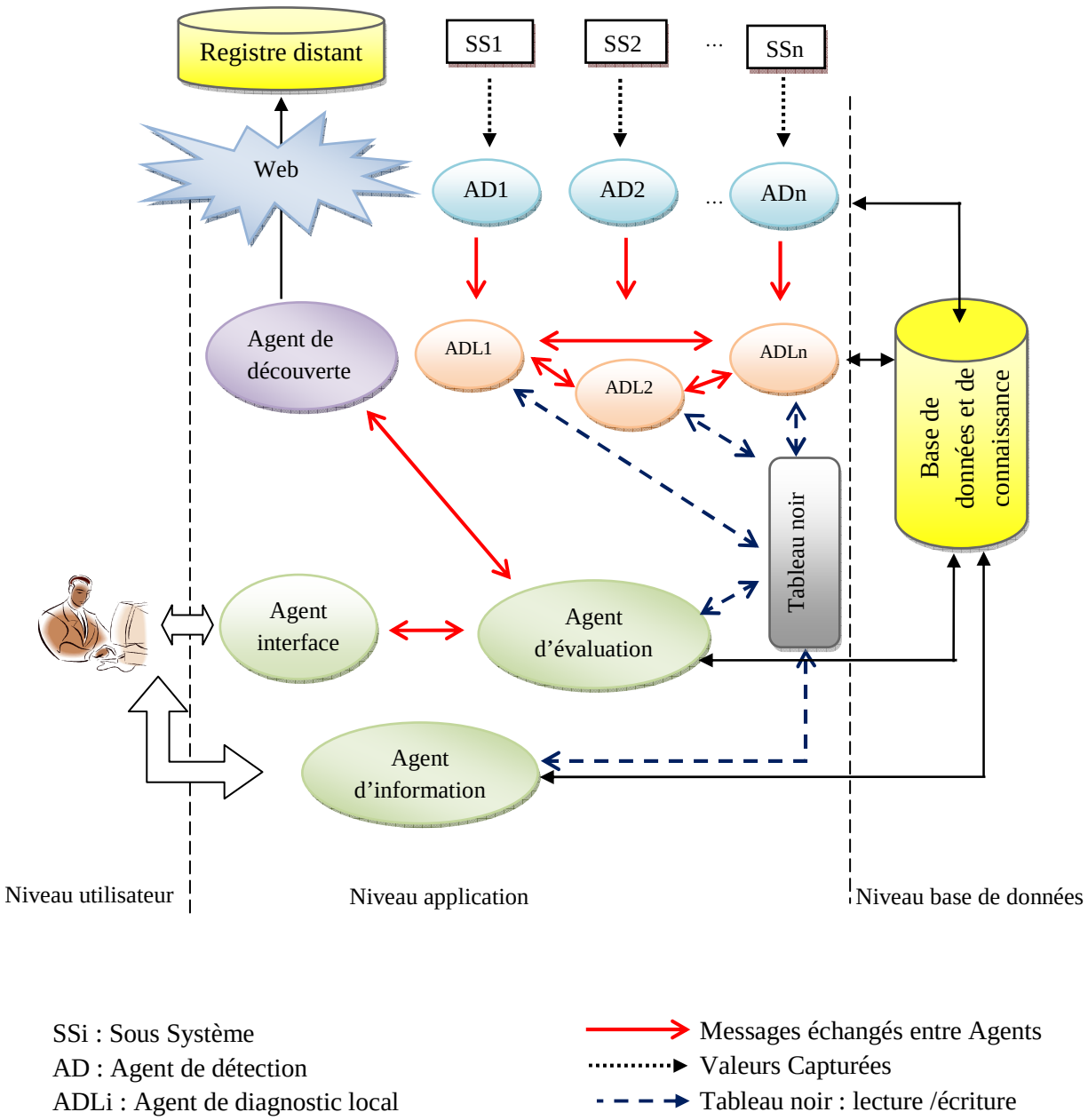


Figure 3.20 SMA pour la surveillance distribuée.

Après l'acquisition des données, les ADi envoient les résultats des tests aux ADLs. Chaque ADLi calcule le diagnostic local pour un SSi en coopérant avec les autres ADLi pour partager les informations et les résultats partiaux. Les diagnostics locaux sont fusionnés par l'agent d'évaluation. Ce diagnostic global est envoyé à l'agent d'interface pour l'afficher aux utilisateurs. La possibilité de découvrir puis d'invoquer des services web classés dans un registre distant par l'agent de découverte suite à une demande de l'agent d'évaluation pour satisfaire les besoins des ADL est détaillée dans le chapitre cinq.

- **Protocole d'interaction**

Un protocole d'interaction est un enchaînement prédéfini de messages. Les protocoles d'interaction sont introduits dans les SMA dans le but de faciliter la spécification et l'implémentation de l'interaction entre les agents.

Ainsi, le protocole d'interaction adapté à notre situation doit fournir aux agents une façon de communiquer utilement. Il est défini comme la pluparts des modèles de coordination [Papadopoulos 1998] par un triplet (A, M, R), où :

- A: représente les Agents de SDAC participants à l'interaction, particulièrement les ADL
- M: représente le Médium de communication (l'infrastructure partagée) qu'est le tableau noir, comme on peut avoir des communications directes entre agents.
- R: les Règles qui permettent de régir une interaction. Ces règles définissent l'ordonnancement des messages elles sont décrites par le diagramme de séquence de la figure 3.21.

La spécification et l'implémentation de ce protocole est indépendantes de l'approche de surveillance utilisée (quantitative ou qualitative) et de l'architecture interne de l'agent.

L'interaction inter-agents de SDAC regroupe et combine plusieurs types de messages. Pour chaque SSi affecté (valeurs incorrectes) des messages contenant les résultats de la détection de défaillance entre les ADs et ADLs doivent être échangés. Ensuite l'ADL doit échanger des messages contenant les statuts des composants structurellement liés avec les autres ADLs. L'agent d'évaluation fusionne les diagnostics locaux et évalue les résultats finaux. Si ces derniers sont satisfaisants il les doit les afficher sinon il doit appeler l'agent de découvert pour rechercher des services web adéquates

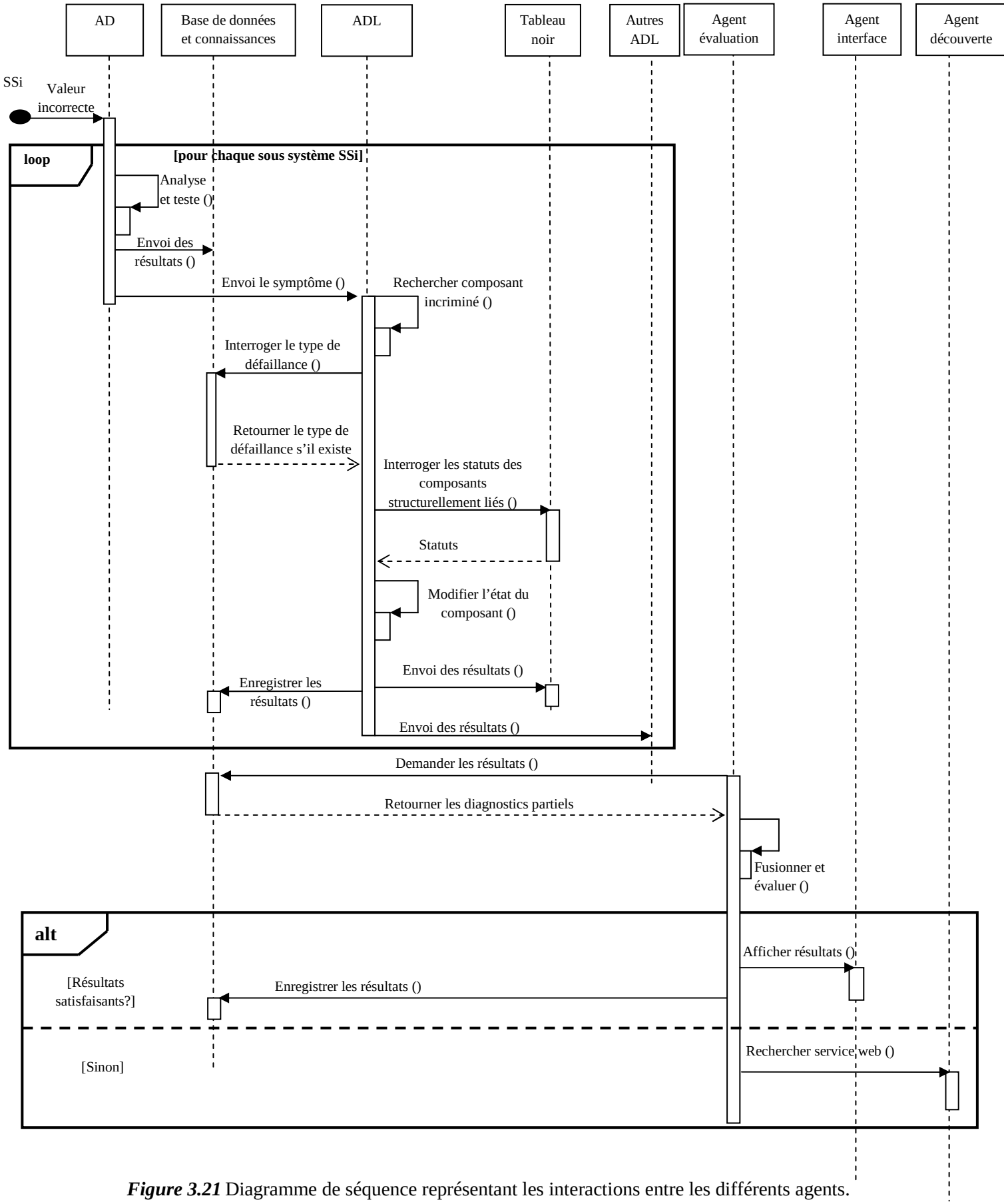


Figure 3.21 Diagramme de séquence représentant les interactions entre les différents agents.

3.4.5 Modèle conceptuel de SDAC

Les agents, les tâches et les messages représentent les objets conceptuels manipulés au cours d'une session du processus de coopération. Ils entretiennent des liens entre eux. Les caractéristiques de chacun, selon UML sont présentées par les tableaux suivants :

- ***Caractéristiques des agents***

La classe agent présente les propriétés statiques de l'agent (Nom, Type, Etat et Compétence) et ses méthodes (Rôle () et Comportement coopérative ()). Celles-ci sont expliquées par tableau 3.2.

Tableau 3.2 Explicatif des caractéristiques des agents.

Nom :	Identification de l'agent.
Type:	Cognitif s'il se base sur des démarches cognitives pour réaliser ses tâches. Réactif si l'agent se base sur des démarches prescrites pour réaliser ses tâches.
Etat :	Un agent peut être actif lorsqu'il est retenu dans le groupe ou inactif. Il peut également se rendre non disponible.
Accointances:	les noms des agents qu'il peut contacter
Compétences :	Description de ce qu'il est capable de faire d'un point de vue technique (Compétence technique) et d'un point de vue traitement réfléchie d'information (Compétence cognitive). Ces compétences peuvent être complétées par des connaissances dans la mémoire de travail.
Rôle ()	correspond aux tâches attendues de l'agent.
Comportement coopérative ()	Coopératif si l'agent participe à la coopération en vue de la résolution du problème.

- ***Caractéristiques des Tâches***

La classe tâche présente les propriétés statiques de la tâche (Nom, Type et Ressources) et sa méthodes (allocation ()). Celles-ci sont expliquées par tableau 3.3.

Tableau 3.3 Explicatif des caractéristiques des tâches.

Nom :	Identification de la tâche.
Type :	Une tâche peut être cognitive ou procédurale.
Ressources :	Liste des ressources nécessaires à la réalisation de la tâche.
Allocation ()	Identification de l'agent choisi pour réaliser la tâche.

- ***Caractéristiques des messages***

La classe message présente les propriétés statiques d'un message (Source, Destination, Numero _message, Date_apparition, Contenue) et ses méthodes (Getmessage() et Setmessage ()). Celles-ci sont expliquées par tableau 3.4.

Tableau 3.4 Explicatif des caractéristiques des messages.

Source	Le nom de l'agent qui envoi le message.
Destination	Le nom de l'agent récepteur de message.
Numero _message	Le numéro de message.
Date_apparition	La date d'apparition de message.
Contenue	Le contenue de message.
Get message ()	La réception de message.
Set message ()	L'envoi de message.

3.5 Conclusion

Dans ce chapitre, le passage du problème identifié, la surveillance distribuée à la solution informatique, suppose de franchir un certain nombre d'étapes : l'analyse, la modélisation et l'implémentation informatique. Nous nous intéresserons plus particulièrement à la méthodologie MAS-CommonKADS pour développer notre approche SADC analysant le problème de la surveillance distribuée par cinq modèles. Ensuite, nous nous sommes appuyés sur le langage UML pour représenter les différents diagrammes de ces modèles.

Nous avons donc présenté les choix des technologies utilisées pour mettre en œuvre notre approche qui tire profit des approches DX et FDI et supporte la distribution à savoir, la technologie à base d'agents coopérant et la technologie SOA à base des services web. L'objectif de SDAC est d'intégrer les agents logiciels et les services web, en une entité

cohérente qui tente de dépasser la faiblesse de chaque technologie tout en renforçant leurs avantages individuels :

- La technologie à base d'agents permet de résoudre les problèmes de l'approche centralisée. L'approche SDAC a entraîné la conception de plusieurs agents dédiés chacun à une tâche particulière. Dans notre travail, nous nous sommes intéressés spécialement à la distribution de l'étape d'analyse diagnostic. Les ADLs sont les plus importants car chaque agent ADL a une vue partielle ou locale du SS à diagnostiquer. Pour construire une solution globale, logique et cohérente il faut que les ADLs coopèrent entre eux afin de partager leurs solutions et faire part de leurs problèmes et coordonner leurs activités. Ainsi, l'objectif principal de quatrième chapitre est de présenter la première partie d'une application informatique implémentant les ADL dans un procédé industriel de fabrication du ciment.
- La technologie SOA à base des services web permet au système de surveillance d'acquérir une certaine flexibilité pour augmenter leur fiabilité. Ainsi, l'approche SDAC propose l'utilisation de l'architecture orientée-services pour la mise en œuvre d'une collaboration machine-machine pour le système de surveillance distribuée, qui offre l'interopérabilité nécessaire des composants du système. L'objectif principal du chapitre cinq et de présenter la deuxième partie de l'application informatique permettant, à base de service web, l'ouverture des systèmes de surveillance et la gestion de l'hétérogénéité de ces systèmes.

Chapitre 4

Application Informatique support à la Surveillance Distribuée à base d'Agents Coopérants: AI-SDAC

Dans ce chapitre, une Application Informatique interactive support à la Surveillance Distribuée à base d'Agents Coopérants baptisée (AI-SDAC) est développée. Elle est écrite sous JAVA et consacrée essentiellement au calcul d'un diagnostic global par des diagnostiqueurs locaux distribués.

La plateforme multi agent AI- SDAC est testée sur les données réelles de notre cas d'étude, la clinkérisation de la cimenterie d'Ain Touta.

4.1 Introduction

Le chapitre précédent a décrit notre approche baptisé SDAC. Nous y avons tout d'abord donné un point de vue macroscopique indiquant ses principales caractéristiques et ses modèles. Ces derniers ont été détaillés progressivement en développant plusieurs aspects théoriques. Afin de valider ces aspects, nous avons développé une application logicielle support à SDAC baptisé AI- SDAC (Application informatique support à la Surveillance Distribuée à base d'Agents Coopérants). Elle doit répondre à notre première motivation dans ce travail qui est la faisabilité d'un calcul de diagnostic global par des diagnostiqueurs locaux distribuées. Ainsi le diagnostic logique dans un contexte agents est appliqué à la détection de défaillance dans le système de la clinkérisation. AI- SDAC doit avoir des capacités d'évolution et d'adaptation à différents contextes i.e. l'ajout des modules à son architecture logicielle où le changement de cas d'étude reste toujours possible pour des progrès éventuels.

Par ailleurs, nous avons fait le choix, d'utiliser le langage de programmation JAVA. En effet, ce langage orienté objet assure la portabilité du code sur toute plateforme informatique, puisque ce dernier s'exécute toujours sur une machine virtuelle JAVA JVM (Java Virtual Machine) qui sera adaptée au matériel informatique (ordinateur, PDA…). Comme nous avons choisi JADE (Java Agent Development Framework); une plateforme JAVA de développement des SMA répondant aux normes FIPA pour tester le fonctionnement global du système. AI- SDAC est développée sous l'environnement de développement intégré (EDI) Netbeans intégrant JAVA et JADE.

Dans ce chapitre, après avoir présenté les modules de l'architecture logicielle d'AI- SDAC, le processus de clinkérisation, nous allons étaler notre analyse et conception de ce processus en spécifiant nos choix structurelles et fonctionnelles et dérouler le fonctionnement du modèle d'agent proposé.

4.2 Architecture logicielle

Pour répondre aux exigences de modularité et d'évolutivité, AI- SDAC est organisée en modules indépendants. Chaque module doit permettre la mise en œuvre de la surveillance coopérative d'un système de production.

Il est envisagé une conception adaptée aux spécificités des processus industriels qui demandent une interface homme-machine permettant un accès aisé:

- Aux experts pour définir les données représentant le modèle de référence
- Aux informations issues de la coopération telles que les résultats de la détection qui indique l'existence de défaillance (composant en fonctionnement normal ou anormal), les résultats de diagnostic (cause de défaillance) et l'affichage des alarmes.
- Aux techniciens pour afficher et introduire les informations des composants matérielles (intitulé, type, dates et nombre de fois des réparations….).

L'architecture d'AI- SDAC proposée prend en compte les différentes étapes du la surveillance industrielle (figure 4.1).

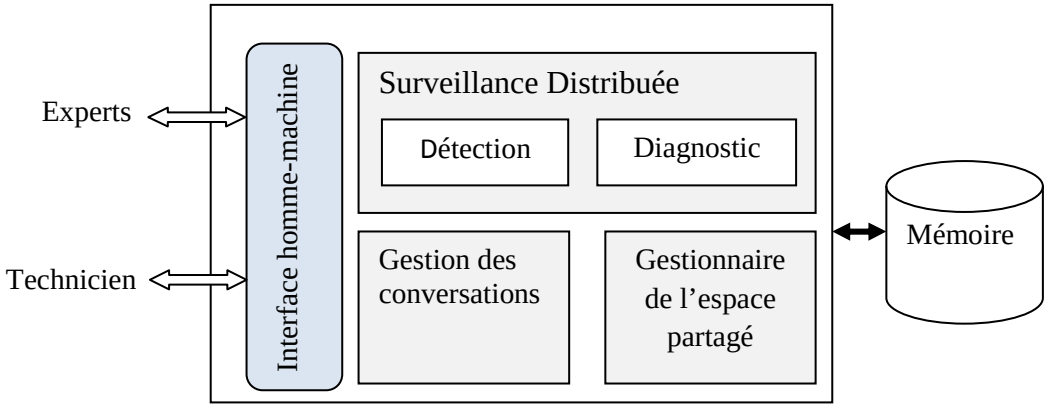


Figure 4.1 Architecture générale d'AI- SDAC.

AI- SDAC est composé de quatre modules de base et une mémoire de résultats et des conversations :

- ***Module Interface homme-machine :*** L'interface est divisée en zones aux contenus et aux fonctionnalités distinctes. Ce module permet aux utilisateurs (expert, technicien…etc.) d'accéder aux espaces d'interaction voire d'interagir avec le système.
- ***Module Gestion de l'espace partagé (ou espace d'interaction):*** L'interaction entre les différents AD et ADL est rendue possible grâce au tableau noir. Pratiquement nous avons utilisé la plateforme JADE qui propose deux solutions possibles pour réaliser un tableau noir (blackboard) :
  - La première proposition consiste à utiliser le service pages jaunes où chaque agent doit être enregistré dans la page jaune pour communiquer avec les autres agents enregistrés et l'outil JADE propose le service DF (directory Facilitator) pour faciliter les communications.

- La deuxième proposition consiste à programmer le tableau noir comme un agent avec des comportements assurant les communications entre les agents et l'échange des données. Selon les besoins rencontrés nous privilégions le second moyen. Il fait ainsi partie des briques de base de notre environnement informatique.

Le tableau noir implémenté est constitué d'un ensemble de comportements de type Cyclique (CyclicBehaviour) qui permet de communiquer avec les différents agents et rendre les variables envoyées public. Ces comportements renvoient un message indiquant que l'action de diagnostic à l'instant t est terminé pour que les ADLs passent au diagnostic à l'instant suivante $t + \Delta t$.

➢ ***Module Gestion des conversations :***

Ce module central permet d'assister les ADL lors de la phase de la coopération pour calculer les DL en garantissant la cohérence de leurs résultats pour établir le DG. Notre module de conversation permet cette coopération de la façon suivante : soit deux composants reliés par un canal (figure 4.2).

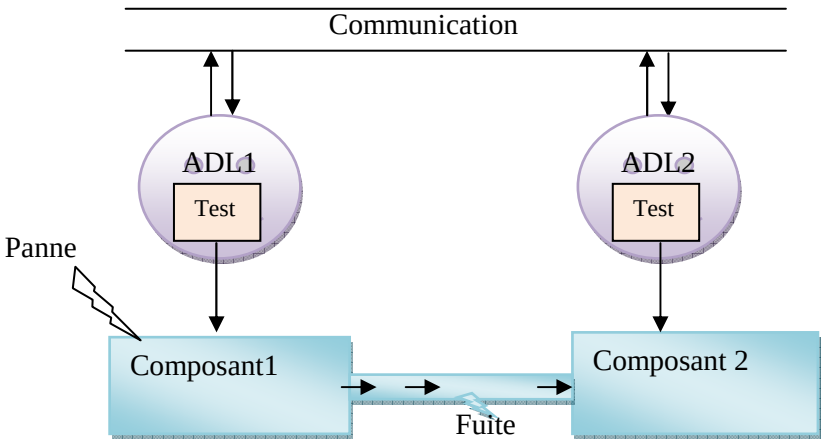


Figure 4.2 Coopération par communication.

Supposons que l'on dispose de deux types de défaillance en même temps : le composant1 est défaillant (le flux en sortie est anormal) et une fuite de canalisation entre ces deux composants. Alors ADL1 signale que le composant1 est défaillant et l'ADL2 signale que le composant2 est en mode nominal. Un appel à la maintenance est fait, le système est arrêté et le composant1 est réparé. Quand le système se remet en fonctionnement, l'ADL2 déclenche un problème de canalisation (valeur d'entrée au composant 2 est incorrecte avec une sortie correcte du composant1).

Ce module permet d'intégrer l'aspect de la distribution qui donne le droit à chaque ADL de connaitre l'état d'évolution de diagnostic pour les autres ADLs. Ce module tire parti de l'architecture dans laquelle il s'intègre et notamment du «tableau noir», qui joue le rôle d'espace partagé par les ADLs et dont ceux-ci se servent pour publier leurs variables partagées.

- ***Module Surveillance Distribuée :***

Ce module permet de donner l'état exact du système de production industriel, qui permettra de prendre les décisions les plus pertinentes possibles. Il s'agit de détecter tous les comportements anormaux ou nominaux et d'effectuer un diagnostic. Afin de répondre à ces objectifs le module surveillance distribuée est décomposé en deux sous modules: détection et diagnostic suivants:

- ***Module Détection :*** Le rôle principal de ce module est de détecter des symptômes grâce aux agents de détection et une base de connaissance qui définit le mode de bon fonctionnement du composant. La détection suit les deux approches quantitative et qualitative présentées dans le chapitre précédent.
- ***Module Diagnostic :*** Le module diagnostic permet d'estimer l'état de santé des composants basé sur les résultants de la détection selon les deux approches également.

4.3 Application industrielle

Notre approche est appliquée à la surveillance dans le système de la clinkérisation de la cimenterie d'Ain Touta. Le procédé industriel est détaillé dans [Mouss 2006] et [Demagh 2013].

4.3.1 Brève présentation de la SCIMAT

Le système étudié est un procédé industriel de fabrication du ciment. Cette installation fait partie de la cimenterie d'Ain-Touta (SCIMAT). Elle a une capacité de 2.500.000 t/an (2 fours) est composée de plusieurs unités qui déterminent les différentes phases du processus de fabrication du ciment (figure 4.3) [Kadri 2013].

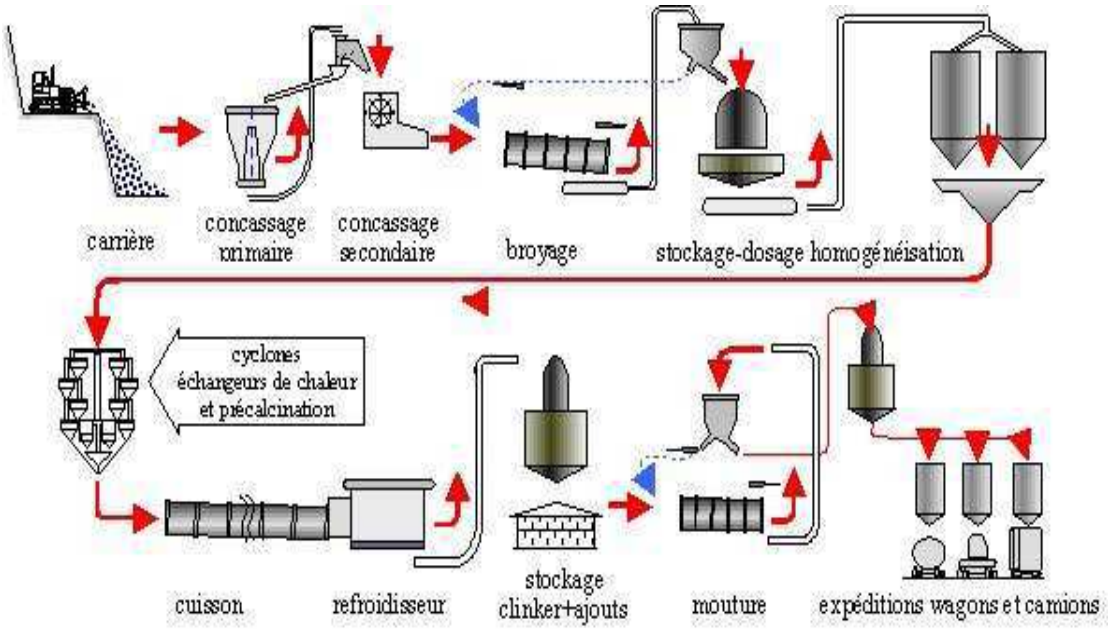
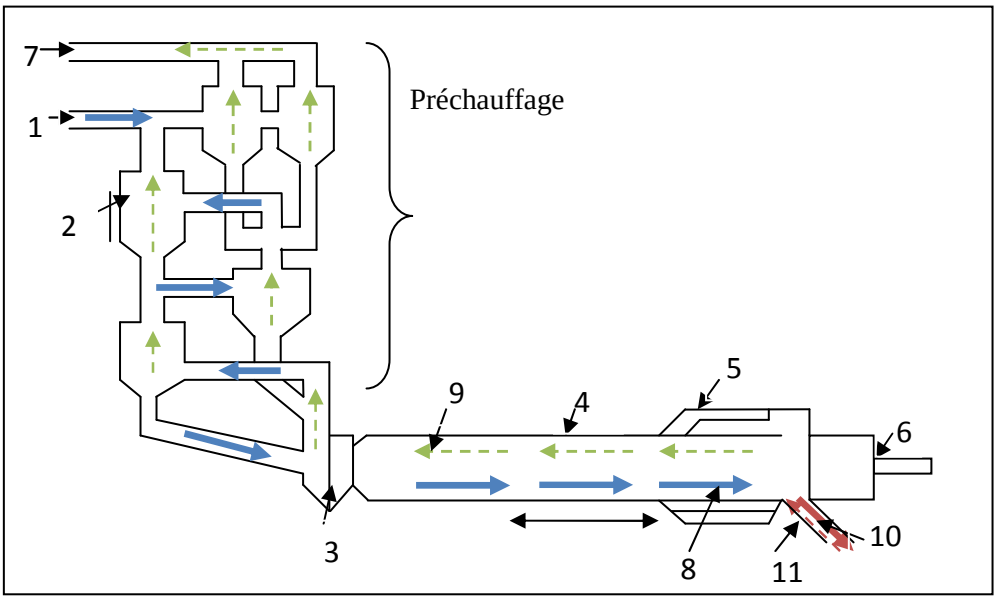


Figure 4.3 Principe de fabrication du ciment.

Chacune des opérations impliquées dans le processus de fabrication du ciment est importante et doit être correcte pour avoir la qualité exigée pour son emploi. Cependant et selon plusieurs chercheurs, l'opération de cuisson ou la clinkérisation est la plus importante.

4.3.2 Description du processus de clinkérisation

Dans tous les systèmes de fours, les matières premières sont soumises au même traitement : réchauffement, calcination, cuisson et refroidissement. L'atelier de cuisson représente la partie centrale de la cimenterie (figure 4.4). Il est composé de deux lignes de cuisson en tout point identiques et complètement indépendantes du point de vue fonctionnement. Une ligne est constituée de trois échangeurs : le préchauffeur à cyclone, le four rotatif et le refroidisseur à ballonnets.



1	Alimentation farine	7	Vers le broyeur sécheur et dépoussiérage.
2	Préchauffeur à cyclones	8	Farine cru
3	Boite à fumée	9	Gaz chauds
4	Four rotatif	10	Clinker
5	Refroidisseur à ballonnets	11	Air secondaire
6	Alimentation gaz		

Figure 4.4 Schéma synoptique de l'atelier de clinkérisation.

4.3.2.1 Le préchauffeur à cyclones

Le cru, séché, broyé et homogénéisé est introduit sous forme pulvérulente dans une tour de préchauffage à cyclones. Cet échangeur gaz/matière reçoit la matière par son extrémité haute et réalise la décarbonatation partielle de la farine crue (25% à 30%) avant d'entrer dans le four. Donc il sert à améliorer le rendement thermique de l'installation de cuisson. Chaque ligne de préchauffeur présente des cyclones jumelés étage, afin d'assurer une séparation efficace de la farine crue gaz de sortie. Les cyclones simples du deuxième, troisième et quatrième étage sont généralement de même taille. La farine crue introduite contient encore 1% d'humidité et le courant gazeux dont la température est d'environ 400°C sert au séchage de la farine crue. Séparée des gaz à chaque traversée de cyclone, la matière est à chaque fois réinsérée à l'étage suivant (au-dessous) et reprise par des gaz de plus en plus chauds. Ainsi elle se réchauffe.

Au niveau du 4ème cyclone et de la boite à fumée, vers 950° commence la décarbonatation. C'est le phénomène chimique par lequel le carbonate de calcium se décompose en chaux et en gaz carbonique.

La chaux ainsi libérée est prête à réagir avec les autres éléments (silice, alumine et fer). La gaz carbonique s'échappe et part avec les gaz brulés, faisant ainsi perdre à la matière environ 35% de son poids.

4.3.2.2 Processus dans les fours de cimenterie

Le four rotatif est le centre du processus de cuisson, il représente l'organe le plus sollicité thermiquement et inclut des apports d'énergie de grandes capacités et de grandes déperditions thermiques. Le four est constitué d'un cylindre d'acier de 68 m de longueur effective et de 4.6 m de diamètre. Ce cylindre appelé aussi virole est revêtu à l'intérieur de briques réfractaires qui protègent les tôles des températures élevées (1850°C pour les gaz et 1450°C pour la matière). Il tourne sur lui-même à vitesse réglable lent (0.67 a 2 tr / min). Il présente une inclinaison de 3% dans le sens de l'écoulement de la matière, qui combinée avec la rotation fait que la matière puisse progresser par gravité, vers le brûleur (figure 4.5).

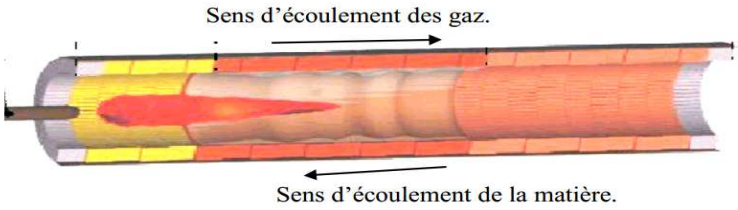


Figure 4.5 Four en coupe.

Le four est le siège d'un équilibre entre deux débits opposés:

- Un débit calorifique des fumées de combustion : L'air secondaire provenant de l'aval du four (côté bruleur) déjà chauffé, se surchauffe en passant à travers la flamme dégagée du bruleur.

- Un débit de matière d'amont en aval: La matière, partiellement décarbonatée, arrive à l'amont du four (zone de décarbonatation) avec une température de l'ordre de 800 °C à 900 °C et devrait atteindre la température de clinkerisation proche de 1450°C à l'aval du four (zone de clinkerisation).

La batterie de ventilateurs, placée à l'extérieur du four côté talus, va refroidir la virole qui par conduction thermique refroidira la matière en contact avec la paroi interne et fera que la matière soudainement refroidie puisse coller au réfractaire et ainsi le protéger de l'abrasion du clinker porté, on parlera alors de constitution d'une couche de croûtage par attachement (figure 4.6). Le croûtage revêt une importance capitale dans le processus de transformation, en plus de son rôle d'isolant thermique, il va protéger le garnissage réfractaire contre l'attaque chimique du clinker en fusion dans cette zone dite de cuisson, zone la plus chaude dans le four. Également, sa présence à des épaisseurs bien précises améliore le mélange de la matière à l'intérieur et favorise ainsi une cuisson homogène du produit.

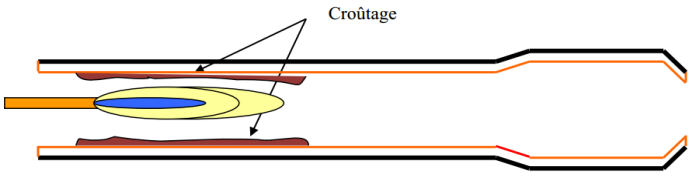


Figure 4.6 Constitution du croûtage dans le four.

4.3.2.3 Le refroidisseur

Il refroidit le clinker sortant du four et récupère de la chaleur. Il agit sur le rendement thermique de l'installation de cuisson et sur la qualité de clinker. L'installation de cuisson d'Ain-Touta est dotée d'un refroidisseur à ballonnets. Ce sont des tubes en tôle d'acier de 19.8 m de long et de 2.1 m de diamètre munis de releveurs. Le clinker est refroidi au contact de l'air frais injecté dans les tubes. On obtient ainsi des grains solides à une température entre 100 et 200°C ensuite le clinker est transporté vers les silos de stockages. L'aire de refroidissement est injectée dans le four comme air secondaire dans le processus de combustion.

4.3.3 Analyse et conception de la clinkérisation

La clinkérisation est structuré en sous-système. Chaque SS est implémenté par plusieurs fonctions. Chaque fonction peut être décomposée en fonctions élémentaires et assurée par plusieurs composants. Certains composants peuvent être utilisés à l'implémentation de plusieurs fonctions. Le diagramme de classe de la figure 4.7 représente ses choix aux niveaux structurel et fonctionnel.

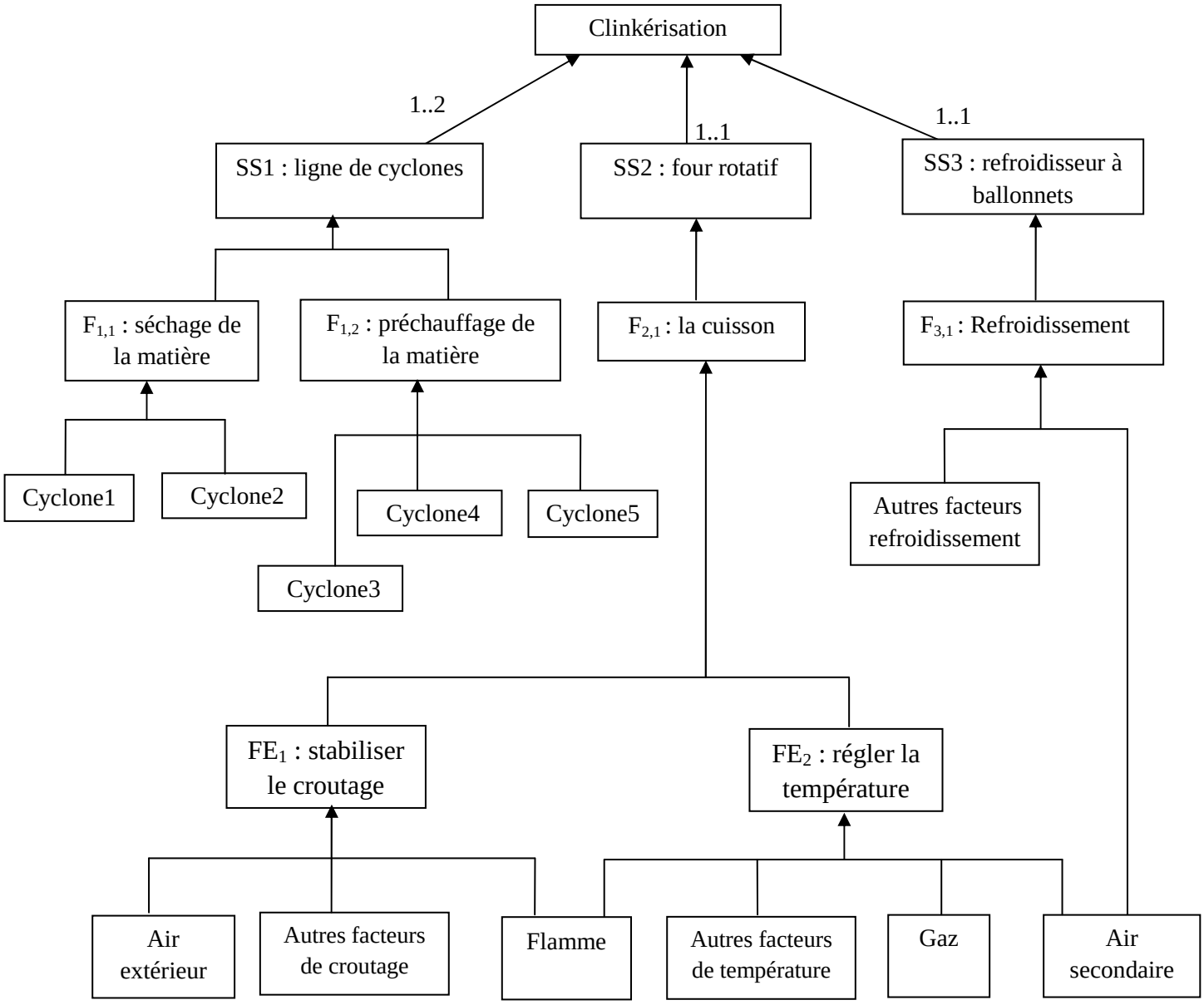


Figure 4.7 Diagramme de classe de la clinkérisation.

Compte tenu du grand nombre de variables au niveau de la clinkérisation et d'après la description du fonctionnement de ce système complexe, nous avons essayé de faire une étude sur certaines variables afin d'évaluer notre approche SDAC. L'analyse fonctionnelle a conduit à :

- découper notre système en trois sous-systèmes (SSi)
- choisir pour chaque SS un nombre réduit de fonctions élémentaires
- choisir un nombre réduit de composants pour implémenter chaque fonction et les composants qui ne sont pas inclus dans notre étude indiqué par « Autres facteurs FEi »

Ainsi, les SSi sont :

SS1: ligne de cyclones

Composé par cinq cyclones à des températures bien déterminés, les deux premiers assurent la fonction de séchage de la matière noté $F_{1,1}$ et les autres implémentent la fonction de préchauffage de la farine crue noté $F_{1,2}$.

SS2: four rotatif

Une fois la farine est préchauffée, elle passe dans le four. La fonction de cuisson est découpée en deux fonctions élémentaires (FEi) :

- *FE1 : Stabiliser le croutage*

  Cette variable varie en fonction de la flamme (diamètre, longueur) générée par le bruleur, l'air extérieur produit par le ventilateur principal et d'autres facteurs de croutage.
- *FE2 : Régler la température*

  Cette variable varie en fonction de la pression de gaz, la flamme (diamètre, longueur), la température de l'air secondaire (air de refroidissement) injecté et d'autres facteurs de température.

SS3: refroidisseur à ballonnets

La fonction de refroidissement notée $F_{3,1}$ est fonction de la température de l'air secondaire et d'autres facteurs de refroidissement.

4.3.4 Application à la ligne de cyclones

Le système de préchauffage est décomposé en cinq SS. Chaque SS est composé d'un seul cyclone.

4.3.4.1 Analyse du système préchauffeur à cyclones

Selon la théorie de Reiter nous pouvons ainsi définir le système de préchauffage à cyclones :

COMPS = /* les composants de système.

(cyclone 1, cyclone 2, cyclone 3, cyclone 4, cyclone 5)

DS= { /* Modèle de bon comportement.

Sortie (cyclone 1) = Entrée (cyclone 1) + V1,

Sortie (cyclone 2) = Entrée (cyclone 2) + V2,

Sortie (cyclone 3) = Entrée (cyclone 3) + V3,

Sortie (cyclone 4) = Entrée (cyclone 4) + V4,

Sortie (cyclone 5) = Entrée (cyclone 5) + V5,

V*i* : valeur ajoutée par cyclone *i*

/* Connexions notées STRU

Sortie (cyclone 1) = Entrée (cyclone 3),

Sortie (cyclone 2) = Entrée (cyclone 3),

Sortie (cyclone 3) = Entrée (cyclone 4),

Sortie (cyclone 4) = Entrée (cyclone 5),

Sortie (cyclon5) = Entrée (Four),}

OBS= {/* Valeurs mesurées.

Entrée (cyclone 1) = 30, Sortie (cyclone 1) = 339,

Entrée (cyclone 2) = 40, Sortie (cyclone 2) = 345,

Entrée (cyclone 3) = 345, Sortie (cyclone 3) = 735,

Entrée (cyclone 4) = 725, Sortie (cyclone 4) = 872,

Entrée (cyclone 5) = 863, Sortie (cyclone 5) = 1210,}

4.3.4.2 Fonctionnement de modèle d'agent proposé

La figure 4.8 présente les SSs, l'organisation de la société d'agents associés et les interactions entre ceux-ci. Ainsi, le SMA que nous avons proposé pour la surveillance distribuée de la ligne de cyclones comporte les agents suivants:

- *Agent de Détection (AD)*, un agent pour l'entrée et un agent pour la sortie du chaque cyclone donc dix agents de détection,
- *Agent de Diagnostic Local(ADL)*, un agent par cyclone. Ces agents coopèrent via le tableau noir pour calculer des DLs en garantissant la cohérence de leurs résultats pour établir le DG.
- *agent d'évaluation* pour la fusion des DL. Cet agent organise la procédure de diagnostic entre l'instant *t* et *t*+Δ*t* et fusionner les diagnostics locaux, ensuite il fait appel à l'agent interface.
- *l'agent interface* : affiche les résultats.

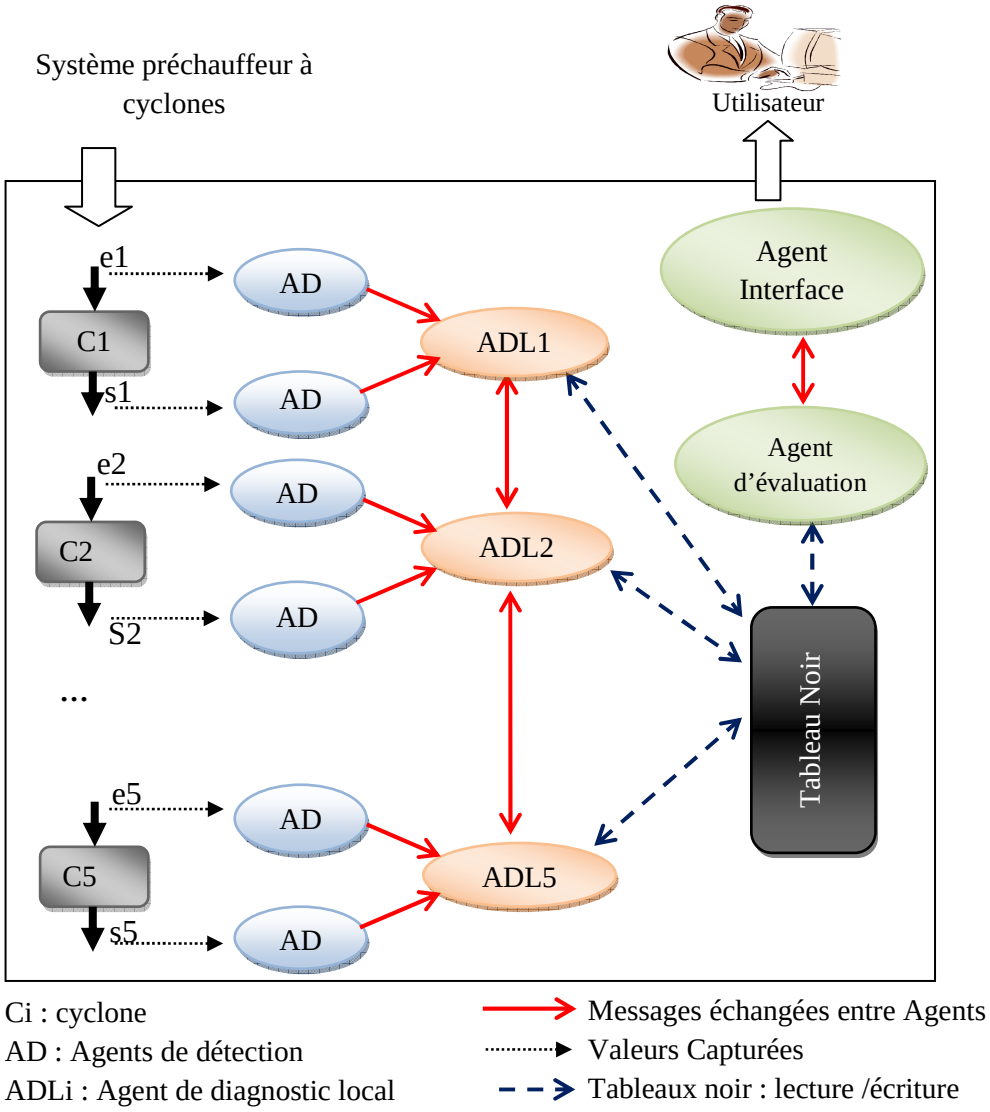


Figure 4.8 SMA pour la surveillance distribuée de la ligne préchauffeur à cyclones.

L'approche quantitative proposée, présentée dans le chapitre 3 §3.3.2.3 est appliquée pour la détection et le diagnostic des défaillances et définit les comportements des AD et ADL.

Chaque AD acquit l'entrée (ei) / sortie (si) de chaque cyclone. Si la valeur de ei/si est incorrecte il envoi un message à l'ADL associé à ce cyclone. Chaque ADLi récupère les valeurs d'ei et si de son cyclone et demande l'état de la sortie de cyclone précédent pour calculer le DL. Les Cyclone1 et 2 sont du même niveau, c'est pourquoi l'ADL3 demande à l'ADL1 la sortie s1 et demande à l'ADL2 s2 pour qu'il puisse évaluer l'état de Cyclone3. La communication directe ou indirecte permet aux ADLs le partage des données et la coopération pour calculer les DLs. Le diagramme de séquence de la figure 4.9 montre cette coopération.

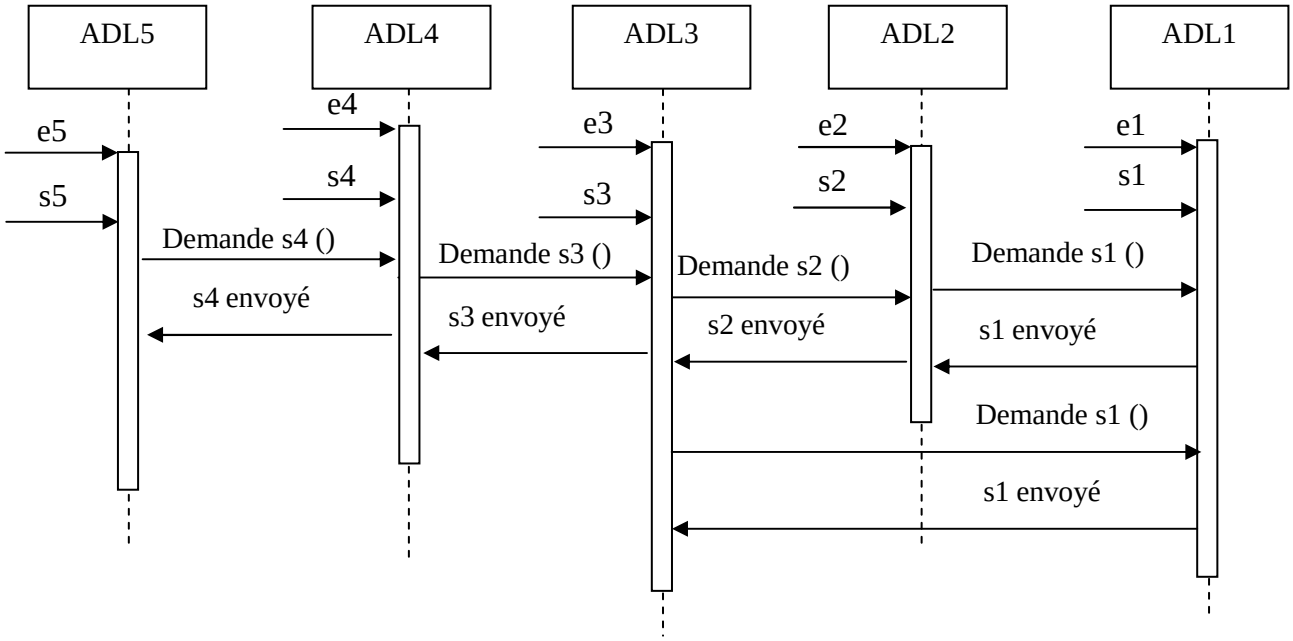


Figure 4.9 Coopération des ADL de la ligne préchauffeur à cyclones.

4.3.5 Application au four rotatif et au refroidisseur à ballonnets

Selon le diagramme de classe de la figure 4.7 le four rotatif et le refroidisseur sont deux SSs différents et le four rotatif est décomposé à son tour en deux SSs physiques. Nous avons donc trois SSs. Chacun est composé par un ou plusieurs composants et implémente une seule fonction : SS_1 pour stabiliser le croutage, SS_2 pour régler la température et SS_3 pour le refroidissement.

4.3.5.1 Analyse du four rotatif et du refroidisseur à ballonnets

Compte tenu du grand nombre des variables au niveau du four et du refroidisseur, nous avons sélectionné un ensemble réduit de variables (figure 4.10) et nous utilisons les notations suivantes:

G : Gaz.
F : Flamme,
A : Air secondaire,
AE : Air extérieur
TE : déviation de la TEmpérature,
CR : instabilité de CRoutage,
RE : problème de REfroidissement.

C : autres facteurs de Croutage
T : autres facteurs de Température
R : autres facteurs de Refroidissement

Selon la théorie de Reiter, nous pouvons ainsi définir les SS du four rotatif et du refroidisseur à ballonnets:

COMPS = /* les composants de système.

(G, A, AE, F, C, T, R)

DS= { /* Modèle de bon comportement.

TE= F (G, F, A), CR= F (AE, F), RE= F (A)}

OBS= {/* Valeurs mesurées : TE, CR, RE}

4.3.5.2 Fonctionnement du modèle d'agent proposé

L'approche qualitative proposée est appliquée pour la détection et le diagnostic des défaillances et définit les comportements des AD et ADL. La figure 4.10 représente SS_1, SS_2 du four rotatif et SS_3 du refroidisseur et le modèle SMA associés :

- *Agent de Détection (AD),* un agent pour la sortie de chaque SS donc trois AD,
- *Agent de Diagnostic Locale(ADL),* un agent pour chaque SS donc trois ADL. Ces agents coopèrent via le *tableau noir* et par envoi de message pour calculer des DLs en garantissant la cohérence de leurs résultats pour établir le DG.
- *agent d'évaluation* pour la fusion des DLs. Cet agent organise la procédure de diagnostic entre l'instant *t* et l'instant $t + \Delta t$ et fusionne les DLs. Il fait appel ensuite à l'agent interface.
- *l'agent interface* : affiche les résultats.

Chaque ADi acquit la sortie globale d'un SSi. Si la valeur est incorrecte il envoi un message à l'ADL associé à ce SS. Le comportement de l'ADL suit l'algorithme distribué développé par l'approche qualitative. Donc, on a besoin d'une analyse sur les variables partagées pour définir les groupes de composants locaux (IS). L'ADL partage les états de ces variables avec les autres ADLs par envoie de messages, c'est l'étape de la propagation. Après la collecte des informations de l'étape précédente, chaque ADL calcul le DL au niveau SS ensuite il coopère avec les autres ADLs pour partager les résultats partiels et calculer le diagnostic niveau global.

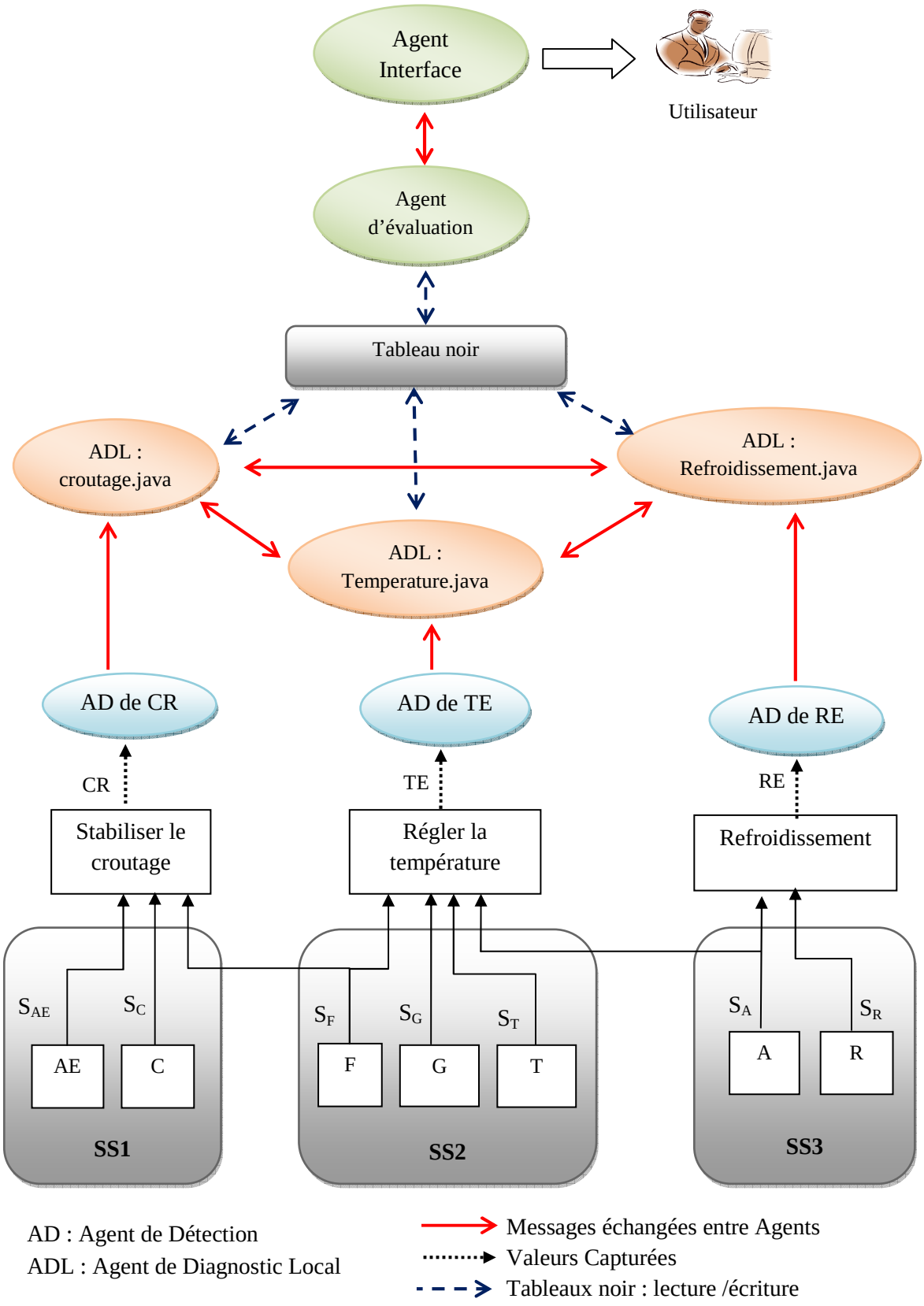


Figure 4.10 SMA pour la surveillance distribuée du four rotatif et du refroidisseur à ballonnets.

4.3.5.3 Diagnostic niveau local

Selon la figure 4.10, nous avons trois sous systèmes avec leurs variables respectives:

- SS1 avec la variable globale en sortie CR et S_F comme variable partagée.
- SS2 avec la variable globale en sortie TE et S_A comme variable partagée.
- SS3 avec la variable globale en sortie RE.

Nous avons associé à chaque SS un ADL développé en Java :

- SS1 avec ADL nommé croutage.java
- SS2 ADL nommé Temperature.java
- SS3 ADL nommé Refroidissement.java

Etape1: Définition des groupes de composants locaux (IS)

Étape Offline et selon la définition d'un IS chaque SS peut avoir un ou plusieurs IS. Dans ce cas les trois IS des trois sous systèmes sont :

- SS1 avec IS= {AE, C}.
- SS2 avec IS= {F, G, T}.
- SS3 avec IS= {A, R}.

Etape 2: Propagation

Étape d'envoi de messages contenant les états des variables partagées entre les ADL. Nous avons uniquement deux variables partagées S_F et S_A donc l'ADL Temperature.java associé au SS2 doit envyer l'état de la variable S_F à ADL croutage.java associé au SS1 comme il demande l'état de la variable S_A à ADL Refroidissement.java (figure 4.11).

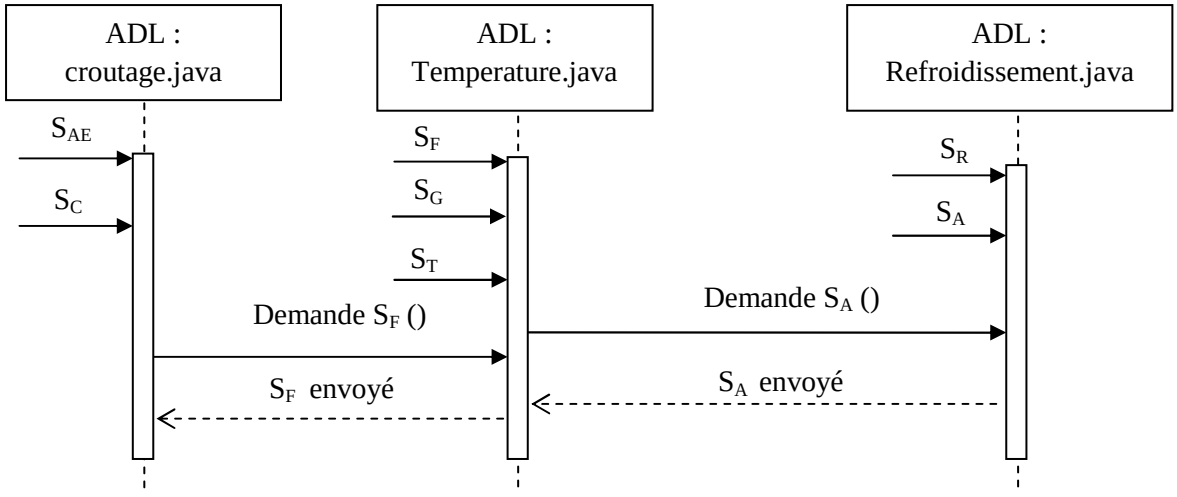


Figure 4.11 Coopération des ADLs de four rotatif et refroidisseur à ballonnets.

Etape 3 : Diagnostic local

Chaque SSi implémente un nombre très réduit de composants ce qui rend le calcul de diagnostic local minimal simple dans le cas d'une seule sortie défaillante et sans considération des autres ADLs (tableau 4.1):

Tableau 4.1 Les DLs minimaux sans coopération.

Observations	Sous système	Diagnostic local minimal
S(CR)=0, S(TE)=1, S(RE)=1	SS1	AE, C
S(CR)=1, S(TE)=0, S(RE)=1	SS2	F, G, T
S(CR)=1, S(TE)=1, S(RE)=0	SS3	R, A

(1: sortie normale, 0: sortie anormale)

Selon le schéma de la figure 4.10, nous remarquons une interrelation entre les différentes variables, engendre un niveau élevé de complexité. Ainsi, ces trois ADLs qui contrôlent l'état de refroidissement, la stabilité du croutage et la déviation de la température doivent coopérés pour trouver les combinaisons possibles des équipements défaillants présentées par le tableau 4.4 où son calcul est détaillé par la section suivante.

4.3.5.4 Diagnostic niveau global

Si nous cherchons le diagnostic de la configuration de la figure 4.10, nous trouvons $2^{nombre\ de\ composants}$ diagnostics possibles. Nous somme contraint d'appliquer un principe fondé sur la notion de diagnostic minimal.

Pour aboutir au diagnostic minimal, il faut d'abord identifier les observations, puis un ensemble de R-conflits minimaux. Nous utilisons ensuite la méthode de Hitting set (l'ensemble d'échantillon) pour définir les diagnostics minimaux possibles.

- ***Les observations possibles***

Les valeurs mesurées sont les sorties globales des sous systèmes: TE, CR, RE (tableau 4 .2).

Tableau 4.2 Les observations possibles.

Observations	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
S(CR)	1	1	1	1	0	0	0	0
S(TE)	1	1	0	0	1	1	0	0
S(RE)	1	0	1	0	1	0	1	0

(1 : sortie normale, 0: sortie anormale)

- ***Génération des R-conflits***

Un R-conflit peut être interprété par le fait que l'un au moins de ses composants est défaillant. Un R-conflit est minimal s'il n'inclus aucun autre R-conflit.

Exemple : L'observation 2 tel que : S(CR) =1. S(TE)= 1. S(RE)= 0.

Considérons dans un premier temps l'observation de la sortie refroidissement S(RE)=0. Cette observation n'est pas cohérente avec la prédiction du modèle. La valeur de cette sortie ne peut être expliquée que si au moins une anomalie au niveau de l'air secondaire ou au niveau de refroidissement (autre variables). On conclut que {A, R} constitue un R-conflit minimal.

Considérons maintenant l'observation de la température S(TE) =1. Celle-ci est en accord avec la prédiction faite par le modèle. Cependant, compte-tenu de la symétrie de la configuration étudiée, cette observation met en évidence un autre R-conflit minimal.

En effet, si les composants, capteur de pression gaz, capteur de température, bruleur et autres composants de refroidissement (sauf le ventilateur de l'air secondaire) fonctionnent correctement, nous ne pouvons observer simultanément S(RE)=0 et S(TE)=1 (ces deux observations devraient être identique quelque soit l'état de marche du ventilateur de l'air secondaire). En conclusion nous pouvons dire que {R, T, G, F} forme un second R-conflit minimal. De la même façon s'il y a un problème au niveau de la flamme (F), on tombe dans le problème précédent de symétrie de configuration et on conclut un autre R-conflit minimal {R, T, G, C, AE}. Selon toutes les observations possibles relatives à cette démarche R-conflit minimal sont présentées sur le tableau 4.3

Tableau 4.3 Les conflits minimaux.

	Conflits minimaux
Observation 1	∅
Observation 2	{R, A}, {R, T, G, F}, {R, T, G, C, AE}
Observation 3	{T, F, G, A}, {R, T, G, F}, {T, C, AE, G, A}
Observation 4	{T, F, G, A}, {R, A}, {T, G, A, C, AE}
Observation 5	{C, AE, F}, {C, AE, T, G, A}, {C, AE, T, G, R}
Observation 6	{C, AE, F}, {R, A}, {C, AE, T, G, A}, {R, T, G, F}
Observation 7	{C, AE, F}, {T, F, G, A}, {T, F, G, R}
Observation 8	{C, AE, F}, {T, F, G, A}, {A, R}

- ***Les diagnostics minimaux***

L'analyse des R-conflits minimaux permet d'engendrer tous les diagnostics minimaux d'un système observé. Cette analyse s'appuie sur la notion de Hitting set.

Pour l'observation 2 tel que : S(CR) =1, S(TE)= 1, S(RE)= 0 et à partir des conflits {R, A}, {R, T, G, F}, {R, T, G, C, AE} nous pouvons engendrer les cinq diagnostics minimaux possibles {R}, {A, T}, {A, G}, {A, F, C}, {A, F, AE}. Cela signifie que les observations effectuées peuvent être expliquées par une situation de défaut simple {R} et deux situations de défauts doubles {A, T}, {A, G}, et deux situations de défauts triples {A, F, C}, {A, F, AE}.

Considérons maintenant les huit jeux d'observations distincts de sortie de système et leurs R-conflits. Nous pouvons déduire directement en utilisant la méthode précédemment décrite les diagnostics minimaux correspondants (tableau 4.4)

Tableau 4.4 Les diagnostics minimaux.

	Diagnostics minimaux
Observation 1	∅
Observation 2	{R}, {A, T}, {A, G}, {A, F, C}, {A, F, AE}
Observation 3	{T}, {G}, {F, C}, {F, AE}, {A, R}
Observation 4	{A}, {T, R}, {G, R}, {F, R, AE}, {F, R, C}
Observation 5	{AE}, {T, F}, {G, F}, {F, A, R}
Observation 6	{C, R}, {C, A, T}, {C, A, G}, {AE, R}, {AE, A, T}, {AE, A, G}
Observation 7	{F}, {C, T}, {C, G}, {C, A, R}, {AE, T}, {AE, G}, {AE, A, R}
Observation 8	{C, T, R}, {C, G, R}, {C, A}, {AE, T, R}, {AE, G, R}, {AE, A}, {F, R}

4.3.6 Plateforme SMA pour la surveillance

La plateforme SMA développée intègre les deux approches quantitative et qualitative appliquées au niveau des sous systèmes préchauffeur à cyclone, four rotatif et refroidisseur à ballonnets. L'application développée propose un espace qui contient des champs pour entrer les valeurs mesurées (l'ensemble des observation) pour chaque cyclone (e1,…e5, s1,…S2) et un espace pour entrer les variables de sortie globales du four et le refroidisseur (TE, CR, RE). L'interface principale de la plateforme présentée par la figure 4.12 propose aussi deux boutons l'un pour lancer le diagnostic «diagnosis » et l'autre pour la « configuration» du mode de bon fonctionnement par un expert. Elle permet aussi d'activer JADE en appuyant sur l'option « RMA agent activate» de l'interface pour gérer les différents agents.

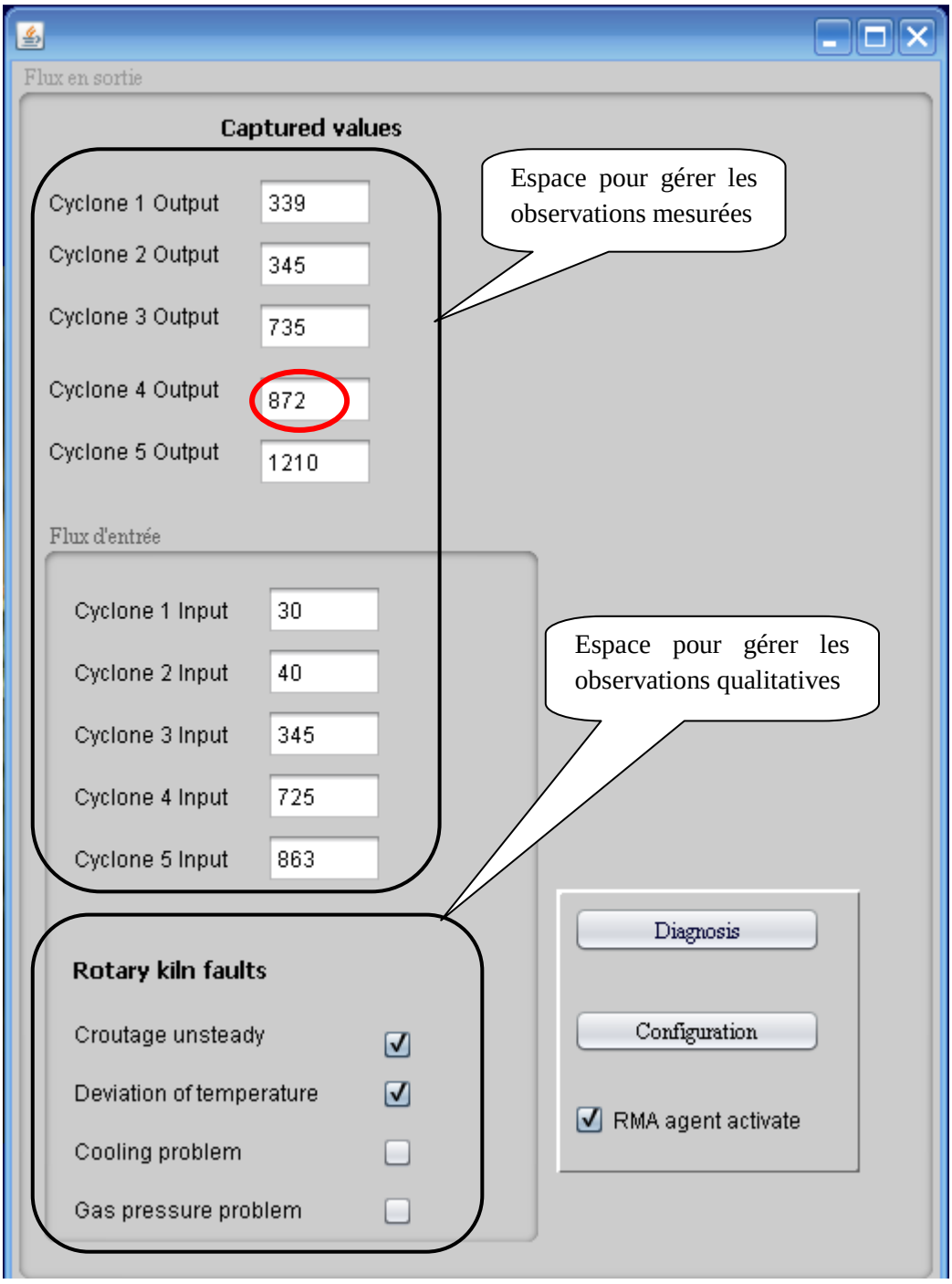


Figure 4.12 Espace pour la saisie des observations.

En appuyant sur le bouton de configuration, un espace privé est réservé à l'expert du domaine qui peut mettre à jour la configuration du modèle de bon fonctionnement via l'interface suivante (figure 4.13):

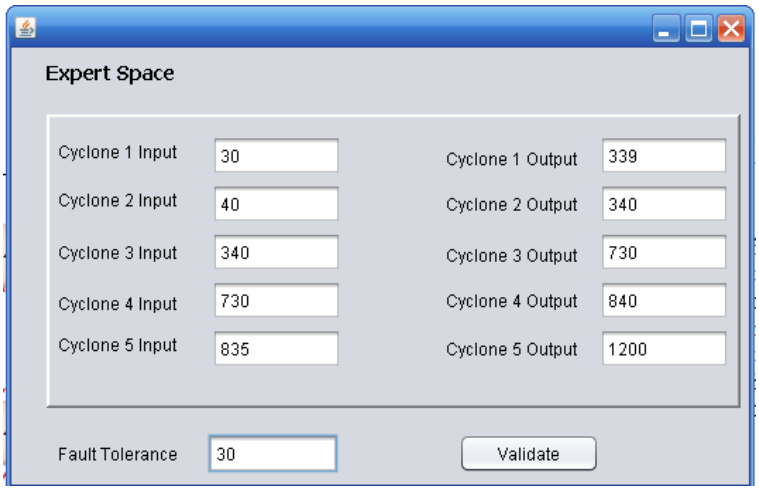


Figure 4.13 Espace de l'expert pour la configuration.

Après la validation de la mise à jour de configuration, nous revenons à la fenêtre principale de la plateforme représentée par la figure 4.12. Nous remarquons que la sortie de cyclone 4 est erronée par rapport aux valeurs du modèle. En cliquant sur le bouton « diagnosis » les résultats sont affichés sur la figure 4.14.

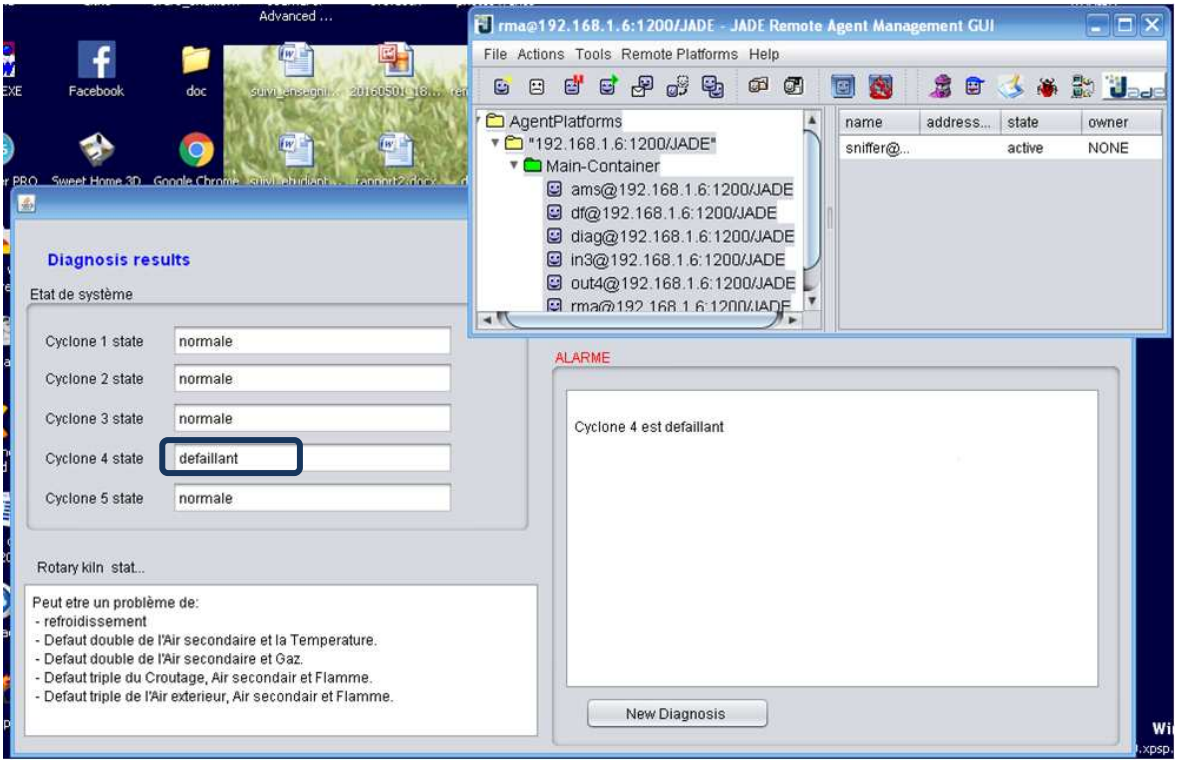


Figure 4.14 Espace des résultats de diagnostic.

4.4 Conclusion

Dans ce chapitre, nous avons détaillé les composantes de l'architecture logicielle de l'application AI-SDAC proposée. Nous avons ensuite présenté notre cas d'étude, la SCIMAT d'Ain Touta et mis l'accent sur les sous systèmes principaux impliqués dans le processus de la clinkérisation. Nous avons décomposé la clinkérisation en trois sous-systèmes : ligne de cyclones, four rotatif et refroidisseur à ballonnets. L'AI-SDAC est développée sous JAVA en utilisant la plate-forme JADE pour implémenter les différents agents. Elle intègre les deux approches de diagnostic, quantitative et qualitative. Elle propose un espace pour gérer les observations quantitatives de la ligne de cyclones et qualitatives pour le four rotatif et le refroidisseur à ballonnets. Un espace pour modéliser le système par un expert et un autre pour afficher les résultats. Pour tester le fonctionnement global d'AI-SDAC, nous avons cherché à l'utiliser dans des conditions proches de la réalité.

Après l'analyse de ces sous systèmes et le test logiciel, nous remarquons que l'aspect coopératif de notre SMA joue un rôle important et sa distribution facilite le diagnostic malgré la complexité trouvée. En effet, chaque ADL évalue un nombre réduit de composants et communique ses résultats aux autre ADL. Les composants défaillants sont informés aux autres ADL en charge de diagnostiquer les composants dépendants structurellement.

Afin de faire coopérer différents outils et systèmes de détection et de diagnostic et gérer d'une manière distribuée et indépendante la surveillance multi-sites dans un univers orienté service, le chapitre suivant fera l'objet de la description et la spécification d'une architecture intégrant les modèles SDAC et SOA pour réaliser cet objectif.

Chapitre 5

Architecture interopérable à base de services web pour la surveillance multi-sites

Pour garantir la tâche de la surveillance, l'ouverture de SDAC et leur coopération avec d'autres systèmes distants est indispensable.

Dans ce chapitre nous utilisons SOA pour organiser et maitriser l'hétérogénéité des systèmes de la surveillance et assurer leur interopérabilité multi-sites.

Dans ce cadre, nous proposons l'architecture SOA-SDAC supportée par une application informatique développée au niveau du système de clinkérisation de la cimenterie d'Ain Touta.

5.1 Introduction

La deuxième contribution principale de ce travail porte sur la proposition d'une architecture orientée-service pour la surveillance distribuée multi-sites. Elle est proposée pour assurer une collaboration machine-machine et offre l'interopérabilité nécessaire à notre modèle multi-agent développé et désigné SDAC dans un cadre multi-sites. En effet, SDAC ne permet pas une communication et une collaboration indépendante des modèles et systèmes technologiques utilisés par les autres centres de surveillances distants. Ainsi, SDAC est couplé avec les concepts du modèle de référence pour l'interopérabilité SOA dans une architecture distribuée et interopérable de surveillance (SOA-SDAC) multi-sites. Aussi, dans ce contexte, les services web sont considérés comme des ressources exploités par des agents logiciels [Chouhal 2011] où nous gardons l'autonomie des deux mondes tout en bénéficiant de leur synergie pour concevoir cette architecture que nous baptisons SOA-SDAC.

Ce chapitre est composé de deux parties. La première est consacrée à la présentation des services web, le concept le plus important pour implémenter la SOA dans le domaine industriel. La deuxième présente leur utilisation pour la mise en œuvre de SOA-SDAC. Après la présentation des composantes et le fonctionnement de celle-ci, nous présentons notre deuxième application industrielle développée pour montrer la faisabilité de la stratégie d'interopérabilité de la SOA-SDAC.

5.2 Présentation des services web

Les services web sont la technologie récente pour l'intégration et l'interopérabilité des systèmes répartis. Basés sur le standard XML, ils sont caractérisés par leur indépendance aux plateformes et aux systèmes d'exploitation. Ceci a conduit leur adoption par les différentes organisations commerciales et industrielles.

Plusieurs définitions ont été proposées dans la littérature. Cette multitude de définitions montre, d'une part, que le service web est en plein essor et que, d'autre part, sa définition dépend fortement de la vision de celui qui l'énonce. Dans notre travail de recherche, le service web est un composant indépendant de tout système, dont l'interface et le contrat d'utilisation sont connus. Il est légèrement couplé i.e. le service web et le programme qui l'invoque peuvent être modifiés indépendamment l'un de l'autre;

ce qui permet une grande flexibilité. Ainsi, nous nous intéressons à deux vues complémentaires à savoir la vue architecturale et la vue technologique (figure 5.1).

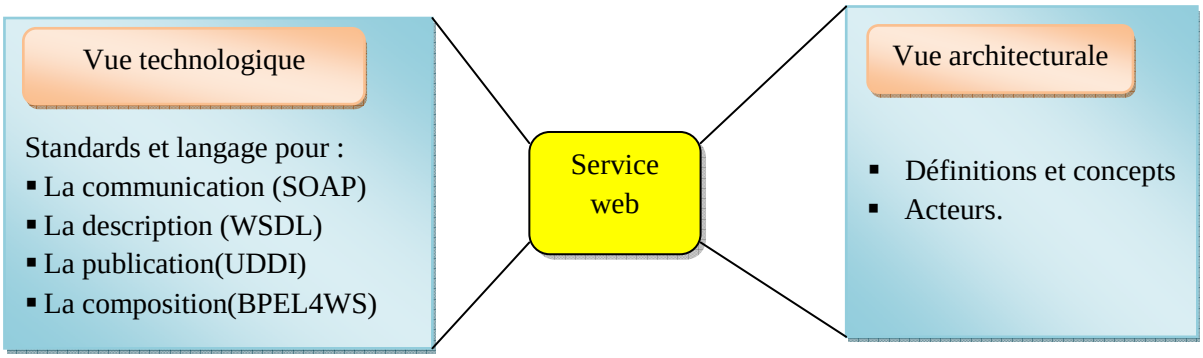


Figure 5.1 Cadre bidimensionnel pour l'analyse bibliographique liée au service web.

5.2.1 Point de vue architecturale

Une architecture orientée service rassemble les grandes applications de l'entreprise en services interopérables et réutilisables. Elle décompose une fonctionnalité en un ensemble de fonctions basiques, appelées services, et de décrire finement le schéma d'interaction entre ces services. L'aspect le plus important de SOA est qu'elle sépare l'implémentation du service de son interface et autorise les applications ou services à communiquer et à travailler ensemble quelle que soit leur plateforme respective, et ce par le biais de services. Les principaux acteurs qui interviennent dans un SOA sont illustrés sur la figure 5.2 [Ishak 2010].

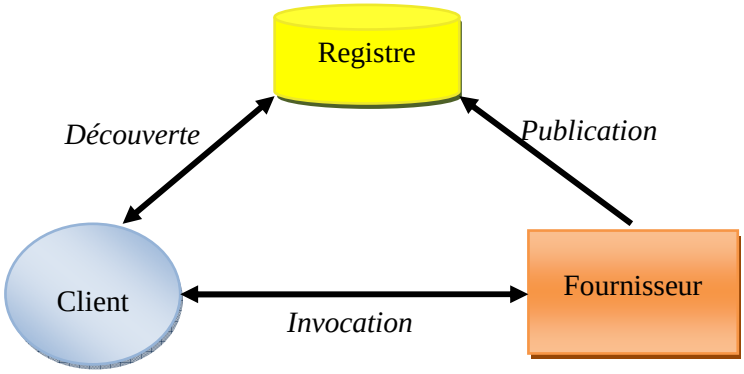


Figure 5.2 Principaux acteurs dans SOA.

- ***le fournisseur*** du service se charge de la description et la publication du service web créé dans un annuaire afin qu'il puisse être trouvé par des clients ;
- ***le registre*** de services fournit des informations sur la description et la localisation des services Web ;

- ***le client*** est une application qui cherche, localise et invoque le service.

Dans [Nickull 2005], l'auteur considère qu'il n'existe pas à proprement parler de spécifications officielles d'un SOA, néanmoins les principales notions fédératrices ou concepts clés que l'on retrouve dans une telle architecture sont représentés sur la figure 5.3.

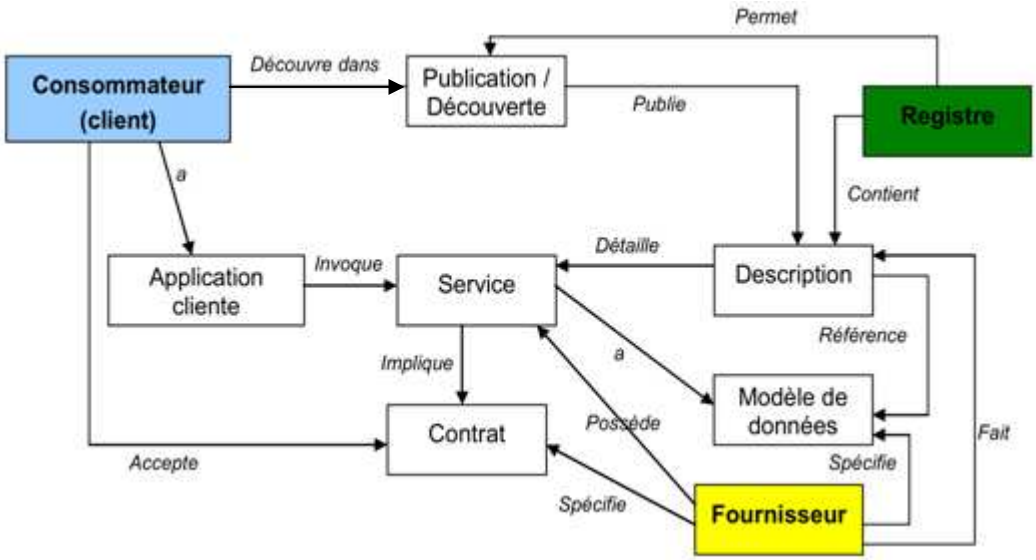


Figure 5.3 Concepts clés de SOA et leurs relations.

La notion de service représente une fonction encapsulée dans un composant que l'on peut interroger à l'aide d'une requête composée de paramètres et fournissant une ou plusieurs réponses. Chaque service doit être indépendant des autres afin de garantir sa réutilisabilité et son interopérabilité. La description du service consiste à décrire les paramètres d'entrée du service, le format et le type des données retournées.

Le registre de services permet la publication des services disponibles aux utilisateurs et la découverte qui recouvre la possibilité de rechercher un service parmi ceux qui ont été publiés. L'invocation, représente la connexion et l'interaction du client avec le service. Le contrat de service est une spécification de l'interaction entre le consommateur du service et le fournisseur du service. Il spécifie le format de la requête et de la réponse du service. Il peut aussi spécifier les niveaux de qualité du service. Le modèle de données spécifie les paramètres impliqués dans l'invocation du service.

5.2.2 Point de vue technologique

La vue technologique décrit la pile de standards et langages sur lesquels s'appuient les services web (figure 5.4).

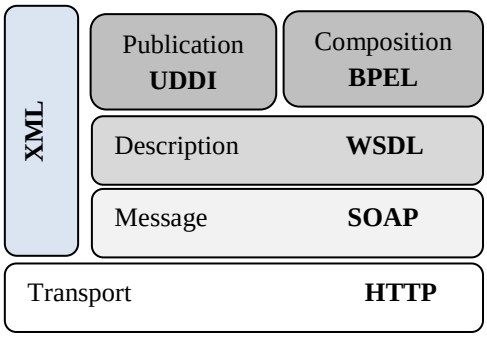


Figure 5.4 Pile de standards et langages des services Web.

Dans les sections suivantes, nous allons décrire l'ensemble des standards et langages relatifs à cette technologie et qui permettent :

5.2.2.1 La communication entre les services

SOAP (Simple Object Access Protocol) [SOAP 2003] est un standard du consortium W3C (*WorldWideWeb Consortium*) définissant un protocole pour la communication et le dialogue avec les services web. Dans un souci d'interopérabilité, les messages échangés lors de l'utilisation du protocole SOAP sont basés sur XML.

5.2.2.2 La description des services

Le langage WSDL (Web Services Description Language) [W3C 2004] est un standard de W3C basé sur une syntaxe XML. Il décrit l'interface publique du service web en définissant l'adresse du service, son identité, les opérations qui peuvent être invoquées par les clients, les paramètres des opérations ainsi que leurs types.

5.2.2.3 La publication des services

UDDI (Universal, Description, Discovery and Integration) [UDDI 2003] fournit un protocole, une API et une plateforme permettant aux utilisateurs de services web, depuis n'importe quel système, de localiser dynamiquement à travers Internet les services web qu'ils désirent utiliser. Aussi elle fournit trois services de bases : Publish, Find et Bind qui gèrent respectivement comment le fournisseur de service web s'enregistre lui-même ainsi que ses services.

5.2.2.4 La composition des services

Les services web présentent la possibilité d'être composés donnant ainsi naissance à des services composites de valeurs ajoutées. La création d'une application

distribuée complexe peut être obtenue par la composition de services web et les différentes dimensions d'un modèle de composition de service web sont détaillées par [Lopes 2005]. Afin de supporter la composition de services, BPEL4WS (Business Process Execution Language for Web Services) [Andrews 2003] est constitue aujourd'hui le standard.

Ces technologies standard sont interdépendantes, même si elles ont été conçues de façon indépendante. La figure 5.5 présente ces technologies des services web en mettant en évidence leur interdépendance et leur utilisation par les acteurs fournisseur et demandeur [Lopes 2005].

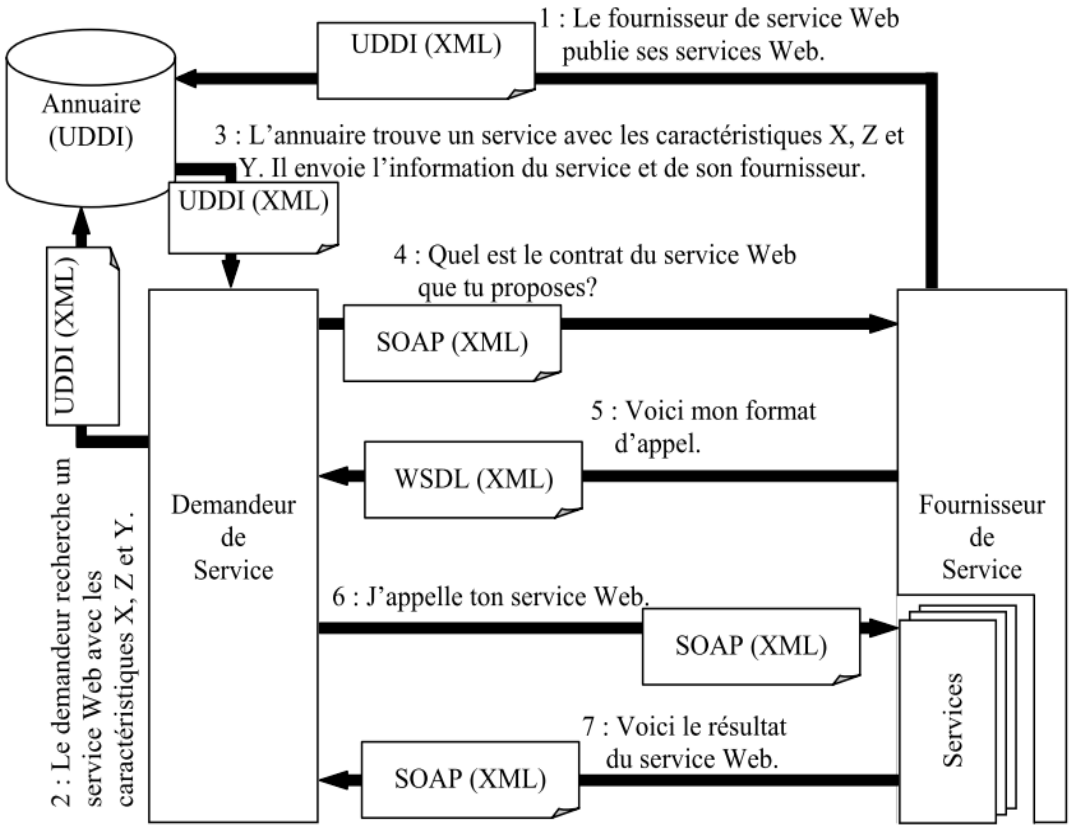


Figure 5.5 Les relations entre les principales technologies des services web.

Les trois acteurs demandeurs, fournisseurs de service et annuaire maintiennent une relation par les biais des technologies de service web :

1. Le fournisseur de service publie ses services web sur l'annuaire en utilisant UDDI.
2. Le demandeur recherche un service web avec les caractéristiques X, Z et Y.
3. L'annuaire trouve un service avec les caractéristiques X, Z et Y, et envoie les informations sur le fournisseur du service et sur le service.

4. Le demandeur de service demande le contrat du service web au fournisseur.
5. Le fournisseur envoie le contrat du service (WSDL).
6. Le demandeur de service appelle le service web selon son contrat (appel en SOAP).
7. Le service web retourne le résultat de l'appel (SOAP).

En se basant sur ces deux vues architecturale et technologique, nous avons développé notre architecture SOA-SDAC présentée sur la section suivante pour assurer l'ouverture de notre modèle d'agent SDAC sur le web et d'acquérir une certaine flexibilité.

5.3 La surveillance distribuée à base de SOA: SOA-SDAC

Suite aux limites au niveau des connaissances et méthodes appliquées pour surveiller les SdP, les défauts ne peuvent pas être détectés ni diagnostiqués correctement par leurs systèmes de surveillance. Cependant, si ces connaissances et ces méthodes ainsi celles des centres de recherche sur la surveillance sont publiés à base de la SOA, les entreprises peuvent tirer pleinement parti des services Internet pour réaliser la surveillance à distance [Lee 2014] [El-Sharkawi 2013] [Rebeuf 2004] et [Arpaia 2010]. Ils peuvent non seulement partager les ressources et améliorer la précision du diagnostic, mais aussi améliorer l'efficacité et réduire les coûts. En effet, en cas de défaut dans le SdP, les ingénieurs peuvent réparer le système en se basant sur leurs propres expériences ou par la recherche dans les documents fournis par les fournisseurs. Très souvent, quand le problème est trop compliqué, les fournisseurs envoient leurs techniciens pour résoudre le problème; ce qui entraîne une augmentation des coûts liés à la réparation. Ainsi, la surveillance à distance présentée par [Pala 2014] [Resceanu 2008] [Shao 2013] [Tan 2011] [Wang 2007] et [Zhao 2010] permet aux experts des fournisseurs de se relier à distance via Internet pour prendre rapidement des décisions, telles que la configuration et les maintenances afin d'assurer les performances des SdP. Comme elle permet au système de surveillance de s'ouvrir à d'autres systèmes distants pour garantir par coopération la tâche de la surveillance.

Dans ce contexte, nous présentons notre proposition de la surveillance distribuée multi-sites qui proviennent d'une alliance de notre modèle agent SADC et de la gestion de l'hétérogénéité des applications apportés par la SOA. En se basant sur notre modèle d'assemblage de SMA et SOA, nous avons conçu et ensuite implémenter les composantes de SOA-SDAC.

5.3.1 Modèle de SOA-SDAC

SOA et SMA sont deux technologies de plus en plus importantes pour la construction des systèmes logiciels. Les objectifs de ces deux architectures partagent quelques similitudes, comme la création de systèmes distribués et flexibles composés d'entités faiblement liées, et qui interagissent les unes avec les autres. Cependant, il existe des différences majeures dans leurs technologies. En effet, les services possèdent des standards pour la description des interfaces ainsi que des protocoles totalement différents des langages de communications des agents. Ainsi, ils ne peuvent pas interagir les uns avec les autres directement. Certains travaux ont étudié ce problème et ont montré que l'intégration des agents et des services web produit des avantages intéressants [Cheaib 2010]. Les services ont une infrastructure bien définie et interopérable, alors que les agents fournissent des capacités sociales et de l'intelligence pour les applications. En conséquence, l'intégration des agents et des services web améliorent l'adaptabilité, l'interopérabilité et l'ouverture du système. Ainsi, l'originalité du modèle de SOA-SDAC (figure 5.6) provient d'une alliance entre SDAC qui est un SMA et SOA.

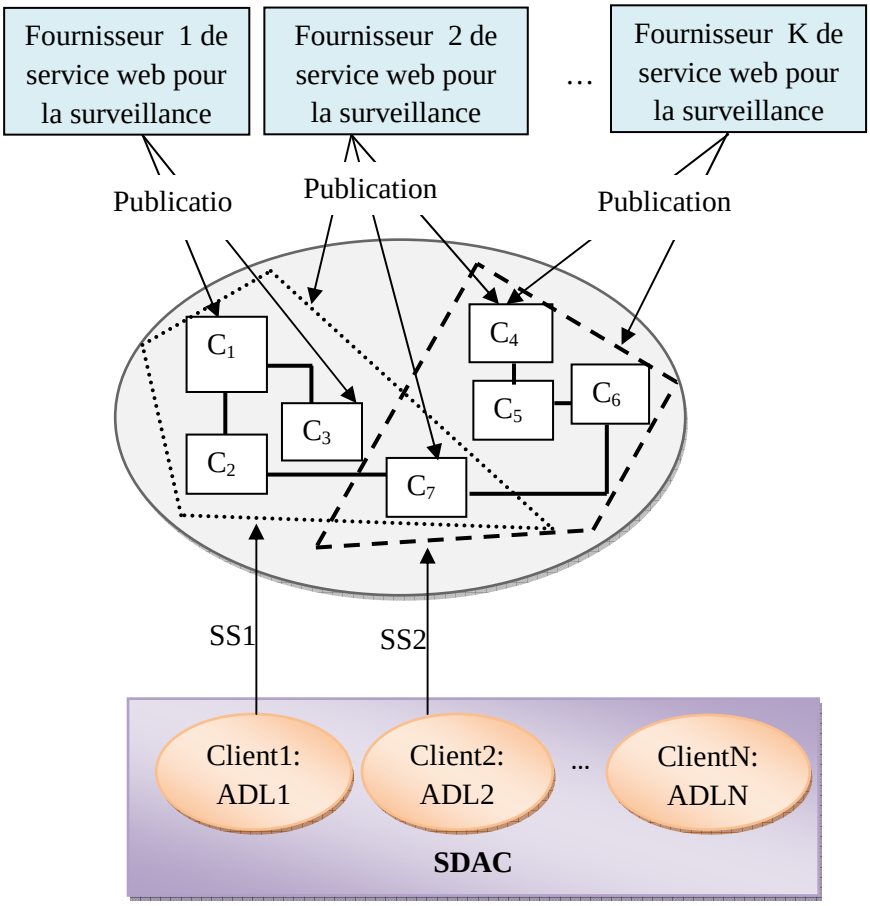


Figure 5.6 Modèle de SOA-SDAC.

Le modèle SOA-SDAC intègre plusieurs fournisseurs qui publient des services web pour la surveillance. Ces fournisseurs liés via Internet peuvent être des centres de surveillance professionnelle, des universités ou des centres de recherches. Un service web regroupe un ou plusieurs savoir-faire dans un domaine de services bien défini. Notre domaine de service est un espace regroupant plusieurs services de détection et de diagnostic au niveau composant (C_i), sous système (SSi) ou système. Un service peut être élémentaire, i.e. n'appelle pas d'autres services, ou être composite construit à l'aide de services élémentaires.

De plus, il intègre notre SDAC pour la surveillance distribuée comme client de ces services web. Dans ce travail, les ADLs sont les clients réels qui appellent les services web selon leurs besoins. En effet, SDAC consiste à décomposer le système complexe en plusieurs SS et associe un ADL à chaque SS. Après l'acquisition et le filtrage de données chaque ADL peut adopter sa propre stratégie de diagnostic (Neuro Flou, Logique floue, etc.) et doit uniquement mettre en œuvre une interface de communication avec les autres diagnostiqueurs locaux. Les ADLs coopèrent pour calculer leurs diagnostics locaux en garantissant la cohérence de leurs résultats pour établir le diagnostic global. Si les résultats finaux sont satisfaits alors ils sont stockés dans une base de données et de connaissance sinon des services web sont découverts pour répondre aux besoins des ADLs. Si aucun service n'est disponible, l'expert du domaine doit s'intervenir pour réguler le problème. La figure 5.7 décrit cette stratégie de diagnostic à base de service web d'un SS par chaque ADL.

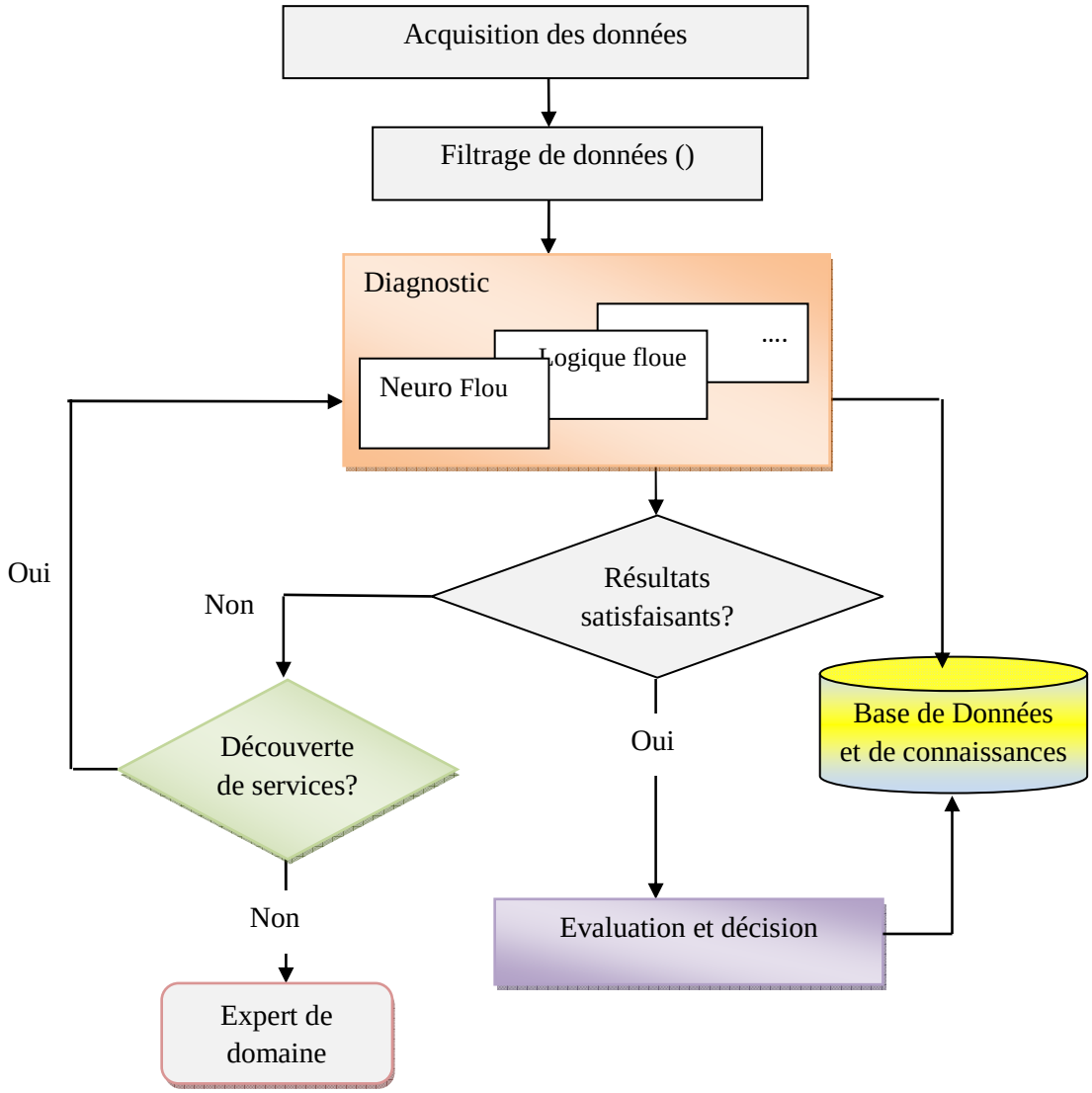


Figure 5.7 Stratégie de diagnostic d'un ADL à base de service web.

5.3.2 Les composantes de l'architecture SOA-SDAC

En se basant sur le modèle de SOA-SDAC de la figure 5.6, l'architecture proposée illustrée par la figure 5.8, s'articule autour de trois acteurs principaux du modèle SOA à savoir Internet UDDI, producteur de surveillance à distance et SDAC.

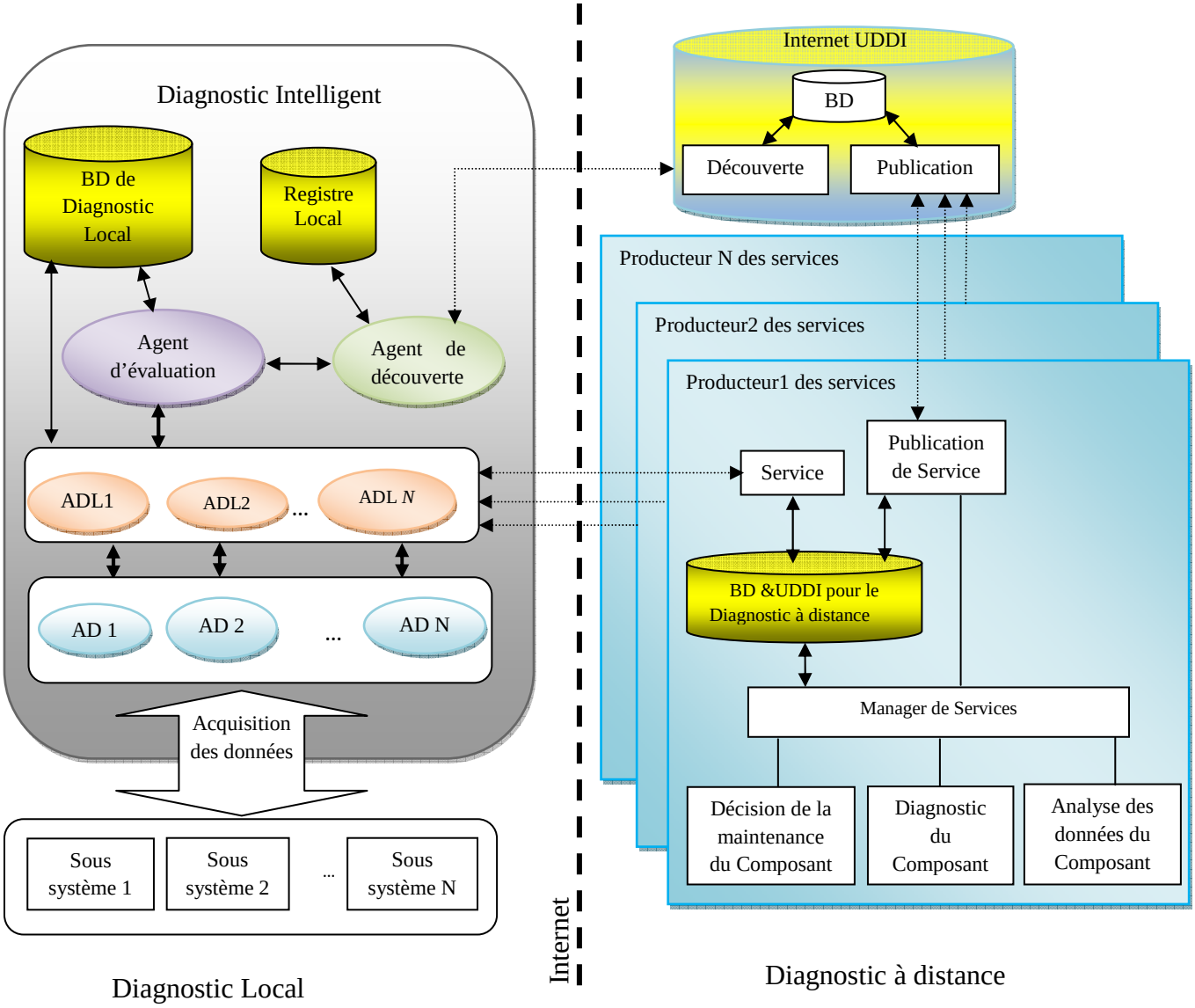


Figure 5.8 Architecture de la SOA-SDAC.

5.3.2.1 Internet UDDI

Il s'agit du registre de services contenant la description des services fournis par les différents producteurs. Il permet la mise en relation du client et des producteurs. Le registre contient une interface de communication permettant aux clients de découvrir les services conduisant à la surveillance de l'état de leurs SdP et aux producteurs de déclarer leur savoir faire en publiant leurs services. Le service de Découverte permet aux clients de rechercher les services ainsi que leurs producteurs. Le service de Publication permet aux producteurs de publier la description de leurs services. Ces deux services communiquent avec une base de données de services (BD) contenant des informations sur les services fournis et leurs fournisseurs.

5.3.2.2 Structure du producteur de surveillance à distance

Un producteur est une entité qui possède des savoir-faire représentés sous forme d'activités de surveillance, réalisables par un ensemble de ressources locales gérées par une application interne de gestion.

Chaque producteur possède une application interne de gestion (Manager de Services), une base de données et UDDI (BD &UDDI pour le Diagnostic à distance), une composante de description et de publication (Publication de Service), le(s) service(s) fournis et les ressources (Décision de la maintenance du Composant, Diagnostic du Composant, Analyse des données du Composant, etc.) permettant la réalisation des activités fournies par le(s) service(s).

L'exposition des activités au public se fait via des services qui implémentent les fonctions de surveillance assurées par le Manager de Services. Ces services sont décrits dans la base de données locale stockant la description des services, les informations concernant les clients et les résultats établis par le Manager de Services.

5.3.2.3 Structure de client SDAC

Le client dispose d'un SMA responsable de la détection et le diagnostic, d'un agent d'évaluation, d'un agent de découverte permettant de découvrir les services recherchés ainsi que leurs producteurs, d'une base de données (BD de Diagnostic Local) contenant les résultats de diagnostic locales description, et d'un registre local. Ce dernier stocke les informations sur les services les plus fréquemment sollicités par le client ainsi que les producteurs avec lesquels il a eu de bonnes collaborations. Il permet d'accélérer la découverte des services, de mieux guider le client dans le choix de ses partenaires en privilégiant les producteurs les plus sollicités. Localement, l'agent de découverte recherche dans le registre local les services en question. Il peut rechercher également dans l'Internet UDDI pour découvrir les services non trouvés dans le Registre local. L'Agent de découverte permet l'intégration de services web et d'agents est implémenté par l'outil WSIG (Web Service Integration Gateway) dispositif clairement utile de JADE qui fournit les moyens d'enregistrer des services dans l'environnement JADE.

Le WSIG est un composant de la plateforme JADE, qui tente de traduire les messages utilisés par les services web, à des messages compréhensibles par les agents logiciels. Ainsi, il supporte la pile des protocoles standards des services web (WSDL, SOAP et

UDDI) (figure 5.9). Le WSIG est une application web composée de deux éléments principaux, le « WSIG servlet » et le « WSIG agent ». Le premier représente la facette vers le monde d'Internet. Il est responsable de:

- La collecte des demandes d'entrées HTTP / SOAP demandes
- L'extraction du message SOAP
- La préparation de l'action d'agent correspondant et le transmettre au "WSIG Agent". Une fois l'action est faite :
- La conversion du résultat de l'action à un message SOAP.
- La préparation des protocoles HTTP / SOAP pour la réponse à envoyer au client.

Le second est la passerelle entre le Web et le monde des agents. Il est responsable de :

- Le transfert des actions reçues par le "WSIG servlet" aux agents qui sont en mesure de les exécuter, et puis collecter les réponses.
- L'enregistrement dans l'annuaire des agents DF (Directory Facilitator) du JADE affin de recevoir des notifications sur les enregistrements des agents dans le système.
- La création du fichier WSDL correspondant à chaque service d'agent enregistré par le DF, et publier le service dans l'UDDI, si nécessaire.

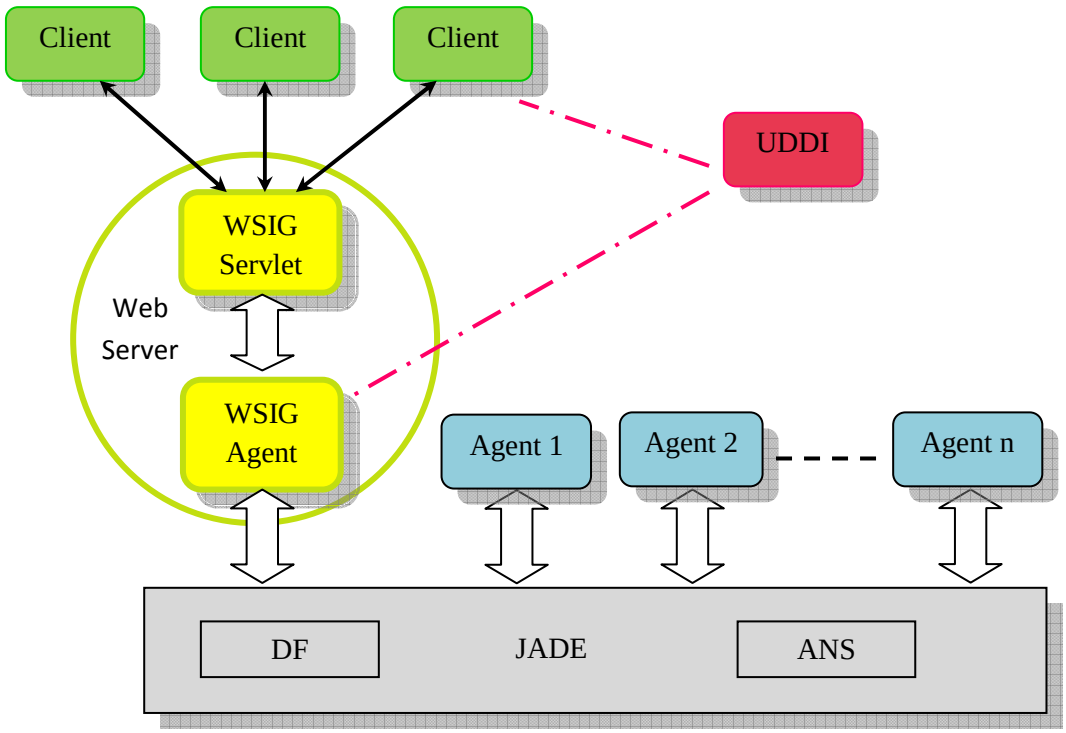


Figure 5.9 Architecture du WSIG [Cheaib 2010].

5.3.3 Fonctionnement de l'architecture SOA-SDAC

Le fonctionnement global de l'architecture SOA-SDAC concerne deux grandes étapes: l'étape de mise en relation des partenaires, et celle d'invocation des services et réponse aux requêtes des clients.

La première se résume par la publication des services par les producteurs et la découverte de ces services par les clients intéressés. Dans la publication, les acteurs concernés sont l'internet UDDI et le producteur de surveillance à distance publiant les descriptions des services fournis dans le l'internet UDDI.

La découverte des services fait intervenir l'internet UDDI et le client SDAC qui peut ainsi découvrir les services publiés et leurs producteurs. La deuxième étape met en interaction le client SDAC et ses producteurs de surveillance à distance. Cette étape permet l'invocation des services distants fournis par les producteurs en faisant intervenir les agents du modèle SDAC chez le client (demandeurs de services) et les services des producteurs concernés. Le fonctionnement est décrit par le diagramme de cas d'utilisation de la figure 5.10.

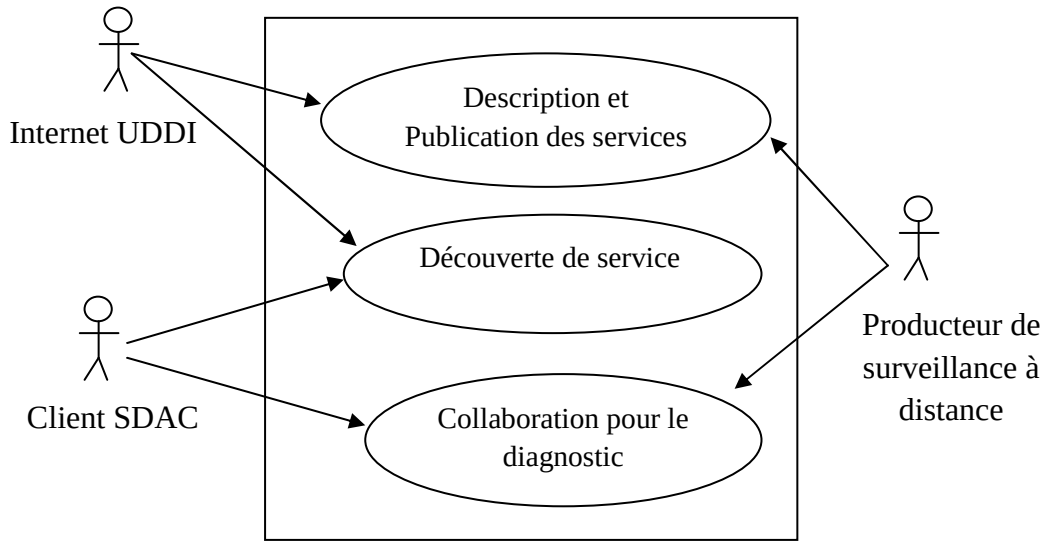


Figure 5.10 Diagramme de cas d'utilisation de SOA-SDAC.

Les diagrammes de séquences suivants illustrent les scénariis des cas d'utilisation de description et de publication des services, découverte des services et collaboration.

5.3.3.1 Description et publication des services

Le diagramme de séquences de la description et de la publication des services par un producteur de surveillance à distance est illustré sur la figure 5.11. Il décrit les services fournis par le producteur et stocke ces descriptions dans la base de données (BD &UDDI pour le diagnostic à distance). Pour les publier, ce module invoque le service de publication de l'Internet UDDI. Il s'authentifie d'abord auprès du registre s'il est déjà inscrit, dans le cas contraire, il doit demander la création de son compte auprès de l'administrateur du l'Internet UDDI. Après l'authentification du module de description et de publication du producteur, ce dernier peut publier de nouveaux services dans les domaines concernés ou modifier les descriptions de ses services. A chaque réception d'une nouvelle description de service, l'Internet UDDI enregistre dans sa base de données les informations contenues dans la description. Après l'enregistrement du service, l'Internet UDDI retourne au producteur correspondant une confirmation de la publication de ses services.

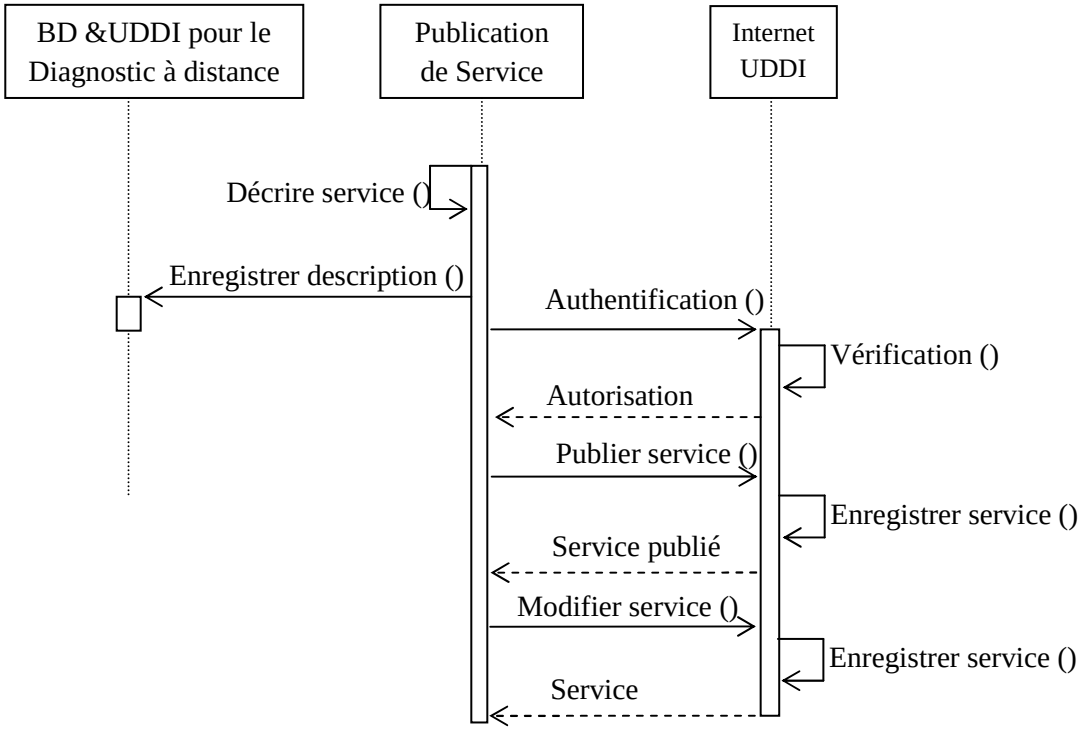


Figure 5.11 Diagramme de séquence pour la publication des services.

5.3.3.2 Découverte des services

La découverte est réalisée par l'agent de découverte des services qui est pratiquement le WSIG, et illustré par le diagramme de séquences de la figure 5.12.

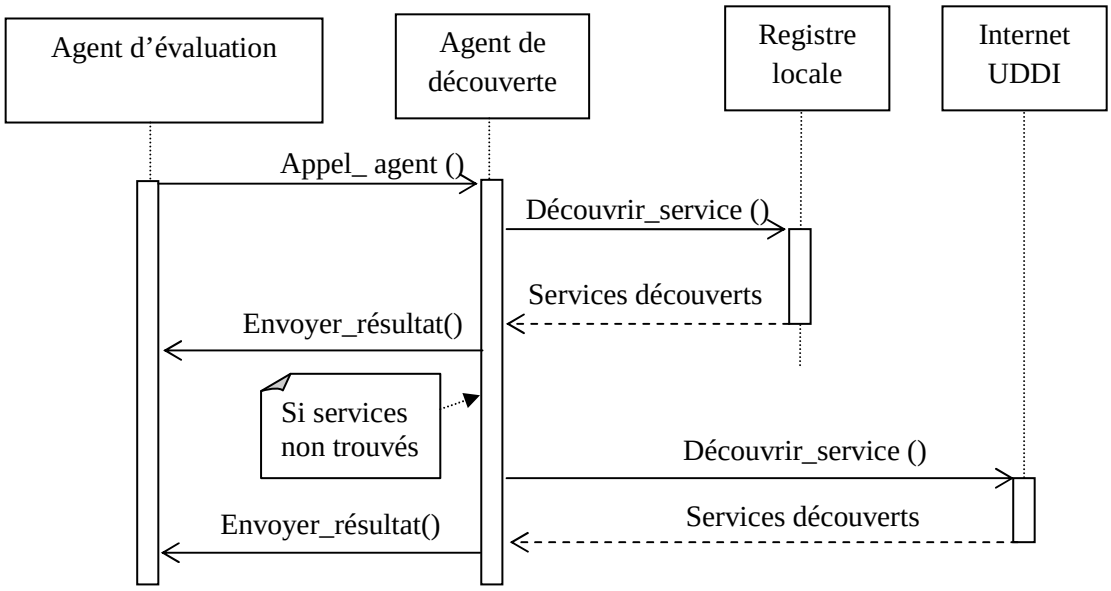


Figure 5.12 Diagramme de séquence pour la découverte de service.

Suite à l'appel de l'agent de découverte des services par l'agent d'évaluation, cet agent recherche d'abord dans le registre local afin de trouver les services les plus fréquemment sollicités par le client. Dans la demande de découverte, l'agent d'évaluation doit préciser le(s) domaine(s) des services recherchés ainsi que les activités souhaitées. Dans le cas où certains services ne sont pas trouvés dans l'agent de découverte de services du client invoque le service de découverte de l'Internet UDDI. Les résultats de la découverte, locale ou globale, sont transmis ensuite à l'agent d'évaluation.

5.3.3.3 Collaboration entre le client SDAC et le producteur de surveillance à distance pour le diagnostic des défauts

C'est une collaboration machine-machine à base de services web et d'agents. Ce type de collaboration est implémenté par une simple architecture client/serveur. Dans notre travail, les services web sont considérés comme des ressources exploités par des agents logiciels. Ainsi, nous gardons l'autonomie des deux mondes tout en bénéficiant de leur synergie pour concevoir une architecture collaborative.

La collaboration est déclenchée suite à une demande d'un agent d'évaluation d'un service ou spécifique envoyée à l'agent de découverte afin de répondre à des besoins d'un agent de diagnostic local (ADL). Après la découverte du service en question et la récupération des informations sur le service et le producteur correspondant l'agent d'évaluation initialise le client SDAC. Des interactions sont ensuite déclenchées entre les agents ADL avec les producteurs afin de gérer et négocier le diagnostic des défauts. Une

fois la requête d'invocation émanant d'un ADL reçu par le service, ce dernier stocke les informations contenues dans cette requête dans la BD &UDDI pour le diagnostic à distance, le diagramme de séquences du traitement d'une invocation du service chez un producteur et la proposition d'un résultat est illustré sur la figure 5.13.

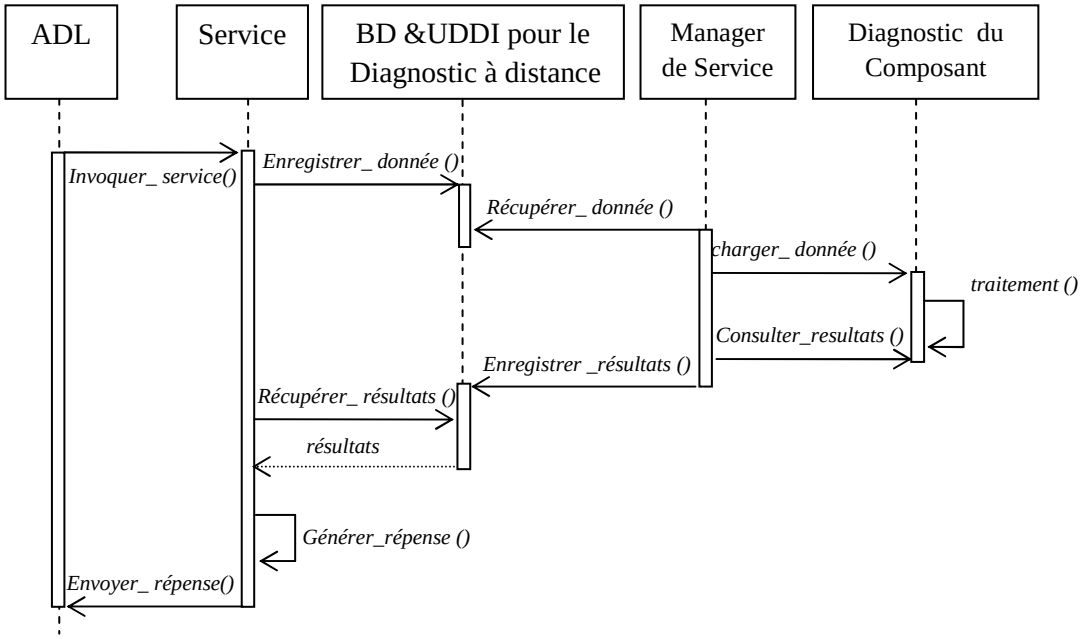


Figure 5.13 Invocation d'un service web par un ADL et envoie du résultat.

5.3.4 Configuration matérielle de SOA-SDAC

La configuration matérielle de la SOA-SDAC est représentée sur la figure 5.14. Dans la disposition matérielle, les modules Diagnostic Intelligent et Manager de Service sont placés dans des serveurs d'applications correspondants, tels que le serveur de diagnostic local et le serveur d'acquisition de données dans le site de diagnostic local et le Manager de Service dans le fournisseur de diagnostic à distance. Un appel à distance est effectué par le *client SDAC* en envoyant un message à un système distant (serveur) : Fournisseur de diagnostic pour exécuter une procédure donnée. Le résultat calculé par le serveur est ensuite envoyé au client. Le client et le serveur utilisent un encodage de données différentes, le client est développé en java et le serveur qui sera présenté par la section suivant est développé en utilisant le langage de programmation C#. Donc, des *proxys* placés dans les Serveurs Web réalisent une conversion juste après l'appel de la fonction par le client, juste avant l'envoi effectif au serveur ou lors de la réception du message de réponse.

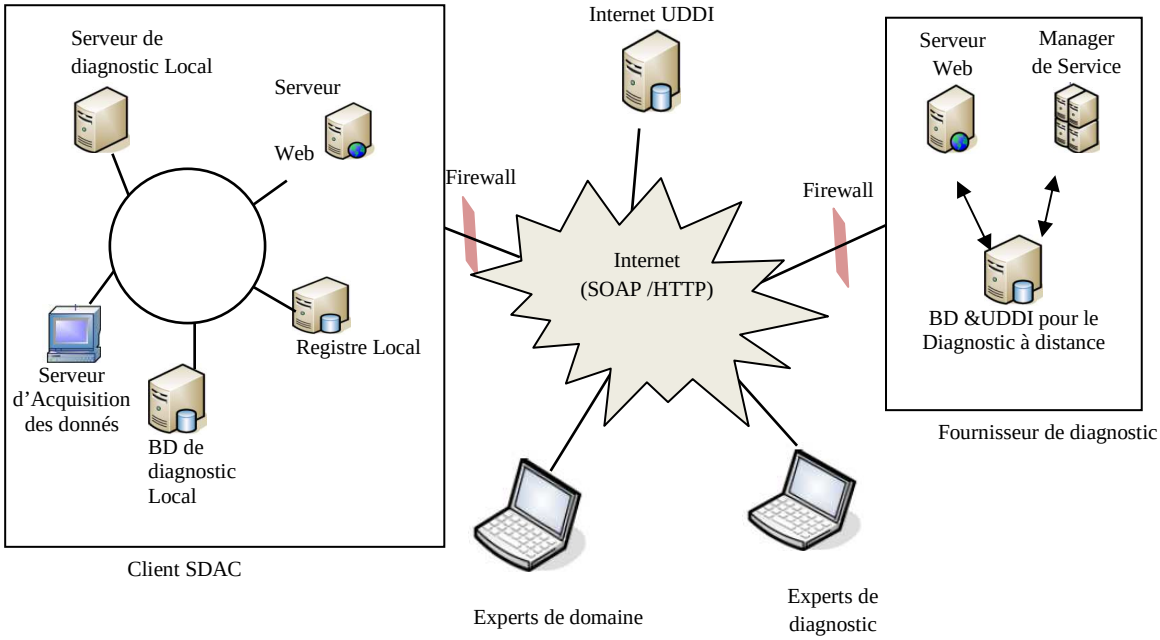


Figure 5.14 Configuration matérielle de SOA-SDAC.

5.3.5 Application industrielle

[Aitouche 2013] à recensé plus de 15 thèses de doctorats, 23 mémoires de magister et 37 mémoires d'ingénieur et master 2, prenant la SCIMAT comme objet d'étude sur tous les aspects de l'industrie, la surveillance, le diagnostic, les ressources humaines, la production, la logistique, etc. La liste n'est pas exhaustive, et une grande partie de ces travaux ont comme résultats des applications informatiques efficaces. L'idée est de regrouper ces savoir-faire dans des domaines de services puis les publiés sous forme de services web i.e. publier certaines fonctionnalités de ces application sous forme de service web en conservant les aspects privés du fonctionnement local. Par conséquence, notre Laboratoire d'Automatique et Productique (LAP) devient un fournisseur de service web lié via Internet. Dans ce travail, nous nous intéressons à l'ensemble des techniques utilisées pour la surveillance et le diagnostic appliquées sur le processus de la production de clinker. Elles utilisent les différentes valeurs capturées des variables (température, débit, etc.) et retournent les résultats (alarme, défaillance, etc.) affectant ce processus (figure 5.15).

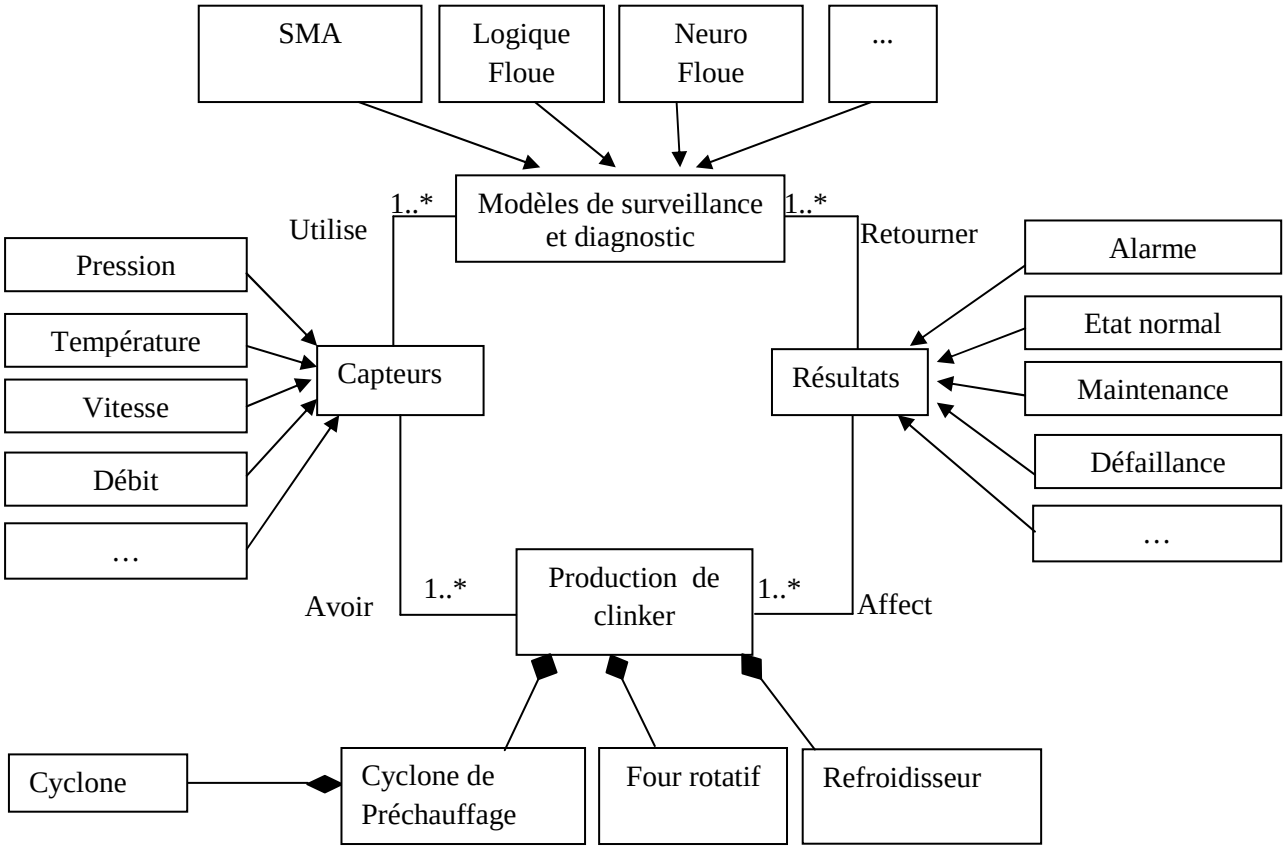


Figure 5.15 Conception des systèmes de surveillance et diagnostic en SCIMAT.

Pour montrer la faisabilité de la stratégie d'interopérabilité du modèle SOA-SDAC. Cette partie est une extension de l'application présentée dans le chapitre 4. Dans cette partie nous intéressons à la composante : *producteur de surveillance à distance* de l'architecture de SOA-SDAC. Ainsi, notre producteur est une entité qui possède une application interne, un service encapsulant une fonctionnalité assurée par cette application, et les ressources permettant la réalisation des activités fournies par le service. Ce producteur peut fournir un ou plusieurs services.

Ce producteur est impliqué à la surveillance distribuée à base de service web d'une ligne de *préchauffeur à cinq cyclones.* Ainsi, l'application interne nommée : *controleur* développée par C# sous l'environnement Visual Studio 2012, permet de détecter tous les comportements anormaux ou nominaux des cyclones. La fonctionnalité de détection des anomalies basée sur la génération de résidu est encapsulée dans un service web nommé *temperature_controller* (figure 5.16).

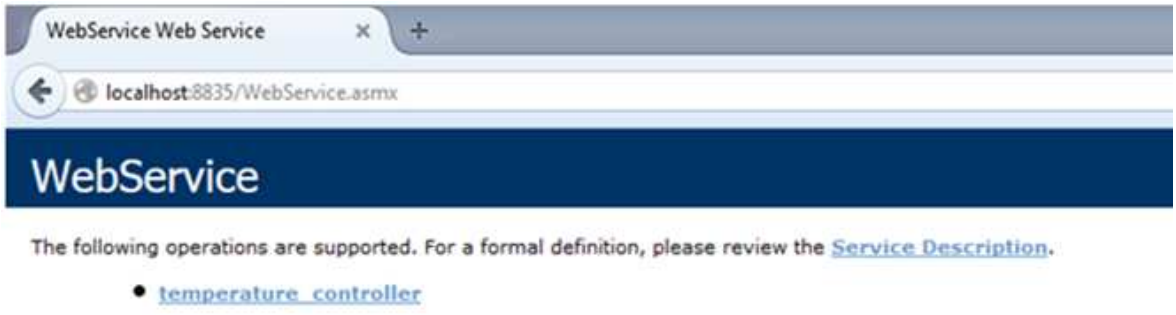


Figure 5.16 Le service web temperature_controller.

Ce service web est invoqué par l'application *controleur* lui-même à chaque fois pour vérifier le statut de chaque cyclone. Plusieurs paramètres à vérifier pour déduire l'état de chaque cyclone. Dans cette implémentation nous nous intéressons uniquement aux deux paramètres les plus importants : variation de la température en fonction de la pression de gaz. La figure 5.17 présente l'introduction de la température de préchauffage pour chaque cyclone.

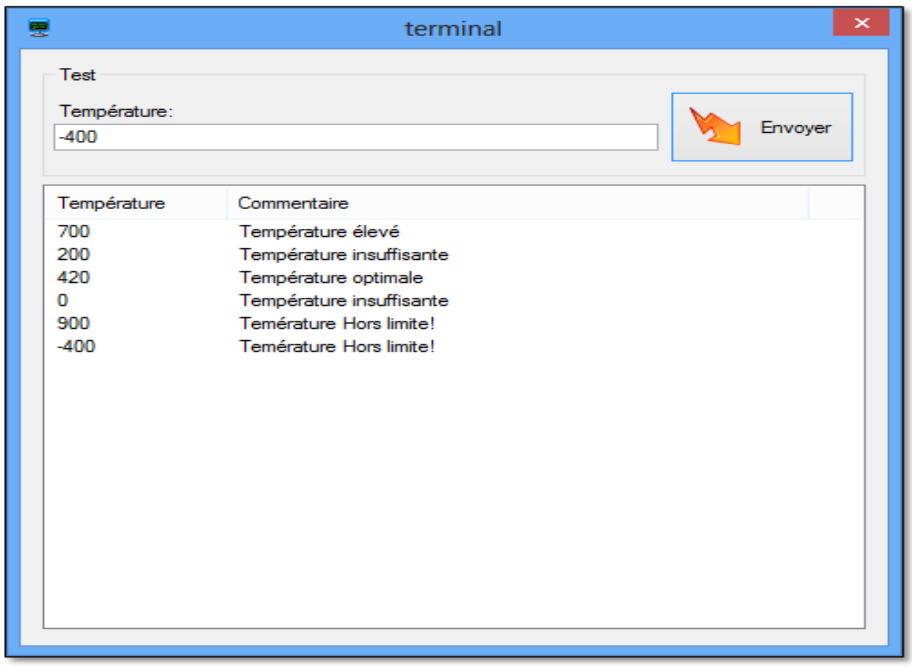


Figure 5.17 Espace pour introduire le paramètre Température.

Après l'introduction de la température, la figure 5.18 illustre les statuts des cinq cyclones en fonction de la pression du gaz et l'historique des directives générés par le service web.

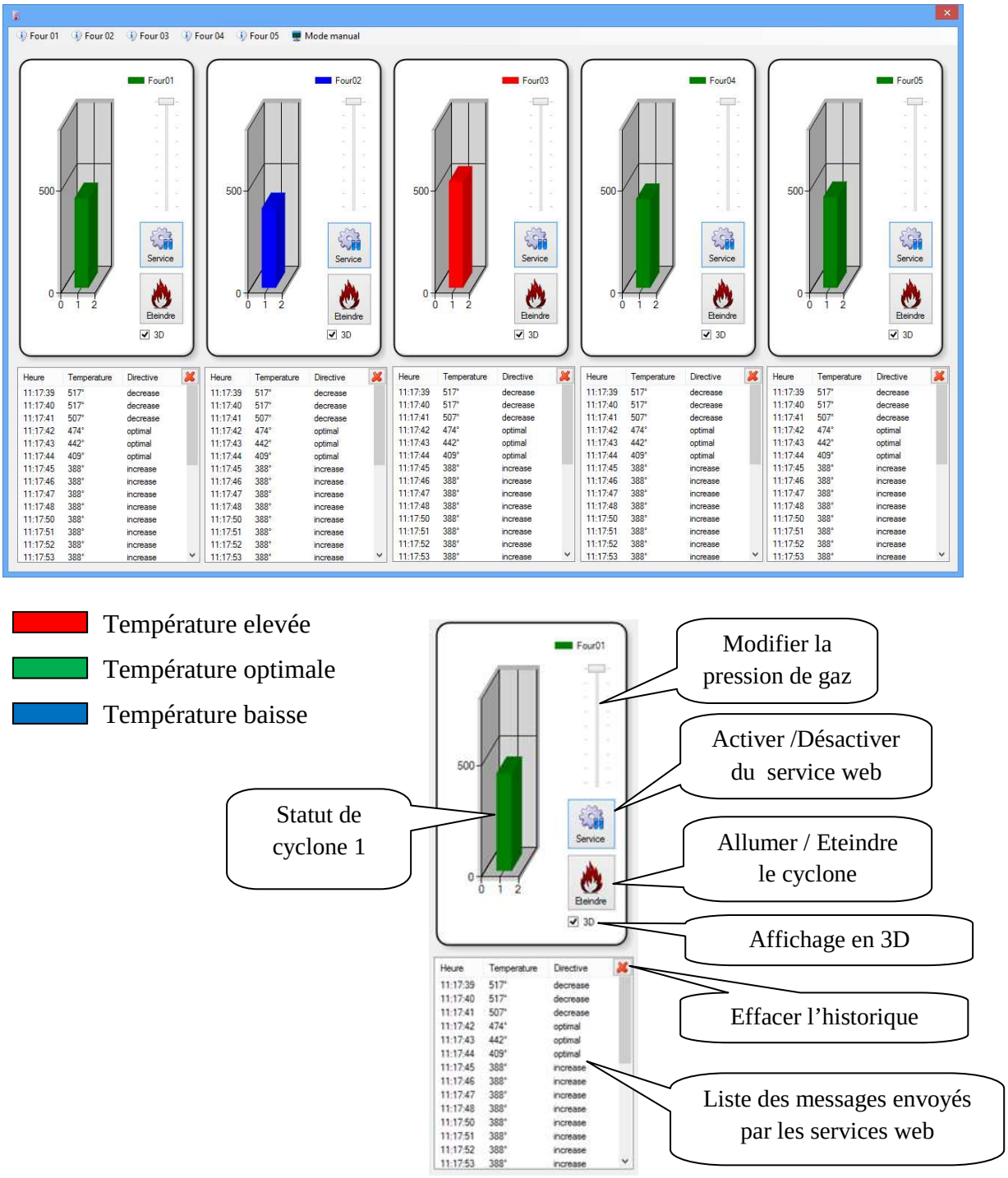


Figure 5.18 Statut des cyclones.

En cliquant sur *Service Description* de la figure 5.16 le fichier WSDL décrivant ce service web s'affiche dans la figure 5.19. Par contre, si on clique sur le nom de service *temperature_controller* on peut interagir avec ce service et afficher les messages SOAP requête et réponse figure 5.20.

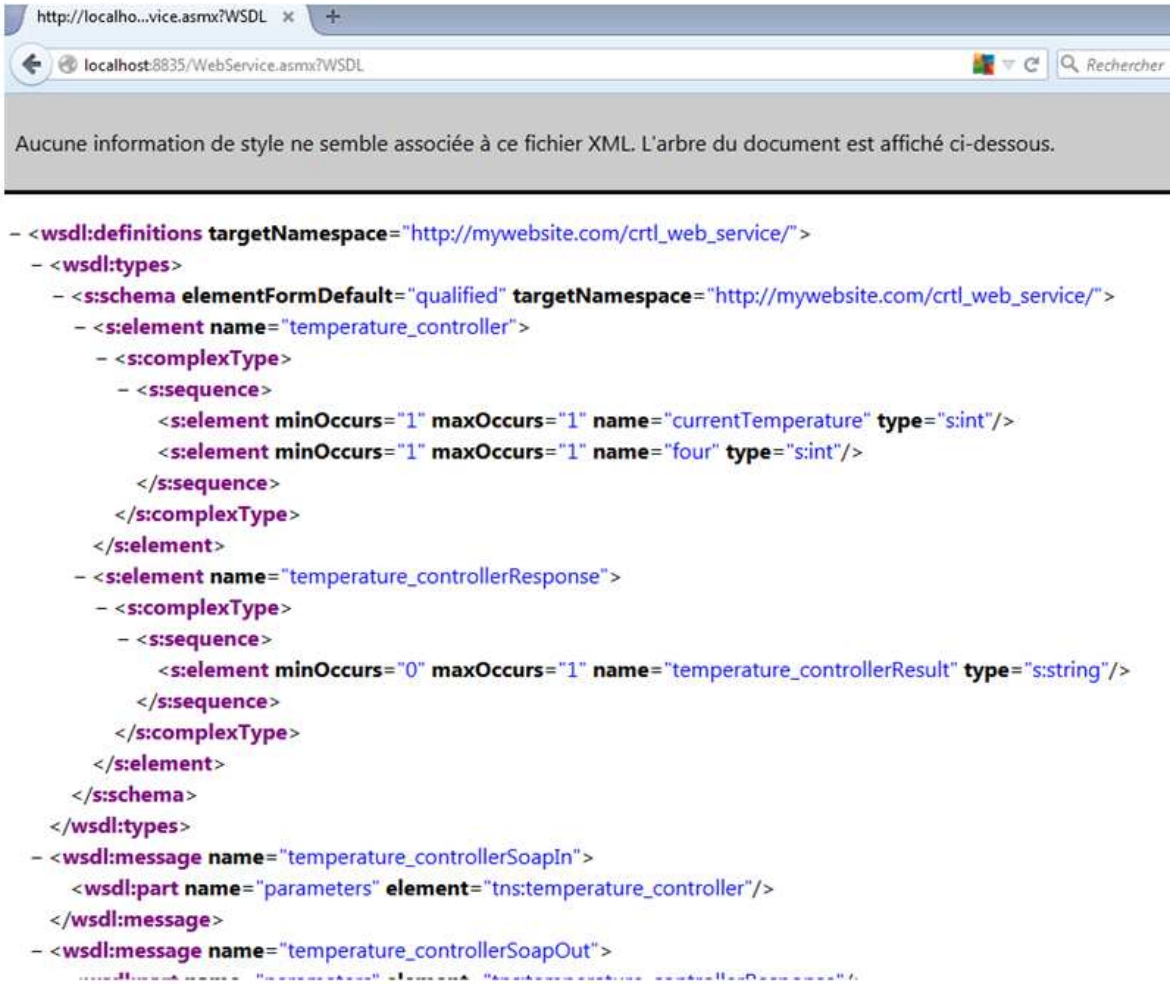


Figure 5.19 Le fichier WSDL de service web: temperature_controller.

Figure 5.20 Les messages SOAP requête et réponse.

5.4 Conclusion

Ce chapitre a mis en évidence le concept de service web et sa pertinence pour l'interopérabilité. Il a présenté à travers la vue architecturale et la vue technologique les différents aspects relatifs au service :

La vue architecturale a présenté l'architecture orientée services dont le principe consiste à structurer le système d'information de l'entreprise comme un ensemble de services qui exposent leur interface fonctionnelle et qui communiquent par messages. L'instanciation la plus connue de cette architecture est la technologie service web. Cette dernière a fait l'objet de la vue technique

La vue technologique a exposé les différents standards et langages sur lesquels se basent les services : SOAP pour la communication et le dialogue avec les services web, le standard WSDL pour la description des services, le registre UDDI pour la publication des services et le langage BPEL pour la composition des services.

Basée sur l'intégration de ces concepts et le modèle à base d'agent *SDAC*, une architecture SOA-SDAC a été proposée. Elle permet une surveillance distribuée multi-sites des systèmes de production et rend interopérable le modèle *SDAC* grâce à la notion de service SOA.

Au-delà de la mise en relation des clients et des producteurs, cette architecture masque à ces différents acteurs la complexité des réseaux d'applications à mettre en place pour la surveillance industrielle. Elle permet une coopération entre les différentes applications de surveillance en offrant un couplage lâche entre les modèles de gestion manipulés par ces applications, ainsi qu'un fonctionnement indépendant de la plate-forme d'implémentation et une possibilité de réutilisation des services.

Conclusion générale et perspectives

La surveillance des systèmes de production actuels est devenue de plus en plus un problème complexe. En effet, il faut prendre en considération plusieurs facteurs déterminants comme: besoin grandissant de protéger les vies et les biens, l'amélioration des performances et la productivité du système complexe et l'augmentation de leur fiabilité. Par ailleurs, les systèmes de surveillance expriment un grand besoin d'ouverture et de coopération à l'échelle mondiale. Ils ont besoin de s'allier à d'autres systèmes de compétences complémentaires afin de coopérer et renforcer l'exploitation sûre des systèmes industriels qui ne sont pas à la portée d'un seul système de surveillance.

Nous nous somme proposés dans ce travail de recherche de résoudre ces problèmes par la distribution de la surveillance et, plus précisément, le diagnostic en les décomposant en une multitude de problèmes élémentaires, en construisant des entités autonomes qui pourraient, en coopérant, participer à la construction d'une solution globale. En outre, nous avons proposé l'ouverture et la coopération inter systèmes de surveillance tout en assurant l'autonomie et la confidentialité des centres de surveillance.

Pour réaliser ce travail, nous avons commencé par étudier les architectures « systèmes » existantes pour déterminer celles qui s'adaptent le mieux à notre problématique. Nous nous sommes, par ailleurs, investis dans l'étude des méthodologies de modélisation en mettant l'accent sur la modélisation multi-agent et dans l'étude des architectures qui assurent l'interopérabilité des systèmes de surveillance multi-sites en mettant l'accent sur les architectures orientées services et les services web.

Notre analyse a pour but d'étudier les technologies permettant la conception et le développement d'un système de surveillance distribué interopérable en améliorant sa qualité, en temps normal comme en cas d'une perturbation. Nous nous sommes donc intéressés à l'intégration des agents logiciels qui fournissent des capacités sociales et les services web qui ont une infrastructure bien définie et interopérable pour améliorer l'adaptabilité, l'interopérabilité et l'ouverture d'un système de surveillance.

L'avènement de ces techniques a eu un impact significatif sur les architectures des systèmes de surveillance actuellement développées. Elles permettent des déploiements distribués en exhibant des solutions adéquates aux problèmes posés par les autres architectures conventionnelles. L'une des contributions majeure de ce travail réside dans la conception et la modélisation des différentes modèles de l'approche proposée pour la surveillance distribuée SDAC. Elle est fondée sur :

- l'utilisation conjointe de méthodes de surveillance issues de la communauté FDI pour la phase de détection et celles développées par la communauté DX pour l'étape de diagnostic afin de pouvoir tirer profit des avantages de chacune et avoir ainsi une certaine complémentarité
- la décomposition du système complexe en sous-systèmes plus facilement gérables.
- la conception de plusieurs agents dédiés chacun à une tâche particulière. Les ADL sont les plus importants. Chaque ADL a une vue locale du sous système à diagnostiquer.
- la proposition de deux approches pour le diagnostic quantitative et qualitative. La première suppose la présence des capteurs en entrées et sorties de chaque sous système et la deuxième se base sur la récupération des sorties globales du système complexe. Chacune propose des algorithmes décrivant les différents comportements de l'ADL.
- la coopération entre les différents agents ADL pour construire une solution globale, logique et cohérente en rassemblant les solutions partielles.
- L'intégration des services web concept de base de SOA pour gérer l'interopérabilité inter centre de surveillance pour optimiser la surveillance.

Nous avons également développé une application informatique pour supporter les aspects théoriques de SDAC :

- la première partie de cette application, AI- SDAC est une plateforme multi agent pour le diagnostic distribué développée en utilisant le langage de programmation Java sous l'environnement de développement Netbeans. L'implémentation du SMA est simplifiée par l'outil multi-agents JADE. Après avoir présenté notre cas d'étude, le système de clinkérisation de la cimenterie d'Ain Touta. Nous avons utilisé l'approche quantitative pour analyser et modéliser le sous système ligne de cyclones et l'approche qualitative pour analyser et modéliser le sous système four rotatif. La plateforme multi agent développée est capable d'assister l'utilisateur en fournissant des informations sur l'état de ces deux sous systèmes. Elle a été testée sur des données très proches de la réalité. Elle permet de partir d'une requête saisie par l'utilisateur provoquant des défauts et propose le diagnostic possible sous forme d'alarme. Les résultats obtenus nous ont permis de valider plusieurs aspects de notre SDAC.
- La deuxième partie présentée est une extension du travail précédent proposé pour régler les limites du modèle multi agent au niveau d'interopérabilité multi site. Ce modèle manque de mécanismes de localisation et de mise en relation des différents sites ainsi que de mécanismes de gestion des hétérogénéités existantes entre les

méthodes et modèles de surveillance utilisés dans ces différents sites. L'intégration de technologie SOA dans SDAC est notre solution pour gérer cette hétérogénéité et l'architecture SOA-SDAC résultat de cette intégration est l'une des contributions principale de ce travail. Elle permet une collaboration machine-machine entre les différentes applications de surveillance. En effet, elle offre un couplage faible entre les modèles de gestion manipulés par ces applications, ainsi qu'un fonctionnement indépendant de la plate-forme d'implémentation. Dans ce sens, la deuxième partie de l'application informatique est développée. Elle joue le rôle d'un fournisseur de services web pour la surveillance. Elle garde son propre mode de fonctionnement, ses technologies et publie certaines fonctionnalités et conserve les aspects privés du fonctionnement local. Elle est développée en utilisant le langage de programmation C# sous l'environnement de développement Visual Studio 2012 pour la surveillance distribuée de la ligne de cyclones de la cimenterie d'Ain Touta. Ce fournisseur publie les services web permettant la détection des comportements anormaux ou nominaux des cyclones. Ces services web peuvent être découverts et consommés par notre modèle multi agent en utilisant WSIG de JADE. Comme ils peuvent être découverts et consommés par n'importe quelle autre application liée par internet.

Plusieurs perspectives d'évolution et d'amélioration peuvent être envisagées si l'on prend en considération la multitude de domaines fondant notre travail. En effet, sur un plan fonctionnel, il serait intéressant d'améliorer la qualité de la surveillance distribuée (temps, coût) en améliorant le modèle multi agents proposé et rendant les agents plus adaptatifs i.e. chaque agent serait capable d'apprendre en fonction de son expérience passée et de son évolution. Ainsi les améliorations à court terme peuvent être au niveau de:

- l'agent d'information qui peut être conçu à base de CBR (raisonnement à partie de cas) ;
- l'agent de découverte doit classer les services web trouvés;
- doter l'agent d'évaluation d'intelligence en utilisant par exemple les réseaux de neurone.

Notre système multi agent de diagnostic est fondé sur la communication entre les différents agents qui le composent et les ADLs sont les plus important. Chaque ADL ne connaît qu'un sous-modèle partiel de l'installation et ignore complètement la représentation des autres sous-systèmes la principale difficulté réside dans la coordination

entre les ADL. Une extension de ce travail pourrait donc être constituée par la recherche d'un protocole de coordination permettant de minimiser le nombre de messages échangés.

Notre étude est fondée sur l'intégration des concepts du modèle SOA dans l'architecture proposée sans faire attention au sémantique des données échangées. Il est nécessaire que les différentes applications utilisées pour la surveillance puissent interpréter de la même manière les données échangées et/ou les fonctions réutilisées. Ainsi, Il est intéressant d'avoir une vision sémantique pour l'interopérabilité.

D'autre part, sur un plan technique, il serait sans doute effectivement fécond d'étudier la composition des services web déjà existants pour la construction de nouvelles fonctionnalités par le système multi-agent proposé pour doter le système de surveillance d'une flexibilité et d'une réactivité considérable qui permet en retour d'accélérer le temps de réponse de système de production surveillée vis-à-vis des changements de son environnement.

Références bibliographiques

[Afnor 2001] Norme AFNOR NF EN 13306, Terminologie de la maintenance, Ed. Afnor, Paris, 2001.

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

[Allem 2010] Khaled Allem, Ramdane Maamri et Zaidi Sahnoun,Un Modèle SMA pour le Diagnostic Collectif, COSI 2010, Ouargla, Algérie, 2010.

[Ardissono 2005] L. Ardissono, L. Console, G. Goy, A. Petrone, C. Picardi, M. Segnan et D. T. Dupré, Advanced fault analysis in web service composition, *In WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pp. 1090-1091, NY, USA, 2005.

[Arpaia 2010] P. Arpaia et C. Manna, Development of a Web-Service e-Diagnostic and e-Maintenance Framework for Industrial Monitoring, *17th Symposium IMEKO TC 4, 3rd Symposium IMEKO TC 19 and 15th IWADC Workshop Instrumentation for the ICT Era*, pp. 8-10, Kosice, Slovakia, September 2010.

[Aitouche 2013] S. Aitouche, Développement d'un système d'E-Management des connaissances pour l'amélioration des performances d'un système de production, Thèse de doctorat en Génie Industriel, Université de Batna 2, 2013.

[Andrews 2003] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S.Thatte, I. Trickovic et S. Weerawarana, Business Process Execution Language for Web Services (BPEL4WS)version, 2003.

[Basseville 1996] M. Basseville et M.O. Cordier, Surveillance et diagnostic de systèmes dynamiques: approche complémentaire du traitement de signal et de l'intelligence artificielle, Rapport de Recherche No2861, INRIA, 1996.

[Buhler 2002] P. Buhler and J. Vidal, Toward the synthesis of web services and agent behaviours, *In Proceedings of the Agent cities: Challenges in Open Agent Environments Workshop*, pp. 25-31, 2002.

[Bratman 1988] M.E. Bratman, D.J. Israel, and M. E. Pollack, Plans and resource-bounded practical reasoning, *Computational Intelligence*, Vol. 4 (Issue 4): 349–355, 1988.

[Baujard 1992] O. Baujard, Conception d'un environnement de développement pour la résolution de problèmes : Apport de l'intelligence Artificielle et

Application à la Vision, Thèse de Doctorat, Université de Joseph Fourier Grenoble I, 1992.

[Biteus 2005] J. Biteus, Distributed diagnosis and simulation based residual generators, Thèse de Doctorat, Université de Linköping, Sweden, 2005.

[Boukadi 2009] K. Boukadi, Coopération interentreprises à la demande : Une approche flexible à base de services adaptables, Thèse de Doctorat, École Nationale Supérieure des Mines de Saint-Étienne, France, 2009.

[Biswas 2004] G. Biswas, M. Cordier, J. Lunze et L. Travé-Massuyès, Diagnosis of complex systems: Bridging the methodologies of the FDI and DX communities, *IEEE Transactions on Systems, Man and Cybernetics - Part B*, Special section, vol. 34 (Issue 5): 2159–2244, 2004.

[Combacau 1991] M. Combacau, Commande et Surveillance des Systèmes a Évènements Discrets Complexes : Application aux Ateliers Flexibles, Thèse de Doctorat, Université Paul Sabatier, Toulouse, 1991.

[Ciancarini 1996] P. Ciancarini, Coordination models and languages as software integrators, *ACM Computing Surveys*, 1996.

[Chouhal 2011] O. Chouhal, H.L. Mouss et R. Mahdaoui, Multi agent system using web services for diagnosing Manufacturing Systems, *International Review of Mechanical Engineering*, Vol. 05(Issue 3): 1156-1160, 2011.

[Cheaib 2010] N. Cheaib, Contribution à la malléabilité des collecticiels : une approche basée sur les services web et les agents logiciels, Thèse de Doctorat, Université d'Evry val d'Essonne, 2010.

[Caire 2003] G. Caire, JADE Tutorial: JADE programming for beginners, TILab, formerly CSELT, 2003.

[Cordier 2004] M.O. Cordier, P. Dague, F. Lévy, J. Montmain, M. Staroswiecki et L. Travémassuyès, Conflicts Versus Analytical Redundancy Relations: A comparative Analysis of the Model Based Diagnosis Approache From the Artificial Intelligence and Automatic Control Perspectives, *Special Issue of the IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 34 (Issue 5): 2163-2177, 2004.

[Dubuisson 2001] B. Dubuisson, E. Boutleux, P. Dague, T. Denoeux, E. Didelet, Y. Gandvalet et M. Masson, Diagnostic, Intelligence Artificielle et reconnaissance de formes, Ed. Hermes, 2001.

[Drogoul 1999] A. Drogoul et J. Meyer, Intelligence artificielle située. *Hermès Paris Science Publications*, 1999.

[David 1988] R. Davis et R. G. Smith, Negotiation as a Metaphor for distributed

problem solving. In A. H. Bond, L. Gasser, editors, Reading in Distributed Artificial Intelligence, pp. 333-356, Morgan Kaufmann, 1988.

[Demagh 2013] Y. Demagh, modélisation et simulation des transferts de matière dans les fours de cimenterie, thèse de doctorat, université de jijel, 2013.

[De Kleer 1987] J. De Kleer et B. C. Williams, Diagnosing multiple faults, *Artificial Intelligence*, Vol.32, pp. 97-130, 1987.

[De Kleer 1989] J. De Kleer et B. C. Williams, Diagnosing with behavioral modes, *In Proceeding Of the 11th International Joint Conference on Artificial Intelligence, IJCAI-89*, Detroit, MI, USA, pp. 1324-1330, 1989.

[De Kleer 1992] J. De Kleer, A. Mackworth et R. Reiter, Characterizing diagnoses and systems, *Artificial Intelligence*, Vol. 56 (Issue 3): 197-222, 1992.

[El-Sharkawi 2013] A. El-Sharkawi, A. Shouman et S. Lasheen, Service Oriented Architecture for Remote Sensing Satellite Telemetry Data Implemented on Cloud Computing, *I.J. Information Technology and Computer Science*, Vol. 5 (Issue 7): 12-26, June 2013.

[Elfallah 2004] A. Elfallah, S. Haddad, T. Melitti et A. Suna, Interopérabilité des systèmes multi-agents à l'aide des services Web, *JFSMA*, 2004.

[Frank 1996] P.M. Frank, Analytical and qualitative model-based fault diagnosis – a survey and somme new results, *European Journal of Control*, Vol. 2,pp. 6-28, 1996.

[Froehlich 1996] P. Froehlich et W. Nejdl, Resolving conflicts in distributed diagnosis, *ECAI'96, 12th European Conference on Artificial Intelligence, John Wiley & sons, Ltd.*, 1996.

[Fabre 2002] E. Fabre, Monitoring distributed systems with distributed algorithms, *In Proceedings IEEE, Conference, on Decision and Control*, pp.411-416, Las Vegas, Dec. 2002.

[Foukarakis 2007] I. Foukarakis, A. Kostaridis, C. Biniaris, D. Kaklamani, et I. Venieris, Webmages: An agent platform based on web services, *Computer Communications*, Vol. 30 (Issue 3): 538-545, 2007.

[Ferber 1995] J. Ferber, Les systèmes multi-agents Vers une intelligence collective, *InterEditions*, Paris. 1995.

[FIPA 1999] FIPA, FIPA Specification, *FIPA Association*, 1999.

[Ferber 1998] J. Ferber et O. Gutknecht, A meta-model for the analysis and design of organizations in multi-agent systems, *In Proceedings of the third international conference on multi-agent systems*, 1998.

[Ghomari 2008] A. R. Ghomari, Approche Méthodologique d'Acquisition de

Connaissances Agrégées à base d'Agents cognitifs coopérants pour les systèmes d'aide à la décision stratégiques, Thèse de Doctorat, Ecole nationale Supérieure en Informatique, Algérie, 2008.

[Gertler 1988] J. Gertler, Survey of Model-Based failure detection and isolation in complex plants, IEEE Control Systems Magazine, Vol. 11, pp. 3-11, 1988.

[Finin 1995] T. Finin, Y. Labrou, et J. Mayfield, KQML as an agent communication language, J. Bradshaw, Ed. Software Agents, MIT Press, Cambridge, 1995.

[Hamscher 1992] W. Hamscher, L. Consol, J. De Kleer, Readings in Model-Based Diagnosis, *Morgan Kaufmann*, San Mateo CA, USA, 1992.

[Issury 2011] I. Issury, Contribution au développement d'une stratégie de diagnostic global en fonction des diagnostiqueurs locaux - Application à une mission spatiale -, Thèse de Doctorat, Université Bordeaux I, 2011.

[Ishak 2010] K. Ishak, Architecture distribuée interopérable pour la gestion des projets multi-sites: Application à la planification des activités de production, Thèse de Doctorat, INP Toulouse, 2010.

[Iglesias 1998] C. A. Iglesias, M. Garijo, J. C. Gonzalez et J. R. Velasco, Analysis and Design of Multiagent Systems using MAS-commonKADS , M.P Singh, A. Rao et M. J. Wooldridge, Editeurs: Intelligent Agents IV: Agent Theories, Architectures and Languages, vol. 1365 de LNAI (Lecture Notes in Artificial Intelligence), Springer-Verlag, 1998.

[Isermann 1997] R. Isermann et P. Balle, Trends in the application of model-based fault detection and diagnosis of technical processes, *Control Engineering Practice*, Vol. 5(Issue5): 709-719, 1997.

[Jennings 1998] N. R. Jennings, K. Sycara, M. Wooldridge, A Roadmap of Agent Research and Development, *International Journal of Autonomous Agents and Multi-Agent Systems*, vol.1(Issue 1): 7-38, 1998.

[Jennings 2000] N. R. Jennings, S. Parsons, C. Sierra, et P. Faratin, Automated negotiation, *In 5th International Conference on The Practical Application of Intelligent Agent and Multi-Agent Systems (PAAM-2000)*, Manchester, UK, 2000.

[Kadri 2013] O. Kadri, L'application des algorithmes de colonies de fourmis pour le diagnostic des systèmes dynamiques et complexes, Thèse de doctorat en Génie Industriel, Laboratoire d'Automatique et productique, Université de Batna 2, Algérie, 2013.

[Lefebvre 2000] D. Lefebvre, Contribution à la modélisation des systèmes dynamiques à événements discrets pour la commande et la surveillance, Habilitation à

Diriger des Recherches, Université de Franche Comté/ IUT Belfort – Montbéliard, 2000.

[Leitão 2001] P. Leitão, F. Restivo, An Agile and Cooperative Architecture for Distributed Manufacturing Systems, *In Proceedings of Robotics and Manufacturing'2001 International Conference*, Cancun, Mexico, pp. 21-24 Mai 2001.

[Liu 2002] Y.J. Liu, De la nécessité et de la façon de coopérer, de s'autoorganiser et de se reconfigurer dans des systèmes de production complexes : modélisation et gestion d'un système virtuel pour une approche multi-agents, Thèse de Doctorat en Génie Industriel, INPG, France, 2002.

[Lopes 2005] D. C. P. LOPES, Étude et applications de l'approche MDA pour des plates-formes de Services Web, Thèse de Doctorat à l'UFR Sciences et Techniques, Université de Nantes, France, 2005.

[Lee 2014] S. Lee, J. Jo et Y. Kim, Environmental sensor monitoring with secure restful web service, *International Journal of Services Computing*, Vol. 2(Issue 3): 30-42, July- September 2014.

[Muller 2008] A. Muller, A. C. Marquez, B. Iung, On the concept of e-maintenance: review and current research, *Reliability Engineering and System Safety*, Vol. 93 (Issue 8): 1165-1187, 2008.

[Muller 2005] A. Muller, contribution à la maintenance prévisionnelle des systèmes de production par la formalisation d'un processus de pronostic. Thèse de Doctorat, Université Henri Poincaré, Nancy, juin 2005.

[Monnin 2004] M. Monnin, N. Palluat, D. Racoceanu et N. Zerhouni,Diagnosis methods using artificial intelligence, application of fuzzy petri nets and neuro-fuzzy systems, *Third Conference on Management and Control of Production and Logistics, MCPL2004*, Santiago de Chile, 2004.

[Mouss 2006] M.D Mouss. Diagnostic et conduite des systèmes de production par approche à base de connaissances, Thèse de doctorat en Génie Industriel, Laboratoire d'Automatique et productique, Université de Batna 2, Algérie, 2006.

[Murthy 2009] V. Murthy et E. Krishnamurthy, Multi set of agents in a network for simulation of complex systems. *Recent advances in Nonlinear Dynamics and synchronization, Theory and applications*, pp. 153-200 Springer Berlin Heidelberg, 2009.

[Maamar 2003] Z. Maamar, Q. Z. Sheng, et B. Benatallah, Interleaving web services composition and execution using software agents and delegation, *AAMAS*

Workshop, 2003.

[Matskin 2005] M. Matskin, P.Kungas, J. Rao, J. Sampson et S. Petersen, Enabling web services composition with software agents, *In Proceedings of the Ninth IASTED International Conference on Internet and Multimedia Systems and Applications, IMSA* , Hawaii, USA, pp. 93-98, 2005.

[Minar 1996] N. Minar The swarm simulation system: a toolkit for building multi-agent simulations, 1996, http://www.santafe.edu/project/swarm

[Nickull 2005] D. Nickull, Service Oriented Architecture (SOA) and Specialized Messaging Patterns, Adobe System Incorporated, technical white paper, 2005.

[Núnez 2005] D. B. Núnez, Automation of distributed model-based diagnosis using structural analysis, Thèse de Doctorat, Université de Sevilla, Espagne, 2005.

[O'Hare 1996] G. M. P. O'Hare et N. R. Jennings, Foundations of Distributed Artificial Intelligence, *Wiley-Interscience*, 1996.

[Pencolé 1999] Y. Pencolé, Decentralized diagnoser approach: applications to telecommunication networks, *Proceedings Of DX'2000, Eleventh International Workshop on Priciples of Diagnosis*, pp. 1762-1768, Décembre 1999.

[Pencolé 2005] Y. Pencolé et M. O. Cordier, A formal framework for the decentralized diagnosis of large scale discrete event systems and its application to telecommunication networks, *Artificial Intelligence*, Vol. 164 (Issue 2): 121-170, 2005.

[Picard 2004] G. Picard, Méthodologie de développement de systèmes multi-agents adaptatifs et conception de logiciels à fonctionnalité émergente, Thèse de Doctorat, Université Toulouse III, 2004.

[Pala 2014] S. Pala, L. Fumagalli, M. Garetti et M. Macchi, Smart sensors for condition based maintenance: a test case in the manufacturing industry, *Second European Conference of the Prognostics and Health Management Society*, June 2014.

[Patton 1991] R. J. Patton et J. Chen, A review of parity space approaches to fault diagnosis", in IFAC Safe process symposium, Baden-Baden, Allemande, 1991.

[Reaidy 2003] J. Reaidy, étude et mise en œuvre d'une architecture d'agents en réseau dans les systèmes dynamiques situés : pilotage des systèmes de production complexes .Thèse de Doctorat en Génie Industrielle, Nîmes, 2003.

[Reiter 1987] R. Reiter, A theory of diagnosis from first principles, Artificial *Intelligence*, Vol. 32(Issue 1): 57-96, 1987.

[Resceanu 2008] I. Resceanu, M. Niculescu, N.G. Bizdoaca et C. Pana, remote monitoring and diagnosis of a mechatronic system, *8th wseas international conference on applied informatics and communications*, pp. 20-22, Rhodes, Greece, August 2008.

[Rebeuf 2004] X. Rebeuf, N. Blanc, F. Charpillet, D. Cheve, A. Dutech, C. Lang, L. Pélissier et J.P. Thomesse, Proteus, des web services pour les systèmes de maintenance, 2004. http://www.proteus-iteaproject.com.

[Starr 1996] B. Starr, M. Ackerman, et M. Pazzani, Do-I-Care: A collaborative web agent, *Conference on Human Factors in Computing Systems*, pp. 273-274, New York, USA, 1996.

[Smith 1980] R.G. Smith, The contract Net protocol: hight-level communication and control in a distributed problem solver, *IEEE Transactions on Computers*, Décembre 1980.

[Schreiber 1994] G. Schreiber, B. Wielinga, H. Akkermans, W. Van de Velde, A. Anjewierden , CML: the CommonKADS Conceptual Modelling Language, *Proceedings EKAW'94*, Vol. 867 pp. 1-25, Springer-Verlag, Heidelberg, 1994.

[SOAP 2003] SOAP, The Simple Object Access Protocol, 2003. http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[Shao 2013] Y. Shao, Z. Lv ,L. Deng et A. Wang, Design of Remote Fault Monitoring System Based on WAMP Press, *3rd International Conference on Electric and Electronics*, Hong Kong ,December 2013.

[Touaf 2005] S. Touaf, Diagnostic logique des systèmes complexes dynamiques dans un contexte multi-agent, Thèse de Doctorat, Université Joseph Fourier, Grenoble 1, 2005.

[Tan 2011] V.V. Tan et M.J. Yi, Development of an OPC Client-Server Framework for Monitoring and Control Systems, *Journal of Information Processing Systems*, Vol. 7 (Issue 2): 321-340, June 2011.

[UDDI 2003] UDDI, The Universal Description, Discovery and Integration protocol, 2003. http://www.uddi.org/

[Vernadat 1996] F. Vernadat, Enterprise modelling and integration: Principles and applications, Chapman & Hall, London, 1996.

[Wooldridge 1995] M. Wooldridge et N .R. Jennings, Agent theories, architectures, and languages: a survey, *Proceedings of the Workshop on Agent Theories,*

Architectures, and Languages on Intelligent Agents (ATAL'95), pp. 1-39, Springer Verlag New York, 1995.

[Wooldridge 2000] M. Wooldridge, N. R. Jennings et D. Kinny, The Gaia Methodology for Agent-Oriented Analysis and Design, *Journal of Autonomous Agents and Multi-Agent Systems*, Vol. 3(Issue 3): 285–312, 2000.

[W3C 2004] W3C, Web Services Glossary, 2004. http://www.w3.org/TR/ws-gloss.

[Wang 2007] C. Wang, L. Xu, et W. Peng, Conceptual design of remote monitoring and fault diagnosis systems, *Information Systems*, Vol. 32 (Issue 7): 996–1004, 2007.

[Zwingelstein 1995] G. Zwingelstein, Sûreté de fonctionnement des systèmes industriels complexes, ENSEEIHT de Toulouse, 1995.

[Zemouri 2003] R. Zemouri, Contribution à la surveillance des systèmes de production à l'aide des réseaux de neurones dynamiques : Application à la e-maintenance, Thèse de Doctorat, Université de Franche Comté, Besançon, 2003.

[Zennir 2004] Y. Zennir, Apprentissage par renforcement et systèmes distribués: Application à l'apprentissage de la marche d'un robot hexapode. Thèse de Doctorat, INSA de Lyon, 2004.

[Zhao 2010] F. Zhao , J. Chen , G. Dong et L. Guo, SOA-based remote condition monitoring and fault diagnosis system, *International Journal of Advanced Manufacturing Technology*, Vol. 46 (Issue 9): 1191-1200, Springer Verlag, London, 2010.

Production scientifique

Publications dans des revues internationales

1. **O. Chouhal**, H.L. Mouss, K. Benagoune and R. Mahdaoui, A Multi-Agent Solution to Distributed Fault Diagnosis of Preheater Cement Cyclone, Journal of Advanced Manufacturing Systems, World scinetific Publisher, Vol. 15, Issue 04, December 2016.
2. **O. Chouhal**, H.L. Mouss, R. Mahdaoui, and M. D. Mouss, A web services based multi agent system for the diagnosis of industrial plants, International Review of Mechanical Engineering(IREME), Vol.5(Issue 6), pp1156-1160, May 2011, ISSN 1970 - 8734, 2011.
3. R. Mahdaoui, H.L. Mouss et M. D. Mouss, **O. Chouhal**, The Temporal Neuro-Fuzzy Systems Learning Using Artificial Immune Algorithm. International Review of Mechanical Engineering (IREME), Vol.4, Issue.1, pp 918-922, May 2012, ISSN 1970 - 8734, 2012.

Communications dans des conférences internationales

1. R. Mahdaoui, H. Mouss, **O. Chouhal**, Reconnaissances de formes par les systèmes neuro-flous temporels : Application au diagnostic Industriel, MCSEAI, Maghebian Conference on Information Technologies, Oran, Algérie 28 au 30 Avril 2008.
2. R. Mahdaoui, H. Mouss et **O. Chouhal**, Surveillance industrielle dynamique par les systèmesneuro-flou: Application à un système de production, CIFA'2008 Conférence Internationale Francophone d'Automatique, Bucarest, Roumanie, 03 au 05 Septembre 2008.
3. R. Mahdaoui, H. Mouss, D. Mouss, **O. Chouhal**, Hybridation des systèmes Neuro Flousetla recherche Tabou pour la reconnaissance des formes, ICISP'09 International Conference on Systems and Processing Information Guelma, Algérie du 02 au 04 Mai 2009.
4. R. Mahdaoui, H. Mouss, D. Mouss, **O. Chouhal**, Surveillance industrielle dynamique par lessystèmes neuro-flous Temporels : Application a un système de production, ICISP'09International Conference on Systems and Processing Information Guelma, Algérie du 02 au 04 Mai 2009.
5. R. Mahdaoui, H. Mouss, D. Mouss, **O. Chouhal**, Apprentissage des systèmes neuro-flous par des données numériques et/ou symboliques pour la reconnaissance

dynamique de formes: Application au diagnostic industriel, ICIEM'10 International Conference on Industrial Engineering and Manufacturing. Batna–Algeria du 09 au 10 Mai 2010.