Remémoration guidée par l'adaptation et maintenance des systèmes de diagnostic industriel par l'approche du raisonnement à partir de cas.

Mohamed Karim Haouchine

Année : **2009**

THESE

présentée à

L'UFR des Sciences et Techniques
de l'Université de Franche-Comté

pour obtenir le

GRADE DE DOCTEUR DE L'UNIVERSITE DE FRANCHE-COMTE

en Automatique
(Ecole Doctorale Sciences Physiques pour l'Ingénieur et Microtechniques)

Remémoration guidée par l'adaptation et maintenance des systèmes de diagnostic industriel par l'approche du raisonnement à partir de cas

par

Mohamed Karim Haouchine

Soutenue le 23 septembre 2009 devant la Commission d'examen :

Rapporteurs :	Luc Lamontagne	Professeur, Université Laval
	Laurent Geneste	Professeur, ENIT
Examinateurs :	Belkacem Ould-Bouamama	Professeur, Université Lille
	Sylvie Despres	Professeur, Université Paris 13
Directeur de thèse :	Noureddine Zerhouni	Professeur, ENSMM
	Brigitte Chebel-Morello	Maître de conférences, Université de Franche-Comté

Remerciements

Le présent manuscrit est le résultat d'un long travail réalisé grâce à de nombreuses personnes qui y ont contribués de près ou de loin et que je tiens tout particulièrement à remercier.

En premier lieu je remercie ma mère qui fut présente à mes cotés, m'a encouragée et accompagnée tout au long de ces années passées au laboratoire d'automatique de Besançon (LAB), actuellement département AS2M (Automatique et systèmes Micro-Mécatroniques). Je remercie tout naturellement mon père, mes grand-mères ainsi que tous les membres de ma grande famille, grâce à qui mes années de travail se sont déroulées dans de bonnes conditions psychologiques. J'associe très chaleureusement Sophie à mes remerciements car elle a su me supporter, me gérer et surtout me gâter lors des nombreux week-ends et soirées de travail.

J'adresse toute ma reconnaissance à Monsieur Noureddine Zerhouni, qui m'a accueilli et dirigé mes travaux au sein de l'équipe Systèmes de Maintenance Intelligents (SyMI). Je le remercie pour sa disponibilité, son précieux soutien, ses valeureux conseils, la confiance qu'il a placé en moi et, et surtout, pour ses qualités humaines.

J'exprime toute ma gratitude à Madame Brigitte Chebel-Morello qui sans elle, ce travail n'aurait pu voir le jour. Je la remercie pour sa présence, sa disponibilité, sa sincérité, ses compétences, son soutien, ses orientations et particulièrement son amitié chaleureuse et son apport maternant.

Je remercie Monsieur Luc Lamontagne, Professeur à l'Université de Laval « Québec » et Monsieur Laurent Geneste, Professeur à l'ENIT de Tarbes pour m'avoir fait l'honneur de rapporter ce travail et d'être venus de si loin pour assister à ma soutenance.

Je remercie également Madame Sylvie Despres, Professeur à l'Université Paris 13 et Monsieur Belkacem Ould-Bouamama, Professeur à l'Université des Sciences et Technologies de Lille d'avoir examiné ce travail et d'y avoir apporté de précieuses remarques.

Je remercie Monsieur Alain Bourjault et Monsieur Nicolas Chaillet, directeurs successifs du département AS2M de l'institut FEMTO-ST, qui ont contribués aux excellentes conditions de travail dans lesquelles se sont déroulées mes recherches. Je remercie très sincèrement l'ensemble du personnel pour son aide précieuse ainsi que pour l'excellente ambiance qui a régnée dans ce département. Je ne peux oublier de remercier « l'équipe de choc » que sont mes collègues et qui au fil du temps, sont devenus mes amis.

Enfin, un petit clin d'œil à mon quartier natal « Patrimoine » à Constantine et à tous mes amis, qui sans les nommer, et ils sont nombreux, se reconnaîtront.

Karim

Table des matières

Table des figures

Introduction générale

Contexte et problématique

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. La fonction maintenance tient une position stratégique dans l'organisation de l'entreprise et répond à des besoins pour maîtriser techniquement et économiquement des équipements industriels. Les services « maintenance », qui s'occupent de cette fonction, interviennent pour maintenir, en condition opérationnelle ou remettre en état de bon fonctionnement, tout équipement quelque soit sa nature. Deux types d'approche de maintenance en découlent : la maintenance *préventive* (approche programmée) et la maintenance *corrective* (approche réactive). La maintenance préventive est exécutée à des intervalles prédéterminés ou selon des critères prescrits, et est destinée à réduire la probabilité de défaillance ou de dégradation du fonctionnement du bien. Quant à la maintenance corrective, elle est exécutée après détection d'une panne et est destinée à remettre un bien dans un état dans lequel il peut accomplir une fonction requise.

Les travaux actuels ont démontré que ces deux approches de maintenance se sont révélées excessivement coûteuses dans leur application, notamment dans les systèmes complexes dans lesquels le cycle de vie est relativement long. Ces coûts correspondent à la non maîtrise des opérations annexes telles que la gestion des pièces de rechange, l'obsolescence associée et la perte de production induite.

Dans le cadre de notre étude, nous nous intéressons à la maintenance corrective et plus particulièrement au diagnostic de pannes sur les équipements industriels. Le diagnostic, dans le contexte industriel, est réalisé grâce à des systèmes d'aide à la décision qui soulèvent généralement des problèmes d'adaptabilité, de flexibilité et d'implantation dans des environnements opérationnels. Il devient donc impératif pour les membres responsables du

service maintenance de réagir en mettant à disposition des outils d'aide puissants, rapides, efficaces et capables de faire face aux contraintes de production. Ces outils doivent donc répondre aux exigences de la fonction maintenance et doivent également permettre de capitaliser un savoir-faire de plus en plus précieux.

Cadre de travail

L'objectif de cette thèse est de créer un outil d'aide au diagnostic industriel sur des équipements complexes et non pas sur des processus industriels continus. Cet outil est en lien avec la maintenance corrective afin de faire de celle-ci une source de profit pour l'industrie.

Cet outil doit être conçu en tenant compte de :

- la modélisation de l'expérience
  - comment l'outil va représenter et formaliser l'expérience terrain ?
- la réutilisation de l'expérience rencontrée
  - comment l'outil va sélectionner, évaluer et adapter l'expérience rencontrée au contexte afin de pouvoir la réutiliser ?
- et l'exploitation des connaissances du système
  - comment l'outil pourra-t-il prendre en compte de nouvelles expériences ?

En effet, nous voulons développer une méthode de diagnostic qualitative notamment orientée expériences. Cela nous amène à choisir l'outil de formalisation de l'expérience terrain : le raisonnement à partir de cas (RàPC). En effet, le RàPC est un outil d'intelligence artificielle, considéré comme l'outil privilégié de modélisation de l'expérience des utilisateurs et d'apprentissage incrémental de ces expériences. Les systèmes de RàPC ont pour fonction de capitaliser l'expertise terrain sous forme de connaissances dans sa mémoire, de pouvoir raisonner dans un domaine à connaissance réduite, de réutiliser des connaissances analogues pour prendre une décision et d'enrichir la mémoire en ajoutant de nouvelles connaissances d'une façon dynamique.

Parmi les différentes applications possibles dans les systèmes de RàPC, allant de la classification à la gestion des connaissances, nous avons développé deux types de systèmes, à savoir un système orienté *mining* s'apparentant à une classification et un système de diagnostic orienté *connaissances.*

Dans le premier type de système, nous comptons développer :

- une représentation triviale du cas ne nécessitant pas de connaissances, un cas étant un exemple traite tout type de valeur ;
- une phase de remémoration prenant appui sur une mesure tenant compte des différentes valeurs d'attributs et de leur présence.

Cependant, notre système orienté mining peut engendrer une explosion combinatoire de cas. Par conséquent, il nécessite une phase de maintenance de la base de cas qui se déclinera en :

- une méthode de structuration de la base de cas prenant appui sur des critères de qualité en l'occurrence : la compétence et la performance ;
- un algorithme d'auto-incrémentation permettant d'intégrer les cas dans la base de cas tout en maintenant sa structuration.

Ensuite, nous nous intéressons à un système de diagnostic raisonnant sur des modèles de connaissances obtenus en appliquant la démarche experte utilisée sur le terrain, à savoir l'analyse fonctionnelle et dysfonctionnelle de l'équipement industriel.

Nous comptons développer les points suivants :

- une représentation de cas plus complexe orientée objet mais qui permet de raisonner sur un ensemble restreint de cas. Cette représentation est issue d'une étude dysfonctionnelle, fondée sur la définition du diagnostic ;
- deux modèles de connaissances : un modèle taxonomique regroupant les composants en familles fonctionnelles et un modèle de contexte décrivant les relations entre les composants de l'équipement industriel à diagnostiquer ;

Toutefois, ce type de système exige de porter une attention particulière au cours des phases de remémoration et d'adaptation afin d'expliciter les cas génériques de la base de cas. De ce fait, nous comptons développer :

- une phase de remémoration guidée par l'adaptation qui s'appuie sur deux mesures de similarité et d'adaptation ;
- un algorithme d'adaptation basé sur les relations de dépendance dédié au diagnostic industriel.

Organisation du mémoire

Ce mémoire de thèse est articulé en cinq chapitres.

Le premier chapitre introduit les notions du diagnostic et les principes fondamentaux du RàPC ainsi que la relation entre le diagnostic et le RàPC. Le RàPC s'appuie sur des connaissances représentant l'ensemble des notions et des principes acquis par l'étude, l'observation ou l'expérience. La modélisation des connaissances passe par l'expertise terrain qui est représentée sous forme de cas. Le cas est manipulé par un cycle comportant différentes phases pour résoudre des problèmes et/ou acquérir de nouvelles connaissances. La connaissance du système de RàPC est organisée sous forme de « containers de connaissances » qui sont associés aux phases du cycle et à la base de cas contenant tous les cas du système. Les différentes notions et principes abordés vont nous permettre de disposer de critères de comparaison entre les systèmes de diagnostic par RàPC que nous étudions au chapitre suivant.

Le deuxième chapitre présente un état de l'art des systèmes de diagnostic basés sur l'approche de RàPC, suivi par une analyse des techniques utilisées dans chaque phase du cycle de ces systèmes. Nous présentons nos choix et nos démarches adoptés pour la mise en place de notre système d'aide au diagnostic par RàPC. Nous mettons en évidence le type de représentation des connaissances qui se répercutera sur le type de système manipulé. Le premier type est un *système orienté mining* dans lequel le cas dispose d'une représentation triviale et contient toute la connaissance du système. Nous développons une phase de remémoration qui est basée sur l'algorithme des k plus proches voisins. Ce type de système nous permet de nous affranchir des problèmes de modélisation des connaissances. Toutefois, il peut engendrer une explosion combinatoire de cas, ce qui nous conduit à proposer une phase de maintenance de la base de cas que nous aborderons au chapitre 3. Le deuxième type est un *système orienté connaissances* qui est associé aux modèles de connaissances du domaine et dans lequel la représentation des cas est générique et orientée objet. De ce fait, nous proposons une représentation du cas à partir d'une analyse dysfonctionnelle formalisant les caractéristiques du diagnostic, associée à des modèles de connaissances issus d'une analyse fonctionnelle de l'équipement à diagnostiquer. La généricité des cas demande un développement particulier de la phase d'adaptation. Par conséquent, nous accordons une

attention particulière à cette phase, au chapitre 4, ainsi qu'à la phase de remémoration parce qu'elles sont liées l'une à l'autre.

Le troisième chapitre est consacré aux systèmes orientés mining. Plusieurs travaux ont été réalisés dans ce cadre afin de maintenir ces systèmes dans un bon état opérationnel. En effet, ces systèmes, arrivés à maturité, nécessitent une phase de maintenance pour garantir une certaine qualité. La maintenance de ces systèmes correspond à celle des containers de connaissances. Toutefois, la plupart des travaux considère que la base de cas est le container de connaissances le plus sensible aux changements. Sa consultation est la plus appropriée pour déclencher des opérations de maintenance. Par conséquent, nous réalisons un état de l'art sur la maintenance de la base de cas (MBC). Nous présentons les critères d'évaluation de la qualité de la base de cas, les principales politiques et stratégies existantes. Nous enchaînons par une proposition d'une méthode de maintenance de la base de cas qui la structure et l'auto-incrémente. En effet, nous nous inspirons des différentes stratégies existantes afin de structurer au mieux la base de cas. Ensuite, cette dernière est amenée à évoluer dans le temps par l'introduction de nouveaux cas qui peuvent altérer sa qualité. De ce fait, nous proposons une méthode d'auto-incrémentation qui apprend les cas d'une manière dynamique tout en respectant les critères de qualité de la base de cas et sa structuration. Malgré ces phases, à long terme un ensemble conséquent de cas peut détériorer le système et nécessiterait une modification de l'algorithme de recherche de cas. Nous avons donc orienté nos recherches vers des systèmes de RàPC orientés connaissances.

Le quatrième chapitre aborde les systèmes orientés connaissances dans lesquels la représentation des cas est générique et plus complexe. Nous nous intéressons particulièrement aux phases de remémoration et d'adaptation. Ainsi, nous présentons un état de l'art sur les différents types de remémorations existants ainsi que les techniques d'adaptation utilisées. Nous constatons que ces deux phases sont liées. Par conséquent, nous développons notre méthode de remémoration guidée par l'adaptation en prenant appui sur la formalisation du cas de diagnostic associé aux modèles de connaissances. La liaison entre les deux phases est caractérisée par deux mesures de remémoration (M_R) et d'adaptation (M_A). Cette dernière permet de sélectionner le cas le plus facilement adaptable. Nous proposons également un algorithme d'adaptation qui prend appui sur les relations de dépendance entre l'espace problème et l'espace solution du cas. Enfin, nous appliquons notre méthode sur un équipement industriel en l'occurrence un moteur à explosion de la société « Renault ».

Le cinquième chapitre étudie la mise en place des deux types de système de diagnostic par RàPC sur l'équipement industriel SISTRE (Supervised Industrial System of pallets TransFer). Nous déployons les différentes représentations utilisées et nous appliquons nos différentes contributions développées dans les phases du cycle.

Chapitre 1

Diagnostic et approche du Raisonnement à Partir de Cas (RàPC)

1. Introduction

Ces dernières années, la maintenance industrielle a connu des mutations profondes et a été transformée d'un centre de coûts en un centre de profits. Ainsi, elle participe à la compétitivité de l'entreprise dans un milieu concurrent. Le cadre d'étude de cette thèse se situe dans la maintenance des systèmes complexes industriels. La maintenance est une fonction regroupant deux grandes classes d'activités : les activités relatives à sa gestion, à son organisation ainsi qu'à ses aspects techniques [Piechowiak, 2003]. Nous nous intéressons à cette deuxième classe d'activités qui comporte des tâches de *prévention*, de *dépannage* et de *diagnostic*. Cette dernière tâche fera l'objet de notre étude.

De nombreux travaux ont porté sur le diagnostic et il existe une pluralité de méthodes dans ce domaine. Parmi ces méthodes, nous nous intéressons à l'approche par raisonnement à partir de cas (RàPC). Ce choix est justifié par le fait que le RàPC est une approche qui permet un raisonnement sur un ensemble restreint de données et de connaissances du domaine, qui s'enrichit au fur et à mesure grâce à un processus d'apprentissage et s'affranchit de la phase de collecte de données d'apprentissage qui pose problème aux industriels.

A la section 2, nous donnons les notions de base concernant le diagnostic ainsi que la relation diagnostic-RàPC. A la section 3, nous présentons les principes fondamentaux du RàPC qui vont nous permettre d'une part, de nous familiariser avec les différents termes utilisés dans le cadre de notre étude et d'autre part, d'introduire les bases des différentes phases de développement d'un système de RàPC. En effet, le RàPC s'appuie sur des connaissances représentant l'ensemble des notions et des principes acquis par l'étude, l'observation ou l'expérience. Il permet la manipulation des connaissances qui a pour but de résoudre les problèmes en retrouvant des situations analogues modélisées dans sa base de connaissances et en les adaptant à la situation considérée. La modélisation des connaissances passe par l'expertise terrain qui est représentée sous forme de cas. Dans la même section, nous abordons la structuration et la représentation du cas. Cette représentation se répercute sur le type de modèle de RàPC manipulé. Nous présentons les différents modèles de RàPC qui existent dans la littérature ainsi que le modèle le plus utilisé : le modèle structurel. Cette étude nous aide à choisir le type de modèle que nous exploitons lors de la mise en place de notre système de RàPC.

Les cas sont rangés dans une mémoire appelée « base de cas ». La base de cas se trouve au centre du cycle du RàPC. Ce cycle met en œuvre le raisonnement par analogie et est doté

de plusieurs phases qui ont pour rôle de manipuler les connaissances du système afin d'atteindre les objectifs fixés tels que la résolution du problème et/ou l'acquisition de nouvelles connaissances. Nous abordons les spécificités des différentes phases auxquelles nous associons le principe du carré d'analogie

Tous les systèmes de RàPC manipulent des connaissances connues sous le nom de « *containers de connaissances* » que nous présentons à la section 4. Enfin, le RàPC peut s'appliquer dans plusieurs domaines. Cependant, son utilisation est soumise à certaines conditions détaillées à la section 5.

2. Le diagnostic

Avant de définir le diagnostic, nous introduisons quelques notions de bases qui l'exploitent. Selon l'AFNOR (Association Française de NORmalisation) dans [Afnor, 2001], un ***bien*** est tout élément, composant, mécanisme, sous-système, unité fonctionnelle, équipement ou système qui peut être considéré individuellement. Ces biens représentent des équipements industriels qui sont constitués de plusieurs composants susceptibles d'être défaillants voir tomber en panne.

Un ***composant*** est un élément discret d'un équipement. C'est l'unité de description minimale de l'analyse systémique. C'est l'élément de base qui est interconnecté avec d'autres composants de même nature ou de nature différente dans un espace mitoyen ou lointain.

Le domaine du diagnostic fait référence aux notions d'observation, de panne, de défaillance et de symptôme [Piechowiak, 2003].

Une ***observation*** est une information obtenue sur un équipement [Afnor, 2001]. Grâce à l'observation, on peut déduire l'état des composants. Les observations interviennent lors de l'apparition des symptômes.

Un ***symptôme*** est un phénomène qui survient sur un dispositif et qui révèle un dysfonctionnement [Afnor, 2001]. Il est également fréquent de regrouper les symptômes en fonction du dysfonctionnement auxquels ils sont liés : on parle alors de syndrome.

L'apparition de symptômes cause une défaillance. Une ***défaillance*** est définie comme la cessation de l'aptitude d'un bien à accomplir une fonction requise [Afnor, 2001]. C'est un évènement correspondant à un dysfonctionnement du service lié à un fonctionnement anormal ou plus exactement non conforme aux spécifications à un instant donné.

La distinction des différentes situations de fonctionnement d'un équipement (fonctionnement normal et anormal des composants d'un équipement) reflète ce que nous

appelons les modes de fonctionnement. On distingue les modes de fonctionnement normal, dégradé et défaillant. La Figure 1.1 montre le passage d'un état de bon fonctionnement à un état de panne en passant par la dégradation et la défaillance.

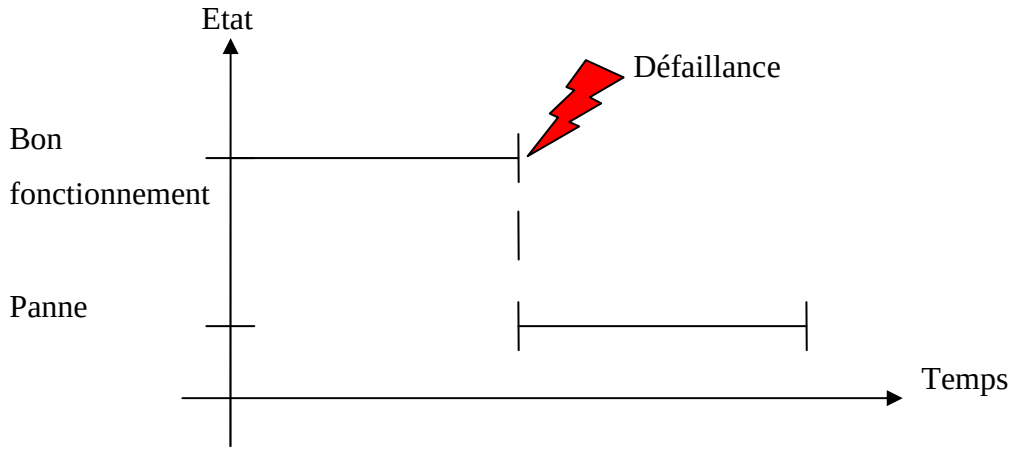


Figure 1.1. Transition d'un état de bon fonctionnement à un état de panne causée par une défaillance

Une ***dégradation*** est une évolution irréversible d'une ou plusieurs caractéristiques d'un bien liée au temps, à la durée d'utilisation ou à une autre cause externe [Afnor, 2001].

De ce fait, la défaillance diffère de la dégradation, qui est une évolution de l'état. Le fonctionnement de l'équipement est amené à évoluer dans le temps. Cette évolution peut se traduire par l'éloignement du mode de fonctionnement, qui est initialement en mode de fonctionnement normal (fonctionnement en mode nominal), à un mode de fonctionnement dégradé. En effet, une dégradation peut aller jusqu'à une panne

Une ***panne*** est définie par un état d'un bien inapte à accomplir une fonction requise, excluant l'inaptitude due à la maintenance préventive ou à d'autres actions programmées ou à un manque de ressources extérieures [Afnor, 2001]. Elle peut également désigner une anomalie, condition anormale diminuant ou supprimant l'aptitude d'une entité fonctionnelle à réaliser une activité requise.

Maintenant, nous allons aborder différentes définitions du diagnostic dans la littérature scientifique. Ces définitions dépendent de la communauté d'étude.

En *reconnaissance des formes*, le diagnostic est défini comme l'ensemble des états de fonctionnement d'un équipement qui est homologue à un ensemble de classes et le vecteur forme est le vecteur composant des paramètres observés sur le système [Dubuisson, 2001].

En *supervision*, qui est considérée comme un processus plus global où est intégré le diagnostic de la défaillance [Piechowiak, 2003], le décompose en deux fonctions à savoir : la localisation qui permet de déterminer le sous-ensemble fonctionnel défaillant et l'identification qui doit déterminer les causes qui ont mené à une situation anormale (cf. Figure 1.2).

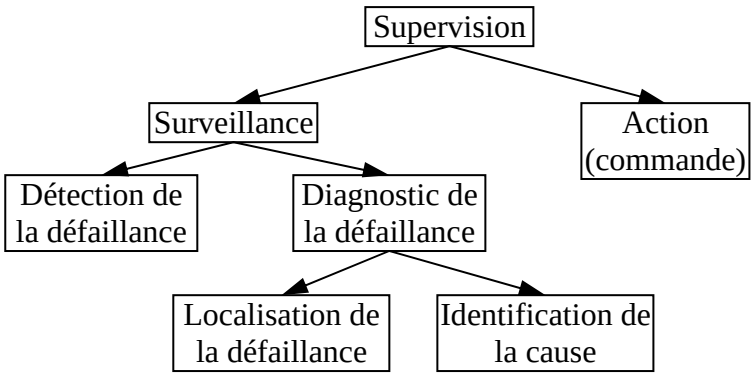


Figure 1.2. Diagnostic de pannes

Dubuisson et al. [2001] considèrent deux types de données de diagnostic : les données numériques (exploitation des observations issues des capteurs par exemple) et les données symboliques (connaissances sur le système étudié). Les mêmes auteurs définissent par la suite deux types de connaissances :

- *la connaissance globale* qui peut être qualifiée de connaissance *a priori* sur le système, cette connaissance s'appuie sur des expériences passées du système,
- *la connaissance instantanée* qui concerne l'ensemble des biens impliqués à un instant donné afin de prendre une décision et l'exploiter ; cette connaissance est issue des observations qui peuvent être numériques ou symboliques.

Peng et Reggia [1990] s'inspirent de l'étymologique du diagnostic et donnent la définition suivante : « Etant donné un ensemble de manifestations observées (symptômes, constatations, etc), il s'agit d'expliquer leur présence, de remonter aux causes, en utilisant un savoir sur le système considéré ». Cette définition met en évidence le raisonnement qui doit être suivi pour réaliser la fonction « comprendre » et elle a été reprise par plusieurs auteurs notamment [Zwingelstein, 1995], [Grosclaude, 2001] et [Bouchon-Meunier & Christophe, 2003].

Une définition normalisée du diagnostic que nous prenons comme référence a été proposée par l'AFNOR [Afnor, 2001]: « *ce sont les actions menées pour la détection de la panne, sa localisation et l'identification de la cause* ». Selon la même source, la localisation de la panne représente les actions menées en vue d'identifier où se situe le fait générateur de la panne (la cause).

Cependant, le diagnostic de défaillance regroupe deux types de méthodes, à savoir : les méthodes quantitatives et les méthodes qualitatives [Toscano, 2005].

- Les méthodes quantitatives : ces méthodes se basent sur des modèles mathématiques (analytique : équations différentielles, espaces d'état, fonctions de transfert, modèles de fatigue…) des signaux et du système. Les modèles mathématiques des signaux concernent le traitement du signal tel que le filtrage, l'analyse spectrale, la méthode des résidus, le seuillage, les tests statistiques… . Quant aux modèles mathématiques du système, ils concernent les redondances analytiques telles que l'espace de parité, les bonds graphs, le filtre de Kalman….
- Les méthodes qualitatives : ces méthodes ne se servent pas de modèles mathématiques mais prennent appui sur des bases de données symboliques et/ou numériques. Ces méthodes concernent les outils d'intelligence artificielle tels que la reconnaissance de formes, les systèmes experts, les réseaux de neurones, le Raisonnement à Partir de Cas (RàPC) …

Nous orientons notre travail vers les méthodes qualitatives car nous ne disposons pas de modèles mathématiques du comportement de l'équipement.

Par ailleurs, Toscano et Lyonnet [2003] ont classé les méthodes de diagnostic en trois groupes, à savoir :

- le premier groupe comprend les méthodes qui reposent sur la connaissance *d'un modèle mathématique du processus de diagnostic*. Ces méthodes s'apparentent à un problème d'estimation paramétrique ou d'état ;
- le deuxième groupe comprend les méthodes se basant sur *la connaissance des différents modes de fonctionnement du processus*. Ces méthodes s'apparentent à un problème de classification permettant de caractériser les éléments défaillants ;

- le troisième groupe comporte les méthodes qui se basent sur *les techniques d'intelligence artificielle* impliquant l'emploi d'un système expert.

Nous nous intéressons dans notre étude au troisième groupe de méthodes. Dans ce cadre, Palluat [2004] propose une étude sur les méthodes de diagnostic basées sur des outils d'intelligence artificielle accomplissant la fonction de recherche de causes. Nous pouvons recenser :

- *les méthodes à base de modèles comportementaux :* elles permettent de construire des modèles qui sont liés directement au fonctionnement du système (les automates d'états finis, les réseaux de Petri « dans une optique d'utilisation en diagnostic de systèmes ») ;
- *les méthodes de reconnaissance de formes :* elles supposent qu'aucun modèle n'est disponible pour décrire les relations de cause à effet. La seule connaissance repose sur l'expertise humaine confortée par un solide retour d'expérience [Zwingelstein, 1995] (les outils statistiques, les réseaux de neurones, la logique floue, les réseaux neuro-flous, les systèmes experts et le RàPC) ;
- *les méthodes à base de modèles explicatifs :* elles reposent sur une analyse profonde du système de manière à avoir les connaissances suffisantes à l'expression de ces relations de cause à effet (les graphes contextuels, la logique floue…).

Dans le même registre, Piechowiak [2003] présente un aperçu « non exhaustif » sur les méthodes d'intelligence artificielle mais utilisées plus précisément dans le diagnostic industriel. Les méthodes abordées sont les réseaux de neurones, les réseaux Bayésiens, les arbres de décision et le RàPC.

En ce qui concerne notre étude, nous adoptons l'approche du RàPC qui est largement utilisée dans le domaine du diagnostic. En effet, le RàPC est bien adapté au développement d'un système d'aide au diagnostic industriel, son utilisation en diagnostic remonte à la fin des années 80 avec l'apparition des systèmes d'aide au diagnostic à base du RàPC. A titre d'exemple, le système PATDEX [Richter & Wess, 1991] dédié au diagnostic des machines complexes est intégré dans le système MOLTKE. Ces systèmes vont être abordés en détail au chapitre 2.

De plus, le choix du RàPC pour notre travail est motivé par les points suivants :

- le RàPC est un moyen de gestion des connaissances du domaine étudié et d'apprentissage ;
- initialement, les connaissances du domaine sont *a priori* insuffisantes pour construire une base de connaissance conséquente et des modèles causaux. Ainsi, le RàPC peut fonctionner avec seulement un ensemble de cas de domaine [Main et al., 2000] ;
- le système que nous voulons développer ne dispose pas d'assez d'expériences et comme le RàPC peut être mis en place avec un ensemble limité d'expériences, alors l'acquisition incrémentale d'une nouvelle connaissance est tout à fait possible durant l'utilisation du système de RàPC. Cette acquisition permet l'ajout de tout nouveau cas jugé pertinent dans la base contenant la connaissance, de contribuer à une large couverture de problèmes et d'améliorer les résultats fournis par le système [Main et al., 2000] ;
- lorsque le système propose une solution (origines de la panne et actions de réparation) suite à une description d'une situation de diagnostic, elle peut ne pas être exactement similaire au problème posé. Malgré cela, le RàPC a la possibilité de raisonner avec des données imprécises ou incomplètes ;
- le système que nous voulons concevoir doit mettre à disposition les expériences apprises et accélérer le processus de résolution de problème. De plus, le diagnostic fait partie des domaines dans lesquels les problèmes se répètent et les solutions déjà trouvées et répertoriées peuvent être réutilisées. De ce fait, le RàPC permet d'éviter la répétition des étapes menant à la solution en réutilisant une solution précédente à partir de sa mémoire.

Nous nous intéressons par la suite aux principes fondamentaux de cette approche du RàPC.

3. Présentation de l'approche du Raisonnement à Partir de Cas (RàPC)

Le RàPC est une approche d'apprentissage et de résolution de problèmes basée sur les expériences passées [Aamodt & Plaza, 1994]. Nous abordons les points suivants :

- l'historique du RàPC qui a commencé à partir de la fin des années 70 ;
- les communautés en RàPC ;
- le raisonnement analogique du RàPC ;
- la définition du cas dans lequel se trouve l'expérience ;
- les types de modèles de RàPC issus de la représentation du cas ;
- la base de cas dans laquelle sont stockés les cas ;
- et la manipulation du cas à travers le cycle du RàPC.

3.1. Historique

Inspiré par les travaux de Minsky et Schank réalisés à la fin des années 70, Schank [1982] formule pour la première fois le paradigme de raisonnement basé sur les cas. En effet, la théorie développée par Minsky [1975] présente un réseau de nœuds et de relations entre ces nœuds ainsi que la notion de « frame (script, schéma) » qui correspond à une structure remémorée qui doit être adaptée pour correspondre à la réalité d'une nouvelle situation rencontrée. Cependant Shank doute de la flexibilité du raisonnement logique et d'une représentation des connaissances ordinaires sous une forme synthétique de propositions indépendamment vraies. Par conséquent, il reprend ces travaux et suppose que le processus de compréhension correspond à un processus d'explication qui s'applique d'une manière itérative [Schank, 1982]. D'ailleurs, Schank est considéré comme l'initiateur du terme « Case-Based Reasoning ». Il introduit à travers le modèle de « mémoire dynamique » un degré de généralité varié connu sous le nom de « MOPS (Memory Organization Packets) » constituant un réseau dense d'expériences. De plus, l'auteur tente d'opérationnaliser le comportement humain et l'optimiser si possible. Dans ce cadre, Gebhardt et al. [1997] définissent le raisonnement à partir d'expériences comme une façon naturelle de penser caractérisant la réflexion humaine sans doute plus encore que le raisonnement avec des règles.

A la fin des années 80, les recherches dans le domaine du RàPC ont réellement commencé à prendre forme et notamment avec les conférences « DARPA » organisées aux Etats-Unis en 1988 [Kolodner, 1988], avant de s'imposer en Europe avec la première conférence Européenne en 1993 à Kaiserslautern [Richter et al., 1993], puis avec la première conférence internationale à Lisbonne en 1995 [Veloso et al., 1995].

3.2. Communautés en RàPC

Selon le type de méthode développée, le RàPC se situe soit dans le domaine de l'IA (machine learning) soit dans le domaine de l'IC (cf. Figure 1.3).

L'IC se place à l'intersection de deux communautés de recherche : l'IA et les sciences cognitives.

L'intelligence artificielle (IA) est un domaine de recherche permettant d'élaborer des systèmes intelligents. L'IA est la « recherche de moyens susceptibles de doter les systèmes informatiques de capacités intellectuelles comparables à celles des êtres humains » [CNTRL, 1979].

Les sciences cognitives regroupent un ensemble de disciplines scientifiques dédiées à l'étude et la compréhension des mécanismes de la pensée humaine, animale ou artificielle, et plus généralement de tout système cognitif, c'est-à-dire tout système complexe de traitement de l'information capable d'acquérir, conserver, utiliser et transmettre des connaissances [Wikipédia, 2009].

L'ingénierie des connaissances est le domaine qui correspond à l'étude des concepts, méthodes et techniques permettant de modéliser et/ou d'acquérir les connaissances pour des systèmes réalisant ou aidant les humains à réaliser des tâches se formalisant a priori peu ou pas [Charlet et al., 2000].

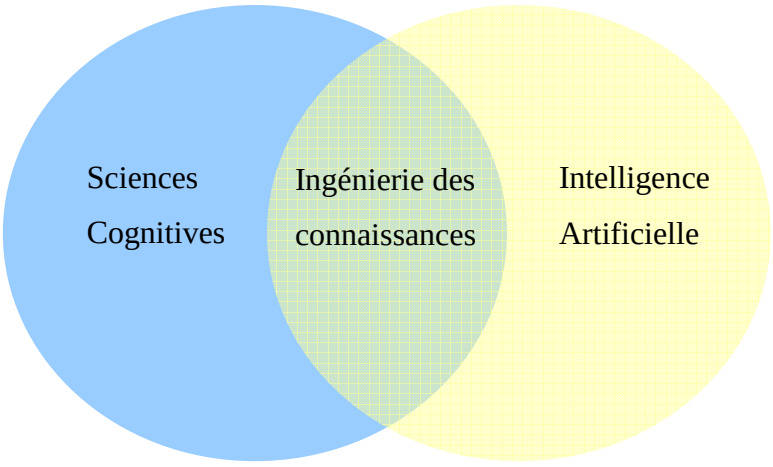


Figure 1.3. L'ingénierie des connaissances à l'intersection des deux communautés des Sciences Cognitives et de l'Intelligence Artificielle

Le RàPC est une approche utilisant un raisonnement par analogie. Mille et al. [1996] l'ont représenté en proposant un modèle de carré d'analogie qui permet de faire le lien entre la description du cas et sa solution.

3.3. Carré d'analogie

Mille et al. [1996] introduisent la présentation du RàPC en utilisant le carré d'analogie (cf. Figure 1.4). Ce carré d'analogie exprime :

- D'une part, le lien entre la description d'un cas et sa solution (la trace du raisonnement menant à la solution) ;
- D'autre part, les liens entre la description et la solution du cas source de la base de cas et du cas cible représentant un nouveau problème à résoudre (similarité entre deux problèmes). Dans ce cas là, la solution du cas cible est adaptée en fonction de la similarité et les descripteurs des cas sources similaires de la base de cas sont adaptés au cas cible.

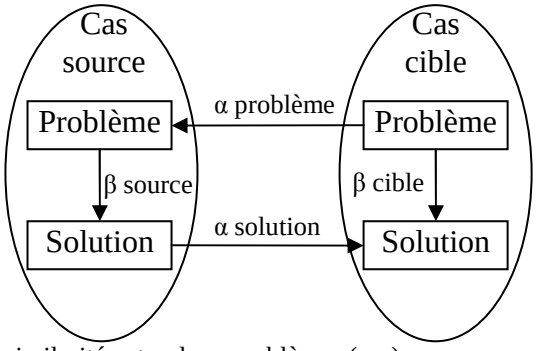


Figure 1.4. Carré d'analogie [Mille et al., 1996]

La mesure de similarité α détermine la similarité du cas source sélectionné à partir des valeurs de descripteurs du problème cible.

Les relations de dépendance β entre les valeurs de descripteurs de problème et les valeurs de descripteurs de solution mettent en évidence les descripteurs de solution qui doivent être adaptés. Les descripteurs de solution dépendent des descripteurs de problèmes source qui sont différents des descripteurs de problèmes cible.

En d'autres termes, si une valeur de descripteur source dépend d'une valeur de descripteur de problème, une modification de la valeur du descripteur de problème entraînera une modification « analogue » à la dépendance du descripteur de solution correspondant. Cette connaissance est nécessaire pour l'adaptation. En fonction de ces dépendances et des écarts α constatés à corriger, l'adaptation permet de proposer une solution cible candidate qui

pourra être vérifiée par rapport à sa conformité aux dépendances particulières qui pourraient exister entre problème et solution cible.

3.4. Définition du cas

Un cas est une expérience représentée par une connaissance. Cette expérience constitue une leçon permettant au système de RàPC de résoudre des problèmes de différentes natures. Selon le domaine d'application et les objectifs à atteindre, les informations contenues dans le cas varient. Fuchs [2006] définit le cas comme étant la description informatique d'un épisode de résolution de problème.

Selon Mille [1995], la définition d'un cas (dans la base de cas) passe par trois étapes : la première étape concerne « la synthèse » qui consiste à trouver une structure permettant de satisfaire des spécifications. La deuxième étape concerne « l'analyse » qui, à partir d'une structure particulière, consiste à trouver le comportement associé. La troisième étape concerne « l'évaluation » qui consiste à vérifier que le comportement est conforme à ce qui est attendu.

Nous allons détailler la structure d'un cas et son indexation dans la base de cas selon plusieurs points de vue existants dans la littérature.

3.4.1. Structure du cas

Tout d'abord, un cas en RàPC est généralement composé de deux espaces disjoints : l'espace des problèmes et l'espace des solutions. L'espace problème concerne la partie dans laquelle on trouve les objectifs à atteindre. Quant à l'espace solution, il regroupe la description de la solution apportée par le raisonnement, sa justification, son évaluation ainsi que les étapes qui ont mené à cette solution.

On peut distinguer deux types de cas : *cas source* et *cas cible*. Le cas source est celui dans lequel les parties « problème » et « solution » sont renseignées. Donc, c'est un cas dont on va s'inspirer pour résoudre un nouveau problème. Le cas source peut aussi contenir une autre partie appelée « information de qualité » [Reinartz et al., 2000]. Cette partie contient des informations sur l'utilisation du cas dans le système. Quant au cas cible, c'est celui qui porte le problème et dont sa partie solution n'est pas renseignée.

Suivant la nature du problème à traiter, il existe plusieurs représentations de cas. Les approches traditionnelles les classent en trois catégories :

- *La représentation textuelle* ;

- *La représentation semi structurée (vecteur de composants)* ;
- *La représentation structurée*.

Cependant, la représentation structurée est la plus utilisée dans la majorité des travaux. Ainsi, le cas est souvent représenté sous la forme d'un ensemble de descripteurs.

Un descripteur « *d* » est défini par une paire *d = (a, v)* où « *a* » est un attribut et « *v* » est la valeur qui lui est associée. Nous empruntons le formalisme de Lieber [2007] pour définir les cas.

Un cas source est représenté par un couple (*srce*, *Sol(srce)*) et le cas cible par le couple (*cible*, *Sol(cible)*), où *Sol(cible)* est inconnue et pour laquelle on voudrait lui apporter un résultat. Comme les cas sont représentés par un ensemble de descripteurs alors :

- ds_i (pour i = 1, …, n) : représente les descripteurs de la partie problème du cas source « srce » ;
- dc_i (pour i = 1, …, n) : représente les descripteurs de la partie problème du cas cible « cible » ;
- Ds_i (pour i = 1, …, m) : représente les descripteurs de la partie solution du cas source « *Sol(srce)* » ;
- Dc_i (pour i = 1, …, m) : représente les descripteurs de la partie solution du cas cible « *Sol(cible)* ».

C'est cette formalisation que nous retenons et que nous exploiterons dans la partie application du système de RàPC développé dans notre cadre d'étude que nous allons aborder au chapitre 4.

3.4.2. Indexation du cas

Les cas sont organisés dans une mémoire appelée *base de cas*. Afin de faciliter cette organisation et ainsi la recherche du cas le plus approprié au problème posé, il faut désormais les indexer. Il est à noter que lors de la recherche des cas, c'est la partie problème qui va être sollicitée. Or, cette partie problème est décrite par un ensemble de caractéristiques pertinentes nommées « *indices* ». Ces indices vont déterminer dans quels contextes et dans quelles situations les cas vont être recherchés et retrouvés pour les proposer au problème rencontré. Alors il faut trouver le moyen de bien manipuler ces indices pour une configuration optimale. Pour cela, il y a plusieurs méthodes d'indexation : *manuelles* ou *automatiques*. Dans le cas des méthodes manuelles, il est supposé que l'objectif d'utilisation des cas, et surtout des

circonstances dans lesquelles les cas seront utiles, soit déterminé précisément. Toutefois, les méthodes d'indexation sont de plus en plus automatisées.

Par ailleurs, le choix des indices dépend du domaine d'application. Comme le précise Kolodner [1996], ces indices doivent vérifier certaines propriétés suivantes :

- *Prédictifs* afin de jouer un rôle déterminant dans le choix d'une solution pour un nouveau problème ;
- *Suffisamment abstraits* pour que le cas ait la possibilité d'être utilisé plusieurs fois pour la résolution de plusieurs problèmes ;
- *Suffisamment concrets* pour que le cas soit reconnu le plus rapidement possible pour la résolution d'un nouveau problème.

Nous concluons que le cas peut donc avoir plusieurs représentations. Par ailleurs, le cas dans le domaine du diagnostic dispose d'une formalisation biens spécifique. Ceci va être le sujet de la sous-section suivante.

- ***Exemple d'un cas adapté au diagnostic***

L'utilisation du RàPC en diagnostic apparaît relativement aisée, avec comme propriété, une structure de cas adaptée. En effet, en diagnostic, un cas décrit une situation diagnostiquée caractérisée par les symptômes observés et les valeurs mesurées. L'objectif est de retrouver la ou les causes et de proposer une action pour une éventuelle intervention de maintenance. Quant à la représentation du cas en diagnostic, elle est déterminée par une liste de descripteurs pouvant être des objets complexes. Ainsi, la structure du cas pourrait être comme suit [Chebel-Morello et al., 2007] :

Problème ⟷ Symptômes (description d'une situation de diagnostic)

Solution ⟷ Origines (plusieurs possibles) + Actions (stratégie de maintenance)

Un exemple de la structure d'un cas dédié au système SISTRE (qui sera détaillé au chapitre 5) est montré sur la Figure 1.5.

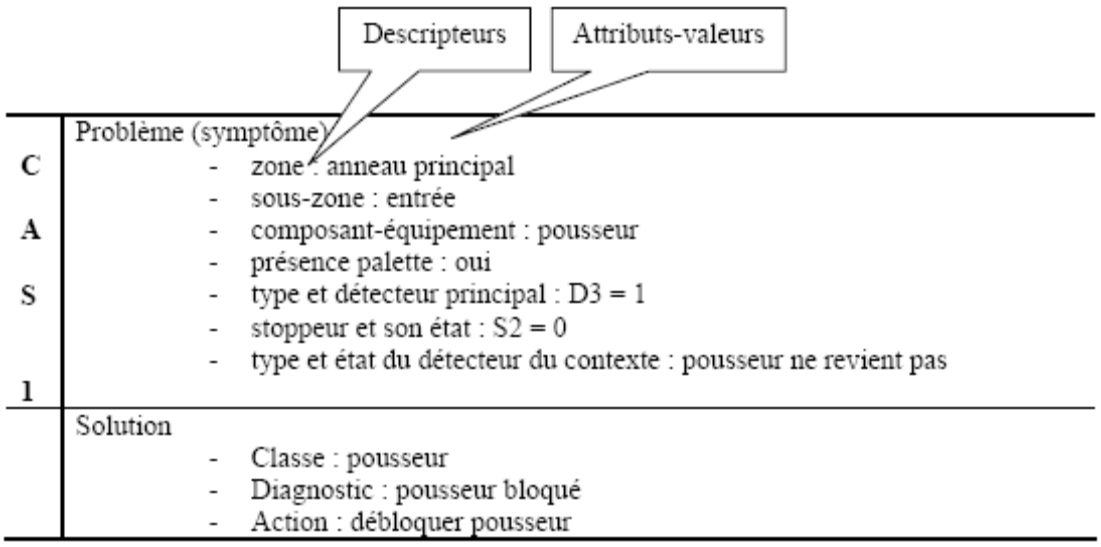


Figure 1.5. Exemple d'une structure d'un cas adapté au diagnostic

Ce cas est composé de sept descripteurs problèmes reflétant la description d'une panne localisée au niveau du pousseur qui se trouve à l'entrée de l'anneau principal (ce que reflètent les trois premiers descripteurs). Cette zone géographique implique des composants notamment un détecteur de présence « D3 », un actionneur électrique « S2 » et un actionneur pneumatique « pousseur » avec leurs états (les descripteurs ds_5, ds_6 et ds_7 respectivement). Quant à la partie solution, elle est composée de trois descripteurs montrant les origines de la panne ainsi que l'action de réparation associée. L'origine concerne donc le pousseur qui est bloqué et l'action de réparation associée est de le débloquer.

La représentation utilisée dans cet exemple est une représentation de type vecteur (liste) des couples attribut-valeur avec une structure plate (elle peut être également hiérarchique). Les attributs peuvent être de différents types : numérique, symbolique, etc., ce qui est le cas dans notre exemple.

Par ailleurs, le choix du type de cas à exploiter dans un système de RàPC se répercute directement sur la nature du modèle de RàPC à manipuler. La section suivant aborde les différents modèles de RàPC existants.

3.5. Modèles de RàPC

Les systèmes utilisant le RàPC comme approche de résolution de problème peuvent disposer de plusieurs modèles. Selon Bergmann et al. [2003] et Lamontagne [2004], ces

modèles peuvent être regroupés en trois grandes familles à savoir : les modèles conversationnels, textuels et structurels.

3.5.1. Le modèle conversationnel

Le modèle conversationnel est essentiellement utilisé dans des systèmes de RàPC dédiés aux applications commerciales tels que les « call centers », consacrés aux services après vente (comme chez Darty qui a une application développée par Kaïdara (http://www.kaidara.com/)). Dans ce type de modèle, les problèmes ne sont pas décrits avant le début de la recherche des cas dans la base de cas car il ne peut disposer de toutes les connaissances *a priori*.

Le cas est renseigné au fur et à mesure de l'importance du problème. En effet, dans quelques domaines, tels que le service clientèle après vente, ce service dispose de questionnaires qui suivent un cheminement suivant les réponses. La caractérisation d'une situation est difficile à déterminer à l'avance surtout pour les utilisateurs néophytes des systèmes de RàPC. De ce fait, le modèle conversationnel a été proposé par *Inference Corporation* (fournisseur leader de logiciels et de services pour la gestion de la relation client et de e-commerce) pour répondre aux exigences.

Le modèle conversationnel met en interaction l'utilisateur et le système de RàPC (d'où la notion de "conversation") afin de déterminer les solutions les plus appropriées au problème à résoudre.

Aha et al., [2001] décrivent un cas dans le modèle conversationnel par le triplet (*Problème P, Questions* Q_A*, Action A*) comme cela est montré sur la Figure 1.6 :

- *Problème P :* cette partie contient une brève description textuelle concernant le problème abordé, de quelques lignes généralement ;
- *Questions-réponses* Q_A *:* une série de question accompagnée par des réponses est présentée sous forme d'index. Le nombre de questions et de réponses varie d'un cas à un autre. Chaque question peut être caractérisée par un poids reflétant son importance dans le cas ;
- *Action A :* tout comme la partie problème, elle décrit textuellement la solution proposée au problème posé.

Cas : 241
Titre : *cartouche d'encre endommagée causant des traces noires*
Description : *l'imprimante laisse de petits points noirs sur les deux côtés de la page. Parfois des larges tâches couvrent également la région à imprimer.*
Questions :
Est-ce que les copies sont de mauvaise qualité ? Réponse : oui Score : (-)
Quels types de problèmes avez-vous ? Réponse : trace noires Score : (default)
Est-ce qu'un nettoyage de l'imprimante règle le problème ? Rép : non …
Actions : *vérifier la cartouche d'encre et la remplacer si le niveau d'encre est faible*

Figure 1.6. Exemple de représentation d'un cas dans un modèle conversationnel de RàPC

La description de la partie problème et de la partie action n'est pas structurée (*free-text*). Cette représentation du cas est une extension de la représentation du cas dans le modèle structurel.

Traditionnellement, le schéma de résolution du modèle conversationnel qui met en interaction l'utilisateur et le système commence par une courte description textuelle du problème à résoudre de la part de l'utilisateur. Ensuite, le système procède à la recherche de la partie problème la plus similaire à la description fournie. Ce qui en résulte une série de questions proposées à l'utilisateur. Ce dernier doit par la suite choisir les questions auxquelles il souhaite répondre. Ce choix se répercutera sur la similarité qui doit être réévaluée à chaque fois pour chacun des cas. Enfin, lorsque la similarité calculée pour un cas atteint un seuil prédéterminé alors sa solution est proposée. Cependant, si aucun cas n'a atteint ce seuil de similarité prédéfini alors le problème sera stocké comme non résolu.

On compte parmi les travaux récents qui ont été faits dans ce domaine, les travaux de l'équipe de recherche de Jean Lieber dans [Armaghan et al., 2008]. Ces travaux présentent les idées principales d'un futur système de RàPC conversationnel destiné à l'assistance des opérateurs d'un service après vente (SAV) d'une entreprise vosgienne. La remémoration est semi-automatique et a pour objectif de préciser à chaque interaction avec le client le sous-ensemble des fiches SAV qui sont susceptibles de correspondre à la définition courante du problème de panne.

3.5.2. Le modèle textuel

Les premiers travaux sur le modèle textuel datent du milieu des années 90 et à ce jour, aucune représentation standard n'est apparue pour ce type de modèle. La description exacte du cas est primordiale dans ces modèles. Par conséquent, la représentation textuelle des cas

joue un rôle très important dans la résolution de problème. Par exemple, obtenir le texte d'un jugement légal servant de jurisprudence à une nouvelle cause [Lamontagne, 2004]. La différence de ces modèles par rapport aux modèles structurels réside dans la représentation du texte.

Les cas dans les modèles textuels sont semi-structurés ou non-structurés :

- *Les cas non-structurés* sont les cas qui disposent que d'un seul attribut dont la description est complètement en texte libre (free-text) ;
- *Les cas semi-structurés* sont les cas qui disposent de plusieurs attributs étiquetés contenant du texte.

Les modèles textuels s'appuient sur la résolution de problèmes à partir d'expériences se trouvant dans des documents textuels. Deux axes sont identifiés dans les travaux portant sur ces modèles à savoir : *structuration de cas textuels* et *extension du modèle de recherche d'information*.

Le premier axe traite les cas textuels. Il a pour objectif de les structurer du mieux possible afin que l'exploitation des méthodes développées soit profitable pour les systèmes de RàPC textuels. Dans cet axe de recherche, les textes sont représentés par un nombre limité de traits qui sont basés sur les caractéristiques du domaine (concepts, catégories, sujets, mots-clés, etc.). Un trait est un attribut ou une caractéristique déterminant la description de problèmes et de solutions du domaine [Lamontagne, 2004].

Quant au deuxième axe, une particularité est accordée à la phase de recherche des cas. Les mesures de similarité appliquées lors de cette phase sont définies soit sémantiquement soit par des extensions au modèle vectoriel de recherche d'information [Salton, 1989]. De ce fait, des mécanismes de recherche des cas plus sophistiqués sont élaborés mais tout en gardant un processus d'indexation le plus simple possible. Cet axe de recherche donne la possibilité d'avoir une indépendance pour les applications génériques par rapport au domaine d'application. L'exemple d'une application concrète est représenté dans le projet FAQFinder [Burke et al. 1997]. FAQFinder (Frequently-Asked Questions) est un système de questions-réponses basé sur les foires aux questions de USENET. Un FAQ est un cas qui contient la description d'un problème sous forme de questions et la description d'une solution sous forme de réponses. Le système contient plus de 600 fichiers FAQ, ce qui nous amène à quelques dizaines de milliers de FAQs individuels.

Les modèles textuels ont donc mené vers les travaux en RàPC textuel (RàPCT). Un état de l'art sur le RàPC textuel a été abordé dans [Lamontagne & Lapalme, 2002]. On peut citer

également les travaux de Bentebibel [2008] qui a mis en place un système de RàPCT pour la sécurité routière « SAARA » et qui exploite non seulement le modèle textuel mais également le modèle structurel et conversationnel.

La différence entre les trois modèles présentés réside dans la structuration du cas et de son traitement durant les phases du cycle du RàPC. En effet, dans le modèle conversationnel, il n'y a aucun traitement textuel lors de l'interaction entre l'utilisateur et le système. Il se limite à une comparaison de mots clés ou des séquences de « n » caractères découpées d'un texte (*n-grammes*). Le texte est uniquement utilisé pour rendre les questions plus compréhensibles à l'utilisateur. Quant au modèle structurel, les valeurs des attributs des cas sont des chaînes de caractères sans syntaxe ni sémantique. De plus, contrairement aux modèles textuels, les attributs du cas sont complètement structurés.

3.5.3. Le modèle structurel

Le modèle structurel est le premier modèle qui a été utilisé lors de l'apparition des premiers systèmes de RàPC. En effet, par sa simplicité, le concepteur du système de RàPC doit disposer à l'avance de toutes les caractéristiques importantes décrivant un cas pour pouvoir réaliser son modèle. Un modèle de données du domaine est donc élaboré grâce à une expertise du domaine d'application permettant de bien caractériser une situation donnée. Ainsi, les cas sont complètement structurés dans ce modèle de données et représentés par des paires <attribut, valeur>. Un attribut représente une spécification importante du domaine étudié. Quant à la valeur, elle vient structurer les attributs et elle est souvent exprimée par une échelle de valeur d'entiers, réels, booléens ou symboliques. Un exemple d'un cas représenté par la paire <attribut, valeur> est montré sur la Figure 1.7.

Cas : *1979*
Atelier : *HMK*
Nombre machines : *07*
Horizon de travail : *25*
Encours : *153*
Débit production : *26.11*
Date : *20/10/2007*

Figure 1.7. Exemple de représentation d'un cas dans un modèle structurel de RàPC

Afouba et al. [2004] déterminent plusieurs types de cas faisant parti du modèle structurel, à savoir : des frames, des objets, des arbres ou des graphes. Ils peuvent également être représentés sur un seul ou plusieurs niveaux hiérarchiques d'attributs.

Le modèle structurel qui permet de formaliser les expériences sous un format complètement structuré a permis par exemple le développement de plusieurs applications commerciales de RàPC comme KAIDARA (http://www.kaidara.com/) ou l'environnement Jcolibri (http://gaia.fdi.ucm.es/projects/jcolibri/jcolibri1/tutorials.html).

Nous exploitons ce type de modèle pour développer notre système de diagnostic par RàPC. Nous justifions ce choix par le fait que le domaine d'étude dans le modèle structurel est connu a priori et que les cas doivent être structurés pour caractériser les composants de l'équipement étudié.

3.6. Base de cas (la mémoire dans les systèmes de RàPC)

Le bon fonctionnement et les performances d'un système de RàPC sont fortement liés à l'organisation de sa mémoire. En effet, la mémoire qui contient tous les cas sources précédemment retenus est appelée *base de cas*. La base de cas est un élément majeur dans l'indexation et l'organisation des cas afin de pouvoir les retrouver facilement et efficacement. Nous pouvons distinguer deux types d'organisation de la base de cas :

- ***Base de cas plate*** dans laquelle les cas sont organisés de manière linéaire (vecteur, tableau, graphe, etc). Autrement dit, les cas sont stockés dans une liste séquentielle. C'est sans doute l'organisation la plus simple. Par conséquent, nous allons exploiter cette organisation dans nos travaux. De plus, cette organisation est prise en compte dans la majorité des travaux de RàPC ;
- ***Base de cas hiérarchique*** dans laquelle la structuration et l'organisation des cas est faite selon des niveaux hiérarchiques donnés.

Nous pouvons également trouver des bases de cas qui sont construites suivant les deux types d'organisations combinées. A titre d'exemple, nous pouvons trouver ce type de combinaison dans les travaux de Malek [2000] qui est exploité par le système ProBis. En effet, c'est un système hybride de RàPC et d'un réseau de neurones incrémental. La mémoire plate est considérée comme le niveau bas de la mémoire et elle est divisée en plusieurs groupes. Chaque groupe est représenté par un prototype dans le réseau. Quant à la mémoire

hiérarchique, elle sert comme un système d'indexation pour les zones de la mémoire plate formant le haut niveau de la mémoire. Elle est constituée du réseau ARN2 permettant de construire des prototypes représentatifs pour chaque classe.

Cependant, lors de la création d'une base de cas, trois points principaux doivent être considérés [Main et al., 2000] :

- La structure et la représentation des cas ;
- Le modèle de la mémoire utilisée pour organiser la base de cas ;
- La sélection des indices qui sont utilisés pour identifier chaque cas.

En effet, lorsque nous évoquons la base de cas, qui représente le cœur du système de RàPC, nous lui associons forcément les cas (et dans notre étude il s'agira des cas dédiés au diagnostic). Alors, lorsqu'une panne (ou défaillance) se produit, il faut localiser le problème, décrire le contexte de diagnostic et donner une stratégie de maintenance. Pour réaliser ces différentes étapes, nous devons passer par différentes phases du *cycle du RàPC*.

3.7. Cycle du RàPC

Le RàPC dispose d'un cycle dont le nombre de phases varie selon les différentes sources bibliographiques. Il peut être composé de trois, quatre ou cinq phases. Fuchs et al. [2006] déterminent trois phases à savoir la remémoration, l'adaptation et la mémorisation. Les premiers auteurs à avoir décrit le cycle du RàPC sont Aamodt et Plaza [1994] et le composent de quatre phases : la remémoration (ou recherche du cas similaire), l'adaptation (ou la réutilisation du cas retrouvé), la validation (ou la révision du cas sélectionné) et la mémorisation (ou l'apprentissage). Quant à Mille [2006], il ajoute une phase préliminaire d'élaboration au début du cycle. La Figure 1.8 montre le cycle de RàPC avec ces cinq phases.

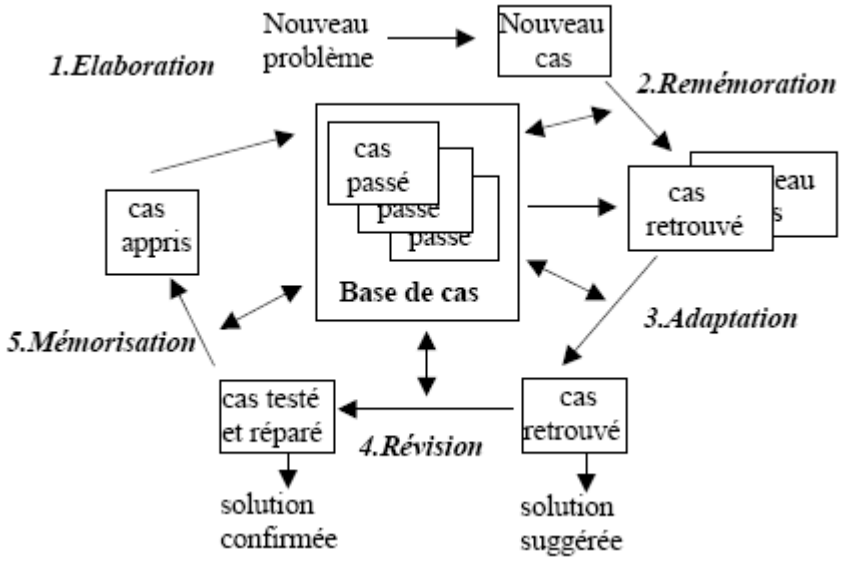


Figure 1.8. Le cycle de raisonnement à partir de cas selon [Mille, 1999]

Le cycle de RàPC que nous exploitons dans notre étude est donc composé de cinq phases :

- ***La phase d'élaboration*** dans laquelle le cas cible est construit en complétant ou filtrant la description d'un problème à partir d'une description éventuellement incomplète ;
- ***La phase de remémoration*** des cas sources à partir de la base de cas en recherchant des correspondances entre descripteurs des cas sources et du cas à résoudre (cible) ;
- ***La phase d'adaptation*** consiste à construire une solution au problème du cas cible inspirée de la solution du (des) cas source(s) le(s) plus similaire(s) ;
- ***La phase de révision*** de la solution proposée en cas d'une éventuelle solution insatisfaisante, alors il serait possible de la corriger. Dans ce cas, la solution est évaluée dans le monde réel en s'appuyant soit sur l'utilisateur, un expert humain, les connaissances du domaine ou sur un processus automatique ;
- ***La phase de mémorisation*** consiste à stocker un nouveau cas résolu dans la base de cas si ce stockage est jugé opportun afin d'enrichir la mémoire du système.

Chacune de ces phases vont être détaillées dans les sous-sections suivantes.

3.7.1. Phase d'élaboration du cas

Cette phase a pour rôle de mettre en place les spécifications du problème à résoudre (cible). De ce fait, différents mécanismes sont mis en place pour passer d'un problème souvent mal exprimé à un problème correctement défini. Il faut donc tenir compte de toutes les spécificités du cas abordées à la section 3.4. La méthode générale consiste à compléter ou filtrer la description d'un problème en se fondant sur la connaissance du domaine pour déduire tout ce qui est possible à partir d'une description éventuellement incomplète, et pondérer les descripteurs en fonction des dépendances identifiées entre les descripteurs du problème cible et les descripteurs de la solution recherchée. C'est à ce moment là que l'ontologie du domaine va intervenir afin de compléter les descripteurs importants.

Une première formalisation de la phase d'élaboration est récemment proposée dans [Fuchs et al., 2006] et donne la définition suivante : « une étape qui consiste, à partir de l'entrée du système de RàPC, à construire le problème cible ». Cette étude évoque l'importance de cette phase dans le RàPC car elle permet la facilité d'exécution et d'évaluation des différentes phases du cycle et l'identification des connaissances qui n'étaient pas prises en compte jusque là.

Dans notre étude, nous tenons compte de cette phase qui va être le point de départ du cycle de RàPC. En effet, notre élaboration consiste à exprimer une défaillance sous forme de cas cible. Une bonne élaboration du cas facilite la recherche d'un cas similaire au problème posé pour l'orienter vers une solution adaptable.

3.7.2. Phase de remémoration

Cette phase consiste à rechercher dans la base de cas le ou les cas sources les plus proches à partir de la description de la partie problème du cas cible, qui vont être utilisés pour le résoudre. Cette phase doit permettre d'obtenir la meilleure solution en effectuant les tâches suivantes : l'identification des caractéristiques pertinentes du problème, la remémoration et la sélection des meilleurs cas parmi les cas sources. L'exécution de ces tâches est dépendante de la représentation de cas, de leur indexation et de leur organisation de la base de cas.

Deux approches sont considérées dans cette phase, celles :

- reposant sur le calcul de la similarité entre le cas source et le cas cible ;
- utilisant, en plus de la notion de similarité, la notion de diversité [Smyth & McClave, 2001].

L'objectif la deuxième d'approche est de remémorer des cas similaires au cas cible et de choisir, parmi ces cas, ceux qui ne sont pas très similaires entre eux. Quant au premier type d'approche, l'objectif est de retrouver le cas de la base de cas similaire au problème actuel en mesurant leur degré d'appariement au sens où il est facilement adaptable à ce nouveau problème. Ainsi le degré de similarité représente la fonction *d'utilité/adaptabilité* de la solution. Il existe plusieurs mesures de similarité génériques dans la littérature scientifique et dans plusieurs domaines tels que l'Analyse des Données (AD), la Reconnaissance des Formes (RF), l'Apprentissage Symbolique (AS), ou encore les Sciences Cognitives (SC). De manière générale, une fonction de similarité est définie dans un univers *U* qui peut être modélisée à l'aide d'un quadruplet : (L_d, L_s, *T*, *FS*) [Lenz, 1999].

- soit L_d le langage de représentation utilisé pour décrire les données ;
- soit L_s le langage de représentation de la similarité ;
- soit *T* un ensemble de connaissances que l'on possède sur l'univers étudier ;
- soit *FS* la fonction binaire de similarité, telle que : $FS : L_d \times L_d \times L_s$.

Lorsque la fonction de similarité a pour objet de quantifier les ressemblances entre les données, le langage L_s correspond à l'ensemble des valeurs dans l'intervalle [0,1], et l'on parlera alors de *mesure de similarité*. Lenz [1999] considère schématiquement que ces mesures interviennent dans trois types de traitement de données à savoir : *la classification, l'identification* et *la caractérisation.*

- *le processus de classification* vise à structurer les données contenues dans *U*, en fonction de leurs ressemblances, sous la forme d'un ensemble de classes à la fois homogènes et contrastées ;
- *le processus d'identification* a pour but de déterminer la classe à laquelle un objet inconnu est susceptible d'appartenir, ou encore, de trouver à quel(s) objet(s) de *U* il est le plus ressemblant ;
- *le processus de caractérisation* permet de construire une représentation explicite des informations qui sont communes à un ensemble de données ; dans ce cas, le langage Ls est souvent un sur-ensemble de L_d ($L_s \in L_d$).

Par conséquent, le choix de la distance utilisée pour la mesure de proximité entre deux points de l'espace d'entrée est très important. Le choix d'une métrique dépend donc de

l'application visée et, plus précisément, par la taille de la base de cas disponible, par le codage utilisé, par le degré de recouvrement entre les différentes classes si elles existent dans la base, par la normalisation utilisée ainsi que par le bruit présent dans la base (cas parasites qui détériorent la qualité de prédiction). Une étude dans ce cadre est faite dans [Rifqi, 1996]. Ces mesures peuvent être *locales* ou *globales*.

- *similarités locales :* elles sont basées sur les caractéristiques du cas. Elles dépendent du type des caractéristiques et des rangs des valeurs des caractéristiques. Généralement, le calcul des similarités locales dépend du type de descripteur et est basé sur la distance.

  - pour les valeurs de descripteurs numériques :

$$sim(a,b) = 1 - \frac{|a-b|}{range}$$

  - pour les valeurs de descripteurs symboliques (mono-valeurs) :

$$sim(a,b) = \begin{cases} 1 \text{ pour a = b} \\ 0 \text{ pour a} \neq \text{b} \end{cases}$$

  - pour les valeurs de descripteurs symboliques (multi-valeurs) :

$$sim(a,b) = \frac{card(a) \cap card(b)}{card(a \cup b)}$$

  - pour les valeurs de descripteurs taxonomiques :

$$sim(a,b) = \frac{h(commonnode(a,b))}{\min(h(a), h(b))}$$

    Où :

    a et *b* : sont des valeurs des descripteurs.

    card : est la cardinalité de l'ensemble

    range : est la valeur absolue de la différence entre la borne supérieure et la borne inférieure de l'ensemble des valeurs

    h : est le poids (le nombre de niveaux) de l'arbre taxonomique

- *similarités globales :* elles sont calculées au niveau des cas ou des objets en agrégeant les similarités locales. Plusieurs similarités globales sont utilisées dans les systèmes de RàPC et aucune d'elles n'est universelles et restent dépendantes du domaine concerné. On peut citer :

- *Weighted Block-City*

$$sim(A,B) = \sum_{i=1}^{n} w_i sim_i(a_i, b_i)$$

- la distance de *Minkowski*. Si *r = 2* alors on retrouve la distance *Euclidienne* et si *r = 1* alors il s'agira de la distance de *Manhattan*.

$$sim(A,B) = \left[\frac{1}{n} \sum_{i=1}^{n} sim_i(a_i, b_i)^r \right]^{\frac{1}{r}}$$

- *Maximum based*

$$sim(A,B) = \max_i w_i sim_i(a_i, b_i)$$

Où :

n est le nombre d'attributs,

w_i est le poids (évalué en fonction de l'importance) de l'attribut *i*,

sim_i est la similarité locale calculée pour l'attribut *i*.

Il existe d'autres types de mesures de similarité qui tiennent compte des historiques, des séquences dans les cas, du temps, de l'espace, des structures complexes, des plans, des séries, etc.

Une fois la mesure de similarité établie pour un système donné, la remémoration des cas dans la base de cas se fait suivant un algorithme de recherche. Plusieurs types d'algorithmes peuvent être alors appliqués. L'algorithme *des K Plus Proches Voisins « KPPV »* (plus connus en Anglais sous le nom *K*-Nearest Neighbors (*K*-NN) [Weiss & Kulikowski, 1991]) est la méthode la plus habituellement utilisée. La méthode utilise donc deux paramètres : le nombre *K* et la fonction de similarité pour comparer le nouveau cas aux cas déjà classés [Yoshua & Chapados, 2003]. On peut également citer l'algorithme basé sur les approches inductives (KD-arbres, ID3, C4.5, etc.), les algorithmes appliqués à l'historique des séquences, l'algorithme de chemin de similarité, l'induction basée sur la connaissance, la recherche basée sur la structure (« template retrieval ») qui est similaire aux requêtes SQL, la recherche Case Retrieval Nets [Lenz, 1999] et l'algorithme basé sur l'exploitation des différentes vues sur les cas – Fish & Shrink – qui combine dynamiquement différentes mesures de similarité pendant la recherche [Börner, 1998].

Nous exploitons dans notre étude les similarités locales et globales, dans la phase de remémoration, qui seront en fonction des spécificités du diagnostic. Un état de l'art sera

consacré au chapitre 4 concernant les différents types de remémoration existant dans la littérature.

3.7.3. Phase d'adaptation

La phase d'adaptation dans le cycle du RàPC est le processus proposant une solution à un nouveau problème à partir des solutions appartenant aux cas sources remémorés [Lopez de Mantaras et al., 2005].

Fuchs et al. [1999] considèrent l'adaptation comme un plan dont l'état initial est la solution de départ et l'état final est la solution adaptée.

Lieber et al. [2004] considèrent que la phase d'adaptation consiste à effectuer un raisonnement par analogie : « *sachant que la solution du cas cible est à la solution du cas source ce que le cas cible est au cas source, connaissant le cas source et sa solution ainsi que le cas cible, que vaut la solution du cas cible ?* » L'adaptation termine « l'inférence analogique » en calculant la solution possible au problème du cas cible inspirée de la solution du cas source le plus similaire.

Cette phase peut se faire soit via une intervention humaine (manuelle) soit d'une manière automatique à l'aide d'algorithmes, de méthodes, de formules, de règles, etc. Concernant l'adaptation automatique, Wilke et Bergmann [1998] déterminent les principaux types d'adaptation automatique :

- ***L'adaptation générative*** part du fait que nous disposons de toutes les connaissances pour résoudre le problème à partir de zéro. Le cas retrouvé retrace le raisonnement ayant mené à la solution ;
- ***L'adaptation transformationnelle*** est contraire à la précédente, c'est-à-dire qu'on ne dispose pas de toutes les connaissances pour résoudre le problème à partir de zéro ;
- ***L'adaptation compositionnelle*** utilise deux ou plusieurs cas similaires remémorés pour effectuer l'adaptation en composant les différentes solutions proposées ;
- ***L'adaptation hiérarchique*** où les cas sont organisés à plusieurs niveaux dans la hiérarchie de généralisation.

D'autres types d'adaptation sont exploités à travers les différents travaux se trouvant dans la littérature que nous aborderons au chapitre 4.

Nous accordons une attention particulière à cette phase car elle est considérée comme la phase la plus complexe et la plus délicate à mettre en œuvre. Nous avons vu qu'il y a

plusieurs types d'adaptations. Nous exploitons l'adaptation générative et hiérarchique pour manipuler les cas de diagnostic du système de RàPC. Un état de l'art sera consacré à ce sujet au chapitre 4.

3.7.4. Phase de validation

Au cours de la phase de révision, la solution proposée à l'issue de la phase d'adaptation sera évaluée. Cette évaluation concerne plusieurs actions pouvant être employées [Mille, 1999] :

- tester la solution proposée dans le monde réel ;
- faire une introspection dans la base de cas en utilisant l'ensemble des descripteurs de problème et de solution afin de vérifier que les cas similaires ont donné entière satisfaction ;
- utiliser une autre méthode d'évaluation de la solution (simulateur, système expert classique, etc.).

La phase de révision consiste donc à continuer éventuellement l'élaboration de la solution cible si besoin. Par conséquent, si avec les précédentes actions la solution est jugée insatisfaisante alors elle va être corrigée. Ces corrections peuvent être apportées par :

- ***l'utilisateur***, qui peut donner sa propre évaluation par rapport à la solution fournie via le système de RàPC. On peut citer à titre d'exemple les travaux de Karoui et al. [2006] ;
- ***un expert humain***, qui reflète l'expertise du domaine considéré. Les travaux de Cordier et al. [2007] en sont l'exemple ;
- ***un processus automatique***, qui part du principe de l'auto-évaluation en utilisant la base de cas. A titre d'exemple les travaux de Malek et Kanawati [2004].

Le cas validé avec les explications de révision devient une source d'apprentissage importante pour faire évoluer les connaissances mobilisées par le raisonnement. Cependant, il n'existe pas encore de méthode standardisée pour rendre compte de la tâche de révision dans le RàPC.

Dans notre étude, cette phase va être exploitée avec des mesures mises en place spécialement afin d'intégrer que les cas contribuant à la qualité de la base de cas suivant des critères bien définis.

3.7.5. Phase d'apprentissage

Cette phase consiste à incorporer ce qui est utile à retenir dans la base de cas et permet de synthétiser les nouvelles connaissances qui vont être réutilisées ultérieurement. Cet apprentissage peut s'effectuer non seulement à partir du succès mais aussi de l'échec dans la résolution du problème cible. Le stockage d'un nouveau cas permet donc d'enrichir la base de cas permettant l'augmentation de l'expérience du système. De plus, Mille [2006] cite les points suivants :

- en cas de révision, la connaissance générale peut être modifiée et en particulier les connaissances duales liées aux tâches « retrouver » et « adapter » ;
- la mesure de similarité peut être affinée pour éviter de sélectionner une classe de solution erronée ;
- « *les connaissances d'influences exprimant la variation d'un descripteur de solution induite par la variation d'un descripteur de problème* » [Cordier et al., 2007], peuvent être affinées pour piloter l'adaptation. Ces connaissances d'influence sont duales des connaissances de similarité, elles sont directement liées aux poids utilisés pour pondérer la mesure globale de similarité ;
- les nouvelles dépendances peuvent être découvertes, etc.

Il peut cependant être utile de garder la « *trace* » de l'ensemble du cycle avec le détail des corrections faites. Même si on n'a pas encore pu mettre à jour les connaissances du système, cette trace pourra être utilisée pour considérer ce cas comme un modèle pour « corriger » en s'inspirant de cette correction, une nouvelle adaptation qui se ferait avec le même type de similarité. Le cas est donc organisé et indexé dans la base de cas selon plusieurs méthodes qui dépendent du type d'application considéré.

L'organisation de la base de cas dépend donc du nombre de cas et de leur organisation à l'intérieur de celle-ci. De ce fait, la base de cas évolue dans le temps au fur et à mesure de l'ajout de nouveau cas. Ceci pourrait perturber son organisation et sa structuration d'une façon générale. Pour y remédier, nous parlerons de la « maintenance de la base de cas ». Puisque la base de cas représente le cœur des systèmes de RàPC et est considérée comme la source de connaissance la plus sensible aux changements, alors des opérations de maintenance sont vitales pour le bon fonctionnement général du système complet de RàPC. Fort de ce constat, la maintenance de la mémoire du système qu'on voudrait mettre en place devient une étape cruciale dans son cycle de vie. Par conséquent, un état de l'art, sur les

travaux de la maintenance des systèmes de RàPC et plus spécifiquement de la base de cas, sera abordé au chapitre 3 suivi par une proposition de maintenance appropriée à notre système.

Roth-Berghofer [2003] définit une connaissance comme un ensemble de *containers de connaissances* qui est approprié pour de nombreuses tâches de diverses natures. Les containers de connaissance (*knowledge containers*) rassemblent donc les connaissances d'un système de RàPC. Ils vont être détaillés dans la section suivante.

4. Containers de connaissances

Richter [1998] définit les systèmes de RàPC comme des systèmes à base de connaissances (SBC). Ils exploitent quatre containers de connaissances distincts à savoir :

- *le container de vocabulaire :* contient toutes les informations sur les définitions et les structures utilisées. Lieber et al. [2004] définissent le container de vocabulaire comme étant l'ensemble des éléments de représentation atomiques utilisés pour représenter les cas dans la base de cas ;
- *le container des mesures de similarité :* contient les mesures nécessaires pour la recherche des cas. Ce container fait appel à ces mesures pour le calcul de la similarité ou la dissimilarité afin de déterminer, lors de la phase de remémoration, les cas sources les plus appropriés au nouveau cas cible ;
- *le container d'adaptation :* contient les règles de transformation de la solution et d'algorithmes décisionnels. Ce container contribue à la modification des cas sources pour s'adapter au mieux au cas cible ;
- *Le container de la base de cas :* représente le contenu et l'organisation de la base de cas. C'est le container dans lequel sont stockées toutes les expériences (cas).

Les trois premiers containers de connaissances sont élaborés avant que le système fonctionne, alors que la base de cas est remise à jour normalement dynamiquement. Selon Richter, chaque container peut porter presque toute la connaissance disponible, et les manipulations sur un container ont de petites répercussions sur les autres.

Wilk [1997] et Wilson [2001] confirment également que les containers de connaissance sont fortement liés entre eux. En effet, la connaissance nécessaire pour résoudre un problème donné peut résider dans l'un des containers comme cela peut être également dans un autre. Un système avec une connaissance complète et contenant beaucoup de cas nécessite très peu d'adaptation. Cependant, lorsqu'un système dispose d'une adaptation puissante, il peut avoir besoin de très peu de cas pour bien fonctionner.

Roth-Berghofer [2003] divise le cycle de RàPC en deux méta-phases : la première concerne une méta-phase d'application dans laquelle trois phases du cycle sont concernées (*remémoration*, *adaptation* et *révision*) et la deuxième concerne une méta-phase de maintenance impliquant les phases de *restauration*, *scrutation* et *mémorisation*. Ainsi, ces deux méta-phases sont en lien direct avec les containers de connaissance. Le même auteur affirme également que les quatre containers de connaissance sont liés entre eux et il va même exploiter les trois premiers containers durant le développement de son système de RàPC appelé « *empolis orenge* » (un système de maintenance pour des usages commerciaux et de recherche universitaire qui emploie la méthodologie INRECA) et le dernier container de connaissance est exploité durant la résolution de l'actuel problème.

La Figure 1.9 présente les quatre containers de connaissance ainsi que leurs relations.

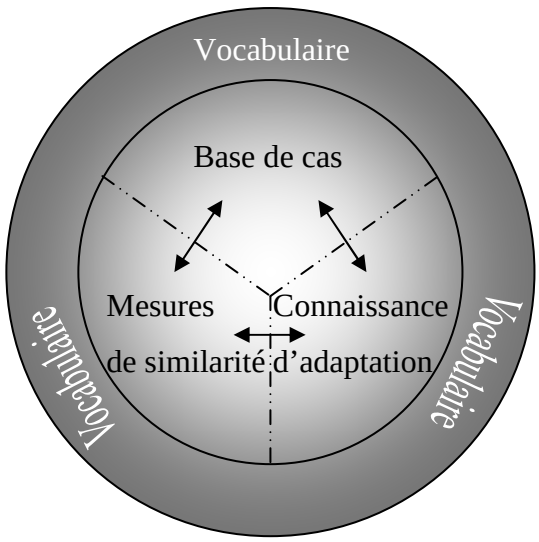


Figure 1.9. Les containers de connaissances [Roth-Berghofer, 2003]

Les flèches signifient que la connaissance peut être transférée d'un container de connaissance à un autre. Le container de vocabulaire représente la base des trois autres containers [Roth-Berghofer, 2003].

Lamontagne et Lapalme [2002] présentent un modèle générique d'un système de RàPC dans lequel ils combinent le cycle du RàPC avec les connaissances permettant de préserver et d'exploiter les expériences passées. Ce modèle générique est présenté sur la Figure 1.10.

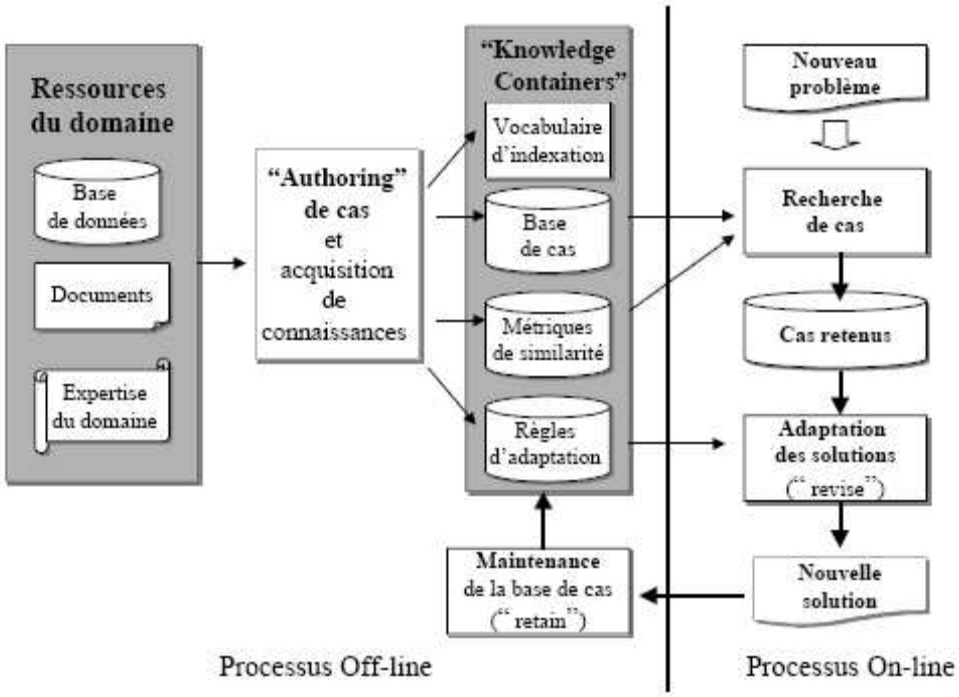


Figure 1.10. Modèle générique d'un système de RàPC [Lamontagne et Lapalme, 2002]

Ce modèle générique met en évidence l'aspect « connaissances ». Il est découpé en deux parties : on-line et off-line.

La partie on-line comporte les phases du cycle de RàPC.

La partie off-line implique les ressources utilisées et la phase d'acquisition et représentation des connaissances ainsi que des *containers de connaissances*. La partie construction (authoring) du cas et acquisition de connaissances guide la structuration initiale de la base de cas et des autres connaissances du système à partir de différentes ressources tels que des documents, des bases de données ou des experts du domaine.

Nous nous intéressons particulièrement à ce modèle dans lequel nous exploitons les containers de connaissance ainsi que les phases du cycle de RàPC.

5. Domaines d'application et conditions d'utilisation du RàPC

Le RàPC possède de nombreuses applications dans des domaines très divers permettant d'accomplir des tâches de différentes natures. Parmi les domaines dans lesquels le RàPC est plus couramment utilisé, nous pouvons citer les domaines médical et industriel, commercial, judiciaire, l'analyse financière, la maintenance, l'enseignement, les services de consultation.

Nous retrouvons également plusieurs types d'applications comme la gestion des connaissances, la planification, l'aide à la décision, la classification, la conception, la navigation des robots, la configuration, le maintien des contraintes de temps appliquées à de petits avions, la justification, la sécurité routière et le diagnostic qui représente justement notre centre d'intérêt.

Selon les auteurs, différentes typologies d'application du RàPC sont proposées. Elles sont dépendantes du domaine abordé et de la nature de la tâche à réaliser.

5.1. Typologies d'applications

Deux types d'applications sont considérés par Kolodner [1993] :

- ***les systèmes de résolution de problèmes*** dans lesquels les solutions de nouveaux problèmes sont dérivées en se servant des solutions déjà connues comme des guides. Ils regroupent des tâches de conception, de planification et de diagnostic ;
- ***les systèmes interprétatifs*** dans lesquels les situations déjà connues servent de contexte pour les nouvelles situations. Ils regroupent des tâches de justification, évaluation, interprétation et classification.

Watson et Marir [1994] séparent les applications, selon le type d'utilisation, en deux catégories à savoir ***les applications commerciales et académiques***. Tandis que Althoff et al. [1995] les décomposent selon le type de la tâche à réaliser :

- ***Les tâches de classification*** comprenant les systèmes de prédiction, de help desk, d'évaluation et de diagnostic ;
- ***Les tâches de synthèse/planification*** comprenant les systèmes de configuration, de conception et de planification.

Althoff et Bartsch-Spörl [1996] quant à eux, divisent les domaines d'application du RàPC en deux classes. Celle relatives aux tâches :

- ***analytiques*** qui sont sous forme de situations. Ces situations doivent être classifiées, retrouver le cas correspondant et exploiter la solution. On retrouve dans ce type de tâche la classification d'objet, le diagnostic médical et la sélection de produits dans le e-commerce ;
- ***synthétiques*** nécessitant une *planification*, une *configuration* et une *construction* de la solution. Elles se présentent sous forme de problèmes auxquels il faut proposer une solution et décrire les conditions d'applications de la dite solution. Ce type de tâche englobe le transport, l'architecture, les systèmes d'aide à la décision, l'enseignement et la gestion des ressources.

Une autre distinction est faite par Althoff [2001] qui introduit la notion de la hiérarchie des applications par rapport à la complexité de la résolution de problème. En effet, il propose quatre niveaux hiérarchiques comme cela est montré sur la Figure 1.11 :

- ***la classification*** se trouve au premier niveau de la hiérarchie dans laquelle la solution du problème est en relation avec la sélection d'une ou plusieurs classes ;
- ***le diagnostic*** vient juste après car il est considéré comme une généralisation de la classification. Cette généralisation concerne la connaissance générale du domaine qui va rechercher les informations nécessaires (symptômes) dans un processus de diagnostic ;
- ***l'aide à la décision*** se trouve au troisième niveau hiérarchique et distingue les symptômes des solutions de problèmes qui ne sont pas toujours directes ni visibles ;
- ***la gestion des connaissances*** est une application plus générale et plus complexe dans laquelle aucune méthode de raisonnement ne peut être utilisée directement.

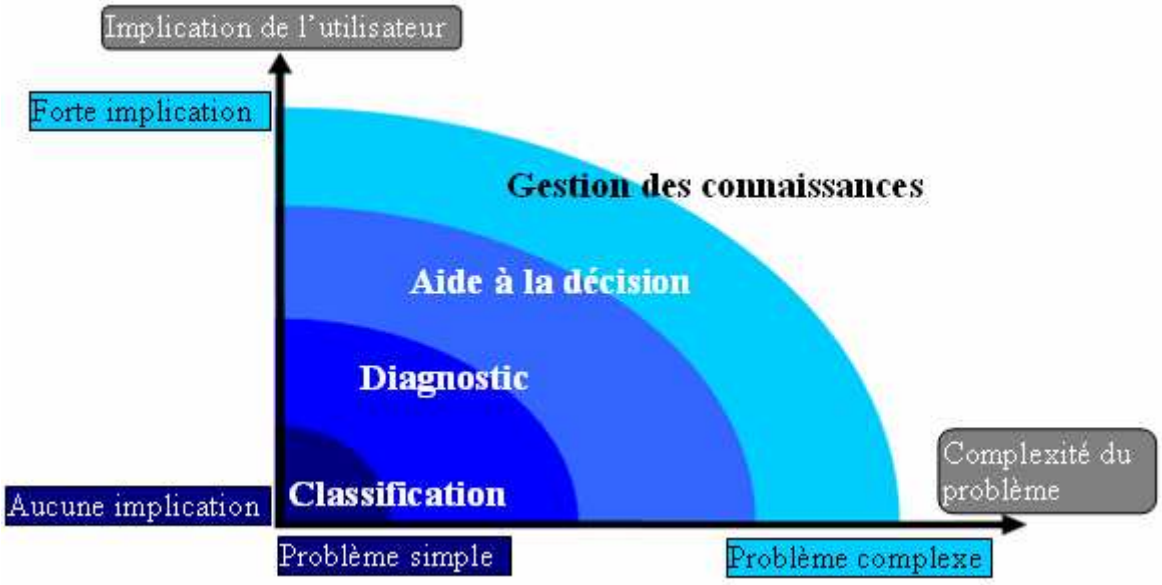


Figure 1.11. Niveaux hiérarchiques des applications des systèmes de RàPC selon Althoff [2001]

Et enfin Richter [2006] propose une catégorisation plus conventionnelle en s'inspirant de la catégorisation faite par Althoff et Bartsch-Spörl [1996]. Les deux types de systèmes de RàPC sont :

- ***résolution de problèmes***, ce sont les systèmes qui s'occupent des tâches analytiques ;
- ***prise de décision*** qui concerne les tâches synthétiques.

Selon les niveaux hiérarchiques proposés par Althoff, nous nous intéressons dans notre étude aux deux premiers niveaux à savoir la classification et le diagnostic. Nous commençons par la classification dans laquelle nous traitons des problèmes simples avec peu d'implication de l'utilisateur. Ce premier niveau concerne les systèmes orienté *mining* que nous étudierons au chapitre 2. Ensuite, vient le deuxième niveau, qui concerne le diagnostic, dans lequel la complexité du problème à traiter est plus importante que les problèmes de classification et que le degré d'implication de l'utilisateur est également plus importante mais pas si forte que les domaines de l'aide à la décision et la gestion des connaissances. Ce deuxième niveau concerne les systèmes dont leur formalisation du cas est plus sophistiquée que celle de la classification. De plus, une modélisation des connaissances du système à diagnostiquer est nécessaire. Nous parlerons donc des systèmes orientés *connaissances*.

Au delà des différentes typologies d'application, le RàPC ne s'applique que sous certaines conditions.

5.2. Conditions d'applications

L'approche du RàPC présente de nombreux atouts. Dans la conception du système de RàPC, nous intégrons la partie dynamique de la base de cas, contrairement aux systèmes à base de connaissance, ce qui facilite grandement sa maintenance. Le RàPC permet également la réutilisation des solutions existantes, la pro-activité des solutions proposées (amélioration avec le temps) et se caractérise par son pouvoir d'adaptation face aux changements de l'environnement.

Initialement, la formalisation des cas n'était qu'une retranscription de l'expérience experte, ce qui privilégie le choix du RàPC par les industriels par rapport aux systèmes à base de connaissance.

Toutefois, l'évolution du RàPC nous amène à acquérir de nouvelles connaissances, ce qui complexifie aujourd'hui la formalisation du cas et de la connaissance associée. Il faut également définir de « bonnes » mesures de similarité et de trouver une « bonne » stratégie d'adaptation.

De plus, l'application du RàPC est conditionnée par un certain nombre d'hypothèses par rapport au type d'application ou le domaine étudié [Kolodner, 1996] :

- ***similarité*** : l'utilisation d'anciens cas pour en résoudre de nouveaux qui lui sont similaires est fondée sur l'hypothèse que des problèmes semblables peuvent être résolus par des solutions semblables [Schank, 1982 ; Riesbeck & Schank, 1989] ;
- ***facilité d'adaptation*** : les différences entre les descriptions des problèmes des cas sont faibles et peuvent être évaluées facilement ;
- ***typicalité*** : les problèmes se répètent et les expériences apprises peuvent être essentielles dans le futur ;
- ***régularité*** : les mêmes actions menées (solutions proposées) dans les mêmes conditions (description d'une situation) ont des conséquences identiques ou proches ;
- ***cohérence*** : les différents petits changements qui peuvent apparaître dans l'univers nécessitent seulement des petits changements dans la façon d'interprétation et engendre que de faibles transformations dans les solutions.

Main et al. [2000] estiment que l'approche du RàPC ne peut être appliquée que si un certain nombre des caractéristiques des problèmes et de leur domaine sont réunies.

- ***L'existence des modèles sous-jacents dans le domaine,*** si la prise en compte des facteurs qui ont conduit au succès ou à l'échec d'une solution est impossible dans la description du cas, ou si le processus est aléatoire alors le RàPC ne serait pas adapté à ce genre d'application ;
- ***L'importance de l'apport de l'adaptation des solutions des cas existants dans la base de cas,*** si la construction entière d'une solution n'est pas significative par rapport à l'obtention de la même solution par adaptation alors le système à étudier peut être conçu avec une méthode autre que le RàPC. Cet apport est jugé suivant un certain nombre de critères tels que : le traitement de l'information et le temps de résolution mis par le système ;
- ***La reproduction des situations,*** si les cas ne sont pas suffisamment similaires aux problèmes posés pour qu'ils soient adaptés, alors il vaut mieux utiliser un modèle qui construit complètement la solution qu'une approche qui s'appuie sur la réutilisation des solutions ;
- ***Les nouveaux cas et les exceptions,*** si le domaine étudié ne fait pas apparaître de nouvelles situations ou d'exceptions alors il convient de mettre en place une modélisation gérée par des règles.

Si toutes les conditions nécessaires sont réunies, alors la mise en place d'un système de RàPC est plausible et l'étude du domaine d'application peut commencer. Cette étude dépend donc de la nature de l'application, de ses caractéristiques et des objectifs à atteindre. Les systèmes de RàPC diffèrent entre eux par la formalisation du cas, l'utilisation de modèles de connaissance, les étapes du cycle et les différents algorithmes et méthodes utilisés dans chacune des étapes. Les conditions d'applications définies ci-dessus sont réunies par : la similarité, la facilité d'adaptation, la typicalité, la régularité et la cohérence, pour définir un système de diagnostic orienté connaissance. Toutefois, nous allons développer dans un premier temps un système de classification orienté mining qui est un cas particulier du RàPC.

6. Conclusion

Pour développer un système de diagnostic pour un équipement industriel, notre choix s'est porté sur le raisonnement à partir de cas (RàPC). Cette approche est privilégiée par les industriels, car elle permet de raisonner sur un nombre restreint de cas et ainsi permet de

s'affranchir d'une longue étape de récolte de données, dédiée à l'entrainement du modèle. Ceci nous a amené à présenter les notions du diagnostic et à choisir parmi les définitions existantes celle de l'Afnor [2001]. Puis, nous avons décrit de manière détaillée les systèmes de RàPC. En effet, cette approche se trouve à l'intersection des sciences cognitives et de l'intelligence artificielle et permet d'exploiter les connaissances du système. Nous avons détaillé les différentes phases du cycle permettant la manipulation du mécanisme de connaissances. Ceci nous a permis de constater que ces phases dépendent les unes des autres et spécialement les deux phases de remémoration et d'adaptation. Nous avons également précisé qu'il existe trois modèles de RàPC qui dépendent entre autre de la représentation du cas, et que nos systèmes seront bâtis sur le modèle structurel. En effet, ce type de modèle est utilisé dans les domaines d'études qui sont *a priori* connus notamment le diagnostic et que les cas doivent être structurés pour qu'ils puissent caractériser le système à diagnostiquer et ses composants.

Nous avons abordé les containers des connaissances : le vocabulaire, les mesures de similarité, l'adaptation et la base de cas, qui sont exploités par le système de RàPC. Ils sont liés entre eux et la moindre modification de l'un des containers peut se répercuter sur l'autre. Ces quatre containers sont mis en place dans une première étape, étape off-line précédant le fonctionnent du système, et donc l'évolution du cycle de RàPC. Lamontagne et Lapalme [2002] ont proposé un tel modèle qui met en relation les ressources du domaine, tels que les bases de données et les documents techniques, et les containers de connaissance dans un processus off-line avec les phases du cycle dans un processus on-line. Nous prenons ce modèle comme référence pour la création d'un système de diagnostic par RàPC dédié à un équipement industriel au chapitre 2.

De plus, ce premier chapitre nous a permis de poser les jalons afin d'avoir des critères de comparaison entre les systèmes de diagnostic par RàPC, que nous recensons au chapitre 2.

Chapitre 2

Etat de l'art des systèmes de diagnostic industriel par le raisonnement à partir de cas et démarche adoptée

1. Introduction

Ce chapitre débute, à la section 2, par un état de l'art relatif aux différents systèmes de diagnostic par RàPC existant dans la littérature. L'analyse de ces systèmes est faite en fonction de leurs différentes représentations du domaine et par rapport aux phases du cycle manipulées. Cette étude a pour objectif la comparaison de ces systèmes selon les points évoqués et de situer nos travaux par rapport à l'existant afin de caractériser le système d'aide au diagnostic à mettre en place. Deux types de systèmes sont étudiés.

Le premier type de système orienté *mining* s'apparente aux systèmes d'apprentissage automatique qui prennent en compte l'ensemble des cas bruts dès l'entrée du système [Pan et al., 2007]. La modélisation est simple à mettre en œuvre, et fait écho à un avantage concédé aux systèmes de RàPC par rapport aux systèmes à base de connaissances.

Nous mettons en place ce type de système à la section 3.1, inspiré d'un modèle de RàPC dans les travaux de Lamontagne et Lapalme [2002] en proposant une formalisation du cas et une phase de remémoration. La représentation du cas se fait sous forme d'attribut-valeur, mais la base de cas comporte un nombre conséquent de cas qui lui-même comprend un nombre important d'attributs. Ce nombre important de cas dans la base de cas pallie l'absence de la phase d'adaptation. Etant donné l'éventuelle explosion combinatoire de cas dans ce type de système, nous abordons une phase de maintenance de la base de cas.

Toutefois, une deuxième formalisation du système de RàPC plus complexe et plus générique est possible. Dans ce cas, nous évoquons le deuxième type de système, système orienté *connaissance (knowledge)*. En effet, ce deuxième type de système prend appui sur des modèles de connaissance, que nous pouvons aisément définir à partir de la modélisation du système complexe à diagnostiquer. Nous manipulons une représentation du cas associé à des modèles de connaissances du domaine. Ce type de système peut facilement raisonner sur un ensemble restreint de cas. Dans une autre optique que le système précédent, nous nous intéressons à développer une phase d'adaptation prenant appui sur des modèles de connaissance.

Nous développons également dans ce type de système, à la section 3.2, une formalisation du cas à partir de la définition de référence du diagnostic et deux modèles de connaissance : un modèle hiérarchique des composants et un modèle de contexte. Cependant, ce type de système nécessite le développement d'une phase d'élaboration spécifique, d'une phase de remémoration et d'adaptation qui seront décrites tout au long de ce mémoire.

Enfin, nous illustrons les caractéristiques du système orienté connaissance et notamment la gestion des connaissances sur un moteur à explosion 1.5 dCi K9K 105ch de la société Renault (http://v3.renault.com/cfm/ module-K9K/fr/index.html).

2. Les principaux systèmes de RàPC dédiés au diagnostic industriel

2.1. Présentation générale

Il y a différents types de systèmes de diagnostic existant dans la littérature, allant du diagnostic médical, tel que CASEY qui traite des pathologies cardiaques et PROTOS [Bareiss et al., 1993] qui s'intéresse au diagnostic des malentendants, aux système industriels en passant par les systèmes « help desk » tel que Chekmate pour les imprimantes industrielles et Caseline concernant les moteurs d'avion Boeing 747-400 jusqu'aux systèmes de diagnostic industriel des machines complexes. Dans le cadre de ce dernier type de systèmes de diagnostic, nous abordons huit principaux systèmes allant des systèmes qui s'occupent des moyens de transport tels que Creek [Aamodt, 2004] dédié aux pannes de voitures, Cassiopee [Bergmann et al., 2003] qui s'intéresse au dépannage de moteurs d'avion Boeing 737, et ICARUS [Varma, 1999] qui développe un système appliqué aux locomotives, en passant par d'autres systèmes, qui s'intéressent aux différents produits de plusieurs natures et de différentes machines tels que Case-Based Reasoning for Gas Turbine Diagnostics [Devany & Cheetham, 2005] relatif aux turbines à gaz de General Electric Energy, FormTool [Cheetham & Graf, 1997], relatif aux couleurs de plastique dans General Electric, $Nodal_{CBR}$ [Cunningham & Smyth, 1994], relatif aux alimentations d'énergie de mode de commutation (SMPS), et Patdex [Richter & Wess, 1991] relatif aux machines complexes, jusqu'aux systèmes de supervision tels que Pad'im [Mille, 1995] qui étudie la conception de systèmes de supervision dans le milieu industriel.

2.1.1. Typologie des systèmes de diagnostic

Althoff et al. [1995] ainsi que Koton [1988] définissent deux types de systèmes de diagnostic basés sur le RàPC :

- ***Systèmes de Help Desk :*** ces systèmes sont conçus dans l'objectif d'aider des néophytes à proposer une solution à un problème donné. Ces systèmes sont généralement conçus pour être des supports de décision pour les utilisateurs non experts en maintenance.

- ***Systèmes de diagnostic général et recouvrement de pannes :*** ces systèmes traitent les applications dans lesquelles sont recherchées les causes de pannes. Ces systèmes couvrent principalement deux axes d'application à savoir : le *diagnostic industriel des machines complexes* et le *diagnostic médical*.

Nous pouvons ainsi schématiser les types de systèmes de RàPC dédiés au diagnostic comme le montre la Figure 2.1.

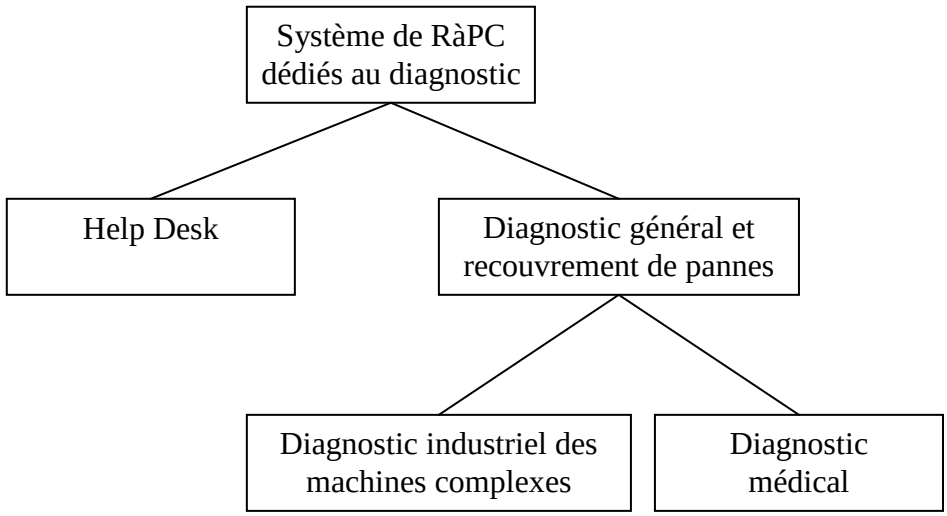


Figure 2.1. Types de systèmes de diagnostic basés sur l'approche du RàPC

2.1.2. Systèmes de type diagnostic médical

Concernant les systèmes de diagnostic médical, nous pouvons citer deux systèmes : CASEY [Koton, 1988] et Protos [Bareiss, 1989].

CASEY est un système de suggestion de thérapie pour les patients présentant des problèmes d'arrêt cardiaque. CASEY est construit autour d'un système expert basé sur des modèles déjà existants. Le cas dans CASEY est représenté sous forme de graphe d'attributs-valeurs. Un modèle causal est incorporé dans le système Casey. Il intervient lors de la remémoration, de l'adaptation et de la mémorisation. CASEY lance une recherche d'un cas similaire au nouveau patient dans la base de cas. Le modèle causal est utilisé pour déterminer

quelles sont les caractéristiques les plus importantes déterminant l'explication causale des cas précédents. Ensuite, il détermine les caractéristiques importantes du nouveau cas et donne par la suite à ces caractéristiques un plus grand poids pour les apparier. Dans la phase d'adaptation, la technique utilisée est l'adaptation basée sur des modèles (*model-based adaptation*) qui s'appuie sur des stratégies de réparation afin d'adapter les solutions au nouveau. Il y a trois types de stratégies de réparation correspondantes aux trois parts de solutions : explication causale, diagnostic et thérapie. Le principe de l'adaptation consiste à modifier l'explication causale précédente pour former le nouveau cas. Les stratégies de réparation modifient l'explication causale en ajoutant ou supprimant des nœuds et des liens.

Protos [Bareiss, 1989] est un système de RàPC pour l'aide au diagnostic des malentendants. Le cas est représenté par un vecteur de caractéristiques. Il n'y a aucun modèle utilisé et la phase de remémoration est basée sur l'algorithme des k plus proches voisins. Protos ne dispose pas de phase d'adaptation. Si la solution proposée par le système n'est pas satisfaisante alors une phase d'apprentissage s'enclenche. De ce fait, le système commence à rechercher une autre solution sinon il doit accepter une solution proposée par l'utilisateur.

2.1.3. Systèmes de type help desk

Nous citons deux systèmes de type « help desk » dédiés à des applications industrielles à savoir : Checkmate [Grant et al., 1996] et Caseline [Watson & Marir, 1994].

Checkmate a été conçu dans le but d'établir un diagnostic de pannes concernant les imprimantes industrielles. Ce système est destiné à assurer des fonctions de documentation et de partage d'informations via un système automatique de diagnostic et de réparation. Checkmate est un système conversationnel, le cas dans ce système est sous forme de description de la panne, du contexte et de l'action de réparation. La communication entre l'utilisateur et le système est traduite en données symboliques. La phase de remémoration est basée sur l'algorithme des k plus proches voisins. Cette phase est appuyée par des règles permettant une recherche plus avancée. Il n y'a pas de phase d'adaptation dans ce système.

Caseline est un système de diagnostic dédié aux Moteurs d'avion Boeing 747-400. Le cas peut être représenté sous forme de liste d'attribut-valeur ou de représentation textuelle. La description du cas comporte le numéro, le nom et la description de la panne. La phase de remémoration est soit basée sur l'algorithme des k plus proches voisins, soit sur les arbres d'induction. Concernant la phase d'adaptation, comme le système Checkmate, il ne dispose pas de cette phase.

Dans cette thèse, nous nous intéressons plus particulièrement aux systèmes de diagnostic industriel des machines complexes en établissant un état de l'art concernant huit systèmes. Cet état de l'art est articulé autour des principaux points suivants : *la modélisation des connaissances, la représentation du cas, la phase de remémoration, la phase d'adaptation ainsi que la phase de maintenance de la base de cas*.

L'état de l'art s'achève avec la présentation d'un tableau récapitulatif, permettant de comparer les huit systèmes étudiés selon les points auxquels nous nous sommes intéressés.

2.2. Case-Based Reasoning for Gas Turbine Diagnostics

Le système présenté dans [Devany & Cheetham, 2005] a été construit pour le diagnostic de pannes des turbines à gaz dans General Electric Energy à Atlanta, GA. Cette application inclut la précision[1], la maintenabilité[2], la modularité[3], le paramétrage[4], la robustesse[5], et l'intégration du système dans une infrastructure existante. Ce système a été intégré dans un environnement de production en 2004, l'architecture du système est présentée à la Figure 2.2.

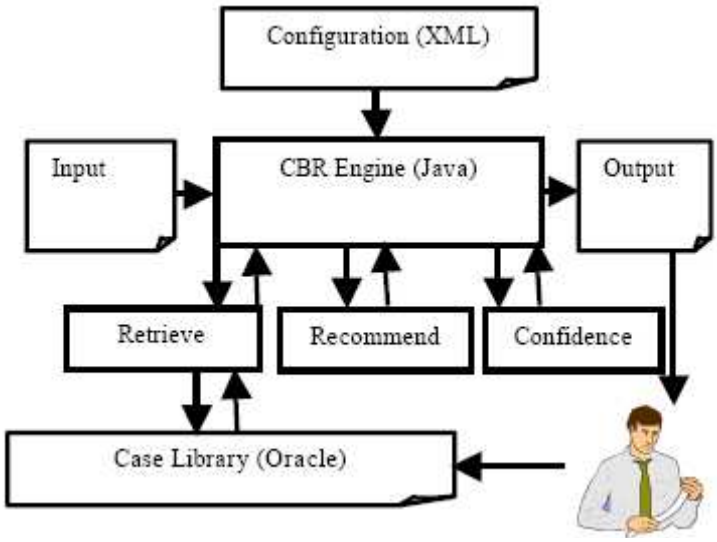


Figure 2.2. Architecture du système Gas Turbine [Devany, 2005]

[1] L'exactitude de la réponse fournie qui est en fonction d'un degré de confiance préétabli

[2] Le pouvoir de réduction de l'intervention humaine suite la solution donnée au problème de diagnostic

[3] La facilité d'implémentation des différents algorithmes dans les différentes étapesphases de RàPC

[4] L'ensemble de valeurs qui sont déterminées par les ingénieurs de connaissance dans le cadre du déploiement du système

[5] Le pouvoir des algorithmes de faire face à la présence du bruit et/ou aux données incomplètes et erronées

La Figure 2.3 montre le prototype interface utilisateur en langue espagnole [Flores-Loredo et al., 2005]. Sur le côté droit, on trouve le signalement de l'état du processus et de la présence de pannes. La turbine à gaz peut être dans les états suivants : démarrage (arranque), arrêt (paro), synchronisation (sincronizacion) ou en production (generaci'on). Au milieu, les fenêtres montrent la probabilité d'occurrence des pannes avec une valeur numérique, et le moment exact pendant lequel la panne s'est produite.

Les cas sont sous la forme de nœuds dans un plan contenant les symptômes, les pannes et les facteurs extérieurs organisés d'une façon hiérarchique. Les cas représentent des valeurs de données décrivant les incidents qui proviennent des capteurs physiques se trouvant au bord de la turbine. Ces données sont collectées sur le site puis transmises et traitées au service M&D à Atlanta. Ce système utilise un modèle probabiliste indiquant la présence ou l'absence de pannes. Ce modèle est représenté par des réseaux bayésiens. Tous les nœuds du réseau bayésien sont des variables continues qui sont obtenues à partir d'un simulateur. Le modèle permet la quantification des signaux en trois valeurs : signal passe haut, passe bas ou stable.

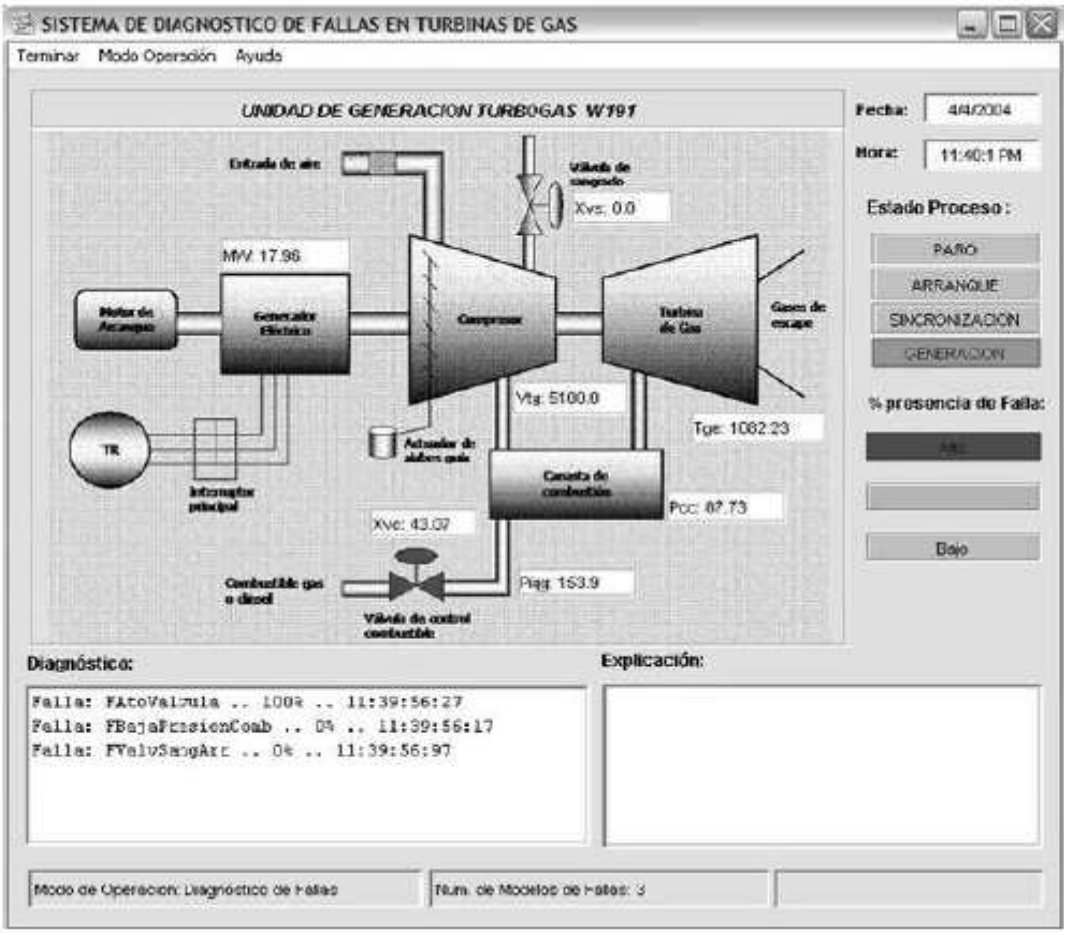


Figure 2.3. Interface utilisateur du prototype du système Gas Turbine [Flores-Loredo et al., 2005]

Dans la phase de remémoration, la base de cas est sollicitée concernant les événements antérieurs qui sont les plus similaires à l'incident courant à diagnostiquer.

Lors de cette phase, une série de variables est utilisée, chaque variable correspond à un signal. Ce signal est discrétisé avant d'être présenté au réseau bayésien. Ensuite, selon ces valeurs, un calcul de probabilité *a posteriori* est établi. Ainsi, la nature de la panne est déterminée. Cette phase doit donner un seul résultat de diagnostic qui vient d'un ensemble discret de candidats potentiels. Par conséquent, il n'y a ni phase d'adaptation ni phase de maintenance de la base de cas.

2.3. Creek

Le système Creek proposé dans [Aamodt, 2004] a été développé au début des années 90 et il est considéré comme l'un des premiers systèmes de diagnostic technique. Creek est dédié au diagnostic de pannes de voitures. Les cas ont une représentation orientée objet. Ils sont représentés par un ensemble de concepts formant une liste de descripteurs. Il y a trois types de descripteurs, à savoir : entité, relation et valeur. Creek utilise un modèle qui est le modèle général du domaine. Ce modèle est un intégrateur de résolution de problèmes et d'apprentissage (cf. Figure 2.4). Il permet de définir les symptômes, la généralisation des défaillances, du domaine d'application en l'occurrence des voitures, et du raisonnement.

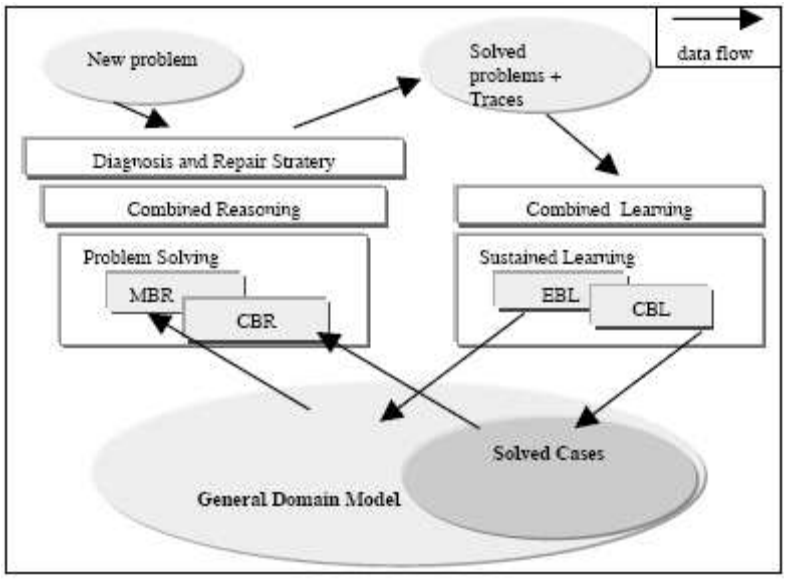


Figure 2.4. L'architecture fonctionnelle du système Creek [Aamodt, 2004]

La remémoration des cas dans Creek est orientée adaptabilité. Cette recherche s'appuie sur une mesure de similarité sémantique prenant en compte le modèle de connaissance générale du domaine. L'évaluation de la similarité est un processus qui est composé de deux étapes : la première dans laquelle les indices sont utilisés pour extraire un ensemble de cas similaires et la deuxième étape pendant laquelle un examen plus approfondi des cas a lieu. Cet examen s'appuie sur la connaissance générale du domaine permettant de générer des explications des appariements caractéristique par caractéristique. Dans la phase d'adaptation, des critères sont utilisés pour justifier le cas le plus adaptable tels que la pertinence et la cohérence. Ensuite, la trace de raisonnement et la généralisation d'attributs sont utilisées afin d'adapter le cas remémoré.

2.4. Cassiopee

Cassiopee est un système dédié au diagnostic des moteurs d'avions Boeing 737 implémenté dans le logiciel Kaidara (cf. Figure 2.5) décrit dans [Bergmann et al., 2003].

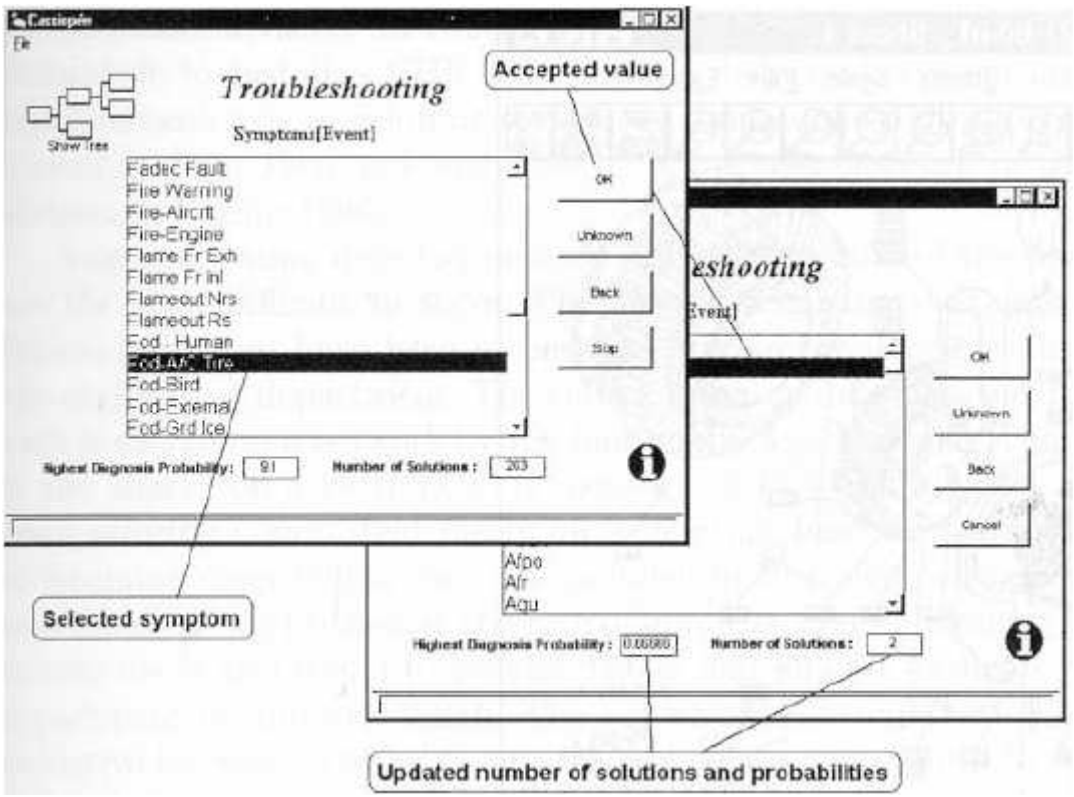


Figure 2.5. Un aperçu du système Cassiopee [Bergmann et al., 2003]

Les cas sont décrits par un ensemble d'attributs-valeurs. Les descripteurs sont sous forme d'une liste hiérarchisée formant une taxonomie avec des relations de classes et de sous-

classes. Cassiopee manipule deux modèles : un modèle conversationnel et un modèle de poids ; ainsi que des règles. Le modèle conversationnel est sous forme d'un système structurel organisé selon un arbre de décision.

Le modèle de poids permet de donner des priorités aux différents descripteurs selon la réussite ou l'échec de l'utilisation de la solution du cas au problème rencontré. La phase de remémoration s'appuie sur des similarités locales et une similarité globale. La similarité locale concerne les attributs du cas. La similarité globale concerne les classes d'objets et elle est définie par une fonction d'agrégation pondérée. L'adaptation est transformationnelle. Elle permet de modifier la solution proposée par le système selon les divergences de la partie problème.

2.5. ICARUS

ICARUS (Intelligent Case-based Analysis for Railroad Uptime Supportest) utilisé par les systèmes « GE Transport » est un système de RàPC pour le diagnostic de pannes des locomotives [Varma, 1999]. Une description du processus d'utilisation de l'historique des données de réparations et d'expertises pour la génération des cas et de validation est abordée. La création de ce système est motivée par le grand volume de données générant des erreurs. Pour une identification fiable et conforme aux problèmes de la locomotive, ces données deviennent donc difficiles à examiner par un opérateur humain. En plus, cette indentification concerne quotidiennement des centaines de locomotives. L'objectif du projet était de construire un outil qui pourrait prendre le registre d'erreurs en entrée et produire les « n » codes de réparation associés à des valeurs de confiance.

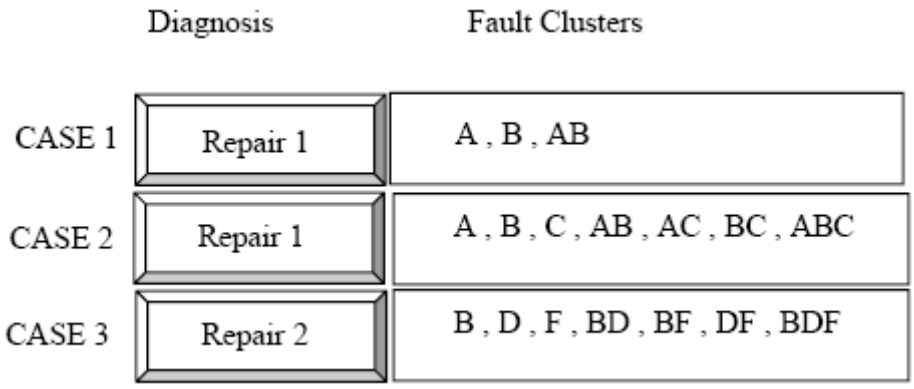


Figure 2.6. La représentation du cas avec les clusters de pannes pour le système ICARUS [Varma, 1999]

Les cas de diagnostic de la base de cas n'étaient pas facilement disponibles, de ce fait, ils ont dû être reconstruits par extraction à partir de l'historique de réparation. A partir de cette extraction qui est une opération très sensible, des experts interviennent pour valider l'exactitude de la date de réparation des cas. La représentation du cas est illustrée par la Figure 2.6.

Les cas cibles sont extraits à partir des registres (carnets de bords) d'erreurs de la locomotive. Ces derniers sont collectés à bord de la locomotive et sont périodiquement téléchargés à partir d'une base de données pour former un cas lorsqu'il y a un besoin de diagnostic. Dans la phase de remémoration, une extraction de caractéristiques a été nécessaire pour identifier les index qui unifieraient la représentation de cas et leurs appariements. Les cas peuvent ainsi s'apparier en tenant compte de :

- la présence ou l'absence des codes de pannes ;
- la fréquence du code de pannes ;
- les combinaisons des codes de pannes ;
- le temps des occurrences du code de pannes (si la fréquence du code de panne est amenée à augmenter pour une réparation donnée) ;
- les Indicateurs d'anomalies dans les données de paramètres (si les paramètres continus sont en dehors des spécifications) ;
- la séquence d'information dans l'occurrence du code de pannes (si les codes de pannes sont produits à plusieurs reprises dans un certain ordre).

Un poids est utilisé pour chaque descripteur de cas. Ce poids est destiné à représenter la capacité de chaque cluster par rapport à son pouvoir d'isolement d'un code spécifique de réparation. Si un cluster contient uniquement des cas contenant un seul code spécifique de réparation alors il aura un poids de 1, sinon, si ce cluster contient les cas contenant des codes multiples de réparation, son poids est alors largement diminué.

$$\text{Poids du cluster}_i \text{ de pannes} = \text{Max}_j\,[P(\text{Réparation}_j/\text{cluster}_i)]$$

Une fois que les poids des clusters de pannes sont connus, l'appariement des cas est alors simple. Lorsqu'un nouveau diagnostic est demandé, une identification de la locomotive présentant des problèmes est faite. La base de données de registre d'erreurs est sollicitée pour des codes de pannes qui se sont produits dans les "n" jours précédant cette demande. Le degré d'appariement entre un nouveau cas et un cas source est calculé comme suit :

$$\frac{[\sum PoidsDesClustersCommunsEntreLeCasremémoréEtLeNouveauCas]^2}{[\sum PoidsDesClustersDansLeCasremémoré]\times[\sum PoidsDesClustersDansLeNouveauCas]}$$

Le code de réparation lié au cas ayant le degré le plus élevé retourne le diagnostic associé. Cependant, le système IRACUS n'utilise aucune adaptation dans son cycle de RàPC.

2.6. General Electric Plastics sites (FormTool)

Cheetham et Graf [1997] ont mis en place un système pour définir et produire des couleurs de plastique souhaitées par le client. En effet, General Electric (GE) doit faire en sorte de faire correspondre le plastique à la norme de la couleur du client. Ainsi la combinaison des couleurs est évaluée afin de choisir la formule optimale de couleur. Le cas à la forme d'attributs-valeurs. Les attributs sont considérés comme des critères à travers lesquels l'appariement de couleur doit satisfaire :

- la couleur du plastique correspondant aux normes dans les conditions multiples d'éclairage ;
- la quantité de couleur suffisante pour cacher la couleur du plastique ;
- le coût de la formule de colorant qui doit être la moins onéreuse possible ;
- la densité optique représentant la quantité limitée de lumière qui peut être transmise par le plastique.

La modification de la couleur ne doit pas changer quand le plastique est exposé aux différentes températures. L'algorithme des k plus proches voisins est utilisé lors de la phase de remémoration. Des termes linguistiques ainsi qu'une fonction floue de préférence sont utilisés pour calculer la similarité pour un seul attribut à la fois. Cette fonction représente un vecteur de valeur, chaque valeur peut être combinée, via une agrégation pondérée, afin de produire une valeur de similarité robuste. Cette fonction permet de transformer une valeur quantitative de chaque attribut en une valeur qualitative. Ainsi, des intervalles sont considérés orientés par des experts. La mesure de similarité utilisée dans cette étude est censée remémorer le cas le plus facilement adaptable. L'adaptation appliquée est une adaptation transformationnelle. Elle est réalisée en exécutant une recherche qui change à plusieurs reprises les chargements des colorants dans la formule remémorée et évalue ensuite la nouvelle similarité. Cette adaptation est basée sur une fonction qui peut évaluer précisément

l'effet d'une adaptation sur le chargement des couleurs. Toutefois, ce système n'aborde pas de phase de maintenance de la base de cas.

2.7. Pad'im

Le système Pad'im proposé dans [Mille, 1995] a été créé pour la conception des systèmes de supervision au sein d'une industrie. Il a pour objectif d'aider l'opérateur à choisir l'environnement correspondant à une nouvelle situation donnée (cf. Figure 2.7). Le cas dans Pad'im est décrit avec un ensemble d'objets de supervision et de connaissances mobilisées. Il est composé d'une partie problème décrivant un épisode de supervision et d'une partie solution proposant le résultat associé.

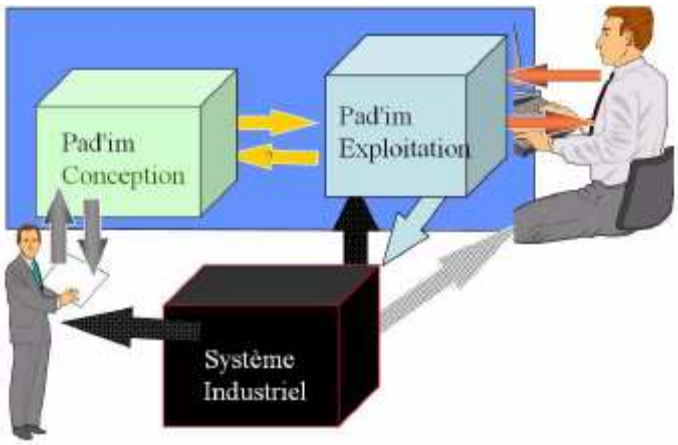


Figure 2.7. L'environnement du système Pad'im [Fuchs, 1997]

Il existe des modèles représentant des connaissances générales du système tels que les objets de supervision, les tableaux de bord et les actions de l'opérateur. Un autre modèle hiérarchique comporte les classes et les objets permettant l'héritage multiple et l'inférence de la valeur de classification. Il dispose également d'un modèle de contexte général exploité lors de la phase de remémoration. Une recherche indexée est lancée lors de la phase de remémoration. Cette recherche est faite grâce à l'algorithme des k plus proches voisins. Deux étapes sont donc enclenchées. La première consiste à la sélection d'un sous-ensemble de cas grâce au modèle de contexte. La deuxième étape s'appuie sur une mesure de similarité conceptuelle et une dissimilarité évènementielle. Ces mesures ont pour but d'évaluer la discrimination des épisodes de supervision entre les différents cas sources. Après avoir sélectionné le cas source le plus similaire, une adaptation substitutionnelle est appliquée lors de cette phase d'adaptation. Cette substitution concerne les objets semblables fournissant une information concernant la présence des objets non similaires. Concernant la maintenance de la

base de cas, nous n'avons pas trouvé d'informations sur la maintenabilité du système. Par conséquent, nous ne pouvons pas aborder cet aspect.

2.8. Nodal$_{CBR}$

Nodal$_{CBR}$ est un système à base de modèles dédié au diagnostic de pannes des alimentations d'énergie de mode de commutation (SMPS) [Cunningham, 1994]. Le système est implémenté dans un environnement hybride (KEE) de développement de systèmes experts. Ensuite, Nodal a été transformé en Nodal$_{CBR}$ en transformant une partie en un système de RàPC.

Le cas est représenté par une liste d'attributs-valeurs (cf. Figure 2.8). Il contient les tâches de diagnostic des différents dispositifs.

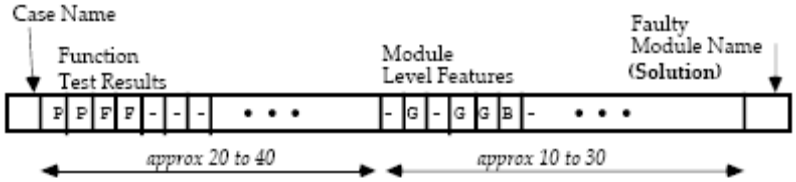


Figure 2.8. La structure typique du cas dans NodalCBR [Cunningham, 1994]

Nodal$_{CBR}$ dispose d'un modèle de classification sous forme d'arbre. C'est une représentation hiérarchique de nœuds de décision. La remémoration est basée sur une sélection de caractéristiques discriminantes. Cette sélection est issue d'un arbre de décision dans lequel les nœuds correspondent aux différents diagnostics (D) et les feuilles correspondent aux différentes classes (C). Aucune adaptation ni maintenance de la base de cas n'est exploitée dans ce système.

2.9. Patdex

Patdex est un système de diagnostic technique dédié aux machines complexes présenté dans [Richter & Wess, 1991]. Il est intégré dans le système de diagnostic MOLTKE (cf. Figure 2.9) proposant des connaissances dans les systèmes experts pour les machines à contrôle numérique CNC (Computerized Numerical Control).

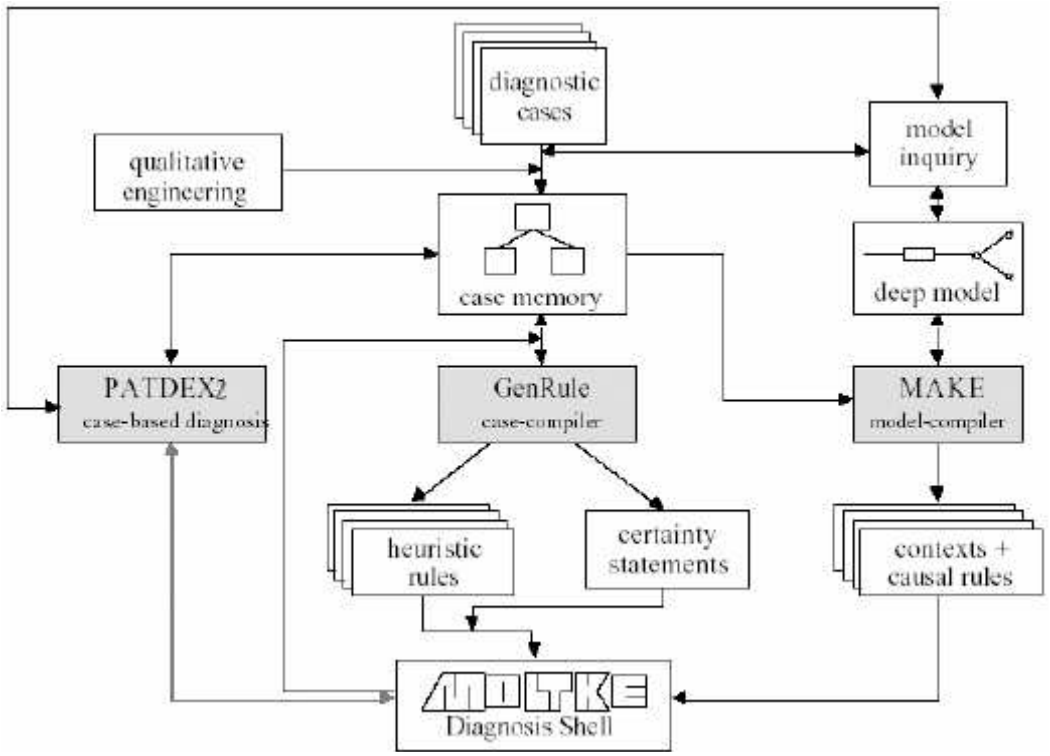


Figure 2.9. Le système Patdex intégré dans MOLTKE [Richter & Wess, 1991]

Le cas dans Patdex reflète un diagnostic représentant un protocole de classification réelle fait par le service technique. Un cas est représenté par une partie problème et une partie solution. La partie problème est caractérisée par une liste de paire (valeur, symptôme). La partie solution contient une solution empirique justifiée. Dans Patdex, le problème représente une situation notée par (Sit) et la solution représente le diagnostic notée par (Φ) et justifié par J. De ce fait, la syntaxe du cas est la suivante : (Sit, J, Φ) [Althoff, 1991].

Trois possibilités sont considérées :

- Sit n'a pas suffisamment de valeurs de symptômes pour déterminer Φ mais possède une bonne estimation ;
- Sit a déterminée Φ mais contient une information redondante ;
- Sit a déterminée Φ mais aucune situation ne pourrait le réaliser.

La recherche se fait par une exclusion progressive selon la nouvelle estimation de la similarité entre le problème et les cas sources. La mesure de similarité prend en compte une éventuelle description incomplète et incertaine de l'information du problème cible [Richter & Wess, 1991]. Il y a deux cas de figure qui peuvent se présenter à la mesure de similarité. Le premier est lorsque la solution « y_0 »ne correspond qu'à un seul problème « x_0 », alors on dit qu'il y a une certitude que $x_0 = y_0$. Le deuxième cas de figure est lorsqu'il y a plusieurs

problèmes qui peuvent être associés à la solution trouvée, alors on dit qu'il y a un degré d'incertitude entre la solution « y_0 » et les autres problèmes. Cette incertitude est donc liée à une information qui est incomplète. Elle est caractérisée par une mesure de certitude ou un facteur $\mu(x)$, exprimant le degré de certitude que l'évènement x_0 a eu lieu. $\mu(x)$ est un nombre réel appartenant à l'intervalle [0, 1]. Par ailleurs, le système Patdex offre des mécanismes de classification et d'apprentissage afin de mettre à jour d'une façon progressive les mesures de similarité utilisées.

Les modèles utilisés dans Patdex sous-jacent à la base de cas sont : un modèle profond représentant un graphe d'expérience et un modèle de contexte associé aux règles causales. La remémoration s'appuie sur l'algorithme des k plus proches voisins et sur les arbres de décision. Les graphes d'expériences sont utilisés par le système dans la phase d'adaptation en comparant les cas sources remémorés au problème rencontré. Le graphe d'expérience est un graphe de poids orienté représentant un système d'expérience [Althoff et al., 1989]. Les nœuds de ce graphe représentent des modèles d'expressions de symptôme, les poids des arcs reliant les nœuds représentent la probabilité d'un modèle (caractérisée par la fin du nœud de l'arc associé).

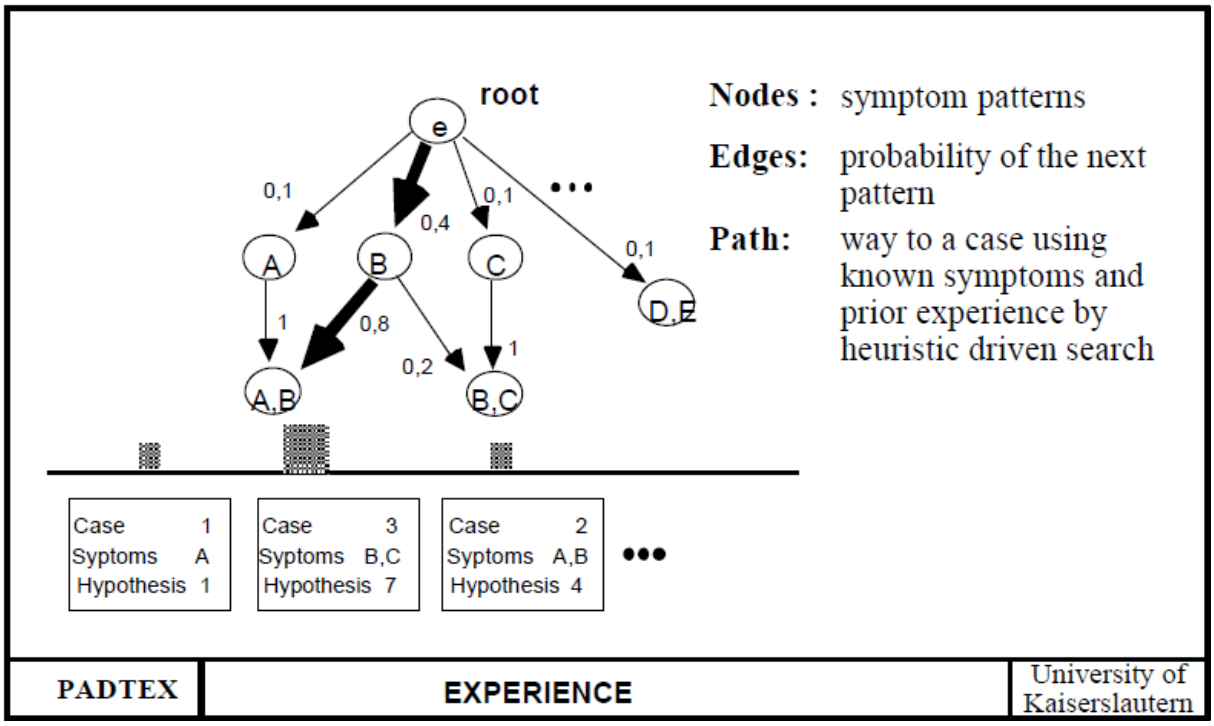


Figure 2.10. La représentation de la connaissance d'expérience dans PATDEX [Althoff et al., 1989]

Ces poids sont mis en place avec l'hypothèse qu'un autre modèle (représenté par le début du nœud de l'arc) décrit une situation courante donnée (par exemple le modèle contient exactement ces expressions de symptôme qui sont connus dans une situation donnée). Chaque modèle d'expressions de symptôme est représenté par un seul nœud. La somme des poids des arcs d'un seul nœud est égale à un (cf. Figure 2.10).

Au départ, l'initialisation de Patdex se fait avec une base de cas qui ne contient aucun cas. La première étape du système est la construction d'un graphe d'expérience contenant tous les modèles décrivant les cas. Le graphe résultant comprend également tous les nœuds représentant les intersections entre tous ces modèles d'expressions de symptôme. Les arcs qui se trouvent dans le nouveau graphe construit possèdent des poids décrivant la fréquence d'un modèle par rapport aux autres. A chaque changement de situation (apparition d'un nouveau symptôme), les informations présentes dans le réseau sont utilisées pour trouver le cas le mieux adaptable pour expliquer la nouvelle situation apparue. Cette recherche est effectuée par une heuristique conduite par le graphe. Le résultat du processus de recherche sera un modèle d'expression ainsi qu'un cas associé. Une autre utilisation de l'expérience représentée dans le graphe est de savoir quelles sont les valeurs symptôme qui peuvent être déterminées à partir d'un ensemble de valeurs connues. Cette connaissance est utilisée pour optimiser l'ordre des tests dans une situation donnée.

De plus, il y a un lien fort entre les résultats de l'expérience du système et ceux obtenus par la mesure de similarité. En effet, les valeurs associées aux cas, en utilisant la mesure de similarité, induisent un ordre donné des cas connus. S'il n'y a aucun cas pour lequel sa valeur de similarité dépasse un certain seuil fixé, alors il y aura un échec d'identification de la défaillance et le système s'arrête sans avoir trouvé un résultat de diagnostic. Toutefois, lorsqu'il y a des cas pour lesquels leur valeur de similarité est supérieure au seuil déterminé, alors ils vont être comparés aux cas proposés par le graphe d'expérience du système. En effet, il y aura une combinaison entre les cas remémorés, suite à la mesure de similarité, et le cas issu du graphe d'expérience, comme le montre la Figure 2.11.

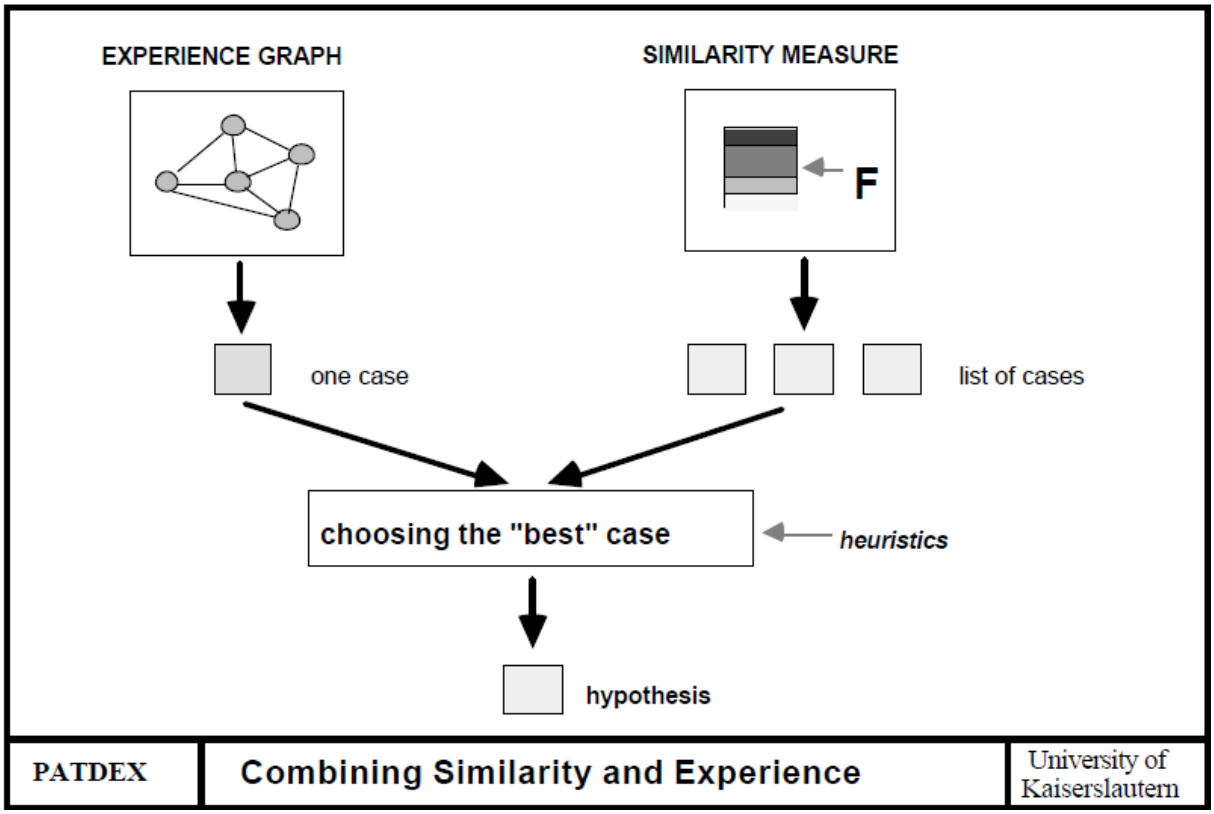


Figure 2.11. Synthèse entre la mesure de similarité et le réseau d'expérience [Althoff et al., 1989]

Ainsi, « le cas le plus approprié » sera choisi parmi les cas qui ont les meilleurs résultats obtenus. Ceci est fait en exécutant une heuristique de résolution de conflits, en tenant compte des facteurs suivants :

- distance entre la valeur de la similarité et la borne supérieure ;
- les coûts pour trouver les valeurs nécessaires pour compléter le cas ;
- les conséquences d'un mauvais diagnostic basé sur un cas ;
- la fréquence d'un cas ;
- nombre de fois où le cas a été choisi comme hypothèse.

Enfin, la solution du cas sélectionné est proposée à l'utilisateur autant qu'un diagnostic. L'utilisateur peut ainsi rejeter, accepter ou modifier cette solution.

Par ailleurs, Patdex n'utilise pas de méthode pour maintenir sa base de cas.

2.10. Tableau de comparaison et bilan des systèmes étudiés

Cette section récapitule les caractéristiques des huit systèmes de diagnostic industriel des machines complexes dans deux tableaux.

Système de RàPC	Domaine d'application	Type de système de RàPC	Outil de Dévelop.	Modèle de connaissance	Structure de la base de cas
Gas Turbine	Turbines à gaz dans General Electric Energy	Connaissance	TrollCreek	Modèle probabiliste par réseaux bayésiens	Hiérarchique
Creek	Voitures	Connaissance	/	Modèle général du domaine	Plate
Cassiopee	Moteurs d'avions Boeing 737	Connaissance	Kate (Kaidara)	Conversationnel + poids	Plate
IRACUS	Locomotives	Mining	/	Pas de modèle	Hiérarchique
FormTool	Plastic (couleurs)	Mining	/	Pas de modèle	Plate
Pad'im	Supervision industrielle	Connaissance	/	Objets de supervision, Modèle hiérarchique, Modèle de contexte général	Hiérarchique
$Nodal_{CBR}$	Energie de mode de commutation (SMPS)	Connaissance	KEE	Modèle de classification sous forme d'arbre	Plate
Patdex	Machines complexes	Connaissance	/	Graphe d'expérience + un modèle de contexte (règles causales)	Plate

Tableau 2.1. Tableau récapitulatif des principaux systèmes de diagnostic industriel des machines complexes par rapport à la nature de l'application et l'environnement de connaissance

Tout d'abord, nous pouvons constater qu'il n'y a pas de méthode commune permettant la construction d'un système de RàPC. Cette construction dépend essentiellement de la représentation du cas et des modèles de connaissance du domaine d'application. Ainsi, les différentes phases du cycle dépendent de la modélisation des connaissances. En effet, les auteurs nous fournissent directement le système avec la représentation du cas associée sans aucune méthodologie.

Le Tableau 2.1 décrit la nature de l'application ainsi que le type de connaissance manipulée selon cinq caractéristiques : le domaine d'application, les outils de développement,

le modèle de connaissance, la structure de la base de cas et le type de système de RàPC appliqué. A partir de ce tableau, nous pouvons constater qu'il y a deux types de systèmes : les systèmes orientés *mining* et les systèmes orientés *connaissance*.

Les systèmes orientés connaissance tels que Gas Turbine, Creek, Cassiopee, Pad'im, $Nodal_{CBR}$, Patdex, utilisent des modèles de connaissance différents les uns des autres. Contrairement aux systèmes orientés mining tels que IRACUS et FormTool qui n'en utilisent pas.

Concernant les outils de développement, la majorité des systèmes n'en possède pas. Toutefois, le système Gas Turbine est implémenté grâce à un outil de développement propre à GE nommé TrollCreek. KEE est un outil de développement système expert hybride implémentant le système $Nodal_{CBR}$. Enfin, Cassiopee est construit à l'aide d'un outil de développement de RàPC commercial Kate. Creek, Patdex et Pad'il sont des systèmes académiques et sont donc créés à l'aide des systèmes de représentation des connaissances. Quant à la structure de la base de cas, elle est pour la plupart des systèmes de nature plate, sauf pour Gas Turbine, IRACUS et Pad'im dans lesquels elle est hiérarchique.

Système de RàPC	**Représentation du cas**	**Remémoration**	**Adaptation**	**Maintenance de la base de cas**
Gas Turbine	Nœuds contenant l'état des composants	Calcul probabiliste des valeurs de signaux discrétisés.	Pas d'adaptation	Pas de maintenance
Creek	Ensemble de concepts formant une liste de descripteurs	Orientée adaptabilité, mesure de similarité sémantique	Trace de raisonnement + généralisation d'attributs	Connaissances dans la base
Cassiopee	Liste des descripteurs hiérarchisés	Similarité locale (les attributs du cas) + similarité globale (les classes d'objets)	Transformatio-nnelle	Raffinement de la base de cas
IRACUS	Liste des descripteurs	Algorithme des KPVV + mesure de similarité pondérée en fonction du code de réparation	Pas d'adaptation	Pas de maintenance

FormTool	Liste des descripteurs	Orientée adaptabilité, algorithme des KPVV + similarité par terme linguistique floue	Fonction spécifique relative aux caractéristiques des couleurs	Pas de maintenance
Pad'im	Ensemble d'objets de supervision et de connaissances	Algorithme des KPVV + mesure de similarité conceptuelle	Substitutionnelle	Pas de maintenance
Nodal$_{CBR}$	Liste des descripteurs	Sélection de caractéristiques discriminantes à partir d'un arbre de décision	Pas d'adaptation	Pas de maintenance
Patdex	Liste des descripteurs	Algorithme des KPVV + arbres de décision Mise à jour des mesures de similarité par apprentissage	Graphes d'expériences	Dans MOLTKE

Tableau 2.2. Tableau récapitulatif des principaux systèmes de diagnostic industriel des machines complexes par rapport aux phases du cycle de RàPC

On s'intéresse dans le Tableau 2.2 à la représentation du cas ainsi qu'aux trois phases du cycle de RàPC à savoir : la remémoration, l'adaptation et la maintenance de la base de cas.

La représentation du cas la plus utilisée est une liste de descripteurs représentée par des attributs-valeurs. Ces descripteurs peuvent être de différents types : symboliques, numériques, taxonomiques, intervalles, etc. Concernant les systèmes orientés mining possèdent une représentation simple. En revanche, les systèmes orientés connaissance, disposent de diverses natures de représentation allant des représentations complexes telles que les nœuds contenant l'état des composants pour « Gas Turbine », les ensembles de concepts pour « Creek » et la liste des descripteurs hiérarchisés pour « Cassiopee » jusqu'à la simple représentation sous forme de liste de descripteurs.

Dans la phase de remémoration, l'algorithme des k plus proches voisins est le plus utilisé (ICARUS, FormTool, Pad'im et Padtex). Il est associé à des mesures de similarité spécifiques au domaine d'application et à des méthodes inductives exploitant la base de cas. Le système orienté mining « FormTool » utilise une mesure de similarité orientée adaptabilité. Cependant, elle n'est pas assez développée et concerne le choix des couleurs de plastique suivant ses caractéristiques. Quant aux systèmes orientés connaissance, seul le système « Creek » emploie une remémoration guidée par l'adaptation. Toutefois, ce dernier ne détaille pas la façon de procéder au cours de cette remémoration guidée.

En ce qui concerne la phase d'adaptation, elle se révèle être une phase sensible car soit les systèmes ne présentent pas d'adaptation (Gas Turbine, IRACUS et $\text{Nodal}_{\text{CBR}}$) soit les systèmes proposent de simples règles d'adaptation (FormTool) ou des graphes d'expériences (Padtex) ou encore utilisent une structure hiérarchique de la base de cas afin de réaliser une adaptation substitutionnelle au niveau des descripteurs de cas (Pad'im). Le système Creek combine l'adaptation générative, en se basant sur la structure hiérarchique, avec la trace de raisonnement. Cette technique essaie de palier au manque d'une éventuelle évolution du système en gardant une trace du raisonnement.

Cependant, l'exploitation de la relation entre les phases de remémoration et d'adaptation est peu utilisée. En effet, contrairement aux systèmes orientés mining, les systèmes orientés connaissance approfondissent le développement de ces phases mais il n'y a que le système « Creek » qui propose une liaison entre ces deux phases mais sans vraiment l'expliquer. Par conséquent, nous accordons une attention particulière au lien entre la phase de remémoration et la phase d'adaptation à travers un état de l'art que nous aborderons au chapitre 4. En effet, le résultat de l'une des phases peut influencer les décisions élaborées par l'autre.

Quant à la phase de maintenance de la base de cas, nous constatons que la plupart des systèmes n'en disposent pas, à savoir cinq systèmes sur huit. Cette phase est mise en place quand le système de diagnostic est en fonctionnement depuis un certain temps. Les systèmes exploitant cette phase, notamment Creek et Cassiopee, restent au niveau de la connaissance ou du raffinement de la base de cas sinon, ils impliquent un module extérieur du système tel que Moltke pour le système Cassiopee. Les systèmes orientés mining (IRACUS et FormTool)

n'en possèdent pas. Dans notre étude, nous nous intéressons à l'étude de la maintenance de la base de cas pour les systèmes orientés mining au chapitre 3.

Cette étude comparative nous permet de situer notre travail et de faire des choix quant à la création d'un système de diagnostic par RàPC. Nous développons deux types de systèmes d'aide au diagnostic par RàPC qui seront dédiés à un système industriel supervisé de transfert de palettes SISTRE (Supervised Industrial System of pallets TransFer). Un premier système orienté mining pour nous affranchir des problèmes de modélisation des connaissances. Un deuxième système orienté connaissance pour réduire la taille de la base de cas et proposer une phase d'adaptation qui fait la spécification des systèmes de RàPC d'après Lieber [2007].

Cependant, nous n'exploitons pas d'outils de développement dans notre étude. Quant à la base de cas, nous utilisons une structure plate pour les deux types de systèmes.

La représentation du cas est sous forme de liste d'attributs-valeurs pour les deux types de système. Les valeurs sont de type modal et numérique. Le cas représente l'expertise du domaine qui est basée sur l'utilisation des outils de sûreté de fonctionnement[6]. Nous traduisons sous forme de cas l'analyse dysfonctionnelle de l'équipement issu d'un retour d'expérience terrain. Nous associons au cas, pour le deuxième système, des modèles de connaissance.

Dans la phase de remémoration, nous exploiterons pour les deux systèmes l'algorithme des k plus proches voisins. Dans le deuxième type de système, une mesure d'adaptation permettra de sélectionner les candidats à la phase d'adaptation.

Toutefois, le premier type de système peut arriver facilement à une explosion combinatoire de cas. On ne s'intéressera donc pas à la phase d'adaptation mais à la maintenance de ce système en l'occurrence la phase de maintenance de la base de cas.

Nous pouvons ainsi résumer nos choix et nos démarches comme suit :

- nous développons un système orienté mining en proposant une représentation du cas et une phase de remémoration. La phase de maintenance de la base de cas étant importante, nous aborderons un état de l'art ainsi que notre contribution au chapitre 3 ;

[6] La sûreté de fonctionnement est « un ensemble des propriétés qui décrivent la disponibilité et les facteurs qui la conditionnent : fiabilité, maintenabilité, et logistique de maintenance » [Afnor, 2001]

- nous développons un système orienté connaissance pour lequel nous proposons une représentation spécifique de cas de la base de cas à laquelle nous lui associons deux modèles de connaissance. Quant aux phases de remémoration et d'adaptation, nous présentrons un état de l'art ainsi qu'une proposition d'une phase de remémoration guidée par l'adaptation et un algorithme d'adaptation au chapitre 4.

3. Nos choix et nos démarches

Nos choix et nos démarches dépendent du type de système à étudier. Concernant les systèmes orientés mining, nous traitons la phase de maintenance de la base de cas afin de pallier l'explosion combinatoire de cas. Quant aux systèmes orientés connaissance, nous nous intéressons aux deux phases de remémoration et d'adaptation ainsi que le lien entre elles.

De ce fait, nous commençons par décrire les deux types de systèmes avant d'aborder la problématique concernant chacun d'eux. Nous partons d'une description générale d'un modèle de système de RàPC que nous déclinons suivant le type de système traité. En effet, nous nous inspirons du modèle initié par Lamontagne et Lapalme [2002] abordé au chapitre 1. Nous exploitons d'une part, dans le processus off-line, les ressources du domaine, les containers de connaissance ainsi que la maintenance de la base de cas et d'autre part, dans le processus on-line, le cycle du RàPC, comportant les phases de recherche et d'adaptation des cas retenus, auquel nous lui rajoutons les phases « d'élaboration du cas cible » ainsi que la phase de maintenance de la base de cas. Nous éclatons cette dernière phase en deux phases à savoir : une phase de structuration de la base de cas dans le processus off-line et une phase de son auto-incrémentation dans le processus on-line (cf. Figure 2.12).

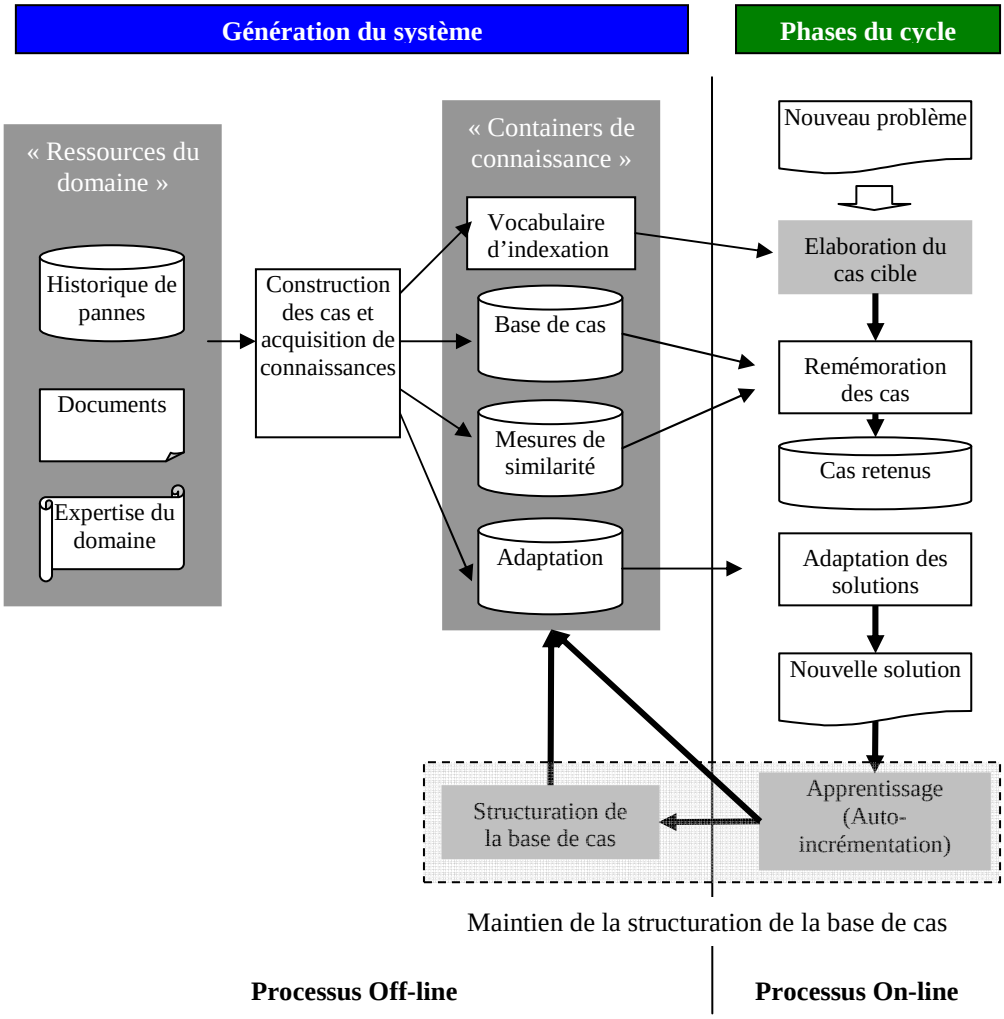


Figure 2.12. Modèle de base d'un système de diagnostic par RàPC

Ce modèle de base met en évidence les quatre containers de connaissance qui sont créés dans le processus off-line ainsi que le cycle du RàPC, exploité dans le processus on-line. Ce dernier processus débute à partir de la génération d'un nouveau problème jusqu'à la nouvelle solution donnée au problème qui est prête à être stockée dans la base de cas si besoin est.

Nous décrivons maintenant les deux types de systèmes dans ce qui suit.

3.1. Systèmes orientés extraction (mining)

Dans la plupart des systèmes de RàPC étudiés, un cas reflète et traduit l'expérience de diagnostic sans pour autant faire référence à un modèle. Nous proposons une représentation complète et détaillée du cas. Cette représentation est simple à mettre en œuvre, elle est issue d'une analyse de l'historique des pannes, sur les arbres de défaillances de l'équipement et ses composants ainsi que sur l'AMDEC (Analyse des Modes de Défaillance, de leurs Effets et de

leur Criticité : méthode d'analyse préventive de la fiabilité). Le cas est formalisé sous forme d'une liste d'attributs-valeurs. Cette liste permet d'une part, de refléter l'ensemble des composants associés à chaque zone géographique, leur valeur et les états de ces composants et d'autre part, de représenter l'évaluation et la solution de la situation diagnostiquée.

En se référant au modèle de base présenté sur la Figure 2.12 sur lequel nous prenons appui pour la description du système orienté mining, nous commençons par le processus offline qui contient les ressources du domaine et les containers de connaissance, ensuite nous passons au processus on-line contenant le cycle de RàPC.

3.1.1. Ressources du domaine

Cette partie contient toutes les ressources permettant la construction du cas et des modèles fonctionnels. Ces ressources concernent l'historique de pannes, les documents de différentes natures ainsi que l'expertise du domaine.

L'historique de pannes nous permet de collecter les informations nécessaires à la construction du cas. Cet historique contient toutes les défaillances produites dans le système à diagnostiquer et les solutions adoptées. Les attributs du cas sont élaborés à partir de ces données.

Les documents sont de différentes natures (documents techniques du système à diagnostiquer ou des documents de fonctionnement, etc.) et sont utilisés pour compléter les informations manquantes dans le cas.

Dans ce type de système, l'expertise du domaine concerne l'expert humain. Il intervient lors d'une opération de dépannage en exploitant ces connaissances et ses expériences.

Le cas est une liste d'attributs-valeurs de taille variable. Les valeurs des attributs peuvent être des valeurs numériques, des valeurs modales ou pas renseignées. La structure du cas est présentée à la Figure 2.13.

Partie problème					Partie solution
Localisation	Etat des composants				Classe
$ds_1 \ldots ds_k$	ds_{k+1}	ds_{k+2}	…	ds_m	Ds_1

Figure 2.13. Structure d'un cas de classification

La partie problème se compose de la localisation de la défaillance et de l'état des composants dans cette zone de défaillance.

La partie solution contient la classe de défaillance correspondant aux états des composants associés. De plus, les valeurs des attributs que nous mettons en place dépendent les uns des autres suivant le contexte d'utilisation.

Le type de système que nous traitons est un système de classification. De ce fait, notre but est de trouver la classe de défaillance adéquate suivant la situation de diagnostic.

Nous avons essayé plusieurs types de représentation en commençant par une formalisation qui tient compte, dans ces descripteurs, de l'ensemble des composants du système à diagnostiquer. Cela peut provoquer une explosion combinatoire de descripteurs qui se répercute sur l'explosion du nombre de cas dans la base de cas. Nous avons pu le constater sur l'équipement industriel SISTRE (Supervised Industrial System of pallets TransFer) que nous présenterons au chapitre 5.

Nous prenons un exemple d'une station SISTRE qui est composée de : neuf descripteurs (D1.. D9), cinq stoppeurs (S1.. S5), trois actionneurs pneumatiques (pousseur, tireur, indexeur), deux balogh de lecture/écriture magnétique (Bal0, Bal1) et d'un robot.

De ce fait, la première formalisation du cas pourrait être comme le montre la Figure 2.14.

Cas mining (SISTRE)																											
Problème																											Solution
Localisation					Etat des composants																						
ds1	ds2	ds3	ds4	ds5	ds6	ds7	ds8	ds9	ds10	ds11	ds12	ds13	ds14	ds15	ds16	ds17	ds18	ds19	ds20	ds21	ds22	ds23	ds24	ds25	ds26	Ds1
Station	Zone	Sous-zone	Composant - équipement	Présente palette	Détecteur									Balogh		Stoppeur						Indexeur	Pousseur	Tireur	Robot	Classe
					D1	D2	D3	D4	D5	D6	D7	D8	D9	Bal0	Bal1	S1	S2	S3	S4	S5	S6	I	P	T	R	
Station 3	Anneau principal	Sortie	Tireur	Présente	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	1	0	0	1	0	Détecteur

Figure 2.14. Exemple d'une première formalisation d'un cas de classification sur l'équipement SISTRE

Ce cas contient ***26 descripteurs de problème*** et ***1 descripteur de solution***. Cette formalisation a conduit à une base de cas contenant ***11600 cas***.

Ensuite, nous avons regroupé l'ensemble des composants assurant les mêmes fonctions en deux descripteurs. Le premier a pour rôle de représenter l'ensemble des composants du même type. Le deuxième représente l'état de ces composants. De ce fait, un cas peut être représenté par un nombre réduit de descripteurs suivant l'état de défaillance des composants.

Nous reprenons le même exemple et nous appliquons le principe du deuxième type de formalisation. Nous obtenons un cas avec un nombre de descripteur réduit comme le montre l'exemple sur la Figure 2.15.

Cas mining (SISTRE) "deuxième formalisation"											
Problème											Solution
Localisation					Etat des composants						
ds1	ds2	ds3	ds4	ds5	ds6	ds7	ds8	ds9	ds10	ds11	Ds1
Station	Zone	Sous-zone	Composant-équipement	Présence Palette	Type Détecteur principal	Etat Détecteur principal	Stoppeur	Etat du stoppeur	Type Détecteur du contexte	Etat Détecteur du contexte	Classe
Station 3	Anneau principal	Sortie	Tireur	Présente	D9	0	S6	1	D8	1	Détecteur

Figure 2.15. Exemple d'une deuxième formalisation d'un cas de classification sur l'équipement SISTRE

Ce cas contient ***11 descripteurs de problème*** et ***1 descripteur de solution***. Cette formalisation a conduit à une base de cas contenant ***750 cas***. De ce fait, il y a eu une réduction de ***55,55%*** par rapport au nombre de descripteurs et de ***93,53%*** par rapport au nombre de cas dans la base de cas.

Par conséquent, nous adoptons la deuxième présentation du cas que nous déploierons dans la suite de notre étude.

3.1.2. Containers de connaissance

Les containers de connaissance sont liés à la construction du cas et l'acquisition de connaissances. Par ailleurs, le choix de la formalisation du cas a une influence sur les containers des connaissances.

Le container de vocabulaire d'indexation est constitué d'un ensemble d'attributs caractérisant la description de la partie problème et de la partie solution du système à diagnostiquer. Les attributs du cas forment ainsi la structure globale de la base de cas et jouent un rôle important dans la phase de remémoration.

Le container des mesures de similarité a pour fonction d'évaluer la similarité entre le cas cible et les cas sources. Nous définissons une mesure qui tient compte des valeurs renseignées ou non renseignées des attributs. Concernant les valeurs renseignées, nous tenons compte des deux types des valeurs d'attributs à savoir : numérique et modale. Ce container est exploité au cours de la phase de recherche du cycle de RàPC.

Le container des connaissances d'adaptation n'est pas exploité dans ce type de système.

Le container de la base de cas contient l'ensemble des expériences sous forme de cas qui sont formalisés par l'ensemble des attributs. Ce container est sollicité par les phases de remémoration, d'adaptation et de maintenance de la base de cas. La base de cas reflète l'analyse dysfonctionnelle du système à diagnostiquer.

3.1.3. Cycle de RàPC

Le cycle du RàPC se place dans le processus on-line et il est enclenché dès l'apparition d'une nouvelle défaillance. Cette nouvelle défaillance (ou nouveau problème) est mise sous forme de cas grâce à l'étape d'élaboration du cas cible.

Elaboration

Cette phase met à disposition un formulaire composé de questions issues d'une structure arborescente concernant les zones géographiques de défaillance. Ce formulaire est proposé aux acteurs et permet de cerner le contexte en localisant la panne. Ceci permet de renseigner la partie « localisation ». Ensuite, la partie localisée comporte un certain nombre de composants associés au contexte. Ainsi, les types de composants et leur état renseignent la deuxième partie du problème : « état des composants ». De ce fait, nous aurons le cas cible élaboré qui nous permet de passer à la phase de remémoration des cas sources les plus similaires à ce cas cible.

Remémoration

Cette phase utilise le container de mesures de similarité et l'algorithme des k plus proches voisins pour la recherche des cas similaires. Pour ce type de système, nous utilisons une mesure similarité qui tient compte de la présence des valeurs des attributs, de leurs types et de leurs valeurs. La mesure de similarité appliquée est donnée comme suit :

$$Sim(Source, Cible) = \frac{\sum_{i=1}^{m} \varphi_i^{\text{Pr}ésence} \times \varphi(s_i, c)}{\sum_{i=1}^{m} \varphi_i^{\text{Pr}ésence}} \quad (1)$$

Où :

m : est le nombre d'attributs

$\varphi_i^{\text{Pr}ésence}$: reflète la présence de la valeur du descripteur *i*.

$\begin{cases} \varphi_i^{\text{Pr}ésence} = 1, \text{ le descripteur est renseigné dans le cas source } et \text{ dans le cas cible} \\ \varphi_i^{\text{Pr}ésence} = 0, \text{ le descripteur n'est pas renseigné dans le cas source ou dans le cas cible} \end{cases}$

$\varphi(s_i, c_i)$: est la similarité locale des valeurs de deux descripteurs du cas source et du cas cible, calculée pour le descripteur *i*.

Pour les valeurs numériques : $\varphi(s_i, c_i) = \frac{1 - |a - b|}{range}$

Où :

a et b : sont les valeurs des descripteurs de si et ci respectivement.

range : est la valeur absolue de la différence entre la borne supérieure et la borne inférieure de l'ensemble des valeurs.

Pour les valeurs modales : $\varphi(s_i, c_i) = \begin{cases} 1 \text{ pour a = b} \\ 0 \text{ pour a} \neq \text{b} \end{cases}$

Un algorithme des K plus proches voisins (Kppv) est associé à cette mesure de similarité pour rechercher les cas sources les plus similaires au cas cible.

Ce type de système, qui est un système de classification, n'utilise pas de phase d'adaptation. Nous affectons tout simplement la classe du cas source le plus proche au cas cible sans avoir recours à une phase d'adaptation. Dans le cas où il y a plusieurs cas source qui ont le même taux de similarité par rapport au cas cible, nous choisissons la classe dominante (c-à-d la classe la plus représentée dans les cas sources sélectionnés). Ceci dépend également de la valeur de « *K* » choisie de l'algorithme Kppv.

Maintenance de la base de cas

La phase de maintenance de la base de cas est une étape importante pour son bon fonctionnement tout au long du cycle de vie du système de diagnostic par RàPC. En effet, la base de cas s'enrichit au fur et à mesure par l'ajout successif de cas, ce qui amène d'une part, à une explosion du nombre de cas dans la base de cas et d'autre part, à avoir des solutions qui peuvent être contradictoires. En effet, l'explosion du nombre de cas se répercute sur le temps de réponse du système dans sa phase de recherche et d'adaptation. De plus, si nous introduisons n'importe quel cas dans cette base de cas nous pouvons obtenir des mauvaises solutions données au problème rencontré. Par conséquent, nous consacrons une étude sur la maintenance de la base de cas et sur son auto-incrémentation au chapitre 3.

Les systèmes de type orienté *mining* sont des systèmes qui disposent de phases de remémoration et d'adaptation simples à mettre en œuvre. Ce sont des systèmes de classification qui ont pour objectif de déterminer la classe de défaillance du problème rencontré.

Les systèmes orientés connaissance sont des systèmes plus sophistiqués. Ils disposent d'une modélisation des connaissances du domaine plus importante et un développement des phases du cycle plus approfondies notamment les phases de remémoration et d'adaptation.

3.2. Systèmes orientés connaissances (knowledge)

Ce type de système nécessite généralement une représentation complexe du cas, d'une modélisation des connaissances du domaine et d'une phase d'adaptation bien développée. Le système à étudier, notamment SISTRE, dispose des composants pouvant être caractérisés par leur valeur, leur état et leur mode de fonctionnement (normal, anormal) ainsi que le contexte dans lequel les composants se trouvent grâce à l'étude de l'AMDEC, de l'historique de pannes et des arbres de défaillances.

Nous prenons appui sur le modèle de base d'un système de RàPC (cf. Figure 2.12). Nous associons à ce modèle de base les modèles de connaissance. Ces modèles sont liés à l'expertise qui reflète l'analyse dysfonctionnelle des composants de l'équipement sous forme de base de cas. La création des modèles de connaissances est réalisée grâce à l'analyse de l'équipement et sa décomposition. De plus, nous mettons en place un cas adapté au diagnostic pour la création de notre système. Nous présentons les modèles de connaissance sous-jacents à la base de cas du système à diagnostiquer ainsi que la manière de construction du cas prenant appui sur la définition de référence du diagnostic.

3.2.1. Ressources du domaine

Dans les systèmes orientés connaissance, l'expertise terrain est exploitée grâce aux outils de sûreté de fonctionnement tels que l'AMDEC, l'historique de pannes et les arbres de défaillances. Les ressources du domaine englobent tous ces outils en plus de la documentation du système à diagnostiquer telles que la documentation technique, la documentation liée aux outils nécessaires au fonctionnement du système, la documentation concernant les pièces de rechange, etc.

De ce fait, nous pallions le manque d'expertise en exploitant tous ces outils d'expertise pour construire un cas de diagnostic.

3.2.2. Containers de connaissance

La mise en œuvre d'un système de RàPC passe tout d'abord par l'acquisition et la représentation des données et des connaissances en présence lors de la construction du cas.

Cette partie concerne la mise en place des modèles de connaissance du système de RàPC.

Modèles de connaissances du système de RàPC

La représentation des connaissances est élaborée à partir de deux modèles : le modèle de contexte et le modèle de taxonomie des composants (cf. Figure 2.16). Le modèle de contexte est issu de l'analyse fonctionnelle de l'équipement du système étudié. La taxonomie des composants classifie des sous-ensembles d'équipements par rapport à leur fonction. Chaque ensemble de composants est regroupé selon la similitude de fonctionnement ou assurant les mêmes fonctions. Les deux modèles sont reliés à la base de cas qui reflète la partie dysfonctionnelle des composants. Nous associons à ces deux modèles et à la base de cas un ensemble de règles de décisions permettant de définir le mode de fonctionnement de l'équipement [Haouchine et al., 2007b].

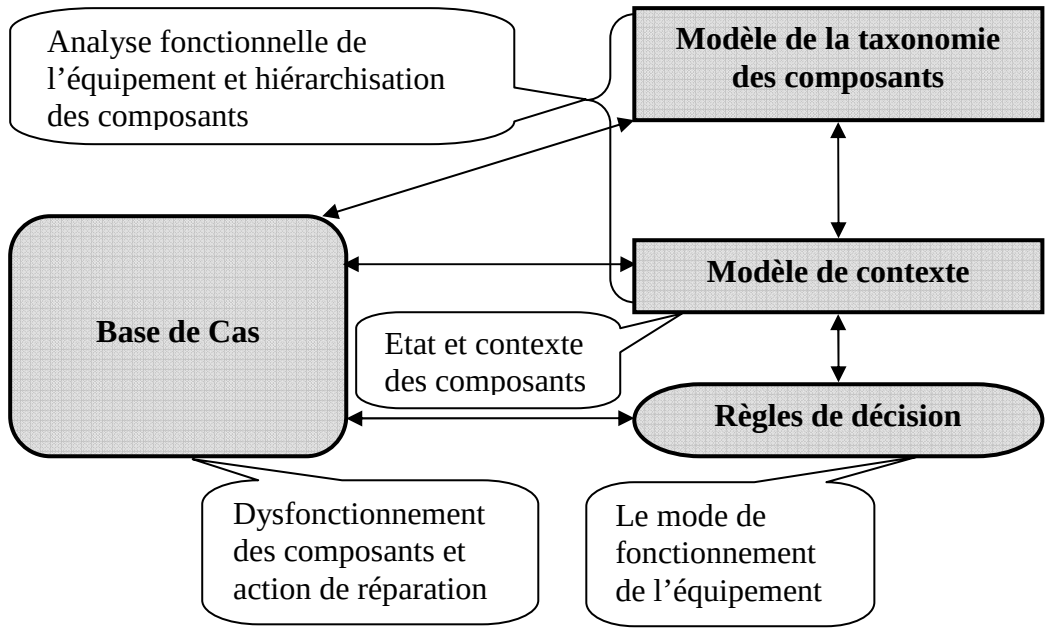


Figure 2.16. Schéma représentant le modèle de gestion des connaissances d'un système de RàPC orienté connaissance dédié au diagnostic technique

Le container de la base de cas est constitué de l'ensemble des cas de diagnostic.

Construction du cas de la base de cas

Pour construire un cas de la base de cas, nous prenons appui sur la définition de référence du diagnostic : « *ce sont les actions menées pour la* ***détection*** *de la panne, sa* ***localisation*** *et* ***l'identification*** *de la cause* [Afnor, 2001] ». Le diagnostic est composé de trois parties : la partie de détection de la panne, la partie concernant la localisation de la zone dans laquelle la panne s'est produite et la partie d'identification du composant défaillant.

La conception du cas prend en compte ces trois parties et s'appuie sur les modèles de connaissances de l'équipement à diagnostiquer abordés précédemment.

Les trois parties du cas sont :

Détection : nous déterminons sur l'équipement la classe de défaillance en prenant appui sur l'AMDEC. Cette classe fait partie de la solution de notre cas.

Localisation : cette partie est décrite dans la première partie du problème du cas. Elle permet de déterminer grâce à un modèle de contexte la zone de panne de l'équipement.

Identification : cette identification concerne d'une part, la deuxième partie problème du cas que l'on nomme « *partie fonctionnelle* » et d'autre part, la partie solution en permettant d'identifier le composant défaillant. La partie fonctionnelle s'appuie sur un modèle hiérarchique des composants dans lequel les classes de ce modèle se retrouvent dans les descripteurs fonctionnels afin d'obtenir un cas générique.

Nous représentons le problème de diagnostic technique de manière la plus générale possible. Un cas a une représentation objet et permet de définir une hiérarchie de descripteurs contenant aussi bien les descripteurs de problèmes que les descripteurs de solution.

Les descripteurs sont représentés par trois attributs. Nous associons à chaque attribut des valeurs modales formant une partition attenante à l'attribut considéré. Une des spécificités de notre cas d'étude est de déterminer un état et un mode de fonctionnement, à chaque composant à diagnostiquer [Haouchine et al., 2008c]. Nous affectons par conséquent à un descripteur donné :

- Un attribut relatif à sa valeur proprement dite (le descripteur est un composant électrique ou un composant mécanique par exemple) ds_i^{Valeur} ;
- Un attribut relatif à l'état de celui-ci ds_i^{Etat} ;

- Ainsi qu'un attribut relatif au mode de fonctionnement qui reflétera l'état normal et/ou anormal des composants de l'équipement $ds_i^{M.F}$.

Les descripteurs de problème de la partie fonctionnelle auront par conséquent trois attributs relatifs à la valeur du composant, son état et son mode de fonctionnement : ds_i = (ds_i^{Valeur} , ds_i^{Etat} , $ds_i^{M.F}$). En tenant compte de toutes ces spécificités, nous pouvons schématiser la structure du cas comme le montre la Figure 2.17.

Partie problème								Partie solution		
Localisation	Fonctionnelle							Classe	Composant défaillant	..
$ds_1 \ldots ds_l$	ds_{l+1}^{valeur}	ds_{l+1}^{Etat}	$ds_{l+1}^{M.F}$	…	ds_m^{valeur}	ds_m^{Etat}	$ds_m^{M.F}$	Ds_1	Ds_2	..

Figure 2.17. Structure du cas de diagnostic

Nous identifions les descripteurs pertinents pour la représentation des cas dans le domaine étudié et définissons une structure appropriée des cas associée aux modèles de connaissances générales. Cette identification nous permet la mise en place de la phase de remémoration, d'adaptation et l'exploitation de la mise en relation entre ces deux phases à travers la création des mesures appropriées.

L'étude de la gestion des connaissances et de la formalisation du cas est appliquée sur un moteur à explosion 1.5 dCi K9K 105ch de la société *Renault* disponible à l'adresse suivante : http://v3.renault.com/cfm/module-K9K/fr/index.html.

Ce modèle de moteur dispose d'un système Common Rail de 2ème génération, turbo à géométrie variable multi-ailettes, pression d'injection de 1600 bars, des injecteurs 6 trous à commande piézo-électriques, des pistons géométriquement modifiés et d'un double volant amortisseur (DVA) qui supprime les effets de bourdonnement à bas régime.

Une étude fonctionnelle et dysfonctionnelle de ce moteur a été faite dans l'optique de la mise en place complète de notre méthode et la vérification de la faisabilité des résultats obtenus.

Dans un premier temps, nous avons fait appel à un mécanicien professionnel afin de comprendre le fonctionnement du moteur, les relations de cause-à-effet entre les différents composants et la réalisation d'un arbre de défaillance. Ensuite, une base de cas ainsi que des modèles de connaissance validés par ce mécanicien ont été mis en place.

De ce fait, grâce à l'étude fonctionnelle et dysfonctionnelle du moteur, nous avons construit une base de cas contenant 20 cas (Figure 2.18 et Figure 2.19). Dans sa partie problème, un cas est composé de 12 descripteurs. Les quatre premiers descripteurs déterminent ***la zone de défaillance de l'équipement « moteur »***. Cette zone est déterminée par « ds_1 : l'état du moteur », « ds_2 : Etat des bougies de préchauffage », « ds_3 : température du moteur » et « ds_4 : sous zone de défaillance ». Les valeurs que peuvent prendre ces quatre descripteurs sont :

- Etat du moteur ∈ [marche, arrêt] ;
- Etat des bougies de préchauffage ∈ [bon, mauvais] ;
- Température du moteur ∈ [basse, moyenne, haute] ;
- Sous zone de défaillance ∈ [partie haute, partie basse, partie bruitée, partie silencieuse].

Les 8 descripteurs restants : [ds_5,…, ds_{12}], déterminent ***la partie fonctionnelle des composants appartenant à la zone de défaillance de l'équipement « moteur »***. Les descripteurs de cette partie représentent les classes de fonctionnement des différents composants de l'équipement qui se trouvent dans la hiérarchie des composants.

Concernant la partie solution du cas, nous trouvons la classe de défaillance du composant concerné par les symptômes décrits dans la partie problème, bien évidemment le composant défaillant, les actions à mener pour remédier à cette panne (ou défaillance) et l'endroit dans lequel il faut intervenir. Par conséquent, cette partie solution est composée de 4 descripteurs, à savoir :

- *Ds_1 : Classe de défaillance*, ce descripteur correspond aux classes de fonctionnement correspondant à la taxonomie des composants ;
- *Ds_2 : Identification du composant défaillant*, ce descripteur détermine l'origine de la panne en précisant de quel composant il s'agit ;
- *Ds_3 : Action de réparation associée*, ce descripteur comprend les actions et les stratégies de maintenance à appliquer au composant défaillant ;
- *Ds_4 : Zone de défaillance*, ce descripteur détermine le lieu dans lequel la défaillance s'est produite.

	Problème																												
	localisation				Partie fonctionnelle																								
	ds1	ds2	ds3	ds4	ds5			ds6			ds7			ds8			ds9			ds10			ds11			ds12		
Index	Etat moteur	Bougies de préchauffage	T° moteur	Sous-zone	Injection	Etat	M.F	Filtrage	Etat	M.F	Explosion	Etat	M.F	Transition	Etat	M.F	Pression	Etat	M.F	Accumulation	Etat	M.F	Mouvement par frottement	Etat	M.F	Mouvement alternatif et rotatif	Etat	M.F
1 (RD forte)	Tourne	Mauvais	-	-	Pompe injection	Non entraî-née	A	-	-	-	Bougies	Etin-celle	N	-	-	-	Filtre	Gaz circ-ule	N	-	-	-	-	-	-	Vilebrequin	Mouv. contin u	N
2 (RD faible)	Arrêt	-	Haute	Partie haute	-	-	-	Pot catalyt-ique	Etan-che	N	-	-	-	Refroidi-sseur	Circ-ule l'air	N	-	-	-	R.G.E	Bonne press-ion	N	-	-	-	-	-	-
3 (RD forte)	Tourne	Mauvais	-	-	Porte injecteur	Transit partie-lle	A	-	-	-	Bougies	Etin-celle	N	-	-	-	Electro-vanne	Carbu passe pas	A	-	-	-	-	-	-	-	-	-
4 (RD forte) & ≠	Tourne	-	-	-	Pompe injection	Entraî-née	N	Monoli-the	Air suffi-sant	N	Bougies	Etin-celle	N	-	-	-	-	-	-	-	-	-	Arbre à cames	Mouv. disc-ontinu	A	-	-	-
5 (RD forte) & ≠	Arrêt	-	Moye-nne	-	-	-	-	Pot catalyt-ique	Etan-che	N	Bougies	Etin-celle	N	Joint de collecteur	Etan-che	N	-	-	-	Vanne de recirculation	Pass-age d'air	N	Poussoirs	Mouv. dévié axe	A	DVA	Mouv. contin u	N
6 (RD faible)	Tourne	Mauvais	-	-	Rampe injection	Absen-ce parti-cule	N	-	-	-	Bougies	Etin-celle	N	-	-	-	Turbo-compre-sseur	aucun bruit	N	-	-	-	Courroie de distribution	Serré e	N	-		-
7 (RD Forte)	Tourne	Mauvais	-	-	Pompe injection	Entraî-née	N	-	-	-	Bougies	Etin-celle	N	-	-	-	Compres seur	Gaz circ-ule	N	-	-	-	-	-	-	Vilebrequin	Mouv. disc-ontinu	A

Figure 2.18. Aperçu de la partie problème de la base de cas d'un moteur à explosion

	Solution			
	Ds1	Ds2	Ds3	Ds4
Index	Classe de défaillance	Identification composant défaillant	Action de réparation associée	Zone de défaillance
1 (RD forte)	Attelage mobile	Pompe injection "livre pas mazout"	Changer pompe	Carburant
2 (RD faible)	Attelage mobile	Turbo-compresseur "Prise d'air"	Changer filtre	Echappe-ment
3 (RD forte)	Attelage mobile	Electro-vanne "pbm masse"	Replacer files électro-vanne	Carburant
4 (RD forte) & ≠	Mouvement par pression	Pistons "déformés"	Changer pistons	Carburant + Echappe-ment
5 (RD forte) & ≠	Mouvement par pression	Axes pistons "décalés"	Replacer axes pistons	Carburant + Echappe-ment
6 (RD faible)	Mouvement rotatif	Pignon vilebrequin "bloqué"	Replacer pignon vilebrequin	Carburant + Echappe-ment
7 (RD Forte)	Mouvement rotatif	Vilebrequin "décalé"	Lubrifier vilebrequin	Carburant

Figure 2.19. Aperçu de la partie solution de la base de cas d'un moteur à explosion

Nous interprétons le *cas 1* avec sa partie problème et sa partie solution. Le cas 1 est un cas qui reflète une défaillance au niveau de la pompe d'injection. La partie problème détermine dans sa partie « localisation » l'endroit du composant défaillant qui est défini par le modèle de contexte. D'après les quatre premiers descripteurs, le moteur tourne, les bougies de préchauffage sont dans un mauvais état et les deux descripteurs restants ne sont pas renseignés. Lorsqu'on se réfère au modèle de contexte que nous allons présenter par la suite (Figure 2.22), nous trouverons un certain nombre de composants qui sont potentiellement défaillants dans la zone spécifiée. Ensuite, la partie fonctionnelle du cas indique l'état des composants dans cette zone. Le descripteur « ds_5 » qui reflète le composant « *pompe d'injection* » appartenant à la classe « *injection* » est dans un état « *non entraîné* » engendrant un mode de fonctionnement « *anormal* ». Le descripteur « ds_7 » reflète l'état de la bougie qui est dans un mode de fonctionnement « *normal* » car elle produit des étincelles. Le descripteur « ds_9 » décrit le composant filtre qui fait circuler normalement le gaz. Le descripteur « ds_{12} » indique que le vilebrequin a un mouvement continu et il est en mode de fonctionnement « *normal* ». Les autres descripteurs ne sont pas renseignés.

Quant à la partie solution, le descripteur « Ds_1 » indique que la classe de défaillance du composant concerné est « *Attelage mobile* ». « Ds_2 » précise que le composant défaillant est la « *pompe d'injection* » accompagné d'une remarque que cette pompe ne livre pas de mazout. Ainsi, l'action de réparation associée qui est décrite pas le descripteur « Ds_3 » préconise un changement de la pompe d'injection. Enfin, le descripteur « Ds_4 » indique que la défaillance s'est produite au niveau du flux de passage du carburant. L'index du cas 1 montre que ce cas a une relation de dépendance forte impliquant un composant dans la partie problème appartenant à la même famille que celle du composant indiqué dans la partie solution.

Modèle hiérarchique des composants du moteur à explosion

L'analyse fonctionnelle des composants du moteur nous a permis de regrouper ces composants en famille fonctionnelle, donnant naissance à un modèle taxonomique des composants. Etant donné que le moteur dispose d'une multitude de composants, nous montrons sur la Figure 2.20 une partie du modèle hiérarchique des composants.

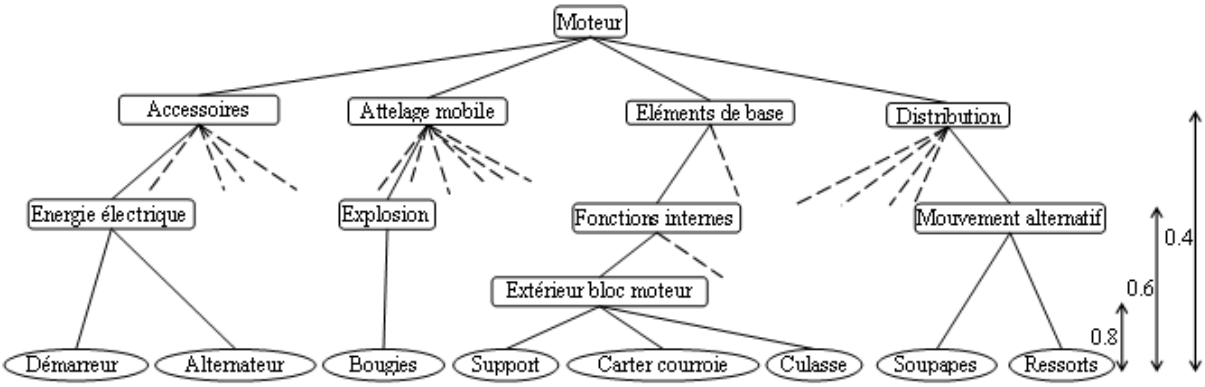


Figure 2.20. Aperçu du modèle hiérarchique des composants moteur

Le modèle hiérarchique est composé de quatre classes : *Attelage mobile*, *Distribution*, *Eléments de base* et *Accessoires*.

Pour les besoins de l'exemple d'illustration qui sera utilisé au chapitre 4, nous ne détaillerons que les deux classes « *Attelage mobile* » et « *Distribution* ».

La classe *Attelage mobile* se compose de sept sous-classes qui comportent les composants suivants :

- injection : porte injecteurs, rampe d'injection, pompe d'injection ;
- explosion : bougies de préchauffage ;
- débit pression : turbocompresseur, électrovanne de pilotage turbocompresseur ;
- filtrage : pot catalytique, monolithe catalytique ;
- mouvement rotatif : vilebrequin, coussinets de vilebrequin, Double Volant Amortisseur (DVA) ;
- mouvement alternatif : bielles, coussinets de bielles ;
- mouvement par pression : pistons, axes de piton.

La classe *Distribution* se compose de cinq sous sous-classes comportant les composants suivants :

- transition : joint de collecteur d'échappement, collecteur d'échappement, tuyaux d'entrée échangeur RGE, refroidisseur de gaz RGE ;
- accumulation : vanne de recirculation gaz d'échappement, boitier de coupure d'air d'admission, système RGE (Recyclage des Gaz d'Echappement) ;
- mouvement rotatif : poulie filtrante, pignon de vilebrequin, poulie arbre à cames, galet tendeur automatique, poulie de pompe d'injection ;
- mouvement alternatif : soupapes, ressorts de soupape ;

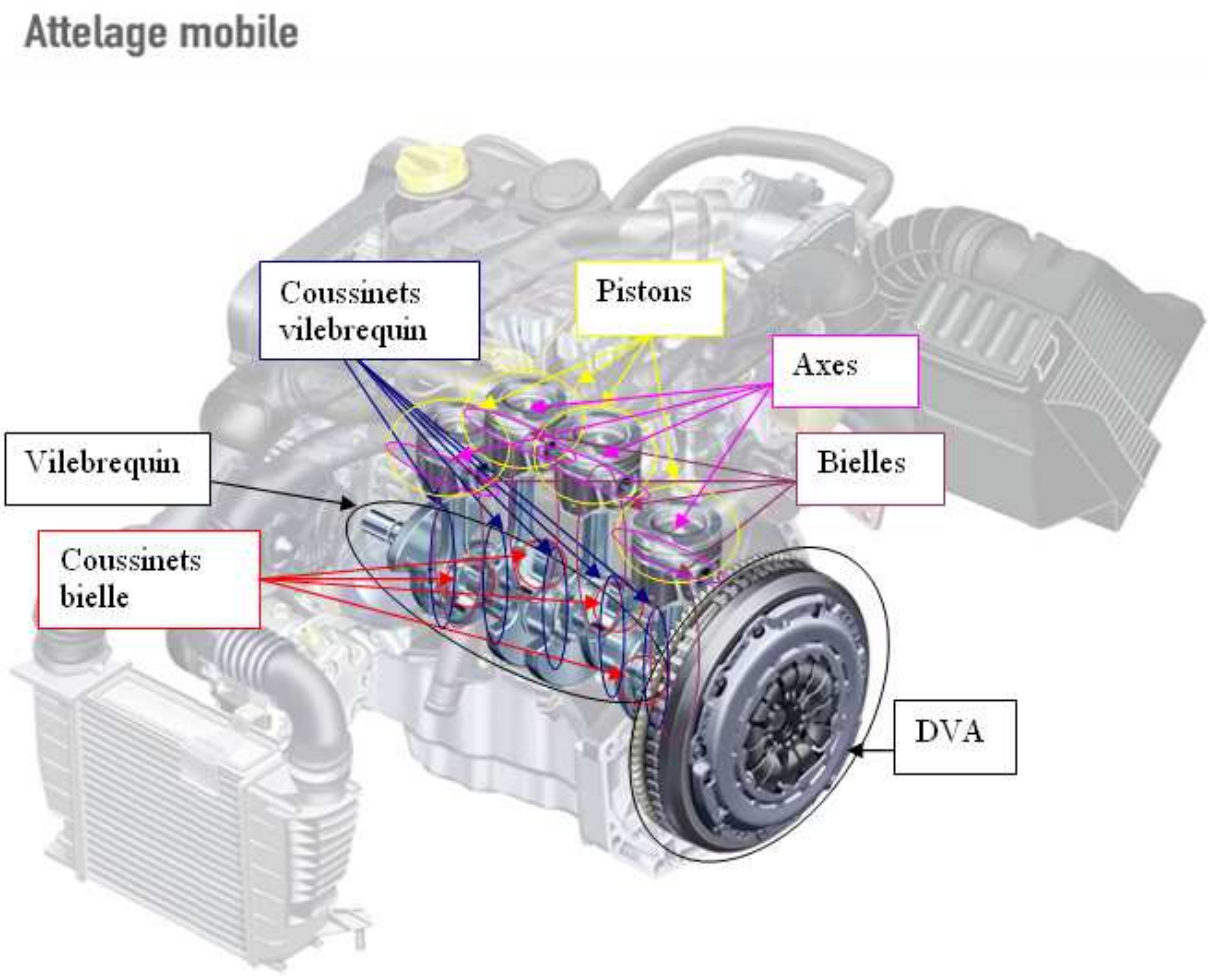


Figure 2.21. Aperçu des composants du moteur appartenant à la famille « Attelage mobile »

- mouvement par frottement : poussoirs de soupape, arbre à cames, courroie de distribution.

La Figure 2.21 montre quelques composants de la classe « *Attelage mobile* » et leur disposition réelle sur le moteur.

Modèle de contexte du moteur à explosion

Le contexte représente les relations de cause-à-effet permettant d'une part la localisation des composants à problèmes, et d'autre part de sélectionner les bons descripteurs par rapport à l'ensemble. Le modèle de contexte est issu d'une décomposition de l'équipement du système de diagnostic en l'occurrence le moteur, qui détermine les fonctions assurées par l'équipement et ses composants. Il reflète le découpage spatial en zones et sous-zones, auxquels sont

associés les composants présents dans l'endroit précis. Ces composants présents constituent donc le contexte dans lequel le composant défaillant est identifié. Un aperçu global du modèle de contexte du moteur est montré sur la Figure 2.22.

Le modèle de contexte est mis en place en tenant compte des différents types de flux dans le moteur. Les flux que nous avons identifiés sont : *échappement, air, lubrification, refroidissement, carburant* et *électrique*.

Nous détaillons uniquement les composants se trouvant dans les flux carburant et échappement car seuls ces deux circuits serviront à illustrer les phases de remémoration et d'adaptation que nous allons aborder au chapitre 4 :

Concernant le flux « *carburant* », nous trouvons les composants suivants :

- filtre décanteur à gazole, circuit alimentation et retour gazole ;
- pompe d'injection ;
- rampe d'injection ;
- portes injecteurs ;
- bougies de préchauffage ;
- pistons ;
- axes de piston ;
- filtre décanteur à gazole ;
- bielles ;
- coussinets de bielle ;
- coussinets de vilebrequin ;
- vilebrequin ;
- arbre à cames ;
- ressorts de soupape ;
- poussoirs de soupape.

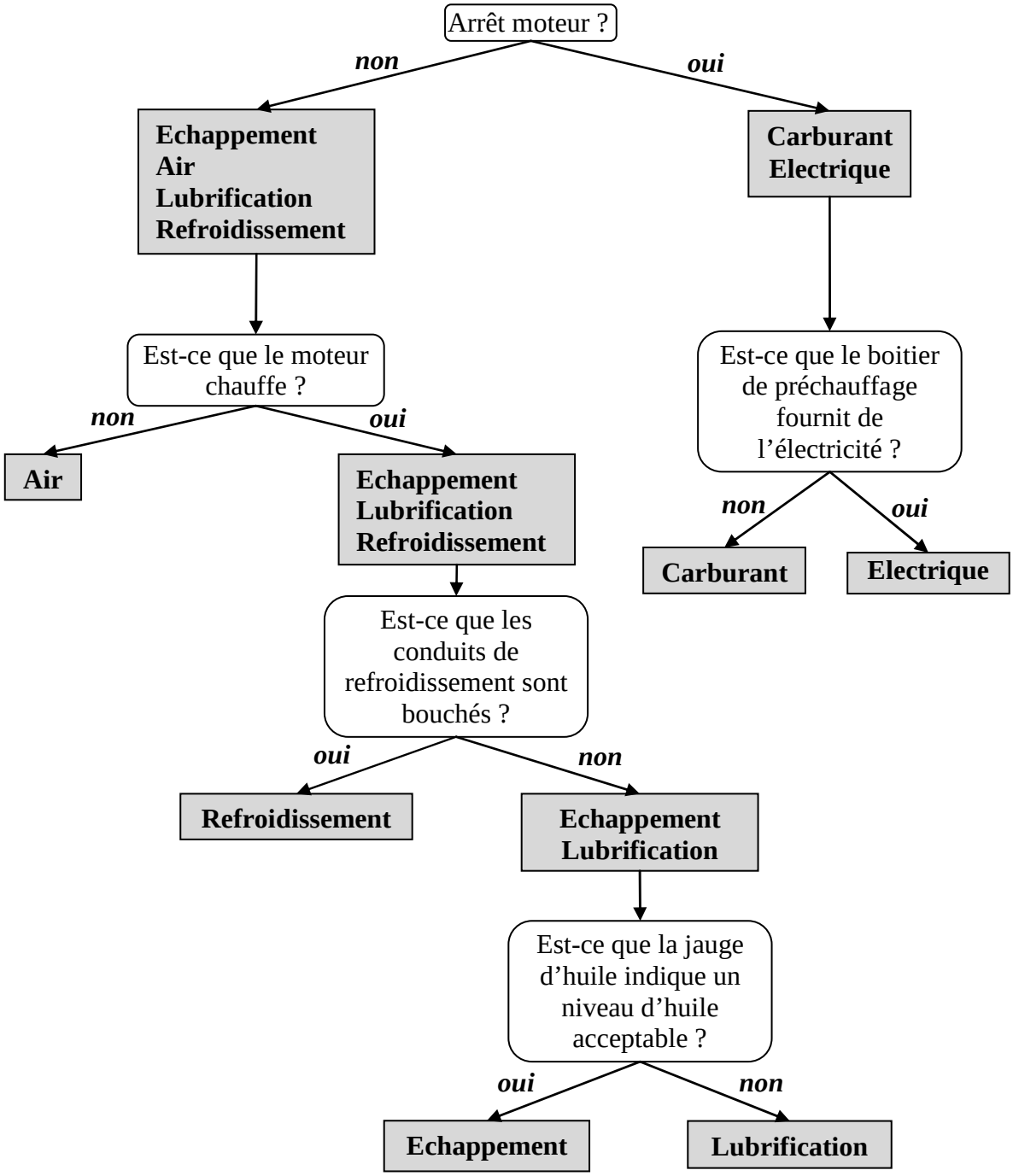


Figure 2.22. Vue globale du modèle de contexte du moteur étudié

Quant au flux « *échappement* », nous trouvons les composants suivants :

- partie haute ;
    - joint de collecteur d'échappement ;
    - collecteur d'échappement ;
    - tuyaux d'entrée échangeur RGE ;
    - refroidisseur de gaz RGE ;

  - vanne de recirculation gaz d'échappement ;
  - boitier de coupure d'air d'admission ;
  - système RGE (Recyclage des Gaz d'Echappement) ;
  - pistons ;
  - axes de piston ;
  - filtre décanteur à gazole ;
  - bielles ;
  - coussinets de bielle ;
  - coussinets de vilebrequin ;
  - vilebrequin ;
  - arbre à cames ;
  - ressorts de soupape ;
  - poussoirs de soupape.
- partie basse ;
  - turbocompresseur ;
  - pot catalytique ;
  - monolithe catalytique ;
  - électrovanne de pilotage turbocompresseur.

En observant les composants présents dans chacun des flux, nous pouvons constater qu'il y a des composants qui appartiennent aux deux flux en même temps car ils se trouvent à l'intersection de ces deux. A titre d'exemple, les composants turbocompresseur et portes injecteurs se trouvent à l'intersection des deux flux « carburant » et « échappement ». Nous notons également que nous pouvons trouver plusieurs classes de fonctionnement dans un même circuit et vice versa (c'est-à-dire qu'il peut y avoir plusieurs circuits dans une même classe de fonctionnement).

Le container de similarité contient des mesures qui tiennent compte de ces trois types d'attributs relatifs à la valeur du composant, son état et son mode de fonctionnement : $ds_i = (ds_i^{Valeur}, ds_i^{Etat}, ds_i^{M.F})$.

Le container d'adaptation prend appui sur les deux modèles de connaissance mis en place permettant de procéder à différents types d'adaptation.

3.2.3. Cycle de RàPC

Le cycle de RàPC commence par l'élaboration du cas cible qui prend appui sur les deux modèles de connaissance. En effet, cette élaboration comporte une première étape de localisation de la défaillance grâce au modèle de contexte qui fournira un certain nombre de composants potentiellement défaillants, présents dans la zone de défaillance localisée. L'état de ces composants est ensuite fourni grâce à un logiciel de supervision. Les règles de décision fournissent le mode de fonctionnement de chaque composant localisé. Ainsi, le cas cible est renseigné. Un exemple d'application concernant l'élaboration du cas cible est donné au chapitre 5 (section 3).

Nous passons maintenant à la deuxième phase du cycle, la remémoration qui sélectionne les cas les plus similaires au cas cible, grâce à l'algorithme des k plus proches voisins, à la taxonomie des composants et aux mesures se trouvant dans le container de mesures de similarité. Cette phase est détaillée au chapitre 4 (section 4.2).

La phase d'adaptation exploitera les relations de dépendance en parcourant un modèle de contexte. Ce modèle définira les relations de cause à effet entre descripteurs. De plus, cette phase s'appuie sur le modèle de taxonomie des composants afin de généraliser les solutions des cas à leur classe fonctionnelle. L'étude de cette phase est faite également au chapitre 4 (section 4.3).

Ce type de système développé n'étant pas arrivé à maturité, nous n'avons pas étudié sa maintenance.

4. Conclusion

Nous avons présenté au chapitre 2 un état de l'art concernant les systèmes de diagnostic par RàPC. Cette analyse bibliographique est réalisée à partir de critères en lien avec la modélisation des connaissances utilisées, la représentation du cas, les phases de remémoration, d'adaptation et de maintenance de la base de cas. Ceci a permis de comparer les différents systèmes de diagnostic par RàPC existants. En effet, nous avons constaté qu'il y a deux types de systèmes : les systèmes orientés *mining* et les systèmes orientés *connaissance*. Ces systèmes se différencient entre autre par la représentation du cas et la modélisation des connaissances du domaine.

Nous avons ainsi construit ces deux types de systèmes en développant les différentes phases décrites dans un modèle générique de RàPC inspiré par Lamontagne et Lapalme

[2002]. En effet, ce modèle se compose de trois parties : la première partie concerne les ressources du domaine spécifiques aux outils de sûreté de fonctionnement (analyse dysfonctionnelle…) à l'historique de pannes, aux documents de différentes natures ainsi qu'à l'expertise du domaine. La deuxième partie comporte les quatre containers de connaissance et la troisième partie comprend les différentes phases du cycle de RàPC. Les deux premières parties s'intègrent dans un processus off-line, tandis que la troisième partie s'exécute dans un processus on-line.

Notre première étude concerne un *système orienté mining* qui a une représentation de cas complète mais triviale. Ce système peut être assimilé à un système de classification dont le cas contient toute la connaissance du système. La représentation du cas est réalisée par l'utilisation des outils de sûreté de fonctionnement à savoir l'AMDEC, les arbres de défaillances, l'historique des pannes, et se présente sous forme de liste d'attributs-valeurs. Cette liste reflète tous les composants du système à diagnostiquer associés à la zone de défaillance, l'état de ces composants ainsi que la classe de défaillance issue de la situation de diagnostic. Les attributs que nous mettons en place dépendent les uns des autres suivant le contexte d'utilisation. Notre système orienté mining dispose d'une simple phase de remémoration en utilisant des algorithmes de recherche classiques (k plus proches voisins, réseaux de lenz…). Nous avons mis en place une mesure de similarité qui tient compte d'une part, des différents types de valeurs à savoir des valeurs numériques et modales et d'autre part, de la présence des valeurs d'attributs. La phase d'adaptation n'a pas été développée. Toutefois, ce type de système peut arriver facilement à une explosion combinatoire de cas. Par conséquent, nous nous intéresserons plus particulièrement à la phase de maintenance de la base de cas qui sera approfondie au chapitre 3.

Notre deuxième étude concerne un *système orienté connaissances (knowledge)* qui a une représentation de cas plus complexe et plus générique. Cette représentation est orientée objet et permet de proposer un modèle hiérarchique. Le cas est associé à deux modèles de connaissances qui peuvent être exploités par les différentes phases du cycle de RàPC. Un premier modèle hiérarchique regroupe les composants en familles fonctionnelles et permet d'exploiter les relations d'héritage. Un deuxième modèle de contexte décrit les relations entre les composants ainsi que le découpage spatial du système à diagnostiquer. Ce modèle permet, d'une part, de localiser la défaillance et, d'autre part, de déterminer plus précisément les composants potentiellement défaillants, tout en démontrant leurs interactions. Contrairement à

la plupart des systèmes de diagnostic industriel, nous mettons un soin particulier au développement de la phase d'adaptation. Cette phase est en liaison directe avec la phase de remémoration. En effet, les systèmes de diagnostic basés sur le RàPC utilisent plusieurs techniques de remémoration. Cependant, il n'y a qu'un seul système orienté connaissance parmi ceux que nous avons étudiés qui exploite la remémoration guidée par l'adaptation. Fort de ce constat, nous ferons un état de l'art sur ces travaux et nous proposerons une méthode de remémoration guidée par l'adaptation au chapitre 4.

Chapitre 3

Proposition d'une méthode de Maintenance de la Base de Cas

1. Introduction

Nous nous consacrons dans ce chapitre à l'étude des systèmes orientés *mining*. Ce type de système ne s'occupe pas de la gestion des connaissances et possède des cas triviaux, ayant une structure complète, qui sont généralement sous forme d'attribut-valeur. Des travaux sont réalisés pour évaluer ces systèmes de RàPC et les maintenir dans un état opérationnel. La maintenance devient par conséquent un sujet central qui a comme objectif de garantir la qualité du système. Cette qualité dépend de plusieurs critères (compétence, performance, etc.). De ce fait, maintes recherches dans ce domaine ont été effectuées dans le cadre de la maintenance des systèmes de RàPC [Roth- Berghofer & Iglezakis, 2001], Richter [1998], [Smyth & McKenna, 1998].

Maintenir un système de RàPC revient à maintenir ses « containers » de connaissance à savoir le container de vocabulaire, de mesures de similarité, d'adaptation et de la base de cas. L'étude de la maintenance des systèmes de RàPC a été abordée dans un rapport interne mais qui n'apporte pas de contribution significative à la maintenance de la base de cas et qu'on omettra volontairement afin de ne pas alourdir ce chapitre.

De façon synthétique, on peut dire que les cas contiennent toute la connaissance d'un système de RàPC et sont stockés dans la base de cas. De ce fait, la consultation de la base de cas est primordiale pour établir des opérations de maintenance Leake et Wilson [2000a].

Nous nous sommes par conséquent particulièrement intéressés aux travaux de maintenance de la base de cas (MBC). Un état de l'art concernant ces travaux sera abordé à la section 2. Une analyse de la maintenance de la base de cas permet de déterminer quand et comment un système de RàPC exécute la maintenance de la base de cas. Il y a deux principales familles de politique de MBC : *le partitionnement de la base de cas* et *l'optimisation de la base de cas*. La politique de partitionnement concerne les bases de cas de très grande dimension. Elle élabore une hiérarchie entre plusieurs petites bases de cas, ce qui induit un traitement des cas en plusieurs étapes. La politique d'optimisation de la base de cas permet de réduire sa taille tout en préservant sa qualité qui est définie par un certain nombre de critères. Nous nous sommes intéressés à cette dernière politique et notamment aux deux stratégies *Backward Elimination (BE)* et *Forward Selection (FS)*. Ces stratégies sont fondées sur des critères de qualité. Les deux critères sur lesquels nous prenons appui sont la compétence (pouvoir de résolution de la base de cas des problèmes cibles) et la performance (efficacité ou problème d'utilité). Généralement, ces deux critères dépendent du nombre de

cas dans la base de cas. Les petites bases de cas possèdent un bon potentiel d'efficacité, mais souffrent de la taille réduite de l'espace de problème cible qu'elles peuvent couvrir, et ont donc une compétence limitée. A l'inverse, les grandes bases de cas ont une bonne compétence, mais elles sont moins efficaces face aux problèmes d'utilité. De ce fait, nous devons trouver un compromis entre le nombre de cas dans la base de cas et ses critères de qualité.

Après la présentation de l'état de l'art, nous exposons notre contribution qui prend appui sur les avantages que présentent les deux types de stratégies. En effet, nous proposons une méthode de structuration de la base de cas développée selon les deux critères : de compétence et de performance. En outre, contrairement à la plupart des systèmes d'intelligence artificielle, les systèmes de RàPC peuvent fonctionner à partir d'un ensemble d'apprentissage incomplet. En effet, ces systèmes peuvent être conçus à partir d'un nombre limité de cas qui s'enrichit de nouvelles expériences résolues (cas) au fil du temps. Nous devons veiller à ce que cette évolution n'altère pas la qualité de la base de cas. Par conséquent, nous proposons une méthode d'auto-incrémentation de la base de cas préservant la qualité des cas et leur structuration.

La validation de ces deux propositions relatives à la maintenance de la base de cas et de son auto-incrémentation est faite à la section 4. Nous procédons à cette validation à travers une étude comparative de notre méthode avec d'autres méthodes existantes, sur des benchmarks de référence, en utilisant le critère de compétence, de performance ainsi que le taux d'apprentissage des cas dans la base de cas.

2. Maintenance de la base de cas

2.1. Introduction

La qualité d'un système de RàPC est définie par la représentation d'un cas, l'organisation de la base de cas, les diverses indexations utilisées, la définition de « bonnes » mesures de similarités pour la recherche de cas ainsi que le lien entre la recherche et l'adaptation du cas [Leake & Wilson, 1998].

Par définition, les systèmes de RàPC sont conçus pour fonctionner sur de longues périodes et/ou sont amenés à évoluer en traitant un grand nombre de données et de cas, ce qui peut poser des problèmes dans la phase de recherche de cas ainsi que dans la phase d'adaptation. Ces phases peuvent demander, lors de leur réalisation, beaucoup de temps et la

qualité la base de cas peut être détériorée ; ce qui se répercute sur les résultats fournis par le système.

Pour anticiper ces problèmes, la maintenance du système de RàPC devient nécessaire. Plusieurs travaux ont porté sur la maintenance des systèmes de RàPC, mettant en application des politiques de mise à jour des représentations des cas et s'intéressant à leur réorganisation dans l'objectif de faciliter le raisonnement futur tout en garantissant une bonne performance du système [Simon & Yeung, 2001]. Richter [1995 ; 1998] définit la maintenance des systèmes de RàPC comme étant des techniques de contrôle et de réaction face aux changements de ses différents containers de connaissances du système, à savoir : le container de vocabulaire, des mesures de similarité, d'adaptation et de la base de cas. Ces containers de connaissances ont été présentés au chapitre 1. Nous pouvons ainsi associer à chaque container de connaissance une maintenance.

Les trois premiers containers sont mis en place lors de la conception du système et donc avant que le système ne soit mis en marche, tandis que la base de cas est remise à jour d'une façon dynamique pendant que le système fonctionne. Richter [1998] considère que chaque container peut contenir toute la connaissance d'un système de RàPC, et les manipulations des connaissances sur un container peuvent changer l'organisation et la représentation des connaissances sur les autres containers.

Nous pouvons considérer que la base de cas joue un rôle central dans les systèmes de RàPC. Ce qui explique que la majorité des travaux faits dans le domaine de la maintenance des systèmes de RàPC est essentiellement fondée sur la maintenance de la base de cas. Iglezakis et Roth-Berghofer [2000] soutiennent qu'il ne peut y avoir de maintenance de système de RàPC sans le balayage de la base de cas. De plus, la base de cas est associée à la connaissance du système et que les opérations de maintenance ne sont déclenchées que grâce à elle.

La Maintenance de la Base de Cas (MBC) est la mise en œuvre des politiques permettant de réviser l'organisation et/ou le contenu (représentation, domaine d'application, contenu d'informations, ou l'implémentation) de la base de cas afin d'améliorer le raisonnement futur [Leake & Wilson, 1998]. Cette maintenance reflète un ensemble de réalités différentes, telles que la suppression des cas peu importants, la sélection des groupes de cas permettant d'éliminer la redondance et d'améliorer le pouvoir de raisonnement du système, et la réécriture des cas afin de réparer des problèmes d'incohérences [Smyth & McKenna, 1998].

Par ailleurs, la maintenance de la base de cas peut commencer par un processus d'analyse [Leake & Wilson, 1998]. Ce processus permet d'enclencher l'opération de maintenance qui peut se faire « en ligne » ou « hors ligne ». Ces opérations de maintenance prennent appui sur des critères d'évaluation de la qualité de la base de cas. Il existe plusieurs critères que nous abordons à la section 2.3. De plus, il y a plusieurs manières d'accomplir la maintenance de la base de cas suivant des politiques et des stratégies. Nous abordons ces dernières à la section 2.4.

2.2. Processus de la maintenance de la base de cas

Leake et Wilson [1998] se sont intéressés au processus de maintenance de la base de cas en présentant une première analyse de ce processus. Cette analyse permet de déterminer quand et comment un système de RàPC exécute cette maintenance en catégorisant les approches de la MBC suivant des politiques bien définies. Ces approches sont décrites selon la manière de rassembler les données pertinentes pour la maintenance à travers un cadencement périodique, ad-hoc ou conditionnel. Ce cadencement permet également de définir le moment de déclenchement de la maintenance suivant une intégration « en ligne » ou « hors ligne ». Cette analyse mène directement aux choix des types d'opérations de maintenance possibles à mettre en place et enfin la manière d'exécuter les opérations sélectionnées suivant un timing choisi.

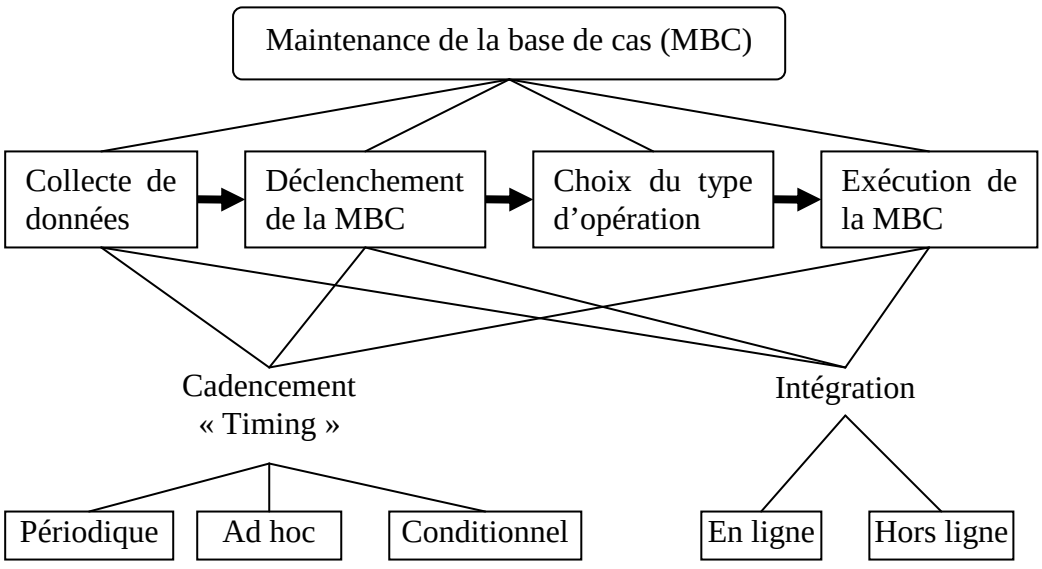


Figure 3.1. Catégorisation de la maintenance de la base de cas selon le cadencement et le mode d'intégration des différentes phases [Leake & Wilson, 1998]

La Figure 3.1, illustre la catégorisation de la maintenance de la base de cas selon le cadencement et le mode d'intégration des différentes phases dédiées à cette maintenance.

Reinartz et al. [2000] ont précisé que la MBC intervient dans le cycle de RàPC et plus précisément à la fin de ce cycle au moment de l'apprentissage des cas dans la base de cas. En effet, ils ont proposé deux nouvelles phases dans ce cycle à savoir une phase de *scrutation* et une phase de *restauration* comme on a pu le voir précédemment (cf. Figure 3.2).

- *La phase de scrutation :* cette phase considère l'état courant de la base de cas et évalue sa qualité. La qualité de la base de cas dépend d'un certain nombre de critères qui sont en fonction du nombre de cas dans la base de cas, de leur pouvoir de résolution de problèmes rencontrés et du temps de réponse. Nous détaillerons cette notion de « qualité » à la section 2.3. Si la qualité est mauvaise, cette phase suggère des changements spécifiques pour avoir la qualité désirée. Elle permet également le déclenchement de la maintenance pendant le fonctionnement « en ligne » du cycle de RàPC en proposant, entre autre, plusieurs types d'opérations de maintenance ;
- *La phase de restauration :* cette phase intervient dans le cas où la qualité de la base de cas est insatisfaisante après la phase de scrutation. Elle permet la sélection des méthodes qui seront utilisées pour choisir un opérateur de modification parmi ceux suggérés par la phase de scrutation. Ces opérateurs permettent le changement du contenu de la base de cas. Ainsi, la sélection d'opérateur de modification a pour objectif d'obtenir le niveau de qualité requis. De ce fait, la qualité de l'information est mise à jour pour refléter le changement.

Les mêmes auteurs précisent qu'un troisième champ est ajouté dans la représentation du cas : « information de qualité » qui contient toutes les données nécessaires pour effectuer la maintenance de la base de cas.

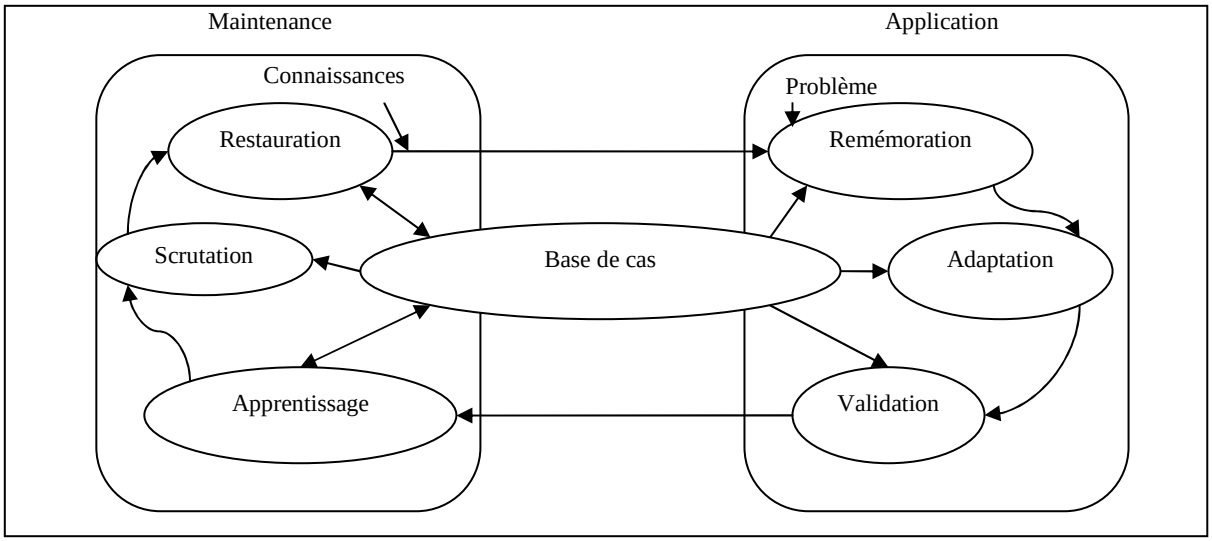


Figure 3.2. Ajout de deux phases au cycle de RàPC [Reinartz et al., 2000]

En effet, le problème est traité en lui donnant une solution adéquate dans la partie « application ». Ensuite, dans la partie maintenance, l'apprentissage du cas dans la base de cas débute. Avant d'ajouter un cas dans la base de cas, la phase de scrutation évalue ce dernier par rapport à l'ensemble de cas de la base de cas suivant des critères de qualité qui vont être abordés à la section suivante. Si le cas est retenu, la base cas est remise à jour dans la phase de restauration.

Par ailleurs, afin d'évaluer la qualité de la base de cas, nous prenons appui sur un certain nombre de critères que nous illustrons dans la section suivante.

2.3. Critères d'évaluation de la qualité de la base de cas

En général, une base de cas est jugée de bonne qualité quand elle permet à un système de RàPC de résoudre le plus grand nombre de problèmes possibles d'une façon correcte en un temps acceptable. L'évaluation de la qualité de la base de cas est primordiale pour permettre d'effectuer sa maintenance. Dans la littérature, plusieurs auteurs, [Racine & Yang, 1996], [Reinartz et al., 2000], [Roth-Berghofer & Iglezakis, 2001], [Smyth & Keane, 1995b], ont proposé des critères d'évaluation permettant de juger la qualité de la base de cas en : pouvoir de résolution (compétence), temps de réponse (performance), conflit entre problèmes et solutions (inconsistance), répétition des problèmes-solutions (redondance), généralisation des cas (degré d'abstraction) et adéquation des réponses fournies (pertinence).

Les différents critères utilisés sont les suivants :

Compétence : la compétence de la base de cas est mesurée par le nombre de problèmes différents pour lesquels la base de cas apporte une bonne solution à un problème donné. Ce critère est étroitement lié à deux notions à savoir le *recouvrement* et *l'atteignabilté* [Smyth & Keane, 1995b].

- L'ensemble de recouvrement (*coverage*) d'un cas représente l'ensemble des cas cibles qu'il peut résoudre.
- L'ensemble d'atteignabilité (*reachability*) d'un cas est l'ensemble de cas qui peuvent être utilisés pour le résoudre.

De ce fait, le critère de compétence vise à maximiser le recouvrement de la base de cas et à minimiser son atteignabilité. Nous verrons en détail ces deux notions à la section 2.4.2 (méthodes de suppression à partir de catégorisation des cas).

Performance : la performance de la base de cas est mesurée par le temps de réponse qui lui est nécessaire pour proposer une solution à un cas cible [Smyth & Keane, 1995b]. Cette mesure est liée directement aux coûts d'adaptation (le temps nécessaire pour adapter la ou les solutions des cas sources les plus similaires au cas cible) et aux coûts de recherche (le temps mis pour remémorer un ou plusieurs cas similaires au cas cible) des cas similaires dans la base de cas par rapport au cas cible. La performance peut donc être calculée en fonction de la *précision* (*accuracy*) de la base de cas, qui représente le pouvoir de classification de cette dernière, et du *nombre de cas* dans la base de cas (*storage*). Il existe dans la littérature plusieurs stratégies dédiées à l'étude de ce critère [Minton, 1990], [Smyth & Cunningham, 1996], [Leake & Wilson, 2000a].

Inconsistance : ce critère peut être défini de plusieurs manières. Un cas est inconsistant avec l'ensemble des cas dans la base de cas s'il apporte une information erronée par rapport au contexte de connaissance. Deux catégories d'inconsistances sont avérées [Racine & Yang, 1996] :

- ***inconsistance intra-cas :*** cette inconsistance apparaît quand les valeurs affectées à différentes caractéristiques dans un seul cas ne satisfont pas toutes les contraintes liées à ces caractéristiques ;
- ***inconsistance inter-cas :*** cette inconsistance apparaît lorsqu'il y a deux ou plusieurs cas qui ont les mêmes caractéristiques dans l'espace problème avec des

solutions complètement différentes. $Cas_1 = ((P_1), (S_1))$ est inconsistant avec un ensemble *E* de cas s'il existe un cas $cas_2 = ((P_2), (S_2)) \in E$ qui peut résoudre un problème plus général ($P_1 \subset P_2$) de manière différente ($S_1 \neq S_2$).

Redondance : avec l'évolution de la base de cas, il est important de déterminer les cas qui ont les mêmes caractéristiques ou qui peuvent être représentés par d'autres cas plus généraux. Un cas « cas_1 » est redondant dans la base de cas s'il est subsumé par un autre cas « cas_2 » de la base de cas. La relation de subsomption entre deux cas : $cas_1 = ((P_1), (S_1))$ et $cas_2 = ((P_2), (S_2))$ est définie par : cas_2 subsume cas_1 si $P_1 \subset P_2$ et la solution de cas_2 est plus précise que celle de cas_1 ($S_2 \subset S_1$) [Racine & Yang, 1996].

Degré d'abstraction : ce critère est lié au niveau de généralisation des cas que contient la base de cas. Une base de cas peut contenir des cas concrets ou des cas généralisés. Cette généralisation représente le degré d'abstraction de la base de cas [Racine & Yang, 1996].

Pertinence : une base de cas a pour objectif de fournir à l'utilisateur uniquement des cas pertinents pour la résolution d'un problème donné en fournissant des réponses adéquates. Par conséquent, les cas qui ne correspondent pas exactement à la résolution d'un problème donné ne doivent pas être gardés dans la base de cas [Racine & Yang, 1996]. Cependant, il existe plusieurs définitions de la pertinence dans la littérature. Ces définitions ont été exposées et discutées dans les travaux de [Blum & Langey, 1997]. La raison de cette variété se résume à la question suivante : *pertinent par rapport à quoi ?*

Généralement, les critères cités ci-dessus ne sont pas tous utilisés en même temps lors de l'évaluation de la qualité de la base de cas. Dans les travaux existants de [Leake & Wilson, 2000a], [Yang & Wu, 2000], un seul critère est utilisé ou au maximum deux qui sont combinés [Smyth & Keane, 1995b].

Nous retenons essentiellement dans notre étude les critères de compétence et de performance car se sont les deux critères qui sont majoritairement utilisés dans la plupart des travaux, liés à l'optimisation d'une base de cas.

2.4. Politiques et stratégies de la maintenance de la base de cas

Il existe différentes politiques et stratégies permettant de procéder à la maintenance de la base de cas. Nous recensons deux politiques principales, à savoir : la politique de *réorganisation* et la politique *d'optimisation* de la base de cas. Chacune de ces deux politiques regroupe un certain nombre de stratégies qui utilisent différents critères pour évaluer la base de cas. Toutes ces politiques visent à atteindre un même objectif, la construction et la restructuration de la base de cas avec une meilleure qualité, en utilisant différents critères, par rapport à son état initial.

Nous développons dans la section suivante les différentes stratégies et critères utilisés dans chacune des politiques.

2.4.1. Politique de partitionnement de la base de cas

La politique de partitionnement de la base de cas, contrairement à celle d'optimisation, conserve tous les cas de la base de cas et partitionne cette dernière en plusieurs espaces de recherche. Ceci permet ainsi de réduire le temps de recherche et de sélectionner de manière incrémentale les attributs qui sont riches en information et qui peuvent couvrir la structure de la base de cas entière [Yang & Wu, 2000].

Il existe plusieurs stratégies qui ont été développées associées à cette politique. Certaines d'entre elles prennent en compte l'arrivée dynamique des données [Malek, 2000] tandis que d'autres utilisent des réseaux de neurones statiques [Roh et al., 2003] ou dynamiques [Fdez-Riverola & Corchado, 2003].

Ces travaux sont décrits ci-dessous :

- **Stratégie de regroupement**

Yang et Wu [2000] ont proposé une stratégie qui consiste à garder tous les cas de la base de cas et à les diviser en plusieurs groupes, formant ainsi des petites bases de cas. L'objectif de cette stratégie est de diminuer le temps de recherche des cas qui peut être coûteux en temps et donc d'améliorer la performance de la base de cas.

De ce fait, afin d'assurer l'efficacité de la maintenance de la base de cas, les auteurs ont suggéré deux idées principales :

- Regrouper les cas les plus proches et former ainsi un groupe constitué par ces cas. Ce processus est répété pour tous les cas de la base de cas créant ainsi une collection de bases de cas distribuées ;

- Permettre à l'utilisateur de sélectionner de manière incrémentale les attributs riches en information pour faciliter la recherche dans les différentes bases de cas distribuées. Ceci est fait afin de couvrir toute la structure de la base de cas originale.

Par conséquent, les petites bases de cas sont construites à partir des résultats du regroupement. Pour chaque base de cas créée, un nom lui est attribué représentant sa description ainsi qu'un ensemble d'attributs communs associés aux cas les plus proches. De plus, les auteurs associent un poids à chaque attribut représentant la moyenne des poids relative aux attributs des cas pour chaque petite base de cas.

- **Stratégies de partitionnement par les réseaux de neurones**

Plusieurs stratégies ont été développées dans ce cadre. Celles-ci utilisent généralement les réseaux de neurones pour indexer la base de cas. Cette dernière est alors divisée en plusieurs parties. Chacune de ces parties est formée par un regroupement de cas similaires. Ces regroupements sont réalisés via un réseau de neurones. Cependant, il existe des systèmes qui utilisent des réseaux de neurones évolutifs (ceux qui gèrent l'arrivée dynamique des données) [Fdez-Riverola & Corchado, 2003] et d'autres qui utilisent des réseaux de neurones statiques [Mujica & Vehi, 2003] et [Roh et al. 2003] concernant les données traitées indépendamment de l'environnement extérieur du système.

Malek [2000] a proposé un modèle hybride de mémoire *Probis* « Prototype-Based Indexing System » qui contient une mémoire à deux niveaux d'hiérarchie et se divise en trois parties : « partie connexionniste, partie mémoire plate partitionnée en plusieurs zones et une partie interface qui les relient ». Cette architecture permet d'une part, d'augmenter l'efficacité d'un système de RàPC en temps de réponse et d'autre part, de simplifier le processus d'apprentissage en termes de complexité de calcul et de traitement des cas frontières et des cas oubliés par le réseau. Le modèle proposé contient une mémoire classique (une mémoire plate partitionnée en plusieurs zones) et un réseau de neurones à base de prototypes ou d'exemples. Ce modèle hybride intervient au niveau de la remémoration, de la mémorisation et de l'organisation des cas en mémoire.

Corchado et Lees [2001] ont proposé un système de RàPC utilisant une base de cas sous forme de réseaux de neurones. Les cas sont considérés comme des entrées d'un réseau de neurones à *fonction de bases radiales* (RBF) [Park & Sandberg, 1991]. L'algorithme des k plus proches voisins est utilisé lors de la phase de remémoration. Un ensemble de cas

similaires est extrait en utilisant plusieurs mesures de similarité. Le réseau mis en place permet de généraliser les solutions des cas remémorés pour fournir la solution au cas cible. Il effectue un apprentissage rapide, il possède de bonnes capacités de généralisation et apprend sans oubli. Un intervalle d'erreur est associé à chaque valeur prédite. Durant cette phase d'apprentissage, deux tâches sont effectuées : d'une part, la sauvegarde de la structure interne (poids des centres) du réseau de neurones et d'autre part, la modification de quelques paramètres constituant les cas. Les poids et les centres du réseau de neurones, utilisés pour la prédiction en temps réel, sont remémorés à partir de la base de cas et adaptés en utilisant l'ensemble d'apprentissage. A chaque cas est associée une erreur moyenne. C'est une mesure de l'erreur moyenne des prédictions précédentes pour lesquelles ce cas a été utilisé pour l'apprentissage du réseau de neurones.

Généralement ces stratégies sont efficaces pour le partitionnement de la base de cas et sont utilisées dans des systèmes hybrides tel que *CASEP2* [Zehraoui et al., 2004]. En effet, CASEP2 (Case based reasoning for SEquence Prediction) est un système hybride pour le traitement et le classement (ou la prédiction) de séquences. Le cas est une expérience précise dans une séquence représentant une succession d'évènements instantanés. CASEP2 combine l'approche du RàPC avec les réseaux de neurones. En effet, sa mémoire est constituée de deux niveaux (cf. Figure 3.3).

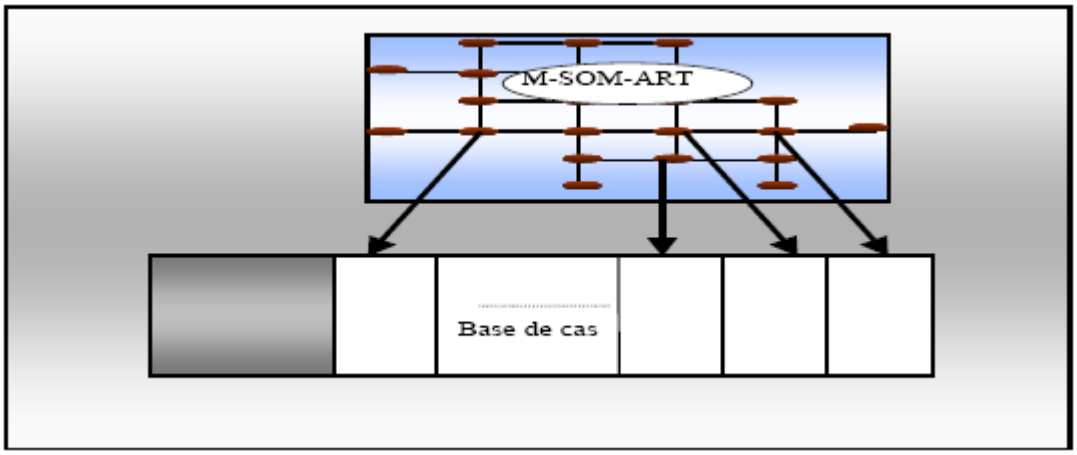


Figure 3.3. Architecture de la mémoire dans le système CASEP2 [Zehraoui et al., 2004]

Le premier niveau de mémoire (M-SOM-ART : Matrice de covariance - Self Organizing Map - Adaptive Resonance Theory) [Zerhaoui & Bennani, 2004] contient les cas

prototypes qui sont représentés par des neurones. Cette mémoire qui est utilisée lors de la phase de remémoration réduit considérablement le temps de recherche car chaque neurone indexe un ensemble de cas similaires dans le deuxième niveau de mémoire. Cette dernière, qui est une simple mémoire plate partitionnée par le réseau de neurones, contient des cas concrets de la base de cas.

L'avantage de cette hybridation réside dans l'efficacité de la remémoration. En effet, le réseau de neurones, qui représente un ensemble de cas, a pour objectif d'améliorer l'efficacité de la remémoration en terme de rapidité d'exécution et la base de cas plate permet d'avoir une réponse précise et adéquate.

La politique de partitionnement de la base de cas est intéressante car elle permet de structurer la base de cas et donc facilite la recherche des cas. De plus, elle permet de fusionner plusieurs techniques et plusieurs types de mémoires afin d'améliorer les résultats concernant le temps de réponse du système ainsi que la rapidité de l'apprentissage des cas dans la base de cas. Cependant, cette politique consiste à garder tous les cas ce qui peut engendrer une baisse de la pertinence de l'information malgré l'aspect d'organisation de la base de cas.

Par ailleurs, la politique d'optimisation de la base de cas, quant à elle, consiste à supprimer les cas les moins importants et donc de diminuer le nombre de cas dans la base de cas. Nous abordons cette politique dans la section suivante.

2.4.2. Politique d'optimisation de la base de cas

Cette politique vise à réduire la taille de la base de cas en enlevant des cas de cette dernière afin de diminuer le temps de recherche, tout en préservant la qualité de la base de cas, suivant des critères prédéfinis. Nous abordons la description des différentes stratégies du point de vue de la sélection de cas présentant un critère maximisé.

Dans le cadre de la réduction d'instances dans le domaine du *data-mining*, Reinartz [2002] s'intéresse aux différentes tâches de la sélection d'instances. En effet, cette sélection se fait en trois étapes (méthodes) : *Sampling*, *Clustering* et *Prototyping* (Figure 3.4).

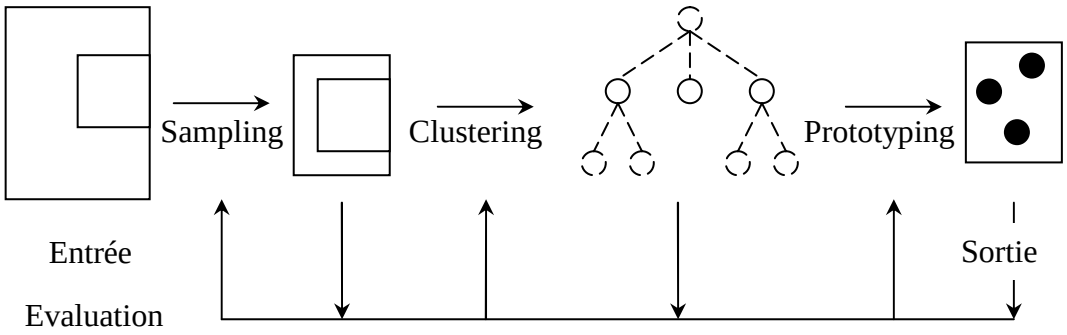


Figure 3.4. Types de méthodes dans la sélection d'instances [Reinartz, 2002]

Cet ordre n'est pas obligatoire. La sélection peut se faire dans n'importe quel ordre. Quelques étapes peuvent être ignorées ou exécutées puis reviennent à l'étape précédente, etc. Enfin, tous ces types de méthodes peuvent être combinés à tout moment.

Le *Sampling* est une méthode qui prend un échantillon « S_i » via un processus aléatoire dans lequel chaque « S_i » reçoit une probabilité appropriée de sélection π_i [Liu & Motoda, 2002]. La probabilité π_i dépend de la taille de l'échantillon qu'on souhaite obtenir. Elle est calculée en fonction du nombre de cas individuels sélectionnés aléatoirement par rapport au nombre de cas initiaux. La mise en œuvre de cette méthode est simple et rapide en terme de temps d'exécution. Cependant, les techniques existantes ne se limitent pas à l'utilisation seule de cette méthode, généralement elle est combinée avec d'autres. Nous pouvons trouver plusieurs méthodes combinées utilisant la méthode Sampling en commun, on peut citer : Simple random simpling, systemic sampling, adaptative sampling et stratified sampling [Cano et al., 2006]. Le Sampling permet donc une sélection aléatoire d'un certain nombre de cas de la base de cas afin de former l'ensemble cas cibles. Cet ensemble de cas cibles va nous servir dans le cadre de notre étude à mesurer la qualité de la base de cas en fonction des cas sources et suivant un critère bien défini.

Le *Clustering* est une méthode qui a pour but de trouver des régularités dans les données non étiquetées [Liu et al., 2002]. Elle consiste à regrouper un ensemble de données en différents paquets homogènes. Ces paquets partagent des caractéristiques communes, qui correspondent le plus souvent à des critères de proximité qui sont définis en introduisant des mesures de distance. Paradoxalement, Smyth et al. [1995] proposent un partitionnement non conventionnel en s'appuyant sur des notions de recouvrement et d'atteignabilité. Quatre catégories de cas sont déterminées. A partir de ces catégories, des prototypes sont définis.

Le *prototyping* est un ensemble de descriptions condensées des ensembles de caractéristiques. En effet, le prototyping suppose qu'une simple caractéristique peut

représenter l'information d'un sous ensemble entier de caractéristiques [Pekalska et al., 2006]. L'idée de base est donc après avoir trouvé les clusters dans un espace plus grand, il faut trouver un prototype qui représente l'ensemble des clusters le plus représentatif possible. Le prototyping permet donc de sélectionner les cas pertinents ou un sous-ensemble de cas qui représente l'ensemble global suivant un critère donné. Cela permet de supprimer quelques cas et de ramener la base de cas à une taille réduite présentant les mêmes caractéristiques que la base de cas originale.

Nous allons décrire les deux principales stratégies dans le cadre de cette politique en abordant les différentes méthodes et critères utilisés dans chacune d'elle [Haouchine et al., 2007a, 2009].

La stratégie Forward Selection (FS)

Cette stratégie consiste, partant d'un ensemble vide, à ajouter un à un les cas jusqu'à ce que tous les cas jugés utiles aient été ajoutés. L'utilité d'un cas dépend du critère utilisé (si on considère à titre d'exemple le critère de redondance, alors un cas utile est un cas non redondant aux cas présents dans la base de cas originale. Ce cas sera sélectionné pour être ajouté à la nouvelle base de cas construite). A chaque étape, la stratégie FS choisit le cas qui, ajouté aux cas déjà sélectionnés, produit le meilleur sous-ensemble intermédiaire de cas selon un certain critère. Précisant qu'une fois qu'un cas ait été ajouté à l'ensemble, il ne peut plus être retiré. L'algorithme général de la stratégie FS est décrit sur la Figure 3.5 [Lereno, 2000].

L'algorithme débute avec un ensemble vide de cas « S' » et ajoute, un à un, les cas de l'ensemble initial « S » à « S' ». A chaque itération, le meilleur cas, selon un critère donné, est sélectionné puis ajouté à l'ensemble « S' » après avoir été retiré de « S ». Le processus de sélection s'arrête lorsqu'un des deux critères d'arrêt est atteint, soit l'ensemble « S » est vide, soit « S' » satisfait le critère *C*.

Cette stratégie consiste donc à construire une base de cas réduite en partant d'une base de cas vierge par l'ajout successif de cas présentant un critère maximisé dans la base de cas entière.

entrée S ensemble d'instances initiales
U mesure

Forward Selection (FS)

S' = Ø

tant que S' ne satisfait pas U ou S = Ø

répéter
Chercher dans S l'objet le plus pertinent ω_{plus}
S' = S + {ω_{plus}}
S = S - {ω_{plus}}

sortie S' ensemble d'instances sélectionnées

Figure 3.5. Algorithme général de Forward Selection (FS)

Nous nous intéressons particulièrement à l'étude des méthodes basées sur les critères de compétence, de performance et sur les règles d'adaptation floue. De ce fait, nous aborderons trois méthodes, chacune d'elles vise à maximiser l'un de ces critères.

- ***Méthodes maximisant le critère de compétence***

Smyth et McKenna [1999b] ont proposé de construire une base de cas en utilisant un algorithme hybride, celui des plus proches voisins CNN (Condensed Nearest-Neighbor) en l'associant à une mesure qui sera appliquée pour chaque cas dans la base de cas originale (initiale). L'algorithme CNN est semblable à celui de l'algorithme général de (FS). La mesure associée est la valeur de *recouvrement relatif « RC »* (*Relative Coverage*).

$$RC(c) = \sum_{c' \in Ensemble\,\mathrm{Re}\,couvrement(c)} \frac{1}{EnsembleAtteignabilité(c')} \qquad (2)$$

Lorsqu'un cas « c' » est recouvert par *n* autres cas, chacun de ces *n* cas va recevoir une contribution de $\frac{1}{n}$ de « c' » par rapport à leur valeur de recouvrement relatif.

La mesure RC détermine donc la valeur de la contribution d'un cas à la compétence de la base de cas. Par conséquent, les cas sont rangés selon leurs valeurs de « RC » dans la base de cas complète. Ensuite, l'algorithme CNN est appliqué pour construire une nouvelle base de

cas réduite. Ceci permet de sélectionner les cas ayant une grande contribution au recouvrement de la base de cas.

Yang et Zhu [2001] déterminent le recouvrement par une mesure de similarité et des coûts d'adaptation. Dans ce cas, le recouvrement d'un cas est considéré comme le voisinage du cas dans certaines limites d'adaptabilité. Si nous considérons $N(x_1)$ comme l'ensemble de cas x_2 dans lequel les solutions $\pi(x_2)$ sont proches de $\pi(x_1)$, alors $N(x_1)$ définit le recouvrement ou le voisinage de x_1.

Formellement : $N(x_1) = \{x_2 | D(\pi(x_1), \pi(x_2)) \leq L\}$.

Où : L : constante limite du seuil du coût d'adaptation de la solution

D : Distance entre les deux solutions des deux cas concernés

Par ailleurs, Yang et Zhu ont proposé un algorithme qui produit une base de cas dont le recouvrement n'est pas inférieur à 63% de celui d'une base de cas optimale.

Dans le même contexte, Mckenna et Smyth [1999] ont démontré que lorsqu'un cas est ajouté dans la base de cas, une des quatre possibilités est envisageable :

- une création d'un nouveau groupe de compétence ;
- l'augmentation de la taille et du recouvrement d'un groupe existant ;
- la formation d'un super groupe à partir d'un regroupement d'un ensemble de groupes déjà existants ;
- l'augmentation de la taille d'un groupe existant mais pas en recouvrement.

Ensuite, au moment de la construction de la nouvelle base de cas réduite, ils ont démontrés que cette dernière passe par quatre phases de développement qui sont les suivants : *enfance*, *adolescence*, *âge adulte* et *vieillesse* (cf. Figure 3.6).

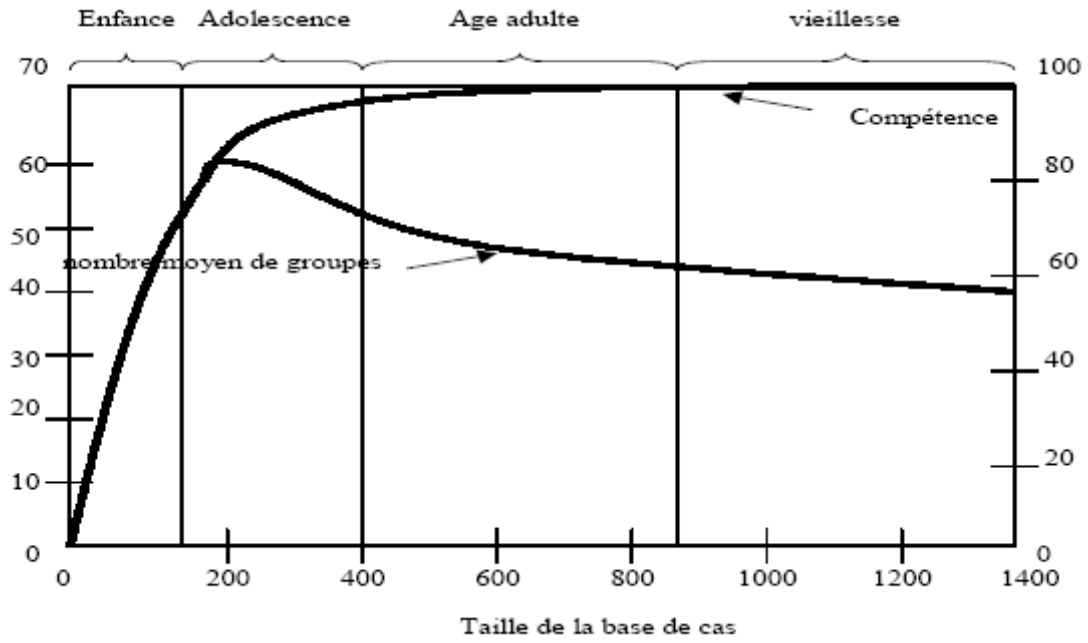


Figure 3.6. Exemple de l'évolution de la base de cas par les quatre phases dans le domaine de planification des voyages [Mckenna et Smyth, 1999]

La Figure 3.6 montre deux courbes : la première (qui augmente puis stagne) représente le nombre moyen de groupes formés. Il s'agit des groupes de compétence qui représentent un ensemble de cas ayant une contribution collective à la compétence de la base de cas. Un groupe de compétence est un ensemble maximal de cas ayant des recouvrements partagés. Tandis que la deuxième courbe (qui augmente et ensuite redescend), représente la compétence de la base de cas

Dans cette étude, il est constaté que l'ajout successif des cas dans la base de cas fait augmenter sa compétence jusqu'à une certaine limite. Une fois que cette limite est atteinte, la compétence stagne tout en diminuant le nombre moyen de groupes de compétence. En effet, la compétence augmente jusqu'à l'âge adulte par l'ajout des cas dans la base de cas. A ce niveau, la compétence maximale est atteinte. Durant et après cette phase (vieillesse), l'ajout d'un nouveau cas se répercute négativement sur le temps de recherche qui ne fera qu'augmenter sans pour autant élever la compétence.

- ***Méthodes maximisant le critère de performance***

Leake et Wilson [2000a] ont défini une mesure de *performance relative RP* (*Relative Performance*) inspirée de la mesure (RC) à partir des travaux de Smyth et McKenna [1999b]. La mesure RP est basée sur le coût d'adaptation de chaque cas de la base de cas.

$$RP(c) = \sum_{c' \in Ensemble\,\mathrm{Re}couvrement(c)} 1 - \frac{CoûtAdapt(c,c')}{\max_{c'' \in ER(c',c)} CoûtAdapt(c'',c')} \qquad (3)$$

En supposant qu'un cas « c » résout un problème « p_1 », le coût de l'adaptation de « c » pour résoudre un nouveau problème « p_2 » du cas « c' » est :

$$\text{CoûtAdapt(c, c')} = \alpha\,|p_1 - p_2| \text{ avec un coefficient } \alpha \text{ fixe et } \alpha > 0 \qquad (4)$$

La valeur de la performance relative d'un cas « RP(c) » reflète sa contribution à la performance d'adaptation comparée à celle des autres cas. Avant d'ajouter un cas dans la base de cas réduite, sa contribution à la performance d'adaptation est estimée. Un ensemble de mesures incluant la mesure du bénéfice de performance produit par l'ajout de chaque cas est estimé. De ce fait, cette mesure utilisée pour guider l'ajout de cas en favorisant les cas ayant une grande valeur de « RP ».

Dans le même cadre, une autre mesure a été développée dans [Leake & Wilson, 2000b] concernant le *bénéfice de la performance relative PB* (*relative Performance Benefit*) qui est basée sur les mêmes notions que la mesure « RP ». Cette mesure de bénéfice donne les mêmes performances, c'est à dire le même taux de réduction de la taille de la base de cas.

$$PerBenefit(c) = \sum_{c' \in Ensemble\,\mathrm{Re}couvrement(c)} \max(0, \min_{c'' \in Atteignabilité(c',c)} (CoûtAdapt(c'',c') - CoûtAdapt(c,c'))) \quad (5)$$

$$PB(c) = \frac{PerBenefit(c)}{\max_{c \in BC} PerBenefit(c)} \qquad (6)$$

Ces mesures sont associées à l'algorithme CNN (Condenced Nearest Neighbor) afin de produire une méthode hybride efficace. En effet, cet algorithme est considéré comme étant la première technique de réduction de la taille d'une base de référence qui prend appui sur des considérations d'ordre statique [Dasarathy, 1991 ; Hart, 1967]. L'algorithme vise à réduire l'ensemble de données dans l'espace d'entrée à un sous-espace représentatif ayant les mêmes propriétés. Les mesures RC, RP et PB sont appliquées afin d'organiser les cas dans la base de cas initiale. Cette organisation est faite dans le but de commencer à choisir les cas qui possèdent une grande valeur de la mesure concernée, qui seront les plus représentatifs, et de les mettre en premier dans la base de cas réduite.

Cependant, la technique ***RC-CNN*** fournit un taux de réduction de la taille de la base de cas supérieur à celui des techniques ***RP-CNN*** et ***PB-CNN***. Cependant, ces deux dernières donnent un résultat nettement meilleur par rapport à ***RC-CNN***, concernant le coût d'adaptation des cas dans la base de cas obtenue.

Dans le même registre, nous pouvons citer la méthode ***NUN-CNN*** (Nearest Unlike Neighbor-CNN) qui prend appui sur l'idée concernant les cas appartenant à des classes différentes. Ces cas sont proches seulement s'ils sont à la limite des frontières des classes respectives. Pour une base de cas à Nc classes, les (Nc - 1) NUN (un par classe) de chaque cas sont identifiés. Par la suite, les NUN de tous les cas de la base initiale sont réunis. Des sous-ensembles NUN sont ainsi crées, qui peuvent être vu comme des représentants optimaux des frontières entre les classes des cas de la base de cas.

- ***Méthodes basée sur les règles d'adaptation floue***

La stratégie basée sur des règles d'adaptation floue est une des approches qui a été mise en place dans un contexte distribué. Cao et al. [2003] se sont intéressés aux différentes méthodes de maintenance de la base de cas concernant des clients sur sites Web. Ils ont proposé une méthodologie de sélection de cas. Cette technique revient à réduire une base de cas à un sous-ensemble représentatif ayant les mêmes propriétés, en remplissant une base de cas initialement vierge à partir d'une base de cas original. Ceci, grâce à une étude qui porte sur les travaux de [Smyth & Keane, 1995] concernant les différentes classes de cas (pivots, support, auxiliaire et couverture). Aussi, Cao et al. [2003] se sont basés sur les travaux de [Shiu et al., 2001] qui utilisent la méthode d'induction par arbre de décision floue. Ensuite, ils ont introduit des règles d'adaptation floue car c'est une méthode qui réduit considérablement le temps de comparaison de tous les cas de la base de cas et offre une solution efficace. Cette méthode est constituée de quatre phases : la première porte sur l'apprentissage des poids des cas, la deuxième sur la division de la base de cas en plusieurs groupes, la troisième utilise des règles d'adaptations floues liées à la représentation des cas et des groupes créés, et la quatrième concerne le choix des cas à partir de chaque groupe basé sur les concepts de recouvrement et d'atteignabilité.

Cette méthode est inspirée des méthodes existantes dans la stratégie Backward Elimination (BE) qui va être abordée dans la section suivante.

La stratégie Backward Elimination (BE)

La stratégie (BE) consiste, à l'inverse de la stratégie FS, à la constitution d'un ensemble de cas par élimination à partir de l'ensemble complet des cas jusqu'à ce que l'ensemble soit vide. La stratégie (BE) enlève à chaque étape un cas dont la suppression donne le meilleur sous-ensemble selon un critère particulier. Lorsqu'un cas est retiré, il ne peut être réintroduit dans l'ensemble des cas sélectionnés. Ci-dessous, la description de l'algorithme général (cf. Figure 3.7) [Lereno, 2000].

entrée S ensemble d'instances initiales
U mesure
C critère d'arrêt

Backward Elimination (BE)

tant que S satisfait U ***et*** ne satisfait pas C

répéter
Chercher dans S l'objet le moins pertinent ω_{moins}
S = S - {ω_{moins}}

sortie S ensemble d'instances sélectionnées

Figure 3.7. Algorithme général de Backward Elimination (BE)

Cet algorithme (BE) démarre sa recherche avec l'ensemble complet « S » des cas initiaux. Il cherche le cas le moins pertinent ω_{moins} selon un critère puis le retire de l'ensemble « S » et l'ajoute à « S' ». Le processus s'arrête lorsque l'ensemble des cas de « S » est vide ou lorsque le critère *C* est satisfait. Contrairement à la stratégie (FS), nous ne pouvons pas, avec ce type d'algorithme, obtenir une liste ordonnée de cas les moins pertinents.

En raison de la nature séquentielle des ajouts ou des suppressions des cas, le sous-ensemble minimum de cas générés par l'une ou l'autre des méthodes n'est pas, par conséquent, le sous-ensemble optimal.

Cette stratégie consiste donc, à partir d'une base de cas entière, à évaluer les cas suivant un critère afin de pouvoir les supprimer et ramener la base de cas à un nombre de cas donné formant une base de cas réduite. Les critères d'évaluation tels que la compétence, la performance, la redondance et l'inconsistance ont été utilisés dans différentes méthodes.

Nous nous intéressons particulièrement dans notre étude aux méthodes de suppression d'une part par balayage de la base de cas et d'autre part à partir de la catégorisation des cas.

- ***Méthodes de suppression par balayage de la base de cas***

Dans cette situation, la base de cas est balayée entièrement dès que sa taille atteint un seuil prédéfini. Il existe plusieurs stratégies balayant la base de cas entière, les plus importantes sont :

- *Suppression aléatoire [Markovitch & Scott, 1988] :* Cette méthode consiste à supprimer aléatoirement un cas de la base de cas dès que sa taille dépasse un certain seuil. L'avantage de cette méthode réside en sa simplicité. Par contre, elle peut avoir des conséquences catastrophiques sur la compétence de la base de cas, car naturellement, elle peut supprimer des cas d'une grande importance présentant un critère maximisé. Cette méthode sera un point de référence pour la comparaison avec les autres méthodes.

- *Ironically [Minton, 1990] :* Cette méthode s'apparenterait à un facteur d'oubli de la connaissance la moins utilisée en calculant la fréquence de remémoration de chaque cas dans la base de cas. Par conséquent, les cas qui ont une fréquence de remémoration faible seront supprimés de la base de cas. Cette méthode est donc simple à mettre en place, mais comme la précédente, elle a des conséquences irréversibles quant à la détérioration de la qualité de la base de cas.

- *Suppression en se basant sur la mesure d'utilité (Utility Deletion : UD) [Minton, 1990] :* Lorsque la base de cas atteint un seuil limite prédéfini, cette stratégie supprime les cas qui ont une valeur de *l'utilité négative*. L*'utilité* est définie par la différence entre la taille de la base de cas, la qualité de la solution et l'efficacité associée aux grandes bases de cas. L'efficacité du système est mesurée par le temps moyen mis pour résoudre un problème cible. La réduction du temps de la solution correspond à l'augmentation de l'efficacité. La qualité de la solution est liée au pourcentage de bonnes réponses fourni par le système. Elle augmente avec la taille de la base de cas [Smyth et al., 1996].

- *Suppression basée sur la redondance et l'inconsistance [Racine & Yang, 1997] :* Cette méthode est destinée à la maintenance des systèmes qui possèdent de grandes bases de cas non structurées ou semi-structurées. Ces bases de cas

sont construites et mises à jour par différents utilisateurs à partir de sources variées. Ce qui conduit à une base de cas contenant des cas redondants et inconsistants. Cette méthode est dotée de deux modules de détection, l'un de redondance et l'autre de l'inconsistance qui interagissent avec l'utilisateur. Suivant une série de test de chaque cas de la base de cas par ces deux modules, ceux-ci sont supprimés ou non de la base de cas avec l'approbation de l'utilisateur. En effet, cette méthode consiste à prendre chaque cas de la base de cas originale et de le faire passer en premier lieu par le premier module de test de redondance. S'il n'y a pas de redondance, un deuxième test est fait par le module d'inconsistance. Si le cas passe avec succès ces deux tests, il sera ajouté à la base de cas. Sinon pour l'un des deux tests, une alerte est générée et le cas sera présenté à l'utilisateur. C'est à ce dernier de choisir d'ignorer l'alerte ou de le supprimer. Ces deux modules qui sont rajoutés ainsi que les tests sont illustrés sur la Figure 3.8.

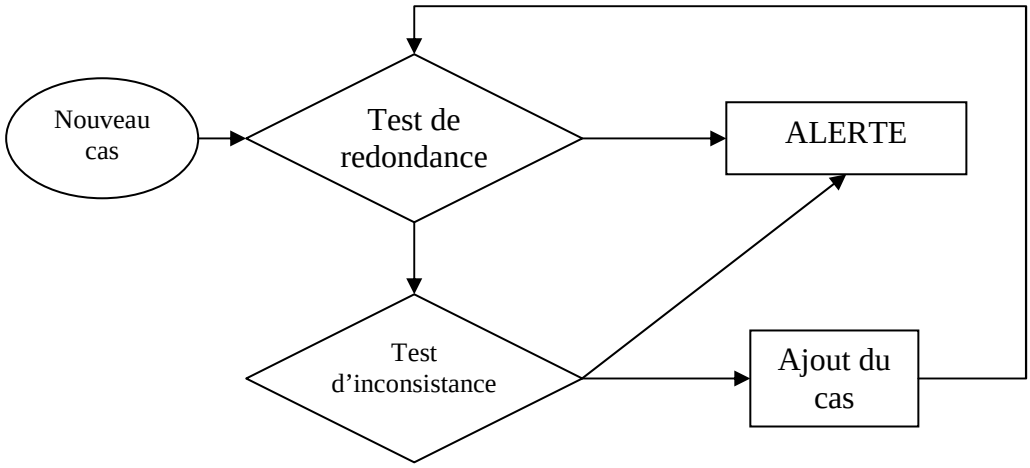


Figure 3.8. Schéma des modules des tests de redondance et d'inconsistance pour la construction de la base de cas [Racine et al., 1997]

- *Suppression basée sur la taille et la densité de la base de cas [Smyth & McKenna, 1998] :* Les auteurs ont proposé une méthode qui permet d'étudier la taille de la base de cas, la densité et la distribution des cas. Cette méthode essaye de garder l'homogénéité de la densité des cas dans la base de cas.

  La densité de la base de cas est l'ensemble de densité des cas dans la base de cas. Nous pouvons dire qu'un groupe est dense, lorsque la similarité entre les cas est grande dans la résolution de problème dans un groupe de cas défini. La

contribution individuelle d'un cas appartenant à un groupe dense est inférieure à celle d'un cas appartenant à un groupe non dense. La densité d'un cas « c » dans un groupe de cas *G* est donnée par la formule suivante :

$$densité(c,G)=\frac{\sum_{c'\in G-c\}} Similarité(c,c')}{|G|-1}$$

La distribution de la base de cas est la façon dont les cas sont organisés dans la base de cas. Une base de cas mal distribuée est une base de cas qui aura une variation considérable de densité, par conséquent, une irrégularité dans la solution donnée aux problèmes.

Cette étude est faite afin de comprendre et de mesurer la compétence de la base de cas. Dans ce contexte, plusieurs modèles ont été proposés par Smyth et Mckenna dans [Smyth & McKenna, 1998], [Smyth & McKenna, 1999b], [Smyth & McKenna, 2002a] et [Smyth & McKenna, 2002b]. Ces modèles reposent sur deux hypothèses principales :

- les cas de la base de cas correspondent à un échantillon représentatif des problèmes cibles. Par conséquent, cette hypothèse permet d'étudier les cas dans la base de cas à travers un ensemble de taille moindre de ces mêmes cas. Ceci est effectué pour avoir le recouvrement et l'atteignabilité des cas ;
- l'espace des problèmes est régulier, dans le sens où les problèmes similaires ont des solutions similaires.

- *Iterative Case Filtering Algorithm (ICF):* Cette méthode consiste à supprimer les cas, d'une façon itérative, jusqu'à obtention d'un un bon résultat global. En effet, les auteurs utilisent des règles répétitives de suppression qui consistent à éliminer les cas dont la taille d'atteignabilité est plus grande que celle de recouvrement, jusqu'à ce que les conditions de la règle appliquée soient satisfaisantes [Brighton & Mellish, 2002].

Nous avons constaté que la majorité de ces méthodes ne donnent pas de résultats satisfaisants pour l'optimisation de la taille de la base de cas selon le critère étudié. De plus, il y a des méthodes qui sont très difficile à implémenter. Quant à celles qui sont facile à mettre en place, elles fournissent des résultats peu convaincants.

Par ailleurs, d'autres travaux ont été consacrés à la catégorisation des cas de la base de cas dans qui vont être abordés dans ce qui suit.

- ***Méthodes de suppression à partir de catégorisation des cas***

Ces méthodes font appel à une modélisation de la compétence de la base de cas. Cette modélisation a été proposée par Smyth et Keane dans [Smyth & Keane, 1995b], [Smyth & McKenna, 1998], [Smyth & McKenna, 1999b], [Smyth & McKenna, 2002a], [Smyth & McKenna, 2002b]. Une catégorisation de cas dans la base de cas est ainsi créée selon leurs compétences. Cette catégorisation prend appui sur deux notions importantes qui sont le *recouvrement* (*coverage*) et *l'atteignabilité* (*reachability*).

Soit une base de cas : BC = {c1,…, cn}, et c© : l'ensemble des cas cibles dans la base de cas. Formellement :

- *Le recouvrement* : pour c ∈ BC, Recouvrement(c) = {c©∈ BC : adaptable(c, c©)} ;
- *L'atteignabilité* : pour c ∈ BC, Atteignabilité(c) = {c©∈ BC : adaptable(c©, c)}.

A partir de ces deux notions et par rapport aux cas de la base de cas, quatre classes de cas ont été ainsi créées [Smyth & Keane, 1995b] :

- *Cas pivot (pivotal case) :* un cas est considéré comme pivot si seulement si son ensemble d'atteignabilité est réduit à un singleton (le cas lui-même). Son espace de recouvrement n'est atteignable par aucun autre cas. Sa suppression réduit directement la compétence du système.

Pivot(c) *ssi* Atteignabilité(c) – {c} = Ø

- *Cas de support (support case) :* un cas est considéré comme un cas de support s'il existe dans son ensemble d'atteignabilité des cas qui ont le même recouvrement. Ces cas existent en plusieurs groupes. La suppression d'un sous ensemble de cas propre à ce groupe n'affecte pas la compétence. Par contre la suppression de tout le groupe réduit considérablement la compétence de la base de cas (comme si nous supprimions un cas pivot). Cependant, lorsque nous laissons un seul cas (celui qui possède le plus grand recouvrement) dans le groupe des cas de support, nous obtenons un cas pivot.

Support(c) *ssi*c©∈ Atteignabilité(c)-{c} : Recouvrement(c©)∈ Recouvrement(c)

- *Cas de couverture (spanning case) :* un cas est considéré de couverture si son espace de recouvrement se trouve à l'intersection des régions dans les espaces de recouvrement des cas au sein de son ensemble d'atteignabilité. Ce cas n'affecte pas directement la compétence du système.

Couverture(c) *ssi* Pivot(c) $\wedge$ Recouvrement(c)$\cap U_{c© \in \text{ Atteignabilité(c)-\{c\} Couverture(c)}} \neq Ø$

- *Cas auxiliaire (auxiliary case) :* un cas est considéré comme auxiliaire si son espace de recouvrement est subsumé par l'espace de recouvrement d'un autre cas. Cette catégorie de cas n'affecte pas du tout la compétence du système. De ce fait, leur suppression totale est possible.

Auxiliaire(c) *ssi* $\exists$ c© $\in$ Atteignabilité(c) – {c} : Recouvrement(c) $\not\subset$ Recouvrement (c©)

Les méthodes développées dans ce cadre génèrent un ensemble de cas cibles à partir des cas sources afin de catégoriser la base de cas.

L'illustration des quatre classes de cas proposées est montrée ci-dessous (cf. Figure 3.9).

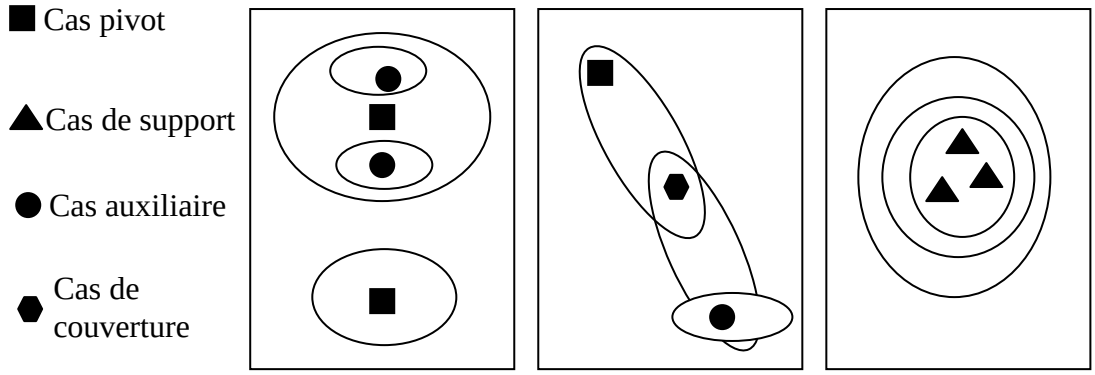


Figure 3.9. Représentation des quatre classes de cas suivant leur espace de recouvrement et d'atteignabilité [Smyth et Keane, 1995]

Nous recensons deux méthodes de suppressions de cas, issues de cette technique de catégorisation [Smyth & Keane, 1995b] :

- *Méthode de suppression de traces (footprint deletion : FD) :* Cette méthode de suppression de cas vise à maximiser la compétence en minimisant la taille de la base de cas en la guidant vers une configuration optimale. Cette base de cas

optimale est appelée *trace de compétence*. Cette stratégie s'appuie sur la catégorisation des cas de la base de cas pour pouvoir supprimer tous les cas dits auxiliaires, de couverture, et de ne laisser qu'un cas de support dans chaque groupe de support et de garder tous les cas pivots. Cette stratégie préserve la compétence mais ne s'occupe pas du problème qui peut se poser au niveau de la performance. Pour pallier cela, les auteurs ont proposé une méthode regroupant les deux critères que nous allons aborder à la section suivante.

- *Méthode de suppression trace-utilité (footprint-utility deletion : FUD) :* Cette méthode améliore la précédente en tenant compte du critère de performance. Ce critère est associé aux différents cas qu'il faut garder lorsqu'il y a deux ou plusieurs cas qui ont la même compétence et qu'il faut en supprimer quelques uns. Elle s'occupe donc conjointement de la compétence et de la performance. Cette méthode fusionne comme son nom l'indique la méthode « *FD* » abordée à la précédente section et la valeur de l'*utilité* qui reflète la performance de la base de cas.

La politique d'optimisation de la base de cas présente plusieurs avantages tels que la réduction de la taille de la base de cas et sa structuration. Cette politique permet d'obtenir de bons résultats quant à l'efficacité de la solution donnée au problème rencontré et à la rapidité du temps de réponse. Elle permet également la combinaison de diverses métriques et de méthodes pour pouvoir trouver un bon compromis « taille – qualité de la base de cas ». Toutefois, la suppression de cas peut engendrer une perte d'information malgré le respect des critères étudiés.

Nous nous intéressons spécialement à cette politique et notamment à cette technique de catégorisation de cas que nous exploitons en partie dans notre proposition de méthode de maintenance de la base de cas. En effet, cette catégorisation est intéressante car une fois mise en place, elle facilite la suppression des cas qui ne contribuent pas à la compétence de la base de cas. Mais cette technique n'est pas suffisante car cette catégorisation peut être coûteuse en temps de calcul lors du balayage de toute la base de cas. Par conséquent, il faut trouver un moyen plus efficace qui permet de faciliter et fluidifier cette catégorisation. Nous discutons la solution suggérée lors de la proposition de notre méthode.

Par ailleurs, nous analysons les travaux de la politique d'optimisation de la base de cas à travers deux tableaux dans la section suivante.

2.5. Synthèse de la politique d'optimisation de la base de cas

Nous nous somme intéressés dans notre étude à la politique d'optimisation de la base de cas car elle permet de réduire sa taille en gardant uniquement les cas qui préservent la qualité de la base de cas suivant le/les critères étudiés. Pour cela, de nombreuses stratégies ont été élaborées pour répondre à ces attentes. Ces stratégies présentent néanmoins quelques limites et contraintes que nous illustrons dans le tableau 3.1. Par ailleurs, nous avons exploité dans notre étude les stratégies utilisant les critères de compétence et de performance car se sont les plus répandues. De plus, nous avons constaté qu'il y avait plusieurs pistes à explorer.

Ainsi, nous mettons en place deux tableaux récapitulatifs illustrant les stratégies étudiées dans le cadre de la politique d'optimisation de la base de cas. Le tableau 3.1 décrit les caractéristiques des différentes méthodes utilisées dans la stratégie (BE) ainsi que la comparaison entre elles selon les points suivants : mesures ou formules utilisées, critères étudiés, critère d'arrêt, échantillon de travail, avantages et inconvénients.

Nous constatons qu'il n'y a aucune mesure ni formule utilisée dans les cinq méthodes présentées dans le tableau 3.1. Concernant les critères utilisés, les quatre méthodes RD, UD, FD et Ironically utilisent un unique critère (compétence pour RD, FD et Ironically ou performance pour UD). Seule la méthode FUD combine les critères de compétence et de performance. Cependant, la performance est prise en compte uniquement lorsqu'il y a deux cas, ayant la même compétence, à supprimer. De plus, il n'y a pas d'évaluation de ce critère de performance.

L'échantillon de travail qui concerne les cas cible est exploité pour évaluer la qualité de la base de cas selon le critère étudié. Cet échantillon de cas représente toute la base de cas pour les méthodes RD, UD et Ironically, tandis que pour les deux autres méthodes FD et FUD, l'échantillon est représenté uniquement par un sous-ensemble sélectionné aléatoirement à partir de la base de cas. Dans le premier cas, l'étude est plus intéressante car c'est toute la base de cas qui est balayée. Cependant, le temps de calcul est plus important que le deuxième cas.

Politique	**Optimisation de la base de cas**				
Stratégie	**Backward Elimination (BE)**				
Méthode	**RD (Random Deletion)**	**UD (Utility Deletion)**	**FD (Footprint Deletion)**	**Ironically**	**FUD (Footprint-Utility Deletion)**
Mesure (formule)	Aucune	Aucune	Aucune	Aucune	Aucune
Critère utilisé	Compétence	Performance	Compétence	Compétence	Compétence + Performance
Critère d'arrêt	Nombre de cas dans la base de cas = seuil prédéfini	Valeur de l'utilité de chaque cas de la base de cas = valeur positive	Nombre de cas auxiliaires = 0 Nombre de cas de couvertures = 0 Nombre de cas de supports dans un groupe de support = 1	Fréquence de remémoration de chaque cas dans la base de cas = seuil prédéfini	Nombre de cas auxiliaires = 0 Nombre de cas de couvertures = 0 Nombre de cas de supports dans un groupe de support = 1
Echantillon de travail	La base de cas complète	La base de cas complète	Sous-ensemble de la base de cas (Aléatoirement)	La base de cas complète	Sous-ensemble de la base de cas (Aléatoirement)
Avantage	Simplicité de la mise en place de la méthode	Bonne performance de la base de cas	Bonne compétence de la base de cas	Bonne compétence de la base de cas	Bonne compétence et bonne performance de la base de cas.
Inconvénient	Peut réduire considérablement la compétence de la base de cas	Très faible compétence de la base de cas	Performance non évaluée	Performance non évaluée	Performance non évaluée

Tableau 3.1. Caractéristiques des stratégies de Backward Elimination (BE)

La méthode RD supprime aléatoirement les cas, donc c'est la méthode la plus simple à mettre en place. Cependant, elle peut diminuer considérablement la compétence de la base de cas car elle peut supprimer les cas importants. La méthode Ironically garde uniquement les cas qui ont servi plusieurs fois. Elle peut donc altérer la compétence de la base de cas en supprimant les cas importants qui ne sont pas souvent remémorés. La méthode FUD est le résultat de la fusion des deux méthodes FD et UD. Elle prend les avantages de la compétence

de FD et de la performance de UD. Cependant, la performance n'est pas évaluée donc nous ne pouvons pas *a priori* tirer des conclusions.

En résumé, les résultats ne sont pas très satisfaisants du point de vue réduction de la taille de la base de cas en fonction de l'optimisation des deux critères.

Ensuite vient le tableau 3.2 qui décrit les caractéristiques des différentes méthodes utilisées dans la stratégie (FS) avec les mêmes critères de comparaison utilisés dans le tableau 3.1.

Politique	**Optimisation de la base de cas**			
Stratégie	**Forward Selection (FS)**			
Méthode	**CNN (Condensed Nearest Neighbor)**	**RC-CNN**	**RP-CNN**	**PB-CNN**
Mesure (formule)	Aucune	RC (Relative Coverage)	RP (Relative Performance)	PB (Performance Benefit)
Critère utilisé	Compétence	Compétence	Performance	Performance
Critère d'arrêt	Tous les cas dans la nouvelle base de cas résolvent les cas dans la base de cas entière	Rapport maximal entre le nombre de cas dans la base de cas et la valeur de la compétence	N(O)-N(R)= Ø N(O) : voisinage de la base de cas originale. N(R) : voisinage de la base de cas réduite.	Rapport maximal entre le nombre de cas dans la base de cas et la valeur de la performance
Echantillon de travail	La base de cas complète	La base de cas complète	La base de cas complète	La base de cas complète
Avantage	- Bonne compétence de la base de cas. - Coût d'adaptation moyen des cas de la base de cas	- Compétence élevée de la base de cas. - Réduction considérable de la taille de la base de cas.	- Faible Coût d'adaptation des cas de la base de cas. - Performance élevée de la base de cas.	
Inconvénient	- La taille de la base de cas n'est pas assez réduite. - Assez faible performance de la base de cas	- Coût d'adaptation des cas de la base de cas assez élevé. - Faible performance de la base de cas	- Compétence de la base de cas non évaluée. - La taille de la base de cas n'est pas assez réduite.	

Tableau 3.2. Caractéristiques des stratégies de Forward Selection (FS)

Nous pouvons identifier qu'aucune stratégie (FS) ne combine plus d'un critère. Cependant, la majorité de ces stratégies utilise des mesures de performance ou de compétence

combinées à l'algorithme CNN. En effet, ces stratégies classent les cas dans la base de cas initiale selon la valeur de la mesure utilisée (RC, RP ou PB), ensuite commence le processus d'ajout de cas dans une base de cas initialement vierge selon l'algorithme CNN. Les deux stratégies RP-CNN et PB-CNN sont très proches du point de vue « résultats » concernant la performance de la base de cas. Ces deux dernières stratégies s'intéressent uniquement à la performance. Tandis que la stratégie RC-CNN ne s'intéresse qu'à la compétence de la base de cas. En effet, cette dernière a pour objectif de réduire la taille de la base de cas tout en maintenant sa compétence.

En résumé, les stratégies FS traitent la qualité de la base de cas en exploitant qu'un seul critère. Le critère utilisé est optimisé en fonction de la réduction de la taille de la base de cas. La majorité de ces stratégies (RC, RP, PB) combine l'algorithme CNN à une mesure de compétence ou de performance.

Après avoir pris connaissance de toutes ces politiques et stratégies et après avoir fait une analyse, nous proposons une méthode visant à combiner les meilleures stratégies des deux types (BE et FS) afin d'obtenir une taille optimale de la base de cas tout en maximisant le critère utilisé.

3. Proposition d'une méthode de maintenance de la base de cas

Après l'état de l'art réalisé sur la maintenance de la base de cas, nous nous sommes intéressés particulièrement à la politique d'optimisation de la base de cas en donnant des justificatifs qui sont précédemment énumérés. Nous abordons maintenant la démarche adoptée ainsi que la méthode proposée pour la réalisation de la maintenance de la base de cas.

La démarche de maintenance de la base de cas passe tout d'abord par une méthode de structuration des cas dans la base de cas, ensuite, par une méthode d'auto-incrémentation des cas dans celle-ci.

3.1. Structuration de la base de cas

Afin de mettre en place la méthode de structuration de la base de cas, nous avons adopté une démarche prenant appui sur une catégorisation des cas, une direction de recherche Backward Elimination (BE), une procédure heuristique et aléatoire, un critère de sélection qui est la compétence ainsi qu'un critère d'arrêt.

3.1.1. Démarche suivie

Nous avons suivi une démarche bien précise concernant notre proposition, cette démarche est caractérisée par les points suivants (Figure 3.10) :

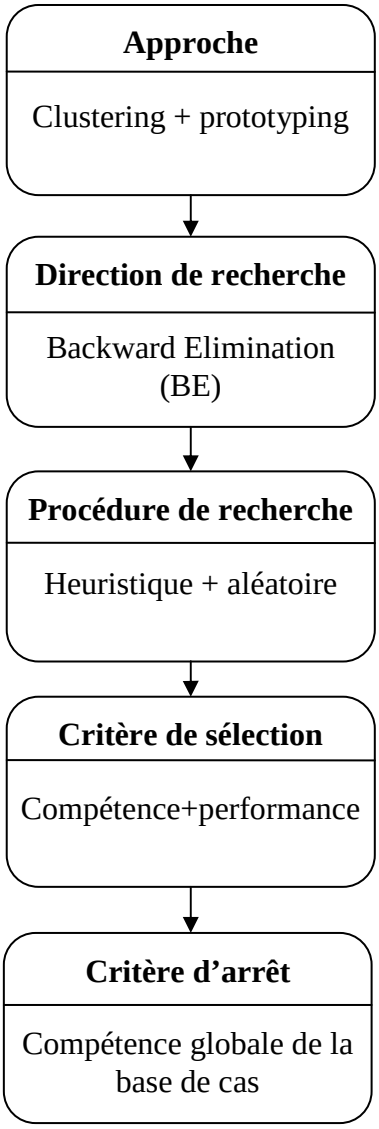


Figure 3.10. Démarche suivie pour la structuration de la base de cas

- ***Approche :*** Clustering + Prototyping (sélection des prototypes)

Notre démarche consiste à regrouper dans la même catégorie les cas jugés similaires (homogénéité intra-classe) suivant le critère de compétence qui prend appui sur les deux notions du recouvrement et de l'atteignabilité. Ensuite, pour certaines catégories, nous sélectionnons un prototype en choisissant un cas qui représentera son groupe. Le cas ou le sous-ensemble de cas choisi sera celui qui possède le plus grand recouvrement par rapport aux

autres cas de son groupe. Ce sous-ensemble recouvrera le même nombre de problème que l'ensemble de la base de cas.

- ***Direction de recherche :*** Backward Elimination (BE)

A partir d'une base de cas complète, nous supprimons des cas pour avoir une base de cas de taille réduite. Après avoir défini les différentes catégories de cas, la suppression se fera suivant un algorithme de catégorisation de cas associé à la valeur de la mesure de compétence mise en place [Haouchine et al., 2007b]. Cette valeur est appelée « *MC* » pour « Mesure de Compétence » que nous abordons par la suite.

- ***Procédure de recherche :*** Heuristique + aléatoire

Pour chaque cas de la base de cas, nous estimons son recouvrement et son atteignabilité sur un sous ensemble de cas tirés aléatoirement. Ce sous-ensemble représentera l'espace problème, c'est-à-dire les cas qui seront potentiellement atteignables par les cas sources. La sélection des cas se fera grâce au critère de compétence que nous voulons optimiser en fonction du nombre de cas restant dans la base de cas.

- ***Critère de sélection :*** Compétence (préservation de la compétence) + performance

L'objectif de cette méthode est d'une part, de préserver la compétence de la base de cas tout en optimisant la taille de la base de cas et d'autre part, d'avoir une bonne performance. En effet, pour avoir une bonne compétence, il faut maximiser le recouvrement des cas et minimiser leur atteignabilité. Concernant la performance, elle est déterminée en fonction de la précision de la base de cas résultante (*accuracy*) et du nombre de cas dans la base de cas (*storage*).

- ***Critère d'arrêt :*** En fonction d'une mesure de compétence globale bien définie

Nous calculons une valeur globale à partir de la mesure *MC* permettant ainsi de déterminer l'arrêt de la suppression. Cette mesure reflète la valeur de la compétence globale sur l'ensemble complet de la base de cas réduite qui est égale à la somme des *MC* de chaque cas dans la base de cas divisée par le nombre de cas présents.

$$CompétenceGlobale(BC) = \frac{\sum_{i=1}^{n} MC(c_i)}{n} \qquad (7)$$

Où *n* : est le nombre de cas dans la base de cas

De ce fait, lorsque la valeur de la compétence globale est égale à « 1 » et que le recouvrement de chaque cas dans la base de cas est unique dans sa propre classe alors la suppression s'arrête.

La mise en place de la méthode est décrite dans la section suivante. En effet, notre méthode s'appuie sur les avantages des méthodes issues des deux stratégies BE et FS.

3.1.2. Mise en place de la structuration de la base de cas

La méthode que nous proposons est basée, d'une part, sur la catégorisation de Smyth et Keane [1995b] qui est issue des stratégies BE, abordée à la section 2.4.2 et d'autre part, sur un algorithme de suppression de cas associé à une mesure de compétence « *MC* » [Haouchine et al., 2006b ; 2007b]. En effet, nous catégorisons les cas de la base grâce à la mesure MC, ensuite, nous supprimons les cas selon un algorithme de maintenance de la base de cas.

Notre mesure est inspirée de la mesure de recouvrement relatif RC dans [Smyth & McKenna, 1999b] utilisée dans la stratégie Forward Selection (FS). Elle estime la contribution individuelle de la compétence d'un cas en fonction de la taille de l'ensemble de recouvrement de ce dernier en attribuant, à chaque recouvrement et atteignabilité d'un cas, une valeur que nous notons : valeur recouvrement « *Vr* » et valeur atteignabilité « *Va* ».

$$MesureCompétence(c) = \frac{Vr(c)}{Va(c)} \qquad (8)$$

Où :

Vr(c) = cardinal de l'ensemble de cas cibles associé au cas source *c*.

Va(c) = cardinal de l'ensemble de cas sources associé au cas cible *c*.

Sachant que pour avoir une base de cas possédant une bonne compétence, il faut que son taux de recouvrement soit élevé et que son taux d'atteignabilité soit faible. En conséquence, la mesure *MC* est utilisée pour guider la suppression des cas dans la base de cas selon le principe suivant : *« Favoriser les cas qui possèdent une grande valeur de « MC » et*

supprimer ceux qui ont une petite valeur ». De ce fait, notre méthode permet de réduire la taille de la base de cas tout en conservant une compétence maximale.

Par ailleurs, notre méthode suit le principe des stratégies BE sachant qu'aucune métrique n'a été développée dans ce type de stratégie. L'objectif est de réduire la taille d'un sous-ensemble ou d'une base de cas tout en préservant les critères de qualité de celle-ci, soit le plus grand nombre possible de problèmes pouvant être résolus (une plus grande compétence) et un bon taux de classification (une bonne performance).

Les valeurs de recouvrement (Vr) et d'atteignabilité (Va) se calculent selon les deux algorithmes suivants. L'algorithme 1 illustre le calcul de ***Vr***.

Algorithme de calcul de « Vr »

Pour tout $cas_i \in$ BC (i=1..n) **faire** // BC : Base de cas
// n : nombre de cas de la base de cas
Pour tout cas_j (j=1..n) $\in$ BC **faire**
Si *Similarité* (cas_i, cas_j) > *Seuil* **alors** // cas_i recouvre cas_j
$Vr_i = Vr_i + 1$ // la valeur du recouvrement s'incrémente
Mémoriser l'index du cas cas_j
FinSi
FinPour
$cas_i \leftarrow (Vr_i$, index $cas_j)$
Fin Pour

L'algorithme de calcul de Vr consiste, dans un premier temps, à former deux groupes de cas. Un premier groupe concerne la base de cas contenant des cas « cas_i » (i=1..n). Le deuxième groupe concerne l'ensemble de cas cible « cas_j » (j=1..n) qui est égal au nombre de cas de la base de cas (hypothèse : un cas source est un cas cible potentiel). Pour chaque « cas_i » de la base de cas, l'algorithme balaye l'ensemble de cas cible « cas_j ». Si la similarité entre un cas_i est un cas_j est supérieur à un seuil prédéfini, alors le cas_i recouvre le cas_j. L'algorithme incrémente un compteur dédié à la valeur de recouvrement du cas_i « Vr_i » et mémorise par la même occasion l'index du cas_j. Ainsi, à la fin du balayage de la base cible pour le premier cas_i, on obtient un couple contenant sa valeur de recouvrement ainsi qu'un ensemble d'index des cas_j qui sont recouvert par le cas_i. L'algorithme répète cette opération « n » fois correspondant au nombre de cas dans la base de cas.

L'algorithme ci-dessous (algorithme 2) illustre le calcul des valeurs d'atteignabilité (Va) de chaque cas de la base de cas.

Algorithme de calcul de « Va »

Pour tout cas_j (j=1..n) ∈ BC **faire** // BC : Base de cas

// n : nombre de cas de la base de cas

Pour tout cas_i (i=1..n)∈ BC **faire** // $cas_i = (Vr_i, \sum_{k=0}^{l} \text{index } cas_k)$, l : nombre de cas recouvert par cas_i

Si $index\_cas_i = index\_cas_k$ **alors** // signifie que cas_j est *atteignable* par cas_i

$Va_j = Va_j + 1$ // la valeur de l'atteignabilité s'incrémente

Mémoriser l'index du cas cas_i

FinSi

FinPour

$cas_j \leftarrow (Va_j, \text{index } cas_j)$

Fin Pour

Cet algorithme part du fait qu'il y a déjà les valeurs de recouvrement des différents cas_i de la base de cas ainsi que les ensembles des index des cas recouverts. Comme dans l'algorithme Vr, deux groupes de cas sont exploités : un groupe de cas de la base de cas et un groupe de cas cible contenant exactement les mêmes cas source. Ensuite, pour chaque cas cible, l'algorithme compare son index avec l'ensemble des index du premier cas_i de la base de cas. S'il y a égalité, alors le cas_j est atteignable par le cas_i. Ainsi de suite pour l'ensemble des cas de la base de cas. Nous obtenons à la fin du premier balayage des cas_i la valeur d'atteignabilité du cas_j ainsi que les index des cas_i qui lui sont atteignable. L'algorithme répète cette opération « n » fois.

En calculant les valeurs de *Vr* et de *Va*, nous pouvons calculer ainsi la valeur de *MC*. Toutes ces valeurs calculées permettent de déterminer les différentes catégories des cas. Le tableau 2.3 illustre les propriétés de la catégorisation des cas [Haouchine et al., 2008a].

Type de cas	**Vr(ci)**	**Va(ci)**	**MC(ci)**
Cas auxiliaire	> 1	= Vr(ci)	1
Groupe de cas de support	> 1	> 1	Mêmes valeurs
Cas de couverture	≥ 1	> 1	≤ 1
Cas pivot	1	1	1

Tableau 2.3. Propriétés des catégories des cas

La valeur de la mesure *MC* est déterminante dans le choix des cas pivots. Les cas pivots doivent être impérativement gardés car leur suppression réduit considérablement et directement la compétence de la base de cas. Ensuite, le cas appartenant au groupe de support et ayant la plus grande valeur de *MC* doit être gardé, tandis que les autres peuvent être supprimés. Ainsi, le cas de support gardé peut être considéré comme un cas pivot. Les cas auxiliaires et les cas de couverture ayant des valeurs de *MC* en-dessous d'un certain seuil de compétence sont tous supprimés. Ces deux dernières catégories sont moins importantes car elles ne contribuent pas directement à la compétence de la base de cas.

La méthode proposée traite les ***cas*** (les valeurs de descripteurs peuvent ne pas être renseignées : valeurs à trous) et les ***instances labélisées*** (les valeurs sont toutes renseignées et les solutions sont représentées par des classes) [Haouchine et al., 2008a]. Cependant, lorsque les instances labélisées sont traitées, la catégorie des cas de couverture est divisée en deux catégories : les *cas de couverture intra-classes* et les *cas de couverture inter-classes*.

Un *cas de couverture intra-classes* est un cas qui est partiellement recouvert par un autre cas appartenant à la même classe.

Un *cas de couverture inter-classes* pour une classe donnée (Classe1 par exemple) est un cas qui est partiellement recouvert par un autre cas appartenant à une autre classe (Classe2 par exemple).

La Figure 3.11 montre un exemple d'un cas pivot ($c_1 \in$ classe2), d'un cas de couverture inter-classes ($c_2 \in$ classe1), d'un cas auxiliaire ($c_3 \in$ classe1), d'un cas de couverture intra-classes ($c_5 \in$ classe3) et d'un groupe de support comportant trois cas de support ($\{c_7, c_8, c_9\} \in$ classe4).

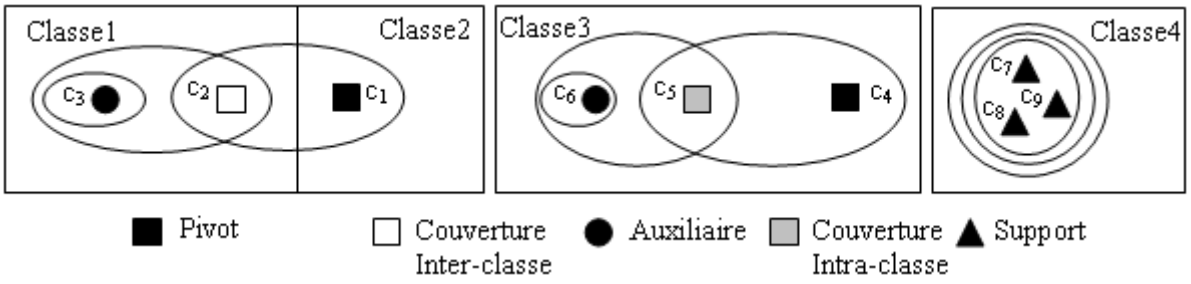


Figure 3.11. Catégorie des cas concernant les instances labellisées

Le cas c_2 appartient à la classe1 et est recouvert par le cas pivot (c_1) qui appartient à la classe2. Le cas c_2 ne doit pas être supprimé parce qu'il contribue à la compétence dans la classe1. Par ailleurs, un seuil de recouvrement est fixé pour la suppression d'un cas de couverture inter-classes, c'est-à-dire, lorsque le recouvrement du cas c_1, par rapport au cas c_2, est supérieur à un seuil prédéfini, alors le cas c_2 est supprimé, sinon il est gardé. Ce seuil est mis en place afin d'éviter la suppression des cas qui ont une grande contribution de recouvrement et qui ont une faible atteignable. Par conséquent, la définition des cas de couverture intra-classes est la suivante :

Soit les cas $\{c_1, c_2\} \in$ même classe,

- *Cas de couverture intra-classe :* Couverture(c_2) ssi $\neg$Pivot(c_2) $\wedge$ Recouvrement(c_2) $\bigcap U_{c1 \in \{(seuil)Atteignabilité(c2)-(c2)\}}$ Recouvrement(c_2) $\neq \emptyset$

De ce fait, les étapes de structuration des cas dans la base de cas sont les suivantes (cf. Figure 3.12) :

- catégorisation de l'ensemble de la base de cas suivant leurs recouvrements et leur l'atteignabilité ;
- sélection des cas prototypes, ce qui revient à sélectionner les cas pivots, les cas de support possédant le plus grand recouvrement dans les groupes de support et les cas de couverture inter-classes qui sont supérieur à un seuil de recouvrement prédéfini.

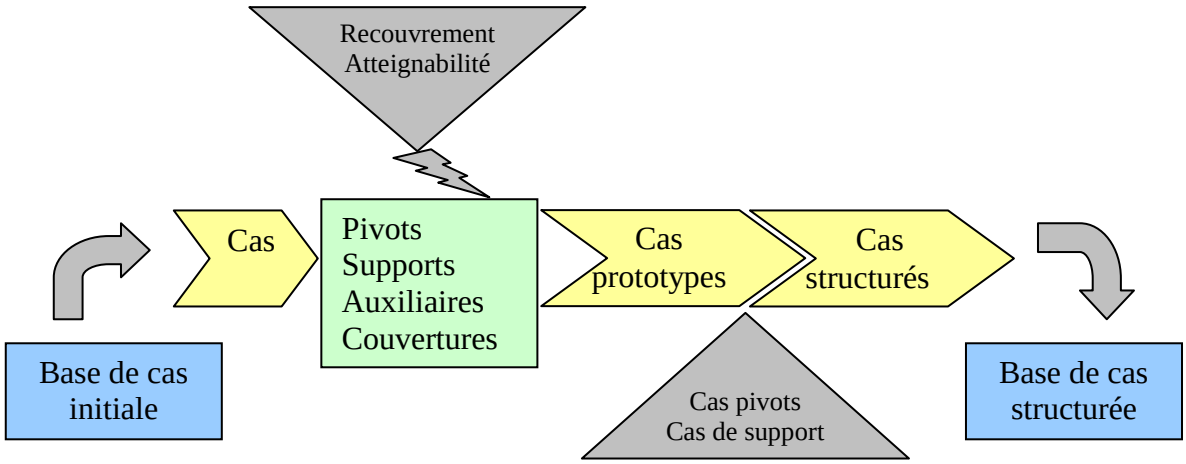


Figure 3.12. Etapes de structuration de la base de cas

Nous utilisons une mesure pour déterminer la fin de la suppression. Cette mesure permet de calculer la compétence globale de la base de cas, en calculant la somme globale des MC de tous les cas de la base de cas normalisée par rapport au nombre de cas existant après la suppression (formule 7).

$$CompétenceGlobale(BC) = \frac{\sum_{i=1}^{n} MC(c_i)}{n}$$

Une base de cas optimale est celle qui a pour valeur de la *CompétenceGlobale(BC)*=1.

L'algorithme de maintenance de la base de cas (algorithme 3) est présenté ci-après. Cet algorithme dépend des valeurs de Vr et Va, de la mesure MC et des catégories des cas [Haouchine et al., 2007a].

Algorithme de la maintenance de la base de cas

---

1. Calculer Vr et Va pour chaque cas de la base de cas.
2. Associer à chaque index du cas son ensemble de recouvrement et d'atteignabilité (index_cas ← (Vr, Va)).
3. Déterminer les catégories des cas.

Si *cas de Support* **alors**

Classifier ces cas selon leurs valeurs de MC de façon croissante.

Pour chaque groupe de support **faire**

Supprimer tous les cas sauf celui qui a la plus grande valeur MC

FinPour

SinonSI *cas Auxiliaire intra-classe* **alors**

Supprimer tous les cas

SinonSi *cas Auxiliaires inter-classe* **alors**

Supprimer tous les cas sauf celui qui possède la plus grande valeur MC

Sauvegarder l'index des cas supprimés

SinonSi *cas Pivot* **alors**

Ne pas supprimer

FinSi

4. **Arrêt** lorsque chaque cas recouvre seulement son propre cas parmi les cas existant dans sa classe et que la valeur de la « CompétenceGlobale(BC) = 1 ».

Ci-dessous, un exemple d'une base de cas contenant quatre cas, illustrant l'espace de recouvrement et d'atteignabilité de ces cas. BC={c_1, c_2, c_3, c_4} (cf. Figure 3.13).

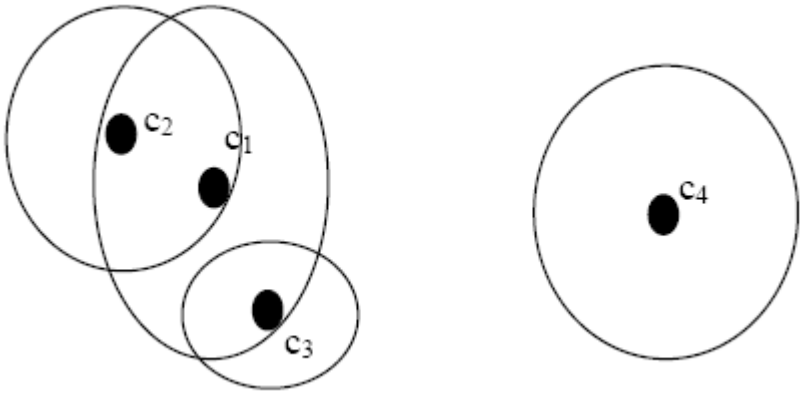


Figure 3.13. Exemple de quatre cas {c_1, c_2, c_3, c_4} avec leur espace de recouvrement et d'atteignabilité.

En calculant les valeurs de Vr et de Va, nous obtenons les résultats suivants [Haouchine et al., 2006a] :

Recouvrement(c_1)={c_1, c_2, c_3} → Vr(c_1)=3, Atteignabilité(c_1)={c_1, c_2} → Va(c_1)=2

Recouvrement(c_2)={c_1, c_2} → Vr(c_2)=2, Atteignabilité(c_2)={c_1, c_2} → Va(c_2)=2

Recouvrement(c_3)={c_3} → Vr(c_3)=1, Atteignabilité(c_3)={c_1, c_3} → Va(c_3)=2

Recouvrement(c_4)={c_4} → Vr(c_4)=1, Atteignabilité(c_4)={c_4} → Va(c_4)=1

Maintenant, nous allons calculer pour chaque cas la valeur MC selon les valeurs Vr et Va, les résultats des calculs sont les suivants :

$MC(c_1) = 1.5$, $MC(c_2) = 1$, $MC(c_3) = 0.5$, $MC(c_4) = 1$.

La détermination des catégories des cas est la suivante :

- $Vr(c_4) = Va(c_4) = MC(c_4) = 1$. Donc, le cas c_4 est un cas pivot ;
- $Vr(c_2) > 1$ (car $Vr(c_2) = 2$), $Vr(c_2) = Va(c_2)$ et $MC(c_2) = 1$. Donc, le cas c_2 est un cas auxiliaire. De ce fait, nous le supprimons ;
- Nous constatons que $MC(c_1) > MC(c_3)$, donc nous supprimons le cas c_3.

Nous obtenons donc la suppression de deux cas (c_2 et c_3) et une base de cas contenant les cas (c_1 et c_4).

En recalculant la valeur de MC pour chaque cas, nous trouvons : $MC(c_1) = MC(c_4) = 1$ avec $Va(c_1) = Va(c_4) = 1$ et $Vr(c_1) = Vr(c_4) = 1$.

Par conséquent, la valeur de la « CompétenceGlobale(BC)=1 ». Ainsi, nous obtenons une base de cas réduite possédant le moins de cas possibles (c'est-à-dire deux : c_1 et c_4) avec un recouvrement maximum.

Cette première partie de structuration de la base de cas permet de catégoriser les cas selon leur recouvrement et leur atteignabilité. L'objectif est de retenir uniquement les cas qui présentent un grand espace de recouvrement et une faible atteignabilité. Nous réalisons cela afin d'avoir une compétence maximal de la base de cas. Par conséquent, nous favorisons les cas pivots ensuite les cas de support qui possèdent le plus grand recouvrement dans leurs groupes de support respectifs. Ceci est dans le but de ne garder que le minimum de cas avec un maximum de possibilité de résolution de problèmes. Notre structuration est guidée par la mesure MC qui permet d'expliciter les quatre catégories de cas. Enfin, cette structuration nous permet d'avoir une base de cas organisée avec peu de cas. De ce fait, la phase de recherche des cas est plus rapide donc une meilleure performance de la base de cas. De plus, le pouvoir de résolution de la base de cas réduite est quasiment pareil que celui de la base de cas initiale. Par conséquent, la compétence de la base de cas est conservée.

Après avoir abordé la méthode de structuration de la base de cas, nous allons, maintenant, mettre en place une méthode de maintien de cette structuration avec l'arrivée dynamique des cas. En effet, les systèmes de RàPC sont amenés à apprendre les cas au fur et à mesure de leur fonctionnement. Cet apprentissage risque de détériorer la qualité de la base de cas même après une phase de maintenance (structuration). De ce fait, il est impératif de mettre

en place une méthode d'auto-incrémentation de la base de cas tout en préservant sa structure et sa qualité. Nous traitons l'auto-incrémentation à la prochaine section.

3.2. Auto-incrémentation de la base de cas

Une méthode de maintenance de la base de cas n'est efficace que si elle est remise à jour selon une fréquence prédéfinie. Cette mise à jour devient indispensable puisque n'importe quel système est amené à évoluer dans le temps. De ce fait, nous avons associé à la méthode de structuration de la base de cas une méthode d'auto-incrémentation des cas permettant l'évolution de celle-ci tout en respectant sa structuration. En effet, cette méthode est complémentaire à la méthode de structuration donnant ainsi une méthode complète de maintenance de la base de cas.

La méthode d'auto-incrémentation est une phase d'apprentissage dynamique des cas dans la base de cas. Elle a été mise en place dans [Haouchine et al., 2009]. L'apprentissage des cas se fait d'une manière incrémentale car les cas sont introduits automatiquement dans la base de cas suivant certaines conditions. Ces conditions sont en relation directe avec la qualité de la base de cas. Comme nous avons pu le voir précédemment, cette qualité est fonction de deux critères, à savoir : la *compétence* et la *performance*. La compétence prend appui sur les deux notions de recouvrement et d'atteignabilité.

Nous avons vu que chaque cas est caractérisé par les deux valeurs « Vr » et « Va ». Toutefois, nous dissocierons dans ces deux notions la partie problème de la partie solution :

- « Vrp » représente le cardinal de recouvrement de la partie problème ;
- « Vrs » représente le cardinal de recouvrement de la partie solution.

Le même raisonnement est appliqué à l'atteignabilité.

- « Vap » représente le cardinal d'atteignabilité de la partie problème ;
- « Vas » représente le cardinal d'atteignabilité de la partie solution.

De ce fait, les deux valeurs « Vr » et « Va » sont définies comme suit :

$$Vr = Vrp + Vrs \quad (9)$$

$$Va = Vap + Vas \quad (10)$$

En résumé, dans un premier temps, une bonne qualité d'une base de cas dépend d'une bonne performance, c'est-à-dire un bon taux de précision avec le moins de cas possible. Dans

un deuxième temps, une base de cas qui a une bonne qualité possède également une bonne compétence, autrement dit, c'est une base de cas qui dispose d'un taux de recouvrement élevé et d'un taux d'atteignabilité faible. Une bonne compétence signifie que le nombre de problèmes pour lesquels la base de cas fournit une bonne solution est élevé.

Par ailleurs, nous définissons une valeur de recouvrement moyen de la base de cas, notée Vr_{BC}, représentée par la formule suivante [Chebel-Morello et al, 2007] :

$$Vr_{BC} = \frac{\sum_{i=0}^{n} Vri}{n} \qquad (11)$$

Où : ***n*** représentant le nombre de cas dans la base de cas.

La valeur « Vr_{BC} » reflète le taux de recouvrement moyen de l'ensemble des cas de la base de cas qui nous servira de repère pour introduire un cas cible adapté. De ce fait, pour qu'un cas cible, qui a été résolu (sa partie solution est renseigné), soit introduit dans la base de cas, il faut que la solution proposée de ce denier n'existe pas dans la base de cas, ce qui signifie que sa partie solution n'est atteignable par aucun cas. Par contre, si la solution existe (sa partie solution est atteignable), il faut que la partie problème soit atteignable par un nombre de cas inférieur au taux de recouvrement moyen de la base de cas « Vr_{BC} » [Chebel-Morello et al, 2007]. Cette dernière condition assure que le cas introduit contribuera à la compétence de la base de cas car son taux d'atteignabilité sera relativement faible.

Nous noterons le cas cible qui a eu une solution mais qui n'est pas encore introduit dans la base de cas par ***« cible* »***, avec : cible* = {problème*, solution*}.

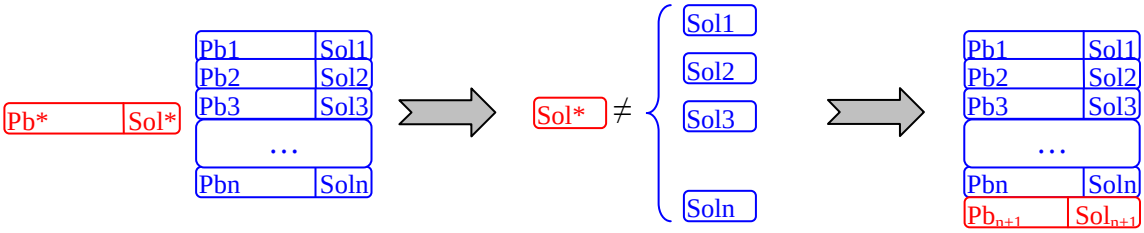
Nous pouvons résumer les étapes de la méthode d'auto-incrémentation des cas dans la base de cas comme suit :

- cas cible avec sa partie solution renseignée → cible* ;

Deux possibilités sont ainsi considérées :

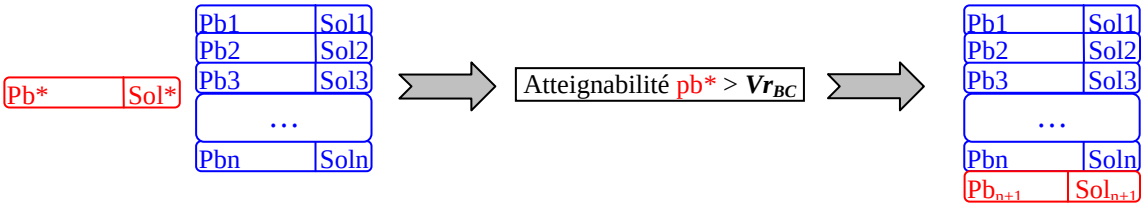
La première est la suivante :

- il n'y a aucune solution dans la base de cas qui est similaire à la solution attribuée au cas cible* (l'espace solution cible* n'est recouvert par aucun espace solution des cas sources) ;
- mémorisation du cas dans la base de cas.



La deuxième est la suivante :

- il y a au moins une solution dans la base de cas qui est similaire à la solution donnée au cas cible* (l'espace solution de cible* est recouvert par un des espaces solution des cas sources) ;
- le recouvrement de l'espace problème du cas cible* qui est atteignable par les espaces de recouvrement des cas source doit être inférieur au recouvrement moyen de la base de cas (l'espace de recouvrement du cas cible* est important, donc sa mémorisation augmentera le taux de recouvrement moyen de la base de « Vr_{BC} »).



L'algorithme d'apprentissage incrémental qui en découle est le suivant :

Algorithme d'apprentissage incrémental

Soit une Base de Cas « BC »

Cas *cible** ← cas cible // Changement de statut du cas cible en lui associant une solution

Pour chaque cas *cible** **faire**

Si Vas > 0 **alors** // La solution du cas résolu est atteignable par d'autres solutions des cas sources

Si Vap < Vr_{BC} **alors** // La partie problème qui est atteignable par la partie problème des cas sources est supérieure à la moyenne du taux de recouvrement de la BC

- Cas source ← cas *cible**

- BC ← BC ⋃ cas source // Introduire le cas dans la base de cas

Sinon

Ne pas apprendre le cas dans la base de cas

FinSi

Sinon

Cas source ← cas *cible** // Changement de statut du cas *cible**

BC ← BC ⋃ cas source // Introduire le cas dans la base de cas

FinSi

FinPour

Le principe de fonctionnement de l'algorithme est le suivant :

Après avoir donné une solution au cas cible, il changera de statut et il deviendra *cas cible**.

Ensuite, en fonction du taux d'atteignabilité de la partie solution, le cas sera admis ou non dans la base de cas. Si « Vas = 0 », ce qui signifie qu'aucun cas similaire à ce nouveau cas résolu n'a été recensé auparavant dans la base de cas, alors il va être ajouté à cette dernière. Par contre si la solution est atteignable « Vas > 0 », alors nous nous intéressons au taux d'atteignabilité de la partie problème par rapport au taux de recouvrement moyen de la base de cas. Si « Vap < Vr_{BC} » (ce qui signifie que le recouvrement du cas résolu est supérieur à celui de la base de cas), alors le cas est admis pour l'apprentissage. Ce cas appris contribuera à l'amélioration du taux de recouvrement de la base de cas et donc à l'amélioration de la compétence globale. De plus, la base de cas sera enrichie par ce nouveau cas appris.

La deuxième partie d'auto-incrémentation de la base de cas vient compléter la première partie de structuration donnant ainsi une méthode de maintenance de la base de cas aboutie. La structuration de la base de cas n'est valable dans le temps que si nous la maintenons. De ce fait, la méthode d'auto-incrémentation prend appui sur les mêmes critères de la méthode de structuration pour garantir des résultats adéquats avec celle-ci. Un cas ne peut être introduit dans la base de cas que s'il remplit certaines conditions dépendant du recouvrement et de l'atteignabilité du cas dans la base de cas. Ce cas apporte une solution inédite par rapport à la base de cas où il contribue à la compétence globale de celle-ci. Par conséquent, la méthode de maintenance garantit d'une part, une base de cas compétente et performante et d'autre part,

son bon fonctionnement dans le temps tout en l'enrichissant par de nouveaux cas. La validation de notre méthode se fera sur des méthodes de référence dans le domaine et sur la comparaison avec des benchmarks type.

4. Validation

La validation de notre contribution concernant la méthode de maintenance de la base de cas se fait sur plusieurs benchmarks type. Cette étude consiste à comparer notre méthode de maintenance à plusieurs autres méthodes de référence concernant l'optimisation de la base de cas. Cette comparaison se fait bien évidemment selon le critère de compétence et de performance dans le cas de la maintenance de la base de cas et sur les résultats concernant l'algorithme d'auto-incrémentation. Nous procédons à une comparaison séparée selon le critère car la plupart de ces méthodes traite un seul critère à la fois. En effet, cette étude comparative se fera en deux parties :

- la première concerne l'étude du critère de compétence ;
- et la deuxième concerne l'étude du critère de performance et de l'auto-incrémentation.

4.1. Etude comparative selon le critère de compétence

Dans cette partie, nous allons étudier les méthodes traitant le critère de compétence et faire la comparaison avec notre méthode qui s'appuie sur la mesure *MC*.

Trois méthodes vont être comparées à la notre, à savoir : la méthode standard *CNN*, la méthode dans laquelle les cas sont ordonnés suivant leur valeur de recouvrement *RC-CNN* et la méthode *NUN-CNN* dans laquelle les cas sont ordonnés suivant leurs distances NUN.

Afin de réaliser cette comparaison, quatre différentes bases de données sont utilisées. Les deux premières ; *Credit* (690 cas, 15 descripteurs et 2 classes) et *Ionosphere* (351 cas, 34 attributs et 2 classes) représentent des problèmes de classification et valables sur le site « *UCI repository machine learning databases* » (www.ics.uci.edu/~mlearn/MLRepository.html). Les deux autres; *Travel* (351 cas, 34 descripteurs) et *Property* (506 cas, 32 descripteurs) représentent des bases de données traditionnelles de RàPC. Ces deux dernières sont disponibles à partir des archives AI-CBR (www.ai-cbr.org).

Le protocole adopté dans cette première étude comparative est la suivante : l'ensemble des cas cibles considéré est un échantillon de 100 cas tirés aléatoirement de la base de

données, qui sont ensuite supprimés de cette dernière. Les 100 cas tirés aléatoirement vont représenter l'ensemble des cas cibles. Ensuite, les quatre méthodes seront appliquées. Cette opération est répétée 100 fois.

Suite à cette opération, une moyenne de la taille obtenue de la base de cas ainsi que la compétence équivalente sont calculées. Le tableau 3.6 illustre les résultats de la comparaison des quatre méthodes utilisant les quatre bases de données que l'ont vient de vous décrire [Haouchine et al., 2008a].

Base de données	Propriétés	CNN	NUN	RC	MC
Travel	Moyenne taille base de cas obtenue	184.28	197	165.42	145.74
	Compétence (%)	89.25	88.72	86.40	90.84
Property	Moyenne taille base de cas obtenue	55.19	57.81	45.44	39.62
	Compétence (%)	95.92	95.53	94.62	95.91
Credit	Moyenne taille base de cas obtenue	344.84	297.4	299.19	215.76
	Compétence (%)	58.85	58.95	60.44	62.37
Ionosphere	Moyenne taille base de cas obtenue	61.93	46.39	49.47	43.87
	Compétence (%)	85.78	84.44	85.3	86.92

Tableau 3.6. Comparaison des différentes méthodes sur des bases de données concernant leur taille moyenne et leur compétence

Les résultats illustrés dans le tableau 3.6 montrent que la méthode *MC* est la plus efficace. En effet, elle fournit la plus petite moyenne de cas avec le plus grand taux de compétence pour les quatre benchmarks. En effet, le taux de réduction produit par la méthode *MC* est sensiblement plus élevé que ceux produits par les trois autres méthodes, notamment au niveau des bases de données représentant des problèmes de classification à savoir « Credit » et « Ionosphere ». Concernant le taux de compétence, et toujours par rapport à ces deux dernières bases de données, la valeur fournie par la méthode *MC* est plus importante que celle obtenue avec les autres méthodes. Cependant, elle est pratiquement la même par rapport à celle fournit par *CNN* (95.91% pour *MC* et 95.92% pour *CNN*) pour la base de données traditionnelle de RàPC « Property ». Ce qui signifie que notre méthode est très efficace pour le traitement des problèmes de classification et relativement efficace concernant des données traditionnelles de RàPC.

4.2. Etude comparative selon le critère de performance et résultats de l'auto-incrémentation

Cette deuxième étude comparative concerne le critère de performance associé au nombre de cas dans la base de cas après opération de maintenance, que nous noterons « *storage* ». Nous allons prendre la méthode *ICF* comme méthode de référence afin d'établir la comparaison avec la méthode *MC*. En effet, Brighton et Mellish [2002] ont démontré dans leur étude expérimentale que leur méthode *ICF* qui a été comparée aux autres méthodes a donné les meilleurs résultats. Les méthodes qui ont été comparées sont : des anciennes méthodes telles que CNN, RNN (Reduced Nearest Neighbour) [Gates, 1972], SNN (Selective Nearest Neighbour Rule) [Ritter et al., 1975], Chang [Chang, 1974], Wilson Editing, {Repeated Wilson Editing, All k-NN} [Tomek, 1976], et des méthodes récentes telle que {IB2, IB3} [Aha et al., 1991], {TIBLE, Cameron-Jones's Extensions} [Cameron-Jones, 1992], RT3 [Wilson & Martinez, 1997] et ICF. Toutes ces méthodes ont été comparées uniquement selon le critère de performance. Nous notons que ce type de comparaison a également été réalisé dans les travaux de Wilson et Martinez [1997].

Notre étude expérimentale porte sur des bases de données disponibles sur « *UCI repository machine learning databases* » (www.ics.uci.edu/~mlearn/MLRepository.html). Ces bases de données, qui couvrent une large variété de domaines, vont être exploitées comme des benchmarks dans notre étude. Nous avons exploité 18 bases de données, qui sont les suivantes : anneal, balance-scale, brea-cancer-l, brea-cancer-w, Cleveland, credit, glass, hepatits, iris, lymphography, mushroom, pima-indians, post-operative, thyroide, voting, waveform, wine, zoo.

Le protocole adopté dans cette deuxième partie d'étude est le suivant : nous avons pris aléatoirement 20% des cas formant la base de test et les 80% restants constituent la base d'entrainement. Ensuite, le taux de précision ainsi que la taille finale de la base de cas (storage) sont calculés. Ce processus est répété plusieurs fois, les moyennes de la précision et de la taille de la base de cas sont montrées au tableau 3.6.

Concernant le protocole de l'auto-incrémentation (auto-apprentissage), nous avons pris appui sur le même protocole qui était utilisé dans la validation. Ensuite, une moyenne de cas appris est calculée et inscrite dans le tableau 3.6 [Haouchine et al., 2009].

Base de données	ICF		MC			Meilleure méthode
	Performance		Performance		Apprentissage (%)	
	Storage (%)	Précision (%)	Storage (%)	Précision (%)		
anneal	22.59	91.35	20.05	100.00	100.00	CM
balance-scale	14.67	81.47	13.78	95.83	81.08	CM
breast-cancer-l	23.51	72.81	4.02	96.56	92.31	CM
breast-cancer-w	4.27	95.14	5.29	93.24	100.00	ICF
cleveland	15.60	72.08	6.00	91.01	92.25	CM
credit	16.89	82.28	11.73	84.78	100.00	CM
glass	31.40	69.64	13.08	72.51	100.00	CM
hepatitis	16.33	82.26	11.03	90.03	100.00	CM
iris	42.08	92.56	10.66	86.99	85.68	CM
lymphography	25.63	77.59	18.92	96.31	93.78	CM
mushrooms	12.80	98.64	14.65	98.22	88.82	ICF
pima-indians	17.22	69.17	8.00	93.09	100.00	CM
post-operative	7.18	65.28	3.33	83.46	98.51	CM
thyroid	21.85	86.63	18.3	86.16	100.00	CM
voting	8.88	91.19	2.50	100.00	96.72	CM
waveform	18.98	91.19	18.53	96.87	97.98	CM
wine	12.00	83.81	3.66	92.94	100.00	CM
zoo	52.78	92.42	18.81	100.00	100.00	CM
Moyenne	20.25	83.08	10.99	92.33	95.95	CM

Tableau 3.6. La taille et la précision de classification équivalente pour chaque base de données suite à l'application des deux méthodes ICF et MC

Le tableau 3.6 illustre les résultats concernant la performance des deux méthodes ICF et MC, qui dépend du taux de bonne classification (précision) et du nombre de cas final (storage) dans les différentes bases de données. Le taux d'apprentissage de la méthode MC est également illustré ainsi qu'une dernière colonne montrant la méthode qui a fourni le meilleur résultat concernant la performance.

A partir de ces résultats, plusieurs constatations peuvent être faites. D'une façon globale, la méthode MC a fourni un meilleur taux de nombre de cas final et un meilleur taux de bonne classification. Il y a quelques bases de données dans lesquelles la méthode MC a donné particulièrement de bons résultats. Si nous prenons les deux bases de données *voting* et *wine*, le taux storage est en-dessous de 4% avec un fort taux de précision. En effet, la méthode MC donne de meilleurs résultats dans la plupart des bases de données « 16/18 ». Cependant, ICF est légèrement meilleure que CM concernant le storage et la précision par rapport aux deux bases : breast-cancer-w et mushrooms. Par conséquent, la méthode CM est largement meilleure que la méthode ICF, comme le montre la moyenne générale du storage et du taux de précision (un storage de 10.99 pour un taux de précision de 92,33% concernant MC).

Finalement, le taux d'apprentissage de la méthode MC est relativement bon car il ne descend pas en-dessous de 81.08%. Ce taux d'apprentissage est égal à 100% par rapport à neuf bases de données, ce qui signifie que la méthode d'auto-incrémentation de la base de cas fonctionne avec succès.

Nous pouvons donc conclure, suite à l'étape de validation de notre contribution, que la méthode de maintenance de la base de cas mise en place répond aux critères que nous avons fixés notamment la compétence, la performance et l'auto-incrémentation de la base de cas. En effet, notre méthode a donné globalement de meilleurs résultats par rapport aux meilleures méthodes qui existent dans la littérature concernant les critères étudiés.

5. Conclusion

Nous nous sommes intéressés dans ce chapitre aux travaux concernant la maintenance des systèmes de RàPC orientés *mining*. En effet, ce processus de maintenance a occasionné de nombreuses études et concerne essentiellement les travaux sur la maintenance des containers de connaissance des systèmes de RàPC. Ces études ont démontré que la base de cas représente le centre des systèmes de RàPC et qu'aucune maintenance ne peut s'enclencher sans la consulter. De ce fait, nous avons établi un état de l'art concernant la maintenance de la base de cas (MBC). Les travaux sur la MBC ont comme objectif d'assurer la qualité de la base de cas selon un certain nombre de critères. Il existe plusieurs critères permettant d'évaluer la qualité de celle-ci. Les plus utilisés sont la compétence et la performance. Ces deux critères sont utilisés dans plusieurs stratégies d'optimisation de la base de cas. Un état de l'art a été consacré à ces stratégies regroupées selon deux directions de recherche, Backward Elimination (BE) et Forward Selection (FS). Chacune des deux directions comporte un certain nombre de stratégies qui présentent des avantages et des inconvénients. Nous avons ainsi proposé une méthode de maintenance de base de cas. La première étape de notre méthode concerne la structuration de la base de cas. Cette structuration prend appui d'une part, sur un algorithme de catégorisation de cas dans la base de cas inspirée par les méthodes appartenant aux stratégies BE et d'autre part, sur une mesure de compétence *MC* que nous avons mis en place, inspirée des méthodes appartenant aux stratégies FS. La mesure MC permet de choisir les cas qui présentent le maximum de recouvrement et le minimum d'atteignabilité. En effet, ces deux dernières notions sont associées à la compétence de la base de cas. Par ailleurs, la structuration mise en place traite également le critère de performance qui est en relation avec le nombre de cas et le taux de bonne classification de ces cas. De ce fait, notre méthode vise à

réduire la taille de la base de cas tout en conservant une compétence maximale avec un meilleur taux de classification.

De plus, la base de cas est amenée à évoluer au fil du temps dans n'importe quel système de RàPC. Cette évolution concerne l'introduction de nouveaux cas dans la base de cas qui doit se faire pendant le fonctionnement du système. En effet, cette évolution peut altérer la qualité des résultats fournis. Cette qualité, comme nous l'avons indiqué, concerne les deux critères de compétence et de performance. Par conséquent, nous avons mis en place un algorithme d'auto-incrémentation des cas dans la base de cas qui constitue la deuxième étape de la méthode proposée. Cette auto-incrémentation prend appui sur les critères étudiés qui dépendent du recouvrement, de l'atteignabilité, du nombre de cas dans la base de cas et de la précision du classifieur.

Ensuite, nous avons validé ces deux propositions suivant un certain protocole. Cette validation a été appliquée sur des benchmarks de référence à travers une étude comparative des meilleures méthodes se trouvant dans la littérature. Cette validation a été faite en deux étapes : la première a traité le critère de compétence et la seconde a traité le critère de performance ainsi que le taux d'apprentissage des cas dans la base de cas.

Nous pouvons ainsi conclure qu'une méthode de maintenance de la base de cas classique n'aurait pas suffit à elle seule pour assurer une qualité de la base de cas tout au long de la vie du système de RàPC. Chose qui n'a pas été tenue en compte dans la majorité des travaux. Nous avons donc essayé de palier à ce manque en mettant en place les deux propositions de structuration de la base de cas et de son auto-incrémentation qui sont complémentaires et liées par les critères choisis.

Les systèmes abordés dans ce chapitre sont des systèmes orientés *mining* qui disposent d'une formalisation simple de cas et contiennent une phase de remémoration, de maintenance et d'auto-incrémentation. Malgré ces phases, à long terme un ensemble conséquent de cas peut détériorer le système et nécessiterait une modification de l'algorithme de recherche de cas. Nous avons donc orienté nos recherches vers des systèmes de RàPC à base de connaissance. Ce deuxième type de système est donc orienté connaissance. Il dispose d'une représentation complexe du cas associée à des modèles de connaissance. De plus, les phases de remémoration et d'adaptation sont plus élaborées. Par conséquent, nous nous intéressons à ce type de système au chapitre 4 en faisant un état de l'art sur les phases concernées et en proposant une méthode qui lie ces deux phases.

Chapitre 4

Proposition d'une approche de remémoration guidée par l'adaptation

1. Introduction

Nous avons abordé au précédent chapitre les *systèmes orientés mining.* En général, ces systèmes ne présentent pas de phase d'adaptation, ce que certains auteurs [Lieber et al., 2004], [Mille et al., 1996] définissent comme une spécificité des systèmes de RàPC. Le nombre de cas dans la base de cas compense l'étape d'adaptation, ce qui justifie la mise en place d'une phase de maintenance. Ainsi, nous avons construit un système orienté mining et proposé une méthode de maintenance de la base de cas et de son auto-incrémentation. Nous avons également développé un système *orienté connaissance (knowledge)* qui a l'avantage de réduire le nombre de cas dans la base de cas et de raisonner sur un nombre limité en prenant appui sur des modèles de connaissance liés à l'équipement à diagnostiquer. En effet, ce type de système dispose de cas qui ont une formalisation spécifique qui sont associés à des modèles de connaissance du domaine et de phases de remémoration et d'adaptation sont plus élaborées. Ainsi, nous désirons développer des phases de remémoration et d'adaptation, qui sont des points clés du système, afin de renforcer notre système orienté connaissance.

Pour cela, nous établissons un état de l'art concernant ces phases aux sections 2 et 3. En effet, il existe plusieurs types de mesures de similarité qui sont utilisés dans la phase de remémoration. Nous exposons les deux classes de cette phase à savoir : la remémoration simple et l'unification remémoration-adaptation. Nous présentons dans la première famille les deux groupes de similarité : similarité de surface et similarité structurelle, et dans la deuxième famille, les six types de remémoration liée à l'adaptation. Quant à la phase d'adaptation, nous recensons les types d'adaptation utilisés à travers les principaux travaux dans la littérature.

Suite à cet état de l'art, nous proposons, à la section 4, une méthode de remémoration guidée par l'adaptation et un algorithme d'adaptation dédiés aux systèmes de diagnostic par RàPC. En effet, la phase de remémoration guidée et la phase d'adaptation sont caractérisées par la formalisation du cas et les modèles de connaissance sous-jacents à la base de cas. Nous proposons la création du lien entre ces deux phases grâce à deux mesures utilisées dans la phase de remémoration : une mesure de remémoration (M_R) et une mesure d'adaptation (M_A). Cette dernière mesure a pour objectif de sélectionner le cas le plus facilement adaptable pour la phase d'adaptation.

Afin de valider notre proposition, nous étudions, à la section 5, la mise en place des deux phases de remémoration et d'adaptation sur un équipement industriel, en l'occurrence le

moteur 1.5 dCi K9K 105ch de la société « Renault » (http://v3.renault.com/_cfm/module-K9K/fr/index.html), que nous avons abordé au chapitre 2.

2. Phase de remémoration

La principale difficulté dans la phase de remémoration réside dans le choix du bon critère de sélection concernant les cas à remémorer [Cordier, 2008]. Lieber et Napoli [1998] traitent les questions concernant l'exactitude (*correctness*) et la complétude (*completeness*) des cas remémorés en RàPC. Les approches les plus traditionnelles utilisent les mesures de similarité rappelées au chapitre 1, tandis que d'autres impliquent l'organisation de la base de cas pour améliorer la qualité de recherche. En effet, nous pouvons classer la remémoration selon deux grandes familles : la première famille que l'on nommera ***« remémoration simple »*** et la deuxième ***« unification remémoration-adaptation »***. Nous nous basons sur l'état de l'art réalisé par Lopez de Mantaras et al. [2005] et Cordier [2008].

2.1. Remémoration simple

La remémoration simple est celle qui donne un cas ou un ensemble de cas à la fin du processus sans tenir compte de l'adaptation de la solution des cas dans la phase suivante. Cette phase est basée uniquement sur des mesures de similarité connues. Ces mesures sont divisées par Lopez de Mantaras et al. [2005] en deux groupes : *similarité de surface* et *similarité structurelle*.

2.1.1. Similarités de surface

Ce type de similarité ne tient compte que des attributs des objets. C'est ainsi le type le plus utilisé dans la littérature. Wess et al. [1993] organisent la base de cas selon la similarité entre les cas. Cette organisation permet de réduire le temps de remémoration. En effet, un arbre binaire « *k-d tree* » est utilisé pour diviser la base de cas en plusieurs groupes de cas, de manière à ce que chaque groupe contienne les cas les plus similaires selon une mesure de similarité donnée. Schaaf [1996] utilise la stratégie de *fish & shrink* dans laquelle les cas sont liés entre eux suivant un aspect de similarités bien spécifique. Cette stratégie suppose que si un cas ne correspond pas à une requête alors il réduira la probabilité que ses voisins soient impliqués.

Dans le même état d'esprit, Smyth et McKenna [1999a ; 2001b] utilisent un algorithme de remémoration basé sur la trace appelé : *footprint-based retrieval*. La trace représente l'ensemble de cas de la base de cas que peut recouvrir un cas (c'est-à-dire, le cas pouvant résoudre le même ensemble de problèmes). L'algorithme comporte deux étapes : la première consiste à identifier la trace du cas « *footprint cases* » qui est le plus similaire au cas cible, noté « *cas de référence* ». Ensuite, la deuxième étape consiste à rechercher un sous-ensemble de cas qui est en relation avec le cas de référence concerné. Finalement, le cas le plus similaire au cas cible est celui qui appartient au sous-ensemble sélectionné. Cet algorithme maximise donc l'efficacité de la phase de remémoration en recherchant seulement un sous-ensemble de cas dans la base de cas.

Toutefois, il existe d'autres études combinant la similarité de surface avec différentes techniques. Simoudis et Miller [1990] confirment que l'utilisation exclusive de la similarité de surface n'est guère suffisante pour discriminer les cas lorsqu'il s'agit d'une grande base de cas. De ce fait, ces auteurs combinent la similarité de surface avec une technique appelée « *la remémoration validée* ». Cette technique consiste à associer à chaque cas une *procédure de validation* liée à un ensemble de tests spécifiques au domaine ainsi que leurs résultats. Ces tests sont appliqués ensuite au cas cible. Afin de valider la remémoration d'un cas, il faut que tous les tests effectués sur le cas cible donnent les mêmes résultats qui se trouvent dans le cas source. Cette approche réduit le nombre de cas à considérer pour l'adaptation. Dans le même cadre, nous trouvons les travaux de Koton [1988], décrits au chapitre 2, qui utilise une phase de *justification* dans le système CASEY pour valider le cas remémoré. D'autres systèmes combinent également la similarité de surface à une technique de filtrage afin d'améliorer les performances de la remémoration, comme par exemple CHEF [Hammond, 1986], KRITIK [Goel & Chandrasekaran, 1989] et PROTOS [Porter et al., 1990].

Ce type de similarité est pertinent quand les descripteurs du cas sont représentés par des groupes de paires d'attributs-valeurs. Cependant, lorsqu'il s'agit d'une représentation plus complexe du cas, les similarités de surface deviennent moins efficaces. Pour pallier, un autre type de mesure est utilisé : *similarités structurelles*.

2.1.2. Similarités structurelles

Ce type de similarité s'intéresse aux relations entre attributs correspondants et repose sur l'utilisation des connaissances du domaine. Nous parlons alors d'une représentation de cas orientée objet. Cette représentation est tout simplement une généralisation de la représentation attribut-valeur [Lopez de Mantaras et al., 2005]. Les objets appartiennent à des classes qui

sont organisées d'une façon hiérarchique. Une classe d'objet détermine les attributs qu'elle peut contenir. Les attributs peuvent être *relationnels*, ce qui veut dire que leurs valeurs peuvent être eux-mêmes des objets. Ainsi, la hiérarchie des classes doit contenir la connaissance utile de similarité. Il y a plusieurs travaux qui se sont fait concernant ce type de similarité. Les similarités utilisées dépendent fortement de la représentation des cas.

Les premiers travaux réalisés se trouvent dans [Gentner & Forbus, 1991]. Börner [1993] s'inspire de ces travaux et propose une approche basée sur la similarité structurelle. Cette similarité est définie comme la structure graphique la plus spécifique que le cas cible a en commun avec le cas source. Cette structure prend appui sur un ensemble de règles de transformations basé sur les connaissances générales. Bergmann et Stahl [1998] abordent le problème de l'évaluation de la similarité des cas ayant une représentation orientée objet qui est souvent limitée aux objets de la même classe. Ils présentent ainsi une étude concernant le calcul de cette similarité. Cette étude permet aux objets des différentes classes d'être comparés et calculés à partir de la connaissance implicite qui se trouve dans la hiérarchie des classes.

Bunke et Messmer [1993] proposent des mesures de similarités structurelles basées sur une représentation graphique de cas. La structure du graphe s'appuie sur des opérations *d'éditions de graphe* (insertion, suppression et substitution des nœuds et arcs dans le graphe). Un algorithme d'appariement de sous-graphe est ainsi appliqué travaillant sur une version compacte de la base de cas dans laquelle se trouvent des sous-graphes communs aux multiples cas. Dans le même état d'esprit, Champin et Solnon [2003] proposent une mesure de similarité basée sur les opérations *d'éditions de graphe* dans un modèle de contraste modifié proposé dans [Tversky, 1977], pour comparer les cas représentés par des graphes étiquetés dans lesquels les nœuds et les arcs peuvent avoir plus d'une étiquette. Arcos et Lopez de Mantaras [1997] proposent un mécanisme de remémoration appelé « *perspectives* pour des représentations structurées de cas ». Les cas et les degrés de similarité sont représentés par des termes de caractéristique qui sont équivalents aux termes du premier ordre. Ces termes peuvent également être considérés comme des graphes acycliques orientés par des caractéristiques et des valeurs. Leur approche *intensive-connaissance* « knowledge-intensive » pour la remémoration utilise un mécanisme de *subsumption* dans les termes de caractéristique. Ce mécanisme permet l'obtention d'une relation d'ordre dans les descriptions du cas sur la base d'un ensemble de configurations appropriées pour le problème cible.

Dans un autre registre, [Brown, 1994 ; Wolverton & Hayes-Roth, 1994 ; Lenz, 1996] s'intéressent à la remémoration des cas, qui ont une représentation de combinaisons

d'attributs-valeurs, décrits par un réseau de nœuds interconnectés dans la base de cas. Un algorithme basé sur des connaissances spécifiques est ainsi exploité afin de réaliser l'activation de nœuds dans le réseau. Cette activation se propage à partir des nœuds du couple d'attribut-valeur du cas cible à travers le réseau dans le but d'activer les nœuds des cas sources similaires au cas cible. L'avantage de cette technique est qu'elle est efficace et assez souple pour traiter des cas qui ont une description incomplète. Cependant, la construction du réseau peut être couteuse en temps et demande une ingénierie de connaissance significative. Aamodt [1994] utilise une similarité orientée objet sur laquelle nous prendrons appui dans notre proposition.

2.2. Unification Remémoration-Adaptation

Avant les années 90, les deux phases de remémoration et d'adaptation étaient exploitées d'une façon complètement indépendante. Jusqu'à ce que Smyth et Keane [1993] apportent un nouveau souffle et suggèrent l'unification entre ces deux étapes dans le sens où les cas remémorés doivent être ceux les plus facilement adaptables afin d'optimiser les résultats. Dans ces travaux d'unification, la mesure de similarité est combinée avec d'autres critères afin de guider le processus de remémoration. D'après Lopez de Mantaras et al. [2005], nous pouvons recenser six types de remémoration liée à l'adaptation qui vont être abordés dans ce qui suit :

Remémoration guidée par l'adaptation (Adaptation-guided retrieval « AGR ») : Ces travaux ont été initiés par Smyth et Keane [1993] et sont connus sous le nom de : *Adaptation-guided retrieval*, noté : *« AGR »*. Ce type de remémoration a été le point de départ de plusieurs travaux prometteurs. Ces travaux ont été également présentés dans [Smyth, 1996] et [Smyth & Keane, 1998], et sont argumentés par la constatation suivante : ce n'est pas le cas le plus similaire, quand la mesure de similarité est choisie *a priori*, qui est le meilleur candidat à l'adaptation. De ce fait, la phase de remémoration doit donc rechercher non seulement des cas similaires pendant son processus mais surtout des cas facilement adaptables. En effet, les auteurs expliquent comment cette remémoration qui est guidée par l'adaptation lie les espaces des spécifications et des solutions en employant la connaissance d'adaptation (Figure 4.1).

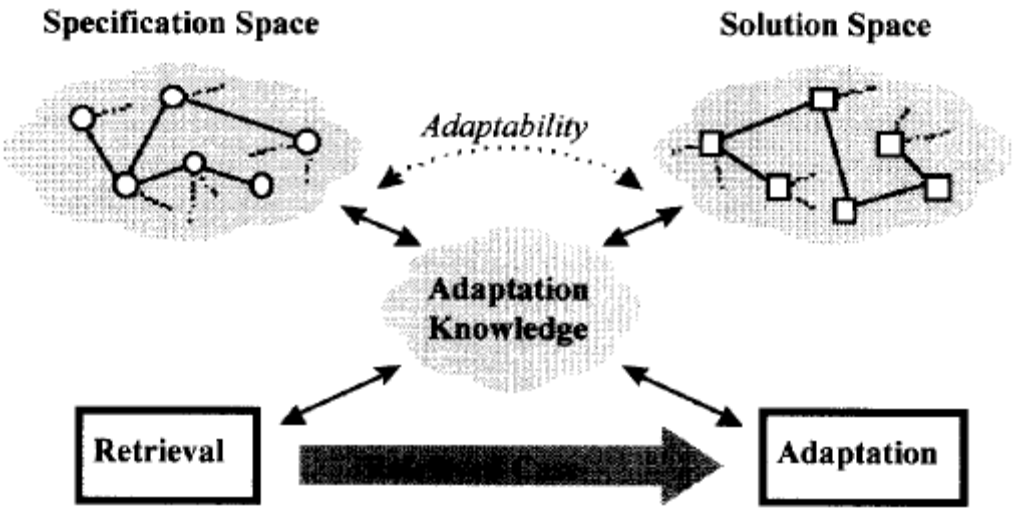


Figure 4.1. La remémoration guidée par l'adaptation liant les espaces des spécifications et des solutions [Smyth & Keane, 1998]

Ce lien permet d'avoir un canal de communication entre la remémoration et l'adaptation. Par conséquent, pendant la phase de remémoration, les connaissances d'adaptation sont utilisées pour intervenir en cas d'éventuels changements dans la solution.

Collins et Cunningham [1996] et Rousu et al. [1996] abordent le principe du calcul de *l'effort d'adaptation* dans des problèmes de planification. Dans le même registre, Leake et al. [1997] abordent également la notion de l'effort d'adaptation et la répercussion des traditionnelles mesures de similarité sémantiques sur l'adaptation. Certes, les cas remémorés sont « similaires » au problème cible, mais parfois difficiles voire impossible à adapter. Par conséquent, Leake prend en compte l'effort d'adaptation au moment de la remémoration afin de faciliter la phase d'adaptation. Cette prise en compte est concrétisée par l'insertion du *coût d'adaptation* dans la mesure de similarité. Alors, l'auteur propose deux étapes lors de l'évaluation de la similarité entre les cas sources de la base de cas et le problème cible. Une première étape de remémoration suivie par une étape d'un ordonnancement des cas remémorés en fonction d'un coût d'adaptation.

Dans un autre registre, Lieber [1999] propose une approche d'adaptation s'appuyant sur la notion de *chemins de similarité*. Cette notion est basée sur l'idée de décomposition de l'adaptation en sous tâches d'adaptation plus simples. Pour cette décomposition, il faut disposer de connaissances dépendantes du domaine. Cette approche comporte deux étapes : la première étape consiste à construire le chemin de similarité. Un chemin de similarité est constitué d'une succession linéaire de problèmes intermédiaires qui sont liés par des relations. A chaque relation est associée une fonction d'adaptation spécifique permettant de passer d'un

problème à un autre. La deuxième étape consiste à calculer les petites adaptations, qui se feront dans l'étape d'adaptation. Le but de cette approche est de décomposer un problème complexe en plusieurs sous problèmes plus simples afin de diminuer la difficulté de l'adaptation en faisant croître la similarité entre les problèmes.

Nous allons nous intéresser particulièrement aux travaux sur l'*AGR*, sur lesquels seront basées nos contributions, qui vont être présentés à la section 4.

Remémoration consciencieuse par la diversité (Diversity-conscious retrieval) : La spécificité de la remémoration simple est de fournir à l'utilisateur des cas qui ne répondent pas exactement à ses exigences mais qui sont très similaires à sa requête. Cependant, le problème qui peut se poser est que les cas remémorés sont souvent très similaires les uns par rapport aux autres, donc, l'utilisateur a finalement un choix limité concernant les résultats [Smyth & McClave, 2001]. De ce fait, les cas peuvent manquer de ce qui est appelé *diversité*. Pour y remédier, un certain nombre d'auteurs [Smyth & McClave, 2001 ; McSherry, 2002 ; McGinty & Smyth, 2003] se sont penchés sur la question en proposant des algorithmes combinant les mesures de similarité et la diversité dans le processus de remémoration afin de parvenir à un meilleur équilibre entre ces caractéristiques souvent contradictoires. Prenons l'exemple de Smyth et McClave [2001] qui propose une approche de remémoration qui sélectionne incrémentalement un ensemble varié de cas à partir d'un ensemble plus vaste de cas ordonnés par similarité. Ce type de problème est généralement posé dans les systèmes de recommandation de RàPC [Lopez de Mantaras et al., 2005], dans lesquels les descriptions des produits disponibles sont stockées dans une base de cas de produits et que la phase de remémoration s'enclenche à partir des besoins de l'utilisateur.

Remémoration conduite par les compromis (Compromise-driven retrieval) : Toujours dans le domaine des systèmes de recommandation de RàPC, il est noté que souvent les cas les plus similaires à la requête de l'utilisateur ne sont pas suffisamment représentatifs des compromis que l'utilisateur peut disposer pour que le cas soit accepté. Or, l'hypothèse de base de la remémoration simple est qu'un cas remémoré est plus acceptable qu'un autre cas s'il est le plus similaire au problème cible. Tandis que la remémoration conduite par les compromis est basée sur le principe qu'un cas donné est plus acceptable qu'un autre, s'il est le plus similaire au problème cible *et* qu'il implique un sous-ensemble des compromis que l'autre cas implique [McSherry, 2003b ; 2004]. Par exemple, aucun cas n'appartient à l'ensemble

remémoré s'il y a un autre cas plus similaire impliquant un sous-ensemble de compromis auquel il est affecté [Lopez de Mantaras et al., 2005].

Remémoration basée sur l'ordre (Order-based retrieval) : Comme dans la remémoration conduite par les compromis, ce type de remémoration n'a pas besoin de mesure explicite de diversité de recommandation parce que l'ensemble de cas remémorés est naturellement diversifié. En effet, ce type offre un langage de requête expressif afin de définir et combiner des relations d'ordre, et le résultat de l'évaluation de la requête ordonne partiellement les cas dans la base de cas. Le langage de requête supporte les requêtes préférant combiner naturellement d'autres valeurs d'informations telles que des valeurs maximales, des valeurs minimales et des valeurs que l'utilisateur préférerait ne pas considérer. Et là encore, nous restons toujours dans le domaine des systèmes de recommandation de RàPC dans lesquels sont appliqués particulièrement la remémoration basée sur l'ordre [Bridge & Ferguson, 2002].

Remémoration orientée explication (Explanation-oriented retrieval) : Doyle et al. [2004] affirment que le cas le mieux expliqué n'est pas nécessairement celui qui est le plus similaire au problème cible. En particulier, ils démontrent comment les cas qui se trouvent entre le problème cible et la frontière de décision peuvent être souvent utiles pour l'explication. Ceci a donc motivé le développement de la remémoration orientée explication. Les explications servent pour différents buts, celui d'enseigner l'utilisateur sur le domaine ou bien de lui expliquer la pertinence d'une question qu'il a posé [Sørmo & Cassens, 2004; Sørmo et al., 2005]. Généralement, le but de cette approche est d'expliquer comment le système a atteint ses conclusions.

Il est toujours important pour des systèmes de RàPC d'expliquer leur raisonnement et justifier leurs suggestions ou solutions [Cunningham et al., 2003 ; Leake & McSherry, 2005]. Par conséquent, il existe plusieurs travaux qui prennent appui sur les explications afin de rechercher le cas le plus similaire ou de discriminer deux ou plusieurs cas pertinents [McSherry, 2003a ; 2005]. En effet, lorsque le système retrouve les cas les plus proches dans la base de cas, il effectue une deuxième étape afin d'obtenir un cas d'explication [Doyle et al., 2004].

Remémoration basée sur l'optimisation (Optimization-based retrieval) : Mougouie et Bergmann [2002] formulent le problème de l'évaluation de la similarité de généralisation des

cas qui est décrite par des attributs continus comme étant un problème de programmation non linéaire et d'introduction d'une méthode de remémoration basée sur l'optimisation. Tartakovski et al. [2004] proposent une méthode de remémoration basée sur l'optimisation fonctionnant sur une structure d'index. L'évaluation de la similarité correspond à un cas particulier d'un problème d'optimisation non linéaire combiné à des nombres entiers.

[Bergmann & Wilk, 1998], [Bergmann, 2002] et [Mougouie & Bergmann, 2002] utilisent le concept de généralisation des cas couvrant le sous ensemble de l'espace total des solutions-problèmes. Ce concept permet de fournir des solutions à un ensemble de problèmes étroitement liés, au lieu que d'un seul problème, grâce à des dépendances entre les attributs explicitement représentées prenant appui sur l'extension des mesures de similarité. Bergmann [2002], par exemple, définit une mesure de similarité entre une requête et une généralisation de cas comme une similarité entre la requête et le cas le plus similaire contenu dans le cas généralisé. Tartakovski et al. [2004] étendent la représentation des cas vers une représentation mixée entre les attributs continus et discrets.

3. Phase d'adaptation

L'adaptation est une étape délicate à mettre en place dans un système de raisonnement à partir de cas. Partant de ce constat, certains auteurs évitent de traiter cette phase et préfèrent développer la partie remémoration [Kasif et al., 1995] en considérant que la richesse de la base de cas peut compenser la phase d'adaptation [Stanfill & Waltz, 1986].

Par contre, d'autres auteurs considèrent que l'adaptation est au cœur des systèmes de RàPC [Lieber et al., 2004] et [Mille et al., 1996]. Cordier et al. [2006], quant à eux, disent que l'adaptation confère au système de RàPC sa qualité de résolveur de problèmes. Fuchs et al. [2000] considèrent que cette phase est la plus complexe et la plus délicate du cycle du RàPC.

Plusieurs types d'adaptation sont utilisés à travers les différents travaux dans la littérature. Nous allons citer les travaux et les systèmes de RàPC les plus utilisés et les plus connus.

Adaptation substitutionnelle

L'adaptation substitutionnelle consiste à ré-instancier quelques parties de la solution remémorée par rapport au cas cible. Elle comprend les différentes modifications concernant le changement des valeurs des attributs.

Plusieurs travaux ainsi que des systèmes de RàPC ont exploité ce type d'adaptation pour résoudre le problème rencontré. Généralement, l'adaptation substitutionnelle est combinée à d'autres types d'adaptations que nous allons aborder dans la prochaine section.

Adaptation transformationnelle

L'adaptation transformationnelle a été proposée par Carbonell dans [Carbonell, 1984]. Elle consiste à changer la structure de la solution [Kolodner, 1993]. Elle ne dispose pas de toutes les connaissances pour résoudre le problème à partir de zéro. Dans ce type d'adaptation, il est supposé qu'il y a des éléments de réponse à un problème qui, grâce à un ensemble de règles d'adaptation, nécessitent des modifications, suppressions ou ajouts selon des écarts de contexte observés entre le cas source et le cas cible. Il est également supposé dans ce type d'adaptation que les solutions proposées sont suffisamment flexibles pour permettre ainsi leurs transformations.

Cependant, il existe des systèmes qui utilisent les deux types d'adaptation substitutionnelle et transformationnelle en même temps.

CHEF [Hammond, 1990], qui un système de planification de menus, utilise une méthode d'adaptation de recettes. Ce système commence par une adaptation substitutionnelle qui consiste à substituer les ingrédients dans le cas remémoré comportant une recette afin de satisfaire les besoins du menu. Si la solution proposée n'est pas satisfaisante alors il procède à une adaptation transformationnelle. Cette adaptation est donc utilisée afin de modifier la solution proposée en ajoutant ou en supprimant des étapes dans la recette. Ces modifications sont issues du résultat des différentes substitutions d'ingrédients. A titre d'exemple : si la solution remémorée reflète une recette contenant de l'agneau et des aubergines et que la recette désirée comporte du poulet et des petits pois, alors le composant viande est remplacé par le poulet et le composant végétal est substitué par des petits pois. Ensuite, l'adaptation transformationnelle qui vient compléter la solution sera comme suit : le poulet qui a pris la place de l'agneau dans la recette comporte une étape en plus dans sa préparation contrairement à l'agneau. De ce fait, une étape devrait être ajoutée qui consiste à plumer le poulet.

SWALE [Schank & Leake, 1989] est un système d'explication à partir de cas destiné à la compréhension d'histoire. Tout comme le système CHEF, il commence par l'adaptation substitutionnelle de la solution remémorée, qui est sous forme d'explications, qui consiste à modifier les acteurs de l'histoire, leurs rôles et éventuellement l'action affectée à chaque acteur. Ensuite, si besoin est, l'adaptation substitutionnelle intervient lors de l'ajout ou la

suppression de quelques parties dans l'explication résultante après l'adaptation substitutionnelle.

Adaptation à base de modèles

Comme on a pu voir au chapitre 2, le système Casey utilise une adaptation basée sur les modèles. En effet, Casey, qui est un système de diagnostic médical dédié aux patients qui ont des problèmes cardiaques, est le premier système qui applique ce type d'adaptation à base de modèle [Koton, 1988]. Ce système utilise des stratégies indépendantes du domaine de réparation pour adapter l'explication remémorée. Ces stratégies prennent en compte les différences entre les symptômes des patients retrouvés lors de la phase de remémoration et ceux des nouveaux patients. Une modification de l'explication causale est appliquée en ajoutant ou en supprimant des nœuds et des liens dans le modèle.

Nous trouvons également dans le même cadre le système CADRE [Falting, 1997]. CADRE (Case Adaptation by Dimensionality REeasoning) est un système de RàPC dédié à l'adaptation basée sur les contraintes de conception des bâtiments. CADRE utilise des modèles de structure, de comportement ainsi que des fonctions pour représenter les cas ainsi que leurs interprétations. Le modèle de structure est modélisé par l'outil commercial AutoCAD (un logiciel de dessin assisté par ordinateur, créé en 1982 par Autodesk [Wikipédia, 2009]). Ce modèle est composé d'objets (les sommets, les lignes et les plans du modèle AutoCAD), de variables de positions ou de dimensions de ces objets et un ensemble de valeurs attribuées aux variables. Des fonctions d'architectures sont prises en compte donnant des précisions à l'utilisateur sur ce qu'il peut faire dans un espace donné (exemple : supposons qu'on dispose d'un espace qui peut être exploité autant qu'un bureau en mettant des tables et des chaises, ou autant qu'une cuisine dans laquelle sont installés un four et un évier. Ces éléments vont définir la fonction de cet espace. Ensuite, cet espace exige un certain comportement, en demandant suffisamment de place et d'éclairage pour le confort de l'occupant. Cependant, s'il n'y a pas assez d'espace dans un bureau pour accéder à la table alors le comportement de « bureau » n'est pas satisfait). Les comportements et les fonctions sont modélisés par des *contraintes* formulées directement sur la structure. Ces contraintes sont réparties en trois catégories : *définitions* (tel que surface(a) = largeur(a)*longueur(a)), *contraintes d'intégrité* (tel que fin(a) = début(b)) et les *restrictions* (tel que surface(chambre) > 8). L'adaptation dans CADRE consiste à restreindre les contraintes des dimensions de l'espace dans AutoCAD à partir d'un bâtiment existant dans la base de cas. Un cas est généralement adapté dans un contexte différent alors que le contexte est souvent à l'origine

des conflits avec ces contraintes. Ces conflits peuvent être résolus par deux formes d'adaptation : *adaptation dimensionnelle* (lorsque seulement les dimensions du cas sont changées) et *adaptation topologique* (lorsque la forme et le nombre des espaces et des murs sont également modifiés). Une solution générale pour l'adaptation dimensionnelle est développée. Cette solution est basée sur les notions de l'expansion de dimensionnalité pour rendre les conflits plus facilement à résoudre et sur une réduction dimensionnelle des sous-séquences afin de limiter la complexité des modifications. Concernant l'adaptation topologique, deux mécanismes sont utilisés. Pour les *changements réguliers* telle que la suppression d'un ensemble d'espaces identiques, il est possible de formuler des règles qui opèrent dans une structure objet hiérarchique construite sur le modèle AutoCAD. Tandis que pour les *changements plus généraux*, il n'est possible de formuler des règles que si un autre cas apporte les connaissances nécessaires pour une nouvelle disposition.

Dans le même registre, nous trouvons aussi le système FAMING qui est un système de RàPC pour la conception à partir de cas des formes pour les différents mécanismes. [Falting, 1997]. FAMING utilise le même principe d'adaptation basée sur les modèles que le système CADRE.

KRITIK est un système de conception multi stratégie à partir de cas proposé dans [Goel & Chandrasekaran, 1989] et [Murdock & Goel, 2001]. Ce système prend appui sur une adaptation transformationnelle basée sur les modèles afin de réutiliser des conceptions pour des dispositifs physiques. L'adaptation est considérée comme l'une des tâches de conception. Elle prend en entrée, d'une part, la spécification des contraintes de la conception désirée et, d'autre part, les caractéristiques des contraintes ainsi que la structure de la conception à obtenir. Ainsi, l'adaptation a pour objectif de donner en sortie une structure de conception modifiée qui satisfait les contraintes spécifiées. L'adaptation dans KRITIK exploite des méthodes à base de modèles qui divisent la tâche de conception en trois sous-tâches : *calcul des différences fonctionnelles*, *diagnostic* et *réparation*. La conception remémorée est tout d'abord vérifiée pour détecter la fonctionnalité qui diffère de la fonctionnalité désirée. Le modèle de conception est alors analysé en détail pour déterminer une ou plusieurs causes possibles pour la différence observée. Enfin, le système procède à des modifications de la conception afin d'obtenir les fonctionnalités désirées.

Adaptation hiérarchique

L'adaptation hiérarchique a une spécificité qui réside dans l'organisation des cas dans la base de cas. En effet, les cas sont rangés d'une façon hiérarchique et organisés en plusieurs

niveaux d'abstraction. Bergmann et Althoff [1998] combinent ce type d'adaptation à d'autres modèles d'adaptation, comme dans le cas du système Déjà Vu [Smyth & Keane, 1995a ; 1998]. Ce dernier est un système de RàPC pour la conception de logiciels de pilotage d'installations industrielles. Il combine l'adaptation hiérarchique avec d'autres types d'adaptation que nous allons aborder ultérieurement. L'adaptation hiérarchique dans ce système est effectuée à différents niveaux de granularité en prenant appui sur des objets qui sont décrits dans une hiérarchie de composition. Wilke et al. [1998] précisent que ce type d'adaptation est réalisé du plus haut niveau au plus bas niveau. En effet, l'adaptation peut avoir besoin des cas simples pour donner la solution adéquate au cas cible. Cependant, elle peut avoir besoin des cas multiples en utilisant différents détails ou différentes parties des solutions réutilisées. Dans tous les cas, l'adaptation commence par adapter la solution au niveau d'abstraction le plus haut dans lequel les informations sont plus générales. Ensuite, la solution est raffinée, si besoin est, en descendant un peu plus dans la hiérarchie pour avoir plus de détail.

Nous allons exploiter ce type d'adaptation dans notre algorithme d'adaptation proposé et présenté à la section 4.3.2.

Adaptation générative

L'adaptation est généralement réalisée en modifiant directement la solution remémorée. Par contre, et contrairement à l'adaptation substitutionnelle et transformationnelle, l'adaptation *générative* est plus complexe. Elle consiste à rejouer la méthode qui a conduit la solution du cas remémorée sur le cas cible afin de lui fournir une solution adéquate à sa partie problème.

Dans ce cadre d'adaptation, nous pouvons trouver le système Prodigy/Analogy [Veloso, 1994; Veloso & Carbonell, 1994]. Ce dernier est un système de planification d'usage universel, il construit une nouvelle solution d'un ensemble de cas de planification. Ce système utilise une méthode de transfert des chaînes de décisions de résolution de problèmes. En effet, des problèmes complexes peuvent être résolus en déterminant les interactions partielles parmi des cas passés déjà résolus. Pour réaliser la phase d'adaptation, le système s'appuie sur un *jeu dérivationnel* pour recalculer le remplacement d'un élément défectueux de la solution remémorée. Ce jeu rappel la façon dont l'élément a été calculé et rejoue le calcul pour le cas cible.

Adaptation compositionnelle

Ce type d'adaptation combine des parties des solutions de multiples cas remémorés pour avoir une solution composée [Wilke et al., 1998]. Plusieurs cas sont ainsi utilisés en même temps pour construire la solution finale du cas cible afin d'obtenir une solution adéquate qui répond aux exigences de l'utilisateur. Généralement, ce type d'adaptation utilise un algorithme récursif afin de choisir au fur et à mesure les cas similaires au cas cible. De plus, l'adaptation compositionnelle peut être utilisée dans deux situations différentes [Arshadi et al., 2000]. La première concerne la solution qui se compose de plusieurs parties indépendantes les unes des autres. En effet, dans cette situation, les solutions de chaque cas source remémoré peuvent être divisées en plusieurs parties indépendantes. De ce fait, la solution finale peut ainsi être composée de différentes parties des solutions remémorées. Cette méthode est efficace lorsqu'il y a peu de conflits entre les composants des solutions [Wilke et al., 1998]. De ce fait, le changement d'un composant n'aura pas d'effet sur les autres. La deuxième situation concerne les solutions des cas remémorés qui ne peuvent être divisées en parties indépendantes. Dans cette situation, la solution est conçue selon des modes de calculs dépendant de l'application. A titre d'exemple, nous pouvons citer le système Airquap [Lekkas et al., 1994], qui est un système de RàPC de prévision des niveaux de pollution. Le système procède au calcul de la solution cible en calculant la valeur moyenne des solutions appartenant aux cas les plus similaires dans la base de cas.

Adaptation évolutionnaire

Ce type d'adaptation existe à travers les méthodes évolutionnaires et notamment les algorithmes génétiques. En effet, ces méthodes ont été étudiées pour des besoins d'adaptation dans le cadre de la conception architecturale [Gómez de Silva Garza & Maher, 2000]. Tout d'abord, juste après la phase de remémoration, les conceptions les plus similaires formeront la population initiale de l'algorithme génétique. Ensuite, la *mutation* et les *opérateurs de reproduction* (*crossover*) sont utilisés pour générer de nouvelles conceptions pour la population. La mutation est considérée comme une *adaptation substitutionnelle* qui modifie aléatoirement des parties de la conception afin de reproduire une nouvelle conception. La reproduction quant à elle, est considérée comme une *adaptation transformationnelle* qui consiste à modifier complètement la structure de la conception. De ce fait, la reproduction produit deux nouvelles conceptions à partir de deux ou plusieurs conceptions parents en inversant des parties de la conception dans chaque parent. Enfin, la fonction *fitness* de l'algorithme génétique évalue les conceptions en calculant la correspondance avec la

conception du cas cible. Ainsi, la conception source ayant le plus grand appariement avec la conception cible sera donc sélectionnée et considérée comme la nouvelle conception.

Nous pouvons également trouver d'autres types d'adaptations qui sont liés aux catégories des connaissances d'adaptation, tels que les *spécialistes d'adaptation* et les *stratégies d'adaptation*.

Spécialistes d'adaptation

Ce type d'adaptation est utilisé pour réaliser des modifications locales suivant les différences des spécifications des descripteurs de problème source et cible. Les spécialistes d'adaptation fonctionnent soit sur des cas abstraits, soit des cas concrets. Ils offrent des procédures d'adaptation spécialisées pour des objets ou des tâches particulières.

Stratégies d'adaptation

Contrairement aux spécialistes d'adaptation, les stratégies d'adaptation sont sous forme générale et réalisent des modifications plus globales en détectant des incohérences ou des problèmes d'interaction.

Le système Déjà Vu [Smyth & Keane, 1995a ; 1998] exploite les deux types d'adaptation sus-cités. Le domaine d'application principal de ce système est le contrôle de véhicules robotisés dans une aciérie. Le principe de base de ce système repose sur la conception d'une solution en réutilisant plusieurs cas de manière combinée. En effet, des parties de cas, spécifiques à la conception, sont combinées pour résoudre de nouveaux problèmes de conception. Les relations de décomposition sont exploitées pour identifier les spécifications de sous-problèmes. L'intégration de plusieurs éléments dans la solution cible replace les relations de décomposition. Dans ce système, les connaissances d'adaptabilité sont stockées dans les parties conditions des règles d'adaptation. Ces connaissances sont organisées en deux catégories : la première concerne les spécialistes d'adaptation et la deuxième concerne les stratégies d'adaptation. Les spécialistes d'adaptation permettent de sélectionner les cas les plus adaptables par rapport au cas cible, qui sont appelés : les cas localement adaptables. Ensuite, les stratégies d'adaptation essaient de reconnaître les conflits dans les cas localement adaptables. Ainsi, pour chaque cas, un coût d'adaptation global est calculé en agrégeant les coûts des spécialistes d'adaptation et les coûts des stratégies d'adaptation. Les cas sont ensuite ordonnés par coût décroissant.

Nous pouvons également recenser d'autres types d'adaptation tels que l'adaptation conservatrice [Lieber, 2007 ; Cojan & Lieber, 2008], la recherche en mémoire et application de cas d'adaptation correspondant à une acquisition progressive de compétences d'adaptation [Leake et al., 1996a], l'adaptation plan [Koehler, 1996] ou encore l'analogie par dérivation et l'analogie par transformation [Carbonell, 1986] et [Veloso, 1994].

A travers cette étude concernant la diversité des différents types d'adaptation, nous pouvons constater que cette étape a toujours été considérée comme étant une étape très importante, au cœur de la majorité des différents types d'application et constitue un défi majeur pour les chercheurs.

Par conséquent, d'autres recherches complémentaires ont été menées. Trois directions principales ont été définies : les démarches unificatrices, les catalogues et les méthodes d'acquisition de connaissances d'adaptation.

Les démarches unificatrices visent à proposer des modèles généraux d'adaptation sous différents angles (principes, algorithmes, etc.). Nous pouvons citer à titre d'exemple les travaux de Hanney et Keane [1996] qui ont pour objectif de construire des règles d'adaptation à partir des différences entre les attributs des paires de cas. Ensuite, ces règles vont être généralisées et raffinées. De plus, des mesures de confiance sont associées à ces règles en fonction de leur degré de généralisation. Il existe d'autres travaux dans cette direction tels que les travaux de [Fuchs et al., 2000] auxquels nous accorderons une attention particulière.

La seconde direction met en évidence des catalogues de stratégies d'adaptation qui sont susceptibles de s'appliquer à plusieurs domaines [Riesbeck & Schank, 1989 ; Lieber, 2002].

La troisième direction concernant les démarches d'Acquisition de Connaissances d'Adaptation (ACA) considère que l'adaptation est dédiée à un cadre d'un domaine d'application et vise à mettre en évidence des principes généraux destinés non à l'adaptation elle-même, mais aux moyens de l'expliciter dans le domaine d'application considéré [Fuchs & Mille, 2005; Lieber et al., 2003]. D'Aquin et al. [2004] divisent l'ACA en deux catégories : *ACA supervisée* concernant les recherches allant vers des méthodologies d'acquisition auprès d'experts et l'*ACA automatique* qui s'appuie sur la connaissance de la base de cas.

Lieber et al. [2004] ont réalisé un état de l'art sur les travaux en ACA ainsi qu'une étude comparative. L'étude comparative est basée sur trois questions : quelles sources de connaissances sont utilisées ? Quelles sont les hypothèses sur la représentation des cas ? Quels sont les types de connaissances d'adaptation acquises ?

En effet, les approches ACA automatiques sont des techniques pouvant utiliser plusieurs représentations de cas, à savoir : un formalisme simple de type attribut-valeur [Corchado & Lees, 2001 ; Jarmulak et al., 2001 ; Lee, 2003], un second formalisme dans lequel l'ensemble des attributs peut varier d'un cas à l'autre mais traduit en un formalisme simple [Anand et al., 1998] ou un formalisme de type MOPs (Memory Organisation Packets [Riesbeck & Schank, 1989]) [Hammond, 1990 ; Leake et al., 1996b]. Ce type d'approche est fondé sur des règles d'adaptation contenant des informations sur la variation entre les problèmes comparés (leur dissimilarité) et des informations sur le contexte (i.e., sur ce que ces problèmes doivent partager pour que la règle s'applique). Concernant les approches ACA supervisée, elles ne nécessitent pas d'hypothèse sur la représentation des cas. Elles se basent également sur des règles d'adaptation qui, en revanche, peuvent être *a priori* quelconque.

Dans nos travaux, nous nous sommes intéressés aux travaux de Fuchs et al. [2000] qui ont proposé une approche générique de l'adaptation dans laquelle les cas sont décrits par un ensemble de descripteurs dont les valeurs sont des nombres ou des contraintes locales sous la forme d'intervalles numériques. Cette approche introduit des opérateurs d'adaptation généraux et indépendants du domaine. Elle est basée sur deux idées principales :

La première concerne une stratégie d'adaptation fondée, d'une part, sur l'appariement entre le cas source et le cas cible et, d'autre part, sur les dépendances entre descripteurs du cas source. L'appariement représente la variation entre le cas source et le cas cible, provoquant des variations de leurs descripteurs respectifs, qui est dans ce cas précis la soustraction des valeurs de bornes inférieures et supérieures entre les descripteurs du cas cible et le cas source. Les dépendances entre un problème et une solution expriment l'influence de la variation d'un descripteur de problème sur les descripteurs solution. Les auteurs supposent que ces dépendances sont connues à l'avance et sont mémorisées dans le cas source (elles peuvent être vues comme des *explications* qui lui sont associées). Elles sont exprimées par des étiquettes de dépendance (ÉD). Ainsi, trois valeurs sont possibles :

- ÉD > 0 : valeurs de descripteurs ds et Ds varient de la même façon ;
- ÉD < 0 : valeurs de descripteurs ds et Ds varient en sens contraire ;
- ÉD = 0 : valeurs de descripteurs ds et Ds sont indépendantes.

La deuxième idée repose sur un algorithme général pour adapter un cas source en un nouveau cas s'appuyant sur une détermination d'intervalles numériques de variations pour les attributs du nouveau cas. Lorsqu'il y a un descripteur cible qui ne dépend d'aucun descripteur

source alors il n'y a aucune variation engendrée, donc la valeur du cas cible peut être conservée. Lorsqu'il y a un descripteur cible qui dépend d'un ou de plusieurs descripteurs source, le résultat de l'algorithme d'adaptation sera l'intervalle (IR, IÉ) tel que :

- IR = [R_{min}, R_{max}] : *intervalle restreint*, dans lequel la valeur par défaut de l'attribut du descripteur concerné peut être choisie ;
- IÉ = [E_{min}, E_{max}] : *intervalle étendu d'erreur maximum*, dans lequel la valeur de l'attribut du descripteur concerné peut être choisie.

Par conséquent, le descripteur de solution du cas cible aura la valeur de l'intervalle suivante : [E_{min}, [R_{min}, R_{max}], E_{max}].

Suite à cet état de l'art, nous pouvons conclure que les deux phases de remémoration et d'adaptation sont fortement liées et qu'il existe plusieurs techniques utilisées. A partir de cette constatation et des particularités du diagnostic dans le cadre des méthodes d'unification remémoration-adaptation, nous proposons une méthode de remémoration guidée par l'adaptation pour un système de diagnostic par RàPC.

4. Proposition d'une méthode de remémoration guidée par l'adaptation

L'adaptation est une étape délicate à mettre en place dans un système de raisonnement à partir de cas et est considérée comme spécifique au domaine d'application que l'on traite. En diagnostic, l'identification des problèmes est la première étape du dépannage. Cependant, excepté dans des cas simples, il n'est pas possible d'établir une table prédéfinie de solutions adaptées à chaque cas, car cela soulève un problème d'organisation de l'information et de sa structuration à l'avance.

Par conséquent, la création d'un système de diagnostic basé sur le RàPC passe par l'exploitation des connaissances du système à diagnostiquer en mettant en place une modélisation de ces connaissances spécifique au domaine du diagnostic ainsi qu'une formalisation particulière des cas avant le développement des phases de remémoration et de l'adaptation. Nous avons abordé au chapitre 2 la description d'un système de RàPC orienté connaissance dédié au diagnostic. Nous allons proposer dans ce qui suit la mise en place des

deux phases de remémoration et d'adaptation ainsi que les mesures appliquées permettant d'unifier ces deux phases.

4.1. Formalisation du problème

Tout d'abord, nous allons commencer par un petit rappel, abordé au chapitre 1 (section 3.4.1), concernant les notations que nous allons utilisées pour les cas sources et cibles : un cas source est représenté par un couple (srce, Sol(srce)) et le cas cible par le couple (cible, Sol(cible)), où Sol(cible) est inconnue et on voudrait lui apporter un résultat.

- dsi, dci (pour i = 1, …, n) : représentent les descripteurs de la partie problème du cas source « srce » (respectivement problème cible « cible ») ;
- Dsi, Dci (pour i = 1, …, m) : représentent les descripteurs de la partie solution du cas source « Sol(srce) » (respectivement solution cible « Sol(cible) »).

Dans le cadre de la théorie d'unification entre l'adaptation et la remémoration, nous avons formalisé notre problème en tenant compte de différents points, qui seront exploités dans ces différentes phases.

Nous avons vu, au chapitre 2, que le cas avait une représentation spécifique dédiée au diagnostic. Cette représentation comporte une partie *localisation* et une partie *fonctionnelle* dans sa partie problème. Concernant la partie solution, elle décrit la classe de défaillance (*détection*) et l'*identification* du composant défaillant. L'une des particularités de notre étude est que la partie fonctionnelle soit décrite par un ensemble de descripteurs qui sont composés de trois différents attributs. Ces attributs sont relatifs à la valeur du composant, son état et son mode de fonctionnement : $ds_i = (ds_i^{Valeur}, ds_i^{Etat}, ds_i^{M.F})$.

La phase de remémoration tiendra compte de ces trois types d'attributs. La phase d'adaptation exploitera les relations de dépendance en parcourant un modèle de contexte. Ce modèle définira les relations de cause à effet entre les descripteurs.

4.2. Etape de remémoration

Nous allons exploiter dans la phase de remémoration l'algorithme des k plus proches voisins en l'associant à une mesure de similarité globale qui est composée d'un ensemble de mesures de similarité locales.

Pour remémorer un cas similaire le plus favorable à l'adaptation, nous devons évaluer dans un premier temps la ressemblance entre les descripteurs et entre les attributs de chaque descripteur. En effet, pour la partie localisation, les descripteurs de problème du cas cible et des cas sources vont être comparés. Ensuite, concernant la partie fonctionnelle, ça sera les attributs des descripteurs qui vont être comparés. Les descripteurs du cas ayant des valeurs qualitatives, la mesure de similarité obtenue pour un descripteur donné dépendra des trois attributs le décrivant qu'on nommera : « *mesure de remémoration* ».

4.2.1. Mesure de Remémoration (M_R)

La Mesure de Remémoration (M_R) dépend de la formalisation du cas en tenant compte de la similarité entre trois attributs composant le descripteur, plus une fonction déterminant la présence de ce descripteur dans le cas source. Ces valeurs sont soient binaires, normées ou comprises dans l'intervalle [0, 1] pour positionner les valeurs comparées dans une hiérarchie.

Ainsi, nous définissons quatre mesures de similarité locales associées à chaque type de descripteurs comme suit [Haouchine et al., 2008b] :

- *l'état du descripteur défini par une mesure de similarité locale* φ^{Etat}

  φ^{Etat} peut avoir deux valeurs possibles :

$$\begin{cases} \varphi^{Etat} = 1, \text{ quand } ds_i^{Etat} = dc_i^{Etat} \\ \varphi^{Etat} = 0, \text{ quand } ds_i^{Etat} \neq dc_i^{Etat} \end{cases}$$

- *l'état du mode de fonctionnement dans le cas de diagnostic défini par une mesure de similarité locale* $\varphi^{M.F}$

  $\varphi^{M.F}$ peut avoir deux valeurs possibles :

$$\begin{cases} \varphi^{M.F} = 1, \text{ quand } ds_i^{M.F} = dc_i^{M.F} \\ \varphi^{M.F} = 0, \text{ quand } ds_i^{M.F} \neq dc_i^{M.F} \end{cases}$$

- *La classe d'appartenance des valeurs des descripteurs définis par une mesure de similarité locale* φ^{Valeur}

La classe d'appartenance est définie dans la hiérarchie des descripteurs. En effet, les valeurs que peuvent prendre les descripteurs, suivent une hiérarchie de descripteur et permet ainsi de rapprocher les descripteurs appartenant à la même classe. Nous définissons ainsi une

mesure dépendante de cette hiérarchie, ce qui permettra de travailler sur des classes et de généraliser ainsi le cas à retrouver.

La Figure 4.2 illustre un exemple de hiérarchie de descripteurs. Cette figure montre deux descripteurs « ds_2 » (qui a trois valeurs) et « ds_3 » (qui a deux valeurs). Ces deux descripteurs sont regroupés par le descripteur « ds_1 ». φ^{Valeur} peut ainsi prendre différentes valeurs, en voici un exemple :

$$\begin{cases} \varphi^{Valeur} = 1, & \text{si } ds_i^{Valeur} = dc_i^{Valeur} \\ \varphi^{Valeur} = 0.8, & \text{si } \{ ds_2^{Valeur} = val1 \text{ et } dc_2^{Valeur} = val2 \} \\ \varphi^{Valeur} = 0.6, & \text{si } \{ ds_1^{Valeur} = val1 \text{ et } dc_1^{Valeur} = val4 \} \end{cases}$$

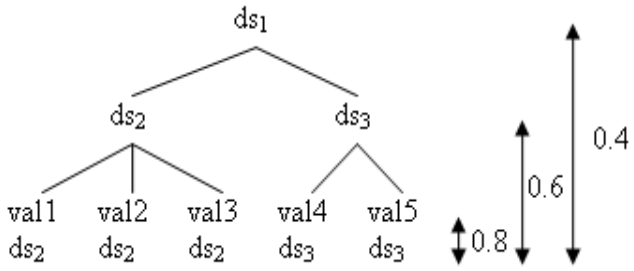


Figure 4.2. Exemple d'une hiérarchie des descripteurs

- *La présence des descripteurs définie par une mesure de similarité locale* $\varphi^{Présence}$

Les descripteurs n'étant pas renseignés obligatoirement dans chaque cas, nous leur affecterons un attribut relatif à leur présence dans le cas ; à savoir $ds_i^{Présence}$

La base de cas contient des cas dans lesquels des descripteurs ne sont pas renseignés. De ce fait, il est important de ne pas en tenir compte.

$\varphi^{Présence} = 1$, le descripteur est renseigné dans le cas source *et* dans le cas cible

$\varphi^{Présence} = 0$, le descripteur n'est pas renseigné soit dans le cas source ou soit dans le cas cible

Par conséquent, nous définissons une mesure de similarité globale qui est l'agrégation de ces différentes mesures et se traduit par la formule suivante :

$$M_R = \frac{\sum_{i=1}^{m} \varphi_i^{Valeur} \times \varphi_i^{Etat} \times \varphi_i^{Présence} \times \varphi_i^{M.F}}{\sum_{i=1}^{m} \varphi_i^{Présence}} \qquad (12)$$

Où *m* est le nombre de descripteurs de problème.

A partir de cette mesure, un ensemble de cas peut être choisi. Afin de sélectionner le cas source remémoré le plus adaptable, nous avons mis en place une mesure que nous appellerons « *mesure d'adaptation* » qui privilégiera les valeurs du mode de fonctionnement des descripteurs.

4.2.2. Mesure d'Adaptation (M_A)

La Mesure d'Adaptation (M_A) a pour objectif de choisir parmi les cas remémorés le cas le plus facilement adaptable lors de la phase d'adaptation. Cette mesure tient compte du mode de fonctionnement des composants en mettant l'accent sur les modes de fonctionnement anormaux dans le cas source et le cas cible en leur affectant un poids plus important. Cette importance sera caractérisée par le poids λ_i. Ce poids est important dans la détermination du composant défaillant. La présence de la valeur du descripteur est prise en compte ce qui permet de faciliter l'adaptation. Enfin, la valeur du descripteur est également prise en compte car plus la valeur du descripteur cible est proche de la classe de la valeur du descripteur source et plus ce descripteur est facilement adaptable [Haouchine et al., 2008b].

De ce fait, la Mesure d'Adaptation (M_A) est définie comme suit :

$$M_A = \frac{\sum_{i=1}^{m} \lambda_i . \times \varphi_i^{Présence} \times \varphi_i^{Valeur}}{\sum_{i=1}^{m} \varphi_i^{Présence}} \qquad (13)$$

Où λ_i représente le poids associé en fonction du mode de fonctionnement.

$$\begin{cases} \text{Si MF}(ds_i, dc_i) = \{\text{normal/normal}\} \rightarrow \lambda_i = 2^0 \\ \text{Si MF}(ds_i, dc_i) = \{\text{anormal/normal } \textit{ou} \text{ normal/anormal}\} \rightarrow \lambda_i = 2^1 \\ \text{Si MF}(ds_i, dc_i) = \{\text{anormal/anormal}\} \rightarrow \lambda_i = 2^2 \end{cases}$$

Les descripteurs qui sont en mode défaillant sont privilégiés en imposant le double du poids choisi car ils représentent les composants qui sont les plus susceptibles d'être défaillants dans l'espace solution. Par conséquent, le cas source possédant la plus grande valeur de la mesure d'adaptation parmi les cas sources remémorés sera le candidat choisi pour la deuxième étape.

Nous précisons entre autre, que la mesure d'adaptation (M_A) n'est calculée que par rapport aux descripteurs de la partie fonctionnelle du cas car c'est dans cette partie que le mode de fonctionnement des composants est exprimé.

4.3. Etape d'adaptation

Nous allons aborder la méthode ainsi que l'algorithme concernant la phase d'adaptation qui prend appui sur le modèle de contexte, sur le modèle hiérarchique des descripteurs et sur les relations de dépendance entre les différents descripteurs de problème et de solution.

4.3.1. Relations de dépendance

L'influence d'un ou de plusieurs descripteurs de problème sur un descripteur de solution est exprimée par les relations de dépendance. Une relation de dépendance est un triplet (ds_i, Ds_j, RD_{ij}). RD_{ij} donne le type de relation entre l'espace problème et l'espace solution pour un cas donné. RD_{ij} peut prendre trois valeurs possibles qui sont déterminées à partir d'un *modèle de contexte* et par une relation de *pertinence forte* : $RD_{ij} \subset$ {Forte ; Faible ; Pas de relation}.

Le modèle de contexte dans le cas du diagnostic technique concerne les relations de cause à effet entre les éléments susceptibles d'être défaillants. Ainsi, des relations de dépendance (RD) entre descripteurs de problème et de solution peuvent être mises en place.

Une relation de pertinence forte entre descripteurs de problème et de solution existe lorsqu'il y a une paire de cas dans la base de cas tels que les cas diffèrent exclusivement par la valeur du descripteur de problème pour deux valeurs différentes du descripteur de solution. Pour identifier ce type de relation, nous pouvons être amené à étudier les descripteurs de problème en relation avec la classe solution et appliquer un algorithme de filtrage de variables tel que STRASS (Strong Relevant Algorithm for Subset Selection) [Senoussi & Chebel-Morello, 2008], que nous avons développé au sein de notre équipe, ensuite en déduire les descripteurs de pertinence forte.

Ce qui nous amène à conclure que grâce aux caractéristiques citées ci-dessus, nous pouvons déterminer la nature de RD_{ij}, ce qui n'est pas le cas pour [Fuchs et al., 2000] qui supposent que ces dépendances sont connues à l'avance et se trouvent dans les cas sources.

Par ailleurs, la signification de chaque type (valeur) de RD_{ij} est comme suit :

RD_{ij} = Forte : ce type de relation existe lorsqu'un descripteur ds_i est fortement pertinent par rapport au descripteur Ds_j.

RD_{ij} = Faible : cette relation de dépendance est exprimée par la relation de cause à effet potentielle qui peut y avoir entre le descripteur de problème et les descripteurs de solution impliquant un ensemble de cas donné. Dans le cas du diagnostic technique, un modèle de cause à effet (ou modèle de contexte) permettra de définir ce sous ensemble de cas et la valeur des descripteurs de la relation.

RD_{ij} = Pas de relation : il y a une parfaite indépendance entre ds_i et Ds_j.

4.3.2. Algorithme d'adaptation

L'algorithme d'adaptation décrit ci-dessous adapte descripteur par descripteur. Il s'appuie sur le modèle de contexte de l'équipement à diagnostiquer, sur le modèle d'hiérarchie des composants et sur les relations de dépendance. Les différents types d'adaptation en l'occurrence l'adaptation substitutionnelle, transformationnelle et hiérarchique sont exploités dans cet algorithme [Haouchine et al., 2008b].

Nous avons identifié trois cas type d'adaptation à savoir :

- ***RD forte*** impliquant au moins un descripteur de solution avec un descripteur de problème appartenant à la ***même*** classe de fonctionnement ;
- ***RD forte*** impliquant au moins un descripteur de solution avec un descripteur de problème appartenant à des classes de fonctionnement ***différentes*** ;
- ***RD faible***.

Entrée : cas remémoré (ds_i^{rem} [7], Ds_j^{rem} [8])

Sortie : descripteurs solution du cas adapté Dc_j

Pour chaque « Ds_j^{rem} » **faire** // j = 1…m.

Créer une liste contenant les valeurs de RDji qui sont en relation avec les descripteurs de problème source « ds_i » → Sélection du couple (RD_{ji}, ds_i^{rem})

FinPour

Pour l'ensemble des couples (RD_{ji}, ds_i^{rem}) **faire**

Si (RD = forte) **alors**

Si (Ds_j^{rem} est de la même classe que ds_i^{rem}) **alors**

Dc_j ← ds_i^{rem} (Mettre la valeur de ds_i^{rem} dans Dc_j)

Sinon

Sélectionner le descripteur de problème « dc_{but} » faisant partie de la même classe hiérarchique que « Ds_j^{rem} »

Affecter la valeur de la solution du descripteur « dc_{but} »

Dc_j ← solution de dc_{but} (Mettre la valeur de la solution de dc_{but} dans Dc_j)

*Aller à **Fin Pour***

Fin Si

Si (RD = Faible) **alors**

Dc_j ← solution de dc_{but} (Mettre la valeur de la solution de dc_{but} dans Dc_j)

Fin Si

Si (RD = aucune) alors // Aucune relation existante

Aucune adaptation n'est faite

Dc_j ← aucune valeur

Fin Si

FinPour

Algorithme 4.1. Algorithme d'adaptation

[7] descripteurs de problème remémorés

[8] descripteurs de solution remémorés

Cet algorithme traite l'adaptation d'un descripteur à la fois. Après la remémoration lorsque nous sélectionnons un cas remémoré (ds_i^{rem}, Ds_j^{rem}) l'étape de l'adaptation s'enclenche. La première étape, l'étape d'initialisation permet de créer une liste de couples ayant une relation soit forte soit faible. Suivant l'intensité de la relation, le traitement diffère. Par conséquent la deuxième étape dépendra des valeurs de RD et des classes des descripteurs.

Si en parcourant la liste, on trouve une valeur de « *RD = forte* » alors nous sélectionnons le couple en question et on regarde la classe de « Ds_j^{rem} » et de « ds_i^{rem} ». S'ils ont la même classe parent, l'influence de cette substitution va être prise en compte dans « Ds_j^{rem} » pour attribuer cette nouvelle valeur à « Dc_j ».

Cependant, dans le cas où les deux descripteurs n'appartiennent pas à la même classe parent, alors il va y avoir une identification des descripteurs appartenant au contexte dans lequel le descripteur « dci » appartient. Nous regardons les différentes classes des différents descripteurs et on sélectionne le descripteur « dc_{but} » qui appartient à la même classe parent que Ds_j^{rem}. Nous appelons ce descripteur cible « dc_{but} ». Ensuite, la valeur de ds_i^{rem} va être déterminée qui va être par la suite être affectée à Dc_j.

Dans le cas où dans la liste il n'y a que la valeur « *RD = faible* », nous sélectionnons la classe parent du descripteur Ds_j^{rem}. Ensuite, nous identifions le descripteur « dc_i » appartenant à la même classe parent que « Ds_j^{rem} » qui va changer de statut (dci → dc_{but}). Après cela, la relation dc_{but} va influencer la transformation de la solution de Ds_j^{rem} qui va être affectée par la suite à « Dc_j ».

Enfin, lorsque toutes les valeurs de RD sont égales à « *pas de relation* » alors il n'y a pas d'adaptation de descripteurs.

5. Etude de la méthode de la remémoration guidée par l'adaptation sur un moteur à explosion

L'étude de la méthode de remémoration guidée par l'adaptation proposée ainsi que l'algorithme d'adaptation s'applique sur un moteur à explosion 1.5 dCi K9K 105ch de la société *Renault* disponible à l'adresse suivante : http://v3.renault.com/cfm/module-K9K/fr/index.html.

L'étude fonctionnelle et dysfonctionnelle de ce moteur présentée au chapitre 2 a conduit à une base de cas contenant vingt cas et deux modèles de connaissance : un modèle de contexte et un modèle de taxonomie des composants. Nous exploitons cette modélisation pour appliquer notre méthode de remémoration guidée par l'adaptation proposée. Tout d'abord, nous commençons par la mise en place des relations de dépendance et ensuite nous traitons les deux phases de remémoration et d'adaptation via trois exemples type d'adaptation.

5.1. Mise en place des relations de dépendance (RD)

Les relations de dépendance permettent d'établir un lien entre l'espace problème et l'espace solution d'un cas. Dans notre étude, les descripteurs, de problème et de solution, reflètent les composants et les classes des composants de l'équipement moteur.

La détermination de la nature des valeurs de « RD » est exploitée uniquement dans la partie fonctionnelle des cas.

La correspondance des différentes valeurs de « RD » est comme suit :

RD = Forte : cette relation est vérifiée lorsqu'il y a un descripteur « représentant un composant ou une classe de composant » dans l'espace problème qui influe directement sur un ou plusieurs descripteurs de l'espace solution. Cette influence se traduit par la répercussion du mode de fonctionnement anormal d'un ou plusieurs descripteurs de problème sur la détermination du composant défaillant représenté par un descripteur de solution.

Cette valeur de « RD » dépend de l'appartenance des composants aux classes de fonctionnement. Par conséquent, il y a deux valeurs possibles :

RD = Forte avec des classes différentes : lorsque des composants, appartenant à deux ou plusieurs classes différentes, se trouvent dans une même zone de défaillance.

RD = Forte avec des classes similaires : ce cas de figure se vérifie lorsque les composants concernés appartiennent à la même classe de fonctionnement.

RD = Faible : cette relation est vérifiée lorsqu'il y a des composants impliqués indirectement (car ils sont dans un mode de fonctionnement normal) mais qui permettent de déterminer la nature de la panne avec la combinaison des composants en mode défaillant. Ce qui signifie que ces composants représentés par les descripteurs de la partie problème ont une répercussion indirecte sur les descripteurs de la partie solution.

RD = Pas de relation : cette valeur concerne les composants qui ne sont pas du tout impliqués dans la détermination du composant défaillant.

5.2. Exemples d'illustration de la méthode de remémoration et d'adaptation

Nous allons illustrer dans cette section la méthode de remémoration guidée par l'adaptation ainsi que l'algorithme d'adaptation à travers trois exemples types. En effet, dans chaque exemple, nous allons aborder la remémoration en fonction de la mesure de remémoration (M_R) des cas sources pour chacun des cas cibles. Ensuite, nous calculons pour chaque cas source remémoré la valeur de la mesure d'adaptation (M_A) afin d'appliquer l'algorithme d'adaptation au cas le plus facilement adaptable. Pour chaque exemple, nous détaillerons les relations de dépendance entre les descripteurs de problème et les descripteurs de solution.

5.2.1. Premier exemple type d'adaptation « RD = Forte & même classe de fonctionnement »

Soit une panne survenue dans le moteur au niveau des *coussinets de vilebrequin* et représentée par le cas *cible 1* (cf. Figure 4.3).

	d1	d2	d3	d4	d5			d6			d7			d8			d9			d10			d11			d12		
Cible1	Tourne	Mauvais			Pompe injection	Non entraî-née	A				Bougies	Etin-celle	N				Filtre	Gaz circ-ule	N							Coussinets vilebrequin	Surf-ace rugu-euse	A
1 (RD forte)	Tourne	Mauvais			Pompe injection	Non entraî-née	A				Bougies	Etin-celle	N				Filtre	Gaz circ-ule	N							Vilebrequin	Mouv. cont-inu	N
4 (RD forte) & ≠	Tourne				Pompe injection	Entraî-née	N	Monoli-the	Air suffi-sant	N	Bougies	Etin-celle	N										Arbre à cames	Mouv. disc-ontin u	A			

Figure 4.3. Le cas source le plus adaptable au cas cible 1

Concernant la phase de remémoration, nous avons fixé un seuil de similarité de 60%.

Nous procédons maintenant au calcul de la mesure de remémoration (M_R). Les cas sources les plus similaires au cas cible 1 sont :

$$M_R(srce_1, cible_1) = \frac{\overbrace{(1\times1)}^{\varphi_i^{Présence}} + \overbrace{(1\times1)}^{\varphi_i^{Etat}} + \overbrace{(1\times1\times1\times1)}^{\varphi_i^{Valeur}} + (1\times1\times1\times1) + (1\times1\times1\times1) + \overbrace{(1\times0\times0.8\times0)}^{\varphi_i^{M.F}}}{6}$$

$$\underbrace{\quad}_{ds_1}\ \underbrace{\quad}_{ds_2}\ \underbrace{\quad\quad}_{ds_5}\ \underbrace{\quad\quad}_{ds_7}\ \underbrace{\quad\quad}_{ds_9}\ \underbrace{\quad\quad}_{ds_{12}}$$

$M_R(srce_1, cible_1) =$ **0.83**

$$M_R(srce_4, cible_1) = \frac{(1\times1) + (1\times0\times1\times0) + (1\times1\times1\times1)}{3} = 0.67$$

Maintenant, nous procédons au calcul de la mesure d'adaptation (M_A) aux deux cas sources les plus similaires au cas cible 1. Nous rappelons que la mesure M_A est calculée seulement par rapport aux descripteurs de la partie fonctionnelle du cas en l'occurrence du descripteur ds_5 jusqu'au descripteur ds_{12}.

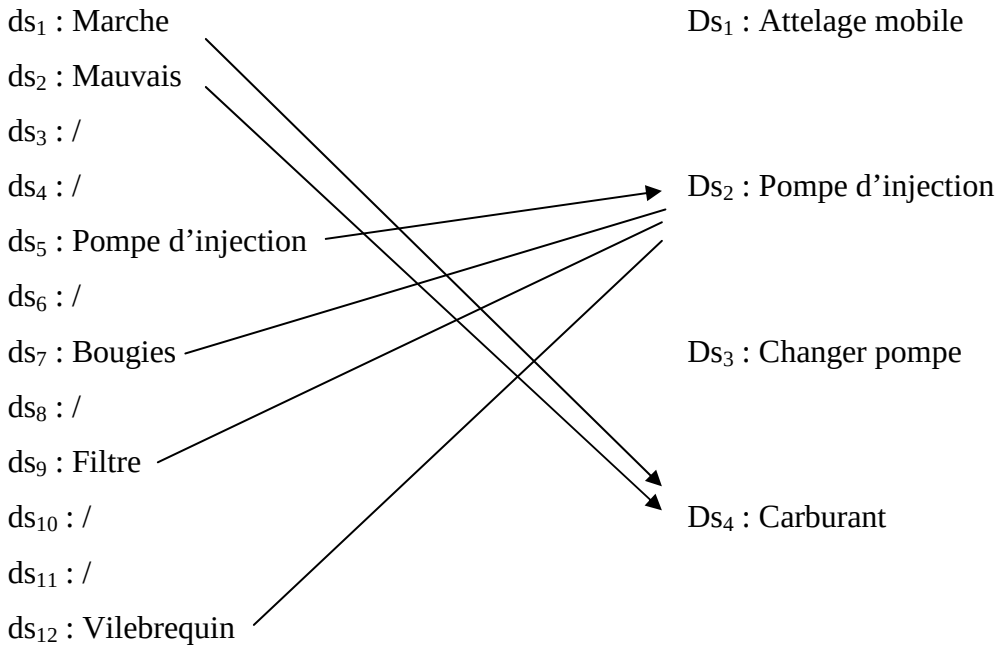
$$M_A(srce_1, cible_1) = \frac{\underbrace{(4\times1\times1)}_{ds_5} + \underbrace{(1\times1\times1)}_{ds_7} + \underbrace{(1\times1\times1)}_{ds_9} + \underbrace{(2\times1\times1)}_{ds_{12}}}{4} = \mathbf{2}$$

$M_A(srce_4, cible_1) = 1.5$

$M_A(srce_1, cible_1) > M_A(srce_4, cible_1)$, par conséquent, le cas source 1 est le plus facilement adaptable au cas cible 1.

Nous remarquons, entre autre, que dans cet exemple le cas source le plus similaire au cas cible 1 est également le plus facilement adaptable suivant sa valeur de M_A.

La représentation des relations de dépendance entre les descripteurs de problème et les descripteurs de solution du cas source 1 est la suivante :



Nous précisons que les flèches représentent les relations « RD = forte », les segments « RD = faible » et lorsqu'il n'y a rien qui relie les descripteurs entre eux alors nous sommes dans le cas de « RD = pas de relation ».

Nous rappelons que pour l'étape d'adaptation, les quatre premiers descripteurs sont destinés à déterminer la zone de la panne qui est exprimée par le descripteur Ds_4. Concernant la détermination du composant défaillant, ce sont les descripteurs « ds_5… ds_{12} » qui sont concernés et ils sont liés directement au descripteur Ds_2. Les descripteurs Ds_1 et Ds_3 sont la conséquence de la valeur du descripteur Ds_2.

Nous pouvons constater dans un premier temps que la valeur de RD du couple (Ds_2, ds_5) est « RD = Forte ». Dans un deuxième temps, les deux descripteurs Ds_2, ds_5 représentent le même composant « *pompe d'injection* » et donc ils ont la même appartenance de classe. De ce fait, nous sommes dans le cas de figure de « *RD = Forte et même classe de fonctionnement* ».

En appliquant l'algorithme d'adaptation, nous obtenons :

- Substituer la valeur du descripteur ds_5^{rem} qui est en mode « *anormal* » par la valeur du descripteur dc_5 = pompe d'injection ;
- Comme les deux valeurs ds_5^{rem} et dc_5 sont égales alors on affecte directement la solution de Ds_2 à la solution cible Dc_2 : *Dc_2 = Pompe d'injection « ne livre pas de mazout »*.

La solution est donc la suivante : *changer la pompe d'injection qui ne livre pas le mazout se trouvant dans la zone « carburant »*.

5.2.2. Deuxième exemple type d'adaptation « RD = Forte & différentes classes de fonctionnement »

Soit une panne survenue dans le moteur au niveau du *joint de collecteur* et représentée par le cas *cible 2* (cf. Figure 4.4).

	d1	d2	d3	d4	d5			d6			d7			d8			d9			d10			d11			d12		
Cible2	Arrêt		Moyenne					Monolithe	Air insuffisant	A	Bougies	Etincelle	N	Joint de collecteur	Etanche	N				R.G.E	Bonne pression	N	Arbre à cames	Mouv. Continu	A	DVA	Mouv. continu	N
1 (RD forte)	Tourne	Mauvais			Pompe injection	Non entraînée	A				Bougies	Etincelle	N				Filtre	Gaz circule	N							Vilebrequin	Mouv. continu	N
5 (RD forte) & ≠	Arrêt		Moyenne					Pot catalytique	Etanche	N	Bougies	Etincelle	N	Joint de collecteur	Etanche	N				Vanne de recirculation	Passage d'air	N	Poussoirs	Mouv. dévié axe	A	DVA	Mouv. continu	N
16 (RD forte) & ≠	Arrêt		Haute	Partie basse	Pompe injection	Entraînée	N	Monolithe	Air suffisant	N	Bougies	Etincelle	N				Turbocompresseur	aucun bruit	N				Arbre à cames	Mouv. Continu	A			

Figure 4.4. Le cas source le plus adaptable au cas cible 2

- ***Calcul de la mesure de remémoration***

$M_R(srce_1, cible_2) = 0.60$

$M_R(srce_5, cible_2) = \mathbf{0.625}$

$M_R(srce_{16}, cible_2) = 0.60$

- ***Calcul de la mesure d'adaptation***

$M_A(srce_1, cible_2) = 1$

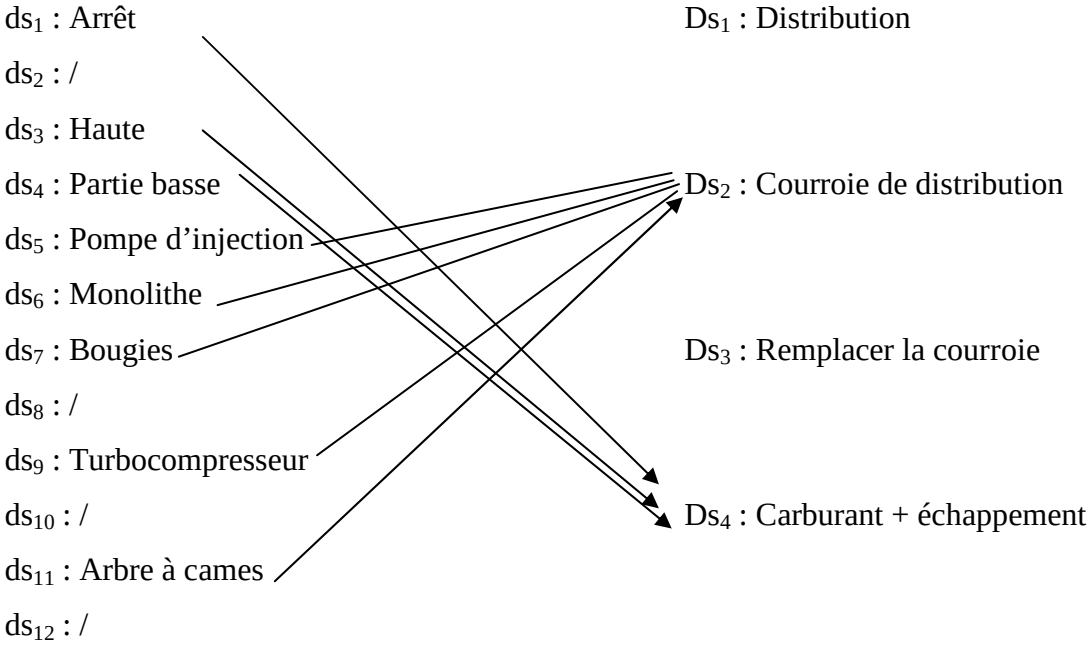
$M_A(srce_5, cible_2) = 1.43$

$M_A(srce_{16}, cible_2) = \mathbf{2.33}$

Par conséquent, le cas source le plus proche au cas cible 2 est le cas source 16.

Nous pouvons constater que dans cet exemple, le cas source le plus similaire n'est pas celui qui a la plus grande valeur de M_A pour qu'il soit sélectionné dans la phase d'adaptation.

La représentation des relations de dépendance entre les descripteurs de problème et les descripteurs de solution du cas source 16 est la suivante :



Nous constatons que la valeur de RD du couple (Ds_2, ds_{11}) est « RD = Forte » et que les composants « arbre à cames » et « courroie de distribution » n'appartiennent pas à la même famille. Ce qui nous amène à dire que nous sommes dans le cas de figure de « RD = Forte & différentes classes de fonctionnement ».

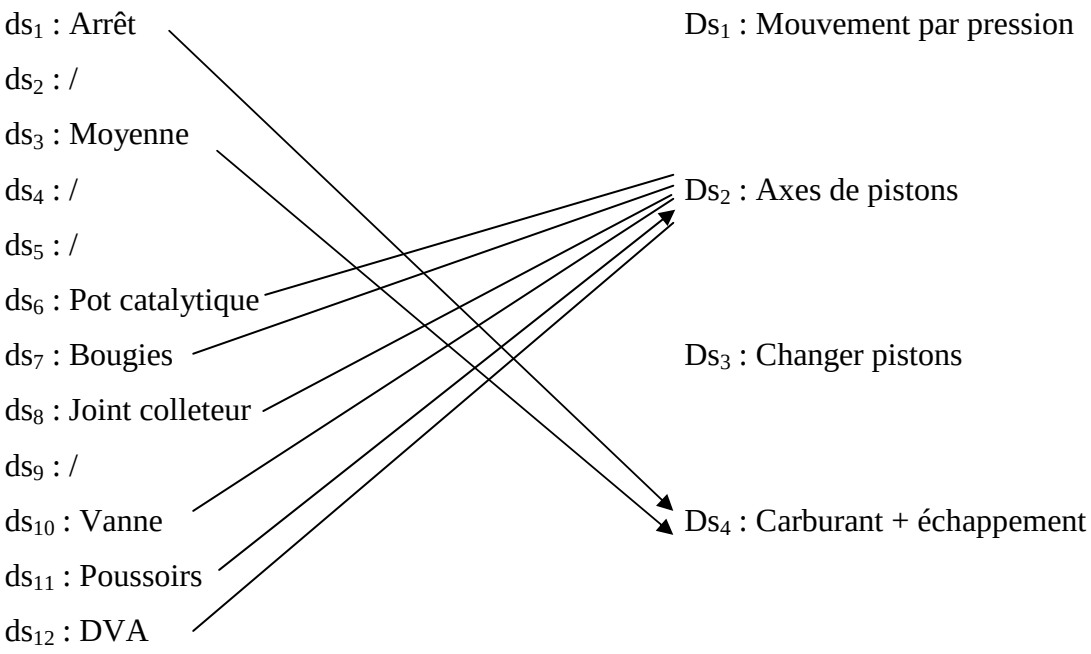
En appliquant l'algorithme d'adaptation, nous obtenons :

- La classe du descripteur solution source Ds_2^{rem} est « Distribution » ;
- Le composant « arbre à cames » du descripteur cible « dc_{11} » (qui correspond au descripteur « ds_{11} » qui est en mode anormal) se trouve dans la zone commune « Carburant + échappement ». Dans cette zone, nous pouvons trouver plusieurs composants. Parmi ceux qui sont représentés dans le cas cible sont : monolithe, bougies, joint de collecteur, RGE et DVA ;
- Le composant « joint de collecteur » appartient à la même classe de Ds_2^{rem} qui est « Distribution » ;
- Substituer la valeur « arbre à cames » du descripteur « dc_{11} » par la valeur « joint de collecteur » dans « ds_{11} » qui va se répercuter sur la valeur du descripteur « Ds_2^{rem} ». De ce fait, la nouvelle valeur de Ds_2^{rem} est la suivante : Ds_2^{rem} = Joint de collecteur non serré ;
- Affecter cette valeur à « Dc_2 » : Dc_2 = Joint de collecteur non serré.

La solution est donc la suivante : *resserrer le joint de collecteur qui est non serré et qui se trouve au niveau de l'intersection des deux circuits : carburant et échappement*.

Cependant, Si on avait choisi le cas source 5, alors son adaptation serait comme suit.

Tout d'abord, la représentation des relations de dépendance est la suivante :



Nous constatons que la valeur de RD du couple (Ds_2, ds_{11}) est « RD = Forte » et que les composants « poussoirs » et « axes de pistons » n'appartiennent pas à la même famille. De ce fait, on se place dans le cadre de « RD = Forte & différentes classes de fonctionnement ».

Ainsi, les étapes d'adaptation sont les suivantes :

- La classe du descripteur solution source Ds_2^{rem} est « Pression » ;
- Le composant « arbre à cames » du descripteur cible « dc_{11} » (qui correspond au descripteur « ds_{11} » qui est en mode anormal) se trouve dans la zone commune « Carburant + échappement ». Dans cette zone, nous trouvons plusieurs composants : RGE, DVA, monolithe, joint de collecteur, etc. ;
- Il n'y a aucun composant qui appartiennent à la classe « Pression » ;
- Les composants « DVA, monolithe, joint de collecteur » appartiennent à la classe « Distribution » qui est du deuxième niveau dans la taxonomie des composants ;
- La solution cerne un ensemble de composants et non un seul composant. Pour se faire, l'expert doit préciser quel est le composant qui est défaillant ;

De ce fait, afin d'adapter le cas source 5, l'algorithme a été obligé de faire une étape en plus qui consiste à accéder à la classe de niveau 2 et cette adaptation n'aboutit pas à la proposition d'un composant mais à un ensemble de composants et laisse à l'expert le soin de déterminer le composant défaillant parmi les composants sélectionnés.

Il est à noter que le composant proposé dans la première adaptation fait partie de l'ensemble des composants proposés dans la deuxième adaptation.

Par conséquent, le cas source 16 est plus facilement adaptable que le cas source 5.

5.2.3. Troisième exemple type d'adaptation « RD = Faible »

Soit une panne survenue dans le moteur au niveau du *vilebrequin* et représentée par le cas *cible 3* (cf. Figure 4.5).

	d1	d2	d3	d4	d5			d6			d7			d8			d9			d10			d11			d12		
Cible3	Tourne	Mauvais			Pompe injection	Entraî-née	N				Bougies	Etin-celle	N				Turbo-compre-sseur	Pré-sence bruit	A				Courroie de distribution	Serré e	N			
4 (RD forte) & ≠	Tourne				Pompe injection	Entraî-née	N	Monoli-the	Air suffi-sant	N	Bougies	Etin-celle	N										Arbre à cames	Mouv. disc-ontin u	A			
6 (RD faible)	Tourne	Mauvais			Rampe injection	Abse-nce parti-cule	N				Bougies	Etin-celle	N				Turbo-compre-sseur	aucu n bruit	N				Courroie de distribution	Serré e	N			
7 (RD Forte)	Tourne	Mauvais			Pompe injection	Entraî-née	N				Bougies	Etin-celle	N				Compre sseur	Gaz circ-ule	N							Vilebrequin	Mouv. disc-ontinu	A
10 (RD faible)	Tourne	Mauvais												Collecteur	Dég-agé	N				Vanne de reci-rculation	Pass-age d'air	N				Coussinets vilebrequin	Surf-ace lisse	N
15 (RD forte) & ≠	Tourne	Mauvais	-	-	Rampe injection	Prése-nce parti-cule	A	Pot catalyt-ique	Etan-che	N	-	-	-	Joint de collecteur	Etan-che	N	-	-	-	-	-	-	Courroie de distribution	Serré e	N	DVA	Mouv. contin u	N

Figure 4.5. Le cas source le plus adaptable au cas cible 3

- ***Calcul de la mesure de remémoration***

$M_R(srce_4, cible_3) = 0.75$

$M_R(srce_6, cible_3) = 0.67$

$M_R(srce_7, cible_3) = \mathbf{0.80}$

$M_R(srce_{10}, cible_3) = 0.67$

$M_R(srce_{15}, cible_3) = 0.67$

- ***Calcul de la mesure d'adaptation***

$M_A(srce_4, cible_3) = 1.2$

$M_A(srce_6, cible_3) = 1.2$

$M_A(srce_7, cible_3) = 1.33$

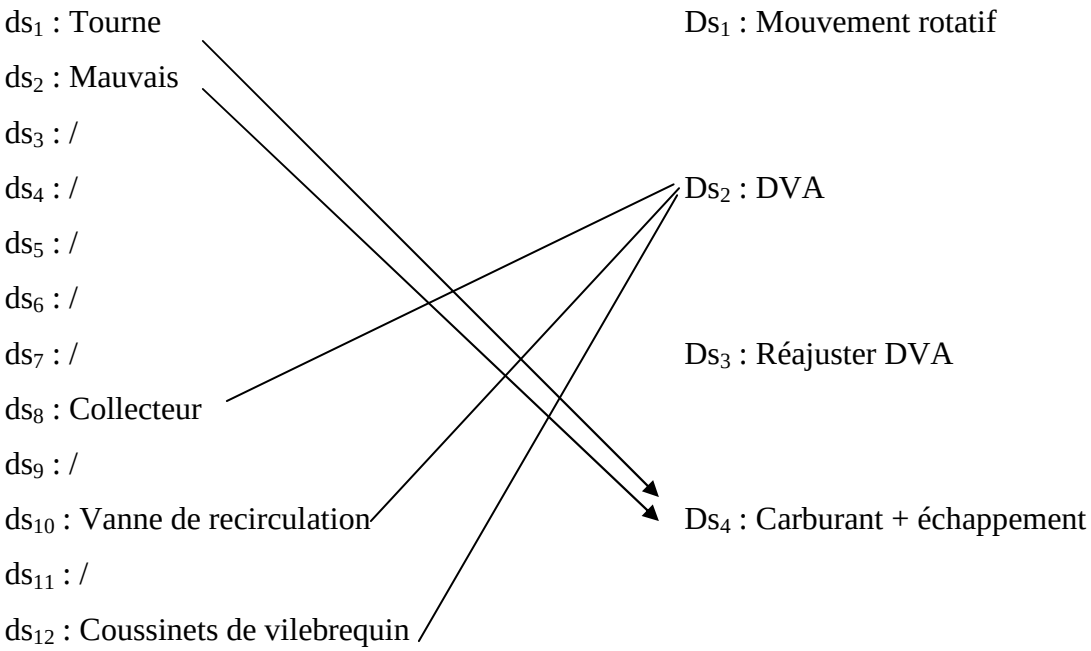
$M_A(srce_{10}, cible_3) = \mathbf{2}$

$M_A(srce_{15}, cible_3) = 1.6$

Par conséquent, le cas source choisi pour la phase d'adaptation est le cas source 10.

Nous pouvons également constater que dans cet exemple, le cas source le plus similaire n'est pas celui qui a la plus grande valeur de M_A pour qu'il soit sélectionné dans la phase d'adaptation.

La représentation des relations de dépendance entre les descripteurs de problème et les descripteurs de solution du cas source 10 est la suivante :



Nous constatons qu'il n'y a que des valeurs de RD faible. Ce qui nous amène au cas de figure suivant : « RD = Faible ».

En appliquant l'algorithme d'adaptation, nous obtenons :

- La classe du descripteur de solution source « Ds_2^{rem} » est « mouvement rotatif » ;
- Le contexte des composants du cas cible « pompe d'injection, bougies, turbocompresseur et courroie de distribution » est l'intersection des deux circuits du carburant et de l'échappement ;
- Dans cette zone, le composant appartenant au même contexte et qui appartient à la même classe de Ds_1^{rem} (Mouvement rotatif) est le composant : *vilebrequin* ;
- Substitution de la valeur « DVA » par la valeur « vilebrequin » dans le descripteur Ds_2^{rem} et mettre la valeur adéquate : Ds_2^{rem} = vilebrequin endommagé ;
- Affecter cette nouvelle valeur au descripteur Dc_2.

De ce fait, la solution donnée au cas cible serait de *changer le vilebrequin qui est endommagé se trouvant dans l'intersection des deux circuits : carburant et échappement.*

Suite à ces trois exemples, nous avons pu voir que le cas source le plus similaire au cas cible n'est pas forcément celui qui est sélectionné pour la phase d'adaptation.

5.3. Evaluation

Afin de procéder à l'évaluation de cette phase d'adaptation, nous réalisons des tests selon un protocole spécifique. Ce protocole concerne la division de la base de cas du moteur à explosion en deux sous-ensembles. Le premier sous ensemble contiendra 40 % des cas tirés aléatoirement de la base de cas. Ce sous-ensemble constituera la base de cas initiale (BCi). Cette dernière contiendra donc 8 cas. Les 60 % des cas restants (12 cas) constitueront le sous-ensemble cible en leur enlevant la partie solution qu'on appellera base cible (Bcib). Les cas de la base cible (Bcib) vont être soumis à la base de cas initiale (BCi) pour donner la solution adéquate.

Le but de ce protocole d'évaluation de notre méthode est de prouver la faisabilité de celle-ci en calculant la précision (accuracy) de la base de cas initiale (BCi). Afin de réaliser ce calcul, nous appliquerons notre méthode d'adaptation qui est basée sur les modèles de connaissance, les mesures utilisées dans la phase de remémoration et les relations de dépendance exploitées par l'algorithme d'adaptation.

Notre méthode va être comparée avec une méthode qui n'utilise pas d'adaptation. Nous indiquons que la précision (accuracy) concernera la détermination de la bonne classe de défaillance. Les résultats issus des deux méthodes sont illustrés sur la Figure 4.6.

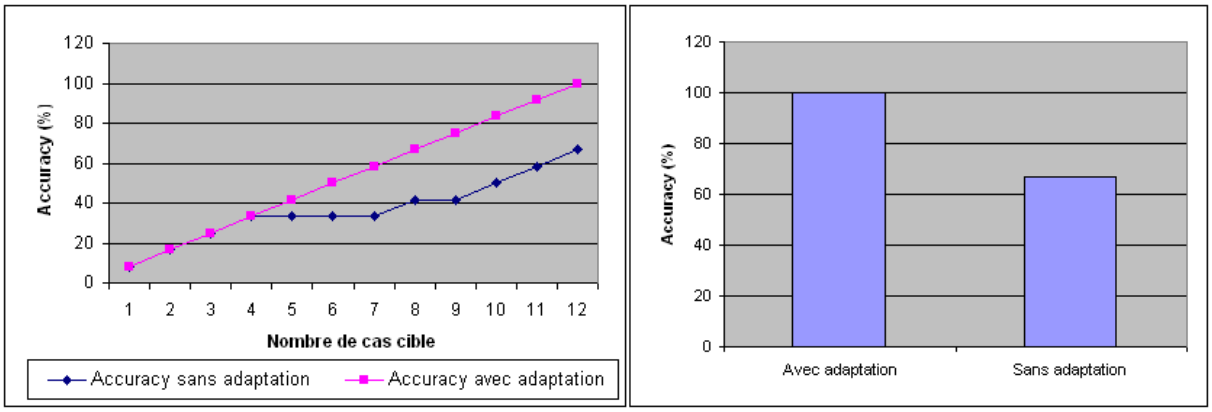


Figure 4.6. Evolution de l'Accuracy suivant le nombre de cas dans la base cible (Bcib) (figure gauche), Accuracy avec notre méthode d'adaptation et sans adaptation (figure droite)

La Figure 4.6 (droite) montre d'une part que la méthode d'adaptation mise en place a réussi une adaptation à 100% sur l'ensemble des cas cible de la base (Bcib) présentés à la base de cas initiale (BCi). D'autre part, comme cela est montré sur la Figure 4.6 (gauche) qui est le

résultat logique de la précédente constatation, les cas cibles sont adaptés avec succès les uns après les autres avec la méthode d'adaptation mise en place.

Cependant, nous constatons que les résultats obtenus sans méthode d'adaptation sont plutôt mitigés. En effet, nous obtenons une adaptation de seulement de 66,67% sur l'ensemble des cas cibles avec une évolution non constante (cf. Figure 4.6 gauche) concernant la résolution des cas cibles.

Dans ce protocole de mesure, les cas cibles correspondent aux cas décrits par les modèles. Une politique de test de plus grande envergure en grandeur nature ne donnerait pas une accuracy de 100%. Nous envisageons en perspective d'implémenter cette méthode sur un cas d'étude grandeur réelle.

Toutefois, d'après les résultats obtenus, nous pouvons conclure que le RàPC peut raisonner à partir d'un nombre restreint de cas, en prenant appui sur des modèles de connaissance et sur la phase d'adaptation ajustée à l'application considérée. De plus, pendant la phase de remémoration, qui utilise uniquement une mesure de similarité ne tenant pas compte de l'effort d'adaptation, ne va pas sélectionner les cas les plus facilement adaptables et entre autre les cas qui ont les mêmes classes. De ce fait, sans adaptation, lorsque la classe du cas remémoré ne convient pas, cela engendre une mauvaise classification qui se répercute sur le taux de bonne classification.

6. Conclusion

Nous nous sommes intéressés dans ce chapitre à l'étude d'un système de RàPC orienté connaissance et plus précisément aux phases de remémoration et d'adaptation. En effet, nous avons établi un état de l'art sur ces deux phases aux sections 2 et 3. Nous avons tenu compte, à partir des travaux de [Smyth & Keane, 1998], du lien entre ces deux phases. En effet, au début des années 90, plusieurs auteurs ont consacré une étude sur la dépendance entre la remémoration et l'adaptation. Nous avons également constaté que la mesure de similarité ainsi que les techniques de remémoration et d'adaptation utilisées dépendent essentiellement de la représentation du cas.

Dans cette optique, nous avons proposé, à la section 4, une méthode de remémoration guidée par l'adaptation sur un système de diagnostic par RàPC orienté connaissance. En effet, ce système dispose d'une base de cas qui est associée à des modèles de connaissance à savoir un modèle de hiérarchie des composants du système à diagnostiquer et un modèle de contexte

reflétant les relations de cause-à-effet entre les composants de ce système. Les deux phases de remémoration et d'adaptation manipulent un cas qui est adapté au diagnostic avec toutes les caractéristiques qui en découlent. Par ailleurs, nous avons proposé la méthode de remémoration guidée par l'adaptation à travers une mesure de remémoration (M_R) qui est associée à une mesure d'adaptation (M_A) afin de sélectionner parmi les cas sources remémorés, celui qui est le plus facilement adaptable. Les deux mesures prennent appui sur les différentes caractéristiques du cas de diagnostic. Enfin, nous avons proposé un algorithme d'adaptation fondé sur les relations de dépendance entre l'espace problème et l'espace solution du cas. Ces relations de dépendance permettent de choisir les étapes d'adaptation à appliquer suivant leurs valeurs. Elles sont également liées à l'appartenance des classes de descripteurs. Cette appartenance aux classes est primordiale dans le choix de la façon d'adapter le cas source candidat.

Nous avons appliqué notre méthode de remémoration guidée par l'adaptation sur un équipement industriel (moteur de type 1.5 dCi K9K 105ch de la société « Renault »). Nous avons ainsi montré, à travers trois cas types, que les cas les plus similaires au cas cible ne sont pas forcément ceux qu'on choisit lors de la phase d'adaptation.

Notre première étude a conduit à prouver l'applicabilité de notre méthode sur le moteur à explosion. En effet, le protocole de test s'est fait sur une base de cas existante. Nous prévoyons comme perspective de mener des tests de plus grande envergure.

Nous appliquerons la méthode proposée sur un système industriel supervisé SISTRE (Supervised Industrial System of pallets TransFer) développée au chapitre suivant.

Chapitre 5

Application

1. Introduction

Nous avons mis en place deux systèmes d'aide au diagnostic par RàPC sur un système industriel, à savoir : un système orienté *mining* et un système orienté *connaissance*. Le premier type de système dispose d'une représentation de cas simple et trivial et exploite une base de cas contenant toute la connaissance du système à diagnostiquer. Nous avons proposé pour ce type de système, au chapitre 3, une méthode de maintenance de la base de cas qui la structure et l'auto-incrémente. Concernant le deuxième type de système, nous avons proposé, au chapitre 4, une méthode de remémoration guidée par l'adaptation ainsi qu'un algorithme d'adaptation pour les cas de diagnostic enregistrés dans la base de cas.

Ces deux systèmes d'aide avec leurs contributions respectives sont appliqués à un système industriel supervisé de transfert de palettes SISTRE (Supervised Industrial System of pallets TransFer). Ce chapitre est organisé en deux grandes parties :

La première partie dessine le cadre général de l'application, en décrivant l'équipement industriel SISTRE et en détaillant ses différents composants à la section 2. Cet équipement est composé de cinq stations identiques, par conséquent, nous nous intéressons à la décomposition structurelle et fonctionnelle d'une station.

La seconde partie, que nous abordons à la section 3, illustre l'étude de nos deux principales contributions à savoir : la maintenance de la base de cas pour les systèmes orientés mining et la remémoration guidée par l'adaptation dédiée aux systèmes orientés connaissance. Concernant la première contribution, nous décrivons tout d'abord la mise en place du système orienté *mining* avec la représentation du cas, l'élaboration du cas cible et la phase de remémoration. Ensuite nous appliquons la méthode de maintenance et d'auto-incrémentation de la base de cas associée à SISTRE. Concernant le système orienté *connaissance*, nous détaillons la structure du cas adapté au diagnostic et la base de cas qui reflète une analyse dysfonctionnelle du système associée à deux modèles de connaissance : le modèle de contexte et le modèle hiérarchique des composants. Ensuite, nous appliquons notre méthode de remémoration guidée par l'adaptation ainsi que l'algorithme d'adaptation à travers trois exemples types de diagnostic.

2. Présentation de SISTRE

Nous présentons dans cette section un système industriel supervisé de transfert de palettes SISTRE (Supervised Industrial System of pallets TransFer) en donnant une vue globale, son fonctionnement ainsi que les composants d'une de ses stations.

2.1. Aspect général de SISTRE

Le système de transfert de palettes SISTRE, schématisé sur la Figure 5.1, est un îlot flexible d'assemblage organisé en double anneau et est constitué d'un système de transfert de palettes et de cinq stations de travail identique.

Ce transfert flexible est une installation automatisée, équipée d'automates de la marque télémécanique (Groupe Schneider). Il permet de définir des gammes d'assemblage en déplaçant des palettes sur différents postes équipés de robots.

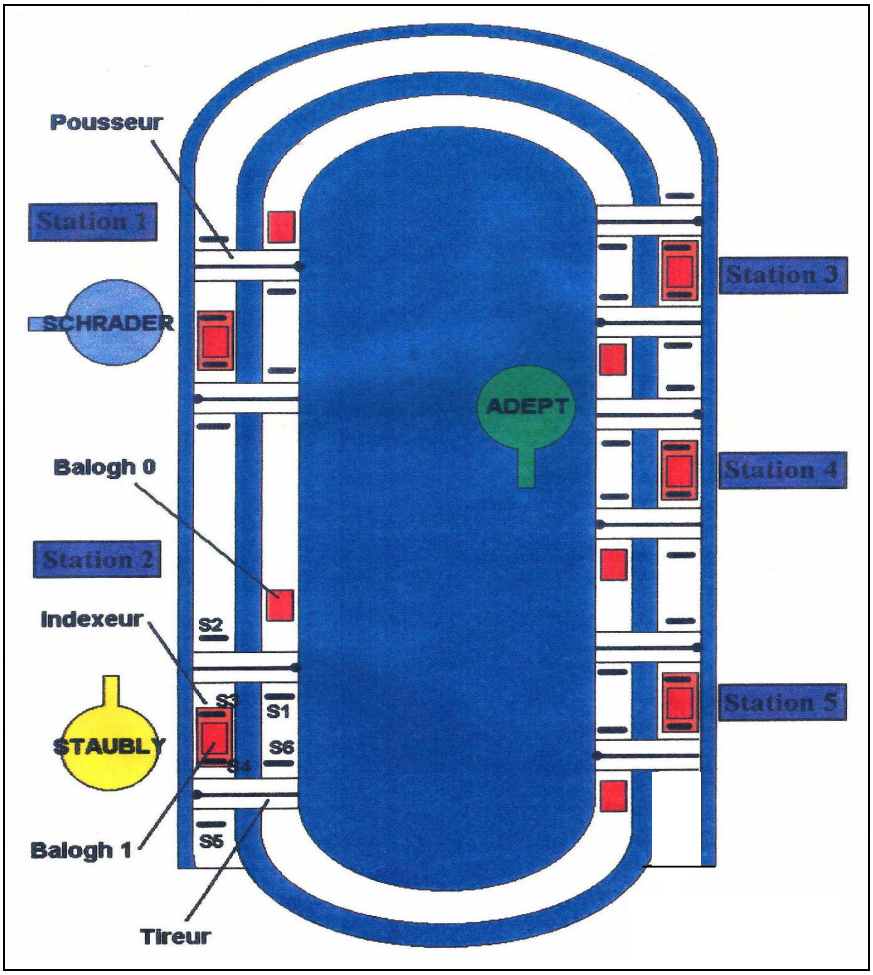


Figure 5.1. Ilot flexible d'assemblage SISTRE

L'assemblage peut se faire sur des pignons pour former un mot ou sur des lego pour faire une construction. Les palettes sont véhiculées sur l'anneau intérieur permettant ainsi le transit entre les différentes cellules. Lorsque l'une des palettes doit subir une opération de la part d'un robot (information lue sur l'étiquette de la palette), cette dernière est déviée sur l'anneau extérieur où se trouve le poste de travail concerné. Il y a quatre robots qui sont disposés aux stations 1, 2, 4 et 5. Ces robots sont chargés de réaliser des tâches d'assemblage ou de manipulation d'objets disponibles sur la palette. A la station 1 se trouve le robot de marque Schrader qui est équipé d'une pince qui peut saisir ou déposer des pions sur les palettes à une position définie dans la gamme d'assemblage. Au niveau des stations 2 et 4 se trouve le robot de marque « Stâubli RX 90 » et de marque « Adept One » respectivement qui sont équipés d'un préhenseur capable de saisir des Legos ou des pions. Enfin, le robot de marque « Adept Cobra 600 » qui se trouve à la station 5 est équipé d'un capteur de force.

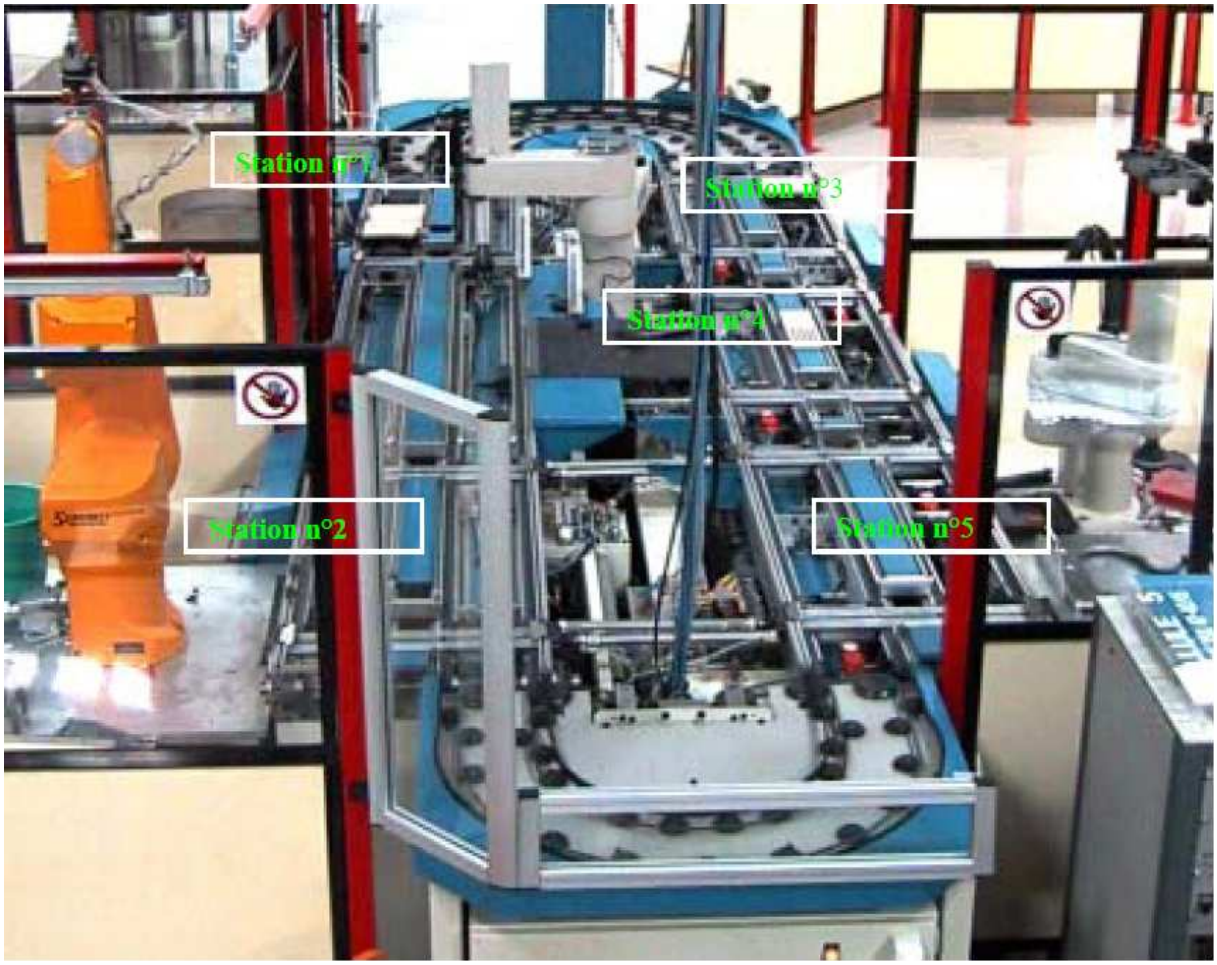


Figure 5.2. Vue d'ensemble du transfert SISTRE

La disposition des cinq stations sur le transfert SISTRE est montrée sur la Figure 5.2. Chacune de ces stations disposent d'un automate et fonctionnent indépendamment les unes des autres.

Le déplacement de la palette au sein de SISTRE représente le flux sur lequel nous allons prendre appui lors de la mise en place de notre système de diagnostic par RàPC. Ce déplacement est assuré par frottement sur des courroies qui elles-mêmes sont entraînées par des moteurs électriques. Les palettes sont munies d'une étiquette magnétique qui leur sert de « mémoire embarquée ». Ces mémoires (étiquettes) peuvent être lues dans chaque station grâce à des plots magnétiques de lecture/écriture, appelées « BALOGH », qui donnent la possibilité de mémoriser une gamme d'assemblage de produits. De ce fait, ces étiquettes permettent donc de déterminer le cheminement des palettes à travers les stations de SISTRE.

2.2. Composition d'une station

Nous avons mentionné précédemment que SISTRE comportait cinq stations identiques. Nous allons décrire l'une de ces stations qui est illustrée sur la Figure 5.3.

Chaque station dispose d'une partie opérative et d'une partie commande. La partie commande concerne les automates de type TSX 47.40, le langage de programmation employé tel que PL7 ou V+ sous l'environnement XTEL ainsi que le système d'exploitation utilisé tel que OS2. La partie opérative comporte des actionneurs pneumatiques et électriques, des robots (qui ont été présentés à la section 2.1), des détecteurs magnétiques Balogh et des détecteurs de présence.

- Les actionneurs pneumatiques

Il y a trois types d'actionneurs pneumatiques :

- Un pousseur, à l'entrée de la station, dévie les palettes de l'anneau intérieur vers l'anneau extérieur en vue d'un traitement sur les postes ;
- Un tireur, à la sortie de la station, ramène les palettes sur l'anneau intérieur si elles sont à la limite du convoyeur ou si elles n'ont pas à travailler sur la prochaine station ;
- Un indexeur permet de bloquer les palettes lors de l'opération d'un robot et assure un positionnement précis de l'étiquette magnétique par rapport au plot de lecture/écriture Balogh.

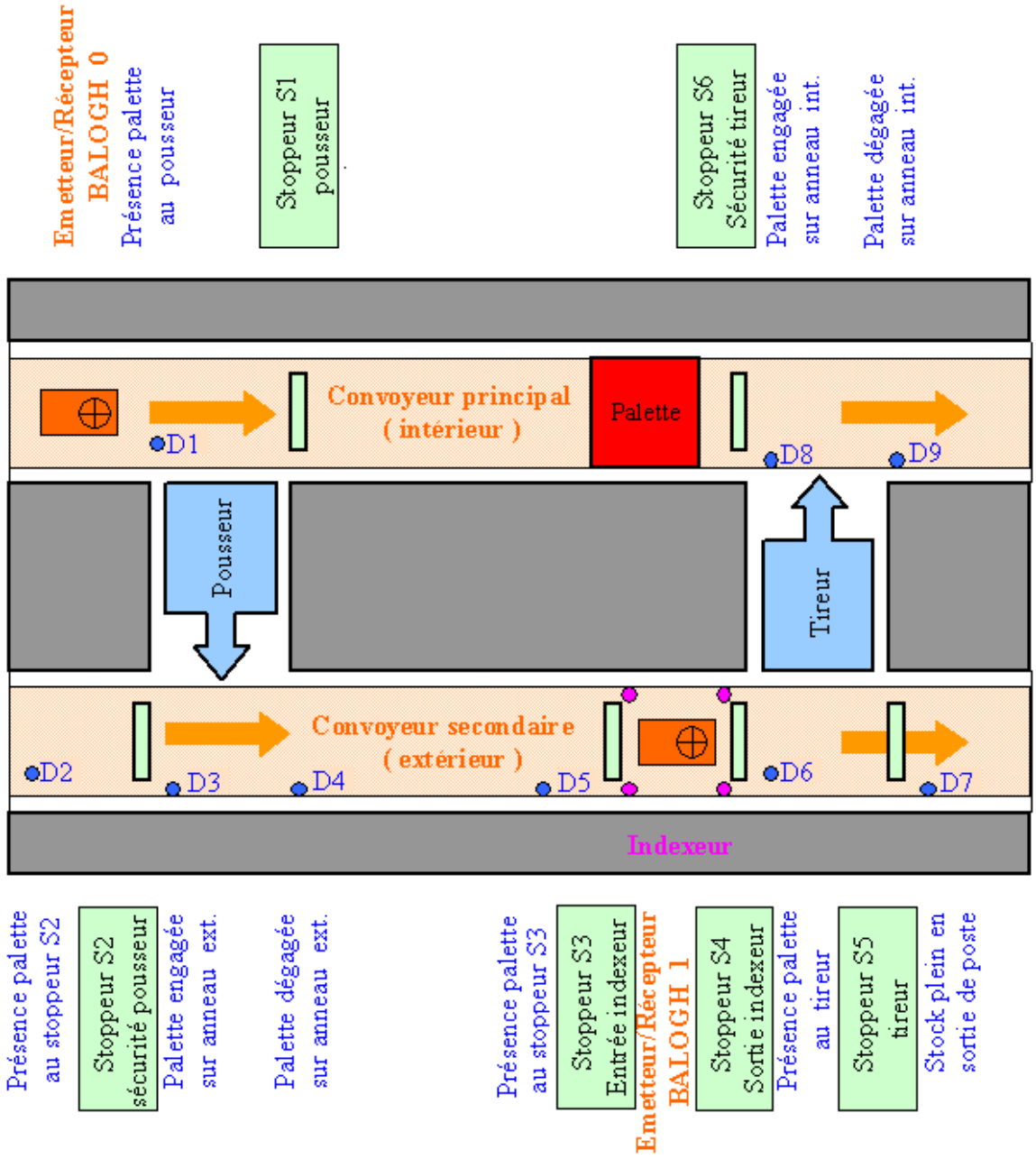


Figure 5.3. Composition d'une station du transfert SISTRE

- Les actionneurs électriques

- 6 stoppeurs (S1 .. S6) permettent de bloquer ou de laisser passer les palettes en différents endroits en fonction du routage de celles-ci, de leur environnement, de l'état des actionneurs, de l'état des robots …

- Les détecteurs magnétiques BALOGH

Ces détecteurs permettent la lecture et l'écriture des informations sur les étiquettes magnétiques des palettes. Il y a deux types de Balogh :

- La première (Balogh 0) se trouve à l'entrée de chaque station (sur l'anneau intérieur) et elle dispose que de la fonction « lecture ». En effet, elle ne fait que la lecture à la volée c'est à dire que la palette n'est pas arrêtée. Elle sert à déterminer si la palette doit seulement transiter par la station ou entrer ;
- La seconde (Balogh 1) se trouve au niveau du poste de travail (anneau extérieur). Elle permet d'effectuer des lectures et des écritures des paramètres relatifs aux opérations sur la bande magnétique de la palette. Ceci nécessite le blocage de la palette sur la tête magnétique.

Ces étiquettes permettent donc de déterminer le cheminement et le transit des palettes entre les différentes stations. Lorsque l'une des palettes doit subir une opération de la part d'un robot (information lue sur l'étiquette de la palette), cette dernière est déviée sur l'anneau extérieur où se trouve le poste de travail concerné.

- Les détecteurs de présence

Chaque station contient neuf capteurs inductifs (D1..D9) permettant la détection de la présence ou du bon positionnement de la palette. En effet les palettes sont munies de deux éléments métalliques (la barre et le pion) positionnés comme le montre la Figure 5.4.

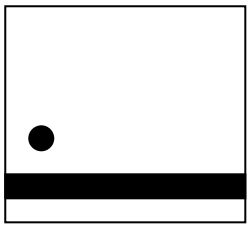


Figure 5.4. Position des parties métalliques sous la palette

La barre permet de détecter le passage d'une palette, le pion permet de s'assurer de sa bonne position (au niveau des actionneurs pneumatiques).

2.3. Fonctionnement d’une station

Lorsqu'une palette entre dans une station, le plot de lecture à la volée (Balogh 0) lit au passage, sur l'étiquette de la palette, si la station doit réaliser la prochaine opération à effectuer sur la palette au niveau du poste. Si ce n'est pas le cas ou si la lecture a mal fonctionné, alors la palette continue son chemin sur l'anneau intérieur. Le détecteur D1 détecte la présence de la palette qui va faire baisser le stoppeur S6 qui était en position initiale élevée (le stoppeur S1 est en position initiale basse). Ensuite, la palette va passer par les détecteurs D8 et D9 qui ont pour fonction la signalisation de sa présence. Dans le cas où la palette doit entrer dans l’anneau extérieur, le stoppeur S1 se met en position élevée pour la bloquer, ensuite le pousseur est chargé de la déplacer de l’anneau intérieur vers l’anneau extérieur. Une fois la palette arrivée, le détecteur D3 détecte sa présence. La palette va ensuite avancer grâce au convoyeur extérieur, en passant par les détecteurs D4 et D5 et en faisant baisser le stoppeur S3, jusqu’à ce qu’elle arrive au niveau de l’indexeur. La palette est ainsi indexée et le robot peut donc procéder à des manipulations selon les directives que comporte la palette. Le second détecteur magnétique (Balogh 1) est le composant qui est chargé de la lecture, sur l'étiquette, des paramètres de l'opération à réaliser, c'est-à-dire le numéro de la tâche en cours, le numéro de la palette, les paramètres de position… Puis, quand l'automate est informé de la fin de l'opération, le résultat et la durée de l'opération sont écrits sur l'étiquette, la palette est désindexée et le stoppeur S4 est baissé. La palette quitte alors la zone poste soit par l'anneau extérieur « si possible » si la prochaine opération est sur la station suivante, soit par l'anneau intérieur grâce au tireur.

3. Mise en place du système de diagnostic par le RàPC sur SISTRE

Après avoir décrit SISTRE, nous allons maintenant vous présenter nos deux systèmes d’aide au diagnostic par RàPC sur ce système de transfert de palettes.

En effet, deux types de systèmes ont été étudiés, à savoir : un système orienté mining et un système orienté connaissance. Ces deux types de systèmes diffèrent principalement par rapport à leur représentation du cas et aux connaissances manipulées. Dans le premier type, les cas sont décrits d’une façon détaillée et les descripteurs représentent les composants du système à diagnostiquer. Dans le deuxième type, les cas sont plus génériques et les descripteurs peuvent représenter les familles fonctionnelles des composants du système à

diagnostiquer. De plus, les cas prennent appui sur les modèles de connaissance du domaine à étudier.

Nous abordons, aux sections 3.1 et 3.2, la mise en place des deux types de systèmes sur SISTRE en détaillant les caractéristiques de chaque type et les contributions :

- pour le premier type orienté mining, nous détaillons la représentation du cas, la phase de recherche ainsi que la phase de maintenance de la base de cas ;
- pour le deuxième type orienté connaissance, nous détaillons la représentation du cas, les modèles de connaissance de l'équipement SISTRE ainsi que les deux phases de remémoration et d'adaptation.

3.1. Système orienté mining

Nous développons pour ce type de système la partie représentation de cas, la phase de remémoration ainsi que la phase de maintenance de la base de cas. En effet, la représentation du cas du transfert SISTRE est simple et se base sur les descripteurs représentant l'ensemble des composants.

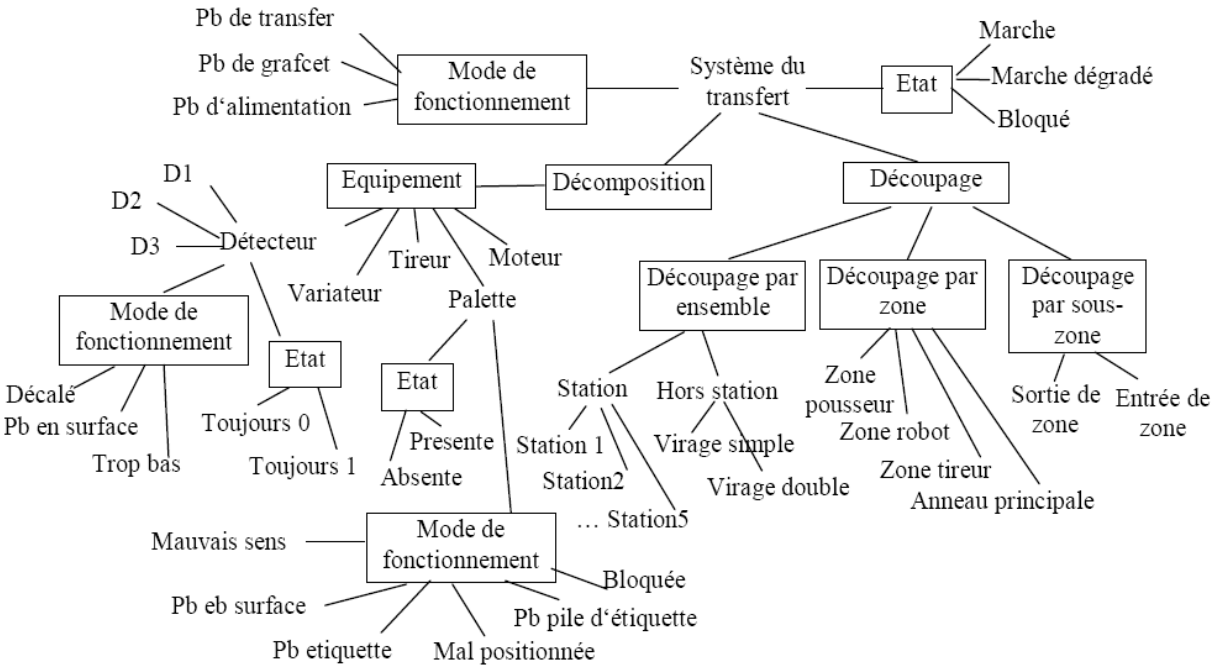


Figure 5.5. Ontologie des descripteurs

3.1.1. Représentation du cas

La représentation des cas du transfert SISTRE prend appui sur une ontologie des descripteurs établie à travers les travaux de notre équipe dans [Rasovska, 2006]. Cette ontologie, qui est illustrée sur la Figure 5.5, permet d'identifier les descripteurs nécessaires et incontournables dans la représentation d'un cas correspondant aux composants du transfert SISTRE.

La décomposition du transfert SISTRE est faite d'une part, suivant l'analyse fonctionnelle des composants et d'autre part, selon la décomposition géographique en ensembles, zones, sous-zones, etc.

Cette étude nous permet de construire une base de cas contenant *750 cas*, *12 attributs* (station, zone, sous-zone, composant-équipement, présence de la palette, type du détecteur principal, son état, stoppeur, son état, type du détecteur de contexte, son état) et *9 classes* (stoppeur, détecteur, balogh, pousseur, tireur, indexeur, tapis intérieur, tapis extérieur et robot). La Figure 5.6 montre un aperçu de la base de cas SISTRE orienté mining.

Base de cas												
Cas	Problème											Solution
	ds1	ds2	ds3	ds4	ds5	ds6	ds7	ds8	ds9	ds10	ds11	Ds1
	Station	Zone	Sous-zone	Composant-équipement	Présence palette	Type détecteur principal	Etat détecteur principal	Stoppeur	Etat du stoppeur	Type détecteur de contexte	Etat détecteur de contexte	Classe du composant
8	Station 1	Anneau secondaire	Entrée	Tapis ext	Présente	D2	1	S2	0			Stoppeur
13	Station 1	Zone poste	Indexeur	Indexeur	Présente	Ball	0					Balogh
16	Station 1	Zone poste	Indexeur	Indexeur	Présente			S4	1	I	1	Indexeur
87	Station 2	Zone poste	Entrée	Tapis ext	Présente	D5	0					Détecteur
232	Station 4	Zone pousseur	Pousseur	Pousseur	Présente	D1	1	S1	1	Pousseur	0	Pousseur
310	Station 5	Zone tireur	Tireur	Tireur	Présente	D6	1			Tireur	0	Tireur

Figure 5.6. Aperçu de la base de cas SISTRE orienté mining

La partie problème du cas est composée de deux sous-parties : la première concerne la localisation de l'endroit de la panne qui est composée des deux premiers descripteurs : ds_1 et ds_2, et la deuxième concerne l'état des composants associés à cet endroit (ds_3.. ds_{10}).

La partie problème peut avoir des cas qui ont tous les descripteurs renseignés (exemple : cas 25), et des cas qui disposent de descripteurs qui ne sont pas tous renseignés (exemple : cas 8, 13, 16, 18 et 34). Lorsqu'une valeur d'un descripteur n'est pas donnée, cela signifie que ce descripteur ne rentre pas en vigueur quant à la détermination de la classe du composant défaillant.

La partie solution fournit la classe du composant défaillant à réparer (Ds_1).

3.1.2. Elaboration du cas cible

Prenons un exemple d'une défaillance qui apparaît dans la station 1 au niveau de l'entrée de la zone « anneau secondaire ». Cette défaillance est représentée par le cas cible suivant :

Cas cible																										
Formulaire Web					Etat des composants (SCADA)																					
dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16	dc17	dc18	dc19	dc20	dc21	dc22	dc23	dc24	dc25	dc26
Station	Zone	Sous-zone	Composant - équipement	Présente palette	Détecteur									Balogh		Stoppeur						Indexeur	Pousseur	Tireur	Robot
					D1	D2	D3	D4	D5	D6	D7	D8	D9	Bal0	Bal1	S1	S2	S3	S4	S5	S6	I	P	T	R
Station 1	Anneau secondaire	Entrée	Pousseur	Présente	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	1	0	0

Figure 5.7. Exemple d'un cas cible dans SISTRE

L'utilisateur détermine les cinq premiers descripteurs du cas cible (dc_1.. dc_5) grâce à un formulaire à remplir, composé de questions fermées, fourni via un portail web. Ensuite, les descripteurs restants (dc_6.. dc_{26}) sont remplis en allant chercher les valeurs des composants à partir d'un système d'acquisition de données SCADA (Supervisory Control And Data Acquisition).

Ainsi, le cas cible est élaboré. Nous passons à la deuxième étape qui consiste à la recherche des cas similaires, dans la base de cas, à ce cas cible.

3.1.3. Phase de remémoration

Cette phase de remémoration prend appui sur une mesure de similarité, qui dépend de la représentation du cas, et de l'algorithme des K plus proches voisins. Nous rappelons que la mesure de similarité, que nous avons décrite au chapitre 2, tient compte de la présence des valeurs des attributs, de leur nature et de leur appartenance à une classe fonctionnelle. Cette mesure est donnée ci-dessous :

$$Sim(Source, Cible) = \frac{\sum_{i=1}^{m} \varphi_i^{\text{Présence}} \times \varphi(s_l, c)}{\sum_{i=1}^{m} \varphi_i^{\text{Présence}}} \qquad (14)$$

Où :

m : est le nombre d'attributs

l : l$^{\text{ième}}$ cas (l = 1..n) où : *n* est le nombre de cas source dans la base de cas

Quant à l'algorithme des Kppv, nous choisissons la valeur « *K = 1* ».

Pour chaque cas source, nous comparons les valeurs des attributs adéquats par rapport aux valeurs des attributs du cas cible. Par exemple, pour la valeur de l'attribut ds_7 du cas 8 (cf. Figure 5.6) : « état du détecteur principal = 1 », qui donne l'état du détecteur « D2 », nous la comparons à la valeur de D2 du cas cible, qui est égale à « 0 » (cf. Figure 5.7), ainsi de suite.

Nous procédons maintenance au calcul de la similarité du cas cible par rapport aux 750 cas sources de la base de cas grâce à la mesure (14). Les cas source les plus similaires au cas cible sont :

$$Sim(Source_7, Cible) = \frac{\overbrace{(1\times1)}^{\varphi^{Présence}} + \overbrace{(1\times1)}^{\varphi(s_7,c)} + (1\times1) + (1\times0) + (1\times1) + (1\times1) + (1\times1)}{7}$$

$Sim(Source_7, Cible) = 85{,}71\%$

$Sim(Source_8, Cible) = 77{,}78\%$

$Sim(Source_9, Cible) = 71{,}43\%$

Les trois cas source les plus similaires sont représentés sur la Figure 5.8.

Cas	Station	Zone	Sous-zone	Composant-équipement	Présence palette	Type détecteur principal	Etat détecteur principal	Stoppeur	Etat du stoppeur	Type détecteur de	Etat détecteur de contexte	Sim (%)	Classe du composant
7	Station 1	Anneau secondaire	Entrée	Tapis ext	Présente	D2	0					85,7	Détecteur
8	Station 1	Anneau secondaire	Entrée	Tapis ext	Présente	D2	1	S2	0			77,8	Stoppeur
9	Station 1	Anneau secondaire	Conv ext	Tapis ext	Présente	D4	0					71,4	Détecteur
…	…	…	…	…	…	…	…	…	…	…	…		…

Figure 5.8. Résultat de la phase de remémoration des cas similaires au cas cible

Le cas le plus proche du cas cible est le cas « $source_7$ ». La classe de ce dernier est « détecteur ». De ce fait, nous nous attribuons cette classe au cas cible.

Nous ne traitons pas la phase d'adaptation dans ce type de système.

3.1.4. Phase de maintenance de la base de cas de la base de cas SISTRE

L'étape de la maintenance de la base de cas SISTRE passe par une première étape de structuration et ensuite par l'auto-incrémentation des cas dans la base de cas.

Structuration de la base de cas

L'échantillon de cas cible que nous prenons représente l'ensemble de la base de cas SISTRE. Nous rappelons que l'étape de structuration de la base de cas s'appuie sur deux critères à savoir la compétence et la performance.

Dans un premier temps, suite à l'application de l'algorithme de structuration de la base de cas, nous obtenons les différentes catégories des cas étudiées au chapitre 3, montrées sur le tableau 5.1.

Catégorie de cas	Cas pivot	Cas auxiliaires	Cas de couverture inter-classe	Cas de couverture intra-classe	Cas de support	Groupe de support
Nombre de cas	80	120	55	150	345	86

Tableau 5.1. Les catégories des cas de la base de cas SISTRE

La méthode de maintenance procède à la suppression des *120 cas auxiliaires* et des *150 cas de couverture intra-classe*. Concernant les cas de support, notre méthode garde les *86 cas de support* qui sont les représentants de chaque groupe de support (chaque représentant possède le plus grand espace de recouvrement). Ensuite, elle supprime les *46 cas de couverture inter-classe* et elle en garde uniquement *9* (correspondant au nombre de classes). Enfin elle garde les *80 cas pivots*. Le total des cas gardé est ainsi égal à ***175 cas***.

De ce fait, nous obtenons les résultats suivants :

Taille initiale de la base de cas		750
Taille de la base de cas obtenue		175
Performance de la base de cas	***Taux réduction***	76,67%
	Taux de précision	100%
Compétence de la base de cas		100%

Tableau 5.2. Statistiques concernant la performance et la compétence de la base de cas SISTRE issues de la méthode de maintenance

La performance qui est traduite par le taux de réduction de la base de cas en fonction de la précision a donné un bon résultat, car la réduction est réalisée plus que ¾ da la base de cas avec une reconnaissance de 100% des classes. Quant à la compétence, elle est de 100%, ce qui montre que le pouvoir de résolution de la base de cas réduite obtenu est le même que la base de cas originale.

La première étape de structuration de la base de cas a fourni de bons résultats en fonction de la compétence et de la performance.

Nous passons maintenant à la deuxième étape concernant l'auto-incrémentation.

Auto-incrémentation de la base de cas SISTRE

Concernant cette étape, nous appliquons un protocole spécifique pour vérifier la faisabilité de la méthode d'auto-incrémentation. Le protocole est le suivant (cf. Figure 5.9) :

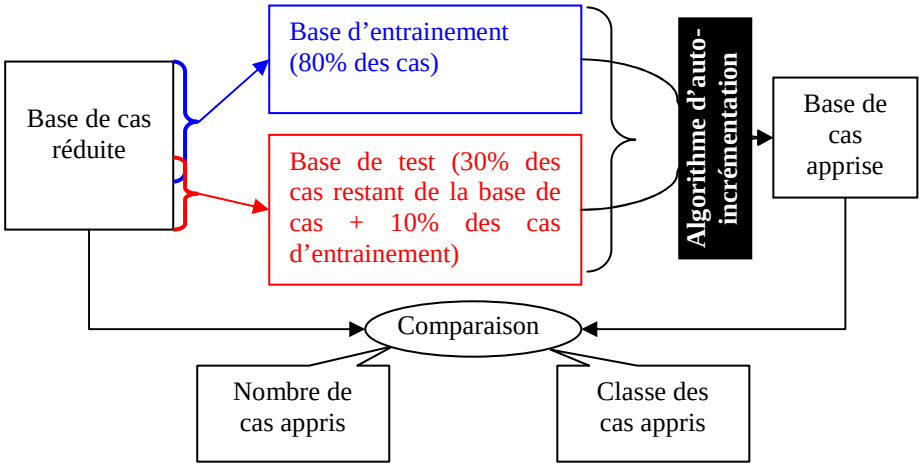


Figure 5.9. Protocole de validation de la méthode d'auto-incrémentation de la base de cas

En appliquant ce protocole sur la base de cas structurée, nous aurons une base d'entrainement qui va contenir 140 cas et une base test de 52 cas (35(20%) + 17(10%)). En appliquant l'algorithme d'auto-incrémentation sur la base d'entrainement, nous obtenons la base de cas apprise contenant tous les cas de la base d'entrainement ainsi que les cas de la base test qui ont rempli les conditions nécessaires. Le tableau 5.3 montre les résultats obtenus.

Taille de la base de cas réduite	175
Taille de la base d'entraînement	140
Taille de la base de test	52
Taille de la base de cas apprise	175
Taux de précision (taux de classification)	100%
Taux d'apprentissage	100%

Tableau 5.3. Statistiques concernant l'apprentissage et le taux de classification de la base de cas SISTRE issus de la méthode de structuration

En comparant les cas de la base d'entrainement apprise résultante avec la base de cas réduite, nous constatons qu'il y a exactement les mêmes cas ainsi que le même nombre de cas. Ce qui nous donne un résultat d'apprentissage à 100%. De plus, les étiquettes de tous les cas de la base de cas apprise sont exactement similaires à celles de tous les cas de la base de cas réduite.

Ce résultat montre que l'algorithme d'auto-incrémentation a bien fonctionné en apprenant que les cas utiles dans la base de cas d'entrainement. Cet algorithme a permis de retrouver la base de cas réduite initiale grâce à un processus d'apprentissage automatique.

Par conséquent, nous pouvons conclure que la méthode de maintenance de la base de cas proposée a donné de bons résultats sur l'équipement SISTRE concernant la structuration de la base de cas (compétence + performance) et son auto-incrémentation.

Nous avons ainsi étudié la mise en place d'un système orienté mining pour lequel nous avons développé la présentation du cas, la phase de remémoration et la phase de maintenance de la base de cas.

Bien que maintenant la structuration de la base de cas dans la phase d'auto-incrémentation, nous sommes amenés à terme à travailler avec un nombre conséquent de cas, ce qui devrait occasionner un ralentissement du système et pourrait altérer, malgré tout, le temps de réponse, ce qui nécessiterait de revoir l'algorithme de recherche. Une des voies d'amélioration de ce système est de concevoir un système orienté connaissance.

3.2. Système orienté connaissance

Pour décrire ce système, nous abordons en premier lieu la formalisation d'un cas ainsi que la base de cas et les modèles de connaissance associés. Ceci concerne le processus off-line. Concernant le processus on-line, nous développons les différentes phases du cycle de RàPC en commençant par l'élaboration du cas cible, la phase de remémoration guidée par l'adaptation et l'algorithme d'adaptation. Nous exploiterons trois exemples types de diagnostic.

3.2.1. Représentation d'un cas

Une première étude concernant la représentation du cas a été réalisée au sein de notre équipe dans [Rasovska, 2006]. A partir de l'historique des pannes et d'une AMDEC[9], les

[9] AMDEC : Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité : méthode d'analyse préventive de la fiabilité

défaillances de l'équipement ont été identifiées. Ensuite, nous avons proposé une modélisation du cas tenant compte de la définition du diagnostic (méthode utilisée dans le chapitre 2). Nous obtenons ainsi une structure du cas comportant les parties localisation, fonctionnelle, classe de défaillance et composant défaillant.

La partie problème est divisée en deux : la partie localisation et la partie fonctionnelle. La partie localisation reflète l'endroit géographique de la défaillance (nous assurons le suivi d'une palette et son déplacement). En effet, sa position géographique prend appui sur un modèle géographique. Cette partie localisation comporte deux descripteurs : « ds_1 = zone » et « ds_2 = emplacement palette ». Cette partie implique un certain nombre de composants potentiellement défaillants. Ces composants sont représentés par la partie fonctionnelle qui comporte cinq descripteurs : « ds_3 = détecteur principal », « ds_4 = actionneur pneumatique », « ds_5 = actionneur électrique », « ds_6 = détecteur de présence » et « ds_7 = détecteur magnétique ». Les descripteurs de la partie fonctionnelle sont caractérisés par trois attributs relatifs à la valeur du composant, son état et son mode de fonctionnement (MF) : ds_i = (ds_i^{Valeur} , ds_i^{Etat} , $ds_i^{M.F}$) comme on a pu voir au chapitre 2. Enfin, la partie solution comporte quatre descripteurs : « Ds_1 = classe de défaillance », « Ds_2 = identification du composant défaillant », « Ds_3 = action de réparation associée » et « Ds_4 = zone de défaillance ». Ce dernier descripteur est en relation directe avec les deux premiers descripteurs de problème.

Les différentes valeurs possibles des descripteurs de problème sont [Haouchine, 2005] :

ds_1 : Zone ∈ {anneau principal, anneau secondaire, zone poste, zone pousseur, zone tireur, zone robot}.

ds_2 : Emplacement palette ∈ {entrée, convoyeur intérieur, convoyeur extérieur, robot, sortie}.

ds_3 : Détecteur principal ∈ {Balogh0 (Bal0), Balogh1 (Bal1), détecteur Di, i = 1…9}.

ds_4 : Actionneur pneumatique ∈ {entrée, convoyeur intérieur, convoyeur extérieur, robot, sortie}.

ds_5 : Actionneur électrique ∈ {Si, i = 1…6}.

ds_6 : Détecteur de présence ∈ {détecteur Di, i = 1…9}.

ds_7 : Détecteur magnétique ∈ {Balogh0 (Bal0), Balogh1 (Bal1)}.

Les valeurs des descripteurs de solution sont :

Ds_1 : classe de défaillance ∈ {détecteur de présence, actionneur électrique, actionneur pneumatique, détecteur magnétique, tapis, robot}.

Ds_2 : identification du composant défaillant. Cela concerne tous les composants de l'équipement SISTRE.

Ds_3 : action de réparation associée. Il y a plusieurs actions de réparation qui sont associées à la classe de défaillance et adaptées au composant défaillant.

Ds_4 : zone de défaillance. Cela concerne tous les endroits de la station. Ce descripteur représente la fusion des deux descripteurs de problème ds_1 et ds_2.

3.2.2. Base de cas

Nous avons créé une base de cas contenant *69 cas* avec *11 attributs* pour chaque cas [Haouchine et al., 2008b]. La Figure 5.10 donne un exemple de 10 cas de la base de cas.

	Problème																	Solution			
	Localisation		Fonctionnelle																		
	ds1	ds2	ds3			ds4			ds5			ds6			ds7			Ds1	Ds2	Ds3	Ds4
N° Cas	Zone	Emp palette	Dét prin	Etat	MF	Act Pneu	Etat	MF	Act Elec	Etat	MF	Dét de pré	Etat	MF	Dét mag	Etat	MF	Classe de défaillance	Identification composant défaillant	Action de réparation associée	Zone de défaillance
1	Anneau principal	entrée	D1	0	an				S1	haut	nor				Bal0	1	nor	Détecteur de présence	D1 décalé	Replacer	Entrée anneau principal
2	Anneau principal	entrée	D1	1	nor				S1	bas	an				Bal0	1	nor	Actionneur électrique	S1 bloqué	Débloquer	Entrée anneau principal
3	Zone poste		Bal1	1	nor	Ind	1	an	S4	haut	nor							Actionneur pneumatique	Obstacle sous Indexeur	Enlever obstacle	Zone poste
4	Anneau principal	sortie	D8	1	an	Tireur	0	nor	S6	haut	nor				Bal1	0	nor	Détecteur de présence	D8 décalé	Changer	Sortie anneau principal
5	Anneau secondaire	Cour ext	D6	1	nor				S5	haut	nor	D5	0	nor	Bal1	1	nor	Actionneur pneumatique	Tireur bloqué	Débloquer	Cour. ext anneau secondaire
6	Zone poste	Ind							S4	bas	nor	D6	0	an				Détecteur magnétique	Champs magnétique fort à proximité Bal1	Nettoyer Balogh	Indexeur Zone Poste
7	Anneau secondaire	Cour ext	D6	0	nor	Pous	0	nor	S5	haut	an	D7	0	nor	Bal1	1	nor	Actionneur électrique	S5 bloqué	Débloquer	Sortie anneau secondaire
8	Zone robot	Ind	Bal0	1	nor	Ind	1	an	S4	haut	nor	D6	0	nor				Robot	Robot bloqué	Débloquer Robot	Zone robot
9	Zone tireur	sortie	D6	1	nor				S6	haut	nor	D8	0	nor	Bal1	1	nor	Tapis	Tapis int. bloqué	Débloquer courroie	Anneau Principal
10	Anneau secondaire	Cour ext	D4	1	an				S3	haut	an	D5	1	nor				Tapis	Tapis ext. bloqué	Débloquer courroie	Cour. ext anneau secondaire

Figure 5.10. Aperçu de la base de cas SISTRE orienté connaissance

Prenons l'exemple du cas 1. C'est un cas qui représente un problème au niveau du « détecteur D1 ». La partie localisation détermine le lieu de la panne qui dépend de la zone dans laquelle la palette est bloquée. En l'occurrence, la palette se situe au niveau de l'entrée de la zone anneau principal. Ensuite, la partie fonctionnelle fournit l'état des composants impliqués dans cette zone. Le descripteur « ds_5 » reflète le composant « stoppeur S1 » qui est en position « haut » et qui a un mode de fonctionnement « normal », le descripteur « ds_7 » concerne la « Balogh 0 » qui a la valeur « 1 », ce qui signifie qu'elle doit entrer dans la zone

de travail pour qu'elle puisse être traitée par un robot. Enfin, le descripteur « ds_3 » reflète le détecteur D1 qui ne détecte pas la présence de la palette et qui est en mode « anormal ».

Quant à la partie solution, le descripteur « Ds_1 » précise la classe de défaillance du composant « détecteur D1 », qui est décalé et indiqué par le descripteur « Ds_2 ». Le détecteur D1, doit être replacé comme le mentionne le descripteur « Ds_3 », qui se trouve au niveau de l'entrée de l'anneau principal indiqué par le descripteur « Ds_4 ».

3.2.3. Modèle hiérarchique des composants de l'équipement

Nous avons procéder à une analyse fonctionnelle de l'équipement qui nous a permis de classer les composants suivant leur type de fonctionnement (cf. Figure 5.11).

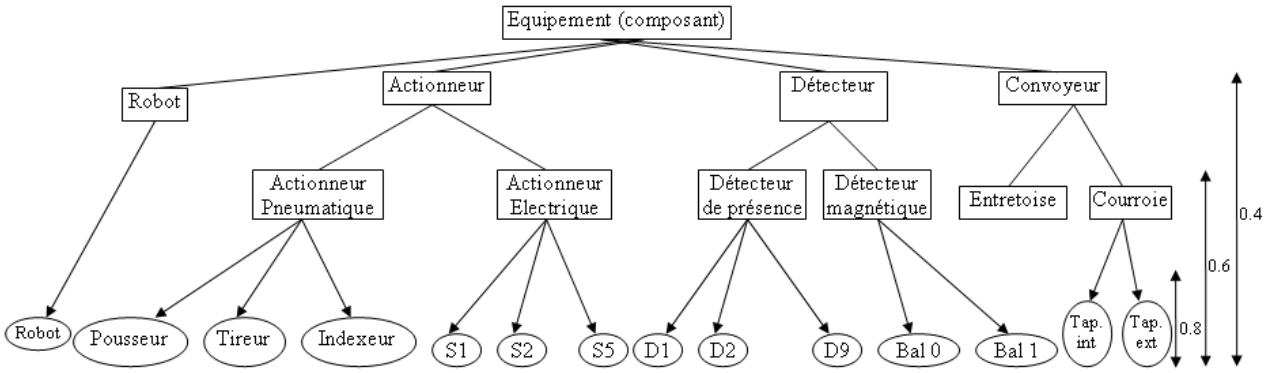


Figure 5.11. Modèle hiérarchique des composants de l'équipement SISTRE

Nous disposons ainsi de plusieurs classes de composants. Par exemple, la classe « détecteur » se décline en deux sous-classes à savoir « détecteur de présence » et « détecteur magnétique ». Ces deux dernières assurent la même fonction de détection de la présence de la palette. De plus, la Balogh assure une fonction supplémentaire de lecture et écriture sur l'étiquette de la palette. Pour chacune des classes un ensemble de composants est défini.

Le modèle hiérarchique est également exploité pour déterminer les valeurs de la mesure de similarité locale « φ^{Valeur} », qui est utilisée dans la meure de remémoration (14), comme cela est montré sur la Figure 5.11 avec les valeurs « 0.8, 0.6, 0.4… ».

3.2.4. Modèle de contexte de l'équipement

Le modèle de contexte reflète les relations causales qui existent entre les composants de l'équipement SISTRE. Ces relations sont déterminées grâce au flux de la palette. En effet, ce flux nous a permis de découper la station en plusieurs espaces géographiques. Les ensembles des composants associés à chaque zone géographique constituent un contexte dans lequel le

composant défaillant est identifié. Six principales zones sont déterminées : zone anneau principal, zone anneau secondaire, zone pousseur, zone tireur, zone poste et zone robot. Toutes les zones sont divisées en trois sous-zones selon l'emplacement de la palette. Les deux premières sous-zones sont des entrées et sorties. Quant à la troisième, il s'agit de la courroie intérieure pour l'anneau principal, courroie extérieure pour l'anneau secondaire, sous-zone pousseur pour la zone pousseur, sous-zone tireur pour la zone tireur et sous-zone indexeur pour la zone indexeur.

Un aperçu du modèle de contexte général construit par les graphes contextuel est donné sur la Figure 5.12.

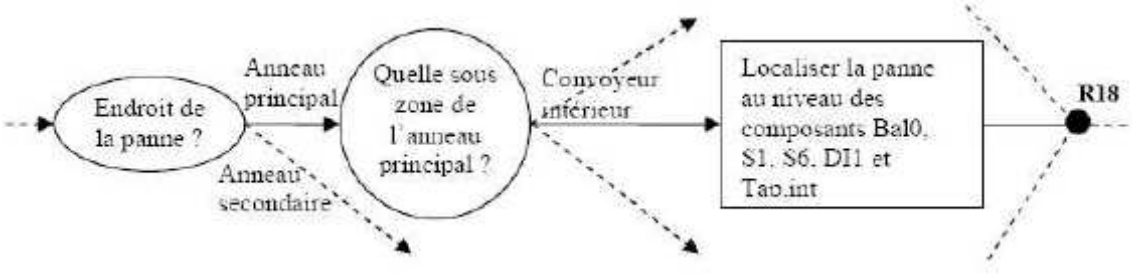


Figure 5.12. Aperçu d'un graphe contextuel du modèle de contexte

La Figure 5.13 montre un exemple de contexte du descripteur « Ds_2 ».

Contexte
« entrée anneau principal »
Ds_2 : Stoppeur S1 bloqué = f(D1, Bal0, Tapis int)
Ds_2 : Détecteur D1 décalé = f(S1, Bal0, Tapis int)
Ds_2 : Poussière dans Bal0 = f(S1, D1, Tapis int)
Ds_2 : Tapis intérieur défectueux = f(S1, D1, Bal0)

Figure 5.13. Exemple du modèle de contexte de l'équipement SISTRE

En effet, le contexte du descripteur « Ds_2 » met en relation des composants en fonction de leur valeur. De ce fait, la valeur de ce descripteur, reflétant un composant, détermine le contexte dans lequel se trouve ce composant. A titre d'exemple, le contexte « entrée anneau principal » a engendré la détermination de la valeur de « Ds_2 = détecteur D1 décalé » selon l'état des autres composants. C'est-à-dire que le contexte qui a identifié les composants S1, Bal0 et Tapis intérieur a impliqué la détermination du composant D1 selon des conditions bien précises.

3.2.5. Mode de fonctionnement des composants de l'équipement

Nous avons mis en place des règles de décision qui déterminent le mode de fonctionnement des composants sélectionnés dans la zone de défaillance. Il y a deux modes de fonctionnement : *normal* et *anormal*. Nous disposons de neuf ensembles de règles de décision relatifs aux composants se trouvant dans les différentes zones de défaillance [Haouchine et al., 2007b].

3.2.6. Phase d'élaboration d'un cas

L'élaboration du cas cible se compose de trois étapes : une première étape de localisation de la panne qui se fait grâce au modèle de contexte (A). Ce modèle nous fournit un certain ensemble de composants pouvant être potentiellement défaillants par leur proximité géographique dans la zone concernée. La phase d'élaboration s'initialisera donc par le remplissage des descripteurs associés à la « localisation de la panne ». Ensuite, la seconde étape consiste en la consultation du système d'acquisition SCADA qui nous fournit l'état actuel des composants concernés par la localisation d'une façon automatique (B). Enfin, une troisième étape de décision qui, à partir de l'état actuel de ces composants, met en œuvre des règles de décisions en donnant le mode de fonctionnement des composants impliqués (C).

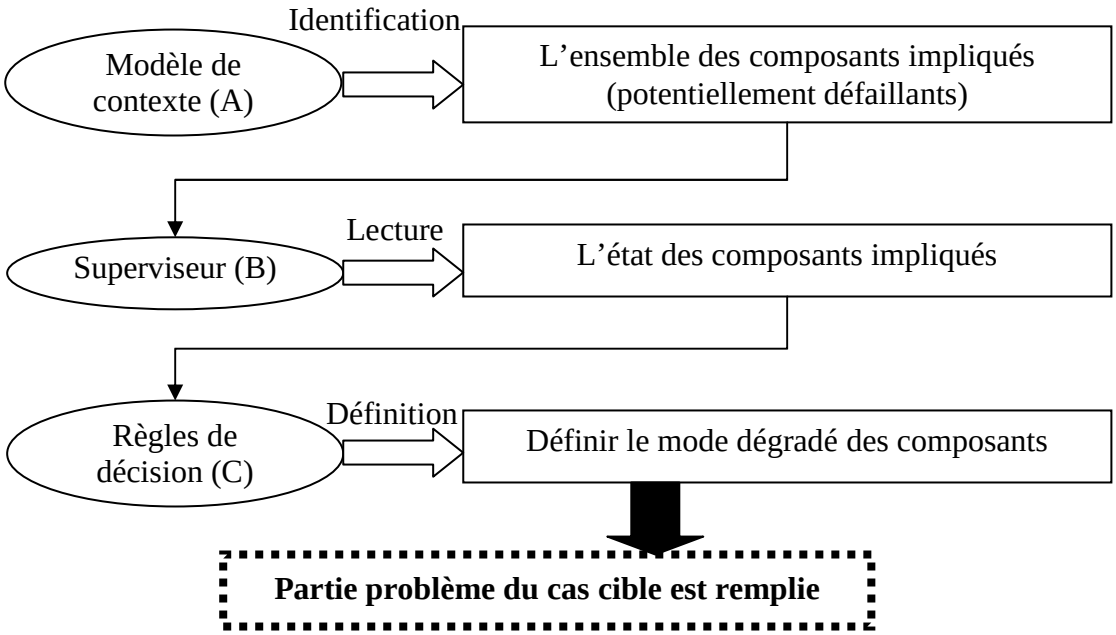


Figure 5.14. Les différentes étapes d'élaboration du cas cible

Ceci permet aux descripteurs de la partie problème du cas cible, c'est-à-dire ses parties localisation et fonctionnelle d'être renseignées (cf. Figure 5.14). Il est à signaler que la

relation de cause à effet est prise en compte lors de la création de la base de cas. Les différents composants sont liés par leur état, ce qui se traduit par les valeurs des descripteurs de la base de cas et donc la définition de leurs modes de fonctionnement. En effet, un composant peut être dans un état de mode de fonctionnement anormal mais il n'est pas défaillant. Ceci peut arriver à cause de l'influence d'un autre composant défaillant.

3.2.7. Phases de remémoration et d'adaptation

Nous avons développé une méthode de remémoration guidée par l'adaptation ainsi qu'un algorithme d'adaptation au chapitre 4. Cette méthode est dédiée spécifiquement au diagnostic de pannes des équipements industriels. Nous avons illustré, au chapitre 4, les deux propositions sur un moteur à explosion. Maintenant, nous allons les appliquer sur le transfert SISTRE et nous montrerons, par la même occasion, la faisabilité de notre contribution sur deux équipements complètement différents.

Les relations de dépendance (RD) sont mises en place selon le même principe de celui abordé au chapitre 4. Nous rappelons que la détermination de la nature des valeurs de « RD » est exploitée uniquement par la partie fonctionnelle du cas. Les différentes valeurs de RD sont : RD forte avec des classes similaires, RD forte avec des classes différentes et RD faible.

Nous allons étudier les deux phases de remémoration et d'adaptation à travers des exemples représentant trois cas types d'adaptation. En effet, le premier exemple concerne le cas où il y a une relation forte entre un (des) descripteur(s) de solution source « Ds » et un (des) descripteur(s) de problème source « ds » et qu'ils ont la même classe fonctionnelle. Le deuxième exemple concerne une relation forte entre les descripteurs mais n'ayant pas la même classe fonctionnelle. Enfin, le troisième exemple concerne le cas où les descripteurs de problème et de solution disposent de relations de dépendance faible.

Premier cas type d'adaptation : « RD = Forte & même classe de fonctionnement »

Soit une défaillance qui est apparue au niveau du détecteur D8 et représentée par le cas *cible 1* (cf. Figure 5.15). Nous rappelons que la mesure de remémoration (M_R) est la suivante :

$$M_R = \frac{\sum_{i=1}^{m} \varphi_i^{Valeur} \times \varphi_i^{Etat} \times \varphi_i^{Présence} \times \varphi_i^{M.F}}{\sum_{i=1}^{m} \varphi_i^{Présence}} \qquad (15)$$

Le seuil de similarité que nous avons fixé est de 60%.

Les résultats issus suite à la phase de remémoration sont les suivants :

$$M_R(srce_4, cible_1) = \frac{(0\times1) + (1\times1) + (1\times1\times0.8\times1) + (1\times1\times1\times1) + (1\times1\times1\times1)}{5}$$

($\varphi_i^{Présence}$, φ_i^{Etat}, φ_i^{Valeur}, $\varphi_i^{M.F}$; ds_1, ds_2, ds_3, ds_5, ds_7)

$M_R(srce_4, cible_1) = 0.76$

$M_R(srce_9, cible_1) =$ **0.8**

Les parties problèmes des deux cas sources 4 et 9 sont représentées sur la Figure 5.15.

	d1	d2	d3			d4			d5			d6			d7		
Cib1	Zone tireur	sortie	D8	1	an				S6	haut	nor	D9	0	nor	Bal1	1	nor
4	Anneau principal	sortie	D8	1	an	Tireur	0	nor	S6	haut	nor				Bal1	1	nor
9	Zone tireur	sortie	D6	1	nor				S6	haut	nor	D8	0	nor	Bal1	1	nor

Figure 5.15. Les cas source les plus similaires au cas cible 1

Nous allons à présent calculer la mesure d'adaptation pour les deux cas sources 4 et 9. Nous rappelons que la mesure d'adaptation (M_A) est comme suit :

$$M_A = \frac{\sum_{i=1}^{m} \lambda_i . \times \varphi_i^{Présence} \times \varphi_i^{Valeur}}{\sum_{i=1}^{m} \varphi_i^{Présence}} \qquad (16)$$

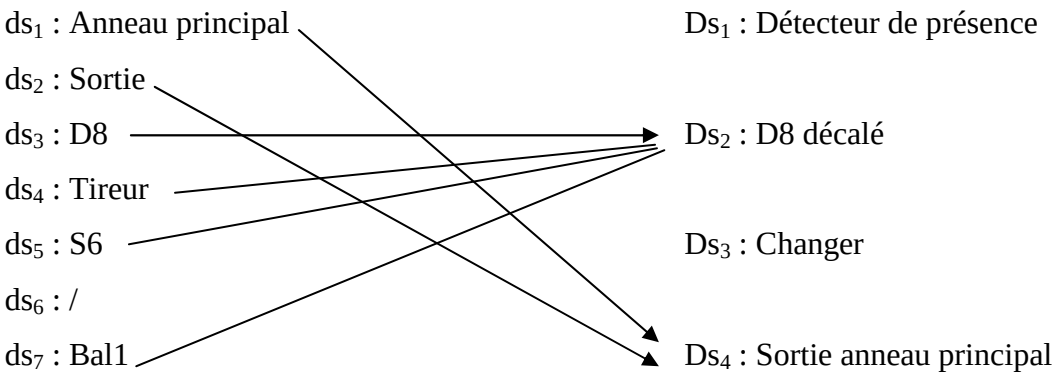
En l'appliquant, nous trouvons les résultats suivants :

$$M_A(srce_4, cible_1) = \frac{(2\times1\times0.8) + (2\times1\times1) + (2\times1\times1)}{3} = \mathbf{1.86}$$

$M_A(srce_9, cible_1) = 1.80$

Nous constatons que $M_A(srce_4, cible_1) > M_A(srce_9, cible_1)$. De ce fait, le cas source 4 est le cas qui va être sélectionné pour la phase d'adaptation. Nous remarquons par la même occasion que le cas source le plus similaire (source 9) n'est pas celui qui a la plus grande valeur de M_A.

La représentation des relations de dépendance entre les descripteurs de problème et les descripteurs de solution du ***cas source 4*** est la suivante :



Afin de déterminer quel type de RD nous allons traiter, nous tenons compte que des relations de dépendance concernant les descripteurs de problème de ds_3 à ds_7. Les deux premiers servent à déterminer la zone de défaillance, en l'occurrence : « sortie anneau principal ».

Nous rappelons également que les flèches représentent les relations « RD = forte », les segments représentent les relations « RD = faible » et s'il n'y a ni flèches ni segments alors « RD = pas de relation ».

Nous constatons d'une part, que la valeur de RD du couple (Ds_2, ds_3) est « RD = forte » et que les autres valeurs sont « RD = Faible » et d'autre part, les deux descripteurs Ds_2, ds_3 représentent le même composant « *D5* » et donc ils ont la même appartenance de classe. De ce fait, nous sommes dans le cas de figure de « *RD = Forte et même classe de fonctionnement* ».

Désormais, nous pouvons appliquer l'algorithme d'adaptation :

- Les deux valeurs ds_3^{rem} et dc_3 sont égales alors nous affectons directement la solution de Ds_2 à la solution cible Dc_2 : Dc_2 = *D8 décalé*.

La solution est donc la suivante : *changer le détecteur D8 qui est décalé appartenant à la classe « détecteur de présence » et se trouvant au niveau de la sortie de l'anneau principal*.

Deuxième cas type d'adaptation : « RD = Forte & classes de fonctionnement différentes »

Soit une panne qui s'est produite dans le moteur au niveau du xxx et représentée par le cas *cible 2* (cf. Figure 5.16). Les résultats du calcul des mesures de remémoration et d'adaptation sont les suivants :

- ***Calcul de la mesure de remémoration***

$M_R(srce_3, cible_2)$ = **0.90**

$M_R(srce_6, cible_2)$ = 0.70

$M_R(srce_8, cible_2)$ = 0.60

Les cas les plus similaires au cas *cible 2* sont les cas sources 3, 6 et 8 (cf. Figure 5.16).

	d1	d2	d3			d4			d5			d6			d7		
Cib2	Zone poste	Ind	D1	1	nor	Pous	1	an	S4	haut	nor	D5	0	an	Bal0	1	nor
3	Zone poste		Bal1	1	nor	Ind	1	an	S4	haut	nor						
6	Zone poste	Ind							S4	bas	nor	D6	0	an			
8	Zone robot	Ind	Bal0	1	nor	Ind	1	an	S4	haut	nor	D6	0	nor			

Figure 5.16. Les cas source les plus similaires au cas cible 2

- ***Calcul de la mesure d'adaptation***

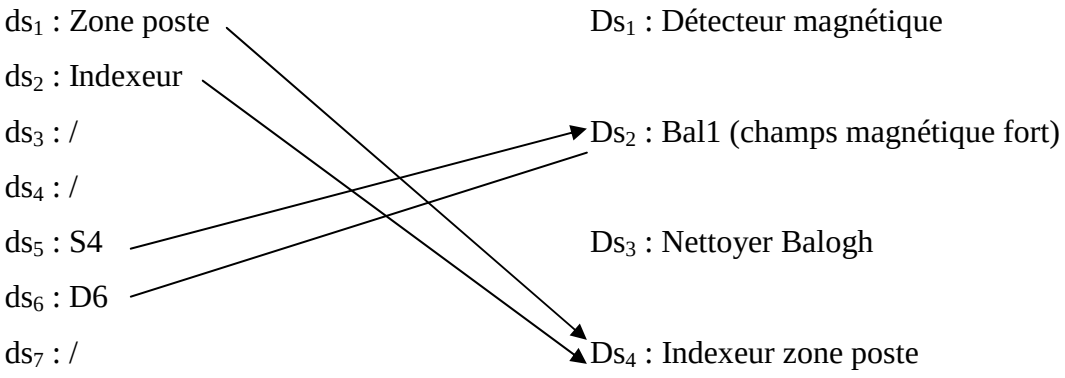
$M_A(srce_3, cible_2)$ = 1.73

$M_A(srce_6, cible_2)$ = **1.80**

$M_A(srce_8, cible_2)$ = 1.70

D'après les résultats de la mesure d'adaptation, le cas source sélectionné pour la phase d'adaptation est le cas 6.

La représentation des relations de dépendance entre les descripteurs de problème et les descripteurs de solution du ***cas source 6*** est la suivante :



Nous pouvons constater que la valeur de RD du couple (Ds_2, ds_5) est « RD = forte » et que les composants « S4 » et « Bal » appartiennent à deux classes différentes. De ce fait, nous sommes dans le cas de figure suivant : « *RD = Forte et différentes classes de fonctionnement* ».

En appliquant l'algorithme d'adaptation, nous obtenons :

- La classe du descripteur solution source Ds_2^{rem} est « Détecteur magnétique » ;
- Le composant « S4 » du descripteur cible « dc_5 » (qui correspond au descripteur « ds_5 » qui est en mode anormal) se trouve dans le contexte de la zone « Indexeur de la zone poste ». Or, dans cette zone se trouvent d'autres composants qui sont : D1, pousseur, D5 et Balogh0 ;
- Le composant « Balogh0 » appartient à la même classe de Ds_2^{rem} qui est « Détecteur magnétique »
- Substituer la valeur « Bal1 » du descripteur « Ds_2^{rem} » par la valeur « Bal0 » de « dc_7 ». De ce fait, Ds_2^{rem} = **Bal0** (champs magnétique fort);
- Affecter cette valeur à « Dc_2 » : Dc_2 = *Bal0 (champs magnétique fort).*

La solution sera donc : *nettoyer la Balogh0, appartenant à la classe « Détecteur magnétique », qui se trouve au niveau de l'indexeur de la zone poste*.

Troisième cas type d'adaptation : « RD = Faible »

Soit une panne qui s'est produite dans le moteur au niveau du xxx et représentée par le cas *cible 3* (cf. Figure 5.17). Les résultats du calcul des mesures de remémoration et d'adaptation sont les suivants :

- ***Calcul de la mesure de remémoration***

$M_R(srce_5, cible_3) = 0.66$

$M_R(srce_7, cible_3) =$ **0.73**

Les cas les plus similaires au cas *cible 3* sont les cas sources 5 et 7 (cf. Figure 5.17).

	d1	d2	d3			d4			d5			d6			d7		
Cib3	Anneau secondaire	Cour ext	D4	0	nor				S3	bas	nor	D5	0	nor	Bal1	1	nor
5	Anneau secondaire	Cour ext	D6	1	nor				S5	haut	nor	D5	0	nor	Bal1	1	nor
7	Anneau secondaire	Cour ext	Bal1	0	nor	Pous	0	nor	S5	haut	an	D7	0	nor	Bal1	1	nor

Figure 5.17. Les cas source les plus similaires au cas cible 3

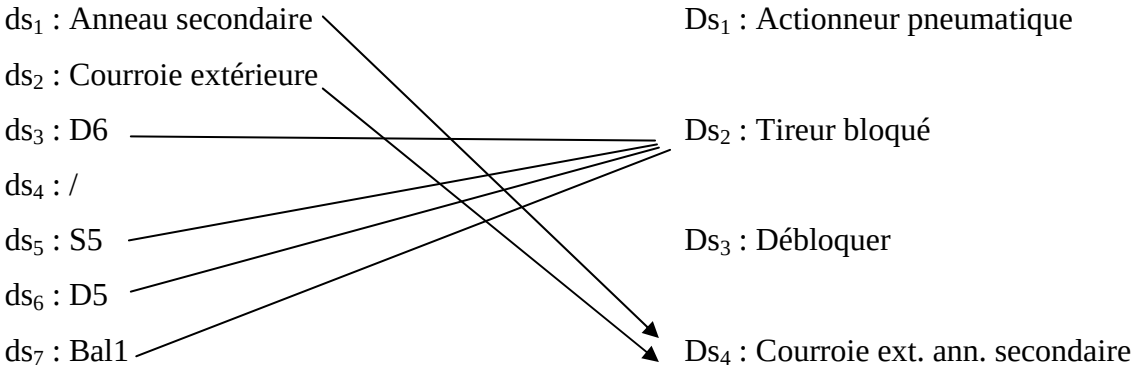
- ***Calcul de la mesure d'adaptation***

$M_A(srce_5, cible_3)$ = **1.80**

$M_A(srce_7, cible_3)$ = 1.60

D'après les résultats de la mesure d'adaptation, le cas source sélectionné pour la phase d'adaptation est le cas 5.

La représentation des relations de dépendance entre les descripteurs de problème et les descripteurs de solution du ***cas source 5*** est la suivante :



Nous constatons que tous les couples (Ds_2, ds_3), (Ds_2, ds_5), (Ds_2, ds_6) et (Ds_2, ds_7) ont des valeurs de « RD = faible ».

En appliquant l'algorithme d'adaptation, nous obtenons :

- La classe du descripteur de solution source « Ds_2^{rem} » est « Actionneur pneumatique » ;
- Le contexte des composants du cas cible « D4, S3, D5 et Bal1 » se trouve au niveau de la courroie extérieure de l'anneau secondaire ;

- Dans ce contexte, le composant appartenant au même contexte et qui appartient à la même classe de Ds_1^{rem} (Actionneur pneumatique) est le composant : *pousseur* ;
- Substitution de la valeur « tireur » par la valeur « pousseur » dans le descripteur Ds_2^{rem}, Ds_2^{rem} = pousseur bloqué ;
- Affecter cette nouvelle valeur au descripteur Dc_2.

De ce fait, la solution est la suivante : *débloquer le pousseur qui est bloqué, appartenant à la classe « Actionneur pneumatique », qui se trouve dans la courroie extérieure de l'anneau secondaire*.

Nous avons constaté à travers ces trois exemples que le cas le plus facilement adaptable n'est pas forcément le cas le plus similaire.

Afin d'approuver les résultats obtenus via notre méthode et afin de la comparer avec une méthode classique qui utilise de simples phases de remémoration et d'adaptation, nous procéderons à un protocole de validation qui va être abordé dans la prochaine section.

3.2.8. Validation des phases de remémoration et d'adaptation

Nous appliquons le même protocole de validation décrit au chapitre 4 pour valider notre proposition. En effet, ce protocole permet de diviser la base de cas en deux sous-ensembles. Le premier sous-ensemble comporte 40 % des cas tirés aléatoirement de la base de cas formant ainsi la base d'entrainement (Be). Le deuxième sous-ensemble comporte les 60 % des cas restants, formant ainsi la base de test (Bt). Les parties solutions de cette base sont enlevées. Ensuite, les cas de la base de test (Bt), comportant 41 cas, vont être soumis à la base d'entrainement (Be), comportant 28 cas, pour essayer de donner une solution adéquate. Enfin, les solutions données vont être comparées aux solutions enlevées initialement de la base de test (Bt) donnant ainsi son taux de précision (accuracy).

Notre méthode, qui utilise des phases de remémoration et d'adaptation élaborées, sera ensuite comparée à une méthode employant une phase de remémoration simple et une adaptation triviale.

Les résultats sont montrés sur la Figure 5.18.

Les résultats de la figure gauche montrent que notre méthode a assuré une parfaite adaptation (100%), sur l'ensemble des cas de la base de test (Bt) présentés à la base d'entrainement (Be), par rapport à la méthode utilisant une adaptation triviale (65.85 %). Quant à la figure droite, nous constatons que notre méthode a une évolution constante du taux de bonne classification. Cependant, l'évolution du taux de précision de la méthode utilisant une adaptation triviale est discontinue. Ce qui prouve que notre méthode s'adapte bien aux différentes natures des cas de la base de cas SISTRE et que l'algorithme d'adaptation est performant grâce notamment à la sélection du cas le plus facilement adaptable suivant les deux mesures utilisées dans la phase de remémoration : M_R et M_A.

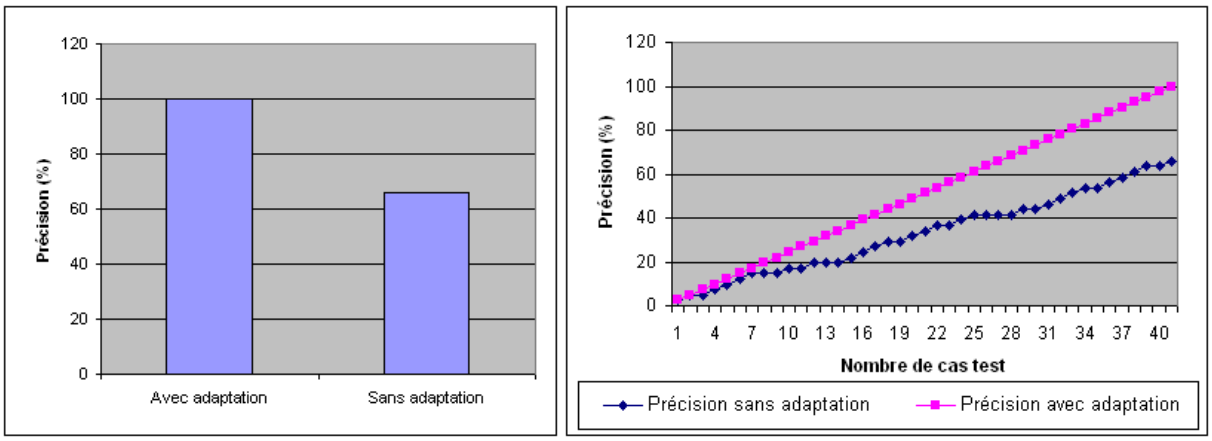


Figure 5.18. Taux de précision avec notre méthode d'adaptation et sans adaptation (figure gauche). Evolution de la précision suivant le nombre de cas dans la base de test (Bt) (figure droite).

4. Conclusion

Nous avons présenté dans ce chapitre le système industriel SISTRE (Supervised Industrial System of pallets TransFer) ainsi que ses différents composants. Ce système est composé de cinq stations identiques, nous avons décrit en détail la composition d'une station.

Tout d'abord, nous avons mis en place deux types de système de RàPC sur ce transfert, un orienté mining et un orienté connaissance. La formalisation d'un cas de diagnostic a pris appui sur des techniques empruntées à la sûreté de fonctionnement et la structure de ces cas a été adaptée au diagnostic dédié à l'équipement SISTRE.

Pour le système orienté mining, nous avons mis en place trois phases : la phase d'élaboration du cas cible, la phase de remémoration et la phase de maintenance de la base de

cas et son auto-incrémentation. La méthode a donné des résultats satisfaisants, d'une part, en catégorisant les cas de la base de cas SISTRE et d'autre part, en apprenant que les cas utiles en les insérant dans cette base de cas.

Pour le système orienté connaissance, nous avons procéder à une analyse fonctionnelle et dysfonctionnelle de cet équipement. Cette analyse nous a permis de créer une base de cas à laquelle nous avons associé deux modèles de connaissance. Le premier modèle reflète les relations de cause-à-effet entre les composants de l'équipement qui sont déterminées par le flux de la palette. Le deuxième modèle est issu d'une analyse fonctionnelle des composants de cet équipement qui les regroupe par familles. Cette mise en place nous a ainsi permis d'étudier notre méthode de remémoration guidée par l'adaptation ainsi que l'algorithme d'adaptation dédié à cet équipement. En effet, nous avons exploité deux mesures de remémoration et d'adaptation dans la phase de remémoration. Cela nous a permis de constater que les cas les plus similaires ne sont pas forcément les cas choisis pour la phase d'adaptation. Une fois que le cas est choisi, trois possibilités d'adaptation s'offrent à lui. Ces possibilités dépendent du type de relation de dépendance entre les descripteurs de solution et les descripteurs de problème. Ainsi, trois exemples types ont été traités. Nous avons mis en place une démarche de validation permettant l'évaluation de nos méthodes de remémoration guidée par l'adaptation et l'algorithme d'adaptation selon le taux de bonne classification des cas.

Nous avons validé nos algorithmes sur un nombre de cas limité afin de prouver la faisabilité de notre méthode. Il reste à déployer l'application en grandeur nature, ce qui est une perspective immédiate de nos travaux.

Conclusion générale

Nous nous sommes intéressés dans cette thèse à la maintenance corrective qui est exécutée après la détection d'une panne et destinée à remettre un bien dans un état dans lequel il peut accomplir une fonction requise [Afnor, 2001]. Les actions de maintenance corrective ne sont exécutées qu'après la réalisation d'un diagnostic. Ce diagnostic identifiera la cause de la défaillance pour donner la nature des actions de réparation à accomplir. Par conséquent, nos travaux ont porté sur l'étude de la mise en place d'un système d'aide au diagnostic industriel basé sur l'approche du raisonnement à partir de cas (RàPC).

Nous avons organisé nos travaux en trois parties.

La première partie introduit les notions de bases du diagnostic ainsi que les principes fondamentaux de l'approche du RàPC. Cette introduction est suivie par un état de l'art sur les systèmes de diagnostic par RàPC. La première partie est organisée au travers des deux premiers chapitres. De plus, cette partie présente nos choix et nos démarches concernant la construction d'un système d'aide au diagnostic par RàPC.

La deuxième partie synthétise l'essentiel de notre contribution scientifique concernant cette construction. En effet, cette partie s'articule autour du troisième et du quatrième chapitre qui présentent nos propositions concernant les systèmes orientés *mining* et les systèmes orientés *connaissances*.

Enfin, la troisième partie qui implique le dernier chapitre, expose la mise en place des deux types de système sur l'équipement industriel SISTRE (Supervised Industrial System of pallets TransFer).

Nous avons commencé par la présentation de notre domaine d'étude qui est le diagnostic. Nous avons adopté l'approche du RàPC, se trouvant à l'intersection des sciences cognitives et de l'intelligence artificielle, qui est largement utilisée dans le domaine du diagnostic. En effet, le RàPC peut s'appliquer dans un domaine à connaissances réduites et qui évolue dans le temps. Par ailleurs, il existe trois modèles de RàPC dépendant en grande partie de la représentation du cas, à savoir : les modèles structurel, conversationnel et textuel.

Nous avons choisi le modèle structurel pour la mise en place de notre système de diagnostic car le domaine d'étude est connu *a priori* et que les cas doivent être structurés pour caractériser les composants du système à diagnostiquer. Puis, nous avons abordé les containers des connaissances : le vocabulaire, les mesures de similarité, l'adaptation et la base de cas, qui sont exploités par le système de RàPC. Nous associons ces containers de connaissances aux ressources du domaine, tels que les bases de données et les documents techniques dans un processus « off-line ». Les cinq phases du cycle de RàPC sont en liaison directe avec les connaissances du système de RàPC et sont considérées dans le processus « on-line ».

Cette première partie d'étude nous a permis de poser les jalons afin d'avoir des critères de comparaison entre les systèmes de diagnostic par RàPC. En effet, nous avons réalisé un état de l'art de ces systèmes traitant les points suivants : la modélisation des connaissances utilisées, la représentation du cas, les phases de remémoration, d'adaptation et de maintenance de la base de cas. A partir de ces points, nous avons analysé et comparé ces systèmes. Nous avons constaté qu'il y a deux types de systèmes : les systèmes orientés *mining* et les systèmes orientés *connaissances*. Chacun des deux types de systèmes présentent des avantages et des inconvénients dans leurs constructions, représentations et la spécificité des phases du cycle. Par conséquent, nous avons décidé de construire nos propres systèmes en proposant des contributions dans les différentes phases du cycle.

Nous avons ainsi développé pour chaque type de système ce qui suit :

- **Système orienté mining**

Le système orienté mining manipule des cas qui ont une représentation complète, triviale et contiennent toute la connaissance de l'équipement à diagnostiquer. Cette représentation est réalisée par l'utilisation des outils de sûreté de fonctionnement à savoir l'AMDEC, les arbres de défaillances, l'historique des pannes. En effet, cette représentation est sous forme de liste d'attributs-valeurs et permet d'une part, de refléter l'ensemble des composants associés à chaque zone géographique, leur valeur et les états de ces composants et d'autre part, de représenter l'évaluation et la solution de la situation diagnostiquée. Nous avons mis en place une phase de remémoration simple qui prend sur un algorithme de recherche classiques (k plus proches voisins) ainsi que sur une mesure de similarité qui tient compte des différents types de valeurs (numériques et modales) et de la présence des valeurs d'attributs.

Ce type de système s'apparente aux systèmes d'apprentissage automatique et nous permet de nous affranchir des problèmes de modélisation des connaissances. Cependant, ce système peut arriver facilement à une explosion combinatoire de cas. Par conséquent, nous nous sommes intéressés à l'étude de la maintenance de la base de cas. Nous avons établi un état de l'art dans le cadre de ce processus de maintenance. Nous avons commencé par l'analyse de la qualité de la base de cas qui passe par l'analyse des critères d'évaluation. Les deux principaux critères sont la compétence et la performance. La compétence dépend de deux notions, à savoir : le recouvrement et l'atteignabilité. Améliorer la compétence de la base de cas consiste à augmenter son taux de recouvrement et diminuer son taux d'atteignabilité. La performance dépend du nombre de cas de la base de cas et du taux de bonnes classifications de celle-ci. Notre objectif est d'optimiser la base de cas. De ce fait, nous proposons une méthode de structuration de la base de cas prenant appui sur un algorithme de catégorisation et une mesure de compétence « *MC* ». La mesure MC reflète le pouvoir de résolution de la base de cas suivant le critère de compétence. Quant à l'algorithme, il permet de définir les quatre types de cas : pivot, support, auxiliaire et de couverture (intra-classes et inter-classes). L'algorithme consiste à garder uniquement les cas pivots et les cas de couverture inter-classes car se sont les cas qui contribuent directement à la compétence et la performance de la base de cas.

En effet, notre méthode proposée est une combinaison originale de deux méthodes issues de deux différentes stratégies à savoir : la stratégie *Backward Elimination (BE)* et la *stratégie Forward Selection (FS)*. En effet, la catégorisation de la base de cas est inspirée de la catégorisation de Smyth et McKenna [1995] issue de la stratégie (BE) associée à une mesure inspirée des travaux de Smyth et McKenna [1998] issue de la stratégie (FS).

Par ailleurs, la base de cas est amenée à évoluer dans le temps par l'introduction de nouveaux cas. Nous avons donc proposé une méthode d'auto-incrémentation des cas dans la base de cas qui doit se faire d'une manière dynamique. Notre méthode s'appuie sur un algorithme d'auto-incrémentation et sur une mesure qui prend en compte les critères de qualité de la base de cas issus de la phase de structuration. L'algorithme permet ainsi d'introduire les cas dans la base de cas tout en conservant sa structuration et de maintenir un taux de bonnes classifications.

Toutefois, un deuxième système, orienté connaissances, peut être mis en place pour réduire la taille de la base de cas. En effet, il dispose de cas générique et d'une phase

d'adaptation développée qui fait la spécification des systèmes de RàPC d'après [Lieber, 2007].

- **Système orienté connaissances**

Le système orienté connaissances manipule des cas qui ont une représentation générique, plus complexe et orientée objet. Cette représentation permet d'avoir une classification hiérarchique des attributs et prend appui sur la définition du diagnostic de l'AFNOR [Afnor, 2001] qui comporte trois parties. La première partie concerne la localisation de la zone de défaillance. La deuxième partie « fonctionnelle » est caractérisée par la valeur du composant associé à une classe fonctionnelle dans le modèle hiérarchique, son état et son mode de fonctionnement. Les modes de fonctionnement sont définis par des règles de décisions. Ces deux parties forment la partie problème du cas. La troisième partie, formant la partie solution du cas, concerne l'identification du composant défaillant et sa classe fonctionnelle. Les cas sont associés à deux modèles de connaissances de l'équipement à diagnostiquer, à savoir : le modèle de contexte et le modèle de taxonomie des composants. En effet, l'analyse de l'équipement détermine des ensembles de ses composants. Chaque ensemble regroupe les composants assurant les mêmes fonctions. Cette analyse fonctionnelle permet donc d'avoir une modélisation hiérarchique de l'équipement. Le modèle de contexte est fondé sur les flux des entités de l'équipement industriel étudié et sur l'analyse de sa décomposition. Cette décomposition détermine les fonctions assurées par l'équipement et ses composants. Nous associons ces deux modèles à la base de cas qui reflète l'étude dysfonctionnelle des composants. Contrairement aux systèmes orientés mining, nous mettons un soin particulier à l'étude des deux phases de remémoration et d'adaptation. Nous avons établi un état de l'art de ces deux phases. Nous avons tenu compte, à partir des travaux de Smyth et Keane [1998], du lien entre ces deux phases. Dans cette mouvance, nous avons proposé une méthode de remémoration guidée par l'adaptation en mettant en place deux mesures : la mesure de remémoration (M_R) et la mesure d'adaptation (M_A). La mesure de remémoration est définie par quatre mesures de similarité locales, associées à chaque type de descripteurs, relatives à l'état du composant, à sa classe d'appartenance, à son mode de fonctionnement et à sa présence. La mesure d'adaptation a pour objectif de choisir, parmi les cas remémorés, le cas le plus facilement adaptable. Cette mesure tient compte du mode de fonctionnement des composants en mettant un poids λ_i. Ce poids est important dans la détermination du composant défaillant.

Ensuite, nous avons proposé un algorithme d'adaptation dédié aux systèmes de diagnostic par RàPC. Cet algorithme s'appuie sur les relations de dépendance des descripteurs de problème et de solution. Ces relations de dépendance sont primordiales dans la détermination des étapes d'adaptation.

Enfin, nous avons appliqué notre méthode de remémoration guidée par l'adaptation ainsi que l'algorithme d'adaptation sur un équipement industriel à savoir un moteur à explosion de la société « Renault ». En effet, nous avons traité trois cas types de diagnostic en fonction de la nature des relations de dépendance. Nous avons conclu que les cas les plus similaires au cas cible ne sont pas forcément ceux choisis pour la phase d'adaptation.

Afin de prouver la faisabilité de la mise en place d'un système de diagnostic par RàPC orienté connaissances ainsi que notre méthode de remémoration guidée par l'adaptation, nous l'avons appliqué sur un système industriel supervisé de transfert de palettes SISTRE.

Nos travaux de thèse et nos contributions ouvrent la voie à de nombreuses perspectives.

Perspectives

Plusieurs perspectives se dégagent suite au travail réalisé. Des directions de recherche et des extensions sont envisageables. Nous pensons particulièrement aux points suivants :

1. Nous avons développé une méthode de maintenance de la base de cas pour les systèmes case-base mining qui traite uniquement des cas simples. Nous envisageons de l'étendre d'une part, pour les autres containers de connaissances et d'autre part, pour les systèmes orientés connaissances. En effet, il serait intéressant de maintenir toute la connaissance du système et pas seulement la base de cas. Pour cela, nous envisageons une proposition d'une maintenance du container de similarité en faisant évoluer les mesures. Au lieu d'avoir une mesure de similarité figée tout au long du cycle de vie du système de RàPC, nous pouvons mettre en place une mesure qui s'adapte au fur et à mesure du nombre de cas et des critères de qualité de la base de cas grâce à un processus d'apprentissage. Concernant le container d'adaptation, nous pouvons établir des règles évolutives qui changent suivant un facteur exprimant les solutions disponibles dans la base de cas. Quant au container de vocabulaire, il permet via un expert la définition d'un système de RàPC. C'est un point non

négligeable dans notre étude car il contribue à la création du système. Nous envisageons donc de prendre en considération sa maintenance dans des travaux futurs.

Concernant la maintenance des systèmes orientés connaissances, nous voulons développer une méthode qui tient compte des modèles de connaissances. Cette méthode s'appuiera sur les mêmes critères utilisés pour la base de cas et devra gérer les relations entre les nœuds et les arcs des modèles et notamment du modèle de taxonomie des composants. L'évolution des modèles de connaissances est également un point important à développer car les équipements peuvent évoluer à n'importe quel moment en rajoutant d'autres composants.

2. Un autre point à considérer concerne le développement de la méthode de remémoration guidée par l'adaptation. Ce principe étant très important dans les travaux de RàPC, nous envisageons de mettre en place une méthode générique pour toutes les applications quelque soit leur nature. Cette méthode tiendra compte de l'ensemble des structures des cas et s'adaptera à la nature des descripteurs utilisés. Une stratégie de mise à jour est également envisageable permettant de faire évoluer les poids des descripteurs des cas de la base de cas.

3. Une autre piste envisageable consisterait à traiter plusieurs palettes à la fois dans l'équipement industriel SISTRE. Dans cette optique, nous serons amenés à traiter l'aspect multi-fautes du diagnostic. Deux solutions possibles sont à envisager. La première consiste à gérer plusieurs cas à la fois lors de l'apparition d'une défaillance. Ce qui nous amène à mettre en place une mesure permettant de choisir plusieurs cas pour l'adaptation. Nous parlerons alors d'une adaptation compositionnelle. Nous combinerons chaque solution proposée par un cas, suivant les renseignements de la partie problème du cas cible, pour former la solution finale au problème rencontré. La deuxième solution consiste à garder les deux mesures créées et de modifier les relations de dépendance entre les descripteurs de problème et de solution. Ces relations vont permettre de gérer les composants de plusieurs natures qui se trouvent dans des zones différentes selon la partie problème du cas cible et le flux de la palette. Ce changement va se répercuter sur le modèle de contexte qui sera plus élargi.

4. Nous comptons établir un transfert de nos méthodes proposées vers des applications industrielles grandeur nature en lien avec la startup « emasystec (http://emasystec.com/) ». En effet, nous souhaiterons intégrer nos deux systèmes d'aide au diagnostic par RàPC dans une plateforme d'e-maintenance sous forme de module. Ce module sera en liaison avec d'autres

modules représentant des bases de connaissances, des systèmes d'acquisition de données et une interface homme-machine.

Références bibliographiques

[Aamodt, 2004] Aamodt A., Knowledge-Intensive Case-Based Reasoning and Sustained Learning. *Proc. of the 9th European Conference on Artificial Intelligence, ECCBR'04*, Lecture Notes in Artificial Intelligence, pp.1-15, Springer, 2004.

[Aamodt & Plaza, 1994] Aamodt A. et Plaza E., Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications*, 7(i): pp 39 59, 1994.

[Afnor, 2001] Maintenance terminology. *European standard*, NF EN 13306, 2001.

[Afouba et al., 2004] Afouba N.M., Kerbata S. et Labarang Z., Le raisonnement à partir de cas : définitions et principes de fonctionnement. *Rapport du projet d'intelligence artificielle*, novembre 2004.

[Aha et al., 1991] Aha D.W., Kibler D., et Albert M.K. Instance based learning algorithms. *Machine Learning*, 6(1): 37–66, 1991.

[Aha et al., 2001] Aha D.W., Breslow L.A. et Muñoz-Avila H., Conversational case-based reasoning. *Applied Intelligence*, vol 14, pp.9-32, 2001.

[Althoff, 2001] Althoff K.D., Case-Based Reasoning. S.K. Chang (Ed.), *Handbook on Software Engineering and Knowledge Management*, pp. 549-588, 2001.

[Althoff & Wess, 1991] Althoff K.D., et Wess S., Case-Based Knowledge Acquisition, Learning and Problem Solving for Diagnostic Real World Tasks. *Proceedings EKAW-91, European Knowledge Acquisition Workshop*, 1991.

[Althoff et al., 1989] Althoff K.D., De la Ossa A., Maurer F., Stadler M. et Wess S., Adaptive Learning in the Domain of Technical Diagnosis. *Proceeding of Workshop on Adaptive Learning, FAW Ulm*, July 1989.

[Althoff et al., 1995] Althoff K., Auriol E., Bergmann R., Breen S., Dittrich S., Johnston R., Manago M., Traphöner R. et Wess S., Case-based reasoning for decision support and diagnostic problem solving: The INRECA Approach. *Proc. of the 3rd German Workshop on CBR*, University of Kaiserslautern, 1995.

[Althoff & Bartsch-Spörl, 1996] Althoff K.D. et Bartsch-Spörl B., Decision support for case based application, *Wirtschaftsinformatik,* ISSN 0937-6429, 1996, vol. 38, no1, pp. 8-16, 1996.

[Anand et al., 1998] Anand S.S., Patterson D., Hugues J.G. et Bell D., Discovering Case Knowledge Using Data Mining. In X. Wu, K. Ramamohanarao & K. B. Korb, Eds., *Research and Development in Knowledge Discovery and Data Mining, Second Pacific-Asia Conference, PAKDD-98, Melbourne, Australia*, April 15-17, 1998, Proceedings, Lecture Notes in Artificial Intelligence 1394, pp. 25–35. Springer, 1998.

[Arshadi & Badie, 2000] Arshadi N. et Badie K., A Compositional Approach to Solution Adaptation in Case-Based Reasoning and its Application to Tutoring Library. *Proceedings of the 8th German Workshop on Case Based Reasoning (GWCBR'00),* Germany, 2000.

[Arcos & Lopez de Mantaras, 1997] Arcos J.L. et López de Mántaras R., Perspectives: A declarative bias mechanism for case retrieval. *In Proceedings of the Second International Conference on Case-Based Reasoning*. Berlin: Springer, pp. 279-290, 1997.

[Armaghan et al., 2008] Armaghan N., Lieber J. et Renaud J., Vers un système de raisonnement à partir de cas conversationnel pour assister des opérateurs d'un service après vente. *16ème atelier de raisonnement à partir de cas*. Nancy, 2008.

[Bareiss et al., 1993] Bareiss E., Porter B. et Wier C., PROTOS: un système apprenti utilisant des cas typiques. *Apprentissage symbolique : une approche de l'intelligence artificielle*, Vol. 1, pp. 105-120, 1993.

[Bentebibel, 2008] Bentebibel R., Raisonnement à Partir de Cas Textuel pour la sécurité routière. Le système SAARA. *Thèse de doctorat*. Laboratoire CRIP5 de l'université Descartes Paris V, Décembre 2008.

[Bentley, 1975] Bentley J.L., Multidimensional Binary Trees Used for Associative Searching. Communications of the ACM, 18(9): 509-517, September 1975.

[Bergmann & Althoff, 1998] Bergmann R. et Althoff K., Methodology for Building Case-Based Reasoning Applications. In *LNAI 1400, ed. Springer*, 1998

[Bergmann & Stahl, 1998] Bergmann R. et Stahl A., Similarity measures for object-oriented case representations. *In Proceedings of the Fourth European Workshop on Case-Based Reasoning*. Berlin: Springer, pp. 25 36, 1998.

[Bergmann & Wilke, 1998] Bergmann R. et Wilke W., Towards a new formal model of transformational adaptation in case-based reasoning. *In the 13th European Conference on Artificial Intelligence, (ECAI'98)*, pages 53–57, Brighton, UK, August 1998.

[Bergmann, 2002] Bergmann R., Experience Management: Foundations, Development Methodology, and Internet-Based Applications. Berlin: *Springer*, 2002.

[Bergmann et al., 2003] Bergmann R., Althoff K.D., Breen S., Göker M., Manago M., Traphöner R. et Weiss S., Developing Industrial Case-Based Reasoning Applications: The INRECA Methodology. *Lecture Notes in Artificial Intelligence*, LNAI 1612, Springer Verlag, Berlin, 2003.

[Blum & Langey, 1997] Blum A. et Langley P., Selection of relevant features and examples in machine learning. *Artificial Intelligence*, **97**: 245–271, 1997.

[Börner, 1998] Börner, K., CBR for Design. Case-Based Reasoning Technology: From Foundations to Applications. *Lecture Notes in Artificial Intelligence*, Springer Verlag, pp. 201-234, 1998.

[Bouchon-Meunier & Christophe, 2003] Bouchon-Meunier B. et Christophe M., Logique floue, principes, aide à la décision. Traité IC2, *série informatique et systèmes d'information*. Lavoisier, 2003.

[Bridge & Ferguson, 2002] Bridge D. et Ferguson A., Diverse product recommendations using an expressive language for case retrieval. *In Proceedings of the Sixth European Conference on Case-Based Reasoning*. Berlin: Springer, pp. 43-57, 2002.

[Brighton & Mellish, 2002] Brighton H. et Mellish C., On the consistency of information filters for lazy learning algorithms. In *Principles of Data Mining and Knowledge Discovery*, Third European Conference, PKDD '02, Proceedings, 2002.

[Brown, 1994] Brown M.G., An underlying memory model to support case retrieval. In Wess, S., Althoff, K.-D. & Richter, M. (eds.), *Topics in Case-Based Reasoning*. Berlin: Springer, pp. 132-143, 1994.

[Bunke & Messmer, 1993] Bergmann R. et Stahl A., Similarity measures for object-oriented case representations. *In Proceedings of the Fourth European Workshop on Case-Based Reasoning*. Berlin: Springer, pp. 25 36, 1998.

[Burke et al., 1997] Burke R., Hammond K., Kulyukin V., Lytinen S., Tomuro N., et Schoenberg S., Question Answering from Frequently-Asked Question Files: Experiences with the FAQ Finder System. *Technical Report TR-97-05*, Department of Computer Science, University of Chicago, 1997.

[Cameron-Jones, 1992] Cameron-Jones R.M., Minimum description length instance-based learning. In *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, Hobart, Australia, pp. 368–373, 1992.

[Cano et al.,, 2006] Cano J.R., Herrera F. et Lozano M., On the combination of evolutionary algorithms and stratified strategies for training set selection in data mining. *Applied Soft Computing Journal* 6 : 3 (2006), pp : 323-332, 2006.

[Cao et al., 2003] Cao G., Shiu S.C.K. et Wang X., A Fuzzy-Rough Approach for Case Base Maintenance. In *the 4th International Conference on Case-Based Reasoning, (ICCBR'01)*, 2001: 118-130, 2001.

[Carbonell, 1984] Carbonell J.G., Learning by analogy: Formulating and generalizing plans from past experiences. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning: An Artificial Intelligence Approach*, pp. 137–162. Springer - Berlin, Heidelberg, 1984.

[Carbonell, 1986] Carbonell J.G., Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition. Michalski R., Carbonell J. et Mitchell T. (Eds.), *Machine Learning, an Artificial Intelligence Approach*, Morgan Kaufmann, Vol 2, pp. 271-391, 1986.

[Champin & Solnon, 2003] Champin P. A. et Solnon C., Measuring the similarity of labeled graphs. *In Proceedings of the Fifth International Conference on Case-Based Reasoning*. Berlin: Springer, pp. 80-95, 2003

[Chang, 1974] Chang C.L. Finding prototypes for nearest neighbor classifiers. *IEEE Transactions on Computers*, (23): pp.1179–1184, 1974.

[Charlet et al., 2000] Charlet J., Zacklad M, Kassel G. et Bourigault D., Knowledge engineering: Recent evolutions and new challenges, Editions Eyrolles, Paris, 2000.

[Chebel-Morello et al., 2005] Chebel-Morello B., Rasovska I. et Zerhouni N., Knowledge capitalization in system of equipment diagnosis and repair help. *Proc. of IJCAI'05 Workshop on Knowledge Management and Organizational Memories*, Edinburgh, Scotland, 2005.

[Chebel-Morello et al., 2007] Chebel-Morello B., Haouchine M.K. et Zerhouni N., Auto-incrémentation d'une Base Dysfonctionnelle de Cas pour un Système d'Aide au Diagnostic et à la Réparation. *Actes de la 3ème Edition du Colloque International Francophone sur la Performance et les Nouvelles Technologies en Maintenance, PENTOM'2007*, Mons, Belgique (2007).

[Cheetham & Graf, 1997] Cheetham W. et Graf J., Case-Based Reasoning in Color Matching. *Proceedings of the Second International Conference on Case-Based Reasoning Research and Development*. Lecture Notes In Computer Science; Vol. 1266, pp. 1 – 12, 1997.

[CNTRL, 1979] La Recherche, janv. 1979, N° 96, vol. 10, p. 61, cité par le dictionnaire du CNTRL (Centre National de Ressources Textuelles et Lexicales), 1979.

[Cojan & Lieber, 2008] Cojan J. et Lieber J., Conservative Adaptation in Metric Spaces. *Proceedings of the 9th European conference on Advances in Case-Based Reasoning*, Vol. 5239, pp. 135-149, 2008.

[Collins & Cunningham, 1996] Collins B. et Cunningham P., Adaptation-Guided Retrieval in EBMT: A case based approach to machine translation. *In Proceedings of EWCBR'96*, LNAI 1168, 1996.

[Corchado & Lees, 2001] Corchado J. et Lees B., Adaptation of cases for case-based forecasting with neural network support, In S. K. Pal, T. S. Dillon & D. S. Yeung, Eds., Soft Computing in Case-Based Reasoning, chapter 13, pp. 293–319. Springer, 2001.

[Cordier, 2008] Cordier A., Interactive and Opportunistic Knowledge Acquisition in Case-Based Reasoning. *Thèse de doctorat*, Laboratoire d'InfoRmatique en Images et Systèmes d'information, Université de Lyon I, Novembre 2008.

[Cordier et al., 2006] Cordier A., Fuchs B., et Mille A., Engineering and Learning of Adaptation Knowledge in Case-Based Reasoning. *15th International Conference on Knowledge Engineering and Knowledge Management EKAW'06,* S. Staab and V. Svatek ed. Podebrady, Czech Republic. pp. 303-317. LNAI 4248. Springer-Verlag Berlin Heidelberg. 2006.

[Cordier et al., 2007] Amélie Cordier, Béatrice Fuchs, Jean Lieber et Alain Mille. Acquisition interactive des connaissances d'adaptation intégrée aux sessions de raisonnement à partir de cas — Principes, architecture IAKA et prototype KAYAK, 2007.

[Cunningham & Smyth, 1994] Cunningham P. et Smyth B., A Comparison of Model-Based and Incremental Case-Based Approaches to Electronic Fault Diagnosis. In: *Proceedings of the Case-Based Reasoning Workshop*, AAAI (American Association of Artificial Intelligence) 1994, Seattle, USA, 1994.

[Cunningham et al., 2003] Cunningham P., Doyle D. et Loughrey J., An evaluation of the usefulness of case-based explanation. *In Proceedings of the Fifth International Conference on Case-Based Reasoning*. Berlin: Springer, pp. 122-130, 2003.

[d'Aquin et al., 2004] d'Aquin M., Brachais S., Lieber J., et Napoli A., Vers une acquisition automatique de connaissances d'adaptation par examen de la base de cas - une approche fondée sur des techniques d'extraction de connaissances dans des bases de données. In *12ème Atelier de Raisonnement à Partir de Cas - RàPC'04*, (Université Paris Nord, Villetaneuse, France), pp.41-52, 2004.

[Dasarathy, 1991] Dasarathy B.V., Nearest Neighbor Norms: NN Pattern Classification Techniques. *IEEE Press*, Los Alamitos, California, 1991.

[Devany & Cheetham, 2005] Devaney M. et Cheetham B., Case-Based Reasoning for Gas Turbine Diagnostics. In *18th International FLAIRS Conference (FLAIRS-05)*, 2005.

[Doyle et al., 2004] Doyle D., Cunningham P., Bridge D. et Rahman Y., Explanation oriented retrieval. In *Proceedings of the Seventh European Conference on Case-Based Reasoning*. Berlin: Springer, pp. 157-168, 2004.

[Dubuisson et al., 2001] Dubuisson B., Boutleux E., Dague P., Denoeux T., Didelet E., Gandvalet Y. et Masson M., Diagnostic, intelligence artificielle et reconnaissance de formes, *Hermès*, 2001.

[Falting, 1997] Faltings B., Case reuse by model-based interpretation. In Maher, M. L. & Pu, P. (eds.) *Issues and Applications of Case-Based Reasoning in Design*. Mahwah, NJ: Lawrence Erlbaum, pp. 39–60, 1997.

[Fdez-Riverola & Corchado, 2003] Fdez-Riverola F. et Corchado J.M., Using instance-based reasoning systems for changing environments forecasting. In *Workshop on Applying case-based reasoning to time series prediction*, Trondheim, Norway, pp. 219-228, 2003.

[Finnie, 2003] Finnie G. et Sun Z., R5 model for case-based reasoning. *Knowledge-Based Systems*, (16): 59–65, 2003.

[Fisher & Patrick, 1970] Fisher F.P. et Patrick E.A., A Preprocessing Algorithm for Nearest Neighbor Decision Rules. In *Proceedings of the National Electronic Conference*, pp. 481-485, December 1970.

[Flores-Loredo et al., 2005] Flores-Loredo Z., Ibarguengoytia P.H., Morales E.F., On line diagnosis of gas turbines using probabilistic and qualitative reasoning. Intelligent Systems Application to Power Systems, 2005. *Proceedings of the 13th International Conference*, Volume , Issue , 6-10 Nov. Page(s): 5 pp, 2005.

[Friedman et al., 1975] Friedman J.H., Bentley J.L. et Finkel R.A., An Algorithm Finding Nearest Neighbors. IEEE Transactions on Computers 1000-1006, 1975.

[Fuchs, 1997] Fuchs B., Représentation des connaissances pour le raisonnement à partir de cas : Le système ROCADE. *Thèse de doctorat*, Laboratoire Image Signal Acoustique de CPE Lyon, 1997.

[Fuchs & Mille, 2005] B. Fuchs et A. Mille., Une modélisation au niveau connaissance du raisonnement à partir de cas. In *ingénierie des connaissances*, R. Teulier, J. Charlet, P. Tchounikine éditeurs, L'Harmattan, 2005.

[Fuchs et al., 1999] Fuchs B., Lieber J., Mille A., et Napoli A., Towards a unified theory of adaptation in case-based reasoning. In Case Based Reasoning Research and Development, *The 3rd International Conference on Case-Based Reasoning (ICCBR'99)*, volume 1650 de LNAI, pages 104–117, Seeon Monastery, Germany, August 25-27 1999. Springer Verlag.

[Fuchs et al., 2000] Fuchs B., Lieber J., Mille A., et Napoli A., An algorithm for adaptation in case-based reasoning. *In the 14th European Conference on Artificial Intelligence (ECAI'00)*, volume 14, pages 45–49, Amsterdam, The Netherlands, August 20-25 2000. IOS Press.

[Fuchs et al., 2006] Fuchs B., Lieber J., Mille A. et Napoli A., Une première formalisation de la phase d'élaboration du raisonnement à partir de cas. *Actes du 14ième atelier du raisonnement à partir de cas*, Besançon, mars, 2006.

[Fukunaga & Narendra, 1975] Fukunaga K. et Narendra P.M. A Branch and Bound Algorithm for Computing k-Nearest Neighbors. *IEEE Transactions on Computers*, pp. 750-753, July 1975.

[Gabel, 2004] Gabel T., On the Use of Vocabulary Knowledge for Learning Similarity Measures, In *Proceedings of the 3rd German Workshop on Experience Management*, Springer, 2004.

[Gates, 1972] Gates G.W., The reduced nearest neighbor rule. *IEEE Transactions on Information Theory*, 18(3):431–433, 1972.

[Gebhardt et al., 1997] Gebhardt F., Voß A., Gräther W., Schmidt-Belz B., Reasoning With Complex Cases. *Kluwer academic publishers*, 1997.

[Gentner & Forbus, 1991] Gentner D. et Forbus K., MAC/FAC: A model of similarity-based retrieval. *In Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Lawrence Erlbaum, pp. 504-509, 1991.

[Goel & Chandrasekaran, 1989] Goel A. et Chandrasekaran B., Use of device models in adaptation of design cases. *In Proceedings of the DARPA Workshop on Case-Based Reasoning*. San Mateo, CA: Morgan Kaufmann, pp. 100-109, 1989.

[Gómez de Silva Garza & Maher, 2000] Gómez de Silva Garza A. et Maher M.L., A process model for evolutionary design case adaptation. In *Proceedings of the Artificial*

Intelligence in Design Conference. Dordrecht: Kluwer Academic Publishers, pp. 393–412, 2000.

[Grant et al., 1996] Grant P.W., Harris P.M. et Moseley L.G., Fault Diagnosis for Industrial Printers Using Case-Based Reasoning. *Engineering Applications of Artificial Intelligence*. Vol.9, No.2, pp.163-173, 1996.

[Grosclaude, 2001] Grosclaude I.., Diagnostic abductif temporel - scénarios de pannes, modèles causaux et traitement de l'information. *Thèse de Doctorat*. Université de Rennes I. (2001).

[Hammond, 1986] Hammond, K.J., CHEF: A model of case-based planning. In Press, A., editor, *Proceedings of the 5th National Conference on Artificial Intelligence*, pages 267–271, Menlo Park, CA, 1986.

[Hammond, 1990] Hammond, K.J., Explaining and Repairing Plans That Fail. In *Artificial Intelligence*, 45, 173–228, 1990.

[Hanney & Keane, 1996] Hanney K. et Keane M.T., Learning Adaptation Rules from a Case-Base. In *Proceedings of the 3rd European Workshop on Advances in Case-Based Reasoning*, Lecture Notes In Computer Science, 1996.

[Haouchine, 2005] Haouchine M.K., Maintenance d'une base de cas dédiée au diagnostic de pannes et à sa réparation. *Mémoire de Master Recherche, spécialité : mécatronique*. Université de Franche-Comté, juillet 2005.

[Haouchine et al., 2006a] Haouchine M.K., Chebel-Morello B. et Zerhouni N., Maintenance d'un Système de Raisonnement à Partir de Cas. *Actes de la Conférence International sur le Contrôle, la Modélisation et le Diagnostic, ICCMD'2006*, Annaba, Algérie, 2006a.

[Haouchine et al., 2006b] Haouchine M.K., Chebel-Morello B. et Zerhouni N., Méthode de Suppression de Cas pour une Maintenance de Base de Cas. *Actes du 14ème Atelier de Raisonnement à Partir de Cas, RàPC'2006*, Besançon, France, 2006b.

[Haouchine et al., 2007a] Haouchine M.K., Chebel-Morello B. et Zerhouni N., Case Base Maintenance Approach. In *Proceedings of International Conference on Industrial Engineering and Systems Management, IESM'2007*, Beijing, Chine, 2007a.

[Haouchine et al., 2007b] Haouchine M.K., Chebel-Morello B. et Zerhouni N., Evolution d'un Système de Raisonnement à Partir de Cas Dédié au Diagnostic Industriel. *Actes du 15ème Atelier de Raisonnement à Partir de Cas, RàPC'2007*, Grenoble, France 2007b.

[Haouchine et al., 2008a] Haouchine M.K., Chebel-Morello B. et Zerhouni N., Competence-Preserving Case-Deletion Strategy for Case-Base Maintenance. *Uncertainty, Similarity, and Knowledge Discovery in Case-Based Reasoning workshop. 9th European Conference on Case-Based Reasoning, ECCBR 2008*, Trier, Germany, 2008a.

[Haouchine et al., 2008b] Haouchine M.K., Chebel-Morello B. et Zerhouni N., Adaptation-Guided Retrieval for a Diagnostic and Repair Help System Dedicated to a Pallets Transfer. In *3rd European Workshop on Case-Based Reasoning and Context-Awareness. 9th European Conference on Case-Based Reasoning, ECCBR 2008*, Trier, Germany, 2008b.

[Haouchine et al., 2008c] Haouchine M.K., Chebel-Morello B. et Zerhouni N., Algorithme d'Adaptation pour le Diagnostic Technique. *Actes du 16ème Atelier de Raisonnement à Partir de Cas, RàPC'2008*, Nancy, France, 2008c.

[Haouchine et al., 2009] Haouchine M.K., Chebel-Morello B. et Zerhouni N., Auto-increment of expertise for failure diagnostic. A paraître dans *13th IFAC Symposium on Information Control Problems in Manufacturing INCOM'09*, Moscow, Russie, Juin 2009.

[Hart, 1967] Hart P.E., The Condensed Nearest Neighbor Rule. *IEEE Transactions on Information Theory*, (14): 515-516, 1967.

[Iglezakis & Roth-Berghofer, 2000] Iglezakis I. et Roth-Berghofer T., A survey regarding the central role of the case base for maintenance in case-based reasoning, In Mirjam Minor, editor, *ECAI Workshop*, Notes, pp. 22–28. Humboldt–Universität zu Berlin, 2000.

[Iglezakis et al., 2004] Iglezakis I., Reinartz T. et Roth-Berghofer T., Maintenance Memories: Beyond Concepts and Techniques for Case Base Maintenance. In *the 7h European Conference on Case-Based Reasonning (ECCBR'04)*, 227-241, 2004.

[Jaczynski & Trousse, 1998] Jaczynski M. et Trousse B., WWW assisted browsing by reusing past navigations of a group of users. *In the 4th European Workshop on Case-Based Reasoning (EWCBR'98)*, volume 1488 de LNCS, pages 160–171, Dublin, Ireland, 1998.

[Jarmulak et al., 2001] Jarmulak J., Craw S. et Rowe R., Using Case-Base Data to Learn Adaptation Knowledge for Design. In Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI'01), p. 1011–1016: Morgan Kaufmann, Inc, 2001

[Karoui et al., 2006] Karoui H., Kanawati R. et Petrucci L., COBRAS: Cooperative CBR system for bibliographical reference recommendation. *In the 8th European Conference on Case-Based Reasonning (ECCBR'06)*, éditeurs Thomas Roth-Berghofer, Mehmet H. Göker, et H. Altay Güvenir, volume 4106 de LNCS, pages 76–90, Ölüdeniz/Fethiye, Turkey, September 2006. Springer.

[Kasif et al., 1995] Kasif S., Salzberg S., Waltz D., Rachlin J. et Aha D., Towards a Framework for Memory-Based Reasoning. *NECI Technical Report* pp. 95-132, 1995.

[Koehler, 1996] Koehler J., Planning from Second Principles. *Artificial Intelligence*, 87, pp. 145-186. 1996.

[Kolodner, 1988] Kolodner, J., Workshop on case-based Reasoning, editor (1988) *DARPA 88*, Clearwater, Florida. Morgan Kaufmann, San Mateo.

[Kolodner, 1993] Kolodner J., Case-Based Reasoning. *Morgan Kaufmann*, San Mateo, UCA, 1993.

[Kolodner, 1996] Kolodner J., Making the implicit explicit: Clarifying the principles of case-based reasoning. *In Case-Based Reasoning: Experiences, Lessons and Future Directions*, pages 349–370, *AAAI press*, Mit Press, 1996.

[Koton, 1988] Koton, P., Reasoning about evidence in causal explanations. In Press, A., editor, *Proceedings of the Seventh National conference on Artificial Intelligence*, pages 256–261, Menlo Park, CA, 1988.

[Lamontagne & Lapalme, 2002] Lamontagne L. et Lapalme G., 2002. Raisonnement à base de cas textuels – état de l'art et perspectives, *Revue d'Intelligence Artificielle*, Hermes, Paris, vol. 16, no. 3, pp. 339-366.

[Lamontagne, 2004] Lamontagne L., Une approche CBR textuel de réponse au courrier électronique. *Thèse de doctorat*, Département d'informatique et de recherche opérationnelle, Université de Montréal, 2004.

[Leake & McSherry, 2005] Leake D. et McSherry D., Introduction to the special issue on explanation in case-based reasoning. *Artificial Intelligence Review* 24(2), 103-108, 2005.

[Leake & Wilson, 1998] Leake D.B. et Wilson D.C., Categorizing case-base maintenance: dimensions and directions. *Advances in Case-Based Reasoning: 4th European Workshop, EWCBR 98*. Proceedings. 1998: pp.196-207, 1998. Springer- Verlag, Berlin, Germany.

[Leake & Wilson, 2000a] Leake D.B. et Wilson D.C., Remembering Why To Remember: Performance-guided case-base maintenance. *Advances in Case-Based Reasoning: Proceeding of EWCBR-2K*, Springer-Verlag, 2000a.

[Leake & Wilson, 2000a] Leake D.B. et Wilson D.C., Guiding Case-Base Maintenance: Competence and Performance. Online *Proceedings of the ECAI'2000 Workshop on Flexible Strategies for Maintaining Knowledge Containers*, 2000b.

[Leake et al., 1996a] Leake D.B., Kinley A. et Wilson D., Linking Adaptation and Similarity Learning. *Proc. of the 8th Annual Conference of the Cognitive Science Society*, 1996a.

[Leake et al., 1996b] Leake D.B., Kinley A. et Wilson D.C., Acquiring Case Adaptation Knowledge: A Hybrid Approach. In *AAAI/IAAI*, volume 1, p. 684–689, 1996b.

[Leake et al., 1997] Leake D.B., Kinley, A. et Wilson D.C., Case-Based Similarity Assessment: Estimating Adaptability from Experience. *In Fourteenth National Conference on Artificial Intelligence*, pages 674–679, Menlo Park, CA. AAAI Press, 1997.

[Lee, 2003] Lee M., A study of an automatic learning model of adaptation knowledge for case base reasoning. *Inf. Sci*., 155(1-2), 61–78, 2003.

[Lekkas et al., 1994] Lekkas G.P., Avouris N.M. et Viras L.G., CBR in Environmental Monitoring Applications. *Applied Artificial Intelligence*, vol. 8, pp. 359-376, 1994.

[Lenz, 1996] Lenz M., Applying case retrieval nets to diagnostic tasks in technical domains. *In Proceedings of the Third European Workshop on Case-Based Reasoning*. Berlin: Springer, pp. 219-233, 1996.

[Lenz, 1999] Lenz M., Case Retrieval Nets as a Model for Building Flexible Information Systems. *Thèse de doctorat*, Université de Humboldt, 1999.

[Lereno, 2000] Lereno E., Apprentissage des problèmes d'ordonnancement : application des méthodes de filtrage de données. *Thèse de doctorat*, l'UFR des Sciences et Techniques de l'Université de Franche-Comté, Décembre 2000.

[Lieber, 1999] Lieber J., Reformulations and Adaptation Decomposition. In Schmitt, S. and Vollrath, I., editors, *Proceedings of the 3rd International Conference on Case Based Reasoning (ICCBR'99)*, Munich, Germany. LSA, University of Kaiserslautern, 1999.

[Lieber, 2002] Lieber J., Recopier c'est déjà adapter : six types d'adaptation par copie. In *10ème séminaire français de raisonnement à partir de cas* - RàPC'2002, pp. 11-21, France, 2002.

[Lieber & Napoli, 1998] Lieber J. et Napoli A., Correct and Complete Retrieval for Case-Based Problem-Solving. In Prade, H., editor, *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI'98)*, pages 68–72, Brighton, United Kingdom.

[Lieber et al., 2003] Lieber J., d'Aquin M., Bey P., Napoli A., Rios M. et Sauvagnac C., Acquisition of Adaptation Knowledge for Breast Cancer Treatment Decision Support. In *9th Conference on Artificial Intelligence in Medicine in Europe2003 - AIME 2003*, Protaras, Chypre, 2003.

[Lieber et al., 2004] Lieber J., d'Aquin M., Brachais S. et Napoli A., Une étude comparative de quelques travaux sur l'acquisition des connaissances d'adaptation pour le raisonnement à partir de cas. *In 12ème Atelier de Raisonnement à Partir de Cas (RàPC'04)*, pages 53–60, Villetaneuse, France, Mars 2004.

[Lieber, 2007] Lieber J., Application of the Revision Theory to Adaptation in Case-Based Reasoning: the Conservative Adaptation. *In 7th International Conference on Case-Based Reasoning - ICCBR'07* 4626 (2007) 239-253.

[Liu & Motoda, 2002] Liu H. et Motoda H., On issues of instance selection. *Data Mining and Knowledge Discovery* 6 (2002) pp : 115-130, 2002.

[Lopez de Mantaras et al., 2005] Lopez de Mantaras R., McSherry D., Bridge D., Leake D., Smyth B., Craw S., Faltings B., Maher M.L., Cox M., Forbus K., Keane M., Aamodt A. et Watson I. Retrieval, Reuse, Revise, and Retention in CBR. *Knowledge Engineering Review*, pp : 215-240, 2005.

[Main et al., 2000] Main J., Dillon T.S. et Shiu S.C.K., A tutorial on case-based reasoning. *Soft Computing in Case Based Reasoning*, pages : 1-28, 2000.

[Malek, 2000] Malek M., Hybrid approaches for integrating neural networks and case-based reasoning : From loosely coupled to tightly coupled models. *In Soft Computing in Case Based Reasoning*, éditeurs Tharam S. Dillon Sankar K. Pal et Daniel S. Yeung, pages 73-94, Mars 2000.

[Malek & Kanawati, 2001] Malek M.et Kanawati R., Cobra : A cbr- approach for predicting users actions in web site. *3rd International Conference on Case-Based Reasoning ICCBR'01*, pages 336-346, 2001.

[Malek & Kanawati, 2004] Malek M. et Kanawati R., A data driven CBR approach: A continuous maintenance strategy. *In the International Conference on Advances in Intelligent Systems: Theory and Applications*, pages 281–290, Luxemburg, November 2004.

[Mangalagiu, 1999] Mangalagiu D., Accélération de l'élgorithme des *k* plus proches voisins par réorganisation arborescente de la base de données. *Thèse de doctorat*, Ecole de polytechnique, Novembre 1999.

[Markovitch & Scott, 1988] Markovitch S. et Scott P.D., The Role of Forgetting in Learning. In *Proceedings of the Fifth International Conference on Machine Learning*, pp. 459-465, 1988.

[McGinty & Smyth, 2003] McGinty L. et Smyth B., On the role of diversity in conversational recommender systems. *Proceedings of the Fifth International Conference on Case-Based Reasoning*. Berlin: Springer, pp. 276-290, 2003.

[McKenna & Smyth, 1999] McKenna E. et Smyth B., How does your case base grow? In Proceedings of the 10th Irish Conference on Artificial Intelligence and Cognitive Science, Cork, Ireland, September 1999.

[McSherry, 2002] McSherry D., Diversity-conscious retrieval. *In Proceedings of the Sixth European Conference on Case-Based Reasoning*. Berlin: Springer, pp. 219-233, 2002.

[McSherry, 2003a] McSherry D., Increasing dialogue efficiency in case-based reasoning without loss of solution quality. *In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*. San Francisco, CA: Morgan Kaufmann, pp. 121-126, 2003a.

[McSherry, 2003b] McSherry D., Similarity and compromise. *In Proceedings of the Fifth International Conference on Case-Based Reasoning*. Berlin: Springer, pp. 291-305, 2003b.

[McSherry, 2004] McSherry D., Balancing user satisfaction and cognitive load in coverage-optimised retrieval. *Knowledge-Based Systems* 17, 113-119, 2004.

[McSherry, 2005] McSherry D., Explanation in recommender systems. *Artificial Intelligence Review* 24(2): 179-197, 2005.

[Mille, 1995] Mille A., Raisonnement basé sur l'expérience pour coopérer à la prise de décision, un nouveau paradigme en supervision industrielle. *Thèse de doctorat,* Université de Saint Etienne, 1995.

[Mille, 1999] Mill. A., Tutorial CBR : Etat de l'art de raisonnement à partir de cas. *Plateforme AFIA'99,* Palaiseau, 1999.

[Mille, 2006] Mille A., Traces based reasoning (TBR) definition, illustration and echoes with story telling. *Rapport Technique RR-LIRIS-2006-002*, LIRIS UMR 5205 CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/Ecole Centrale de Lyon, january 2006.

[Mille et al., 1996] Mille A., Fuchs B. et Herbeaux O., A unifying framework for Adaptation in Case-Based Reasoning. In A. Voss, Ed., *Proceedings of the ECAI'96 Workshop: Adaptation in Case-Based Reasoning*, p. 22-28, 1996.

[Minsky, 1975] Minsky M., A framework for representing knowledge. *In the psychology of Computer Vision*, éditeur P.H. (Ed.)Winston, pages 211–279, New York, McGraw Hill, 1975.

[Minton, 1990] Minton S., Qualitative Results Concerning the Utility of Explanation-Based Learning. *Artificial Intelligence*, vol. (42): pp. 363-391, 1990.

[Mougouie & Bergmann, 2002] Mougouie B. et Bergmann R., Similarity assessment for generalized cases by optimization methods. *In Proceedings of the Sixth European Conference on Case-Based Reasoning*. Berlin: Springer, pp. 249-263, 2002.

[Mujica & Vehi, 2003] Mujica L.E. et Vehi J., Damage identification using case based reasoning and self organizing maps. In *3rd European Symposium on Intelligent Technologies, Hybrid Systems and their implementation on Smart Adaptive Systems*, Oulou, Finland, July 2003.

[Murdock & Goel, 2001] Murdock J.W., Goel A.K., Donahoo M.J. et Navathe S., Method-specific knowledge compilation. *Data mining for design and manufacturing: methods and applications, Kluwer Academic Publishers*, Norwell, MA, USA, pp. 443 - 463, 2001.

[Neschen, 1995] Neschen M., Hierarchical Binary Vector Quantisation Classifiers for Handwritten Character Recognition. In *17th DAGM Symposion on Pattern Recognition Bielefeld*, September 13-15, 1995.

[Nunez et al., 2004] Nùñez H., Sànchez-Marrè M., Cortés U., Comas J., Martìnez M., Rodrìguez-Roda I. et Poch M., A comparative study on the use of similarity measures in case-based reasoning to improve the classification of environmental system situations. *Environmental Modelling & Software*, (19): 809–819, 2004.

[Palluat, 2004] Palluat N., Méthodologie de surveillance dynamique à l'aide des réseaux neuro-flous temporels. *Thèse de doctorat*, Ecole Doctorale Sciences Physiques pour l'Ingénieur et Microtechniques, Université de Franche-comté, 2004.

[Pan, 2007] Pan R., Yang Q. et Pan S.J., Mining competent case bases for case-based reasoning. *Artificial Intelligence*, (171): 1039-1068, 2007.

[Park & Sandberg, 1991] Park J. et Sandberg I.W., Universal approximation using radial-basis-functionnetworks. *Neural Computation*, 3(2): 246-257, 1991.

[Pekalska et al., 2006] Pekalska E., Duin R.P.W. et Paclik P., Prototype selection for dissimilarity-based classifiers. *Pattern Recognition* 39 : 2 (2006), pp : 189-208, 2006.

[Peng & Reggia, 1990] Peng Y. et Reggia J.A., Abductive inference models for diagnostic problem solving. *Symbolic Computation*, Springer-Verlag New York, Inc, 1990.

[Perner, 2006] Perner P., Case-base maintenance by conceptual clustering of graphs. *Engineering Applications of Artificial Intelligence*, (19): 381–393, 2006.

[Piechowiak, 2003] Piechowiak S., Intelligence artificielle et diagnostic. *Techniques de l'ingénieur*, S 7 217, décembre, 2003.

[Porter et al., 1990] Porter B., Bareiss R. et Holte R., Concept learning and heuristic classification in weak theory domains. *Artificial Intelligence* 45(1-2), 229-263, 1990.

[Portegys, 1995] A search Technique for Pattern Recognition Using Relative Distances. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(9): 910-914, September 1995.

[Racine & Yang, 1996] Racine K. et Yang Q., On the consistency Management of Large Case Bases: the Case for Validation. In *Proceedings of the AAAI–96 Workshop on Knowledge Base Validation, American Association for Artificial Intelligence. Verification and Validation Workshop*, 1996.

[Racine & Yang, 1997] Racine K. et Yang Q., Maintaining Unstructured Case Bases. *Case-based reasoning research and development, Lecture notes in computer science*: pp. 553-564. Providence RI, July 25-27, 1997.

[Rasovska, 2006] Rasovska I., Contribution à une méthodologie de capitalisation des connaissances basée sur le raisonnement à partir de cas : Application au diagnostic dans une plateforme d'e-maintenance. *Thèse de doctorat*, l'UFR des Sciences et Techniques de l'Université de Franche-Comté, Juillet 2006.

[Reinartz, 2002] Reinartz T., A unifying view on instance selection. *Data Mining and Knowledge Discovery* 6 (2002) 191-210, 2002.

[Reinartz et al., 2000] Reinartz T., Iglezakis I. et Roth-Berghofer T., On quality measure for case base maintenance. *Lecture Note in Artificial Intelligence 1998, Advances in Case-Based Reasoning*, 5th European Workshop, EWCBR 2000, pages 247-259, 2000.

[Richter & Wess, 1991] Richter M.M. et Wess S., Similarity, Uncertainty and Case-Based Reasoning in PATDEX. *Automated reasoning*, essays in honour of Woody Bledsoe, Kluwer, 1991, pp. 249-265.

[Richter et al., 1993] Richter M.M., Wess S., Althoff K.D. et Maurer F., First European Workshop on Case-Based Reasoning, University of Kaiserslautern, Germany, *Lecture Notes in Artificial Intelligence*, vol 837, Springer Verlag, Berlin, 1993.

[Richter, 1995] Richter M.M., The Knowledge Contained in Similarity Measures. Invited talk, *First International Conference on Case-Based Reasoning (ICCBR'95)*, Sesimbra, Portugal, 1995.

[Richter, 1998] Richter M.M., Introduction. *In Case-Based Reasoning Technology: From Foundations to Applications*, Edited by M. Lena, B. Bartsc-Sporl, H. D. Burkhard, et S. Wess. Springer-Verlag, Berlin, pp. 1-15, 1998.

[Richter, 2006] Disponible sur http://wwwagr.informatik.unikl.de/~richter/Vorlesungen/KM/.

[Riesbeck & Schank, 1989] Riesbeck C.K. et Schank R.C., Inside Case-Based Reasoning. *Lawrence Erlbaum Associates*, Cambridge, MA, 1989.

[Rifqi, 1996] Rifqi M., Mesures de comparaison, typicalité et classification d'Objets flous : théorie et pratique. *Thèse d'université*, Université Pierre et Marie Curie, Paris VI, Paris, 1996

[Ritter et al., 1975] Ritter G.L.,Woodruff H.B., Lowry S.R. et Isenhour T.L., An algorithm for the selective nearest neighbour decision rule. *IEEE Transactions on Information Theory*, 21(6):665–669, 1975.

[Roh et al., 2003] Roh T.H., Oh K.J. et Han I., The collaborative filtering recommendation based on som clusteringindexing cbr. In *Expert Systems with Applications*, volume 25, pp. 413-423. Elsevier Science, 2003.

[Roth-Berghofer & Iglezakis, 2001] Roth-Berghfer T. et Iglezzakis I., Six Steps in Case–Based Reasoning: Towards a maintenance methodology for case–based reasoning systems. In *Proceedings of the 9th German Workshop on CBR, GWCBR'01*, Germany, 2001.

[Roth-Berghofer, 2003] Roth-BerghoferT., Developing maintainable CBR Systems: Applying SIAM to empolis orenge. *Wissensmanagement* 2003: 305-306, 2003.

[Rousu & Aarts, 1996] Rousu J. et Aarts R.J., Adaptation Cost as a criterion for solution evaluation. *In Proceedings of EWCBR'96*, LNAI 1168, éd. Springer, 1996.

[Salton, 1989] Salton G., Automatic Text Processing : The Transformation, Analysis, and Retrieval of Information by Computer. *Addison-Wesley*, 1989.

[Schaaf, 1996] Schaaf J. W., Fish & shrink: A next step towards efficient case retrieval in large scale case-bases. *In Proceedings of the Third European Workshop on Case-Based Reasoning*. Berlin: Springer, pp. 362-376, 1996.

[Schank, 1982] Schank R.C., Dynamic Memory: A Theory of Reminding and Learning in Computers and People. Cambridge University *Press*, New York, NY, 1982.

[Schank & Leake, 1989] Schank R.C. et Leake D., Creativity and learning in a case-based explainer. *Artificial Intelligence* 40(1-3): 353-385, 1989.

[Senoussi & Chebel-Morello, 2008] Senoussi H. et Chebel-Morello M., A New Contextual Based Feature Selection. *IEEE World Congress on Computational Intelligence*, WCCI'08., Hong Kong : Chine, 2008.

[Shiu et al., 2001] Shiu S.C.K., Sun C.H., Wang X.Z. et Yeung C.S., Transfering case knowledge to adaptation knowledge, an approach for case-base maintenance. *Special issue of the journal Computational Intelligence*, 17(2): 295-314, 2001.

[Simon & Yeung, 2001] Simon C.K.S. et Yeung D.S., Transfering case knowledge to adaptation knowledge: an approach for case-base maintenance. *Computational Intelligent*, Volume 17, Number 2, 2001.

[Simoudis & Miller, 1990] Simoudis E. et Miller J., Validated retrieval in case-based reasoning. *In Proceedings of the Eighth National Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press, pp. 310-315, 1990.

[Smyth, 1996] Smyth B., Case Adaptation and Reuse in Déjà Vu. *Research report*, 1996.

[Smyth & Cunningham, 1996] The Utility Problem Analysed: A Case-Based Reasoning Perspective. In 3rd European Workshop on Case-Based Reasoning. Lausanne, Switzerland, 1996.

[Smyth & Keane, 1993] Smyth B. et Keane M.T., Retrieving adaptable cases: The role of adaptation knowledge in case retrieval. In *the European Workshop on Case-Based Reasonin (EWCBR'93)*, pages 209–220, Kaiserslautern, Germany, November 1993.

[Smyth & Keane, 1995a] Smyth B. et Keane M.T., Experiments on adaptation-guided retrieval in case-based design. In *Proceedings of the First International Conference on Case-Based Reasoning*. Berlin: Springer, pp. 313-324, 1995a.

[Smyth & Keane, 1995b] Smyth B. et Keane M.T., Remembering To Forget: A competence Preserving Deletion Policy for Case-Based Reasoning Systems. In *Proceeding of the 14th International Joint Conference on Artificial Intelligent (IJCAI 1995)*, Morgan-Kaufmann. pp. 377-382, 1995b.

[Smyth & Keane, 1998] Smyth B. et Keane M.T., Adaptation-guided retrieval: Questioning the similarity assumption in reasoning. *Artificial Intelligence* 102(2), 249-293, 1998.

[Smyth & McClave, 2001] Smyth B. et McClave P., Similarity versus diversity. In *the 4th International Conference on Case-Based Reasoning ICCBR'01*, volume 2080 de LNCS, pages 347–361, London, UK, 2001. Springer-Verlag.

[Smyth & McKenna, 1998] Smyth B. et McKenna E., Modelling the competence of case-bases. In Smyth, B. and Cunningham, P., editors, *Proceedings of the 4th European Workshop on Case-Based Reasoning (EWCBR'98)*, pages 208–220. Springer, 1998.

[Smyth & McKenna, 1999a] Smyth B. et McKenna E., Footprint based retrieval. *In Proceedings of the Third International Conference on Case-based Reasoning*. Berlin: Springer, pp 343-357, 1999a.

[Smyth & McKenna, 1999b] Smyth B. et McKenna E., Building Compact Competent Case-Bases. Case-based reasoning research and development. *Lecture notes in computer science*. 1650: 329-342. Seeon Monastery, 27-30 July, 1999.

[Smyth & McKenna, 2001] Smyth B. et McKenna E., Competence guided incremental footprint-based retrieval. *Knowledge-Based Systems* **14**(3-4), 155-161, 2001b.

[Smyth & McKenna, 2002a] Smyth B. et McKenna E., Competence guided incremental foot-print. *Journal of Knowledge-Based Systems*, In press, 2002a.

[Smyth & McKenna, 2002b] Smyth B. et McKenna E., Competence models and the maintenance problem. *Computational Intelligence: Special Issue on Maintaining Case-Based Reasoning Systems*, In Press, 2002b.

[Sørmo & Cassens, 2004] Sørmo F. et Cassens J., Explanation goals in case based reasoning. *In Proceedings of the ECCBR 2004 Workshops* (Technical Report 142-04). Universidad Complutense de Madrid, Departamento de Sistemas Informáticos y Programación, pp. 165-174, 2004.

[Sørmo et al., 2005] Sørmo F., Cassens J. et Aamodt A. Explanation in case based reasoning - perspectives and goals. *Artificial Intelligence Review* 24(2) 103-108, 2005.

[Stanfill & Waltz, 1986] Stanfill C. et Waltz D., Towards memory-based reasoning. *Communications of the Association for Computing Machinery*, 29:1213-1228, 1986.

[Sun et al., 2004] Sun Z., Finnie G. et Weber K., Case base building with similarity relations. *Information Sciences*, (165): 21–43, 2004.

[Tartakovski et al., 2004] Tartakovski A., Schaaf M., Maximini R. et Bergmann R., MINLP based retrieval of generalized cases. In *Proceedings of the Seventh European Conference on Case-Based Reasoning*. Berlin: Springer, pp. 404-418, 2004.

[Tomek, 1976] Tomek I., An experiment with the edited nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(6): 448–452, 1976.

[Toscano, 2005] Toscano R., Commande et diagnostic des systèmes dynamiques : modélisation, analyse, commande par PID et par retour d'état, diagnostic. *Ellipses* 2005.

[Toscano & Lyonnet, 2003] Toscano R. et Lyonnet P., Synthèse de la fonction de classification d'un système de diagnostic industriel. *JESA*, 37(3), pp. 311-332, 2003.

[Tversky, 1977] Tversky A., Features of similarity. *Psychological Review* 84(4), 327-352, 1977.

[Varma, 1999] Varma A., ICARUS: Design and Deployment of a Case-Based Reasoning System for Locomotive Diagnostics. In *3rd international conference on case-based reasoning (ICCBR-99)*, vol. 1650, pp. 581-595.

[Veloso, 1994] Veloso M.M., Planning and Learning by Analogical Reasoning. *Lecture Notes in Computer Science*, 886, Springer, Berlin, 1994.

[Veloso & Carbonell, 1994] Veloso M. et Carbonell J.G., Case-based reasoning in Prodigy. In Michalski, R. & Tecuci, G. (eds.) *Machine Learning: A Multistrategy Approach*, Volume IV. San Francisco, CA: Morgan Kaufmann, pp. 523–548, 1994.

[Veloso et al., 1995] Veloso M., Carbonell J., Pérez A., Borrajo, D., Fink, E., et Blythe, J., Integrating Planning and Learning: The PRODIGY Architecture. *Journal of Experimental and Theoretical Artificial Intelligence*, 7(1):81–120, 1995.

[Vong et al., 2002] Vong C.M., Leung T.P. et Wong P.K., Case-based reasoning and adaptation in hydraulic production machine design. *Engineering Applications of Artificial Intelligence*, (15): 567–585, 2002.

[Watson & Marir, 1994] Watson I. et Marir F., Case-Based Reasoning: A Review. *The Knowledge Engineering Review*, 1994.

[Weiss & Kulikowski, 1991] Weiss S.M. et Kulikowski C.A., Computer systems that learn. Artificial intelligence ISSN 0004-3702, *Morgan-Kaufmann*, 1991, 1993, vol. 62, no2, pp. 363-378, 1991.

[Wess et al., 1993] Wess S., Althoff K.D. et Derwand G., Using K-D trees to improve the retrieval step in case-based reasoning. *In Proceedings of the First European Workshop on Case-Based Reasoning*. Berlin: Springer, pp. 167-181, 1993.

[Wikipédia, 2009] Disponible sur: http://fr.wikipedia.org.

[Wilson, 2001] Wilson D.C., Case-base maintenance: the husbandandry of experience. *Thèse de doctorat*, Department of Computer Science Indiana University, Juillet 2001.

[Wilke & Bergmann, 1998] Wilke W. et Bergmann R., Techniques and Knowledge Used for Adaptation During Case-Based Problem Solving. Proc. of IEA-98-AIE, *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 1998.

[Wilson & Martinez, 1997] Wilson D.R. et Martinez A.R., Instance pruning techniques. In *Machine Learning: Proceedings of the 14th International Conference*, D. Fisher (Ed.). San Francisco, CA, 1997.

[Wolverton & Hayes-Roth, 1994] Wolverton M. et Hayes-Roth B., Retrieving semantically distant analogies with knowledge-directed spreading activation. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press, pp. 56-61, 1994.

[Yang & Wu, 2000] Keep it simple: A case-base maintenance policy based on clustering and information theory. In *13th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2000: advances in artificial intelligence, Proceedings Lecture Notes in Artificial Intelligence*, Montréal PQ, Springer-Verlag, Berlin, Vol.1822: 102-114, pp.14-17, 2000.

[Yang & Zhu, 2001] Yang Q. et Zhu J., A case addition policy for case-base maintenance. *Computational Intelligence Journal*, A Special Issue on Case-Base Maintenance. Blackwell Publishers, Boston MA UK, 17(2): pp.250-262, 2001.

[Yoshua & Chapados, 2003] Yoshua B. et Chapados N., Extensions to Metric-Based Model Selection. *Journal of Machine Learning Research* 3: 1209-1227, 2003.

[Zerhaoui & Bennani, 2004] Zerhaoui F. et Bennani Y., M-Som-art: Growing self organizing map for sequences clustering and classification. In *16th European Conference on Artificial Intelligence (ECAI 2004)*, Valancia, Spain, 2004.

[Zerhaoui et al., 2004] Zerhaoui F., Kanawati R. et Salotti S., CASEP2: Hybride case-based reasoning system for sequence processing. *European Conference on Case-Based Reasoning (ECCBR'2004)*. Madrid, Spain, August 2004.

[Zwingelstein, 1995] Zwingelstein G., Diagnostic des défaillances : Théorie et pratique pour les systèmes industriels, *Hermès*, 1995.

Résumé

Le développement des nouvelles technologies des différents produits et composants a rendu la nature des systèmes de plus en plus complexe. Cette complexité s'est répercutée sur le bon fonctionnement des équipements avec l'apparition de nouvelles pannes et l'accroissement des coûts engendrés. La maintenance est devenue un élément indispensable pour le maintien en condition opérationnelle de tout équipement quelque soit sa nature. Dans ce contexte nous nous intéressons à la maintenance corrective et plus particulièrement au diagnostic de pannes des équipements industriels. Nous développons une méthode basée sur le raisonnement à partir de cas (RàPC), méthode largement employée dans le domaine du diagnostic industriel. Le RàPC est une approche de résolution de problèmes et d'apprentissage. En diagnostic, une large variété de systèmes de RàPC a fait ses preuves, systèmes allant de problèmes de classification (systèmes orientés extraction « case-base mining ») aux systèmes à base de connaissance (systèmes orientés « connaissance »). Nous avons déployé dans le premier type de système, où la formalisation du cas est triviale, une méthode de maintenance du système. La maintenance de l'ensemble passe par la maintenance de la base de cas qui représente le cœur de ces systèmes de RàPC. Cette méthode de maintenance est composée d'une étape de structuration associée à une étape d'auto-incrémentation de la base de cas, afin de garantir la qualité du système tout au long de son évolution. Quant au deuxième type de système, nous avons mis en place un système fondé sur des modèles de connaissances associés aux différentes phases de manipulation du cycle de RàPC. Nous avons proposé une méthode de remémoration guidée par l'adaptation prenant appui sur deux mesures, une de similarité et une d'adaptation, et un algorithme d'adaptation spécifique au domaine du diagnostic industriel. Nos propositions ont été implémentées et validées sur une plateforme d'e-maintenance GaMA-Frame (Global asset MAintenance). Cette plateforme intègre notre module de diagnostic par RàPC ainsi que les différents modèles de connaissance liés à l'équipement à diagnostiquer SISTRE (Supervised Industrial System of pallets TRansfEr).

Mots-clés : diagnostic industriel, raisonnement à partir de cas, maintenance des systèmes de raisonnement à partir de cas, management des connaissances, modèles de connaissance, apprentissage incrémental, remémoration guidée par l'adaptation, adaptation.

Abstract

The development of new technologies of the various products and components made the nature of systems more and more complex. This complexity is reverberated on the good functioning of the equipment with the appearance of new failures and the increasing involved costs. Maintenance became an indispensable element to maintain any equipment in operational condition regardless of its nature. In this context we are interested in corrective maintenance and more particularly in fault diagnostic of the industrial equipment. We have developed a method based on case-based reasoning (CBR), this latter being widely used in the industrial diagnostic domain. The CBR is an approach for problem solving and learning. In diagnostic, several CBR systems, ranging from classification problems "case-base mining" to knowledge-based systems, have proven their effectiveness. We have deployed in the first type of system a method insuring its maintenance where the case formalization is trivial. The whole system's maintenance requires the maintenance of the case-base which represents the heart of CBR systems. This method of maintenance is composed of a structuring step and an auto-increment step of the case-base, both aim at ensuring the quality of the system throughout its evolution. As for the second type of system, we have implemented a system which relies on knowledge models associated with different handling phases of the CBR cycle. We have proposed an adaptation-guided retrieval method based on two measures, one of similarity and one of adaptation, and an adaptation algorithm dedicated to the industrial diagnostic field. Our proposals have been implemented and validated on a GaMA-Frame (Global asset MAintenance) e-maintenance platform. This platform integrates our CBR diagnostic module and different knowledge models to diagnose faults on SISTRE (Supervised Industrial System of pallets TRansfEr) equipment.

Keywords: diagnostic, case-based reasoning, maintenance of case-based reasoning systems, knowledge management, knowledge models, incremental learning, adaptation-guided retrieval.