Modélisation de documents et recherche de points communs - Proposition d'un framework de gestion de fiches d'anomalie pour faciliter les maintenances corrective et préventive

Grégory Claude

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)

Spécialité Informatique

Présentée et soutenue par :
Grégory Claude
Le mercredi 16 mai 2012

Titre :
Modélisation de documents et recherche de points communs
Proposition d'un framework de gestion de fiches d'anomalie pour faciliter les maintenances corrective et préventive

---

ED MITT : Image, Information, Hypermedia

Unité de recherche :
Institut de Recherche en Informatique de Toulouse (IRIT - UMR 5505)

Directeur(s) de Thèse :

Florence Sèdes
Marc Boyer (co-directeur)

Rapporteurs :

Dominique Rieu
Laurent Chaudron

Autre(s) membre(s) du jury :

Claude Chrisment (président)
Yannick Meiller (examinateur)
Christian Chaplais (invité Intercim)

Résumé

La pratique quotidienne d'une activité génère un ensemble de connaissances qui se traduisent par un savoir-faire, une maîtrise, une compétence qu'une personne acquiert au cours du temps. Pour les préserver, la capitalisation des connaissances est devenue une activité essentielle dans les entreprises. Elle permet d'éviter leur évaporation, notamment lorsque la personne quitte le poste, et d'éviter de les oublier, notamment lorsque la personne n'exécute plus l'activité pendant un certain temps. Pour cela, les connaissances doivent d'abord être explicitées et formalisées. Elles doivent ensuite être stockées et organisées au travers d'un système de gestion de connaissances qui les met à disposition d'autres personnes qui en ont l'utilité. Grâce à ce système, elles sont donc finalement réutilisées et exploitées par d'autres personnes qui peuvent elles-mêmes s'approprier les connaissances et les enrichir.

Nos travaux de recherche ont pour objectif de modéliser et mettre en œuvre un tel système afin d'extraire et de formaliser les connaissances issues des anomalies qui surviennent dans un contexte de production industrielle et de les intégrer dans un framework facilitant la maintenance corrective et préventive. Ce framework structure la connaissance sous la forme de groupes d'anomalies. D'une part, il aide dans la résolution d'une nouvelle anomalie en proposant automatiquement des solutions utilisées par le passé pour résoudre un problème similaire ; d'autre part, il permet de comprendre les raisons de l'apparition d'anomalies récurrentes et d'en empêcher l'apparition future en modifiant le processus de production industrielle au niveau conceptuel. Les groupes d'anomalies peuvent être rapprochés des *patterns* : ils représentent un problème auquel une ou plusieurs solutions sont associées. Ils ne sont pas définis a priori, c'est l'analyse des anomalies passées qui génère des groupes

pertinents, qui peuvent évoluer avec l'ajout de nouvelles anomalies (par exemple une nouvelle solution pour résoudre un problème).

Pour identifier ces patterns, supports de la connaissance, un processus complet d'extraction et de formalisation de la connaissance est suivi, *Knowledge Discovery in Databases*, bien connu dans le domaine de la gestion des connaissances. Ce processus a été appliqué dans des domaines aussi variés que l'astronomie, le marketing, la sécurité ferroviaire, etc. Nous lui donnons ici une nouvelle dimension, le traitement d'anomalies et plus particulièrement celles qui surviennent au cours de processus de production industrielle. Les étapes génériques qui le composent, depuis la simple sélection des données jusqu'à l'interprétation des *patterns* qui supportent les connaissances, sont considérées pour affecter à chacune un traitement spécifique pertinent par rapport à notre contexte applicatif. Les expérimentations sur différents jeux de données montrent que notre prototype fournit des résultats satisfaisants en termes de qualité des patterns obtenus à l'aide de différents algorithmes de classification et de performance en temps d'exécution de ces algorithmes.

Mots-clés

Gestion de connaissances, Processus de KDD,

Traitement d'information, Réutilisation d'information,

Modélisation, Classification de documents, Clustering,

Maintenance, Anomalie, Exception, Processus de production industrielle.

Abstract

The daily practice of an activity generates a set of knowledge that results in a know-how, a mastery, a skill a person gains over time. In order to take advantage of this experience, capitalization of knowledge has become an essential activity for companies. It prevents its evaporation, especially when the person leaves the job, and avoids forgetting it, especially when the person no longer carries out the activity for some time. Capitalization of this knowledge is realized in several steps. First, the knowledge must be clarified and formalized. Then, it must be stored and organized in a knowledge management system that makes it available to other people. With this system, the knowledge is finally reused by other people who may acquire this knowledge themselves and enrich it.

Our research work aims to model and implement such a system that extracts and formalizes knowledge from defects that occur in the context of industrial production, and to integrate it into a framework in order to facilitate corrective and preventive maintenance. This framework organizes the knowledge in the form of defects' groups. On the one hand, it helps to solve a new defect by automatically proposing solutions used in the past to solve a similar problem. On the other hand, it allows to understand the reasons why recurrent defects occur and to prevent their future appearance by modifying the industrial production process at a conceptual level. Defects groups can be compared to *patterns*: they represent a problem to which one or more solutions are related. They are not defined a priori; the analysis of past defects generates relevant groups, which may change with the addition of new defects (e.g. a new solution to solve a problem).

To identify these patterns, a complete process of knowledge extraction and formalization is adopted, *Knowledge Discovery in Databases*, well known in

the domain of knowledge management. This process has been applied in fields as diversified as astronomy, marketing, railway safety, etc. In this work, we give a new dimension to this process, the processing of defects, especially those that occur during industrial production processes. The generic steps that compose it, from the simple data selection to the interpretation of *patterns* that support knowledge, are considered. A specific processing, relevant to our applicative context, is assigned to each of these steps. Experimentations on different data sets show that our prototype provides satisfactory results in terms of quality of patterns obtained using different classification algorithms and in terms of performance in the execution time of these algorithms.

Keywords

Knowledge Management, KDD process

Information processing, Information reuse

Modeling, Documents classification, Clustering

Maintenance, Defect, Exception, Industrial production process

Remerciements

Je tiens tout d'abord à remercier Monsieur Marc Boyer, Maître de Conférences à l'Université Paul Sabatier et co-directeur de cette thèse, qui a permis qu'elle existe. Il n'a pas seulement trouvé le sujet, le laboratoire, le doctorant et l'entreprise, il s'est réellement battu pour que tous ces partenaires trouvent un accord alors même que cela semblait de plus en plus compromis. Avec Amélie Faure que je remercie également, à l'époque Directrice Administrative et Financière de l'entreprise, je sais qu'ils ont passé beaucoup de temps en discussions (y compris la nuit parfois à cause du décalage horaire) pour trouver une solution satisfaisante pour tout le monde. Il a su m'encadrer durant ces années en se rendant à chaque fois disponible pour répondre à toutes mes demandes et interrogations malgré ses nombreuses obligations. Le rôle joué dans la mise au point de ce travail et les conseils prodigués ont été importants et nécessaires, particulièrement dans les moments difficiles.

Merci à Madame Florence Sèdes, Professeur à l'Université Paul Sabatier et directrice de cette thèse, pour m'avoir permis de faire de cette thèse ce qu'elle est. Les réunions périodiques de travail ont toujours été riches en idées et ont été un moteur dans l'élaboration et l'évolution scientifiques de cette thèse. Elle a réussi à ne jamais quitter son rôle « d'élévateur au niveau scientifique » ce qui m'a permis de produire un travail de meilleure qualité que ce qu'il aurait été si elle m'avait laissé poursuivre mes propositions incomplètes, mal affinées ou trop orientées ingénieur.

Je remercie également Monsieur Claude Chrisment, Professeur à l'Université Paul Sabatier et responsable de l'équipe SIG (Systèmes d'Information Généralisés), pour m'avoir accueilli au sein de cette équipe de recherche. Merci à tous les membres de l'équipe, en particulier ceux de la composante D2S2 (Documents et Données Semi-Structurées et usage), Mesdames Marie-

Françoise Canut et Nadine Jessel, et Monsieur André Péninou, ainsi que les post-doctorants (qui naviguent vers d'autres horizons au moment où j'écris ces lignes) Mademoiselle Mihaela Brut et Monsieur Sébastien Laborie, pour l'attention portée à l'avancement de mes travaux de recherche.

Toute mon amitié va vers les doctorants, actuels ou anciens, de l'IRIT et en particulier Dana Al-Kukhun, Dana Codreanu, Anass El-Haddadi, Reda Jourani, Ana-Maria Manzat, Jonathan Petit et Dieudonné Tchuente. Nos échanges, nos discussions et votre soutien m'ont permis d'apprécier ses années en votre compagnie, de traverser les moments difficiles et ont surtout fait naître des liens d'amitié qui, je l'espère, ne seront pas rompus après nos départs respectifs de l'IRIT, lorsque nos chemins se sépareront pour aller voguer vers d'autres cieux.

Enfin, je tiens à remercier les membres de la société Intercim, société qui a participé au financement de cette thèse par le biais d'une bourse CIFRE et qui a fourni le domaine d'expérimentation. Malgré le peu de temps passé dans leurs locaux parisiens, je m'y suis toujours senti bien intégré grâce à la convivialité et la bonne ambiance de tous ses membres. Ces moments passés en entreprise ont été très enrichissants pour moi. Je remercie en particulier les membres de l'équipe de Recherche à Paris, Messieurs Christian Chaplais, Gaël Durand et Hadrien Szigeti ainsi que Messieurs Paul Meyer et Ed Pries aux Etats-Unis, pour les nombreuses discussions qui, elles aussi, ont permis de forger cette thèse. Je n'oublie pas non plus Mme Suy-Lang Huet, secrétaire multi-casquettes, à qui ma situation de doctorant qui plus est travaillant à distance a donné du travail supplémentaire mais qui a toujours su m'accueillir avec le sourire et m'aider dans la conduite des actions, notamment administratives, à mener.

Merci à ma compagne qui a su m'épauler et me soutenir tout au long de ces années, même dans les moments plus difficiles.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

Quel que soit le domaine considéré, le système d'information des entreprises produit une quantité importante de données dont la croissance est en perpétuelle accélération au cours des années. Ces données sont renseignées sous forme électronique, créées puis stockées, constituant ainsi la mémoire de l'entreprise. Les bases de données qui les conservent sont donc riches en *connaissances*, hébergeant des données sur l'historique qui peuvent s'avérer utiles pour les utilisateurs actuels du système d'information. Cependant, il est nécessaire d'appliquer des techniques pour analyser ces données et en extraire de la connaissance, de manière automatique ou semi-automatique. Ainsi, il est possible de mettre en évidence des modèles dans les données, des patterns qui structurent la connaissance, ce qui est nettement plus exploitable que les données brutes.

Contexte et problématique

De nombreux secteurs économiques se sont emparés de ces enjeux d'extraction et de gestion de la connaissance. (Fayyad et al., 1996a) présentent des applications de découverte de connaissances (KDD : Knowledge Discovery in Databases) issues de données dans de nombreux domaines très variés. En sciences, le système Skicat utilisé par les astronautes permet d'exécuter l'analyse d'images et de répertorier les objets dans l'espace (Fayyad et al., 1996b). Dans le monde de l'entreprise et des affaires, la détection de fraudes est effectuée grâce au système Prism pour les cartes de crédit et le système Fais pour le blanchiment d'argent (Senator et al., 1995). Le système Cassiopee est utilisé pour la fabrication du Boeing 737 afin de diagnostiquer et prédire les défauts et en déterminer des familles (Manago et al., 1996). D'autres applications du monde de l'entreprise et des affaires sont présentées dans le marketing pour identifier les comportements d'achat (Agrawal et al., 1996), les finances pour le management de portefeuilles (Hall et al., 1996) ou encore la télécommunication pour gérer les alarmes. Des applications dans d'autres domaines sont aussi développées, le système Advanced Scout pour interpréter les données de la National Basketball Association (NBA) (Bhandari et al., 1997) et les agents intelligents pour, par exemple, suggérer des morceaux de musique à l'utilisateur en fonction de son opinion sur différents morceaux musicaux. Plus récemment, nous pouvons relever le système Gesconda qui s'intéresse aux données environnementales

(Gibert et al., 2006) ou encore la prise de décision dans la gestion de la sécurité du trafic ferroviaire (Zhang et al., 2010).

Quelles que soient les données en entrée, les connaissances à produire et le secteur d'activités, transformer les données en entrée en connaissances utiles en sortie nécessite d'appliquer un processus complet de découverte de connaissances, le processus de KDD. Ses cinq étapes sont : (1) la sélection des données, (2) le pré-traitement des données, (3) la transformation des données, (4) la fouille de données et (5) l'interprétation et l'évaluation des patterns dans les données. La découverte de la connaissance ne dépend pas seulement de la fouille de données, de l'algorithme utilisé. La fouille de données n'est qu'une étape du processus, parmi celle de l'interprétation des résultats et celles de la préparation des données au préalable. Quel que soit le domaine considéré, ce processus est applicable et si le même algorithme de fouille de données peut être utilisé dans plusieurs applications, la réalisation des trois premières étapes, c'est-à-dire la préparation des données, ainsi que la cinquième étape qui concerne l'interprétation du résultat en sortie de l'étape de fouille de données sont spécifiques au domaine considéré.

Nous souhaitons considérer le traitement de déclaration d'événements inattendus. Ce cadre dans lequel nous nous plaçons peut s'appliquer à divers domaines. Dans un contexte de maintenance logicielle, la détection d'un bug dans une application déclenche un processus de résolution de ce bug dans lequel 4 étapes principales se succèdent : a) analyse et compréhension du logiciel, b) choix d'une solution, c) implémentation de la solution et d) clôture de l'intervention. Dans un contexte de vidéosurveillance, le comportement inhabituel d'une personne donne lieu à une action en réponse au problème détecté. Par exemple, si une personne passe dans le champ d'une caméra qui couvre un lieu interdit d'accès, une équipe de sécurité va se rendre sur place pour faire sortir cette personne ou pour l'arrêter. Ou encore, si la caméra de surveillance d'un distributeur automatique de billets détecte une agression, le distributeur peut refuser de délivrer de l'argent. Dans un contexte socio-technique ambiant, des exceptions par rapport au comportement de la personne peuvent être relevées. Par exemple, dans le cadre du maintien à domicile, la détection d'une personne âgée à terre depuis trop longtemps va déclencher la venue de personnels de santé, qui plus est si elle est dans une posture inconfortable.

Compte tenu de notre contexte d'étude, fourni par la société Intercim impliquée dans ce travail de recherche, nous allons particulièrement nous intéresser aux exceptions, que nous appellerons anomalies dans la suite de ce mémoire, qui surviennent lors de l'exécution de processus de production industrielle.

Nous souhaitons dégager les connaissances utiles sur les raisons qui ont conduit à l'apparition d'anomalies ainsi que sur les moyens mis en place pour les résoudre. L'analyse de ces connaissances a pour but de détecter et d'étudier

les anomalies pour permettre, par exemple, de prendre des décisions aussi optimales que possible en maintenance corrective et préventive : d'une part, connaître les solutions typiques appliquées à un problème spécifique pour faciliter la correction d'une nouvelle anomalie présentant ce problème précis, d'autre part, mettre en évidence les anomalies récurrentes pour prévenir leur apparition incessante. Appliqué à ce domaine fermé de gestion des anomalies relatives à la production industrielle, le processus de KDD doit permettre d'en capitaliser les informations, d'en extraire la connaissance.

Le processus de KDD est appliqué dans de nombreux domaines. Les cinq étapes qui le composent sont suffisamment génériques pour prendre en compte des données hétérogènes selon leur structure, leur valeur sémantique, le domaine dont elles sont issues, mais répondre à des besoins identiques, identifier des patterns dans les données et les interpréter. Dans ce travail de thèse, nous appliquons le processus de KDD à notre contexte de traitement d'anomalies en production industrielle. Nous voulons donc lui donner une nouvelle dimension, propre à la gestion d'anomalies. Pour cela, nous devons nous interroger sur la manière dont les étapes génériques qui énoncent les lignes directrices à suivre doivent être transformées en étapes spécifiques à la gestion d'anomalies qui définissent les traitements pertinents à appliquer. Que représente chacune des étapes dans notre contexte de maintenance en production industrielle, quels sont les types de traitement impliqués et comment réaliser la mise en œuvre ? Nous souhaitons donc mettre en place une architecture globale de gestion d'anomalies basée sur le processus de KDD qui permette d'extraire de la connaissance à partir des anomalies passées pour améliorer la correction d'anomalies présentes et éviter l'apparition d'anomalies futures.

Contributions

Les données stockées dans le document qui sert de support à une déclaration d'anomalie, document que nous appellerons fiche d'anomalie, peuvent être de différentes natures, aussi bien sur leur aspect structurel que sur la valeur sémantique de leur contenu. Ces deux propriétés qui caractérisent chaque donnée doivent être prises en compte au travers des deux premières étapes qui composent le processus de KDD, les étapes de sélection (1) et de pré-traitement (2) des données. Nous devons donc modéliser la fiche d'anomalie pour que les traitements appliqués lors de ces deux premières étapes soient pertinents par rapport à notre contexte de gestion d'anomalies. Ensuite, les étapes de transformation (3) et de fouille (4) de données doivent s'appuyer sur cette modélisation afin d'en utiliser le plein potentiel. Même si de nombreux algorithmes existent, l'étape de fouille de données doit offrir une approche

spécifique au traitement des fiches d'anomalie, pour mettre en évidence la connaissance, sous la forme de patterns, base de l'interprétation des résultats. Enfin, pour effectuer l'étape d'interprétation et d'évaluation des patterns (5) à partir des résultats issus de la fouille de données, il est nécessaire d'en proposer une représentation facilitant son interprétation par un expert du domaine. Des pistes de représentation graphique sont proposées ainsi que des indicateurs qui permettent d'évaluer la qualité des patterns obtenus.

Dans un premier temps, nous définissons un modèle de descripteur de fiche d'anomalie. Pour cela, nous nous sommes attachés à décrire et spécifier les éléments des fiches d'anomalie qui stockent les données. Ceux-ci définissent le caractère structuré ou non-structuré ainsi que la valeur sémantique de la donnée qui y est saisie. Afin de pouvoir considérer toutes les anomalies nominales (i.e. celles que l'on peut prévoir) mais aussi les anomalies non-nominales (i.e. celles auxquelles on n'a pas pensé et que l'on ne pouvait pas prévoir), issues de maintenances variées, dans des domaines aussi différents que celui du logiciel ou de l'industrie, il est nécessaire de proposer non pas un modèle de fiche d'anomalie mais bien un modèle de descripteur de fiche d'anomalie. Cette modélisation permet de soutenir les étapes de sélection (1) et de pré-traitement (2) des données. De la même manière que ces deux premières étapes ont pour rôle de mettre en évidence les critères de regroupement entre documents, cette modélisation permet la recherche de descripteurs liés en vue de regroupements de fiches d'anomalie similaires.

Dans un deuxième temps, nous proposons une approche exploitant le modèle de descripteur de fiche d'anomalie pour extraire des patterns dans les données. L'étape de transformation (3) des données est traduite par une structuration hiérarchique de la base de fiches d'anomalie. Cette structuration est conditionnée par nos objectifs de maintenance corrective et préventive. Dans l'étape de fouille (4) de données, nous ne voulons pas considérer les bases de fiches d'anomalie comme des bases documentaires classiques qui n'agrègent pas les réponses à une requête, qui se contentent de retourner l'ensemble des documents répondant à la requête, triés selon un critère de pertinence, de date, etc. Nous préférons regrouper les réponses similaires selon les critères de regroupement mis en évidence dans le modèle de descripteur de fiche d'anomalie, ce qui permet d'identifier des récurrences dans les données. La restitution des résultats en est optimisée, elle est plus claire pour l'utilisateur qui peut plus facilement s'approprier les réponses qui lui sont retournées (Soulé-Dupuy, 2001). Cette approche est particulièrement adaptée à notre contexte de maintenance dans lequel de nombreuses fiches d'anomalie similaires voire identiques sont créées et historisées. Bien que l'algorithme de classification utilisé soit commun, il est intégré dans une approche propre au traitement des fiches et permet donc une création pertinente de groupes de fiches. Nous utilisons une version incrémentale de l'algorithme pour adresser le problème du traitement de gros volumes de documents. Nous en proposons

une version améliorée pour répondre au biais de l'ordre de sélection des documents introduit par cette version incrémentale.

Enfin, pour proposer une représentation des groupes obtenus et valider leur qualité, des indicateurs a) intra-groupes, qui évaluent la qualité intrinsèque de chaque groupe, b) inter-groupes, qui évaluent l'intérêt de chaque groupe par rapport aux autres, et c) de gain, qui évaluent la pertinence gagnée entre l'absence de groupes et les groupes obtenus, ont été retenus. Ces indicateurs sont à la base d'algorithmes fournissant une représentation graphique des groupes. Les indicateurs et la représentation des groupes obtenus par l'étape de fouille de données permettent à un expert d'effectuer l'étape d'interprétation et d'évaluation des patterns (5). Un pattern est une généralisation d'un groupe de fiches d'anomalie, il regroupe les informations caractéristiques d'un groupe. Une manière de le représenter est de construire la fiche d'anomalie générique d'un groupe.

Organisation du mémoire

Ce mémoire de thèse présente en chapitre 1 le contexte de production industrielle et plus particulièrement la maintenance associée à ce contexte. Après que les termes anomalie et exception employés tout au long de ce mémoire aient été définis, les différents enjeux de cette maintenance sont exposés. Notre contexte étant lié à un domaine spécifique dont la maintenance n'est pas traitée dans la littérature, la maintenance dans d'autres domaines, le domaine industriel et le domaine logiciel, est présentée. Cela permet de dégager les principaux enjeux, notamment les enjeux communs, et ainsi de définir ceux de notre maintenance en particulier pour la situer par rapport à ces deux maintenances bien connues.

Le chapitre 2 s'intéresse à l'extraction et à la gestion de connaissances. Le processus de découverte de connaissances (processus de KDD) avec les cinq étapes qui le composent est détaillé. Pour supporter la connaissance, ce processus a pour objectif la création de patterns dans les données, qui servent à l'interprétation par des experts. En maintenance, les données manipulées sont issues des documents relatifs aux exceptions. Différents traitements d'exception ainsi que plusieurs patterns d'exception proposés dans la littérature sont étudiés pour déterminer la structure la plus adaptée à notre contexte de production industrielle. Le cadre applicatif de notre travail est aussi énoncé avec les caractéristiques des bases documentaires que nous voulons traiter et les différents traitements applicables sur des documents textuels.

Pour créer des patterns dans les données, l'étape de fouille de données du processus de KDD consiste en l'application d'un (ou plusieurs) algorithme(s) de classification de documents. Le chapitre 3 en dresse une liste des plus

utilisés dans la littérature aussi variée que le clustering, les cartes auto-organisatrices ou la classification par facette. Leurs points forts et leurs points faibles sont discutés pour permettre de déterminer plus tard celui qui répondra le mieux à nos objectifs. Les métriques d'évaluation de tels algorithmes sont développées.

Le chapitre 4 présente la modélisation de fiche d'anomalie que nous proposons. Cette modélisation s'appuie sur une étude empirique du processus de résolution d'anomalie et du document qui est manipulé au cours de ce processus, la fiche d'anomalie, notamment sur sa structure et sur les éléments qui le composent. Le modèle de descripteur de fiche d'anomalie élaboré se veut suffisamment générique pour représenter toute fiche d'anomalie de notre domaine de production industrielle. Des évaluations sur différentes bases de fiches d'anomalie, issues de production industrielle mais aussi issues du domaine logiciel, permettent de le valider.

Le chapitre 5 décrit le framework que nous proposons pour traiter les fiches d'anomalie. Il définit une structuration de la base de fiches d'anomalie pour les traiter selon deux dimensions. La dimension corrective a pour but la correction immédiate d'un problème. La dimension préventive a pour but d'éviter l'apparition de problèmes récurrents. Ce framework permet aussi l'intégration des nouvelles fiches d'anomalie au fur et à mesure de leur arrivée. Plusieurs méthodes de l'algorithme de classification retenu sont exposées. Un enrichissement de la méthode incrémentale est proposé pour diminuer l'importance du biais de l'ordre de sélection des documents. De plus, un ensemble d'indicateurs a été sélectionné parmi la grande variété existante dans la littérature pour évaluer la qualité des groupes obtenus par l'algorithme de classification.

Le chapitre 6 est dédié à l'implémentation et aux expérimentations. Les choix d'implémentation sont discutés notamment en ce qui concerne l'utilisation des techniques bien connues en Recherche d'Information pour manipuler les éléments de fiche d'anomalie qui stockent des données plein texte. Le modèle implémenté composé du modèle de descripteur de fiche d'anomalie (chapitre 4) et des éléments nécessaires à l'utilisation du framework (chapitre 5) est présenté et les différents modules du prototype logiciel sont exposés. Le framework basé sur le modèle de descripteur de fiche d'anomalie est testé sur plusieurs jeux de données réelles issus du monde industriel sur des critères de qualité des résultats obtenus et de performance des méthodes de l'algorithme utilisé.

Le dernier chapitre fait le bilan de ce travail et présente quelques perspectives.

Chapitre 1

Maintenance et exceptions en production industrielle

1.1. Introduction

Ce chapitre introduit le contexte particulier de maintenance dans lequel le domaine applicatif de notre travail se situe : le traitement d'anomalies qui surviennent au cours de processus de production industrielle. La maintenance est un terme qui est utilisé dans de nombreux domaines, elle est définie par la Fédération Européenne des Sociétés Nationales de Maintenance (EFNMS, European Federation of National Maintenance Societies) comme « *toutes les actions qui ont pour objectif de garder ou de remettre une chose en état de remplir la fonction qu'on exige d'elle ; ces actions regroupent toutes les actions techniques et toutes les actions d'administration, de direction et de supervision correspondantes* ».

Afin d'exposer le contexte de production industrielle dans lequel nous nous situons et de donner notre vision de la maintenance et de ses enjeux dans ce contexte, nous définissons d'abord le terme *anomalie* que nous employons tout au long de ce mémoire lorsque nous parlons de maintenance dans notre contexte, notamment par rapport au terme *exception* plus communément répandu dans la littérature relative à la maintenance en général. Puis, nous présentons deux maintenances bien connues, les maintenances industrielle et logicielle, au travers de leurs principaux enjeux. Bien que l'objet sur lequel porte chacune de ces deux maintenances soit différent, plusieurs enjeux communs sont dégagés. Cela nous permet ensuite de reprendre ces enjeux pour présenter la maintenance dans notre contexte de production industrielle et de la situer par rapport aux maintenances industrielle et logicielle.

1.2. L'anomalie, une exception particulière

Quel que soit le contexte, dans la littérature, les auteurs définissent une exception de la même manière. Dans le contexte logiciel, une exception est une occurrence qui dévie du processus normal idéal nécessitant un changement dans l'objectif principal de la tâche courante (Somekh et al., 2007). Dans le contexte des transactions sécurisées, une exception représente tout élément qui

provoque l'arrêt du comportement d'un processus normal (Wang et al., 2004). Dans le contexte de processus métier, une exception représente tout type d'erreur ou d'événement inhabituel qui intervient à un endroit particulier, c'est-à-dire, à une activité spécifique du processus (Curbera et al., 2003), elle concerne tous les écarts par rapport à la définition du processus original qui peuvent causer un dysfonctionnement (Yaxiong et al., 2010). Toujours dans le contexte de processus métier, (Eder et al., 1996) et (Edelweiss et al., 1998) précisent encore mieux ce qu'est une exception en faisant la distinction entre exception et panne, deux types de problèmes qui peuvent survenir lors de l'exécution d'un processus. Une exception est causée par un incident au niveau du système ou par une nouvelle situation introduite par l'environnement externe. On peut prévoir comment la traiter même s'il est difficile d'identifier tous les types d'exceptions qui peuvent survenir. Une panne est due à l'équipement, à la communication entre appareils ou aux erreurs dans les programmes. On ne peut pas les prévoir et leur traitement ne peut pas être modélisé.

Une exception est donc une occurrence d'un événement inattendu qui symbolise un écart entre ce qui devrait être réalisé et ce qui est réellement réalisé. Il est possible de la prévoir et donc de concevoir le type de traitement à y appliquer pour la résoudre. Le terme anomalie que nous employons par la suite dans ce mémoire correspond bien à celui d'exception. La différence réside dans l'aspect général du terme exception. Celui-ci est employé dans de nombreux domaines mais des synonymes propres à chaque domaine peuvent le remplacer. Par exemple, dans le contexte logiciel, on peut utiliser le terme bug. Ici, nous situant dans le contexte de production industrielle que nous définissons dans la suite de ce chapitre, nous préférons le terme anomalie au terme général d'exception. Une anomalie est donc une exception qui survient au cours d'un processus de production industrielle. Ensuite, comme le précisent (Yaxiong et al., 2010), une fois détectée, ce processus est suspendu jusqu'à ce que la cause de l'exception ait été trouvée et que les mesures appropriées pour la résoudre aient été prises.

1.3. Enjeux de la maintenance

1.3.1. La maintenance industrielle

Plusieurs définitions de la maintenance industrielle sont présentes dans la littérature, certaines étant plus ou moins complètes. Nous allons considérer la maintenance industrielle comme (Mechin, 2007) la définit, à savoir « *l'ensemble de toutes les actions techniques, administratives et de management durant le cycle de vie d'un bien, destinées à le maintenir ou à le*

rétablir dans un état dans lequel il peut accomplir la fonction requise ». Un bien étant « *tout élément, composant, mécanisme, sous-système, unité fonctionnelle, équipement ou système qui peut être considéré individuellement* ».

Pour répondre à cette définition, de nombreux auteurs présentent différentes catégories de maintenance industrielle. (Komonen, 2002) utilise une division des activités de maintenance en deux catégories, la maintenance d'après-casse et la maintenance planifiée. (Luxhoj et al., 1997) et (Dowlatshahi, 2008) définissent les maintenances corrective, préventive, prédictive, à base de condition et intelligente. (Alsyouf, 2007) présente les maintenances réactive, préventive, prédictive et holistique. Nous n'allons nous intéresser qu'à seulement deux catégories, ce sont les maintenances traditionnelles rappelées par (Komonen, 2002), chacune d'elles pouvant regrouper plusieurs catégories présentées dans ce paragraphe (Mechin, 2007) :

- La *maintenance préventive* (ou proactive) qui est exécutée à des intervalles prédéterminés ou selon des critères prescrits et destinée à réduire la probabilité de défaillance ou la dégradation du fonctionnement d'un bien. Celle-ci peut être programmée, systématique, conditionnelle ou prévisionnelle.
- La *maintenance corrective* (ou réactive) qui est exécutée après la détection d'une panne et destinée à remettre un bien dans un état dans lequel il peut accomplir la fonction requise. Celle-ci peut-être palliative ou curative.

La maintenance corrective seule ne peut pas être une bonne solution. C'est la pire des politiques étudiées par (Savsar, 2004) et les coûts sont prohibitifs (El Aoufir et al., 2007) : « *se suffire à pratiquer de la maintenance corrective avec un minimum de ressources se traduira certes par un faible budget de maintenance mais entraînera à coup sûr des coûts d'indisponibilité et de dysfonctionnements exorbitants* ». Cependant elle reste indispensable et doit être vue comme une maintenance complémentaire de la maintenance préventive et non pas comme un échec (Despujols, 2004).

La maintenance préventive réduit les coûts par rapport à la maintenance corrective. Il est plus coûteux de réparer une machine en panne que d'effectuer un entretien régulier visant à éviter une panne (Jin et al., 2009). Elle doit aussi permettre de gagner du temps sur le long terme. Pour évaluer le coût de la maintenance, différents critères peuvent être retenus tels que le temps moyen entre les défaillances, le temps moyen de réparation et le temps d'attente moyen (Komonen, 2002).

Pour effectuer ces deux maintenances, il est essentiel que les documents qui servent de support aux exceptions soient aussi complets et précis que possible. Pour cela, ce travail doit être intégré dans le processus global de l'entreprise (Luxhoj et al., 1997). Il doit être valorisé et suffisamment de temps doit être

accordé à l'opérateur pour sa rédaction. L'opérateur doit connaître la finalité de cette opération, à savoir que chaque document ne sert pas uniquement à la mise en place de la traçabilité mais que son analyse permettra de déterminer si elle appartient à un groupe connu dont on peut réutiliser la ou les solution(s) et plus tard, de prévenir ce type d'exception grâce aux opérations de maintenance préventive. Ces documents étant à la base de la maintenance, ils ont un impact important sur sa qualité. Mais ce n'est pas le seul facteur qui influence la qualité. Le manque de documentation et de structuration de la connaissance ainsi que le manque de ressources dans les équipes de maintenance contribuent à un appauvrissement de la qualité de la maintenance (Luxhoj et al., 1997). Il faut donc casser le mur, appelé « barrière » par (Levner et al., 1998), entre la maintenance industrielle et la production afin qu'elles soient toujours en étroite collaboration, les opérations de maintenance devant être intégrées aux plannings des opérations de production.

Une bonne maintenance industrielle améliore la productivité, la rentabilité, la qualité du produit et réduit les coûts (Alsyouf, 2007). La qualité de la maintenance est donc un facteur déterminant de ces différents éléments. Mais une bonne maintenance industrielle ne suffit pas pour produire une pièce parfaite. Les défaillances de production conduisant à une action de maintenance peuvent avoir différentes origines : les éléments de l'équipement, la qualité du matériel et des pièces de rechange, la conception, le contrôle du processus de fabrication et l'erreur humaine (Alsyouf, 2009). Ces éléments interviennent dans le système productif décrit par Mechin (Mechin, 2007) et dont nous proposons une représentation en Figure 1.

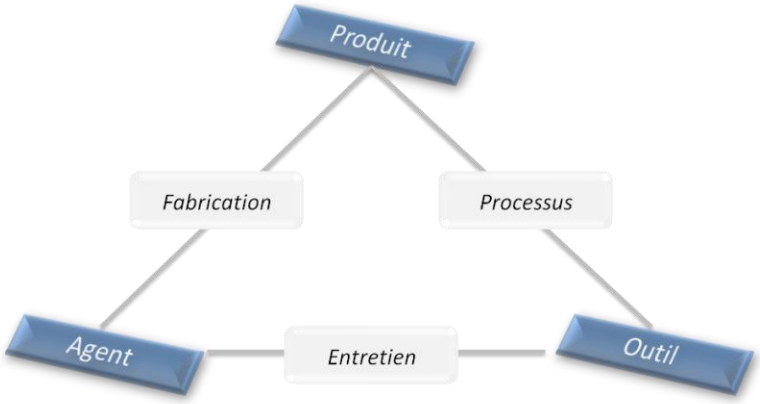


Figure 1 – Le système productif (Dans un souci d'harmonisation de notre discours, nous avons remplacé le terme « process » utilisé par (Mechin, 2007) par le terme « processus »)

Le processus assure la liaison entre l'outil et le produit. Le produit est fabriqué par l'outil en suivant le processus défini lorsque toutes les données en entrée nécessaires sont réunies. La fabrication assure la liaison entre l'agent et le produit. L'agent effectue l'action de fabrication du produit, et ce, même si

celle-ci est automatisée et qu'il ne fait que piloter au travers d'actions de surveillance, de réglages, d'alimentation de l'installation en matières diverses. L'entretien assure la liaison entre l'agent et l'outil. C'est l'une des composantes de la maintenance industrielle.Le besoin de formalisation du processus de maintenance est un problème souvent évoqué. En effet, pour être efficace, le processus de maintenance doit être clairement défini. Le concept de la maintenance doit être défini par l'entreprise avec les stratégies à employer en fonction des différentes situations qui se présentent à la maintenance. Celui-ci est propre à chaque entreprise car chacune possède des besoins spécifiques même si la structure sous-jacente pour atteindre ce concept est analogue pour toutes les entreprises. « *Une structure maintenance bien cadrée, définie, formalisée, est absolument essentielle à la bonne exécution des différentes composantes de la fonction maintenance à ses différents niveaux de gestion* » (El Aoufir et al., 2007). Les auteurs expriment le besoin d'un organigramme de la maintenance (responsabilités) et des procédures et processus de la maintenance (activités). Ce sont ces éléments qui doivent composer ce qu'ils appellent une structure maintenance. Ils présentent différents processus de maintenance et y attachent des documents permettant d'effectuer une bonne maintenance. De nombreux auteurs présentent des cadres pour modéliser et formaliser tout processus de maintenance. (Waeyenbergh, 2004) et (Waeyenbergh, 2004) proposent un framework, CIBOCOF, qui apporte une structure à la gestion des problèmes de maintenance. Une approche fonctionnelle est présentée par (Despujols, 2004). Différentes fonctions de la maintenance sont exposées : études, préparation des travaux, ordonnancement, réalisation, gestion des moyens du service maintenance, approvisionnement des matières et des prestations extérieures. La démarche TPM (Total Productive Maintenance) est aussi une solution souvent évoquée (Ahuja et al., 2008). Elle fait participer toutes les personnes à différents niveaux de maintenance dont le but est l'optimisation de l'utilisation de chaque outil de fabrication. Cela doit se faire par pallier successifs, l'important n'étant pas tant l'indicateur phare, le TRS (Taux de Rendement Synthétique) mais plutôt l'évolution de celui-ci (Gatti, 2003). Le lien de fabrication entre le produit et l'agent concerne un tout autre domaine. Pour améliorer la qualité du produit, il faut éviter les erreurs humaines, dont l'agent est responsable.

1.3.2. La maintenance logicielle

Selon (IEEE, 2005), la maintenance logicielle est « *le processus de soutien de la production de logiciels pour détecter et corriger les défauts, optimiser la performance et assurer la validité appropriée aux utilisateurs finaux* ».

Avant d'exposer notre étude de la maintenance logicielle, il nous paraît utile d'exprimer notre positionnement par rapport à l'évolution logicielle par rapport

à celle-ci. Ces deux activités du génie logiciel modifient un logiciel et c'est l'écart estimé (dont la mesure n'est pas forcément évidente) entre les deux versions du logiciel qui détermine si l'on doit parler de maintenance logicielle ou d'évolution logicielle. Certains auteurs ne vont traiter que de la maintenance logicielle (Giovannini, 1989), (Lambolez, 1994), (Barros, 1997), (Harjani et al., 1992), (Haziza et al., 1992) ou que de l'évolution logicielle (Lehman et al., 1997), (Mens et al., 2005), (Sneed et al., 2008) parce qu'ils ne veulent considérer qu'une partie spécifique propre à la maintenance logicielle ou à l'évolution logicielle. Nous allons au contraire prendre le parti de traiter ces deux sujets sans distinction comme l'ont fait (Bennett et al., 2000), (Bohner, 2002), (Fleurquin et al., 2005), (Alloui, 2009). Nous considérons que l'objectif de ces deux activités est de modifier un logiciel pour lui permettre de correspondre aux besoins des utilisateurs, comme le rappelle la première loi de Lehman (Lehman, 1980) selon laquelle un logiciel doit être modifié continuellement sans quoi il deviendra progressivement moins utile.

De la même manière que nous avons relevé des catégories de maintenance industrielle, nous pouvons présenter différentes catégories de maintenance logicielle. Déjà dans le milieu des années 1970, (Swanson, 1976) en expose trois catégories qui sont toujours valables aujourd'hui :

1. La *maintenance corrective* qui est l'ensemble des activités ne modifiant pas les spécifications initiales et mettant le logiciel en conformité avec celles-ci
2. La *maintenance adaptative* qui est l'ensemble des activités traitant des problèmes de changement d'environnement
3. La *maintenance perfective* qui est l'ensemble des activités visant à intégrer ou améliorer de nouvelles spécifications non fonctionnelles.

A ces trois catégories se sont ajoutées au cours des années :

4. La *maintenance évolutive* qui est l'ensemble des activités visant à intégrer ou améliorer de nouvelles spécifications fonctionnelles
5. La *maintenance préventive* dont le rôle est d'améliorer la qualité du logiciel, la maintenabilité du logiciel pour faciliter les actions de maintenance future.

De cette liste de catégories de maintenance logicielle, nous pouvons noter deux points. D'abord, la maintenance corrective, qu'elle soit industrielle ou logicielle, a toujours pour objectif de corriger une exception rencontrée. Ensuite, les autres catégories ne semblent pas trouver de correspondance. Alors que la maintenance industrielle préventive doit anticiper les futures exceptions pour éviter qu'elles ne se produisent à nouveau, on ne retrouve pas cet aspect prévisionnel dans les catégories de maintenance logicielle. En effet, ces catégories sont différentes selon la modification que l'on veut apporter mais aucune, y compris la maintenance logicielle préventive, n'a pour objectif la

prévention d'exceptions (ou bugs logiciels). On considère que lors du développement du logiciel, les différents tests (boîte noire, boîte blanche, de non-régression…) ont permis de corriger les éléments ne répondant pas aux spécifications attendues. Malgré cette différence qui, pour notre travail, revêt un caractère assez important, la maintenance industrielle et la maintenance logicielle possèdent des enjeux communs.

Le problème du mur, présenté en maintenance industrielle, est un écueil souvent rencontré par la maintenance logicielle. Encore une fois, il correspond au manque d'information lors du passage du logiciel, du développement à la maintenance. Ce problème « *[d'] acquisition, [de] transfert et [de] gestion de la connaissance* » (Haziza et al., 1992) empêche le mainteneur de travailler de manière efficace. L'idéal est qu'une personne de l'équipe de développement soit affectée à la maintenance et que la documentation ait été produite et soit à jour ; ce qui n'est pas toujours le cas car, en fin de projet, on favorise l'écriture du code au détriment de la rédaction de la documentation. Dans certains cas, comme celui de l'aérospatial, cela n'est pas suffisant car le volume documentaire produit est très important. Des outils de Recherche d'Information doivent être développés pour donner au mainteneur les documents ou éléments de documents utiles à la résolution de l'exception logicielle (Lambolez, 1994).

Le besoin de formalisation du processus de maintenance est aussi un problème récurrent. Alors que lors du développement d'un logiciel, un processus bien défini est suivi, par exemple le cycle en V ou le cycle en spirale, la maintenance logicielle est souvent confrontée à une absence de processus ou utilise un processus bien plus pauvre que celui du développement (Haziza et al., 1992). L'ensemble des travaux sur ce problème converge vers un même modèle de processus (Harjani et al., 1992), (Lambolez, 1994), (Barros, 1997), (Alloui, 2009), (Haziza et al., 1992) :

1. Analyse et compréhension du logiciel
    a. Enregistrement du bug
    b. Recherche de problèmes similaires dans l'historique
    c. Localisation du problème (erreur, partie du système à modifier)
    d. Recherche de solutions
2. Décision : choix d'une solution ou rejet de l'intervention (passage à l'étape 4)
3. Implémentation de la solution (si décision de solution)
    a. Mise en œuvre
    b. Validation
4. Clôture de l'intervention, archivage

Un parallèle intéressant peut être fait entre l'évolution d'un logiciel et l'évolution biologique (Godfrey et al, 2008) illustrant bien cette vision :

- Le code source est représenté par le génotype, le génotype étant l'ensemble des gènes dont l'expression est indépendante de l'environnement
  exemple : *la couleur des yeux*
- Le programme exécutable, la plateforme technique et l'environnement de l'utilisateur sont représentés par le phénotype, le phénotype étant l'ensemble des gènes dont l'expression dépend ou peut dépendre de l'interaction avec l'environnement
  exemple : *la taille*.

Nous pouvons donc déterminer deux types d'exception, celles qui surviennent obligatoirement quel que soit l'environnement (lieu de fabrication, opérateur, date…) qui sont à rapprocher du génotype. Appelons-les exceptions de conception. Et celles qui surviennent parce que l'environnement l'a provoqué (erreur humaine…) qui, elles, sont à rapprocher du phénotype. Appelons-les exceptions environnementales. Evidemment, certaines conditions environnementales irréalisables ou la confrontation de plusieurs conditions environnementales sont incompatibles ce qui, bien que dépendant de l'environnement, serait plutôt à classer dans les exceptions de conception.

Quel que soit son type, lorsqu'une exception survient, on parle de maintenance corrective pour la résoudre. Pour faciliter ce travail, un aspect réutilisation est considéré. A la manière de la maintenance logicielle et plus largement du génie logiciel, il est plus fiable et plus efficace de réutiliser des modèles et des composants déjà développés et testés par le passé. La problématique étant de les retrouver parmi les librairies existantes.

De plus, la partition des exceptions décrite ci-dessus peut donner lieu à une maintenance préventive. Considérons les exceptions de conception. Si l'on peut mettre en évidence la raison de leur apparition, il peut être intéressant et même utile que ces informations soient utilisées pour corriger le problème en amont, c'est-à-dire en modifiant la conception, et ainsi ne plus avoir de maintenance corrective à effectuer sur ce genre d'exception en évitant leur apparition.

Evidemment, une modification apportée au niveau de la conception peut avoir des répercussions par la suite. Cette analyse d'impact est un élément essentiel dans la maintenance logicielle. En effet, contrairement au développeur, le mainteneur doit considérer le fait que la modification qu'il apporte sur une partie du logiciel, aussi petite soit elle, peut avoir des impacts sur l'ensemble du logiciel, sur les fonctionnalités (le code source) mais aussi sur les autres éléments du logiciel tels que les exigences, la conception ou les jeux de tests (De Lucia et al., 2008). Ainsi, des études proposent une analyse de cet impact (Barros, 1997), (Bohner, 2002) avec des solutions telles que les techniques de

program slicing (Zhao, 1998) (Gallagher et al., 1991). C'est une « technique de décomposition des éléments d'un programme liés à un calcul particulier. Cela consiste à découper les parties d'un programme qui peuvent affecter directement ou indirectement les valeurs calculées à un certain point d'intérêt du programme » (Zhao, 1998). L'analyse d'impact est d'autant plus importante que lorsque la modification est acceptée, elle est ensuite intégrée à la nouvelle version du logiciel. Un impact non contrôlé aura donc une influence sur toute utilisation future du logiciel. A ces impacts internes à la maintenance, nous pouvons aussi ajouter les impacts des problèmes qui sont survenus durant le développement du logiciel (qualité de la documentation, de la programmation, …) et qui affectent nécessairement le travail du mainteneur (Chent et al., 2009).

1.4. La maintenance dans notre contexte de production industrielle

1.4.1. Le contexte

Pour assurer la qualité d'un produit final, le traitement et la traçabilité des exceptions qui surviennent durant son processus de production industrielle est devenue une activité indispensable. En effet, la gestion des informations relatives aux exceptions peut représenter jusqu'à 80% du volume de l'information concernant le produit final. L'exploitation de cette masse d'informations apporte donc une réelle plus-value permettant, par exemple, de comprendre les raisons de mise au rebut, de diminuer cette mise au rebut et d'anticiper les problèmes de fabrication. Un parallèle peut être établi avec les exceptions logicielles et les nombreux systèmes permettant le suivi des activités de gestion des bugs. Nous pouvons citer IBM Rational Clearquest (http://www-01.ibm.com/software/awdtools/clearquest/) ou Atlassian Jira (http://www.atlassian.com/software/jira/), et d'autres gratuits comme Mantis Bug Tracker (http://www.mantisbt.org/) ou BugZilla (http://www.mozilla-europe.org/fr/products/bugzilla/). Ces logiciels proposent de nombreuses fonctionnalités permettant notamment la création et le suivi de bugs en temps réel, la production de rapports, la configuration et la personnalisation de l'environnement. Certains offrent d'autres fonctionnalités telles que des outils de planification de projet.

Nous pouvons cependant nous interroger sur la capitalisation de toute l'information relative aux exceptions traitées avec de tels outils, qu'elles soient logicielles ou non. En effet, ceux-ci sont utilisés pour le suivi des exceptions permettant principalement de les déclarer, de suivre le processus conduisant à leur résolution et de les clore. Les rapports générés sont essentiellement statistiques (par exemple, le nombre d'exceptions créées ou résolues durant une

période, les exceptions en cours triées selon leur criticité, etc.) et sont présentés sous forme graphique (e.g., histogrammes, courbes, camemberts, graphiques à jauges, etc.).

Dans les activités de maintenance, nous pouvons relever plusieurs intérêts à l'exploitation des informations relatives aux exceptions (Goh et al., 2009), particulièrement dans les industries aéronautique, spatiale et pharmaceutique où notamment la main d'œuvre très présente provoque un taux élevé d'exceptions durant le processus de production. Un premier intérêt, à court terme et en temps réel, est de faciliter les activités de maintenance corrective en assistant le mainteneur dans sa tâche de recherche de solution pour résoudre un problème. Un second intérêt, à long terme et en offline, est de prévenir l'apparition d'exceptions récurrentes en mettant en évidence les raisons de leur apparition.

Lorsqu'une entreprise définit le processus de production industrielle d'une pièce, elle met au point le processus standard de production. Cependant, elle doit aussi prendre en compte les événements inattendus, les anomalies, qui surviennent et qui correspondent à un écart par rapport à ce processus standard de production. Pour chaque anomalie, un processus de résolution doit être mis en place pour revenir dans une situation standard d'exécution du processus de production (Claude, 2009). Au cours de son exécution, nous pouvons relever plusieurs types d'anomalie :

- Celles qui font état d'une pièce présentant un dommage mineur et qui ne nécessitent pas de traitement supplémentaire (par exemple, un petit morceau de peinture qui se décolle en faisant un trou, mais qui sera caché par l'assemblage d'une autre pièce dessus)
- Celles qui font état d'une pièce présentant un dommage sérieux et qui nécessitent un traitement supplémentaire pour que cette pièce redevienne conforme (par exemple, le perçage d'un trou trop petit par rapport à ce qui était prévu)
- Celles qui font état d'une pièce présentant un dommage irréversible et qui ne peuvent donc pas être corrigées.

Toutes ces anomalies portent sur l'exécution du processus de production industrielle. Leur gestion relève donc d'une maintenance que l'on peut qualifier de maintenance de processus de production. Celle-ci se différencie de la maintenance industrielle qui porte sur l'outil de fabrication et de la maintenance logicielle qui porte sur le bien produit, c'est-à-dire le logiciel.

Cependant, nous pouvons noter des activités similaires telles que la correction et la prévention d'erreurs, l'analyse d'impact, et des problèmes rencontrés semblables tels que le besoin de formalisation du processus de maintenance ou la qualité de la maintenance. L'étude de ces différentes maintenances nous permet de confronter les activités et les problèmes des maintenances

industrielle et logicielle avec la maintenance de processus de production qui nous concerne particulièrement dans cette étude.

1.4.2. Les enjeux de la maintenance de processus de production

Pour résoudre et éviter l'apparition d'exceptions, différents types de maintenance sont appliqués selon le contexte. Lorsque l'on travaille sur la fabrication de pièces et que l'objet de la maintenance est l'outil de fabrication, on parle de maintenance industrielle. Lorsque l'on travaille sur la production de logiciels, on parle de maintenance logicielle pour résoudre les bugs qui surviennent.

Bien qu'elle se situe dans un contexte industriel, la gestion des anomalies que nous considérons ne s'inscrit pas dans la maintenance industrielle. Nous ne travaillons donc pas sur l'outil de fabrication – objet de la maintenance industrielle – ou sur le produit logiciel – objet de la maintenance logicielle – mais sur le processus de production industriel de pièces. C'est pourquoi, cette maintenance, que nous qualifions de processus de production, se différencie des autres types de maintenance.

La section précédente a présenté les enjeux liés aux maintenances industrielle et logicielle. Certains de ces enjeux sont communs aux deux maintenances, définitions et catégories similaires, besoin de qualité et de formalisation du processus de maintenance. D'autres sont spécifiques à l'une d'elles, la productivité pour la maintenance industrielle, la réutilisation et l'analyse d'impact pour la maintenance logicielle (Table 1).

	Maintenance industrielle	***Maintenance logicielle***
Objet	Outil de fabrication	Produit logiciel
Enjeux communs	Définition de la maintenance Catégories de la maintenance Qualité, retour d'expérience Formalisation du processus de maintenance	
Enjeux spécifiques	-	Réutilisation Analyse d'impact

Table 1 – Enjeux des maintenances industrielle et logicielle

Les définitions des maintenances industrielle et logicielle sont très proches. En les adaptant, nous définissons la maintenance de processus de production comme *le processus de soutien de la production industrielle pour détecter et*

corriger les anomalies d'exécution du processus de production, optimiser la performance et assurer la validité appropriée aux utilisateurs finaux.

De cette définition, nous pouvons dégager deux dimensions : une première dimension corrective (« corriger les anomalies d'exécution du processus de production ») et une deuxième dimension préventive (« détecter […] les anomalies d'exécution du processus de production, optimiser la performance »). Ces deux dimensions rappellent les deux catégories principales de maintenance industrielle et logicielle.

D'une part, le rôle de la maintenance corrective est de résoudre une nouvelle exception pour que l'élément sur lequel cette anomalie porte redevienne conforme aux spécifications. La maintenance industrielle corrige l'outil de production, la maintenance logicielle corrige le logiciel produit et la maintenance de processus de production corrige le processus de production industrielle en cours d'exécution.

D'autre part, la maintenance préventive a pour but d'anticiper et de prévenir de l'émergence d'exceptions connues, donc de gérer les exceptions récurrentes. Un retour d'expérience depuis la maintenance vers la conception est indispensable pour la mise en place de cette catégorie de maintenance. C'est à partir de l'étude des exceptions passées et des opérations de maintenance effectuées qu'une maintenance industrielle préventive efficace peut être mise en place (Chitra, 2003), (Despujols, 2004). Des méthodes partant de ce constat ont été proposées (Gharbi et al., 2000), (Song, 2009). Concernant la maintenance de processus de production, l'étude des anomalies passées doit permettre la modification du processus de production industrielle au niveau de sa conception, et non pas au niveau de son exécution comme en maintenance corrective.

De la même manière que pour les maintenances industrielle et logicielle, en maintenance de processus de production, ne pratiquer qu'une maintenance corrective coûte cher et ne permet pas de mobiliser toute la connaissance capitalisée. Mettre en place une maintenance préventive permet de réduire le temps de fabrication puisque chaque anomalie créée doit être traitée. Il est alors tout à fait possible de se retrouver dans une situation dans laquelle la gestion des anomalies survenues au cours de l'exécution du processus de production prend plus de temps que l'exécution même de ce processus. Cependant, bien qu'étant sensible aux coûts, la maintenance de processus de production s'intéresse principalement à la qualité du produit final contrairement à la maintenance industrielle dont l'objectif majeur est la réduction des coûts.

Cependant, ce retour d'expérience n'est pas mis en œuvre. Il existe un mur entre la conception et la fabrication. Les équipes en charge de ces étapes ne communiquent pas, ou très peu. L'équipe de conception modélise le processus de production industrielle et l'équipe de fabrication l'instancie pour fabriquer

autant de pièces que souhaité. Pas ou peu de documentation transite dans ce sens. Mais c'est encore pire dans le sens de la fabrication vers la conception où aucune information n'est « remontée ». Il arrive donc que certains problèmes liés au processus de production industrielle lui-même soient contournés à l'exécution sans que la conception ne soit avertie du problème dont elle est pourtant la source. Ce mur doit donc être cassé pour améliorer la communication de la conception vers la fabrication et permettre un retour d'expérience de la fabrication vers la conception afin d'éviter de répéter les mêmes anomalies et donc mettre en place une maintenance préventive efficace.

Le besoin de formalisation du processus de maintenance très présent en maintenance industrielle et logicielle se retrouve dans la maintenance de processus de production. Le processus défini pour la maintenance logicielle est parfaitement adapté à la maintenance de processus de production. La définition de l'exception est représentée par les étapes 1 (analyse et compréhension du logiciel) et 4 (clôture de l'intervention, archivage), la solution fonctionnelle par les étapes 2 (décision : choix d'une solution ou rejet de l'intervention) et 3 (implémentation de la solution). Nous pouvons cependant y voir une différence majeure. Alors que pour la maintenance logicielle, la solution consiste à modifier une ou plusieurs parties du code du logiciel (ainsi que la documentation qui s'y rattache), pour la maintenance de processus de production, la solution consiste à ne modifier que l'exécution du processus courant de production en ajoutant, supprimant ou modifiant une ou plusieurs étapes. Il est donc tout à fait possible de retrouver plusieurs fois la même anomalie mais sur des pièces différentes. Notre vision de la maintenance de processus de production préventive intervient à ce niveau. Si un même type d'anomalie se répète, il faut pouvoir le détecter et modifier le processus de production industrielle au niveau de sa conception et non pas seulement l'exécution de celui-ci. Encore une fois, pour cela, il faut casser le mur pour que la fabrication retourne ces informations à la conception dont le rôle ne doit pas consister qu'à modéliser les processus de production industrielle mais qui doit évoluer vers un rôle de maintenance de ces processus (Table 2).

Les enjeux de réutilisation et d'analyse d'impact spécifiques à la maintenance logicielle trouvent aussi leur place dans la maintenance de processus de production. En effet, pour résoudre une nouvelle anomalie, on peut souhaiter réutiliser une solution qui a été mise en place une ou plusieurs fois pour des anomalies dont le problème relevé est similaire à celui de l'anomalie courante. De la même manière que dans le contexte logiciel, l'élément réutilisé est plus fiable, de meilleure qualité, car il a pu être testé, évalué, amélioré. Choisir une solution qui a été utilisée pour résoudre avec succès de nombreuses anomalies est un choix plus sûr que d'exécuter une nouvelle solution qui n'est forcément pas aussi affinée et dont le résultat de l'exécution est plus incertain.

	Maintenance industrielle	*Maintenance logicielle*	*Maintenance de processus de production*
Objet	Outil de fabrication	Produit logiciel	Processus de production
Enjeux communs	Définition de la maintenance Catégories de la maintenance Qualité, retour d'expérience Formalisation du processus de maintenance		
Enjeux spécifiques	-	Réutilisation Analyse d'impact	

Table 2 – Enjeux des maintenances industrielle, logicielle et de processus de production

En ce qui concerne l'impact, celui-ci se situe à deux niveaux. Si l'on parle de maintenance de processus de production corrective, la modification de l'exécution du processus de production industrielle d'une pièce n'impacte en rien celui d'une autre pièce, même identique. La modification ne porte que sur l'exécution du processus à partir duquel l'anomalie a été créée. Cette modification peut avoir tout de même un impact direct sur les étapes suivantes du processus de production, la modification d'une étape pouvant empêcher l'exécution correcte d'une étape ultérieure. En revanche, si l'on considère une maintenance de processus de production préventive, l'impact est similaire à celui de la maintenance logicielle. En effet, la modification du processus de production industrielle au niveau de sa conception, visant à éviter l'apparition d'un problème récurrent, aura un impact sur les instanciations futures de ce processus de la même manière qu'une modification du code source d'un logiciel a un impact sur les exécutions futures de celui-ci.

1.5. Conclusion

Nous avons présenté trois maintenances qui se différencient notamment par l'objet sur lequel chacune d'elle porte. Les maintenances industrielle et logicielle qui portent respectivement sur l'outil de fabrication et le produit logiciel nous ont permis de dégager des enjeux communs que nous retrouvons dans la maintenance de processus de production qui elle, porte sur le processus de production. Le besoin de formalisation du processus de maintenance ainsi que la qualité et le retour d'expérience sont les éléments qui sont ressortis de

cette étude. Ce chapitre présentait donc le contexte de notre travail de thèse, le traitement d'anomalies qui surviennent au cours de processus industriels de fabrication, que nous avons placé dans un cadre plus général de maintenance pour en déterminer les principaux enjeux.

Parmi ces enjeux, la qualité des actions de maintenance et le retour d'expérience depuis la maintenance apparaissent comme importants. A notre sens, ce sont deux éléments essentiels dans la conduite d'une maintenance de processus de production industrielle efficace. Leur mise en œuvre passe par une exploitation des informations relatives aux anomalies résolues pour en extraire de la connaissance. En structurant cette connaissance issue du passé et en la mettant à disposition, le traitement d'une anomalie courante peut être amélioré et la compréhension des anomalies passées peut être renforcée. L'extraction et la gestion de la connaissance étant au cœur de l'accomplissement de ces enjeux, nous en présentons les concepts et les techniques dans le chapitre suivant.

Chapitre 2

Extraction et gestion de connaissances

2.1. Introduction

Effectuer des activités de maintenance corrective et préventive peut s'avérer coûteux en temps et en argent, que ce soit en maintenance logicielle, industrielle ou en maintenance de processus de production. L'exploitation des informations contenues dans les documents qui ont servi de support à ces mêmes activités de maintenance peut rendre ces dernières plus rapides, plus faciles et de meilleure qualité. En effet, les informations sur les activités passées peuvent aider à comprendre l'activité présente à réaliser voire empêcher l'apparition d'activités futures.

Ce chapitre présente les aspects de l'extraction et de la gestion de connaissances qui intéressent notre domaine de production industrielle. Les éléments relatifs aux documents, à leur structuration, à leur représentation, à leur contenu, ainsi que le type de corpus de documents que nous considérons sont exposés. L'aspect pattern d'exception que l'on retrouvera plus tard dans le chapitre 5 au travers de groupes de documents est discuté et des exemples de patterns proposés par différents auteurs dans la littérature sont détaillés.

Le processus de découverte de connaissances (processus de KDD) que nous voulons suivre pour extraire et organiser la connaissance des documents utilisés lors de la réalisation des activités de maintenance est présenté. Toutes les étapes qui le composent doivent être prises en compte pour obtenir des connaissances utiles à partir des informations sources. Il est donc nécessaire de les comprendre pour les appliquer aux activités de maintenance de processus de production.

Ce chapitre présente un état de l'art du domaine seulement. D'autres éléments relatifs à l'état de l'art mais qui concernent des choix plutôt techniques sont exposés dans les chapitres suivants.

2.2. Documentation et logiciel

2.2.1. Structure d'un document

2.2.1.1. Document non-structuré, structuré et semi-structuré

Dans la littérature, trois classes de documents sont distinguées. En fonction de sa structure, tout document appartient à l'une d'elles. Un document qui ne présente pas de structure explicite (hormis la ponctuation et les espacements) est un document non-structuré. Au contraire, celui qui présente une structure explicite, régulière, connue a priori est un document structuré. Entre les deux, se trouve le document semi-structuré qui présente une structure implicitement déclarée, irrégulière et qui peut être incomplète.

Un document *non-structuré*, appelé aussi document « plat », n'intègre aucune marque explicite d'élément de structure. Toutes les informations contenues dans le document sont présentes sous la forme d'un bloc textuel. Les seuls éléments utilisés que l'on peut considérer comme éléments de structure sont les signes de ponctuation et la mise en page (alinéa, passage à la ligne, etc.). Ce sont des marquages effectués par l'auteur qui, dans le but de communiquer une information au lecteur, propose une structuration primaire de son information (Tannier, 2006).

Dans un document *structuré*, une structure régulière et prépondérante à base de marquage descriptif est établie. Le document se présente sous la forme d'un ensemble d'informations, le contenu, organisé selon une structure hiérarchique logique qui permet d'identifier les différents éléments d'information ainsi que leur rôle dans le document. Chaque élément du contenu est une information non-structurée. Dans cette classe de documents, on ne parle plus de texte mais de données.

La structure d'un document *semi-structuré* est définie principalement par son aspect irrégulier, implicite et partiel (Abiteboul, 1997). On peut donc voir un document semi-structuré comme un document structuré dans lequel la structuration des données est plus souple soit parce que les informations sur les données sont insuffisantes, soit parce que qu'il n'est pas nécessaire d'imposer une structure rigide (le Système de Recherche d'Information qui sera mis en place sur l'ensemble des documents n'en aura pas l'utilité), soit parce qu'on n'en a pas les moyens (XML sans DTD). Un même langage de description de la structure peut être utilisé pour représenter un document structuré et un document semi-structuré, par exemple le langage XML (eXtensible Markup Language).

2.2.1.2. Structure physique et structure logique

En fonction de la manière d'aborder la Recherche d'Information, un document peut être appréhendé selon deux approches (Fuhr et al, 2001). La première approche, orientée document, considère le document sous sa forme textuelle dont le rôle principal est sa lecture. Les éléments de structuration permettent essentiellement de faciliter sa représentation visuelle (paragraphe, style de police, etc.). La deuxième approche, orientée données, a pour but la restitution de la partie la plus pertinente du document par rapport à une requête donnée. Le document est considéré comme une source de données et est utilisé pour représenter et échanger les données qu'il contient.

Quelle que soit la classe d'un document, celui-ci possède une structure physique. Elle correspond à sa présentation, à la manière dont sont agencées les données qui le composent. Il peut être composé d'un ensemble de pages, chacune d'elles composées de différents éléments textuels (ligne, paragraphe, titre de section, etc.) et multimédia (image, audio, vidéo). Cette structure physique dépend de l'environnement de présentation du document. Un même document ne sera pas présenté de la même manière s'il est affiché sur un écran d'ordinateur, sur une tablette tactile ou sur un téléphone portable.

En revanche, seuls les documents structurés et semi-structurés possèdent une structure logique, connue a priori dans le premier cas, inconnue ou partiellement connue a priori dans le deuxième. Elle est définie à la création du document et correspond à l'organisation hiérarchique de ses données. Contrairement à la structure physique, elle est indépendante de l'environnement de présentation. Elle est spécifiée par l'auteur ce qui lui permet d'organiser ses idées pour leur faire suivre un enchaînement selon un sens de lecture (Géry, 2002). Alors que la structure physique n'est d'aucune utilité pour la Recherche d'Information, la structure logique apporte une information supplémentaire utilisable par les Systèmes de Recherche d'Information pour améliorer la pertinence des résultats retournés.

2.2.2. Notre contexte de processus de production

2.2.2.1. Des corpus documentaires volumineux

Lors de l'exécution de processus de production industrielle, un grand nombre de fiches d'anomalie est produit. Ce nombre grandit avec la quantité de processus utilisés, leur complexité et le nombre d'étapes qui les composent. Dans les entreprises, plusieurs dizaines de fiches d'anomalie, voire plusieurs centaines pour les plus importantes, sont créées quotidiennement. Ces chiffres, qui peuvent paraître assez élevés, sont aussi dus au fait que dans certains

domaines sensibles, tout écart, aussi infime soit-il, entre ce qui est prévu et ce qui est réellement réalisé doit être signalé au travers de la création d'une fiche d'anomalie. Cela assure la traçabilité ainsi qu'une qualité optimale du produit final. Une fiche d'anomalie peut donc traduire un réel problème qu'il faut impérativement résoudre mais aussi un problème minime qui ne nécessite pas d'action corrective.

Contrairement à la maintenance logicielle, par exemple dans le contexte spatial (Lambolez, 1994), (Barros, 1997), où le corpus documentaire à considérer par le mainteneur est très important à cause de la diversité des documents produits pendant le cycle de vie du logiciel (cahier des charges, spécifications, documents de conception, manuel utilisateur, plan de validation, jeux de tests, etc.), la maintenance de processus de production ne considère que l'ensemble des fiches d'anomalie passées. Certes, il existe des documents qui stockent les informations relatives à l'exécution du processus de production industrielle qui a conduit à l'apparition d'une fiche d'anomalie, mais soit elles n'expliquent pas le problème rencontré, soit elles sont reprises dans la fiche d'anomalie pour décrire le problème. Toute l'information nécessaire au mainteneur est donc présente dans les fiches d'anomalie.

Des liens entre fiches d'anomalie peuvent être indiqués par les mainteneurs pour traduire une similarité entre elles. Cependant, ces référencements sont réalisés durant la correction de l'anomalie ou une fois qu'elle a été résolue et la taille des corpus documentaires ne permet pas de rendre ce travail exhaustif. Pourtant, il serait intéressant que ces liens soient produits automatiquement avant la correction de l'anomalie. Cela permettrait au mainteneur de prendre connaissance d'informations passées pouvant l'aider à corriger le problème auquel il est confronté.

2.2.2.2. Caractéristiques d'un corpus documentaire

Un corpus de fiches d'anomalie issues de processus de production industrielle présente une certaine hétérogénéité de structuration des documents, une certaine homogénéité de langage ainsi qu'une bonne qualité du contenu.

2.2.2.2.1. Hétérogénéité de structuration des documents

Une anomalie a pour support une fiche d'anomalie dans laquelle les informations qui lui sont relatives sont stockées. C'est donc un document qui possède une structure particulière, définie à l'avance. En effet, pour effectuer une maintenance efficace, les différents types d'anomalie qui peuvent survenir sont envisagés. Pour chacun d'eux, un document type est établi dans lequel les informations à collecter, les différents utilisateurs impliqués et la succession d'étapes à suivre sont définis. Même dans un domaine précis, restreint, ne concernant qu'une seule ligne de production, cette variété de types d'anomalie est présente et se traduit par une hétérogénéité des fiches d'anomalie utilisées.

Il est donc nécessaire d'établir une représentation unique des fiches d'anomalie afin de les traiter toutes ensemble indépendamment de leur type mais aussi pour considérer les nouveaux types qui pourraient apparaître par la suite. Pour cela, une modélisation suffisamment générique pour englober toute fiche d'anomalie peut être proposée (Sèdes, 1998).

2.2.2.2.2. Homogénéité de langage dans le contenu des documents

Dans un corpus, toutes les fiches d'anomalie sont issues d'un même domaine. Qu'il soit industriel ou non, ce domaine est restreint. Cela implique que le vocabulaire utilisé pour renseigner les informations est contrôlé, que la qualité terminologique du corpus est bonne. Les termes employés sont donc stabilisés, par convention ou par usage. Il est ainsi possible de supposer que les termes sont pérennes pour représenter l'information, que les mêmes mots sont employés pour définir les mêmes concepts, les mêmes idées.

De plus, le champ sémantique est partagé et réduit. Les termes employés sont donc normalisés, par convention ou par usage. Tous les documents décrivent des anomalies qui sont survenues sur différents processus de production industrielle d'un domaine restreint. Il est ainsi possible de supposer que les termes sont non-ambigus pour représenter l'information. Ils sont explicites et définissent un concept ou une idée bien précise. Les problèmes éventuels de synonymie et de polysémie sont donc grandement réduits.

2.2.2.2.3. Bonne qualité du contenu documentaire

D'une manière générale, le contenu documentaire est de bonne qualité, les informations requises sont renseignées et le sont de manière précise. Cela s'explique par le fait que le contexte dans lequel sont produites les fiches d'anomalie soit sensible et nécessite un fort besoin de traçabilité. En effet, dans des domaines tels que l'aérospatial ou l'automobile par exemple, les anomalies qui surviennent durant le processus de production industrielle sont des éléments à traiter aussi importants que l'exécution du processus de production lui-même. Les informations présentes dans une fiche d'anomalie doivent donc être de bonne qualité (égale à celles qui concerne l'exécution du processus de production) et exhaustives (complétude de l'information) pour décrire le problème rencontré afin de le corriger mais aussi pour expliquer comment ce problème a été résolu.

Cette bonne qualité du contenu documentaire est traduite par le fait que dans certaines industries, l'aéronautique en particulier, le temps passé sur les activités imprévues, le traitement d'anomalies, peut représenter plus de 50% du temps de fabrication, soit plus de temps que celui passé sur les activités prévues, l'exécution du processus standard de production. Cela se traduit aussi par le fait que les informations relatives aux anomalies peuvent représenter jusqu'à 80% du volume de l'information concernant le produit final.

2.3. Gestion de la connaissance

2.3.1. Donnée, information et connaissance

Pour pouvoir parler de connaissance, il est nécessaire de définir ce qu'est une connaissance par rapport à une information ou une donnée. En effet, ces trois termes ont un sens précis et, bien qu'ils soient liés, ils ne sont pas interchangeables. La connaissance ne peut être atteinte qu'après avoir transformé la donnée en information puis l'information en connaissance. Evidemment, ces deux étapes de transformation font appel à des traitements différents (Figure 2).

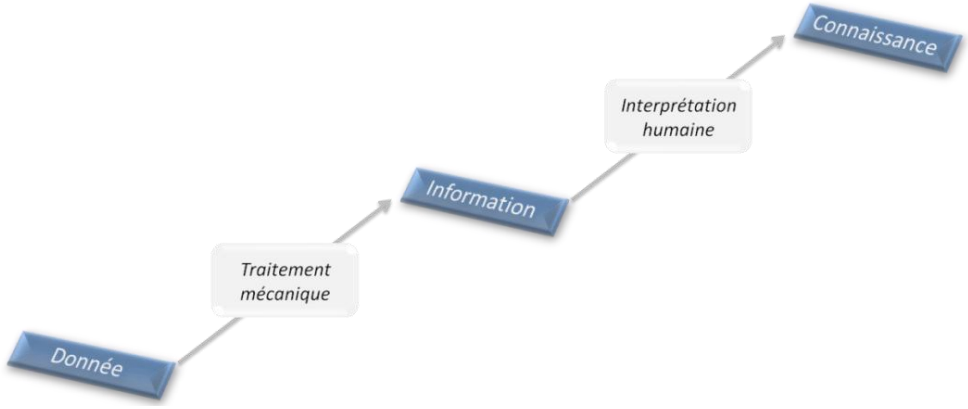


Figure 2 – De la donnée à la connaissance

D'abord, la donnée, aussi appelée donnée brute, est un élément descriptif sur lequel aucun traitement, aucune analyse n'ont été effectués. Par exemple, chaque champ d'un formulaire possède des données saisies, renseignées par un utilisateur. Dans une base de données, la donnée correspond simplement à l'intersection d'un enregistrement (une ligne) avec un champ (une colonne) dans une table.

Ensuite, on parle d'information lorsque les données sont regroupées, organisées, lorsqu'elles ont subi un traitement mécanique, c'est-à-dire soit un traitement informatique, soit un traitement manuel, humain mais sans interprétation. Par exemple, l'ensemble des données d'un formulaire saisies par un utilisateur représente de l'information. Ou encore, le fait d'avoir sélectionné, traité ou transformé des données de manière automatique génère de l'information.

Enfin, pour parler de connaissance, il faut intégrer une interprétation humaine. Sans cet aspect, l'information reste de l'information. Un outil informatique ne peut pas générer de connaissance à lui seul. L'interprétation humaine basée sur l'expérience et sur des modèles, des théories est indispensable pour

transformer l'information en connaissance. Par exemple, un algorithme identifiant des groupes de consommateurs ne génèrera que de l'information. C'est l'interprétation humaine de ces groupes obtenus, l'intérêt et les applications utiles qui en découlent qui en font de la connaissance.

2.3.2. Les patterns d'exception, supports de la connaissance

La gestion de la connaissance est devenue une activité indispensable dans les entreprises pour leur permettre de profiter d'un avantage concurrentiel sur le marché. (Ergazakis et al., 2005) définissent la gestion de la connaissance comme étant « *le processus de création de valeur à partir des actifs intangibles de l'entreprise […] dont l'essence est de fournir des stratégies pour procurer la bonne connaissance à la bonne personne au bon moment et dans le bon format* ». Elle doit donc permettre de stocker et de généraliser la connaissance pour la mettre à disposition de tous les utilisateurs, et ainsi la rendre utile. Pour mettre en place cette gestion de la connaissance, la notion de pattern, qui associe un problème et une solution, est couramment utilisée. Un pattern peut être défini comme « *décrivant un problème qui se produit encore et encore dans notre environnement et décrivant aussi le cœur de la solution à ce problème d'une telle manière que l'on peut appliquer cette solution plus d'un million de fois sans jamais la réaliser deux fois de la même façon* » (Alexander et al., 1977).

(Persson et al., 2002) décrivent un besoin de capturer la connaissance grâce à des termes abstraits, des modèles, et de stocker cette connaissance pour la mettre à disposition de tout le monde. Pour arriver à la rendre utile de cette manière, il est nécessaire de la généraliser en fonction de la façon dont on veut l'utiliser. Pour cela, les auteurs proposent une approche, the HyperKnowledge Approach, sur la gestion de connaissances, approche qui a été expérimentée dans une société de conseil dans le secteur de l'électricité (Persson et al., 2003) et sur une chaîne de conception de carreaux en céramique (Agost et al., 2006). C'est une approche en 4 étapes successives.

D'abord la connaissance existante est capturée et la nouvelle connaissance est créée. Cette étape est particulièrement importante si le processus de découverte de la connaissance est collaboratif. L'ensemble des connaissances doit être considéré comme un tout, pas comme des parties individuelles.

Puis, cette connaissance est structurée et représentée dans une forme qui permette de la rendre disponible et de la réutiliser, à l'aide de patterns génériques et abstraits. Pour cela, ils proposent un pattern d'exception défini selon deux dimensions : une dimension orientée usage et une dimension orientée connaissance. La dimension orientée usage décrit le pattern lui-même,

donne des éléments en rapport avec l'exception alors que la dimension orientée connaissance décrit l'exception suivant deux axes, le problème et la solution (Table 3).

Dimension orientée usage	
Nom	Nom du pattern
Contexte	Contexte qui nécessite l'utilisation du pattern
Contraintes	Contraintes impliquées par le pattern
Organisationnel	Détails sur les aspects organisationnels
Conséquences	Conséquences de l'application du pattern
Auteurs	Auteurs du pattern
Informations complémentaires	Informations pouvant apporter un complément sur le pattern
Applicabilité	Situations dans lesquelles le pattern peut être utilisé
Autres noms	Autres noms donnés au pattern
Exemples	Exemples d'utilisation du pattern
Dimension orientée connaissance	
Problème	Problème que le pattern tente de résoudre
Solution	Solution à appliquer pour résoudre le problème

Table 3 – Pattern d'exception proposé par (Persson et al., 2002)

Ensuite, cette connaissance est appliquée et partagée pour corriger des problèmes récurrents en effectuant une recherche de connaissance à l'aide de mots-clés ou en parcourant la hiérarchie de patterns. Les patterns permettent de réutiliser la connaissance existante, de présenter des propositions qui peuvent être facilement adaptées et réutilisées pour résoudre un problème spécifique.

Enfin, la connaissance peut être transformée et ce, de plusieurs manières :

- *De la connaissance spécifique vers la connaissance générique* : pour généraliser un problème et une solution afin qu'ils puissent être appliqués à nouveau dans d'autres contextes.
- *De la connaissance tacite vers la connaissance explicite* : pour transformer la connaissance tacite, peu verbalisable car basée sur les acquis, le savoir-faire, l'expérience des individus, en connaissance explicite, plus tangible et manipulable.
- *De la connaissance distribuée vers la connaissance intégrée* : pour mutualiser la connaissance issue de plusieurs sources différentes.

De la connaissance opérationnelle vers la connaissance stratégique : pour supporter la prise de décision dans une organisation, la connaissance est agrégée.

Les différentes transformations possibles entre connaissances tacites et explicites représentent un sujet largement traité dans la littérature. Le modèle SECI (Nonaka, 1994), (Nonaka et al., 1998), orienté conversion des connaissances dans les organisations, représente les quatre transformations possibles (Figure 3).

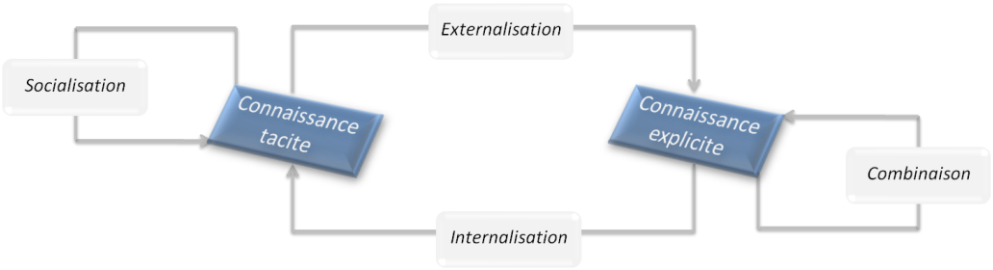


Figure 3 – Modes de transformations de la connaissance de (Nonaka, 1994)

Les auteurs ne s'intéressent donc pas seulement à la transformation de la connaissance tacite en connaissance explicite, à l'externalisation que (May et al., 2003) appellent découverte de patterns qui permet de diffuser la connaissance. Ils s'intéressent à toutes les transformations possibles :

- Tacite vers tacite, *socialisation*, pour mutualiser les connaissances personnelles intangibles ou pour les transférer.
- Explicite vers explicite, *combinaison*, pour synthétiser la connaissance en combinant des connaissances explicites, réutiliser des connaissances déjà existantes.
- Explicite vers tacite, *internalisation*, pour créer de la nouvelle connaissance tacite, c'est-à-dire que la nouvelle connaissance devient une compétence naturelle, l'individu acquiert, s'approprie la connaissance.

Ces mêmes auteurs qui proposent l'approche HyperKnowledge définissent aussi un autre pattern d'exception comme mettant en relation un problème récurrent et sa solution, chaque problème ayant des caractéristiques uniques qui le distinguent des autres problèmes (Persson et al., 2006). Ils distinguent les patterns conceptuels qui associent des solutions éprouvées, bien établies, à des problèmes récurrents et les patterns organisationnels qui doivent être vus comme des propositions abstraites et génériques qui peuvent être facilement adaptées et réutilisées. Ils présentent un pattern organisationnel composé de 5 éléments (Table 4). Pour avoir des informations qui sont à la base de la création de connaissance, ils vont même plus loin en définissant un anti-pattern. Contrairement au pattern qui doit capturer un savoir-faire, l'anti-pattern capture une mauvaise solution pour un problème (Table 5).

Nom	Nom du pattern
Problème	Problème que le pattern tente de résoudre
Contexte	Contexte qui provoque l'apparition du problème
But	But à atteindre en résolvant le problème
Solution	Solution à appliquer pour résoudre le problème

Table 4 – Pattern d'exception proposé par (Persson et al., 2006)

Nom	Nom de l'anti-pattern
Problème	Problème que le pattern tente de résoudre
Anti-solution	Solution appliquée pour résoudre le problème, conditions d'exécution
Résultats	Résultats obtenus, conséquences inattendues après application de la solution
Erreurs principales	Causes qui ont conduit à l'échec de la résolution du problème

Table 5 – Anti-pattern d'exception proposé par (Persson et al., 2006)

Nom	Nom du pattern
Contexte	Contexte qui provoque l'apparition du problème
Problème	Problème que le pattern tente de résoudre
Contraintes	Contraintes impliquées par le problème
Solution	Solution à appliquer pour résoudre le problème
Organisationnel	Détails sur les aspects organisationnels
Contexte résultat	Résultat obtenu, conséquences de l'application de la solution
Patterns associés	Patterns existants ayant un lien avec le pattern, avec lequel il peut être en conflit

Table 6 – Pattern d'exception proposé par (May et al., 2003)

(May et al., 2003) se situent dans la conception et le développement logiciels et ont la même vision des activités de la gestion de connaissance, à savoir présenter et structurer l'information, construire des patterns qu'ils appellent « frameworks conceptuels » et aussi supporter les utilisateurs qui possèdent la connaissance critique. Ils présentent un pattern d'exception plus complet que celui de (Persson et al., 2006). On y retrouve le nom, le contexte, le problème,

la solution et l'aspect organisationnel auxquels s'ajoutent les forces du pattern, le contexte résultat et les patterns qui lui sont liés (Table 6).

(Lerner et al., 2010), quant à eux, présentent un pattern d'exception dans un contexte de modélisation de processus (Table 7).

Nom	Nom du pattern
But	Comportement récurrent que le pattern capture
Applicabilité	Situations dans lesquelles le pattern peut être utilisé
Structure	Structure générale du processus à mettre en place
Participants	Rôle des différentes parties du processus, des personnes qui y prennent part
Exemples	Deux exemples de processus qui utilisent le pattern
Variations	Variations, changements mineurs qui peuvent être apportés au pattern pour obtenir des effets légèrement différents

Table 7 – Pattern d'exception proposé par (Lerner et al, 2010)

Tous ces exemples de pattern s'accordent avec les définitions de (Alexander et al., 1977) et (Persson et al., 2006) présentées plus haut. On retrouve dans chacun d'eux, un élément qui est renseigné par la description de l'incident rencontré, le problème que le pattern capture et doit résoudre et un autre élément dans lequel la solution à mettre en place pour résoudre le problème est renseignée. Ces deux éléments appelés *problème* et *solution* dans les deux premiers exemples sont appelés respectivement *but* et *structure* dans le troisième. Néanmoins, ce sont bien les mêmes informations qui y sont renseignées. A un type de problème est donc bien associé un type de solution. A un même problème capturé plusieurs fois, une même solution est appliquée dont la réalisation peut varier. En plus de ces deux éléments indispensables dans un pattern, le *contexte* d'apparition de l'exception, appelé *applicabilité* dans le troisième exemple, apparaît aussi comme important. Les autres éléments tels que les patterns associés ou les exemples semblent être moins importants. Ils doivent être utilisés plutôt dans un domaine particulier, pour définir un pattern plus spécifique aux besoins.

Pour modéliser une exception, (Somekh et al., 2007) s'intéressent à ses caractéristiques (Table 8). L'accent n'est pas porté sur l'association problème-solution mais uniquement sur le problème. En effet, on la description du problème au travers de l'élément *type de déclencheur*. Les autres éléments ne définissent pas le problème lui-même, l'incident rencontré, ils sont plutôt des métadonnées sur ce problème. Cette modélisation peut être vue comme un patron de pattern de problème d'exception.

Type de déclencheur	Evénement qui déclenche l'exception au cours du processus
Prédictibilité	Prévisible, imprévisible
Source	Interne (systémique), externe (environnementale)
Synchronisation	Synchrone (spécifiée avant un point de décision), asynchrone
Fréquence	Rare (peut faire l'objet d'une procédure spécifique), fréquente
Mesurabilité	Mesurable (valeur donnée atteinte, possède une unité), immesurable
Niveau de portée	Relative à une unique tâche, à un bloc de tâches, au système (toutes les tâches)
Gravité	Ignorable, faible, réelle, importante

Table 8 – Caractéristiques d'une exception par (Somekh et al., 2007)

Un pattern d'exception est donc une structure qui rassemble un ensemble d'informations concernant notamment un incident rencontré de façon répétée, appelé problème, et un protocole à appliquer pour le résoudre, appelé solution. Son intérêt principal est la réutilisabilité. Plusieurs exceptions ayant un problème similaire et une solution similaire peuvent ainsi être regroupées dans un même pattern. Les patterns peuvent être définis a priori, en prévoyant à l'avance les différents problèmes qui peuvent être rencontrés et en leur associant la solution adaptée. Cela nécessite une excellente connaissance du domaine pour prévoir une liste de patterns aussi exhaustive que possible. Ils peuvent aussi être définis a posteriori, en analysant les informations relatives aux exceptions qui se sont produites dans le passé, informations sur les problèmes détectés et sur les traitements appliqués sur chacun d'eux pour les résoudre. Grâce à un algorithme de regroupement basé sur la similarité entre problèmes et entre solutions, des groupes peuvent être déterminés. Chacun de ces groupes fait émerger un type de problème et un type de solution correspondant, chacun d'eux peut donc être qualifié de pattern.

2.3.3. Traitement des exceptions

Pour traiter les exceptions différentes techniques peuvent être utilisées.

(Shah et al., 2007) proposent une approche utilisant une ontologie dans un système multi-agents. Cette approche permet de résoudre le problème d'interopérabilité des processus de diagnostic des exceptions entre organisations en représentant et en analysant les exceptions de manière

uniforme. Pour cela, les actions des agents ainsi que les concepts et les prédicats sont parfaitement spécifiées. Pour chaque action définie par l'approche, les informations à connaître qui décrivent l'exception sont spécifiées. Les agents vont ainsi renseigner ces informations. Certains diagnostiquent le problème (« *diagnosis agents* »), d'autres corrigent ce problème (« *problem solving agents* »).

(Wang et al., 2004) proposent aussi un système multi-agents mais dans un contexte de transactions sécurisées. Encore une fois, différents agents intelligents sont proposés pour fournir un ensemble de fonctionnalités pour traiter les exceptions. Parmi eux, on retrouve ceux qui diagnostiquent le problème (« *diagnostic agents* ») et ceux qui corrigent ce problème (« *resolution agents* »). La connaissance liée à la gestion des exceptions est aussi stockée dans des agents (« *repository agents* »).

(Curbera et al., 2003) se situent dans un contexte de BPEL4WS (Business Process Execution Language for Web Services). « *BPEL4WS fournit un langage pour la spécification formelle de processus métier et de protocoles d'interaction métier. De la sorte, il étend le modèle d'interaction des web services et lui permet de soutenir les transactions commerciales. BPEL4WS définit un modèle d'intégration interopérable qui devrait faciliter le développement de l'intégration des processus automatisés en intra-entreprise et en business-to-business (B2B).* » http://www.ibm.com/developerworks/library/specification/ws-bpel/. Le traitement des exceptions retenu est différent des approches habituelles dans les processus métier. En effet, le traitement d'exception dans le BPEL4WS n'est pas contraint par une topologie fixe et permet de considérer aussi bien les processus très structurés que ceux à base de graphe orienté.

(Yaxiong et al., 2010) proposent de contrôler leur processus métier en continu et de faire des analyses en temps réel pour connaître l'état des processus, détecter une exception, la notifier et la corriger en se basant sur la connaissance extraite de ce même modèle de processus métier instancié dans le passé. En étudiant les données historiques, des actions préventives peuvent être mises en place pour garantir le bon déroulement de la prochaine instance du modèle du processus métier.

Enfin, (Kao et al., 2010) proposent un système de gestion des exceptions basé sur le modèle PDCA (Plan-Do-Check-Action) pour traiter les exceptions qui surviennent au cours d'un processus de fabrication dans l'industrie du semi-conducteur. Ici, l'étape d'action a pour but le traitement d'exceptions. Ce modèle adopte le modèle 8D (8 disciplines) dans lequel 8 étapes sont suivies : 1) préparation du processus et construction d'une équipe, 2) description du problème, 3) identification et mise en place des actions immédiates, 4) identification des vraies causes, 5) validation des actions correctives permanentes, 6) mise en œuvre des actions correctives permanentes, 7) prévention de toute récidive et 8) félicitation de l'équipe, afin notamment de

décrire le problème et d'y trouver une solution pour le corriger (traitement à court terme pour une correction immédiate) mais aussi d'en déterminer les causes profondes et les actions correctives permanentes (traitement à long terme pour une prévention de la récidive).

2.3.4. Le processus d'extraction de la connaissance

(Clark, 2011) distingue deux modes de création de la connaissance, la généralisation et l'expansion appelés respectivement *probability estimates view* et *enacted salience view* par (Lampel et al., 2009).

Dans le premier mode, des classes générales d'exception sont définies. Chaque nouvelle exception est classifiée dans une de ces classes, chacune est considérée comme une instance spécifique d'une classe plus générale. On s'intéresse donc aux éléments des exceptions qui permettent de calculer la similarité entre elles et ainsi de les regrouper selon cette similarité. Cela permet de tirer des enseignements largement applicables.

Dans le second mode, l'accent n'est pas porté sur la recherche de similarité entre exceptions mais au contraire sur les éléments spécifiques de chacune d'elles. Si une exception est considérée comme critique, il est intéressant de chercher à recueillir des informations complémentaires, de comprendre le problème afin d'en extraire une connaissance nouvelle. Ce mode consiste à transformer des exceptions spécifiques en classes d'exceptions plus générales en y prêtant attention, en cherchant des données complémentaires et en raisonnant sur l'exception.

L'extraction de connaissances (Knowledge Discovery in Databases, KDD) a pour but de transformer des données brutes, volumineuses, difficilement manipulables, en une autre forme qui soit plus compacte (un compte-rendu, un bilan par exemple) ou plus abstraite (un pattern qui explique les données par exemple) ou plus utile (un pattern prédictif pour estimer les valeurs des prochaines occurrences par exemple). Cette transformation ne peut pas être manuelle car elle serait lente, coûteuse et très subjective. De plus, cela devient irréalisable lorsque l'on veut traiter de gros volumes de données.

Pour répondre à ce besoin d'automatisation ou au moins de semi-automatisation du traitement de telles données, le processus de KDD est mis en place. Il fournit des outils pour automatiser le processus entier d'analyse de données afin d'en extraire la connaissance utile. Il met en jeu des algorithmes de modélisation pour des ensembles de données conséquents et contenant du bruit, algorithmes robustes et supportant le passage à l'échelle. Ce processus interactif et itératif proposé par (Fayyad et al., 1996a) est composé de cinq étapes successives dont l'entrée est l'ensemble des données brutes et la sortie est la connaissance extraite de ces données (Figure 4).

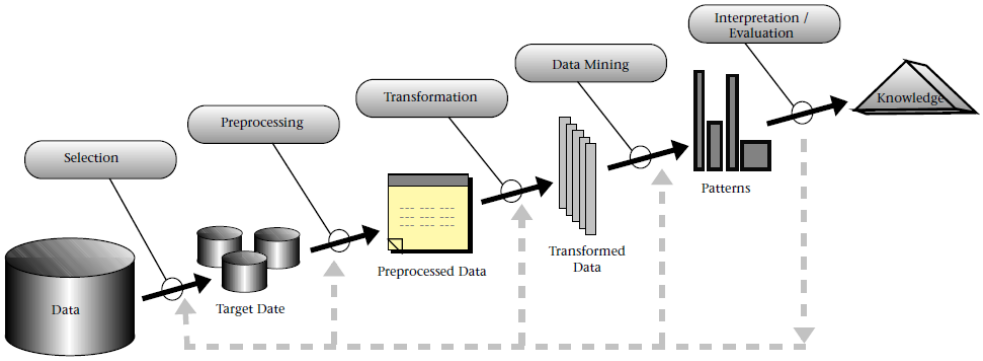


Figure 4 – Le processus de KDD défini par (Fayyad et al., 1996a)

C'est un processus qualifié d'interactif car l'utilisateur est impliqué dans chacune d'elles, il doit prendre des décisions tout au long de l'exécution du processus. C'est un processus qualifié aussi d'itératif car lors de la dernière étape d'interprétation des résultats, il est possible de revenir à toute étape précédente afin d'améliorer les résultats obtenus. Les cinq étapes qui composent ce processus sont :

1. La sélection des données (*selection*). Toutes les données brutes que l'on possède ne sont pas nécessairement utiles pour générer de la connaissance. Cette étape filtre les données jugées pertinentes, les données cibles que l'on va réellement traitées.
2. *Le pré-traitement des données (preprocessing).* Une fois les données sélectionnées, des traitements mineurs leur sont appliqués pour éliminer le bruit ou gérer les valeurs manquantes par exemple. Cette étape de nettoyage des données fournit des données pré-traitées pour améliorer la qualité des résultats obtenus par les étapes suivantes.
3. *La transformation des données (transformation).* Cette étape est entièrement dépendante de l'étape de fouille de données qui suit. L'objectif est d'organiser les données pré-traitées selon une structure qui soit adaptée à l'algorithme de fouille de données. On peut utiliser un algorithme de régression ou de classification par exemple.
4. *La fouille de données (data mining).* Ce n'est qu'une étape de ce processus, ce n'est pas le processus lui-même. Elle consiste à effectuer l'analyse des données et à appliquer des algorithmes d'extraction de connaissances pour produire différents patterns. Elle prend en entrée les données transformées et en extrait des patterns à l'aide d'algorithmes spécifiques tels que des algorithmes de clustering ou de classification à base de règles ou d'arbres par exemple. La qualité des patterns obtenus est directement liée à la qualité des traitements des trois étapes précédentes.
5. *L'interprétation, l'évaluation des patterns (interpretation / evaluation).* Cette étape implique une visualisation des patterns obtenus pour qu'un

expert puisse évaluer les résultats obtenus. Il est nécessaire que les patterns produit par l'étape de fouille de données soient valides sur les données avec un certain degré de certitude, soient nouveaux (au moins pour le système et de préférence pour l'utilisateur), et soient potentiellement utiles et compréhensibles. C'est à cette étape que la connaissance est créée, seule l'interprétation humaine est capable de réaliser cette opération. Si les résultats obtenus ne sont pas interprétables ou semblent perfectibles, la décision de revenir à l'une des étapes précédentes est possible.

2.4. Représentation de documents textuels

Pour traiter un document, tous les algorithmes ne travaillent pas nécessairement directement sur le document lui-même. Les termes qui composent chaque document peuvent être extraits. Ce sont ces termes que les certains algorithmes manipulent.

2.4.1. Pré-traitement du texte

Deux traitements principaux sont couramment utilisés. Le premier consiste à supprimer les mots vides. Ce sont les mots communs qui n'apportent pas de valeur significative au texte. On distingue deux types de mots vides, ceux relatifs à la langue et ceux relatifs au contexte, au thème traité. Les mots vides relatifs à la langue concernent tous les mots de liaison qui permettent de structurer les phrases. Ce sont les déterminants, les conjonctions, les prépositions, les pronoms, etc. Les mots vides relatifs au contexte concernent au contraire les noms communs, les verbes, etc., les mots qui apportent du sens. Cependant, dans un contexte précis, un mot peut être tellement employé dans les documents qu'il n'est plus considéré comme discriminant. On distingue donc les mots vides par rapport au langage utilisé et les mots vides par rapport à la collection de documents considérés. Techniquement, lorsqu'un mot, quel qu'il soit, possède une distribution uniforme sur l'ensemble des documents considérés, il est considéré comme un mot vide.

Le deuxième traitement est la lemmatisation. Il consiste à représenter les mots sous une forme plus simple. Une première solution simple consiste à normaliser la forme des mots en réduisant les verbes sous leur forme infinitive, et les noms communs et les adjectifs au masculin singulier. Les avantages de ce type de traitement sont d'une part de réduire le nombre de formes de mots d'un document et d'autre part d'augmenter le nombre d'occurrences des formes obtenues par rapport aux formes initiales. Des solutions permettent des

réductions plus importantes. Par exemple, l'algorithme de suppression de suffixe de Porter (Porter, 1980), couramment utilisé, réduit les mots à leur racine. L'article fondateur introduisant cet algorithme a été publié à nouveau 26 ans plus tard sous une forme identique (Porter, 2006), démontrant qu'il est toujours d'actualité et renforçant sa crédibilité. Cet algorithme basé sur un dictionnaire de suffixes applique cinq étapes successives sur chaque mot pour supprimer progressivement toutes les parties du mot seulement relatives à son suffixe et donc ne conserver que la racine du mot. En fait, on conserve la racine du mot avec le préfixe s'il y en a un, cet algorithme ne travaillant que sur les suffixes. Cependant, l'utilisation de préfixes est assez faible, contrairement à l'utilisation de suffixes. Aucun élément linguistique n'est utilisé. Le fait de ne conserver que la racine des mots accentue encore plus les avantages de ce type de traitement. Dans son article, Porter donne l'exemple des mots « connect », « connected », « connecting », « connection » et « connections » qui ont tous le même sens. Après avoir appliqué son algorithme, ils sont tous représentés par « connect ». Les mots « generalization » et « oscillators » deviennent respectivement « gener » et « oscil ».

2.4.2. Algorithmes de réduction de dimensions

Malgré ce travail de pré-traitement du texte, de nombreux termes peuvent avoir été extraits et il peut être intéressant de réduire ce nombre de termes. Pour cela, il existe de nombreux algorithmes qui déterminent les termes influents à conserver et les termes peu influents que l'on peut omettre sans perdre d'information, ou en n'en perdant que très peu. Nous présentons quatre algorithmes parmi les plus connus et utilisés.

2.4.2.1. Décomposition en valeurs singulières (Singular Value Decomposition, SVD)

L'algorithme de SVD (Stewart, 1993) décompose une matrice rectangulaire en valeurs propres et vecteurs propres. C'est une généralisation du théorème spectral qui énonce qu'une matrice normale, carrée, peut être diagonalisée par une base de vecteurs propres. L'objectif de cet algorithme est de trier les dimensions selon leur importance et de déterminer à partir de quel rang, la suppression des dimensions qui suivent minimise un certain critère, ici, la norme de Frobenius :

$$Norme\ de\ Frobenius = \|A\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}\left|a_{ij}\right|^2 = trace(AA^T) = \sum_{i=1}^{\min\{m,n\}}\sigma_i^2 \quad \textbf{(1)}$$

Pour cela, une matrice terme-document A = mxn est décomposée en une matrice A = UΣVT avec :

- U (vecteur singulier à gauche) : matrice mxm, ses colonnes sont les vecteurs propres A*AT, dits d'entrée ou d'analyse.
- Σ : matrice diagonale mxn, contient les valeurs singulières de A (les racines carrées des valeurs propres A*AT).
- V (vecteur singulier à droite) : matrice nxn, ses colonnes sont les vecteurs propres AT*A, dits de sortie.

On cherche à trouver le meilleur rang r pour l'approximation A' = UΣ'VT de la matrice A, Σ' gardant les r valeurs singulières les plus grandes de Σ, soit les dimensions les plus influentes. On utilise la norme de Frobenius à minimiser, la distance entre les matrices M et M' doit être minimale.

2.4.2.2. Réduction multidimensionnelle (MultiDimensional Scaling, MDS)

L'algorithme de MDS (Cox et al., 2008) a pour objectif d'assigner des coordonnées dans un espace euclidien à un ensemble d'objets, de telle manière que la répartition spatiale qui en résulte obéisse à des relations de similarité (ou de dissimilarité) pour des propriétés données de ces objets. Comme c'est un algorithme très orienté visualisation, la réduction de dimensions doit être très importante, il ne faut en garder que 2 voire 3 pour permettre une représentation géométrique interprétable. L'algorithme de MDS consiste en une recherche itérative de vecteurs dans un espace à n dimensions, de telle sorte que la matrice des distances entre les objets tende à minimiser un critère donné de stress :

$$Stress = \frac{\sum \left(f\left(d_{ij}\right) - d'_{ij}\right)^2}{\sum f\left(d_{ij}\right)^2} \quad (2)$$

L'algorithme se déroule ainsi :

1. Assignation de coordonnées arbitraires aux objets dans un espace à n dimensions.
2. Calcul des distances euclidiennes entre les paires de points pour former une matrice Dhat.
3. Comparaison de la matrice Dhat avec la matrice D d'entrée en évaluant la fonction de stress. Plus la valeur obtenue est faible, meilleure est la correspondance entre les deux matrices.
4. Ajustement des coordonnées de chaque point dans la direction qui minimise au mieux le stress.
5. Répétition des étapes 2 à 4 jusqu'à ce que la fonction de stress ne diminue plus.

Ci-dessous, un exemple (http://www.analytictech.com/borgatti/mds.htm) de matrice d'entrée (Table 9) et de sa représentation spatiale suite à l'application de la méthode MDS. Les axes n'ont pas de réelle valeur sémantique, c'est la position spatiale des objets les uns par rapport aux autres qui importe dans cette représentation.

	Boston	NY	DC	Miami	Chicago	Seattle	SF	LA	Denver
Boston	0	206	429	1504	963	2976	3095	2979	1949
NY	206	0	233	1308	802	2815	2934	2786	1771
DC	429	233	0	1075	671	2684	2799	2631	1616
Miami	1504	1308	1075	0	1329	3273	3053	2687	2037
Chicago	963	802	671	1329	0	2013	2142	2054	996
Seattle	2976	2815	2684	3273	2013	0	808	1131	1307
SF	3095	2934	2799	3053	2142	808	0	379	1235
LA	2979	2786	2631	2687	2054	1131	379	0	1059
Denver	1949	1771	1616	2037	996	1307	1235	1059	0

Table 9 – Matrice de distances (en km) entre villes américaines

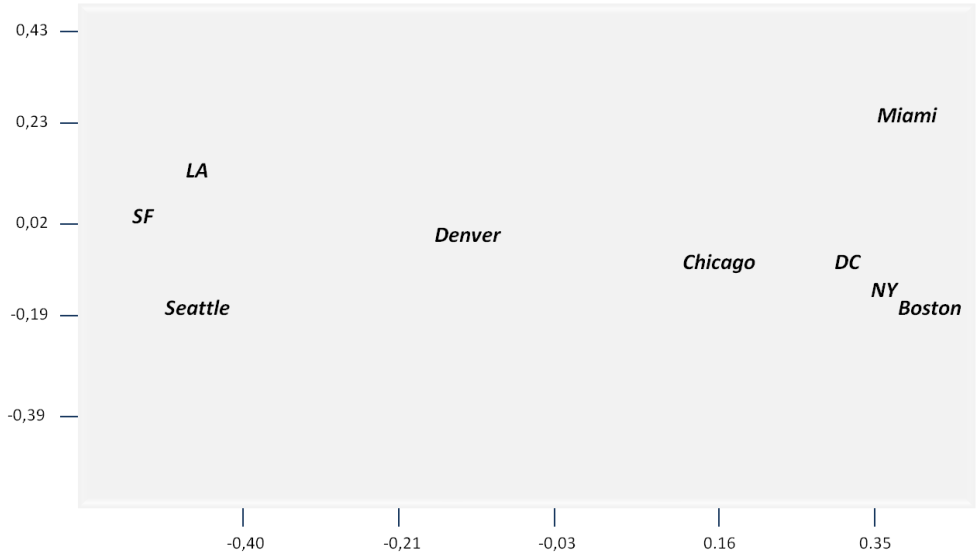


Figure 5 – Réprésentation spatiale de villes américaines après application de l'algorithme MDS

2.4.2.3. Analyse en composante principale (Principal Component Analysis, PCA)

L'algorithme de PCA (Abdi et al., 2010) a pour objectif la réduction du nombre de dimensions en maintenant autant que possible la variance. Pour cela, les variables liées entre elles (corrélées) sont transformées en nouvelles variables indépendantes les unes des autres (non corrélées). Ces nouvelles variables sont appelées composantes principales. C'est un algorithme couramment utilisé pour la réduction de bruit, la visualisation de données et en tant que pré-traitement précédent l'application d'un algorithme de fouille de données tel que le clustering.

Dans un premier temps, des transformations sont appliquées sur la matrice d'entrée M de manière à obtenir une matrice centrée (en soustrayant la moyenne) réduite (en divisant par l'écart-type). La réduction permet d'éviter qu'une variable a forte variance occulte les autres variables. Ensuite, on cherche le vecteur u tel que la projection du nuage d'objets sur u ait une variance maximale : $\pi_u(M) = M.u$ et on répète l'opération pour déterminer autant de vecteurs que nécessaire.

2.4.2.4. Analyse sémantique lantente (Latent Semantical Analysis, LSA)

L'algorithme de LSA (Deerwester et al., 1990), aussi appelée Latent Semantique Indexing, LSI, identifie des modèles de relation entre les termes et les concepts contenus dans un document au contenu non-structuré. Il se base sur le fait que les mots utilisés dans le même contexte tendent à avoir le même sens. Il cherche donc à trouver des relations sémantiques entre les termes. Lors de la recherche (requête), les résultats retournés sont conceptuellement similaires au sens des critères de recherche même si les résultats ne partagent pas de termes avec les critères de recherche. Les bénéfices de cet algorithme sont nombreux. Il résout les problèmes de synonymie (qui diminuent le rappel) et de polysémie (qui diminuent la précision), il est tolérant au bruit, il facilite la classification non-supervisée et son approche mathématique le rend indépendant du langage utilisé, il n'est pas nécessaire d'employer de dictionnaire ou de thésaurus.

Le point de départ de l'algorithme est une matrice terme-document A. D'abord, les valeurs de la matrice sont multipliées par des poids locaux et globaux. Ensuite cette matrice est transformée en une matrice A = TSDT grâce à une réduction par SVD avec :

- T : matrice de vecteurs terme-concept
- S : matrice de valeurs singulières
- D : matrice de vecteurs concept-document

En réduisant les dimensions, s'arrêtant au rang k, on a $A \approx A_K = T_K S_K D_K^T$. Certaines versions de l'algorithme de LSA peuvent calculer seulement les k premières valeurs singulières et les vecteurs de termes et de documents, d'autres doivent tronquer.

2.4.3. Pondération de termes

La pondération des unités textuelles permet de leur affecter un poids en fonction de leur pertinence, de leur potentiel informatif. Elle est principalement utilisée en Recherche d'Information pour déterminer la similarité entre une requête fournie par l'utilisateur et les documents présents dans la base. Généralement, ce poids est compris entre 0 et 1. Dans le cas binaire, 0 indique l'absence du terme dans le document alors que 1 indique sa présence. Dans le cas non binaire, plus généralement utilisé, la mesure tf-idf (Term Frequency – Inverse Document Frequency) est couramment employée, particulièrement en Recherche d'Information (Salton et al., 1983), (Salton et al., 1988) et en Text Mining (Feldman et al., 1998), (Fatudimu et al., 2008) Cette mesure statistique permet d'évaluer l'importance d'un terme par rapport à un document extrait d'un corpus de documents et par rapport au corpus même.

La fréquence de terme (TF) est le nombre d'occurrences de chaque terme dans un document. Pour éviter les biais, ce nombre est normalisé en le divisant par le nombre d'occurrences de tous les termes du document.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad \textbf{(3)}$$

Où le numérateur est le nombre d'occurrences du terme ti dans un document Dj et le dénominateur le nombre d'occurrences de tous les termes du même document.

La fréquence inverse de document (IDF) est un indicateur de rareté d'un terme dans les documents du corpus. Plus le nombre de documents possédant le terme est grand, plus la valeur de l'IDF pour ce terme est petite.

$$idf_i = log \frac{|D|}{|\{d_j : t_i \in d_j\}|} \quad \textbf{(4)}$$

Où le numérateur est le nombre de documents dans le corpus et le dénominateur le nombre de documents possédant le terme ti.

Le poids d'un terme est le produit de ces deux mesures.

$$w_{ip}^{d_j} = tf_{i,j}.idf_i \quad \textbf{(5)}$$

Pour calculer la similarité entre documents, la mesure cosinus, aussi appelée similarité cosinus, est une mesure fréquemment utilisée, notamment lorsque la pondération des termes est effectuée avec la mesure tf-idf.

$$sim(d1, d2) = \sum_{i=1}^{m} \left(w_{ip}^{d1} w_{ip}^{d2} \right) \Big/ \sqrt{\sum_{i=1}^{m} \left(w_{ip}^{d1} \right)^2 . \sum_{i=1}^{m} \left(w_{ip}^{d2} \right)^2} \quad (6)$$

2.5. Conclusion

Dans ce travail de thèse, l'extraction et la gestion de connaissances occupe une place prépondérante. Ce chapitre a notamment exposé la manière dont ce thème est traité dans la littérature ainsi que la manière de manipuler des documents, en particulier textuels. Cela passe par un processus de découverte de connaissances, le processus de KDD, qui détermine des patterns dans les données. Dans la littérature, de nombreux patterns d'exception sont présentés, tous associant des éléments qui concernent le problème rencontré avec des éléments qui concernent la solution à appliquer pour résoudre le problème.

Nous proposons de mettre en évidence les patterns sur des données de notre contexte spécifique, c'est-à-dire les données issues de documents relatifs à la maintenance de processus de production industrielle. Pour effectuer ce travail, les cinq étapes du processus d'extraction de connaissances seront la ligne directrice de notre proposition. Nous définirons pour chacune de ces étapes un traitement spécifique à la maintenance de processus de production industrielle. Les deux premières étapes seront définies grâce à la modélisation de fiche d'anomalie que nous proposons au chapitre 4 et les trois étapes suivantes par le framework que nous proposons au chapitre 5.

Pour effectuer l'étape de fouille de données, il est nécessaire de classifier les documents, les fiches d'anomalie en ce qui nous concerne. Le chapitre suivant fait un tour d'horizon des différents algorithmes de classification, des environnements de regroupement utilisables.

Chapitre 3

Classification de documents

3.1. Introduction

La classification de documents est une activité qui consiste à regrouper les documents d'un corpus en classes. Il existe deux grands types de classification, la classification supervisée et la classification non supervisée. Dans la classification supervisée, les classes possibles sont connues à l'avance grâce à une technique externe. Chaque document est donc naturellement associé à la classe qui lui correspond le mieux selon la formule de similarité utilisée. Il est aussi possible qu'un document appartienne à plusieurs classes en utilisant des probabilités d'appartenance du document pour chacune des classes. Dans la classification non supervisée, les classes possibles sont inconnues à l'avance. C'est au cours de l'exécution de l'algorithme que les classes sont déterminées et que les documents y sont associés. Comme en classification supervisée, un document peut n'appartenir qu'à une classe ou bien appartenir à plusieurs classes en utilisant des probabilités d'appartenance. Nous pouvons aussi citer des types de classification qui se situent à la rencontre du supervisé et du non-supervisé : la classification semi-supervisée dans laquelle seulement certaines classes sont connues et la classification partiellement supervisée dans laquelle on ne connaît pas toutes les probabilités d'appartenance d'un document aux classes.

Compte-tenu de notre contexte, nous allons nous intéresser à la classification non supervisée. En effet, nous ne pouvons pas connaître à l'avance les classes possibles. Ce chapitre présente donc différents algorithmes de classification non supervisée. Plusieurs méthodes de clustering sont exposées (hiérarchique, par partition, hybrides et subspace) et comparées selon de nombreux critères. D'autres environnements de regroupement à notre disposition sont ensuite présentés : le raisonnement à base de cas et son cycle en 4 étapes adapté à la résolution de nouveaux problèmes en se basant sur les expériences passées, les cartes auto-organisatrices, l'analyse formelle de concepts et les R-tree. La classification par facettes est aussi détaillée. Pour évaluer la performance de ces algorithmes, quatre métriques basées sur la matrice de confusion sont présentées.

3.2. Le clustering

3.2.1. Caractérisation des méthodes de clustering

Il existe de nombreuses familles de méthodes de clustering. La grande quantité de méthodes présentes dans la littérature amène naturellement à les hiérarchiser. Pour les comparer entre elles, plusieurs critères peuvent être appliqués. 14 critères sont proposés par L. Candillier dans sa thèse (Candillier, 2006), et sont définis comme suit.

3.2.1.1. Les connaissances a priori

Ce sont des connaissances requises de la part de l'utilisateur. C'est un critère important car si l'utilisateur ne les connaît pas toutes, les informations qu'il va donner risquent d'être éloignées de la réalité de son problème et les groupes obtenus ne seront pas pertinents. De nombreuses méthodes demandent le nombre de groupes à obtenir ou la distance minimale entre deux groupes (seuil de similarité).

3.2.1.2. La présentation des résultats

Le choix de la famille de méthodes utilisée a un impact important sur la présentation des résultats. Donc, même si cette caractéristique semble très avale, il est nécessaire de l'intégrer dès le début. Une méthode hiérarchique exposera plutôt une hiérarchie de groupes alors qu'une méthode par partition fournira plus généralement une partition de l'ensemble des documents. Evidemment, il est possible d'afficher les résultats sous d'autres formes, selon les besoins de l'utilisateur. Cependant, plus on s'éloigne d'une présentation standard des résultats, plus il est difficile de la mettre en place.

3.2.1.3. La complexité

La complexité des méthodes détermine le temps de réponse de la méthode. Ainsi, sur une même base de données une méthode de complexité linéaire aura un temps de réponse inférieur à une méthode de complexité quadratique. Cependant, les groupes obtenus n'auront peut-être pas la même précision. Il faut donc garder à l'esprit l'utilisation que l'on veut en faire ainsi que le volume des données qui doit être traité.

3.2.1.4. Le caractère déterministe

Avec des données identiques en entrée, une méthode déterministe donnera toujours le même résultat. En revanche, toujours avec des données identiques en entrée, une méthode stochastique pourra donner des résultats différents car elle n'exécutera pas la même séquence d'opérations, celle-ci étant définie de manière aléatoire. L'utilisation de l'une ou l'autre méthode peut avoir un impact sur la précision du résultat, le temps de calcul et la mémoire utilisée (Bossy et al., 1997).

3.2.1.5. Le caractère incrémental

Une méthode incrémentale a la capacité de traiter les données en entrée au fur et à mesure de leur intégration dans l'algorithme sans avoir à relancer le processus sur les données déjà traitées. En revanche une méthode non incrémentale est exécutée sur un ensemble de données fournies en entrée et si on ajoute de nouvelles données en entrée, elle devra être exécutée à nouveau sur l'ensemble des données.

3.2.1.6. Le caractère any-time

A n'importe quel moment du déroulement de la méthode, une méthode any-time est capable de fournir un résultat intermédiaire, même s'il n'est pas optimal, alors qu'il faut attendre la fin de l'exécution totale d'une méthode qui n'est pas de type any-time pour obtenir un résultat, cette dernière n'étant pas capable de fournir un résultat intermédiaire.

3.2.1.7. Le chevauchement de groupes (overlapping)

Certaines méthodes ne permettent d'associer un document qu'à un seul groupe, ce sont les méthodes hard, et d'autres proposent au contraire d'associer un document à plusieurs groupes, ce sont les méthodes soft. Un document appartenant à plusieurs groupes possède un degré d'appartenance différent pour chacun d'eux. Ainsi, on peut se ramener d'une méthode soft à une méthode hard en associant chaque document au groupe dont l'appartenance est la plus forte.

3.2.1.8. La prise en compte du contexte

On se propose ici de séparer les méthodes traditionnelles qui ne prennent pas en compte le contexte et qui considère donc que tous les groupes obtenus sont définis suivant les mêmes dimensions des méthodes de subspace clustering qui

cherchent les dimensions les plus pertinentes pour chacun des groupes et ce de manière individuelle.

3.2.1.9. La tolérance au bruit

La tolérance au bruit est la capacité d'une méthode à gérer le bruit qui peut exister dans les données, c'est-à-dire le fait que certains documents ne suivent pas la distribution générale des autres documents. Une méthode tolérante au bruit sera donc capable de créer des groupes avec les documents non bruités et ce, sans qu'elle ne rattache obligatoirement chaque document bruité à un groupe.

3.2.1.10. La tolérance à l'effet de chaîne

On parle d'effet de chaîne lorsque deux groupes proches mais distincts sont reliés par une chaîne de documents. En conséquence, une méthode tolérante à l'effet de chaîne présentera deux groupes alors qu'une méthode non tolérante à l'effet de chaîne n'en présentera qu'un seul.

3.2.1.11. La tolérance aux groupes de tailles variées

Certaines méthodes permettent de ne produire que des groupes de tailles similaires. Cependant, certaines données peuvent donner lieu à l'élaboration de groupes de tailles différentes auquel cas il faut utiliser une méthode tolérante aux groupes de tailles variées.

3.2.1.12. La tolérance aux groupes de densités variées

De la même manière que pour la tolérance aux groupes de tailles variées, certaines méthodes ne peuvent produire que des groupes de densités similaires.

3.2.1.13. La tolérance aux groupes de forme quelconque

Encore une fois, certaines méthodes ne produisent que des groupes de formes similaires et d'autres permettent d'en faire abstraction dans leur génération.

3.2.1.14. La tolérance aux groupes concentriques

On parle de groupes concentriques lorsqu'un groupe est inscrit dans un autre. Là encore, une méthode tolérante aux groupes concentriques produira deux groupes alors qu'une méthode non tolérante aux groupes concentriques n'en produira qu'un seul.

3.2.2. Les méthodes de clustering

3.2.2.1. Les méthodes hiérarchiques

3.2.2.1.1. La méthode ascendante (spécification vers généralisation)

Au démarrage, chaque document est considéré comme un groupe puis les groupes sont regroupés selon des critères de similarité jusqu'à ce qu'il ne reste plus qu'un unique groupe à la racine de la hiérarchie.

3.2.2.1.2. La méthode descendante (généralisation vers spécification)

Au démarrage, tous les documents sont regroupés en un unique groupe puis le groupe est divisé successivement selon des critères de dissimilarité jusqu'à ce qu'on obtienne autant de groupes que de documents dans les feuilles de la hiérarchie.

3.2.2.2. Les méthodes par partition

3.2.2.2.1. La méthode statistique

Les données sont considérées comme ayant été générées en suivant une loi de distribution dont il faut déterminer les paramètres permettant d'associer les documents aux composantes de cette loi. Le clustering K-means, très largement utilisé, en est un cas particulier (Celeux et al., 1992), (MacKay, 2003).

3.2.2.2.2. La méthode stochastique

L'espace des solutions possibles est parcouru, la solution rencontrée qui optimise un critère donné est retenue. Il existe différentes heuristiques de parcours qui permettent de ne pas avoir à parcourir entièrement l'espace des solutions, chacune d'elles ayant ses caractéristiques propres (Berkhin, 2002). On notera la technique de la recherche Tabou, celle du recuit simulé et celle des algorithmes génétiques.

3.2.2.2.3. La méthode basée sur la densité

Chaque groupe est considéré comme un groupe de forte densité entouré par des régions de faibles densités. La méthode de référence est le DBSCAN (Density-Based Spatial Clustering of Applications with Noise) (Ester et al., 1996). Le point de départ est un document auquel les documents les plus proches selon un critère de densité sont associés pour obtenir un groupe. Cette phase est répétée sur un document qui n'appartient pas à un groupe jusqu'à ce qu'il n'y en ait plus.

3.2.2.2.4. La méthode basée sur les grilles

L'espace est partitionné en cellules grâce à une grille et chaque groupe de cellules denses connectées est identifié comme un groupe (Brezellec et al., 2001). Des travaux plus récents présentent un algorithme en deux phases, la première permettant d'effectuer un « pré-clustering » basé sur les grilles et la deuxième définissant les groupes par un apprentissage semi-supervisé (Wei et al, 2007).

3.2.2.2.5. La méthode basée sur les graphes

Chaque groupe est considéré comme un ensemble de nœuds connectés. Pour déterminer les groupes, le graphe complet des données dans lequel un nœud correspond à un document et un arc à la distance entre deux documents est d'abord construit. Le Minimal Spinning Tree en est ensuite dérivé, graphe connexe dont la somme des arcs, correspondant aux distances, est minimale et les arcs les plus longs (dont la valeur est supérieure à un seuil donné) sont supprimés.

3.2.2.2.6. La méthode spectrale

Le cœur du clustering spectral est le Laplacien de la matrice d'adjacence du graphe, qui représente les données, obtenu à partir de la formulation du problème de clustering des données sous forme d'un problème de coupe de graphe normalisée (Luxburg, 2007). Il existe deux types de clustering spectral, le normalisé et le non-normalisé. (Luxburg et al., 2004a) et (Luxburg et al, 2004b) prouvent qu'il est préférable de travailler sur le cas normalisé car la convergence vers une partition finie de l'ensemble des données dépend de moins de conditions que le cas non-normalisé.

3.2.2.3. Les méthodes hybrides

La méthode hiérarchique et la méthode par partition sont utilisées ensemble. Cela permet d'utiliser les points forts de chacun d'eux, la méthode hiérarchique étant plus souple que la méthode par partition qui elle est moins complexe en temps. On peut donc tirer avantage de chacune des deux méthodes. (Steinbach et al ., 2000) proposent une méthode consistant à partir d'un seul groupe composé de tous les documents, de le diviser selon un certain critère et de rechercher les deux sous- groupes en utilisant l'algorithme K-means. Les divisions sont arrêtées une fois que le nombre de groupes recherchés est obtenu. (Wang et al., 1997) proposent une autre méthode utilisant une méthode hiérarchique descendante combinée à une méthode basée sur les grilles. Ainsi, le point de départ est à nouveau un seul groupe composé de tous les documents. En descendant dans la hiérarchie, la résolution de la division de l'espace par les grilles est de plus en plus fine. (Karypis et al., 1999) proposent une méthode qui dans un premier temps crée un ensemble de petits groupes

grâce à une méthode de partition basée sur les graphes. Dans un second temps, ces groupes sont fusionnés en utilisant un algorithme hiérarchique ascendant.

3.2.2.4. Les méthodes subspace

L'idée de ces méthodes est le fait que tous les groupes à obtenir ne sont pas nécessairement définis suivant les mêmes dimensions. Ainsi, les méthodes de subspace clustering projettent les groupes dans différents sous-espaces (Parsons et al., 2004). Ces méthodes présentent souvent des chevauchements de groupes qu'il est intéressant d'analyser car ils peuvent n'être que de l'information redondante (Liu et al., 2004). Parmi les méthodes de subspace clustering existantes, on notera Tuareg (contexTUAlized clustERinG) qui utilise un algorithme descendant et un algorithme stochastique pour éviter les biais (Candillier et al., 2004), son amélioration SuSE (Subspace Selection embedded in an EM algorithm) qui considère que les connaissances a priori peuvent être vues comme un modèle de sélection dans un cadre probabiliste permettant une meilleure gestion du bruit (Candillier et al., 2006), SSC (Statistical Subspace Clustering) (Candillier et al., 2005) et DUSC (Dimensionality Unbiased Subspace Clustering) (Assent et al., 2007), une méthode de subspace clustering basé sur la densité ne nécessitant en entrée qu'une estimation du seuil de densité.

3.2.3. Caractéristiques des méthodes de clustering

Toutes les caractéristiques des méthodes présentées précédemment sont rappelées dans les tableaux suivants (Candillier, 2006) (Table 10 et Table 11). M est le nombre de dimensions qui décrivent les documents et N est le nombre de documents.

	Méthode hiérarchique	Méthode K-means	Méthode statistique	Méthode stochastique
Connaissance a priori	nombre de groupes ou seuils	nombre de groupes K	nombre de groupes K	nombre de groupes K
Présentation des résultats	hiérarchie	K centroïdes	K centroïdes et matrices de covariances	partition
Complexité	O(M x N²)	O(M x N x K)	O(M² x N x K)	O(M x N x K)
Déterministe	oui	non	non	non
Incrémentale	non	oui	oui	oui
Any-time	non	oui	oui	oui
Chevauchement	non	non	oui	non
Prise en compte du contexte	non	non	non	non
Tolérance :				
au bruit	non	non	oui	oui
à l'effet de chaîne	non	oui	oui	oui
aux groupes de tailles variées	oui	non	oui	oui
aux groupes de densités variées	oui	oui	oui	oui
aux groupes de forme quelconque	oui	non	non	non
aux groupes concentriques	oui	non	non	non

Table 10 – Caractéristiques des méthodes de clustering (1/2)

	Méthode basée sur la densité	**Méthode basée sur les grilles**	**Méthode basée sur les graphes**	**Méthode subspace**
Connaissance a priori	critère de densité du voisinage	taille de grille et critère de densité	distance minimale entre groupes	nombre maximal de groupes
Présentation des résultats	partition	ensemble de cellules connectées	partition sous forme de graphe	ensemble de règles
Complexité	O(M x N^2)	O(M x taille de grille)	O(M x N^2logN)	O(M x NlogN)
Déterministe	oui	oui	oui	non
Incrémentale	non	non	non	non
Any-time	non	non	non	oui
Chevauchement	non	non	non	non
Prise en compte du contexte	non	non	non	oui
Tolérance :				
au bruit	oui	oui	non	oui
à l'effet de chaîne	non	non	non	oui
aux groupes de tailles variées	oui	oui	oui	oui
aux groupes de densités variées	non	non	oui	oui
aux groupes de forme quelconque	oui	oui	oui	non
aux groupes concentriques	oui	oui	oui	non

Table 11 – Caractéristiques des méthodes de clustering (2/2)

3.2.4. Synthèse

Pour déterminer la méthode répondant le mieux au problème posé, il faut définir les caractéristiques auxquelles doit répondre la méthode à utiliser. En effet, le choix de la méthode de clustering est guidé par le besoin en termes d'usage et de données. Il faut donc évaluer les caractéristiques nécessaires et les prioriser. Une fois cette analyse effectuée, la méthode répondant le mieux au problème peut être déterminée. Ensuite, il est possible d'utiliser la méthode en l'état ou de la modifier pour améliorer certaines de ses caractéristiques.

Il apparaît toutefois impossible de déterminer une méthode dont les caractéristiques sont « idéales ». Par exemple, une méthode possédant toutes les caractéristiques de tolérance semble difficile à obtenir. En effet, en améliorant certaines caractéristiques, on en réduit nécessairement d'autres. Le poids que l'on porte aux caractéristiques (nécessaire, importante, mineure) joue donc un rôle très important.

3.3. Le raisonnement à base de cas (Case-Based Reasoning, CBR)

3.3.1. Présentation du CBR

C'est une approche de résolution de problèmes qui utilise des expériences passées pour résoudre de nouveaux problèmes (Leake, 1996), (Lamontagne et al., 2002). Cette approche est comparable au comportement humain. En effet, lorsque nous rencontrons un problème, il nous arrive de chercher une solution en le comparant aux problèmes similaires que nous avons déjà vécus. Nous cherchons donc à reproduire ou à adapter une expérience passée à partir des expériences que nous possédons. L'ensemble des expériences composent la base de cas, un cas étant composé de deux éléments : le problème et la solution.

Le cycle du CBR se déroule en 4 étapes : la recherche, la réutilisation, la révision et l'apprentissage (Figure 6). A partir d'un problème donné en entrée, il permet de retrouver les solutions qui ont déjà été utilisées pour des problèmes similaires et que l'utilisateur peut réutiliser. En sortie, il génère donc une solution qui répond au problème donné en entrée. Ce problème et cette solution deviennent naturellement un nouveau cas à intégrer dans la base de cas.

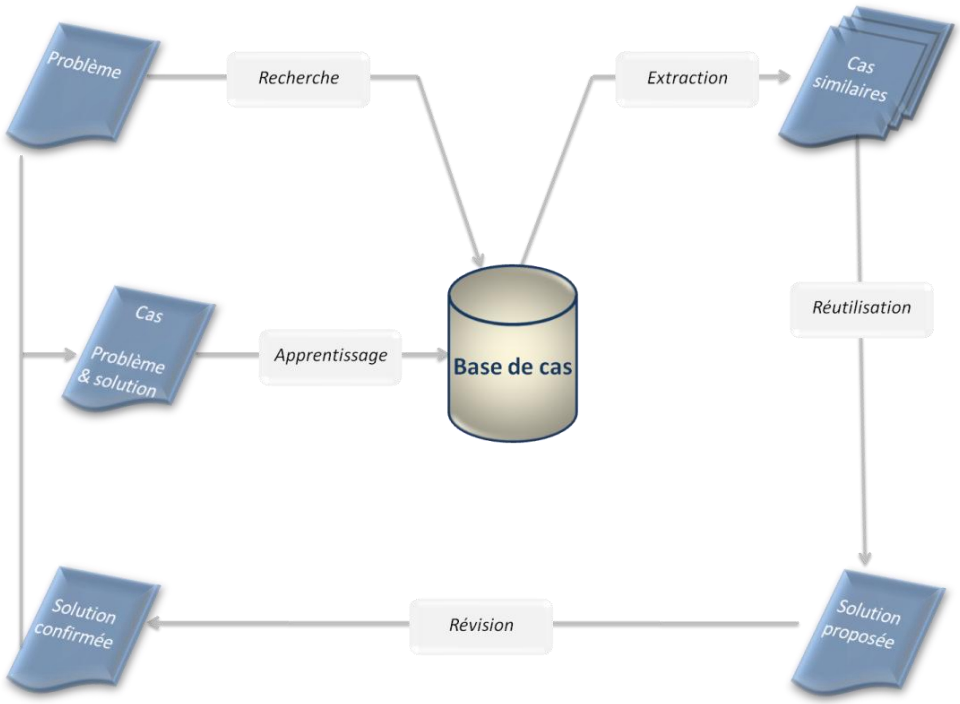


Figure 6 – Environnement CBR (adapté de (Aamodt et al., 1994))

3.3.2. Les étapes du CBR

3.3.2.1. La recherche

Cette étape totalement automatique permet de rechercher les cas de la base de cas dont le problème est similaire au problème donné en entrée. La technique de sélection des plus proches voisins est largement utilisée. Evidemment, elle implique l'utilisation d'une métrique de similarité. La similarité entre deux problèmes peut être déterminée de nombreuses façons. Cependant, les métriques de similarité sont difficilement hiérarchisables, la pertinence de leur utilisation dépend du domaine dans lequel elles sont placées.

3.3.2.2. La réutilisation

Pour résoudre le problème courant, il faut proposer une seule solution. L'étape de recherche peut retourner de nombreux cas similaires et donc autant de solutions envisageables. Une intervention humaine est donc obligatoire. Plusieurs possibilités s'offrent à l'utilisateur :

- Il peut choisir de réutiliser une solution existante sans effectuer de changement de celle-ci.

- Il peut aussi choisir de modifier une solution existante pour la rendre plus pertinente dans la résolution du problème courant.
- Il peut également choisir de créer une solution en réutilisant et en adaptant plusieurs solutions existantes.
- Il peut enfin choisir de ne pas utiliser les expériences passées et proposer une solution totalement nouvelle.

3.3.2.3. La révision

Encore une fois, cette étape ne peut pas être automatique, au mieux elle peut être semi-automatique, l'intervention humaine étant nécessaire. Elle consiste à tester la solution proposée en fonction du problème posé. On peut faire appel à un logiciel de simulation ou à un expert. Quelle que soit la méthode retenue, elle requiert une intervention humaine pour déterminer si la solution est pertinente ou si, au contraire, il faut la revoir et la modifier.

3.3.2.4. L'apprentissage

Lorsqu'une solution a été confirmée pour un problème donné, cette nouvelle expérience est considérée comme un nouveau cas et est donc naturellement ajoutée dans la base de cas. Ainsi, la base de cas n'est pas statique mais en perpétuelle évolution avec l'apport de tout nouveau cas résolu.

3.3.3. Synthèse

Le CBR est idéal pour la réutilisation d'expériences passées. Le principal problème réside dans le fait que plus la base de cas est volumineuse, plus le temps de réponse est long (De Mantaras et al., 2005). Pour pallier à cela, il est possible d'organiser les cas dans la base de cas et de ne pas seulement considérer cette base comme un « sac » de cas. L'utilisation d'index est une possibilité mais on peut aussi mettre en place des structures évoluées tels que les cas généralisés présentés dans travaux de R. Bergmann (Bergmann et al., 1999a), (Bergmann et al., 1999b). Cette structure organise les cas sous forme d'arbre dont le parcours pour la recherche permet de restreindre le nombre de cas comparés et donc diminue le temps d'exécution. En contrepartie, l'intégration d'un nouveau cas dans la base de cas demande un travail supplémentaire pour garder la base de cas organisée. Grâce à une structuration de la base de cas, il est donc tout à fait possible de diminuer le problème du passage à l'échelle. Cependant, l'interaction avec l'utilisateur paraît difficile car les groupes ne sont pas clairement définis. Nous pouvons aussi nous interroger sur la diversité des cas retournés, ou plutôt l'absence de diversité

(Mougouie et al., 2003), et sur l'application d'un tel système en entreprise, la mise en place de la maintenance de la base de cas (Leake et al., 2001).

3.4. Les cartes auto-organisatrices (Self-Organizing Maps, SOM)

Les cartes auto-organisatrices utilisent un réseau neuronal non supervisé (Kohonen, 1982). Comme dans le cas du CBR, c'est une approche comparable au comportement humain, et plus particulier à celui du cerveau. A l'aide d'une grille, l'espace des objets, les documents, et discrétisé et au fur et à mesure de l'exécution de l'algorithme, les objets les plus similaires sont les plus proches. Elle propose une visualisation finale des éléments et des groupes d'éléments (généralement en deux dimensions) permettant une analyse aisée. C'est une méthode comparable à la méthode de clustering K-means (Bacão et al., 2005) qui n'est pas incrémentale par défaut mais des études tendent à y remédier (Merkl et al., 2003).

3.5. L'analyse formelle de concepts (ou treillis de Galois)

C'est une méthode qui permet notamment une visualisation des éléments sous forme graphique dont les premiers travaux ont été effectués par R. Wille (Wille, 1982). L'Analyse Formelle de Concepts (AFC) permet une analyse qualitative des données (Ganter et al., 1999), c'est un outil de classification qui permet de recenser des objets, les documents, et des propriétés et ainsi de déterminer des corrélations entre eux : relation entre objets, relation entre propriétés, relation entre objets et propriétés. L'opérateur de Galois permet de construire des règles inductives (Guigues et al., 1986), c'est-à-dire une généralisation des informations (faits et règles) et ainsi révéler des liens de causalités syntaxiques. Des travaux relatifs aux liens causaux entre propriétés, appelés Analyse Formelle de Règles (AFR) ont aussi été menés (Chaudron et al. 2002).

Les informations nécessaires à l'élaboration d'un treillis de Galois doivent être de nature exclusivement symbolique, toutes les données numériques devant être discrétisées. Il existe des outils de transformation de données numériques en données symboliques. La plupart utilise une méthode de partage d'intervalles, soit celle où l'on compose des groupes en fonction de seuils établis, soit celle produisant des groupes de taille égale.

L'AFC permet de traiter une matrice binaire appelée contexte, dont les lignes sont indexées par des objets (observations, patients, environnements, incidents…) et dont les colonnes sont indexées par les propriétés potentielles

de ces objets (Table 12). Formellement, un contexte est un triplet C = (O, P, ζ) où O = {objets}, P = {propriétés} et ζ : O → P(P). Le traitement de ce contexte passe par l'usage de l'opérateur de classification dit opérateur de Galois.

	Propriété A	Propriété B	Propriété C	Propriété D	Propriété E	Propriété F
Objet 1	x		x		x	
Objet 2				x	x	
Objet 3			x		x	x
Objet 4	x		x		x	

Table 12 – Exemple de matrice binaire utilisée pour la construction du treillis de Galois

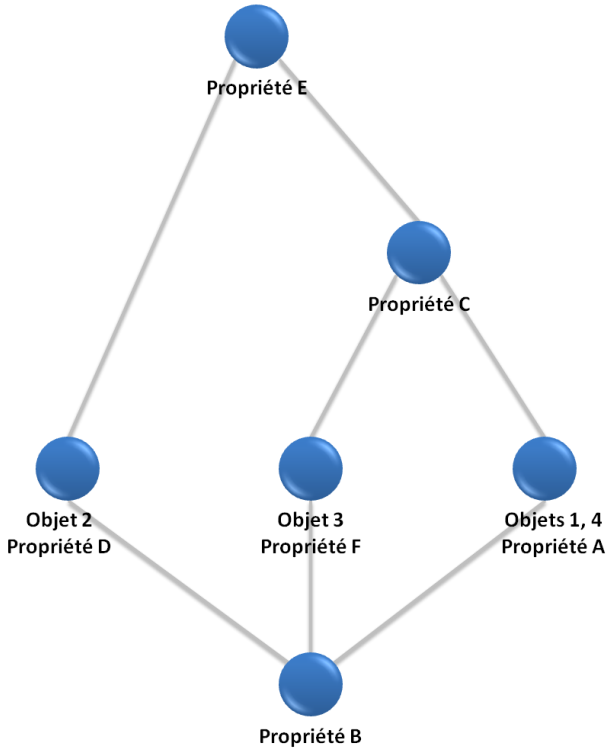


Figure 7 – Exemple de treillis de Galois

L'opérateur de Galois, noté ' sur O × P est une application duale telle que ' : P(O) ↔ P(P) et ∀A ∈ P(O), ∀B ∈ P(P), A' = {p ∈ P | (∀o ∈ A) p ∈ ζ(o)} et B' = {o ∈ O | (∀p ∈ B) p ∈ ζ(o)}. Cet opérateur permet de construire des couples de « paquets significatifs » d'objets et de propriétés, appelés concepts (disques bleus en Figure 7). Un concept est un couple (A, B) tel que A ⊂ O, B ⊂ P et A' = B, B' = A. Un treillis est un graphe dont les concepts sont ordonnés avec A ∈ P(O), B ∈ P(P), où A est appelé extension et B l'intension du concept, et A' = B et B' = A. Une fois cet ensemble de concepts produit, on munit l'ensemble des extensions de la relation d'inclusion, qui engendre un

ordre partiel. La représentation de cet ordre partiel est un treillis de Galois. Les concepts vérifient la propriété suivante : (A1, B1) << (A2, B2) $\leftrightarrow$ A1 $\subset$ A2. Un treillis sur lequel on applique un algorithme d'étiquetage devient lisible (étiquettes sur les concepts en Figure 7).

Les groupes obtenus sont définis de manière très rigide grâce au modèle mathématique. Cet environnement reste un peu simple pour notre utilisation. Il est peu robuste et peu enclin au changement. De plus, il est difficile de générer des groupes faisant émerger la connaissance. Les données sont seulement hiérarchisées, tous les liens entre elles sont conservés mais aucune indication quant aux différents groupes possibles n'est donnée. De plus, non seulement cet algorithme n'est pas incrémental mais aussi il possède une complexité exponentielle ce qui rend difficile le passage à l'échelle et la recherche en temps réel.

3.6. Les R-Tree

Une autre possibilité de classification est la représentation de la base de données sous forme d'arbre. Dans cette optique, ce n'est pas la définition des groupes qui est prépondérante mais le fait d'effectuer rapidement une recherche de similarité. L'organisation sous forme d'arbre permet de ne comparer qu'un ensemble restreint d'éléments en ne parcourant que certaines branches potentielles de l'arbre et en ne se préoccupant pas des branches que l'on sait inintéressantes.

L'algorithme R-tree et ses variantes (notamment R+-tree et R*-tree) répondent à cette volonté d'organisation. R+-tree se démarque de R-tree en réduisant les chevauchements, ce qui permet d'améliorer les performances de recherche. De son côté, R*-tree prend en compte les chevauchements lors de l'insertion et de la suppression d'un élément (Maindorfer et al., 2008). Grâce à une structure dynamique et l'utilisation d'heuristiques, le passage à l'échelle ne pose aucun souci. Des méthodes de recherche encore plus performantes sont d'ailleurs toujours d'actualité (Xiang et al., 2008).

Nous pouvons cependant nous interroger sur l'apport d'un système qui trouve tout son intérêt dans la gestion de données géographiques (Guttman, 1984) dans un domaine différent. De plus, la création initiale de l'arbre n'est pas triviale. Des techniques pour la faciliter utilisant des méthodes de clustering sont apparues : la méthode de clustering K-means (Brakatsoulas et al., 2002), les cartes auto-organisatrices (Choi et al., 2004), (Kun-Seok et al., 2001). Enfin, aucun groupe n'est défini, les éléments sont seulement organisés.

3.7. La classification par facettes (ou analytico-synthétique)

Il existe deux techniques pour créer un schéma de classification, la classification énumérative et la classification analytico-synthétique. La classification énumérative crée un schéma grâce à une approche top-down. Considérant un corpus de documents, celui-ci subit plusieurs divisions successives selon des sujets différents pour donner lieu à des sous-ensembles. Cela implique une forte subjectivité dans le choix des sujets choisis et demande un effort significatif pour la maintenance lors de l'ajout d'un nouveau sujet. A l'inverse, la classification analytico-synthétique produit un schéma grâce à une approche bottom-up. Celle-ci considère chaque document, extrait les sujets significatifs qui le composent et regroupe ces sujets extraits de tous les documents dans des catégories (ou facettes). De cette manière, un document est assimilé à une composition de sujets.

Le principe d'utilisation de la classification par facettes est simple. Chaque document (entité) du domaine peut être vu suivant différentes perspectives (facettes), chacune d'elles possédant une liste de valeurs (termes). Son utilisation trouve toute son efficacité sur un domaine spécifique avec un vocabulaire fermé. Les facettes doivent être mutuellement exclusives et la liste des termes de chacune d'elles doit être exhaustive. Un dictionnaire peut être associé à chaque terme pour prendre en compte les abréviations, les synonymes, etc. De même que les listes exhaustives de termes, la classification par facettes stricte impose qu'une entité ne possède au maximum qu'un terme pour chaque facette. Néanmoins, des évolutions ont été proposées. D'une part, il est possible de considérer une liste dynamique de termes dont la valeur pour chaque entité est issue de l'entité courante elle-même (field value) au lieu de ne considérer qu'une liste statique de termes dont les valeurs sont déterminées à l'avance (fixed value) (Mahoui et al., 2006). D'autre part, il n'y a pas de réelles contraintes à autoriser l'association de plusieurs termes d'une même facette pour une même entité même si T. Wilson rappelle que « ceci est paradoxal, controversable, et si vous adhérez à la théorie des facettes originale de S. R. Ranganathan, hérétique » (Wilson, 2009).

Le schéma de classification par facettes est construit selon un processus appelé Literary Warrant. Il consiste à sélectionner toutes les entités d'un domaine si leur nombre est assez restreint ou un échantillon d'entités représentatif du domaine si leur nombre est trop important, extraire les termes propres à chacune de ces entités (souvent à partir de leur titre dont les concepts extraits peuvent être considérés comme significatifs de l'entité) et créer les facettes en regroupant les termes proches, les termes que l'on peut intégrer sous une même catégorie. A cela s'ajoute un travail sur les termes de chaque facette : un listing exhaustif, un choix de représentation (plate, hiérarchique, ligne de temps, …) et un choix de tri (alphabétique par exemple). Des identifiants (sous forme de

chiffres ou de lettres) peuvent être associés à chaque facette et à chaque terme pour les désigner (travaux d'A. Slavic (Slavic, 1998) et de V. Broughton (Broughton et al., 2007)). Une description détaillée de toutes ces étapes est donné par J. Mills (Mills, 2004).

Pour illustrer ce processus, utilisons l'exemple de W. Denton (Denton, 2009) sur la classification de produits vaisselle présents dans un grand centre commercial. Comme il le fait remarquer, ce choix est tout à fait approprié car le domaine est familier, assez petit et son langage restreint. Il a donc relevé les caractéristiques de chacun des produits et relève une classification sur la présentation des produits. Ceux-ci sont organisés selon leur forme (Table 13).

A partir de cet échantillon, il propose une classification par facettes orientée clients (des chimistes ou des ingénieurs auraient des besoins différents) en utilisant six facettes (Table 14).

Une autre illustration, disponible sur Internet et utilisable pour tester la classification par facettes est FacetMap (Wilson, 2009). FacetMap propose une classification des vins selon trois facettes : la catégorie, la région et le prix. A noter que l'on voit bien ici une représentation spécifique du prix selon une ligne de prix (du moins cher au plus cher). La classification par facettes est présente dans de nombreuses applications sur Internet, 69% des sites de e-commerce considérés par H. Adkisson (Adkisson, 2003). On la retrouve aussi dans la réutilisation logicielle (Mili et al., 1997) avec les travaux de R. Prieto-Diaz (Prieto-Diaz et al., 1987), (Prieto-Diaz, 1991), (Prieto-Diaz, 2003).

Elle a aussi été présentée récemment dans la gestion des anomalies industrielles dans l'aérospatial (Goh et al., 2009). Son utilisation permet de trouver des problèmes similaires et des modèles de problèmes et de mettre en place un retour d'expérience entre la maintenance et l'étape de conception. En revanche, la recherche d'anomalies similaires est manuelle, l'utilisateur doit naviguer dans les facettes et le classement des anomalies retournées n'est pas pris en compte. Les auteurs se sont intéressés à deux problèmes : la combinaison de termes ne renvoyant pas de résultats et l'aide à la détection de facettes.

Pour résoudre le premier point, ils ont mis en place un système appelé Adaptative Concept Matching dont le principe est de partir du corpus de documents que l'on diminue au fur et à mesure des sélections de termes dans les facettes par l'utilisateur. Dans le même temps, les termes des facettes ne correspondant plus à aucun document en cours de sélection sont supprimés. En conséquence, quelque soit la sélection de termes effectuée par l'utilisateur, au moins un résultat doit pouvoir être affiché.

Liquide pour lave-vaisselle	
Cascade pure rinse formula	No name lemon gel
Electrasol lemon gel	Palmolive spring blossom gel
En poudre pour lave-vaisselle	
Cascade complete	No name premium formula
Cascade, fresh scent	No name premium formula, lemon-scented
Electrasol double action, fresh scent	Sunlight lemon-fresh
En gel	
Electrasol gelpacs, orange blossom	
En tablettes	
Electrasol tabs	No name premium formula tablets
Liquide pour lavage à main	
Ivory classic	Palmolive antibacterial
No name liquid dish detergent, lemon-scented	President's choice antibacterial hand soap & dishwashing liquid
Palmolive aroma therapy, lavender and ylang-ylang	President's choice invigorating aroma therapy, passion flower
Palmolive aroma therapy, mandarin and green tea essence	President's choice relaxing aroma therapy, ruby red grapefruit
Palmolive original	President's choice tough on grease
Palmolive spring sensations, fresh green apple	Sunlight, antibacterial
Palmolive spring sensations, ocean breeze	Sunlight, lemon fresh
Palmolive spring sensations, orchard fresh	

Table 13 – Liste de produits vaisselle, exemple de (Denton, 2009)

Marque	*Forme*	*Agent*	*Effet sur l'agent*	*Odeur*	*Propriété spéciale*
Cascade	Gel	Lave-vaisselle	Relaxant	Brise de l'océan	Anti-bactérien
Electrasol	Liquide	Personne	Tonifiant	Citron	
Ivory	Poudre			Fleur d'oranger	
No Name	Tablette			Fleur de la passion	
Palmolive				Fraîcheur des fruits	
President's Choice				Lavande et ylang ylang	
Sunlight				Mandarine et thé vert	
				Pamplemousse	
				Pomme verte	

Table 14 – Classification par facettes de produits vaisselle (Denton, 2009)

De la même manière que l'outil DARE (Domain Analysis and Reuse Environment) (Frakes et al., 1998) et (Prieto-Diaz, 2003) propose une aide à la détection de facettes en combinant une approche bottom-up avec une approche top-down, les auteurs veulent aussi faciliter la création des facettes. Ils s'intéressent particulièrement aux informations non structurées, sous forme de texte libre. Ils proposent d'extraire les mots de ces informations textuelles libres (en supprimant les mots stop) et d'utiliser des méthodes statistiques pour analyser la fréquence de ces mots et extraire les groupes de mots les plus fréquents. Ainsi, des informations non structurées peuvent conduire à la création de facettes mais cela ne fait que proposer des concepts à l'expert dont l'intervention est toujours essentielle pour déterminer des facettes ayant du sens. Cependant, la classification des documents suivant des facettes issues d'informations textuelles libres est beaucoup moins efficace que celles suivant des facettes issues d'informations structurées. Sur l'exemple donné par les auteurs, exemple comportant neuf facettes, cinq sont issues d'informations structurées dont quatre permettent chacune une classification supérieure ou égale à 90% des documents alors que parmi les quatre issues d'informations non structurées, une seule permet la classification de plus de 50% des documents, les autres n'en permettant même pas le tiers chacune.

Les avantages de la classification par facettes sont donc nombreux. L'ensemble de la communauté scientifique semble d'accord sur ces avantages rappelés par V. Broughton (Broughton, 2006). La classification par facettes permet d'avoir

de nombreuses possibilités de vues sur les entités du domaine. Ainsi, elles sont adaptées à différentes catégories de personnes ayant des besoins différents. Certaines vont s'intéresser à une certaine facette, d'autres à une autre, d'autres encore à la combinaison de deux facettes. Chacune pourra avoir la vision qu'elle veut de l'entité en fonction de ses besoins sans être contrainte par une vision unique à suivre. La structure logique de la classification par facettes est compatible avec une manipulation automatique et permet une maintenance facile. Enfin, elle possède un pouvoir explicatif puissant et permet la mise en place d'interfaces graphiques simples pour la navigation, grâce au schéma de classification mis en place, et pour la recherche, en sélectionnant des termes dans les facettes ou en saisissant des mots-clés dans un champ libre qui sont ensuite comparés aux termes des facettes.

En revanche, la classification par facettes souffre de certaines difficultés. D'abord, il n'est pas toujours évident de déterminer les facettes et plus précisément les bonnes facettes. Rien ne permet de dire si une facette est pertinente hormis la connaissance d'un expert du domaine. (Giess et al., 2007) et (Giess et al., 2008) s'interrogent sur les exemples proposant une classification par facettes car ils sont très simples (à l'image des produits vaisselle et des vins) et les facettes identifiables à la lecture ne sont pas nécessairement celles qui sont les plus utiles et les plus descriptives. Ensuite, la création du schéma de classification demande un effort humain important. Des solutions ont été proposées pour aider à déterminer les facettes mais aucun outil ne peut être entièrement automatique, l'intervention de l'expert reste cruciale. Toutefois, ce travail initial permet d'avoir par la suite une maintenance facile malgré l'apport de nouvelles informations clés identifiées. Par ailleurs, ce schéma est en évolution continue car il ne peut pas être optimal dès le premier essai et il doit être raffiné.

3.8. Métriques d'évaluation

3.8.1. La matrice de confusion

Afin de mesurer la qualité d'un algorithme de classification, différentes mesures sont couramment utilisées. Nous allons en présenter quelques-unes ici, les mesures rappel-précision, ROC (Receiver Operating Characteristic), d'exactitude (accuracy en anglais) et la F-mesure. Toutes ces mesures sont basées et calculées grâce à la matrice de confusion (Table 15). Cette matrice contient en ligne les informations de classification idéale, que l'on voudrait obtenir, qui sert de référence et en colonne les informations de classification effectivement obtenues par un algorithme.

	Documents classés positifs	***Documents classés négatifs***
Documents réellement positifs	TP (True Positive)	FN (False Negative)
Documents réellement négatifs	FP (False Positive)	TN (True Negative)

Table 15 – Matrice de confusion

Où :

- TP est le nombre de documents que l'algorithme a correctement classés de manière positive. Ce sont les documents que la classification de référence et la classification par l'algorithme font appartenir à la classe.
- TN est le nombre de documents que l'algorithme a correctement classés de manière négative. Ce sont les documents que ni la classification de référence ni la classification par l'algorithme ne font appartenir à la classe.
- FP est le nombre de documents que l'algorithme a classés à tort de manière positive. Ce sont les documents que la classification de référence ne fait pas appartenir à la classe alors que la classification par l'algorithme les y fait appartenir.
- FN est le nombre de documents que l'algorithme a classés à tort de manière négative. Ce sont les documents que la classification de référence fait appartenir à la classe alors que la classification par l'algorithme ne les y fait pas appartenir.

Une classification parfaite est obtenue lorsque ces deux derniers, FP et FN, sont égaux à 0. Dans ce cas, l'algorithme a correctement classé tous les documents de manière positive et négative.

3.8.2. La mesure rappel-précision

Cette mesure calcule la proportion de documents correctement classés positifs parmi tous les documents réellement positifs de la classification de référence, le rappel, par rapport à la proportion de documents correctement classés positifs par tous les documents classés positifs par l'algorithme, la précision.

Le rappel se calcule de la manière suivante :

$$Rappel = \frac{TP}{TP + FN} \quad (7)$$

La précision se calcule de la manière suivante :

$$Précision = \frac{TP}{TP + FP} \quad (8)$$

Cette mesure se représente sous la forme d'une courbe avec en abscisse le rappel et en ordonnée la précision. Les valeurs obtenues sont comprises entre 0 et 1. Un algorithme parfait fournit un classement dont le rappel et la précision sont égaux à 1, tous les documents qui doivent appartenir à la classe y appartiennent (le rappel) et seulement ceux qui doivent y appartenir y appartiennent (la précision) (Figure 8). Lorsque le rappel est égal à 1, on parle de système exhaustif. Lorsque la précision est égale à 1, on parle de système sélectif. Toutefois, une classification parfaite est difficilement atteignable, les algorithmes fournissent des résultats plus ou moins pertinents et plus ou moins précis. Même si théoriquement le rappel et la précision sont indépendants, il est courant que lorsque l'on cherche à augmenter la précision, le rappel diminue (le silence augmente (silence = 1 – rappel)) et inversement, lorsque l'on cherche à augmenter le rappel, c'est la précision qui diminue (le bruit augmente (bruit = 1 – rappel)).

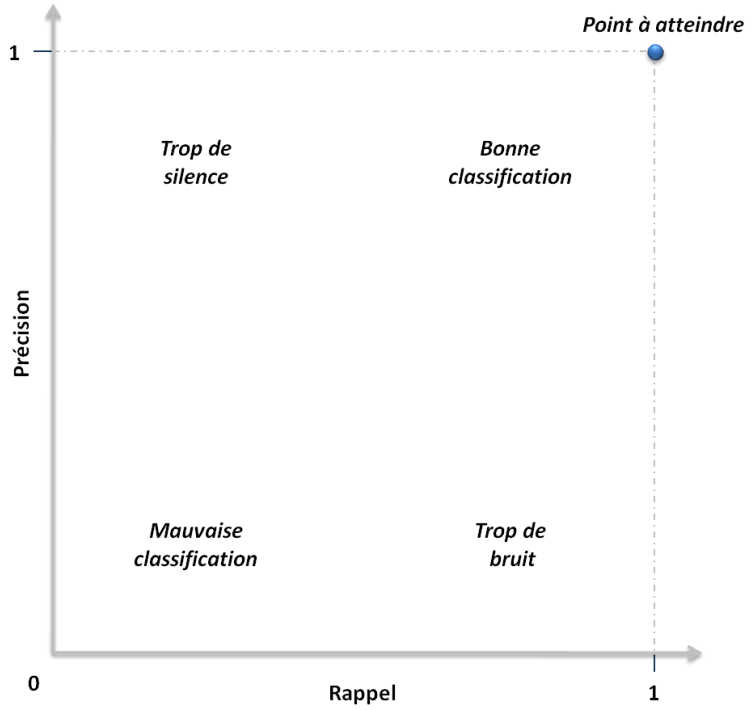


Figure 8 – Mesure rappel-précision

3.8.3. La F-mesure

L'objectif de la mesure rappel-précision est de maximiser le rappel et la précision. Cependant, il n'est pas toujours évident de déterminer la qualité d'un algorithme de classification par rapport à un autre à l'aide de cette mesure. Certains vont avoir un rappel plus faible mais une précision plus élevée que

d'autres et inversement. La F-mesure est une mesure qui pondère le rappel et la précision et fournit une seule valeur de qualité de classification d'un l'algorithme ce qui permet une comparaison entre algorithmes beaucoup plus aisée.

La F-mesure se calcule de la manière suivante :

$$Fmesure = \frac{2 * précision * rappel}{précision + rappel}$$
ou
$$Fmesure = \frac{2 * TP}{2 * TP + FP + FN} \quad (9)$$

Comme pour le rappel et la précision, sa valeur est comprise entre 0 et 1. Plus le rappel et la précision sont élevés, plus la F-mesure est élevée. Pour une classification parfaite, précision = rappel = 1 (ou FP = FN = 0), et F-mesure = 1 aussi.

Cette formule est un cas particulier d'une formule générale dans lequel l'importance accordée à la précision et au rappel est la même. La formule générale est la suivante :

$$F_\beta = \frac{(1 + \beta^2) * précision * rappel}{(\beta^2 * précision) + rappel}, \ \beta \in [0; +\infty[\quad (10)$$

Le paramètre β permet de moduler l'importance accordée au rappel et à la précision. On retrouve bien le cas particulier de la F-mesure pour lequel $\beta = 1$. En revanche, lorsque $\beta < 1$, F_β est plus orientée précision et lorsque $\beta > 1$, F_β est plus orientée rappel.

3.8.4. La mesure d'exactitude

La mesure d'exactitude détermine la proportion de documents correctement classés, les vrais positifs et les vrais négatifs, parmi l'ensemble des documents considérés. De la même manière que la mesure rappel-précision, elle se base sur la matrice de confusion et, comme la f-mesure, elle fournit une seule valeur de qualité de classification d'un algorithme comprise entre 0 et 1.

La mesure d'exactitude se calcule de la manière suivante :

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \quad (11)$$

3.8.5. La mesure ROC

Une autre mesure pour évaluer la qualité de classification d'un algorithme est la mesure ROC (Receiver Operating Characteristic). Tout comme la mesure rappel-précision, elle se base sur la matrice de confusion et tout comme la F-

mesure, cependant, le couple de valeurs comprises entre 0 et 1 permet de comparer aisément les classifications obtenues par les différents algorithmes. Graphiquement, la mesure ROC se représente sous la forme d'une courbe ayant pour abscisse le taux de documents que l'algorithme a classés à tort de manière positive, le taux de faux positifs (FPR : False Positive Rate) et pour ordonnée le taux de documents que l'algorithme a correctement classés de manière positive, le taux de vrais positifs (TPR : True Positive Rate) (Figure 9).

Le taux de faux positifs se calcule de la manière suivante :

$$FPR = \frac{FP}{FP + TN} \quad \textbf{(12)}$$

Le taux de vrais positifs se calcule de la manière suivante :

$$TPR = \frac{TP}{TP + FN} = Rappel \quad \textbf{(13)}$$

Un algorithme inutile car donnant une classification aléatoire est la ligne y = x (TPR = FPR). La courbe d'un algorithme parfait passe par le point (0,1) (TPR = 1 et FPR = 0). Une courbe située en-dessous de la ligne de classification aléatoire signifie que l'algorithme effectue une classification pire qu'une classification aléatoire.

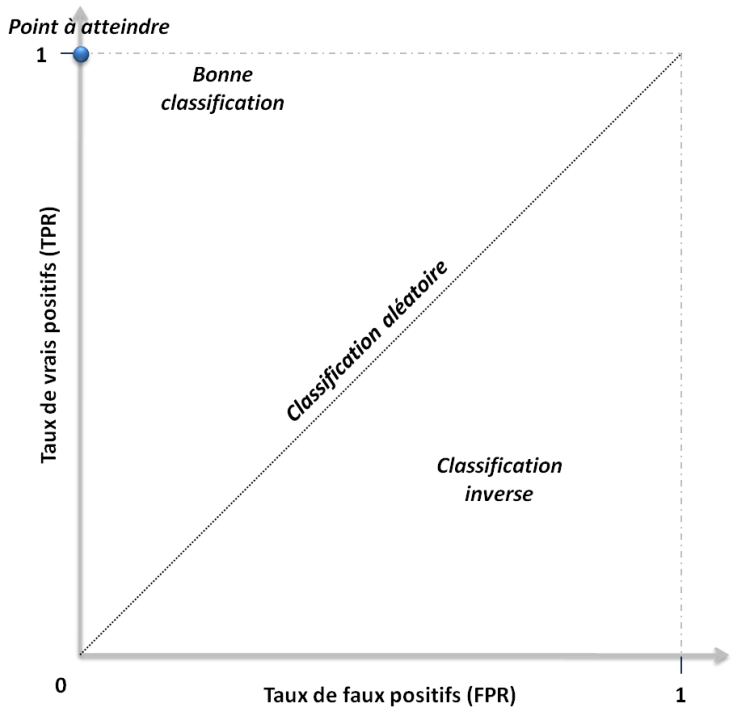


Figure 9 – Mesure ROC

3.9. Conclusion

Ce chapitre présentait les différents algorithmes de classification non supervisée, les environnements de regroupement bien connus dans la littérature. Les méthodes de clustering sont nombreuses et variées mais aucune n'est optimale, certaines forces sont contrebalancées par certaines faiblesses. Le choix de la méthode est donc vraiment dépendant de l'usage qui doit en être fait. Ce choix est d'autant plus large que nous considérons d'autres algorithmes de classification qui peuvent compenser les lacunes de certaines méthodes de clustering. Cependant, nous pouvons déjà nous rendre compte, par exemple, des problèmes de performance et de passage à l'échelle de certains algorithmes, de clustering ou non. Le choix d'un environnement de regroupement, d'un algorithme précis doit donc être appliqué judicieusement en fonction des besoins et des objectifs.

Ce choix est essentiel pour l'étape de fouille de données (étape 4) du processus de KDD puisque celle-ci concerne l'exécution d'un algorithme de classification. Il dépend de la manière dont les données sont transmises en entrée, des caractéristiques propres que doit avoir l'algorithme lors de son exécution et de la manière dont les données sont agrégées en sortie. Le modèle de descripteur de fiche d'anomalie présenté dans le chapitre suivant est un premier élément qui va guider ce choix. D'autres éléments tels que le passage à l'échelle ou la pertinence des groupes sont considérés dans le chapitre 5.

Chapitre 4

Proposition d'un modèle de descripteur de fiche d'anomalie

4.1. Introduction

Le traitement d'exceptions que nous situons dans le domaine particulier de la production industrielle nous conduit naturellement à nous interroger sur la structure et le contenu des documents qui servent de support au signalement, à l'identification et à la résolution de ces exceptions, documents que nous appelons fiches d'anomalie. Comme nous allons le développer dans ce chapitre, une anomalie est un événement qui sert de déclencheur à un processus de résolution. Les informations relatives à l'anomalie sont stockées dans sa fiche. Une fois le processus de résolution exécutée, c'est-à-dire qu'une solution a été appliquée ou qu'aucune solution n'était nécessaire (anomalie négligeable), l'anomalie disparaît mais sa fiche est conservée, d'une part pour des besoins de traçabilité et d'autre part parce que cette fiche est porteuse de connaissances à capitaliser. Le caractère restreint et spécifique du domaine laisse à penser qu'il est possible de réaliser une modélisation unique de toute fiche d'anomalie qui soit indépendante de la nature de l'anomalie et du processus de production industrielle.

Ce chapitre présente notre vision du processus de résolution d'anomalie basée sur l'expérience industrielle. A partir des éléments qui entrent en jeu, nous définissons la structure d'une fiche d'anomalie et décrivons les composants génériques qui la constituent et qui contiennent les informations saisies par les utilisateurs. Le modèle de descripteur de fiche d'anomalie que nous proposons découle de cette étude. Il met en évidence les critères de regroupement entre fiches d'anomalie ce qui permet la recherche de descripteurs liés en vue de la correction et de la prévention d'anomalie. Pour valider ce modèle, nous considérons deux bases de plusieurs milliers de fiches et montrons que le modèle peut s'y appliquer puis nous le comparons au modèle de document de la base de bugs logiciels BugZilla.

4.2. Le processus de résolution d'anomalie

Lorsqu'au cours d'un processus de production industrielle, un écart est constaté entre ce qui est spécifié par ce processus et ce qui est effectivement produit, une anomalie est déclarée pour matérialiser cet écart et tenter de le résorber. A partir de là, le processus de résolution de l'anomalie est initié. Ce processus permet la collecte d'informations sur l'anomalie, informations relatives au problème relevé et relatives à la solution mise en place pour le résoudre.

Ces informations sont collectées dans un document, que nous appelons *fiche d'anomalie*, qui sert de support à l'anomalie. Qu'elle soit électronique ou papier, cette fiche est composée d'éléments qui stockent les données saisies par les utilisateurs. Même dans un domaine restreint, plusieurs types d'anomalies peuvent apparaître : documentaires, électriques et mécaniques par exemple. Pour traiter les spécificités de chacun de ces types d'anomalie, les fiches de description sont hétérogènes en structure. Par conséquent, plusieurs *patrons de fiche d'anomalie* (templates) qui référencent les différentes structures possibles peuvent exister ; chaque fiche d'anomalie est donc une instance de l'un d'eux. Ces patrons ne doivent pas être rapprochés des patterns exposés dans les chapitres précédents. Alors qu'un pattern représente un modèle dans les données, une solution liée à un problème, un patron de fiche d'anomalie représente un modèle de formulaire de saisie des informations relative à une anomalie. Son instanciation donne lieu à la création d'une fiche d'anomalie.

A partir d'une expérience basée sur un processus industriel réel, nous représentons un processus de résolution d'anomalie par un processus initié par un déclencheur, la détection d'une anomalie. Au cours de ce processus, les données sur le problème et sur la solution à mettre en place sont renseignées dans la fiche d'anomalie instanciée à partir d'un patron de fiche d'anomalie. Lorsque la solution a été exécutée ou si l'écart est acceptable (aucune solution nécessaire), l'anomalie est clôturée et la fiche d'anomalie est stockée (Figure 10). L'anomalie disparaît, elle est considérée comme traitée. En revanche, la fiche qui lui sert de support et qui possède toutes les données relatives à l'anomalie est conservée.

Quelques détails sur les objets intervenant dans ce diagramme d'activités :

- *Anomalie détectée* : à ce stade, guidé par les procédures d'assurance-qualité, l'utilisateur sélectionne le patron de fiche utilisé pour résoudre l'anomalie. Nous considérons un ensemble de patrons de fiches car toutes les anomalies ne se résolvent pas de la même façon selon leur type. Plusieurs patrons sont ainsi définis pour permettre à l'utilisateur de choisir celui qui sera le plus approprié pour la résoudre.

- *Fiche d'anomalie créée* : c'est une instance d'un patron de fiche d'anomalie dans laquelle les données doivent être collectées.
- *Fiche d'anomalie dont le problème est renseigné* : la description du problème rencontré est renseignée. C'est à partir de ces informations que les systèmes d'aide à la résolution d'anomalie et les personnes en charge de cette résolution vont proposer une solution.
- *Fiche d'anomalie dont une solution au problème est renseignée* : la proposition d'une solution peut être réalisée en plusieurs fois, par plusieurs personnes (support à la fabrication, assurance-qualité, etc.) à des moments différents (peut s'étaler sur plusieurs jours voire plusieurs semaines). Toutefois, toutes les personnes possèdent une vision globale de l'anomalie. La fiche d'anomalie n'est pas un ensemble d'éléments cloisonnés que chacun d'eux doit renseigner sans se soucier des autres éléments mais bien une seule instance complétée au fur et à mesure de sa résolution par les différentes personnes. Si une solution n'est pas complète, le processus de collecte d'information sur la solution se poursuit. Si une solution n'est pas validée, le processus de collecte d'information sur la solution continue jusqu'à trouver une solution satisfaisante. Si la solution est complète et validée par un responsable, son exécution peut être lancée. La solution peut être de n'apporter aucune modification (écart acceptable).
- *Fiche d'anomalie dont la solution a été exécutée* : des informations sur le succès ou l'échec de l'exécution de la solution sont collectées.
- *Fiche d'anomalie archivée / Anomalie close* : le processus de résolution d'anomalie est terminé, l'anomalie est close, la fiche qui sert de support à cette anomalie est archivée. Cela permet de conserver la traçabilité des actions qui ont été effectuées. C'est aussi une source de connaissances à capitaliser pour identifier les patterns afin de faciliter les maintenances correctives et préventives.

La résolution d'une anomalie se termine par l'archivage de sa fiche. Sans capitalisation de l'information contenue dans les fiches d'anomalie, ce processus de résolution souffre de plusieurs problèmes :

- *Lenteur de traitement* : l'activité de collecte de données est entièrement manuelle, aucune aide n'est apportée à l'utilisateur pour le guider dans la résolution de l'anomalie, que ce soit dans le renseignement d'éléments concernant le problème ou ceux concernant sa solution. Le temps passé par un utilisateur pour chercher et proposer une solution pourrait être réduit en utilisant de manière semi-automatique la connaissance issue de l'expérience des anomalies passées. La solution la moins coûteuse en temps ou en argent pourrait être préférée pour résoudre un problème.

- *Coût de traitement* : il découle du point précédent, plus la résolution de l'anomalie prend de temps, plus le processus est coûteux puisqu'il bloque le processus de production industrielle sur lequel porte l'anomalie et parfois sur l'ensemble des étapes de fabrication du produit final. Déterminer en amont, au moment où l'anomalie est détectée, le cas où aucune modification n'est nécessaire car l'anomalie est négligeable permettrait une prise de décision rapide et réduirait le coût de traitement, le processus de résolution étant raccourci. Dans certains contextes, il pourrait en être de même pour les cas où il n'y a pas de solution rentable, où les solutions possibles à mettre en œuvre sont trop coûteuses.
- *Absence de création de connaissances* : aucune aide à l'analyse des fiches d'anomalie passées n'est réalisée. Aucune mesure pour comprendre pourquoi les anomalies surviennent n'est prise, ce qui permettrait d'éviter l'apparition d'anomalies récurrentes.

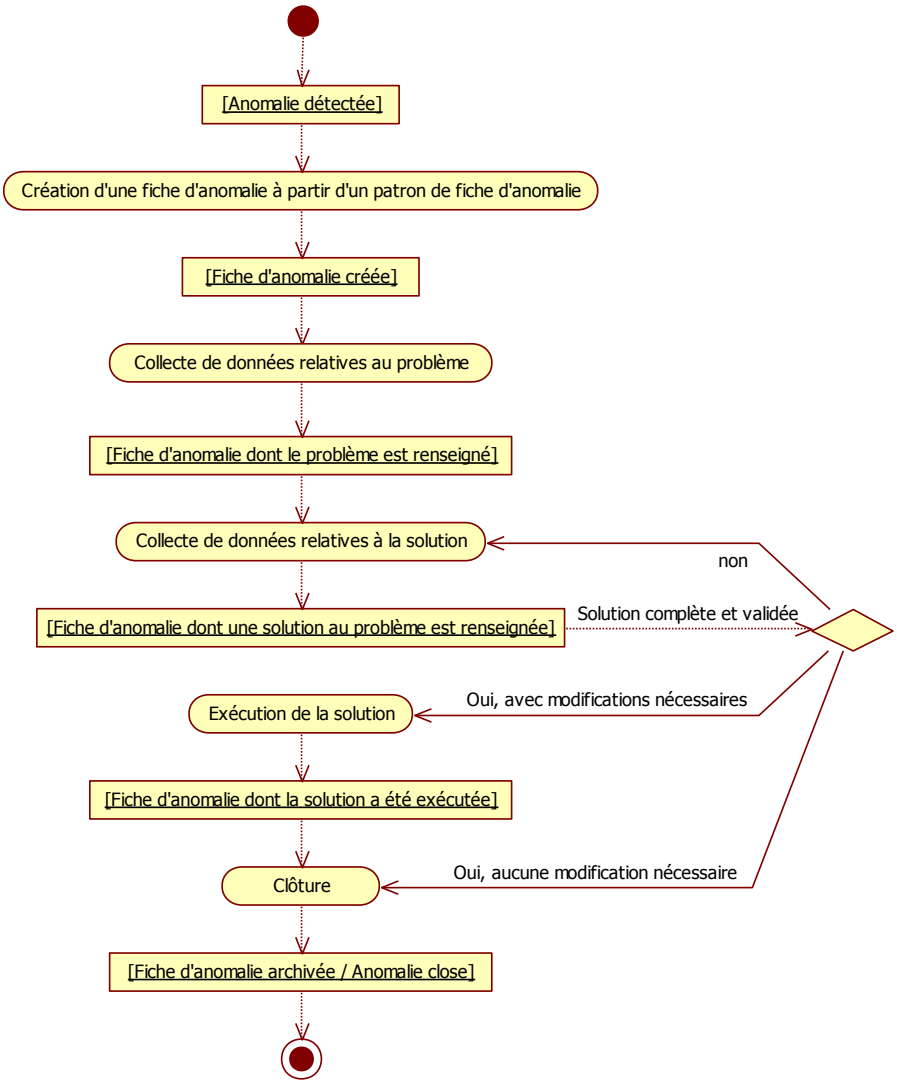


Figure 10 – Processus de résolution d'anomalie

4.3. Exemple de fiche d'anomalie

Considérons un processus de fabrication au cours duquel 10 trous sont percés dans une pièce métallique constituant l'aile d'un avion. Pendant le forage du 7ème, l'ouvrier casse son outil à l'intérieur, ce qui élargit le trou au-delà du diamètre prévu. A partir de ce moment, l'ouvrier déclare une anomalie donc crée une fiche d'anomalie dans laquelle il informe du problème rencontré (Figure 11).

Il attend ensuite les instructions du service chargé de résoudre les anomalies sur la solution à mettre en œuvre pour poursuivre l'exécution de ce processus de fabrication.

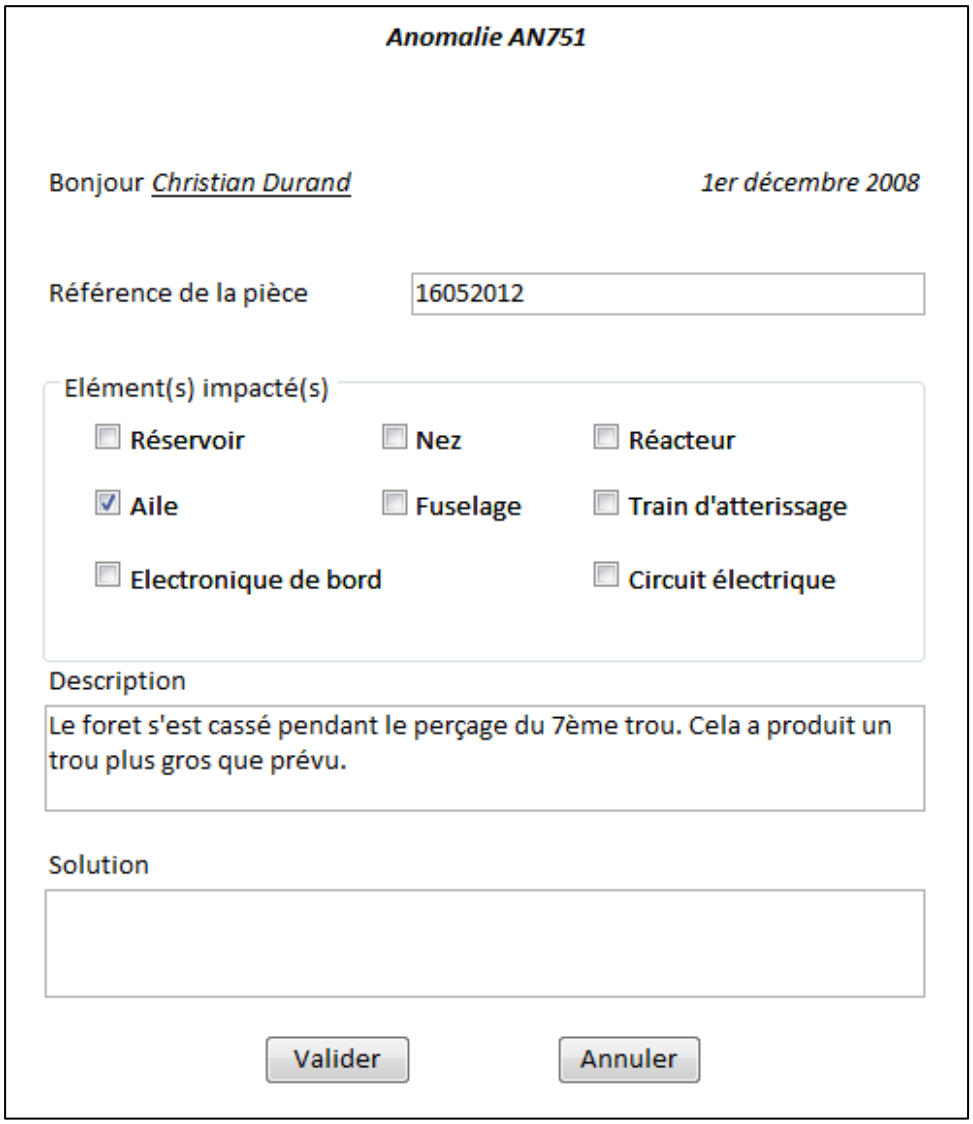


Figure 11 – Exemple de fiche d'anomalie

Deux heures plus tard, le service de support à la fabrication analyse cette anomalie. Typiquement, trois solutions sont envisageables.

Première solution : *utiliser le trou en l'état*.

Il s'agit d'une concession qui doit être validée par le bureau d'étude et qui sera partagée avec le client. Du fait des dizaines d'anomalies que le bureau d'étude reçoit tous les jours, cette validation n'est effectuée que plusieurs jours après la déclaration de l'anomalie. Le processus de fabrication est mis en attente jusqu'à ce que l'anomalie soit close. Retard prévu : 10 jours.

Deuxième solution : *réparer le trou*.

Différentes méthodes de correction standards peuvent être utilisées selon l'inspection complémentaire effectuée par le service de support à la fabrication. Ces nouvelles informations sont renseignées dans la fiche d'anomalie. Pour effectuer la réparation, il est possible de percer un trou plus grand pour le rendre plus propre (sans point de stress résiduel d'où pourrait partir une déchirure) et pour y mettre un rivet de taille supérieure. Il est aussi possible de combler le trou pour lui rendre un diamètre correct.

Quel que soit le type de réparation choisi, il nécessite une modification de conception. Evidemment, une telle réparation nécessite d'être détaillée dans la fiche d'anomalie pour définir précisément les étapes à suivre. Le processus de fabrication est mis en attente jusqu'à ce que l'anomalie soit close. Retard prévu : 20 jours.

Troisième solution : *mettre la pièce au rebut et en fabriquer une nouvelle*.

La simplicité de cette solution doit être contrebalancée avec le coût de mise au rebut plus celui du report de tous les processus de fabrication dépendant de la disponibilité de cette pièce d'aile d'avion. La décision est généralement prise dans les 3 à 5 jours par la direction de l'usine.

Dans tous les cas, le processus de fabrication, et donc la pièce en cours de fabrication, est mis en attente pour au moins 10 jours, et probablement davantage, ce qui retarde les autres processus de fabrication qui en sont dépendants. Si la fabrication ne peut pas attendre, et dans la mesure du possible, l'exécution du processus de fabrication et de ceux qui en sont dépendants peuvent se poursuivre avec une alarme sur la pièce pour indiquer qu'elle doit être corrigée avant le premier vol ou avant la livraison au client. Evidemment, cela a aussi un coût important : plus le problème est corrigé tardivement, plus le coût de correction est élevé.

4.4. Définition de fiche d'anomalie

4.4.1. Structure de fiche d'anomalie

Dans une organisation, toutes les anomalies ne se résolvent pas de la même manière selon leur type. Plusieurs patrons de fiches d'anomalie sont ainsi définis pour utiliser le plus approprié. Ils impliquent des éléments spécifiques à chacun d'eux mais aussi d'autres éléments qui, au contraire, peuvent être utilisés dans plusieurs patrons. Considérant les anomalies nominales (i.e. celles que l'on peut prévoir), les fiches mettent en jeu des éléments que l'on peut qualifier de réguliers. En effet, chacun d'eux est nommé distinctement et les données qu'un utilisateur y renseigne sont clairement identifiables. Certains de ces éléments définissent des données structurées, dont la liste de valeurs est finie, et d'autres définissent des données non-structurées, comme un texte libre. Considérant les anomalies non-nominales (i.e. celles qui ne peuvent pas être prévues), la fiche met en jeu des éléments non-réguliers. Ces éléments sont, par exemple, des commentaires ou une description de la première occurrence d'un nouveau type d'anomalie. Ils définissent des données non-structurées. En conséquence, une fiche d'anomalie est un document structuré qui met en jeu divers éléments contenant des informations structurées et non-structurées. Pour générer de la connaissance à partir des informations contenues dans les fiches d'anomalie, une modélisation de fiche d'anomalie est nécessaire afin d'exploiter au mieux toute l'information structurée et non-structurée qui y est présente.

Dans la littérature relative à la maintenance, nous n'avons pas relevé de travaux qui s'intéressent à une telle modélisation, qui permettrait d'exploiter les informations relatives aux anomalies. Ce domaine s'intéresse principalement à la formalisation du processus de maintenance, qui optimise la résolution des anomalies. En maintenance industrielle, (Waeyenbergh et al., 2004) proposent un cadre, CIBOCOF, pour la gestion des problèmes de maintenance. Une approche fonctionnelle est présentée par (Despujols, 2004) où les différentes fonctions de maintenance sont décrites : les études, la préparation des travaux, l'ordonnancement, la réalisation, l'approvisionnement des matières premières et des prestations extérieures. Le raisonnement TPM (Total Productive Maintenance) est souvent cité comme une solution (Ahuja et al., 2008). Il implique toutes les personnes à différents niveaux de maintenance dans le but d'optimiser l'utilisation de chaque outil de fabrication. En maintenance logicielle, tous les travaux convergent vers un modèle de processus comparable (Haziza et al., 1992), (Lambolez, 1994), (Barros, 1997), (Alloui, 2009) dont les principales étapes sont : 1) l'analyse et compréhension du logiciel, 2) le choix d'une solution ou rejet de l'intervention, 3) la mise en

œuvre de la solution et 4) la clôture de l'intervention. Bien que la formalisation du processus de maintenance conduise à faciliter la résolution de l'anomalie, elle ne définit pas comment en exploiter les informations que les fiches d'anomalie seraient susceptibles de contenir.

Nous pensons qu'une telle modélisation est une réelle contribution pour optimiser le processus de maintenance sur la durée en exploitant au mieux les informations capitalisées dans les bases de fiches d'anomalie. Cette modélisation doit être suffisamment générique pour prendre en compte tous les patrons de fiches d'anomalie, qu'il s'agisse de traiter des anomalies nominales ou non-nominales. Il ne s'agit pas de proposer une modélisation générique applicable à tout domaine et tout contexte : nous souhaitons nous focaliser sur des objectifs de maintenance corrective et préventive. Il est donc nécessaire de proposer un modèle de descripteur de fiche d'anomalie pour décrire la représentation de son contenu informationnel et sa structure. Cela consiste à représenter toute fiche d'anomalie par l'ensemble des informations qui la caractérisent, son descripteur, i.e. l'ensemble de mots-clés extraits des données contenues dans les éléments qui la composent. Le fait de travailler sur les descripteurs plutôt que sur les documents eux-mêmes permet une comparaison de leur contenu pour évaluer la similarité entre deux documents. Ainsi, avec un modèle qui permet d'agréger les informations, il est possible de trouver les documents pertinents à une requête (qui peut être exprimée sous forme d'un document), donc de trouver des fiches d'anomalie similaires à une nouvelle fiche d'anomalie. Bien que nous nous intéressions au contenu, il est aussi important de considérer la structure des fiches d'anomalie pour extraire les mots-clés de manière pertinente et effectuer des comparaisons adaptées entre descripteurs.

4.4.2. Description des éléments constitutifs de fiche d'anomalie

La fiche d'anomalie contient des données sur les caractéristiques de l'incident observé (i.e. le problème) et sur le protocole appliqué pour le résoudre (i.e. la solution). Il met en jeu des *attributs* dans lesquels les données sont renseignées. Nous définissons ces attributs selon deux propriétés, leur *structuration* et leur *qualité descriptive* (Table 16).

La propriété de *structuration* partitionne les attributs entre attributs *contraints* et attributs *libres*. Les attributs *contraints* permettent de renseigner des informations précises, structurées. L'ensemble des valeurs que peuvent prendre ces attributs peut être connu à l'avance ou non. Même si cet ensemble n'est pas défini à l'avance, la valeur de l'attribut n'est pas unique sur l'ensemble des fiches d'anomalie et peut donc se retrouver dans d'autres fiches d'anomalie. Les attributs *libres* permettent de renseigner des informations non-structurées, sous forme textuelle ; parmi eux, nous noterons tout particulièrement la

description du problème de l'anomalie et la description de sa solution. La valeur d'un tel attribut est une description textuelle libre. La fiche d'une anomalie non-nominale est essentiellement composée de ce type d'attribut alors que celle d'une anomalie nominale est essentiellement composée d'attributs contraints. Les attributs contraints peuvent donc être rapprochés de questions fermées tandis que les attributs libres doivent être rapprochés de questions ouvertes. Cette première catégorisation fait ressortir le fait qu'il est nécessaire d'appliquer un traitement spécifique sur les attributs libres pour en extraire des informations aussi aisément manipulables que celles des attributs contraints.

Classification des attributs selon leur structuration	
Contraint	Attribut structuré ayant une liste finie (statique ou dynamique) de valeurs
Libre	Attribut non-structuré, texte libre
Classification des attributs selon leur qualité descriptive	
Descriptif de problème	Attribut relevant de la description du problème
Descriptif de solution	Attribut relevant de la description de la solution
Non-descriptif	Attribut n'étant pas considéré comme descriptif, ni du problème, ni de la solution

Table 16 – Propriétés des attributs de fiche d'anomalie

Tout patron de fiche d'anomalie met en jeu de nombreux attributs durant le processus de résolution. Cependant, tous les attributs n'apportent pas la même nature d'information. La propriété de *qualité descriptive* les divise en trois groupes : les attributs *descriptifs de problème*, les attributs *descriptifs de solution* et les attributs *non-descriptifs*. Grâce à cette seconde catégorisation, nous mettons en évidence les attributs porteurs de données pertinentes. Nous donnons ainsi la possibilité de partitionner les attributs en :

- Attributs *descriptifs de problème*, qui informent sur le problème rencontré, l'incident observé.
- Attributs *descriptifs de solution*, qui exposent la solution retenue pour remédier au problème.
- Attributs *non-descriptifs*, qui ne rentrent dans aucune des deux catégories précédentes.

Un attribut peut contenir une valeur renseignée par l'utilisateur mais aussi une valeur remontée du système (pas saisie directement) qui découle du contexte de sa création et permettra sa traçabilité. Ces informations contextuelles sont liées au moment où l'on déclare l'anomalie. Un exemple d'attribut contextuel est la date de création de l'anomalie ou encore l'identification de la personne qui l'a

déclarée. Par définition, un tel attribut est contraint. En revanche, il peut être descriptif de problème, descriptif de solution ou non-descriptif. Le contexte du problème ainsi que celui de la solution sont donc conservés. Il est même possible d'aller au-delà de notre vision en distinguant les anomalies contextuelles des autres (Chandola et al., 2009). Celles-ci sont produites dans un contexte spécifique et les fiches qui leur servent de support possèdent des attributs contextuels et comportementaux dont le traitement préventif leur est propre. En ce qui nous concerne, nous ne faisons pas cette distinction. Notre but est d'utiliser toutes les informations potentiellement utiles contenues dans les attributs, contextuels ou non.

C'est un expert du domaine qui décide du classement des attributs dans les partitions proposées. En effet, alors que la propriété de structuration d'un attribut se déduit directement de celui-ci, la propriété de qualité descriptive d'un attribut est une connaissance qui doit être apportée par un expert du domaine, qui connaît bien le patron de fiche d'anomalie, car ce classement dépend du contexte. Evidemment, le coût de ce travail augmente avec le nombre d'attributs considérés. Quel que soit le nombre de fiches d'anomalie à traiter, c'est bien le nombre d'attributs du ou des patrons de fiche d'anomalie qui conditionne le temps de cette classification. Il n'y a donc pas de réel problème de passage à l'échelle encore moins si les patrons de fiche d'anomalie ont des attributs en commun (auquel cas le choix de sa propriété de qualité descriptive n'est fait qu'une seule fois). Pour minimiser l'interaction humaine, on peut imaginer que l'expert soit assisté dans ses choix par un outil lui proposant une première classification en utilisant, par exemple, une ontologie.

4.5. Modélisation de fiche d'anomalie

4.5.1. Le modèle de descripteur de fiche d'anomalie

Nous proposons de modéliser une fiche d'anomalie sous la forme d'un ensemble d'attributs, chacun d'eux ayant un rôle dans le processus de résolution et contenant des données, structurées ou non (Figure 12). Il faut noter que même si l'on se place dans un seul domaine, plusieurs patrons de fiches d'anomalie peuvent exister et être utilisés car tous les types d'anomalie ne se résolvent pas de la même manière, en suivant les mêmes étapes, en utilisant les mêmes attributs. Toutefois, un même attribut peut être présent dans plusieurs patrons de fiche d'anomalie. Deux anomalies résolues en suivant deux patrons de fiches différents peuvent donc avoir des attributs en commun.

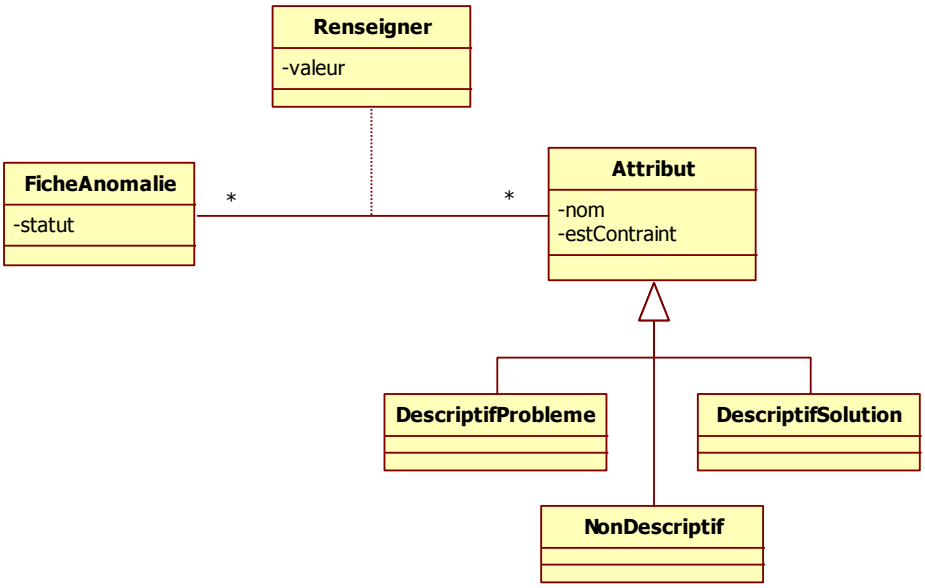


Figure 12 – Le modèle de descripteur de fiche d'anomalie proposé

Ce modèle représente notre vision d'une fiche d'anomalie comme un ensemble d'attributs renseignés par des utilisateurs. Il définit explicitement les deux propriétés qu'un attribut doit avoir, à savoir sa structuration et sa qualité descriptive (Claude et al., 2012). Quel que soit le patron de fiche d'anomalie utilisé, qu'elle soit nominale ou non, l'application de ce modèle sur une fiche d'anomalie permet d'en extraire son descripteur, c'est-à-dire un ensemble structuré d'informations issues des données relatives à l'anomalie. En effet, il suffit d'identifier les attributs (ce sont les éléments qui stockent les données relatives à l'anomalie) et de définir leur classification selon les propriétés de structuration et de qualité descriptive. Une fois ce travail préliminaire fait, le descripteur de l'anomalie est automatiquement créé.

4.5.2. Evaluation du modèle de descripteur de fiche d'anomalie

Ce modèle de descripteur de fiche d'anomalie a été testé sur deux bases de fiches d'anomalie réelles fournies par la société Intercim, impliquée dans ce travail de recherche. Intercim propose des solutions de développement et mise en place de systèmes d'exécution et d'optimisation des procédés industriels pour les industries aéronautique et pharmaceutique. Le traitement des anomalies est donc un problème qui se pose naturellement. La première base contient plus de 7000 fiches d'anomalie logicielle issues de la base d'anomalies logicielles d'Intercim. La deuxième contient près de 75000 fiches d'anomalie de production industrielle issues de la base d'un client d'Intercim. Ce modèle de descripteur de fiche d'anomalie a aussi été testé sur la base de fiches d'anomalie logicielle Bugzilla.

4.5.2.1. Evaluation sur la base de fiches d'anomalie logicielles d'Intercim

L'ensemble des 7216 fiches d'anomalie collectées entre septembre 2005 et décembre 2008 est issu d'un même patron de fiche d'anomalie qui met en jeu 8 étapes. Ces étapes ne sont pas exécutées séquentiellement, c'est au cours de l'exécution du processus et en fonction des informations qui sont renseignées que la succession d'étapes est définie. Dans ce patron de fiche, nous avons relevé 91 attributs. La partition selon la propriété de structuration révèle que 20 attributs sont libres alors que 71 sont contraints. Un des experts qui a défini le patron de cette fiche a indiqué la partition selon la propriété de qualité descriptive suivante : 6 attributs sont descriptifs de problème, parmi eux un seul est libre (Table 17). Le problème principal dans ce jeu de données est qu'aucun attribut ne décrit la solution, aucun d'eux ne peut être défini comme descriptif de solution. Néanmoins, il permet de tester le modèle sur des données et un volume réels.

	Attributs contraints	**Attributs libres**	***Total***
Attributs descriptifs de problème	*5*	1	*6*
Attributs descriptifs de solution	0	0	*0*
Attributs non-descriptifs	66	19	*85*
Total	*71*	*20*	*91*

Table 17 – Partitionnement des attributs des fiches d'anomalie logicielle d'Intercim

4.5.2.2. Evaluation sur la base de fiches d'anomalie de production industrielle d'un client d'Intercim

L'ensemble des 74205 fiches d'anomalie collectées entre octobre 2006 et septembre 2009 est issu de cinq patrons de fiche d'anomalie qui mettent en jeu de nombreuses étapes, bien plus que pour les fiches d'anomalie logicielle présentées ci-avant. Ces étapes ont été régulièrement modifiées, plus d'une dizaine de fois pour certaines. Tout comme le cas précédent, ces étapes ne sont pas exécutées séquentiellement, c'est au cours de l'exécution du processus et en fonction des informations qui sont renseignées que la succession d'étapes est définie. Dans ces patrons de fiche, nous avons relevé 136 attributs. La partition selon la propriété de structuration révèle que 16 attributs sont libres alors que 120 sont contraints. Un des experts qui a défini les patrons de ces

fiches a indiqué la partition selon la propriété de qualité descriptive suivante : 5 attributs sont descriptifs de problème, parmi eux 2 sont libres et 5 attributs sont descriptifs de solution, parmi eux 4 sont libres (Table 18).

	Attributs contraints	**Attributs libres**	***Total***
Attributs descriptifs de problème	3	2	*5*
Attributs descriptifs de solution	1	4	*5*
Attributs non-descriptifs	116	10	*126*
Total	*120*	*16*	*136*

Table 18 – Partitionnement des attributs des fiches d'anomalie de production industrielle d'un client d'Intercim

4.5.2.3. Evaluation sur la base de fiches d'anomalie logicielles de Bugzilla

Ce modèle peut aussi être appliqué sur d'autres bases de fiches d'anomalie connues telles que Bugzilla. La fiche d'anomalie permettant la saisie de données est composée de 18 éléments (Table 19). A l'exception de *URL* et *Attachments*, tous ces éléments peuvent être aisément définis comme des attributs ayant une structure et une qualité descriptive (Table 20). Concernant la propriété de structuration, seuls trois attributs sont libres, *Summary*, *Status Whiteboard* et *Additional Comments*, tous les autres étant contraints.

Concernant la propriété de qualité descriptive, comme nous l'avons exposé plus haut dans ce chapitre en §4.4.2, seul un expert peut décider du classement des attributs dans les partitions de cette propriété. Nous présentons néanmoins une classification pour illustrer cet exemple jusqu'à sa finalisation. Nous avons définis un ensemble d'attributs comme non-descriptifs car ce sont des attributs que l'on pourrait qualifier d'organisationnels, qui permettent de faciliter le traitement et la traçabilité de l'anomalie mais qui n'informent pas réellement du problème rencontré ou de la solution à mettre en place. Nous avons aussi défini un ensemble d'attributs descriptifs de problème. *Assigned To* fait partie de cette liste en considérant que la personne responsable de la résolution est un expert du domaine. *Additional Comments* peut être descriptif de problème et de solution car toutes les personnes en relation avec la fiche d'anomalie peuvent ajouter un commentaire. Un commentaire d'une personne qui a déclaré l'anomalie doit être considéré comme descriptif de problème tandis qu'un commentaire d'une personne responsable de la résolution du problème doit être

considéré comme descriptif de solution. Cependant, il semble que cet attribut ne soit utilisé que pour décrire le problème.

Product and Component	Type de l'anomalie, un produit est composé d'un ou plusieurs composants
Status and Resolution	Etat de l'anomalie, depuis sa création jusqu'à sa clôture (unconfirmed, new, assigned, resolved, reopen, verified, closed)
Assigned To	Personne responsable de la résolution de l'anomalie
URL	URL éventuelle associée à la fiche d'anomalie
Summary	Phrase résumant le problème, servant à l'affichage résumé de la fiche d'anomalie
Status Whiteboard	Zone de texte libre pour ajouter des notes et des mots-clés à la fiche d'anomalie
Keywords	Liste de termes prédéfinis pour assigner des mots-clés à la fiche d'anomalie
Platform and OS	Environnement informatique
Version	Version du produit sur laquelle porte l'anomalie
Priority	Priorité de l'anomalie
Severity	Gravité du problème, de mineur à bloquant
Target	Version du produit pour laquelle l'anomalie devra être résolue
Reporter	Personne qui a déclaré l'anomalie
CC List	Liste de personnes à qui un mail est envoyé quand la fiche d'anomalie évolue
Attachments	Fichiers joints à la fiche d'anomalie
Dependencies	Liens avec d'autres anomalies, l'anomalie peut être bloquée par d'autres anomalies ou au contraire elle peut bloquer d'autres anomalies
Votes	Votes éventuels
Additional Comments	Descriptions supplémentaires sur l'anomalie, par exemple les étapes à suivre pour reproduire l'anomalie

Table 19 – Eléments constituant une fiche d'anomalie de Bugzilla

Les éléments *URL* et *Attachments* ne peuvent être aisément définis comme des attributs du fait qu'ils ne contiennent pas de valeur textuelle. *URL* faisant référence à une page web et *Attachments* listant des fichiers associés à la fiche d'anomalie, la propriété de structure n'est pas adaptée. Pour les considérer comme des attributs, nous pourrions les définir comme libres dans un sens plus large que simplement le sens « texte libre ». Cela implique toujours qu'un traitement spécifique doit être appliqué sur ces attributs mais ce traitement doit être différent de celui appliqué sur un texte libre. De plus nous pouvons les considérer comme descriptifs de problème, ou à défaut comme non-descriptifs.

	Structure	***Qualité descriptive***
Product and Component	Contraint	Descriptif de problème
Status and Resolution	Contraint	Non-descriptif
Assigned To	Contraint	Descriptif de problème
URL	Libre	Descriptif de problème
Summary	Libre	Descriptif de problème
Status Whiteboard	Libre	Descriptif de problème
Keywords	Contraint	Descriptif de problème
Platform and OS	Contraint	Descriptif de problème
Version	Contraint	Non-descriptif
Priority	Contraint	Non-descriptif
Severity	Contraint	Non-descriptif
Target	Contraint	Non-descriptif
Reporter	Contraint	Non-descriptif
CC List	Contraint	Non-descriptif
Attachments	Libre	Descriptif de problème
Dependencies	Contraint	Non-descriptif
Votes	Contraint	Non-descriptif
Additional Comments	Libre	Descriptif de problème

Table 20 – Propriété des attributs de fiche d'anomalie de Bugzilla

Dans ce patron de fiche d'anomalie de Bugzilla, nous avons donc relevé 18 attributs. La partition selon la propriété de structuration révèle que 5 attributs sont libres alors que 13 sont contraints. En suivant la partition selon la propriété de qualité descriptive que nous proposons, nous obtenons 9 attributs descriptifs de problème, parmi eux 5 sont libres (Table 21). Comme pour la

base de fiches d'anomalie logicielle d'Intercim présentée plus haut en §4.5.2.1, le problème principal dans ce jeu de données est qu'aucun attribut ne décrit la solution, aucun d'eux ne pouvant être défini comme descriptif de solution.

	Attributs contraints	**Attributs libres**	***Total***
Attributs descriptifs de problème	4	5	*9*
Attributs descriptifs de solution	0	0	*0*
Attributs non-descriptifs	9	0	*9*
Total	*13*	*5*	*18*

Table 21 – Partitionnement des attributs des fiches d'anomalie logicielle de Bugzilla

4.5.2.4. Evaluation générale

Le modèle de descripteur de fiche d'anomalie que nous proposons est donc suffisamment générique pour être adapté aux documents que nous voulons manipuler. Sur les deux bases de fiches d'anomalie logicielle, le modèle a mis en évidence le fait que les informations sur la solution, c'est-à-dire la façon dont le bug a été résolu, ne sont pas renseignées, non pas parce que les utilisateurs ne prennent pas le temps de le faire mais parce qu'aucun attribut n'est dédié à ce rôle. Sur la base de fiches d'anomalie de production industrielle, le modèle a une nouvelle fois montré qu'il était adapté, même lorsque le nombre de fiches d'anomalie est important et que celles-ci sont issues de patrons différents dont les étapes ont été modifiées au cours du temps. Ce modèle décrit donc toute fiche d'anomalie en définissant les attributs qui la composent selon une *propriété de structuration* qui indique ceux pour lesquels un traitement spécifique doit être appliqué (aspect syntaxique) et selon une *propriété de qualité descriptive* qui informe de la nature des données qui y sont renseignées (aspect sémantique).

4.6. Conclusion

Ce chapitre présentait notre vision du processus de résolution d'anomalie à partir duquel nous avons dégagé les éléments qui interviennent dans les fiches d'anomalie ce qui nous a conduits à la proposition d'un modèle de descripteur de fiche d'anomalie. Nous avons validé ce modèle en menant une évaluation

sur plusieurs bases de fiches d'anomalie issues de domaines aussi variés que la maintenance logicielle et la maintenance de processus de production.

Pour atteindre nos objectifs de maintenance corrective et préventive, ce modèle doit être intégré et utilisé dans un environnement permettant le regroupement des fiches d'anomalie passées et résolues. Les groupes obtenus seront à la base des processus de correction et de prévention des nouvelles anomalies. Pour cela nous proposons dans le chapitre suivant un framework global de gestion de fiches d'anomalie qui s'appuie sur la modélisation d'anomalie exposée dans ce chapitre pour établir des groupes de fiches d'anomalie pertinents, c'est-à-dire des patterns d'anomalie associant des problèmes à des solutions. Les deux bases de fiches d'anomalies qui ont servi à l'évaluation du modèle de descripteur seront à nouveau utilisées au chapitre 6 pour mener des expérimentations sur ce framework afin de le valider et de juger de son intérêt.

Chapitre 5

Framework de gestion de fiches d'anomalie pour la construction d'un système d'aide à la maintenance

5.1. Introduction

Pour faciliter la correction et la prévention d'anomalie, nous voulons nous servir des informations contenues dans les fiches d'anomalie passées et résolues. Afin d'extraire et d'exploiter ces informations de manière pertinente, nous avons élaboré dans le chapitre précédent un modèle de descripteur de fiche d'anomalie. Celui-ci est principalement basé sur le fait que les informations présentes dans les fiches d'anomalie peuvent être de différentes natures, elles décrivent soit le problème rencontré soit la solution mise en œuvre pour le résoudre. Cette représentation uniforme de toute fiche d'anomalie nous permet de comparer les fiches d'anomalie entre elles pour en dégager, à un niveau physique, des groupes de fiches d'anomalie, et à un niveau plus logique, des patterns d'anomalie reliant des problèmes et des solutions.

Dans ce chapitre, nous exposons l'approche retenue pour classifier les fiches d'anomalie présentes dans une base. Avec cette approche hiérarchique qui considère les informations sur le problème indépendamment des informations sur la solution, la classification obtenue permet de faciliter les activités de maintenance corrective et préventive grâce aux groupes de problèmes et aux groupes de solutions obtenus (Claude et al., 2011), (Claude et al. 2012). Le choix de l'algorithme de regroupement est discuté et deux versions supplémentaires sont proposées pour pallier à sa principale faiblesse. Des indicateurs ont été retenus pour évaluer la qualité des groupes, indépendamment des autres groupes (indicateurs intra-groupe) et en fonction des autres groupes (indicateurs inter-groupes).

5.2. Gestion des bases de fiches d'anomalie

5.2.1. Systèmes de traitement de bases de fiches d'anomalie

Enmaintenance préventive, des démarches d'analyse telles que l'Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité (AMDEC) très utilisée dans le secteur de l'automobile (Bowles, 2003), de l'aéronautique (Balaban et al., 2009) et du ferroviaire (Li et al., 2009) ou l'Analyse des Dangers et Points Critiques pour leur Maîtrise (HACCP), dérivée de l'AMDEC et utilisée dans les industries agro-alimentaire, chimique et pharmaceutique, sont couramment utilisées pour prévenir l'apparition d'anomalies. Ces démarchent impliquent des individus ayant diverses compétences et expériences dont le but est de :

1. Rechercher et décrire les défaillances potentielles d'un système depuis leur(s) origine(s) (causes) jusqu'à leur(s) conséquence(s) (effets),
2. Quantifier au travers d'un indice de criticité les risques qu'entraînent ces défaillances pour l'utilisateur,
3. Hiérarchiser les actions correctives à initier sur le patron du processus pour optimiser et pérenniser la fiabilité du système.

Notre but est le même que celui de ces méthodes. Nous souhaitons appliquer des actions correctives sur le patron du processus de production industrielle pour prévenir l'apparition d'anomalies. En revanche, alors que ces méthodes visent, dans un premier temps, à identifier manuellement les défaillances potentielles, travail qui peut s'avérer long et fastidieux, nous préférons nous servir des informations contenues dans les fiches d'anomalie passées. Leur regroupement, selon leurs similarités, doit dégager automatiquement des groupes de fiches d'anomalie dont l'analyse permet de définir les actions correctives à appliquer sur le patron du processus de production industrielle.

La maintenance corrective repose sur deux techniques. La première consiste à rechercher des fiches d'anomalie pertinentes, car similaires sur certains points à la nouvelle fiche d'anomalie, en utilisant des mots-clés. Pour cela, l'utilisateur est invité à saisir un ou plusieurs mots-clés, ce qui va initier un processus de recherche de ces mots-clés dans la base de fiches d' anomalie qui ont été traitées par le passé et ainsi sélectionner et présenter à l'utilisateur des fiches d'anomalie similaires. En utilisant les informations d'une ou plusieurs de celles-ci, et notamment les informations liées à la solution, l'utilisateur peut proposer une solution plus simplement et plus rapidement. Cependant, cette méthode s'adresse plutôt à des utilisateurs connaissant bien le domaine et à-même de fournir des mots-clés significatifs et discriminants lors de la recherche. De plus, la pertinence des solutions est discutable puisqu'elles sont

présentées de manière brute à l'utilisateur, sans qu'elles n'aient été classées selon des critères tels que le temps de mise en œuvre, le coût ou tout simplement la qualité.

Nous pouvons mentionner une variante de cette technique comparable sur la méthode mais bien différente sur l'application : elle est basée sur la connaissance d'utilisateurs dédiés à la résolution d'anomalies. Ainsi, lorsqu'ils doivent résoudre le problème d'une nouvelle fiche d'anomalie, ils se basent sur leur propre expérience, ils se rappellent comment ils ont déjà résolu ce genre de problème. Il y a donc ici aussi un processus de recherche de fiches d'anomalie similaires, mais celui-ci est fondé sur la mémoire de l'utilisateur en utilisant son expérience. Evidemment, cette approche connaît ses limites avec la disparition de ces utilisateurs de l'entreprise.

La seconde technique consiste à organiser les fiches d'anomalie selon une structure définie a priori pour permettre à l'utilisateur de naviguer en choisissant différentes catégories. Adaptée à la gestion des fiches d'anomalie industrielles dans l'aérospatial, cette approche a été proposée par (Goh et al., 2009) en utilisant une classification par facettes (Prieto-Diaz et al., 1987), (Giess et al., 2008). Son utilisation permet la navigation dans une arborescence de groupes de fiches d'anomalie pour non seulement faciliter la recherche de solutions employées par le passé mais aussi mettre en place un retour d'expérience entre le service de maintenance et le service de conception. Ce dernier définit les patrons de processus de production industrielle sur lesquels les anomalies sont déclarées. Toutefois, un effort important doit être fait en amont pour concevoir un tel schéma de classification de la base de fiches d'anomalie pleinement utilisable. L'utilisation de descripteurs caractérisant les fiches d'anomalie, dont nous proposons un modèle dans cet article, faciliterait ce travail.

Pour pallier aux limites de ces approches (connaissance (experte) du domaine, travail important en amont), nous proposons une approche qui permet de classifier de manière automatique les fiches d'anomalie en mettant en évidence les critères de regroupement identifiés dans le modèle de descripteur de fiche d'anomalie que nous avons défini. Nous pouvons ainsi déterminer différents groupes de fiches d'anomalie que nous plaçons au cœur des activités de maintenance, corrective et préventive. De cette manière, nous utilisons l'information relative aux anomalies passées pour non seulement en résoudre une nouvelle mais aussi pour améliorer la qualité du processus de production industrielle en remontant ces informations depuis les équipes de production jusqu'aux équipes de conception, retour d'expérience rarement présent dans les entreprises (Levner et al., 1998).

5.2.2. Structuration de la base de fiches d'anomalie

5.2.2.1. Définition des catégories

Tout d'abord, comme nous l'avons évoqué précédemment, toutes les anomalies ne se résolvent pas de la même façon. Il n'y a donc pas de sens à comparer toutes les fiches d'anomalie d'une base entre elles. Quelle établir entre une anomalie de documentation et une anomalie qui survient lors d'une étape de fabrication ? Nous pouvons imaginer que l'anomalie de fabrication est due à l'anomalie de documentation mais il serait difficile de trouver une similarité entre leurs fiches correspondantes, leur descripteurs donc, sur leur problème ou sur leur solution. Nous distinguons donc différentes catégories de fiches d'anomalie à l'intérieur d'une même base de fiches d'anomalie. Ces catégories sont définies suivant plusieurs attributs, qui sont considérés comme très discriminants.

L'intérêt de créer des catégories de fiches d'anomalie est de proposer un premier regroupement simple pour éviter de comparer toutes les fiches d'anomalie entre elles alors que l'on sait qu'il en existe des populations distinctes. Toute la difficulté réside donc dans le fait de déterminer les critères discriminants conduisant à la création de catégories pertinentes. A cause de l'information non-structurée qu'ils contiennent, les attributs libres ne sont pas retenus, cela demanderait un sérieux travail d'analyse d'un expert ou l'utilisation de techniques de fouille de texte, de discrétisation, de logique floue, etc. Nous devons donc nous tourner vers les attributs contraints. Comme nous voulons faciliter la recherche de fiches d'anomalie similaires à une nouvelle fiche d'anomalie dont on ne connaît évidemment pas la solution, cette classification doit différencier les fiches d'anomalie exclusivement selon leur problème. Nous nous intéressons donc plus particulièrement aux attributs contraints descriptifs de problème. Parmi eux, certains vont contenir des valeurs tellement caractéristiques que chercher une similarité entre deux fiches d'anomalie qui n'ont pas la même valeur pour l'un de ces attributs ne semble pas avoir de sens.

Ainsi, dans l'ensemble des attributs, seuls quelques-uns sont utilisés pour définir les catégories. Grâce au modèle de descripteur que nous proposons, nous pouvons filtrer les attributs qui peuvent servir de critères de catégories de fiches d'anomalie. Un premier filtrage est exécuté au travers de la propriété de structure : nous ne conservons que les attributs contraints. Ensuite, un second filtrage est effectué au travers de la propriété de qualité descriptive : nous ne conservons que les attributs descriptifs de problème. Parmi les attributs restants, dont le nombre peut être assez grand malgré l'étape de filtrage, un expert, qui connaît bien le domaine, détermine ceux qui sont réellement

discriminants. Ce travail est nécessairement spécifique au domaine, aucun attribut ne peut être défini comme critère de catégories de fiches d'anomalie par défaut.

5.2.2.2. Définition des groupes de problèmes et de solutions

Suite à ce premier travail de division de la base de fiches d'anomalie, à l'intérieur de chacune des catégories, les groupes de fiches d'anomalie similaires sur le problème sont déterminés. Des calculs de similarité entre fiches d'anomalie, plus précisément entre les parties problème de leur descripteur, sont effectués. Un algorithme de clustering utilise les valeurs obtenues pour créer les groupes de problèmes.

Puis, à l'intérieur de ces groupes de problèmes, des groupes de fiches d'anomalie similaires sur la solution sont déterminés. De la même manière que les groupes de problèmes sont créés, des calculs de similarité entre les parties solution des descripteurs des fiches d'anomalie sont effectués et un algorithme de clustering utilise les valeurs obtenues pour créer des groupes de solutions.

5.2.2.3. Définition des prototypes de groupe

Pour chaque groupe de problèmes et de solutions, un prototype est construit. C'est un résumé des informations des fiches d'anomalie du groupe, une fiche d'anomalie artificielle. C'est un descripteur de fiche d'anomalie construit automatiquement qui est représentatif du groupe. Il est donc composé d'une partie problème et d'une partie solution. La partie non-descriptive est inutile puisqu'elle n'intervient pas dans la structuration de la base de fiches d'anomalie.

Chaque partie est construite de la manière suivante. Les éléments de la partie problème du descripteur de toutes les fiches d'anomalie du groupe sont intégrés dans la partie problème du descripteur du prototype. Il en est de même pour la partie solution du descripteur de prototype. Les éléments de la partie solution du descripteur de toutes les fiches d'anomalie du groupe y sont intégrés.

Ainsi, la partie problème du prototype d'un groupe de problèmes est utilisée pour calculer la similarité avec la partie problème du descripteur d'une nouvelle fiche d'anomalie. Il n'est donc pas nécessaire de comparer la nouvelle fiche d'anomalie avec toutes les fiches d'anomalie qui appartiennent au groupe. Concernant les groupes de solutions, d'une part, la partie solution du prototype est utilisée pour présenter un résumé de la solution envisageable. D'autre part, pour déterminer la solution du groupe de problèmes la plus adaptée à un nouveau problème, la partie problème du prototype et celle du descripteur de la nouvelle fiche d'anomalie sont comparées.

5.2.2.4. Illustration de la structuration de la base de fiches d'anomalie

Pour illustrer la structuration de la base de fiches d'anomalie, prenons un exemple dans lequel deux attributs contraints et descriptifs de problème, *Patron de fiche d'anomalie* et *Pièce en cause*, servent de critères de catégories de fiches d'anomalies. Nous considérons trois patrons de fiche : *documentation*, *électrique* et *mécanique*, et cinq éléments d'avion sur lesquels une anomalie peut survenir : *aile*, *cabine de pilotage*, *fuselage*, *nez* et *train d'atterrissage*. Les fiches d'anomalie de la base sont donc divisées en quinze catégories. Dans chaque catégorie, les groupes de problèmes et de solutions sont déterminés et pour chacun d'eux un prototype est créé (Figure 13). La partie inférieure de la figure est un zoom sur l'une des catégories.

Le groupe de problèmes, composé d'un seul groupe de solutions, situé en haut au centre de la figure est un cas particulier pour l'affichage de leur prototype respectif. Le fait qu'il n'y ait qu'un groupe de solutions qui compose le groupe de problèmes implique que les deux prototypes sont exactement les mêmes. Ils possèdent les mêmes informations sur le problème et sur la solution puisque ces informations sont extraites du même ensemble de fiches d'anomalie. Ils doivent donc être représentés au même endroit. En conséquence, le prototype du groupe de problèmes est masqué par le prototype du groupe de solutions.

	Groupe de problèmes
	Groupe de solutions
	Fiche d'anomalie dont les informations sur le problème ont été renseignées mais pas celles sur la solution
	Fiche d'anomalie dont les informations sur la solution ont été renseignées mais pas celles sur le problème
	Fiche d'anomalie dont les informations sur le problème et sur la solution sont renseignées
	Prototype de groupe de problèmes
	Prototype de groupe de solutions

Table 22 – Légende des illustrations représentant une base de fiches d'anomalie

La légende utilisée sera la même pour les prochaines illustrations de ce mémoire représentant une base de fiches d'anomalie (Table 22). La couleur rouge fera référence aux éléments relatifs au problème tandis que la couleur bleue fera référence aux éléments relatifs à la solution.

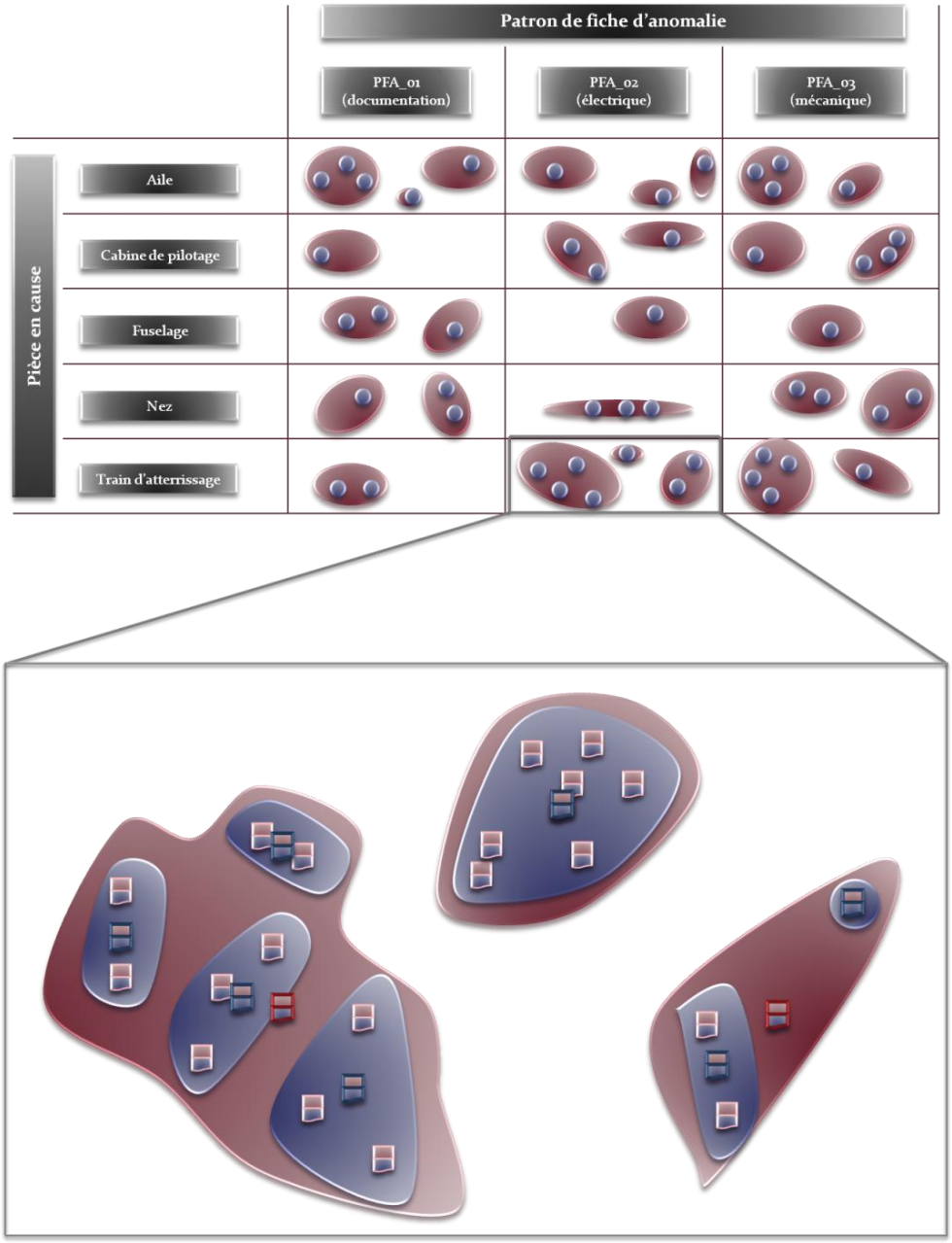


Figure 13 – Illustration de la structuration de la base d'anomalies

5.2.2.5. Evolution du contenu de la base de fiches d'anomalie

Lorsqu'une nouvelle fiche d'anomalie est close, elle est intégrée dans la classification de la base avec les autres fiches d'anomalie. La classification de la base est donc dynamique, elle évolue avec les nouvelles fiches d'anomalie

qui sont intégrées. Les prototypes des groupes évoluent avec la modification des groupes. Cet aspect est illustré en Figure 14. La fiche d'anomalie intégrée est en surbrillance verte et les éléments modifiés par l'ajout de cette fiche d'anomalie sont en surbrillance orange.

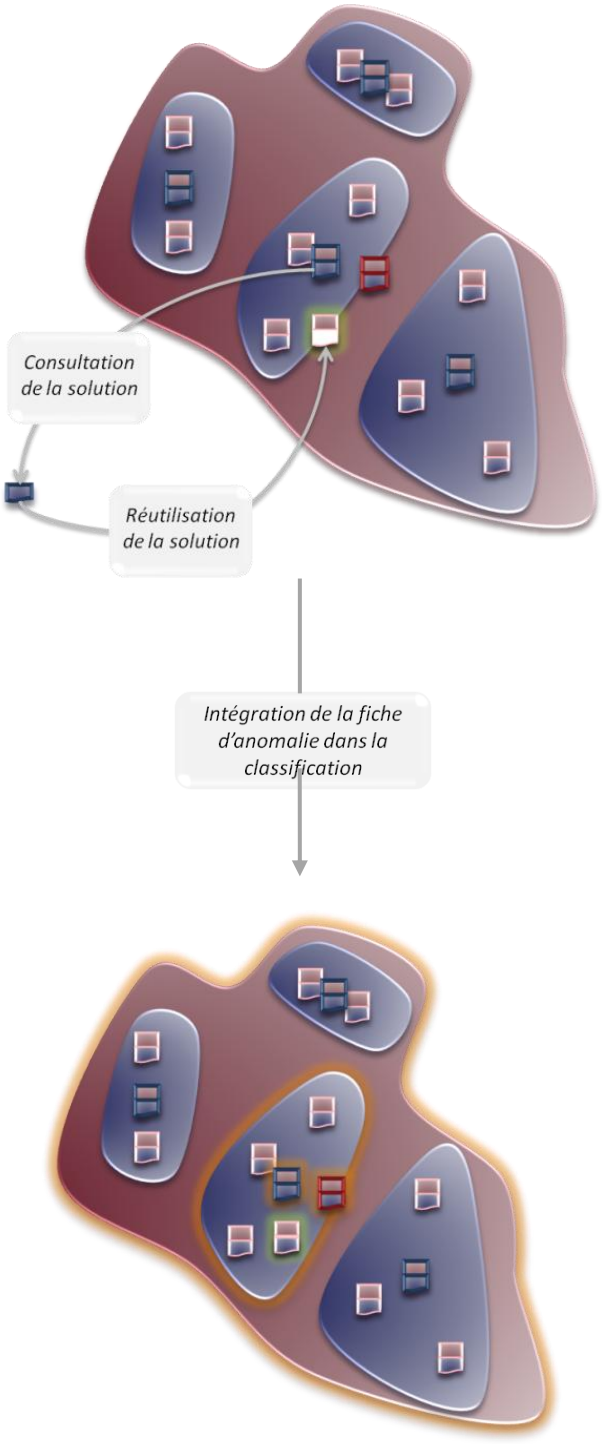


Figure 14 – Dynamicité de la classification

5.2.2.6. Application de l'approche à la maintenance corrective

Pour faciliter la maintenance corrective, l'utilisateur se voit proposer les solutions de fiches d'anomalie dont le problème est similaire à celui de la fiche d'anomalie courante. Pour cela, toutes les fiches d'anomalie passées ne sont pas considérées mais seulement celles appartenant à la même catégorie que la fiche d'anomalie courante. Ensuite, les groupes de problèmes et les groupes de solutions sont utilisés pour trouver automatiquement et rapidement la solution la plus appropriée pour résoudre le problème courant.

Prenons l'exemple de la figure ci-dessous (Figure 15) qui représente les fiches d'anomalie d'une catégorie organisées en groupes. Suite à la détection d'un problème, une nouvelle fiche d'anomalie arrive dans cette catégorie, le problème y est renseigné et une solution est recherchée. Dans un premier temps, la partie problème de cette nouvelle fiche d'anomalie est comparée avec la partie problème de chacun des prototypes des groupes de problèmes de la catégorie. Pour indiquer les similarités, cette même fiche d'anomalie représentée en surbrillance verte est positionnée spatialement par rapport aux prototypes des groupes de problème. Ces similarités sont symbolisées au travers des lignes en pointillés rouges. Ici, seul le groupe de gauche est retenu, la similarité avec les autres prototypes étant trop faible. Dans un second temps, la partie problème de cette nouvelle fiche d'anomalie est comparée avec la partie problème de chacun des prototypes des groupes de solutions de ce groupe de problèmes. Ces similarités sont symbolisées au travers des lignes en pointillés bleus.

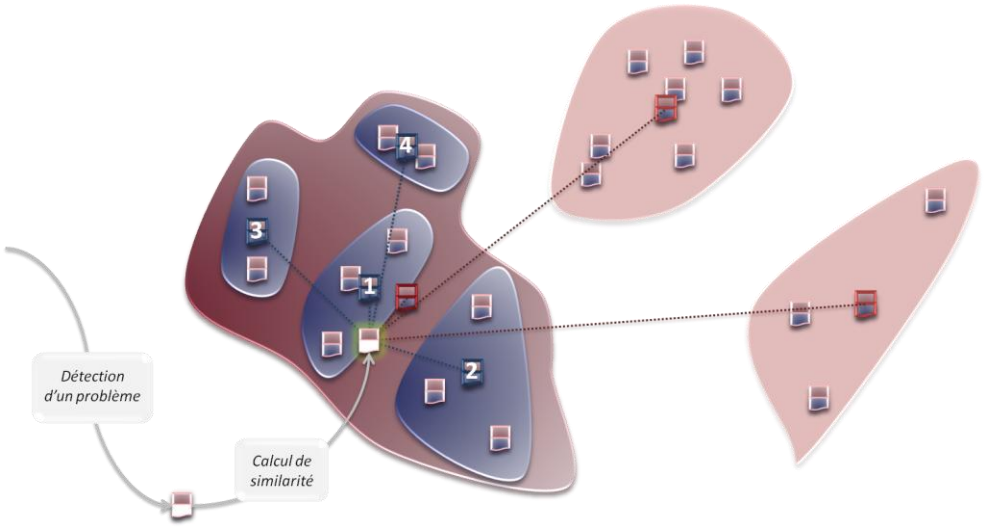


Figure 15 – Application de l'approche à la maintenance corrective

Les solutions passées sont ordonnées, ici de 1 à 4, non pas selon la taille des groupes de solutions mais selon la pertinence de la solution par rapport au problème. L'utilisateur peut ainsi rapidement consulter la meilleure solution

proposée ou plusieurs solutions et réutiliser ces informations, en les adaptant si besoin, pour proposer une solution au problème présenté dans la nouvelle fiche d'anomalie.

5.2.2.7. Application de l'approche à la maintenance préventive

Pour faciliter la maintenance préventive, les anomalies récurrentes peuvent être facilement déduites en utilisant les groupes de problèmes définis. Prévenir l'apparition de telles anomalies résulte de l'analyse des groupes volumineux de fiches d'anomalie par un expert pour comprendre les raisons de leur apparition et mettre en place des moyens empêchant de les reproduire, en modifiant, par exemple, le patron du processus de production industrielle.

Prenons l'exemple de la figure ci-dessous (Figure 16) qui représente les fiches d'anomalie d'une catégorie organisées en groupes. Le regroupement en haut au centre est assez particulier. D'une part, il n'y a qu'un seul groupe de solutions dans le groupe de problèmes et d'autre part, le nombre de fiches d'anomalie de ce groupe de solutions est bien plus important que celui des groupes de solutions des autres groupes de problèmes. L'utilisateur peut ainsi rendre compte de l'aspect répétitif d'une anomalie qui, ici, présente toujours un problème semblable et une solution similaire dans les fiches d'anomalie. Une anomalie peut aussi être récurrente même s'il y a plusieurs groupes de solutions dans un groupe de problèmes. C'est le nombre de fiches d'anomalie dans un groupe, de manière absolue, relativement par rapport aux autres groupes ou son évolution dans le temps, qui peut révéler cette récurrence.

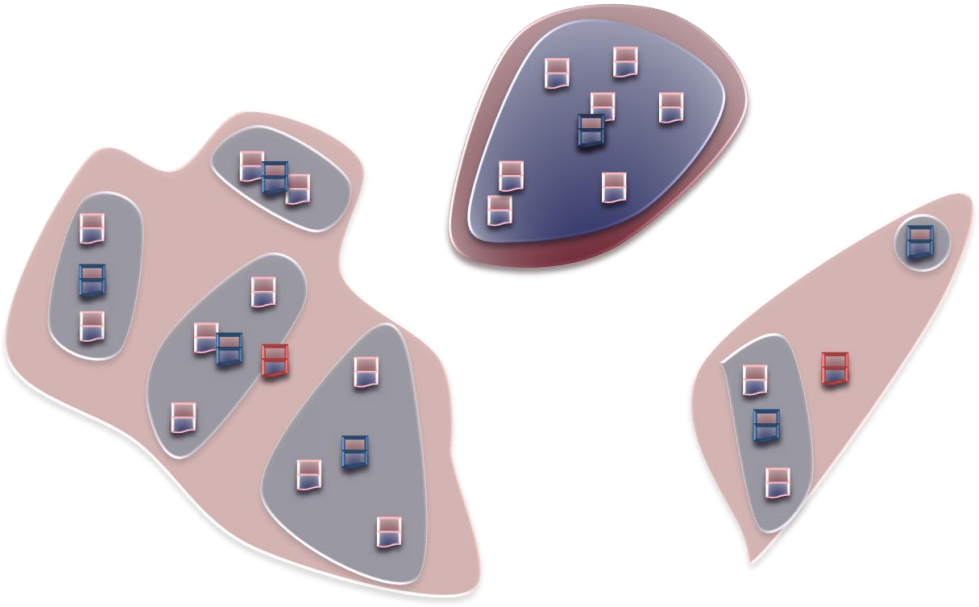


Figure 16 – Application de l'approche à la maintenance préventive

5.2.2.8. Synthèse de l'approche

Pour atteindre cet objectif d'aide à la maintenance corrective et à la maintenance préventive, deux processus doivent être mis en place, l'un offline pour effectuer la classification des fiches d'anomalie (Figure 17) et l'autre online pour effectuer la recherche de fiches d'anomalie similaires (Figure 18). Cette deuxième activité se base sur les résultats issus de la première.

Nous appelons attribut défini tout attribut dont les propriétés de structuration et de qualité descriptive ont été déterminées ainsi que, dans le cas d'un attribut contraint descriptif de problème, s'il est critère de catégories d'anomalies. Concernant le processus de classification des fiches d'anomalie, suite à l'extraction des descripteurs, chaque fiche d'anomalie est représentée en deux parties, une partie problème et une partie solution. Le regroupement sur le problème s'effectue en calculant la similarité entre les parties problème des descripteurs. De même, le regroupement sur la solution s'effectue en calculant la similarité entre les parties solution des descripteurs.

Concernant le processus de recherche de fiches d'anomalie similaires, les premières activités sont identiques au processus décrit précédemment, à savoir la catégorisation de la nouvelle fiche d'anomalie et l'extraction de son descripteur. Avec ces informations et la classification de la base des fiches d'anomalie obtenue avec le processus décrit précédemment, la partie problème du descripteur de la nouvelle fiche d'anomalie est comparée à la partie problème du descripteur de tous les prototypes de problèmes de la même catégorie. Ceux ayant une similarité suffisante sont retenus. Ces prototypes représentent chacun un groupe de problèmes dans lequel des groupes de solutions ont été construits. La partie problème de chacun de ces groupes de solutions est comparée avec la partie problème du descripteur de la nouvelle fiche d'anomalie. Les groupes de solutions sont ainsi ordonnés selon leur pertinence par rapport au problème de la fiche d'anomalie courante.

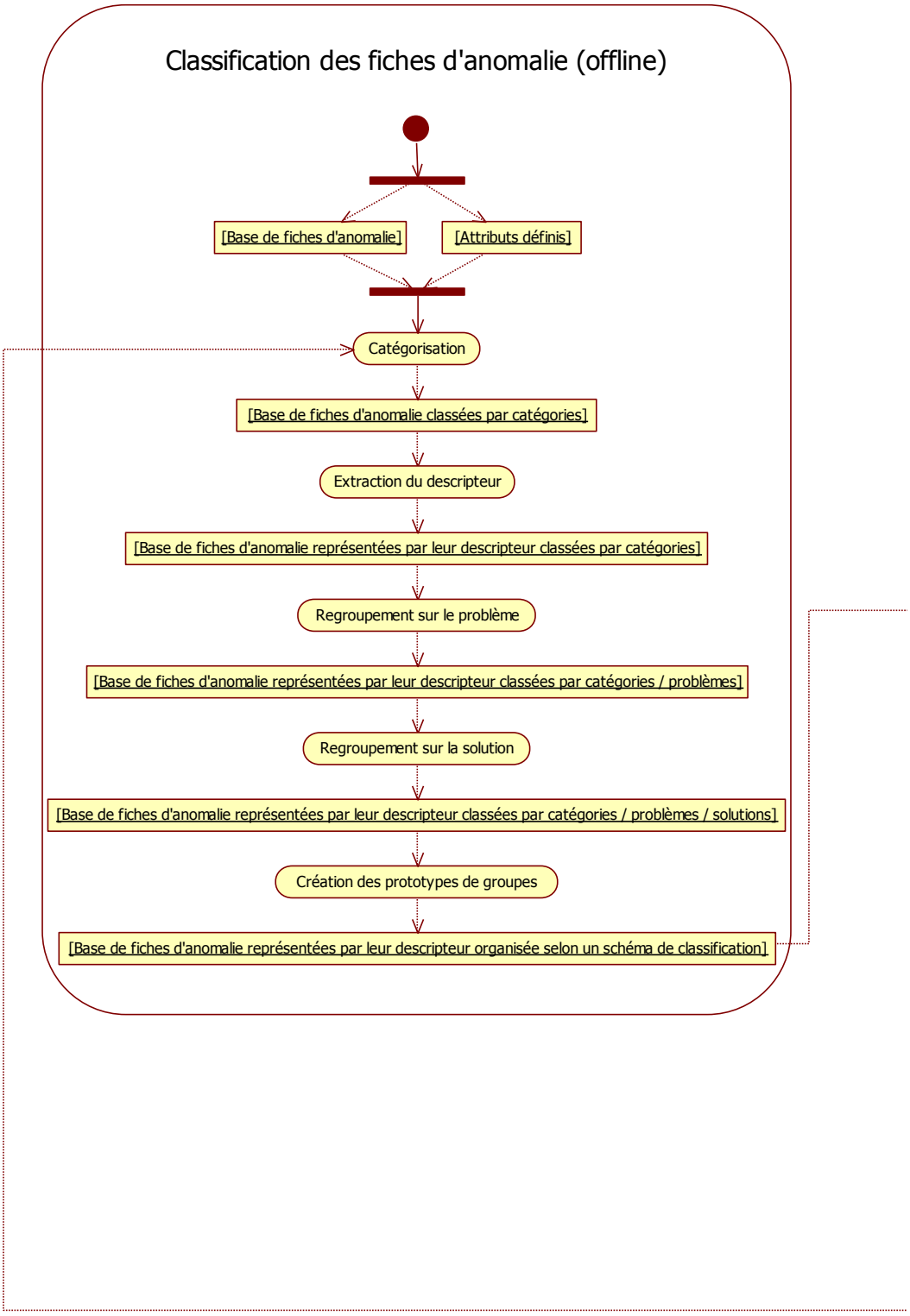


Figure 17 – Processus de classification de fiches d'anomalie et de recherche de fiches d'anomalie similaires (1/2)

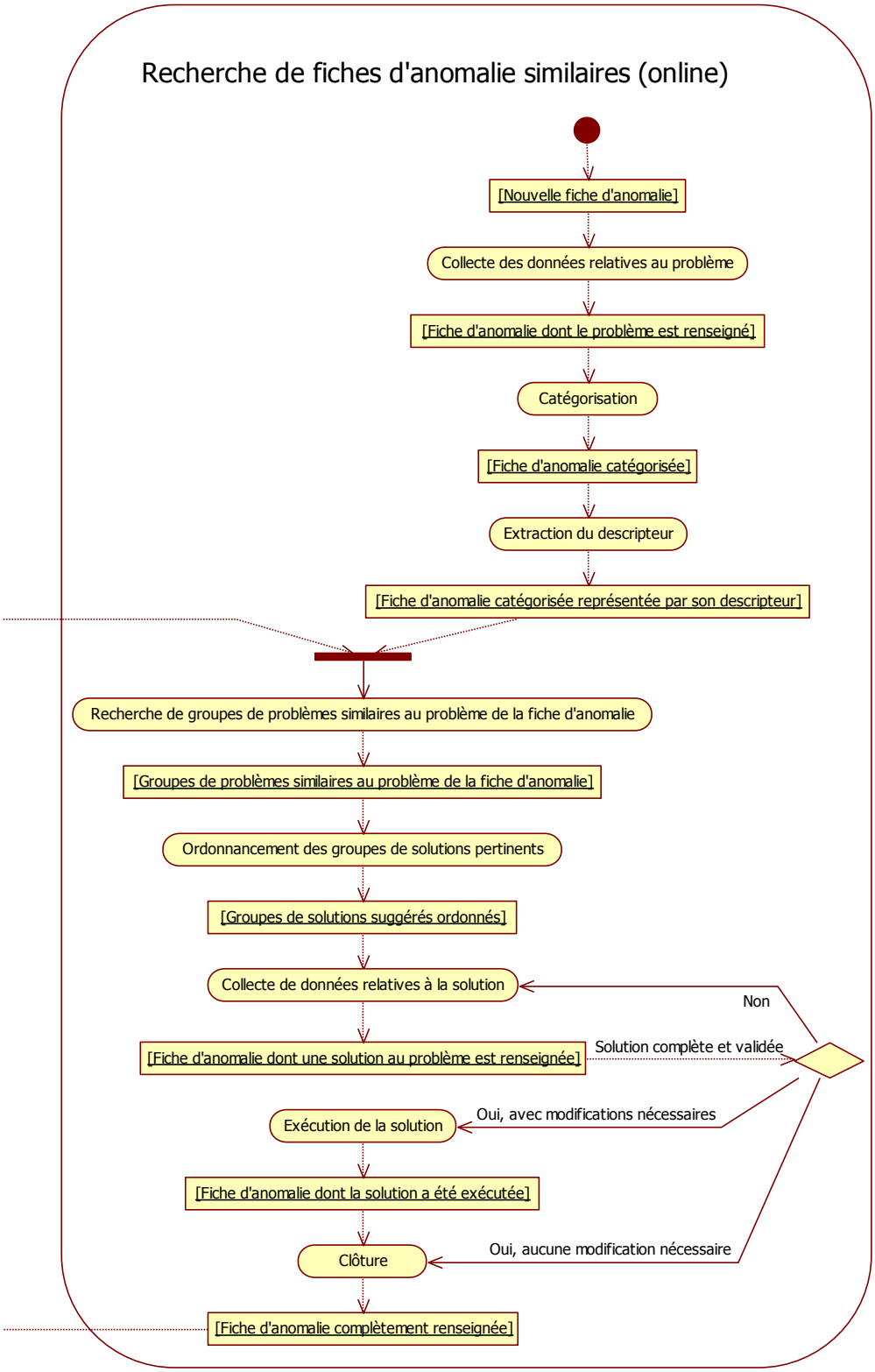


Figure 18 – Processus de classification de fiches d'anomalie et de recherche de fiches d'anomalie similaires (2/2)

5.3. Contribution sur l'algorithme de CAH (Clustering Ascendant Hiérarchique)

5.3.1. Choix de l'algorithme de CAH

Les environnements de regroupement exposés dans le chapitre 3 présentent des avantages et des inconvénients. Pour déterminer le plus adapté aux processus de classification et de recherche, nous avons pris en compte cinq paramètres essentiels au framework de gestion de fiches d'anomalie que nous proposons :

- *Le passage à l'échelle* : support d'un volume de données important, manipulation de plusieurs dizaines (voire centaines) de milliers de fiches d'anomalie et prise en compte de dizaines d'attributs (réellement utilisés parmi les centaines d'attributs qui composent un patron de fiche d'anomalie).
- *Le temps de réponse* : pour retourner des éléments ou des groupes similaires en temps réel
- *La pertinence des groupes* : les groupes obtenus sont-ils clairement définis ?
- *La visualisation* : correspond au caractère explicatif, au fait que l'utilisateur puisse comprendre pourquoi les groupes ont été formés de cette manière
- *L'aspect dynamique* : correspond à l'aspect interactif, au fait que l'utilisateur puisse modifier manuellement les groupes obtenus.

Nous avons classé les environnements de regroupement en fonction de ce que nous avons pu déduire de leurs points forts et de leurs points faibles décrits dans la littérature. Le paramètre solution adaptée est une agrégation de ces cinq paramètres, c'est la moyenne des notes données pour chacun des cinq paramètres (Table 23).

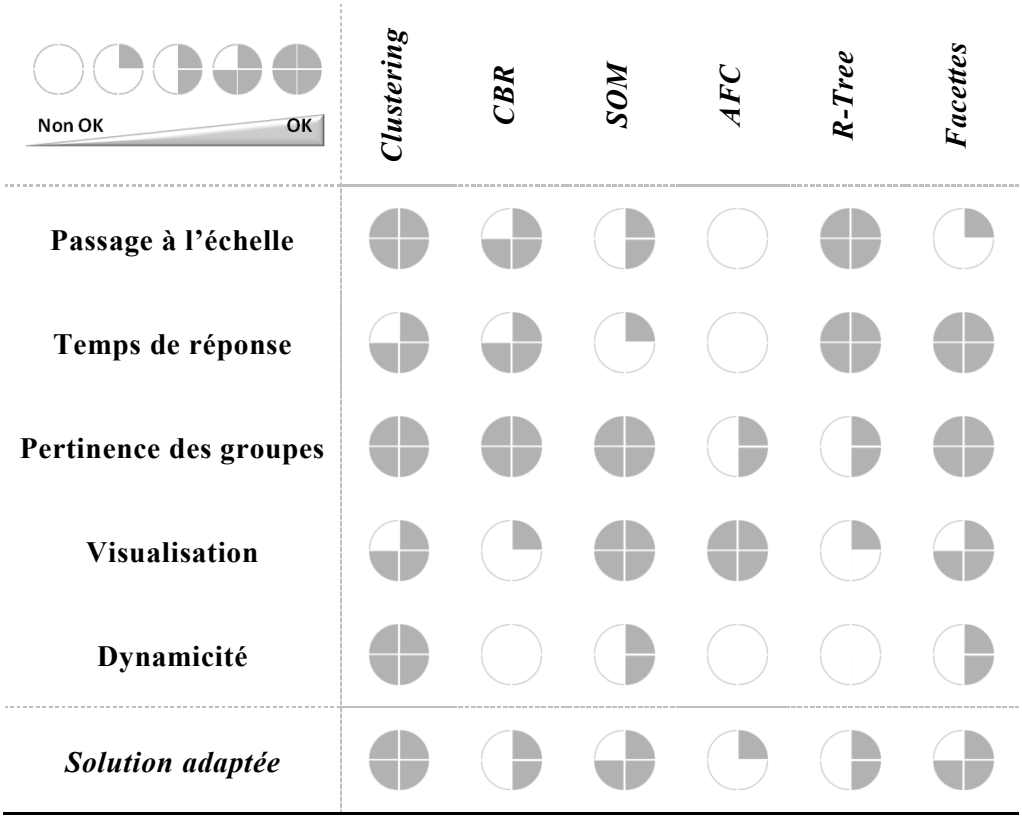


Table 23 – Comparaison des environnements de regroupement en fonction des besoins de notre framework global

Parmi les environnements de regroupement, nous avons choisi de nous tourner vers les méthodes de clustering qui, du fait de leur variété, offrent de nombreuses possibilités pour atteindre les paramètres retenus. Plus particulièrement, nous nous sommes intéressés à l'algorithme de clustering ascendant hiérarchique et ce, pour plusieurs raisons (Table 24). La raison principale concerne la caractéristique *Connaissance a priori*. Contrairement aux autres algorithmes de clustering comme le très utilisé K-means, celui-ci n'oblige pas à définir a priori le nombre de groupes à obtenir mais seulement le seuil de similarité. Dans notre framework, il serait difficile de choisir le nombre de groupes de problèmes de chaque catégorie et le nombre de groupes de solutions de chaque groupe de problème. Un seuil de similarité est beaucoup plus adapté.

Parmi les caractéristiques apparaissant comme des points faibles, nous pouvons relever deux points. Premièrement, il ne permet pas le chevauchement, c'est-à-dire qu'un document doit être membre d'un seul groupe, ce qui se traduit dans notre framework par le fait qu'une fiche d'anomalie ne peut appartenir qu'à un seul groupe de problèmes et un seul groupe de solutions. Elle ne peut pas être membre de plusieurs groupes de problèmes ou de solutions avec un pourcentage d'appartenance pour chacun d'eux. Cela correspond bien à notre stratégie. Nous considérons qu'une fiche d'anomalie ne doit appartenir qu'à un

groupe. Le problème ou la solution d'une fiche d'anomalie ne doit avoir de similarité assez forte qu'avec un seul groupe. Nous préférons que les groupes soient fusionnés ou qu'un nouveau groupe apparaisse plutôt que de permettre le chevauchement qui rend la visualisation moins explicite. Deuxièmement, la complexité quadratique de cet algorithme le rend difficile à utiliser sur un large volume de documents. Cependant, des versions incrémentales existent et résolvent ce problème de passage à l'échelle.

Concernant les autres caractéristiques, elles nous apparaissent comme mineures. Aucun algorithme n'est tolérant à tout, il faut toujours trouver un compromis. L'aspect any-time et la prise en compte du contexte ne nous est pas utile. La présentation des résultats sous forme de hiérarchie peut être modifiée pour apparaître sous forme de partition en utilisant un seuil de similarité. Nous le montrons dans la section suivante.

Connaissance a priori	nombre de groupes ou seuils
Présentation des résultats	hiérarchie
Complexité	$O(M \times N^2)$
Déterministe	oui
Incrémentale	non
Any-time	non
Chevauchement	non
Prise en compte du contexte	non
Tolérance :	
au bruit	non
à l'effet de chaîne	non
aux groupes de tailles variées	oui
aux groupes de densités variées	oui
aux groupes de forme quelconque	oui
aux groupes concentriques	oui

Table 24 – Caractéristiques de la méthode de clustering hiérarchique

5.3.2. Principe de l'algorithme de CAH

Contrairement à l'algorithme de clustering descendant hiérarchique qui prend en entrée tous les documents sous forme d'un seul groupe et qui effectue des divisions successives, l'algorithme de clustering ascendant hiérarchique considère chaque document comme un groupe et effectue des regroupements

successifs. A chaque itération de l'algorithme, les deux groupes les plus proches sont fusionnés. Cela continue jusqu'à ce qu'il n'y ait plus qu'un seul groupe. Le schéma de classification obtenu est un dendrogramme (Figure 19).

La légende utilisée sera la même pour les prochaines illustrations de ce mémoire représentant une base de documents (Table 25).

	Groupe
	Groupe considéré
	Document
	Document considéré
	Représentant de groupe, appelé centroïde
	Centroïde de groupe considéré
	Similarité avérée entre deux documents, deux centroïdes ou un document et un centroïde
	Calcul de similarité entre deux documents, deux centroïdes ou un document et un centroïde

Table 25 – Légende des illustrations représentant une base de documents

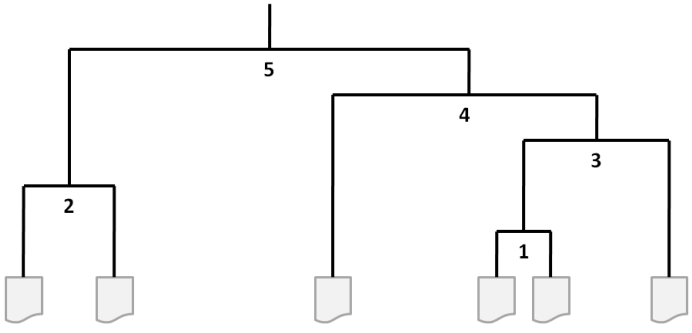


Figure 19 – Exemple de dendogramme obtenu par l'algorithme de CAH

Pour obtenir des groupes, il suffit de choisir une valeur de seuil de similarité qui élague l'arbre en « coupant » le dendrogramme. Les documents dont la

similarité entre supérieure au seuil seront regroupés, ceux dont la similarité est inférieure au seuil seront dans des groupes distincts (Figure 20).

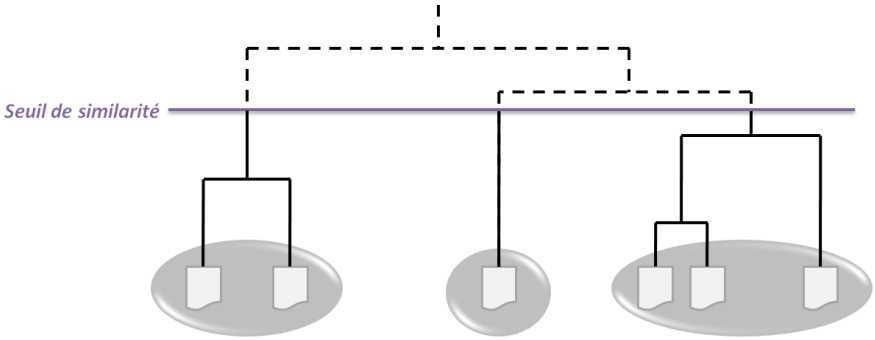


Figure 20 – Exemple de groupes obtenus avec l'algorithme de CAH

5.3.3. Méthodes de CAH

5.3.3.1. La méthode single-link

Dans cette méthode, aussi appelée MIN, la similarité entre deux groupes est la similarité la plus élevée (ou la distance minimale) entre deux documents issus chacun d'un des groupes (Figure 21). Au départ, tous les documents sont isolés, les similarités deux à deux entre chacun d'eux sont calculées. Puis des regroupements sont effectués si deux documents ont une similarité supérieure à un seuil prédéterminé. L'algorithme étant déterministe, l'ordre des regroupements n'a aucune importance. Cette méthode souffre de l'effet de chaîne et produit donc des groupes volumineux qui peuvent être relativement hétérogènes. Sa complexité est en $O(N^2)$ due à la comparaison entre tous les documents deux à deux, N étant le nombre de documents.

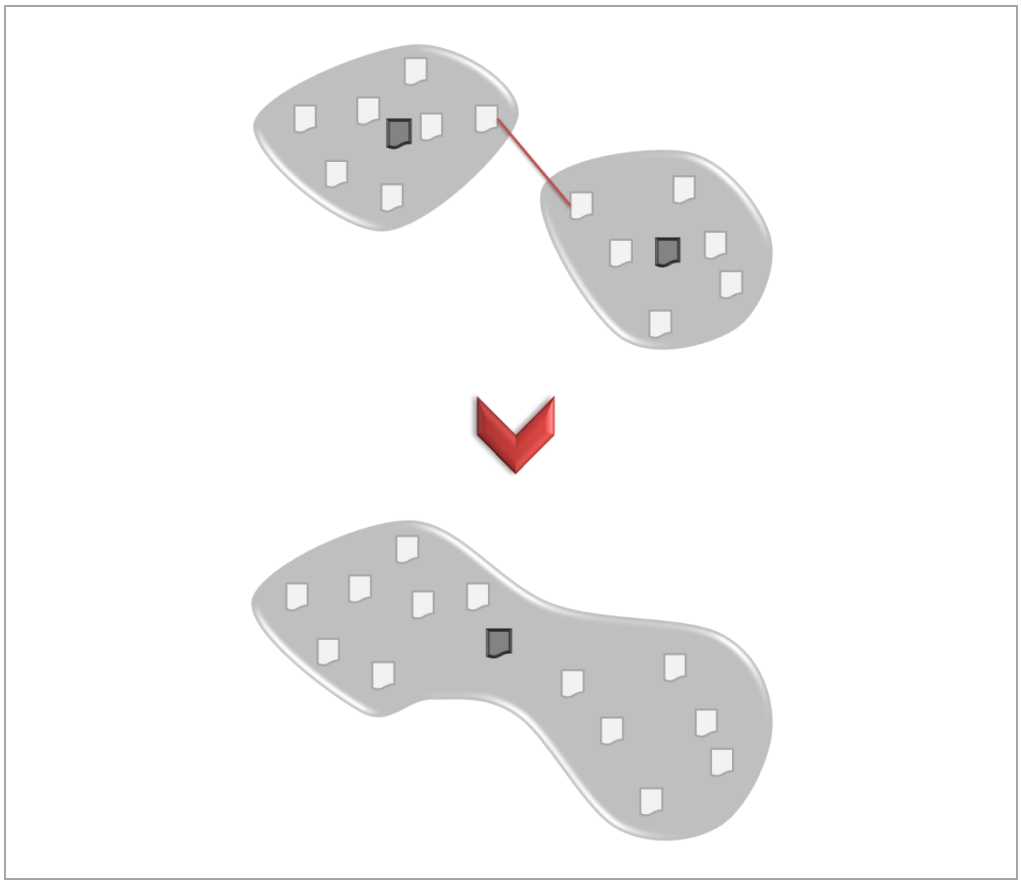


Figure 21 – Regroupement par la méthode single-link

Formellement, la distance minimale entre deux groupes G_1 et G_2 s'exprime de la manière suivante :

$$min\{d(G_1, G_2)\} = min\{d(x, y) : x \in G_1, y \in G_2\} \quad \textbf{(14)}$$

5.3.3.2. La méthode complete-link

Dans cette méthode, la similarité entre deux groupes est la similarité la plus faible (ou la distance maximale) entre deux documents issus chacun d'un des groupes (Figure 22). En comparaison de la méthode single-link, on se rend compte qu'avec un même jeu de données en entrée, il est nécessaire d'avoir un seuil de similarité plus faible avec la méthode complete-link pour produire les mêmes groupes. Au départ, tous les documents sont isolés, les similarités deux à deux entre chacun d'eux sont calculées. Puis des regroupements sont effectués si tous les documents de deux groupes ont une similarité supérieure à un seuil déterminé à l'avance. Selon l'ordre des regroupements, les groupes obtenus peuvent être différents. Les regroupements sont donc effectués en parcourant les similarités entre documents de la plus élevée à la plus faible (au-dessus du seuil). Cette méthode ne souffre pas de l'effet de chaîne et produit plus de groupes que la précédente. Ceux-ci sont plus compacts, plus

homogènes. Sa complexité est en $O(N^2.\log(N))$, N étant le nombre de documents.

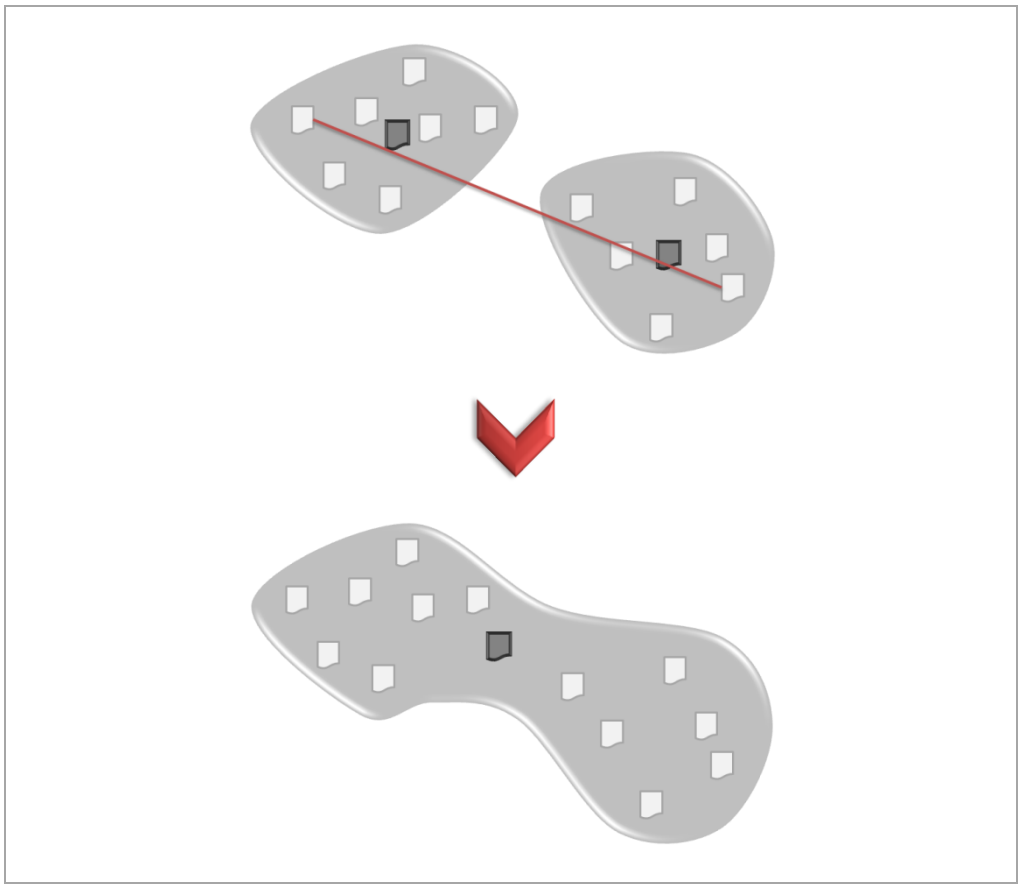


Figure 22 – Regroupement par la méthode complete-link

Formellement, la distance minimale entre deux groupes G_1 et G_2 s'exprime de la manière suivante :

$$max\ \{d(G_1, G_2)\} = max\ \{d(x, y) : x \in G_1, y \in\ G_2\} \quad \textbf{(15)}$$

5.3.3.3. La méthode incrémentale

Dans cette méthode, la similarité n'est pas calculée entre groupes mais entre chaque document pris séquentiellement et les centroïdes des groupes obtenus au moment où le document est considéré (Figure 23). Dans cette méthode, il n'est pas utile d'effectuer l'étape préliminaire de calcul de similarité entre documents deux à deux, les documents sont intégrés dans les groupes au fur et à mesure de leur entrée. De plus, les centroïdes de groupes sont nécessairement calculés puisqu'ils interviennent dans l'établissement des groupes. Cette méthode souffre de l'ordonnancement des documents mais voit sa complexité grandement réduite à $O(N.\log(N))$, N étant le nombre de documents.

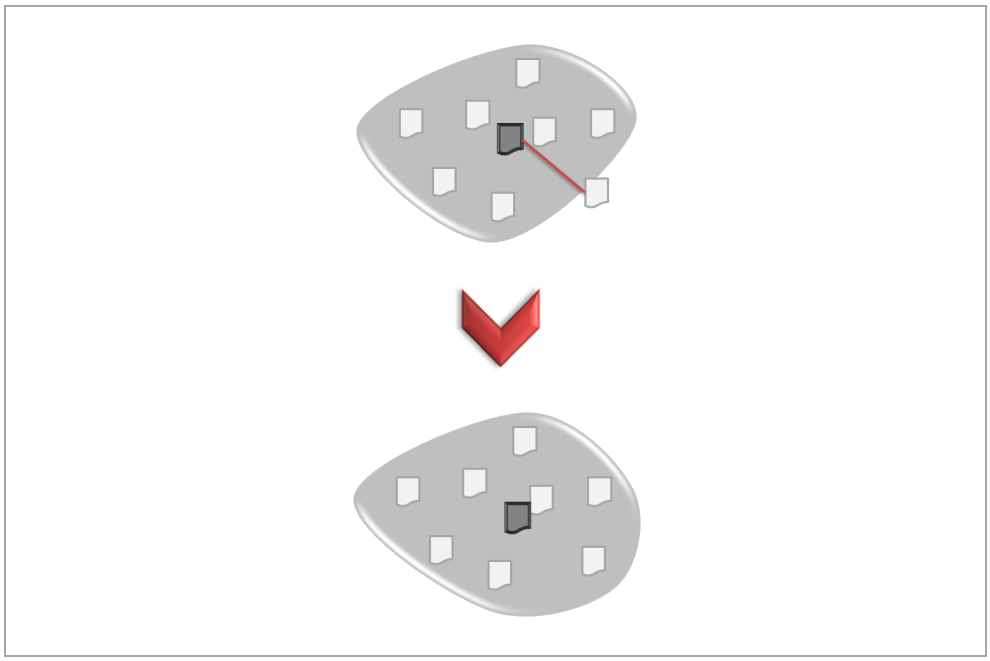


Figure 23 – Regroupement par la méthode incrémentale

5.3.4. Proposition de modification de l'algorithme de CAH incrémental

5.3.4.1. Problème induit par la méthode incrémentale

Le clustering incrémental est une variante du clustering (non-incrémental à l'origine) qui répond aux problèmes de traitement de corpus volumineux de documents et d'utilisation de corpus dynamiques. En effet, le clustering incrémental réduit grandement la complexité sur le temps et permet ainsi un temps d'exécution de l'algorithme bien plus court que dans sa version non-incrémentale. De plus, il autorise l'ajout de nouveaux documents en considérant les groupes déjà obtenus au lieu de devoir relancer son exécution sur l'ensemble des documents de la base. De nombreux algorithmes de clustering incrémental ont ainsi été proposés : l'algorithme K-means dans une version incrémental (Serban et al., 2005), CLASSIT (Gennari et al., 1989) ou encore COBWEB (Fisher, 1987a), (Fisher, 1987b) qui définit un clustering conceptuel dans lequel chaque groupe est considéré comme un modèle qui le décrit plutôt que comme une collection de documents qu'il regroupe. ICHAMELEON (He et al., 2006) est une version incrémentale de l'algorithme de clustering hiérarchique CHAMELEON. Nous pouvons aussi relever des algorithmes basés sur la densité tels que ICGD (Chen et al., 2007) ou partant de l'algorithme CADD (Song et al., 2009). Concernant le clustering hiérarchique incrémental, la méthode consiste uniquement à intégrer le nouveau document dans le groupe avec lequel il est le plus proche même s'il est aussi très proche d'autres groupes. Lors de l'ajout d'un nouveau document,

on effectue un calcul de distance (la distance euclidienne par exemple), entre le nouveau document et les centroïdes des groupes à partir de la racine du dendrogramme. Une fois que la distance la plus courte est trouvée, le document est inséré dans le dendrogramme (Ribert et al., 1999 ; Gurrutxaga et al., 2009). Lors de l'ajout d'un document, trois cas peuvent se présenter et sont habituellement traités de la manière suivante :

- *Cas n°1. Toutes les similarités entre le document et les centroïdes sont inférieures au seuil de similarité* : un nouveau groupe est créé, composé uniquement du nouveau document qui en devient le centroïde.
- *Cas n°2. Une seule similarité entre le document et les centroïdes est supérieure ou égale au seuil de similarité* : le document est intégré dans ce groupe, le centroïde de ce groupe est actualisé.
- *Cas n°3. Plusieurs similarités entre le document et les centroïdes sont supérieures ou égales au seuil de similarité* : le document est intégré dans le groupe pour lequel sa similarité avec le centroïde est la plus élevée, le prototype de ce groupe est actualisé.

Cependant, l'inconvénient du clustering incrémental est que l'ordre dans lequel les documents sont ajoutés induit un biais dans la création et l'évolution des groupes. Des ordonnancements différents de documents produiront des groupes non seulement différents mais qui pourront perdre de leur pertinence au fur et à mesure de l'ajout de nouveaux documents. Pour résoudre ce problème, (Young et al., 2010) utilisent une mesure de pseudo-entropie au cours de l'exécution de l'algorithme. D'autres auteurs proposent des post-traitements consécutifs à l'exécution de l'algorithme. Certains consistent à vérifier si les groupes sont bien formés. Dans le cas où ils ne le sont pas, des documents sont réassignés d'un groupe à un autre (Shaw et al., 2009). D'autres consistent à reformer des groupes en réunissant deux groupes en un seul (*merging*) ou au contraire à en séparer un en deux groupes (*splitting*) (Barbu et al., 2005).

D'autres préoccupations sur les propriétés des groupes lors de l'ajout d'un nouveau document sont présentes dans la littérature. Lors de l'ajout d'un document dans le dendrogramme, (Widyantoro et al., 2002) veulent conserver une densité similaire entre les documents d'un groupe (homogénéité) et veulent que la densité d'un groupe soit toujours supérieure à celle de son parent (monotonie). (Young et al., 2010) utilisent deux métriques sur chaque centroïde, la déviation standard et le taux estimé de mise à jour, pour conserver des groupes stables, dont les centroïdes ne sont que peu modifiés à l'ajout d'un objet.

Comme tous ces auteurs, nous souhaitons diminuer l'importance de l'ordre de sélection des documents. Cependant, nous voulons appliquer notre traitement aux moments pertinents (pas de post-traitement long et coûteux) et sur les groupes pour lesquels ce traitement pourrait être pertinent (pas nécessairement sur tous les groupes). Pour cela, nous utilisons un déclencheur : le fait qu'au

cours de l'exécution de l'algorithme, un document possède une similarité avec plusieurs groupes supérieure au seuil de similarité. Le traitement consiste à tenter de fusionner les groupes pendant l'exécution de l'algorithme en utilisant un calcul de similarité entre centroïdes pour déterminer si des groupes sont proches et doivent être fusionnés.

Le seuil de similarité utilisé pour déterminer si un document est similaire au centroïde d'un groupe et si deux centroïdes de groupe sont similaires est déterminé de manière expérimentale. En effet, en fonction du domaine d'étude, de la manière dont sont renseignées les valeurs dans les attributs, le seuil de similarité pour lequel les groupes obtenus sont optimaux (dans le sens où ils correspondent à ce qu'un expert pourait attendre en construisant lui-même les groupes) varie. Pour chaque base, l'étude du seuil de similarité est effectuée sur un échantillon. Elle détermine le seuil de similarité sur le problème et celui sur la solution, les deux pouvant être différents au sein d'une même base.

5.3.4.2. Apport à l'algorithme

Le déclencheur de fusion de groupes est activé lorsqu'un nouveau document à intégrer possède une similarité avec plusieurs groupes supérieure au seuil de similarité. Lors de l'ajout d'un document, nous avons relevé trois cas présentés précédemment. Lorsque le document sélectionné n'a de similarité (suffisante) avec aucun des groupes déjà présents, il devient le centroïde d'un nouveau groupe (cas n°1). S'il a une similarité suffisante avec un seul groupe, il est intégré à celui-ci et le centroïde de ce groupe est mis à jour avec les informations du document (cas n°2). Le point de notre amélioration porte sur le cas où un document possède une similarité suffisante avec plusieurs groupes (cas n°3). Dans ce cas, nous voyons trois possibilités :

- Fusion des groupes considérés en un seul groupe, intégration du document dans celui-ci et création du centroïde. A la manière de la méthode single-link, cette solution peut produire un effet de chaîne. Il serait donc nécessaire de tester les groupes finaux pour séparer ceux qui seraient hétérogènes (post-traitement de *splitting* des groupes).
- Intégration du document dans le groupe pour lequel sa similarité est maximale et mise à jour du centroïde. C'est la solution couramment utilisée. A la manière du complete-link, cette solution produit de petits groupes homogènes mais, du fait de l'ordre de sélection des documents, il pourrait être nécessaire de tester les groupes finaux pour regrouper ceux qui seraient très proches (post-traitement de *merging* des groupes).
- Tentative de fusion des groupes en calculant la similarité entre leurs centroïdes. Cette mesure est appelée distance inter-groupes. Si la valeur de cette mesure est supérieure au seuil de similarité, les groupes sont

fusionnés et le document est intégré dans ce nouveau groupe sinon le document est intégré dans le groupe pour lequel sa similarité avec le centroïde est la plus élevée.

Le troisième point est la solution qui a été retenue. Nous proposons l'algorithme suivant. Lorsqu'un nouveau document *D* possède une similarité avec plusieurs groupes supérieure au seuil :

1. Calcul des distances inter-groupes entre eux, c'est-à-dire évaluation de la similarité entre les centroïdes deux à deux (distance = 1 – similarité).
2. Si la similarité maximale obtenue entre deux groupes est supérieure au seuil, fusion de ces deux groupes.
3. Répétition des étapes 1. et 2. jusqu'à ce qu'aucune fusion ne se produise ou qu'il ne reste qu'un seul groupe.
4. Calcul de similarité entre le document *D* et les nouveaux groupes obtenus.
5. Intégration du document *D* dans le groupe pour lequel la similarité est maximale.

Formellement, soit *D* un nouveau document qui possède une similarité *S* supérieure au seuil de similarité σ avec plusieurs centroïdes $G_{1_c} \ldots G_{n_c}$, centroïdes respectifs des groupes $G_1 \ldots G_n$.

Soit $\gamma = \{G_i \mid S(D, G_{i_c}) > \sigma\}$

On désigne par $\oplus$ l'ajout d'un groupe dans γ.

$$\forall i, \forall j, G_i \in \gamma, G_j \in \gamma, max(S(G_{i_c}, G_{j_c})) > \sigma$$

$$\Rightarrow \gamma = \gamma \setminus \{G_i, G_j\} \oplus \{G_i \cup G_j\}$$

$$\exists k, G_k \in \gamma, max_{k=1}^{n}(S(D, G_{k_c})) \Rightarrow G_k = G_k \cup \{D\}$$

Pour compléter cet algorithme, nous pouvons ajouter un post-traitement identique à celui de la méthode complete-link, c'est-à-dire, déterminer pour chaque groupe sa similarité avec le groupe dont il est le plus proche (mesure de distance minimum inter-groupes). Cette étape peut s'avérer utile si deux groupes doivent être fusionnés mais qu'aucun document possédant une similarité avec ces deux groupes supérieure au seuil de similarité ne s'est présenté. Cependant, c'est une étape très coûteuse puisqu'elle compare tous les groupes deux à deux et qui rend l'utilisation de l'algorithme proposé inutile puisque cette dernière étape effectuera tous les regroupements possibles. L'utilisation de l'algorithme proposé permettra simplement de réduire le nombre de groupes donc de comparaisons entre groupes de cette dernière étape.

5.3.4.3. Algorithme proposé

Début
Initialisation de la liste des centroïdes listeCentroïdes à vide

Parcours des documents *D*
Initialisation de la liste des centroïdes similaires au document courant ListeCentroïdesSim à vide

// parcours des centroïdes courants pour connaître ceux avec lesquels le document a une similarité suffisante
Parcours de listeCentroïdes
sim = similarité entre le document courant et le centroïde courant C_C du groupe G_C
Si sim ≥ seuil de similarité **Alors**
Ajout du centroïde C_C dans ListeCentroïdesSim
Fin si
Fin parcours

// si aucune similarité suffisante, le document devient un groupe
Si ListeCentroïdesSim est vide **Alors**
Création d'un nouveau groupe *G* dont le document est le centroïde
Ajout de *G* dans listeCentroïdes
Sinon

// si une seule similarité suffisante, le document est intégré au groupe
Si ListeCentroïdesSim ne contient qu'un centroïde C_C **Alors**
Ajout du document *D* au groupe G_C dans listeCentroïdes
Mise à jour du centroïde C_C avec les informations du document dans listeCentroïdes

// si plusieurs similarités suffisantes, tentative de fusion des groupes avant d'intégrer le document
Sinon
fusion = vrai
// fusion des groupes si la similarité entre eux est suffisante
Tant que fusion **Et** ListeCentroïdesSim contient plus d'un centroïde **Faire**
maxSimCentroïdeCentroïde = similarité maximale obtenue entre deux centroïdes C_1 (du groupe G_1) et C_2 (du groupe G_2) de ListeCentroïdesSim
Si maxSimCentroïdeCentroïde ≥ seuil de similarité **Alors**
Fusion des deux groupes G_1 et G_2 de listeCentroïdes
Création du centroïde fusion C_f dans listeCentroïdes
Suppression de C_1 et C_2 de ListeCentroïdesSim
Ajout du nouveau centroïde C_f dans ListeCentroïdesSim
Sinon
fusion = faux
Fin si
Fin tant que

// recalcul des similarités entre le document et les centroïdes des groupes
maxSimDocumentCentroïde = 0
Parcours de ListeCentroïdesSim
sim = similarité entre le document courant *D* et le centroïde courant C_c du groupe G_c
Si sim > maxSimDocumentCentroïde **Alors**
maxSimDocumentCentroïde = sim
Fin si

Fin parcours

// Ajout du document dans le groupe
Ajout du document D au groupe G_c dans listeCentroïdes
Mise à jour du centroïde C_c avec les informations du document D dans listeCentroïdes

Fin si

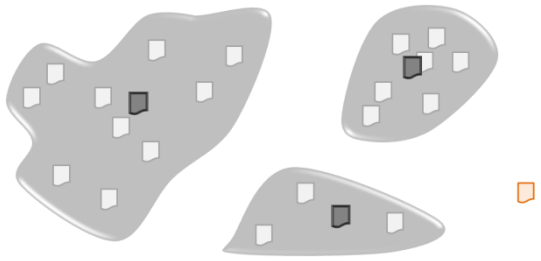
Fin si

Fin parcours

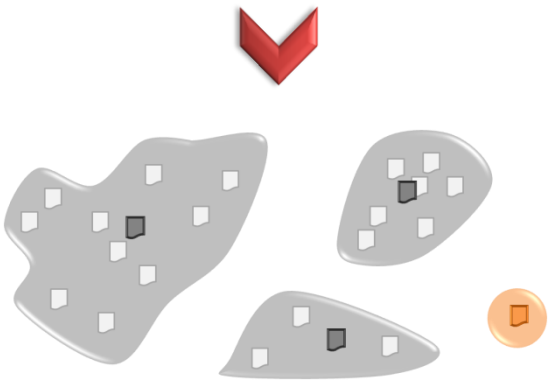
Fin

5.3.4.4. Illustration des trois cas possibles

Cas n°1. Le document n'est similaire à aucune centroïde : création d'un nouveau groupe (Figure 24).



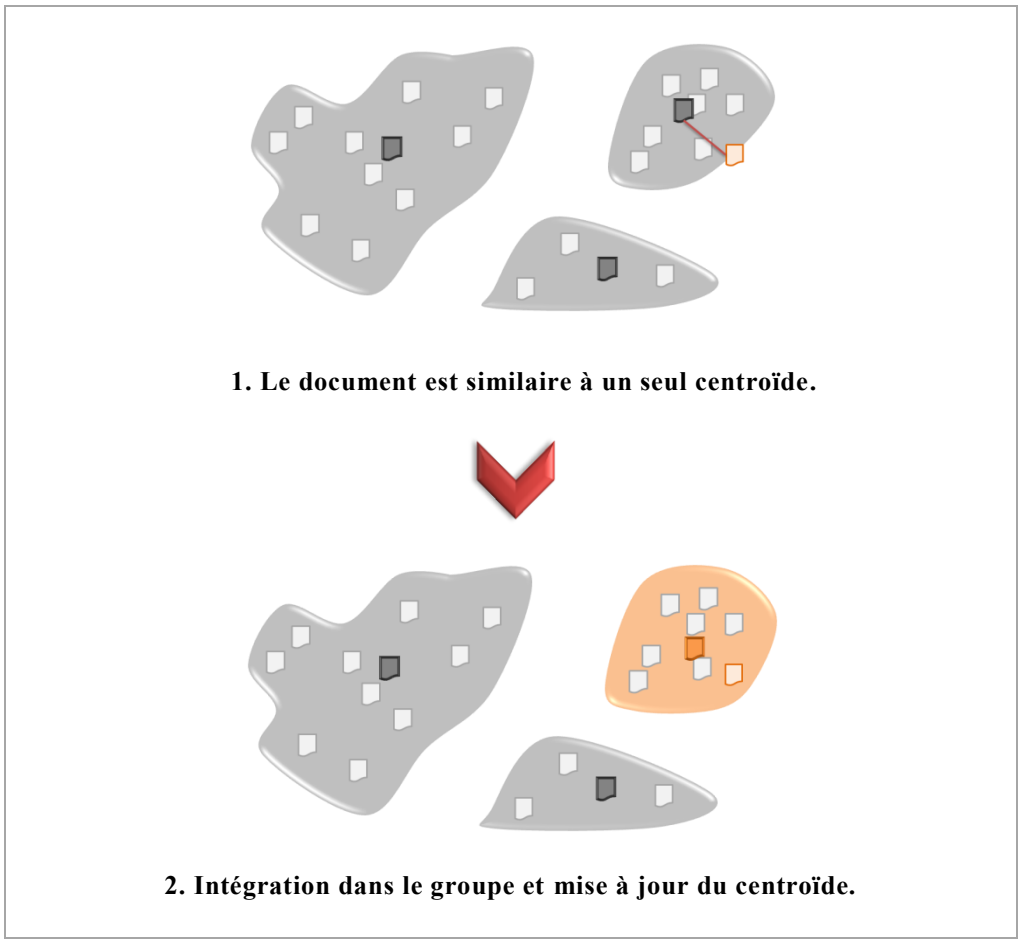
1. Le document n'est similaire à aucun centroïde.



2. Création d'un nouveau groupe dont il devient le centroïde.

Figure 24 – Intégration d'un document similaire à aucun centroïde

Cas n°2. *Le document est similaire à un seul centroïde* : intégration dans le groupe (Figure 25).

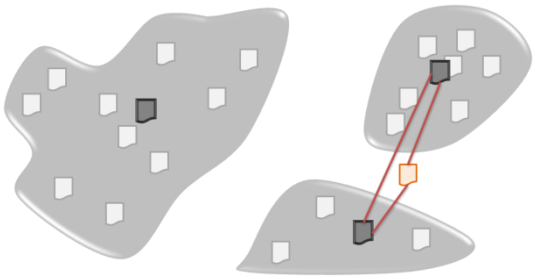


1. Le document est similaire à un seul centroïde.

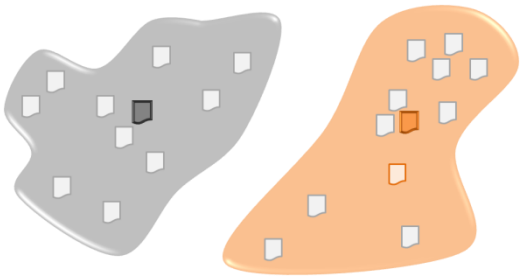
2. Intégration dans le groupe et mise à jour du centroïde.

Figure 25 – Intégration d'un document similaire à un seul centroïde

Cas n°3. *Le document est similaire à plusieurs centroïdes* : tentative de fusion des groupes (Figure 26 et Figure 27).

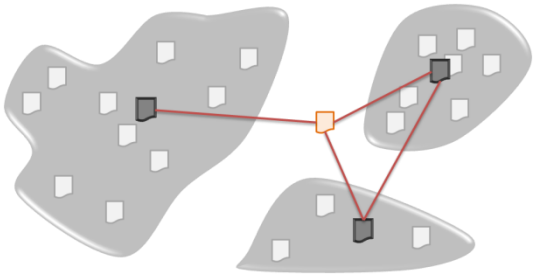


1. Le document est similaire à deux centroïdes similaires entre eux.

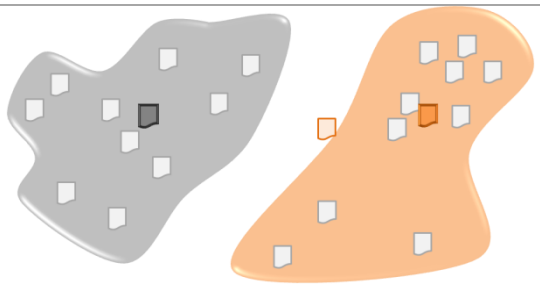


2. Fusion des deux groupes, intégration du document et création d'un centroïde.

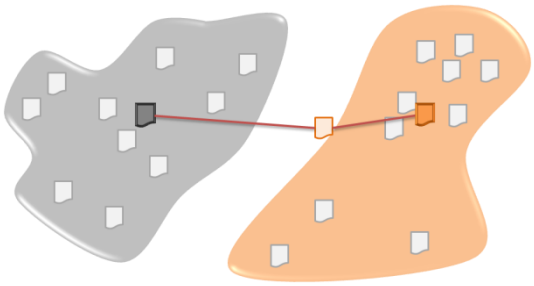
Figure 26 – Intégration d'un document similaire à plusieurs centroïdes – Exemple 1



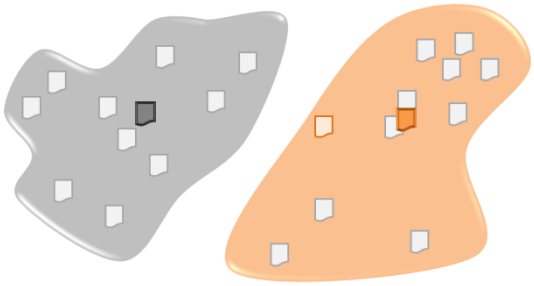
1. Le document est similaire à plusieurs centroïdes, la similarité maximale entre 2 centroïdes est supérieure au seuil de similarité.



2. Fusion des deux groupes et mise à jour du centroïde.



3. Le document est similaire à 2 centroïdes mais ces 2 centroïdes ne sont pas similaires entre eux.



4. Intégration du document dans le groupe le plus proche et mise à jour du centroïde.

Figure 27 – Intégration d'un document similaire à plusieurs centroïdes – Exemple 2

5.3.4.5. Forces et faiblesses de l'algorithme proposé

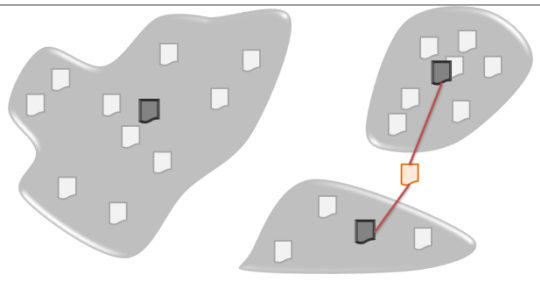
Nous voyons deux avantages majeurs dans notre proposition : le fait que nous définissions les conditions de déclenchement de la fusion de deux groupes ou

plus et le fait de déterminer quels groupes sont suffisamment similaires pour être potentiellement fusionnés.

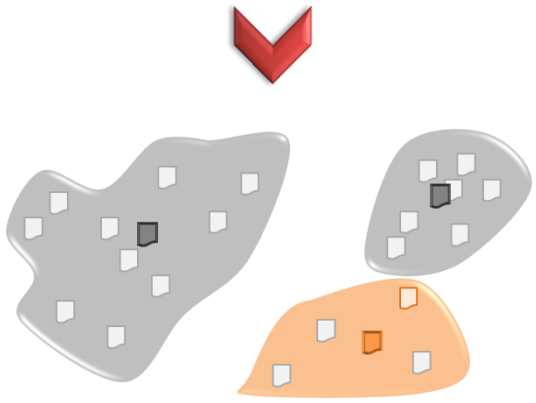
D'une part, utilisant la méthode incrémentale, nous ne souhaitions pas faire seulement un post-traitement. Nous reduisons le biais de l'ordre de sélection au cours de la création et de l'évolution des groupes durant l'algorithme. Nous ne voulions pas non plus exécuter une tentative de fusion des groupes selon un certain nombre ou un certain pourcentage de documents sélectionnés ou bien selon des jalons temporels. Nous proposons donc de chercher à fusionner des groupes lorsqu'un document possède une similarité avec plusieurs groupes supérieure au seuil de similarité, ce qui nous différencie des approches standards intégrant automatiquement le document au groupe avec lequel la similarité est maximale.

D'autre part, contrairement à un post-traitement au cours duquel tous les groupes obtenus devraient être comparés deux à deux, la recherche de fusion de groupes ne porte que sur un nombre restreint de groupes. Ce nombre restreint de groupes est pertinent puisque ceux-ci ont une similarité supérieure au seuil de similarité avec un même document. Ils sont donc potentiellement similaires entre eux.

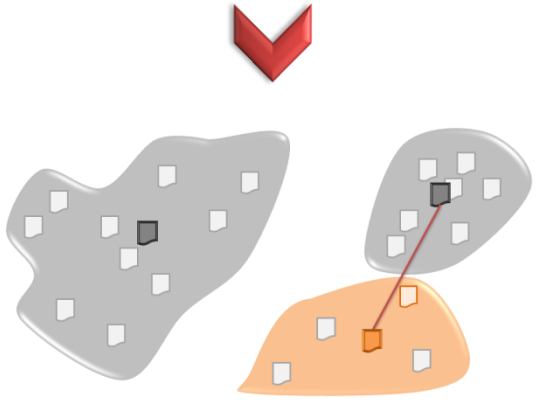
En revanche, nous pouvons relever une certaine faiblesse dans l'algorithme que nous proposons. Elle intervient lors de l'ajout d'un document dans un groupe alors que celui-ci est similaire à plusieurs centroïdes de groupe mais que ces centroïdes ne sont pas suffisamment similaires entre eux pour fusionner leurs groupes. Nous tentons de fusionner des groupes avant l'ajout d'un document mais pas après l'ajout de ce document dans un groupe. Pourtant, son ajout modifie la position du centroïde du groupe et peut donc le rapprocher d'un autre centroïde de groupe (Figure 28 – les étapes 1 et 2 sont effectuées par l'algorithme proposé, les étapes 3 et 4 adresseraient la faiblesse relevée). Nous pensons cependant que la similarité n'est pas grandement modifiée. La fusion après ajout d'un document serait assez rare. De plus, avec notre proposition, ces groupes pourraient être fusionnés plus tard si un nouveau document est similaire aux centroïdes de ces groupes.



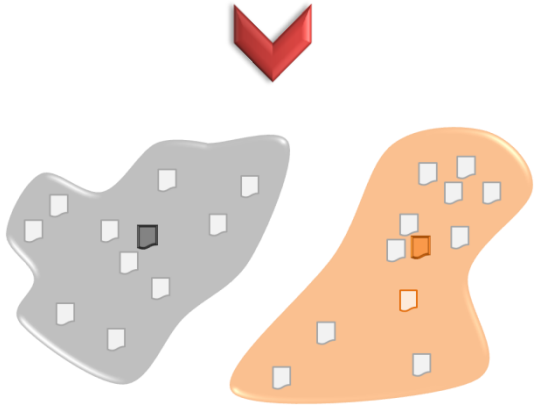
1. **Le document est similaire à deux centroïdes mais la similarité entre ces centroïdes est inférieure au seuil.**



2. Intégration du document dans le groupe le plus proche et mise à jour du centroïde.



3. Le nouveau centroïde est similaire au centroïde duquel le document ajouté était aussi similaire.



4. Fusion des deux groupes et mise à jour du centroïde.

Figure 28 – Faiblesse de l'algorithme proposé

5.4. Indicateurs de qualité des résultats

5.4.1. Pourquoi des indicateurs ?

Un algorithme de clustering produit des groupes selon des critères de similarité entre les éléments des groupes. Cependant, il ne donne que peu ou pas d'information sur la qualité de la classification obtenue. L'utilisation d'indicateur a pour but de faciliter le travail d'analyse de la classification par un expert pour connaître les documents récurrents, comprendre pourquoi certains groupes sont volumineux et d'autres non et savoir quels sont les groupes parmi lesquels il peut avoir confiance. Nous avons retenu différents indicateurs que l'on peut agréger selon quatre types : les indicateurs de document, de groupe, de gain et de moyenne. Pour les sélectionner, nous explorons l'ensemble des indicateurs en nous intéressant particulièrement à ceux qui décrivent les groupes selon l'une des deux dimensions importantes définies par (Dubes et al., 1979) : la compacité qui mesure la cohésion des documents d'un groupe par rapport aux documents des autres groupes et l'isolation qui mesure l'écart entre un groupe et les autres groupes. La compacité calculée avec des distances doit être minimisée alors que l'isolation, toujours calculée avec des distances, doit être maximisée. Les indicateurs retenus ainsi que leur mode de calcul ont été sélectionnés dans les travaux de (Raskutti et al., 1999) et (Nguyen et al., 2008).

5.4.1.1. Les indicateurs de document

Les indicateurs de document indiquent la pertinence de l'appartenance d'un document à son groupe. Un document peut être correctement classé en cours d'exécution de l'algorithme mais les groupes peuvent évoluer de sorte qu'à la fin de l'exécution le document devrait être placé dans un autre groupe. La silhouette est l'indicateur retenu pour jouer ce rôle.

5.4.1.2. Les indicateurs de groupe

Au-delà des simples indicateurs de taille et taille relative, les indicateurs de groupe retenus traduisent la qualité du groupe lui-même. Ils se divisent en 2 parties, les indicateurs intra-groupes qui s'intéressent au groupe lui-même, à sa constitution indépendamment des autres groupes et les indicateurs inter-groupes qui s'intéressent à la position du groupe par rapport aux autres groupes. La densité et le radius sont des indicateurs intra-groupes qui informent respectivement de la compacité du groupe (les documents sont-ils proches les uns des autres ?) et de l'homogénéité du groupe (y a-t-il une valeur

extrême, un document très isolé par rapport aux autres ?). Les indicateurs inter-groupes de distance informent de l'isolement et de la centralité d'un groupe.

5.4.1.3. Les indicateurs de gain

Les indicateurs de gain informent de l'intérêt des regroupements effectués. Des groupes peuvent être créés sans pour autant apporter un réel avantage par rapport à l'absence de groupes. Parmi les indicateurs de groupe, seuls la densité et le radius ont été retenus pour évaluer leur apport, ce sont les indicateurs les plus adaptés pour cette évaluation. Pour être pertinents, les groupes obtenus doivent avoir une meilleure densité et un radius plus faible qu'en l'absence de classification.

5.4.1.4. Les indicateurs de moyenne

Les indicateurs de moyenne reprennent les indicateurs de groupe et permettent d'avoir une idée générale sur les groupes contenus à l'intérieur d'un groupe parent. Moins qualitatifs que les indicateurs précédents, ils sont à rapprocher d'indicateurs statistiques.

5.4.2. Deux modes de calcul

Un même indicateur peut être calculé de deux façons. Il peut être soit le résultat du calcul de distances entre documents deux à deux soit le résultat du calcul entre les documents et le centroïde du groupe. Le premier mode de calcul est précis mais coûteux puisqu'il nécessite de connaître au préalable toutes les distances entre documents deux à deux. Le deuxième est bien plus rapide mais donne des résultats plus approximatifs.

Notre contexte d'application, le traitement d'anomalies qui surviennent au cours de processus industriels de production, implique un nombre important de fiches d'anomalie à considérer. Comme nous le verrons dans le chapitre suivant (§6.3.1.3 et §6.3.2.3), les méthodes non-incrémentales single-link et complete-link de l'algorithme de clustering ascendant hiérarchique ne sont pas utilisables sur un corpus volumineux. Ces méthodes ne supportent pas le passage à l'échelle. En conséquence, nous appliquons les méthodes incrémentales. Cela implique d'une part que nous ne calculons pas donc ne connaissons pas les distances entre documents deux à deux. D'autre part, les centroïdes des groupes sont nécessairement calculés durant l'exécution des méthodes incrémentales. Nous allons donc nous intéresser particulièrement au deuxième mode de calcul qui, comme dans le cadre de l'utilisation des méthodes de clustering ascendant hiérarchique, seront à-même de supporter le passage à l'échelle. Cependant, nous présentons les deux formules de calcul

pour plusieurs indicateurs afin de montrer que le deuxième mode de calcul est une approximation satisfaisante du premier mode.

Par la suite, pour exprimer les formules nous utilisons les notations suivantes :

G *représente le groupe considéré.*

G_i *représente un groupe.*

G_D *représente le document D considéré du groupe* G.

G_C *représente le centroïde du groupe* G.

P_G *représente le groupe parent du groupe* G.

$\delta(x, y)$ *représente la distance entre les documents* x *et* y.

$|G|$ *représente le nombre de documents dans le groupe* G.

Dans le cadre de notre framework, le groupe P_G parent du groupe G est une catégorie si G est un groupe de problèmes ou un groupe de problèmes si G est un groupe de solutions.

5.4.3. Les indicateurs retenus

Pour le calcul des indicateurs, chaque formule présentée dans cette section est précédée de deux informations :

- Une première information entre accolades pour indiquer le mode de calcul : *mode 1* correspond au calcul entre documents deux à deux, *mode 2* correspond au calcul entre documents et centroïdes.
- Une deuxième information entre crochets pour indiquer le type de l'indicateur : *document*, *groupe*, *gain* et *moyenne*. Cela permet de différencier deux indicateurs de même nom mais qui portent sur des éléments différents.

5.4.3.1. Les indicateurs de document

Parmi tous les indicateurs relatifs au document, nous n'en avons retenu qu'un seul : la silhouette. Pour le calculer, il est nécessaire de définir au préalable deux autres indicateurs utilisés en interne : la distance moyenne intra-groupe et la distance minimum inter-groupes. Eux aussi sont relatifs au document.

5.4.3.1.1. La distance moyenne intra-groupe (utilisée en interne pour le calcul de silhouette)

C'est la distance moyenne du document à tous les autres documents du groupe auquel il appartient (Figure 29). Elle s'exprime entre 0 et 1. Une distance intra-groupe proche de 0 indique que le document est similaire à la plupart des

documents de son groupe. A l'inverse, une distance intra-groupe proche de 1 indique que le document est éloigné de la plupart des documents du groupe.

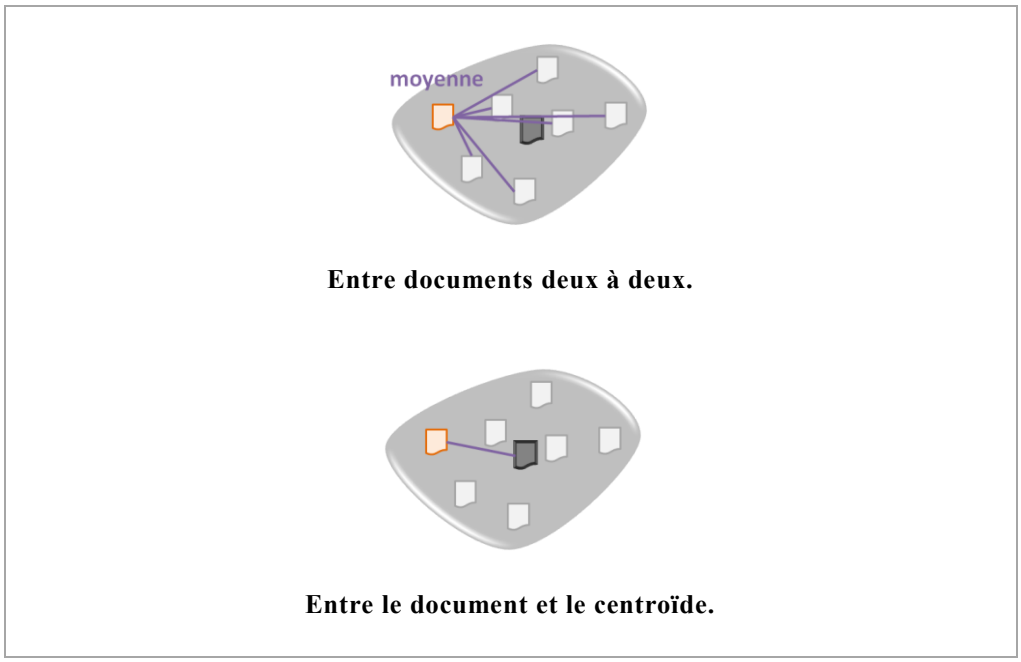


Figure 29 – Calcul de la distance moyenne intra-groupe d'un document

$$\{mode\ 1\}\,[document]distance\ moyenne\ intra\text{-}groupe(G_D) = \frac{\sum_{x \in G, x \neq G_D} \delta(G_D, x)}{|G| - 1} \quad \textbf{(16)}$$

Cas particulier : *division par 0*.

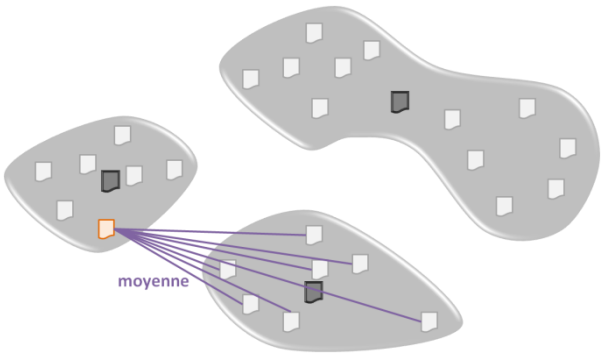
$|G| - 1 = 0$, ce qui implique que $|G| = 1$. Le document est seul dans son groupe,

$$\{mode\ 1\}\,[document]distance\ moyenne\ intra\text{-}groupe(G_D) = 0 \quad \textbf{(17)}$$

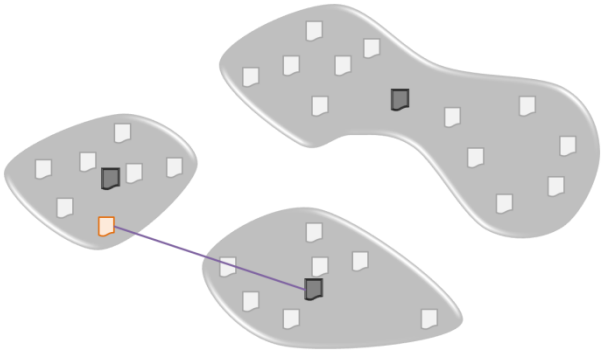
$$\{mode\ 2\}\,[document]distance\ moyenne\ intra\text{-}groupe(G_D) = \delta(G_C, G_D) \quad \textbf{(18)}$$

5.4.3.1.2. La distance minimum inter-groupes (utilisée en interne pour le calcul de silhouette)

C'est la plus petite moyenne des distances du document à tous les documents de chaque groupe auquel il n'appartient pas (Figure 30). C'est donc la moyenne des distances du document aux documents du groupe dont il est le plus proche. Elle s'exprime entre 0 et 1. Une distance inter-groupes proche de 0 indique que le document est similaire aux documents de ce groupe (auquel il n'appartient pas). A l'inverse, une distance inter-groupes proche de 1 indique que le document est éloigné de ce groupe. La valeur obtenue est la même quel que soit le mode de calcul (entre documents deux à deux ou par rapport au centroïde).



Entre documents deux à deux.



Entre le document et le centroïde.

Figure 30 – Calcul de la distance minimum inter-groupes d'un document

$$\{mode\ 1\}\ [document] distance\ minimum\ inter\text{-}groupes(G_D) = min_{i=1}^{k} \frac{\sum_{x \in G_i, G \neq G_i, P_G = P_{G_i}} \delta(G_D, x)}{|G_i|} \quad \textbf{(19)}$$

$$\{mode\ 2\}\ [document] distance\ minimum\ inter\text{-}groupes(G_D) = min_{i=1, G \neq G_i, P_G = P_{G_i}}^{k} \delta(G_{i_C}, G_D) \quad \textbf{(20)}$$

Cas particulier : *le document est présent dans le seul groupe G qui compose P_G.*

$|G_i| = 0$, le groupe est considéré comme pertinent, il est bien isolé,

$$\{mode\ 1\&2\}\ [document] distance\ minimum\ inter\text{-}groupes(G_D) = 1 \quad \textbf{(21)}$$

5.4.3.1.3. La silhouette

La silhouette d'un document indique à quel point le document appartient à son groupe plutôt qu'au groupe dont il est le plus proche. Elle s'exprime entre -1 et 1. Une silhouette proche de -1 indique que le document n'est pas bien classifié. Une silhouette proche de 0 indique que le document pourrait aussi bien appartenir à son groupe actuel qu'au groupe dont il est le plus proche. Une

silhouette proche de 1 indique que le document peut être considéré comme bien classifié.

Soient $a(G_D)$ la distance moyenne intra-groupe du document D dans le groupe G et $b(G_D)$ la distance minimum inter-groupes du document D du groupe G, distances calculées entre documents deux à deux,

$$\{mode\ 1\}\ [document]silhouette(G_D) = \frac{b(G_D) - a(G_D)}{max\big(a(G_D), b(G_D)\big)} \qquad \textbf{(22)}$$

Soient $a_C(G_D)$ la distance moyenne intra-groupe du document D dans le groupe G et $b_C(G_D)$ la distance minimum inter-groupes du document D du groupe G, distances calculées en utilisant le centroïde,

$$\{mode\ 2\}\ [document]silhouette(G_D) = \frac{b_C(G_D) - a_C(G_D)}{max\big(a_C(G_D), b_C(G_D)\big)} \qquad \textbf{(23)}$$

Cas particulier : *division par 0*.

$max\big(a_C(G_D), b_C(G_D)\big) = 0$, donc $a_C(G_D) = 0$ et $b_C(G_D) = 0$. Si $a_C(G_D) = 0$ alors tous les documents sont identiques ou il n'y a qu'un document. Si $b_C(G_D) = 0$ alors il existe un autre groupe dont le centroïde est identique à celui de G car il n'est pas possible que G soit le seul groupe dans P_G sinon $b_C(G_D)$ = 1. Même s'il paraît surprenant, voire improbable, qu'un algorithme produise deux groupes différents mais ayant deux centroïdes identiques, le document pourrait tout aussi bien appartenir à chacun des deux groupes. Cette remarque s'applique aux deux modes de calcul,

$$\{mode\ 1\&2\}\ [document]silhouette(G_D) = 0 \qquad \textbf{(24)}$$

5.4.3.2. Les indicateurs de groupe

A l'exception des indicateurs de taille et de silhouette, tous ces indicateurs s'expriment entre 0 et 1.

La densité et le radius sont des indicateurs intra-groupes qui utilisent les distances, les valeurs obtenues doivent être minimisées pour obtenir des groupes pertinents. Ainsi, une valeur proche de 0 indique que les documents du groupe sont très similaires (groupe homogène). A l'inverse, une valeur proche de 1 indique que les documents du groupe sont très éloignés (groupe hétérogène).

La distance minimum inter-groupes et la distance moyenne inter-groupes sont des indicateurs inter-groupes, leur valeur doit être maximisée pour obtenir des groupes pertinents. Ainsi, une valeur proche de 1 indique que le groupe est très éloigné de son plus proche voisin, donc de tous les autres groupes (groupe distinct). A l'inverse, une valeur proche de 0 indique que le groupe est très proche de son plus proche voisin (groupes analogues).

5.4.3.2.1. La taille

La taille est le nombre de documents qui composent le groupe.

$$[groupe]taille(G) = |G| \quad \textbf{(25)}$$

5.4.3.2.2. La taille relative

La taille relative est le nombre de documents qui composent le groupe par rapport au nombre de documents dans le groupe parent.

$$[groupe]taille\ relative(G) = \frac{|G|}{|P_G|} \quad \textbf{(26)}$$

Cas particulier : *division par 0*.

$|P_G| = 0$, il n'y a pas de documents dans le groupe parent de G, donc pas de documents non plus dans G. Ces groupes n'ont pas lieu d'exister,

$$[groupe]taille\ relative(G) = null \quad \textbf{(27)}$$

5.4.3.2.3. La densité

La densité est le nombre moyen de documents par unité d'espace (Figure 31).

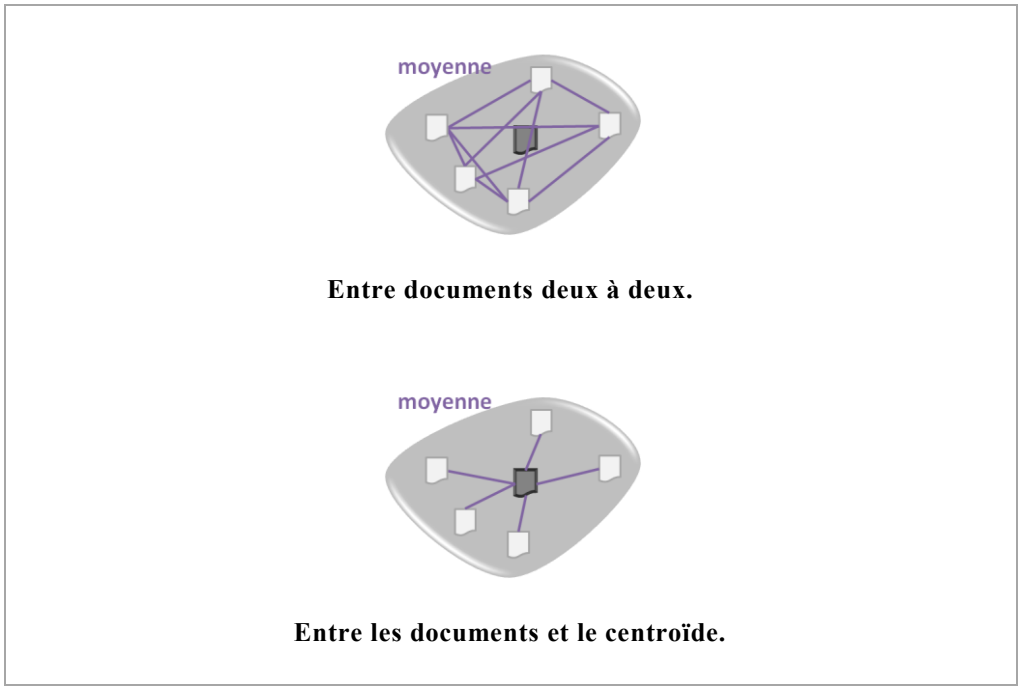


Figure 31 – Calcul de la densité d'un groupe

$$\{mode\ 1\}\ [groupe]densité(G) = \frac{\sum_{x,y \in G} \delta(x,y)}{|G|^2} \quad \textbf{(28)}$$

$$\{mode\ 2\}\ [groupe]densité(G) = \frac{\sum_{x \in G} \delta(G_C,x)}{|G|} \quad \textbf{(29)}$$

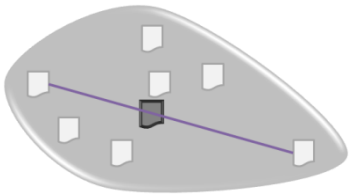
Cas particulier : *division par 0*.

$|G| = 0$, le groupe ne contient pas de documents, il n'a pas lieu d'exister,

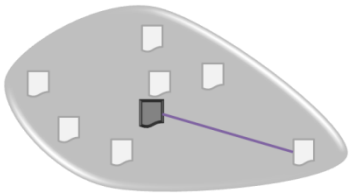
$$\{mode\ 1\&2\}\ [groupe]densité(G) = null \quad \textbf{(30)}$$

5.4.3.2.4. Le diamètre et le radius

Le diamètre d'un groupe est la distance maximale entre toute paire de documents du groupe. L'équivalent en utilisant les centroïdes est le radius, c'est-à-dire la distance maximale entre le centroïde et tous les documents du groupe (Figure 32). Dans un groupe homogène, le diamètre peut être estimé à 2 fois le radius.



Entre documents deux à deux (diamètre).



Entre les documents et le centroïde (radius).

Figure 32 – Calcul du diamètre et du radius d'un groupe

$$\{mode\ 1\}\ [groupe]diamètre(G) = max(\delta(x,y) : x,y \in G) \quad \textbf{(31)}$$

$$\{mode\ 2\}\ [groupe]radius(G) = max(\delta(G_C,x) : x \in G) \quad \textbf{(32)}$$

Cas particulier : *le groupe ne contient pas de documents*.

Le groupe n'a pas lieu d'exister,

$$\{mode\ 1\}\ [groupe]diamètre(G) = 0 \quad \textbf{(33)}$$

$$\{mode\ 2\}\ [groupe]radius(G) = 0 \quad \textbf{(34)}$$

5.4.3.2.5. La distance minimum inter-groupes

La distance minimum inter-groupes est la distance la plus petite entre le groupe considéré et tous les autres groupes contenus dans le groupe parent (Figure 33).

Ce calcul est le même que celui correspondant au document à ceci près que ce n'est pas un document qui est considéré mais le centroïde du groupe.

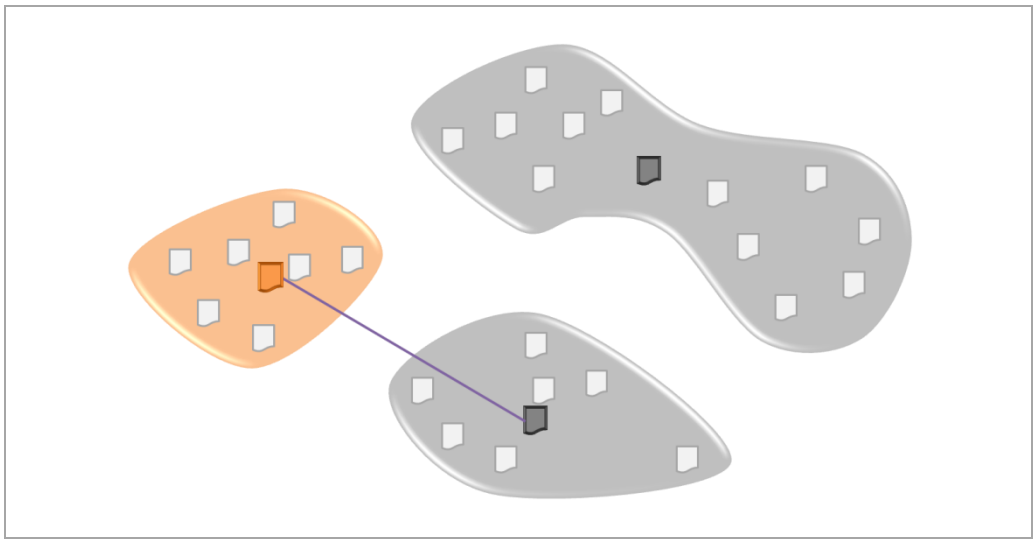


Figure 33 – Calcul de la distance minimum inter-groupes d'un groupe

$$\{mode\ 2\}\ [groupe] distance\ minimum\ inter\text{-}groupes(G) = min_{i=1, G \neq G_i,\ P_G = P_{G_i}}^{k} \delta(G_C, G_{i_C}) \quad \textbf{(35)}$$

Cas particulier : *le groupe parent P_G ne contient que ce groupe G.*

$$\{mode\ 2\}\ [groupe] distance\ minimum\ inter\text{-}groupes(G) = 1 \quad \textbf{(36)}$$

5.4.3.2.6. La distance moyenne inter-groupes

La distance moyenne inter-groupes est la distance moyenne entre le groupe considéré et tous les autres groupes contenus dans le groupe parent (Figure 34). Elle permet d'évaluer si le groupe est « au milieu de la galaxie » (valeur proche de 0), c'est-à-dire qu'il partage des points communs avec la plupart des autres groupes et est donc ordinaire ou s'il se situe « dans un bras de la galaxie» (valeur proche de 1), c'est-à-dire que même s'il partage des points communs avec quelques groupes, il est assez spécifique.

$$\{mode\ 2\}\ [groupe] distance\ moyenne\ inter\text{-}groupes(G) = \frac{\sum_{i=1,\ G \neq G_i,\ P_G = P_{G_i}}^{k} \delta(G_C, G_{i_C})}{k} \quad \textbf{(37)}$$

Cas particulier : *le groupe parent P_G ne contient que ce groupe G.*

$$\{mode\ 2\}\ [groupe] distance\ moyenne\ inter\text{-}groupes(G) = 1 \quad \textbf{(38)}$$

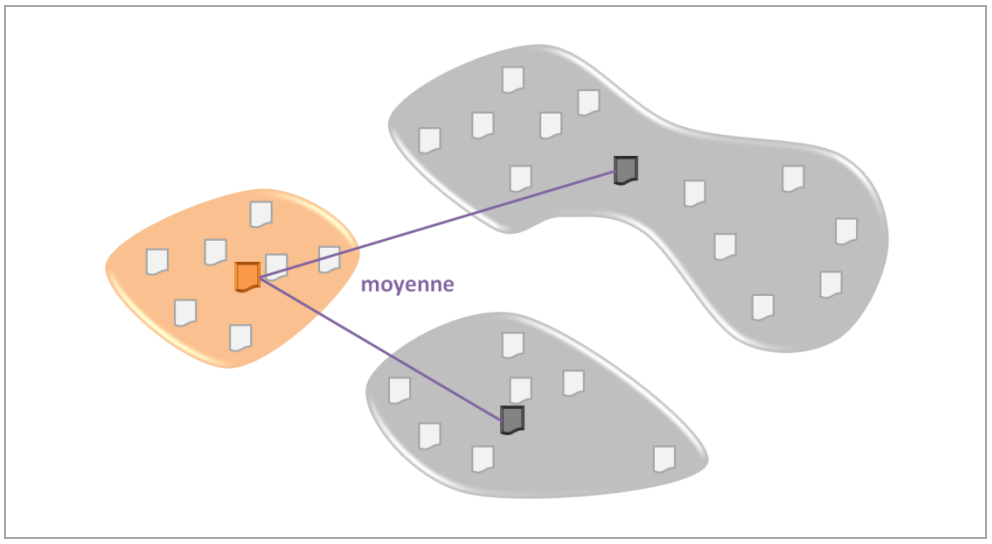


Figure 34 – Calcul de la distance moyenne inter-groupes d'un groupe

5.4.3.3. Les indicateurs de gain

Pour certains indicateurs, en plus de la valeur absolue, résultat brut de la formule de calcul utilisée indépendamment des autres éléments du corpus, un coefficient relatif est calculé pour évaluer l'intérêt des groupes obtenus par rapport à une absence de groupes. Ce coefficient relatif dépend d'une part de la valeur de l'indicateur du groupe ($valIndicGpe$) et d'autre part de la valeur de l'indicateur appliqué sur l'ensemble des documents du parent du groupe ($valIndicGpeParent$). Il permet de comparer la valeur de l'indicateur par rapport à celle de son groupe parent pour estimer si la création de ce groupe améliore l'indicateur. Utilisant des calculs de distance, les valeurs des indicateurs doivent être minimisées. La formule de calcul est la suivante :

$$gain = \frac{valIndicGpeParent - valIndicGpe}{valIndicGpeParent} \quad \textbf{(39)}$$

Nous définissons les coefficients relatifs de groupe de la densité et du radius. La valeur obtenue est comprise entre $-\infty$ et 1. Une valeur négative indique que le groupe considéré est de moins bonne qualité que si aucun groupe n'avait été créé, alors qu'une valeur positive indique qu'il est de meilleure qualité, que le regroupement est pertinent par rapport aux documents en entrée. Une valeur nulle indique que le regroupement n'a rien apporté. Cependant, ces coefficients relatifs nous intéressent pour estimer seulement l'amélioration apportée par le groupe. Ainsi, toute valeur négative ne sera pas considérée en tant que telle et se verra réduite à 0. La formule de calcul devient :

$$gain = 0 \; si \; valIndicGpeParent \leq valIndicGpe \quad \textbf{(40)}$$

$$gain = \frac{valIndicGpeParent - valIndicGpe}{valIndicGpeParent} \; sinon \quad (41)$$

Note : pas de division par 0 possible car $valIndicGpe$ est défini sur l'intervalle [0 ; 1]. Si $valIndicGpeParent$ lui est strictement supérieur, $valIndicGpeParent$ ne peut pas valoir 0. Il en est de même pour les coefficients relatifs de densité et de radius qui suivent.

5.4.3.3.1. Le coefficient relatif de densité

$$[gain]densité(G) = 0 \; si \; [groupe]densité(P_G) \leq [groupe]densité(G) \quad (42)$$

$$[gain]densité(G) = \frac{[groupe]densité(P_G) - [groupe]densité(G)}{[groupe]densité(P_G)} \; sinon \quad (43)$$

5.4.3.3.2. Le coefficient relatif de radius

$$[gain]radius(G) = 0 \; si \; [groupe]radius(P_G) \leq [groupe]radius(G) \quad (44)$$

$$[gain]radius(G) = \frac{[groupe]radius(P_G) - [groupe]radius(G)}{[groupe]radius(P_G)} \; sinon \quad (45)$$

5.4.3.4. Les indicateurs de moyenne

Ces indicateurs permettent d'évaluer la valeur obtenue pour un groupe en utilisant les valeurs obtenues par les groupes dont il est le parent. Ce ne sont pas des indicateurs qui portent sur un groupe lui-même mais plutôt sur un ensemble de groupes (rassemblés dans un groupe parent).

Pour les 4 indicateurs utilisant un ratio, une division par 0 est possible si le nombre de groupes à l'intérieur d'un groupe parent est nul. Dans ce cas, la valeur de l'indicateur est fixée à null.

5.4.3.4.1. Le nombre de groupes

C'est le nombre de groupes dont le groupe considéré est le parent.

$$[moyenne]nombre(G) = Count(G_i) : P_{G_i} = G \quad (46)$$

5.4.3.4.2. La taille moyenne de groupes

C'est le nombre moyen de documents par groupes dont le groupe considéré est le parent.

$$[moyenne]taille(G) = \frac{|G_i|}{Count(G_i)} : P_{G_i} = G \quad (47)$$

5.4.3.4.3. La densité moyenne

C'est la densité moyenne des groupes dont le groupe considéré est le parent.

$$[moyenne]densité(G) = \frac{\sum_{i=1,\, P_{G_i}=G}^{k} [groupe]densité(G_i)}{k} \quad \textbf{(48)}$$

5.4.3.4.4. Le radius moyen

C'est le radius moyen des groupes dont le groupe considéré est le parent.

$$[moyenne]radius(G) = \frac{\sum_{i=1,\, P_{G_i}=G}^{k} [groupe]radius(G_i)}{k} \quad \textbf{(49)}$$

5.4.3.4.5. La distance minimum inter-groupes moyenne

C'est la distance minimum inter-groupes moyenne des groupes dont le groupe considéré est le parent.

$$[moyenne]distance\ minimum\ inter\text{-}groupes(G) = \frac{\sum_{i=1\,,\, P_{G_i}=G}^{k} [groupe]distance\ minimum\ inter\text{-}groupes(G_i)}{k} \quad \textbf{(50)}$$

5.4.3.5. Synthèse sur les indicateurs

Nous avons effectué cette sélection d'indicateurs pour avoir des informations sur la qualité de la classification obtenue, dans notre cas celle obtenue avec l'algorithme de clustering ascendant hiérarchique.

Parmi tous ceux retenus, nous en voyons quatre très importants. Ce sont les indicateurs de groupe de densité, radius, distance minimum inter-groupes et distance moyenne inter-groupes. Les deux premiers sont des indicateurs intra-groupe qui informent de la qualité propre du groupe indépendamment des autres groupes alors que les deux derniers informent de la qualité du groupe par rapport aux autres groupes. Grâce à eux, il est facile de savoir si un groupe est pertinent mais aussi de comparer des classifications obtenues par différents algorithmes. Ils seront présentés dans le chapitre suivant pour comparer les méthodes incrémentales de l'algorithme de clustering ascendant hiérarchique.

Bien qu'utiles, les autres indicateurs ont un intérêt moindre. Les indicateurs de gain permettent de juger de l'intérêt d'effectuer la classification, est-ce que les groupes obtenus sont réellement plus pertinents que l'ensemble de départ. Les indicateurs de moyenne ont plutôt un intérêt statistique et donnent une idée générale sur l'ensemble des groupes. En ce qui concerne l'indicateur de silhouette, il porte sur une granularité très fine, un document, ce qui le rend peut-être moins intéressant. De plus, même en utilisant la formule utilisant les

centroïdes, son temps de calcul pour tous les documents d'un corpus devient rapidement élevé avec l'augmentation du nombre de documents dans ce corpus.

5.5. Conclusion

Nous avons présenté trois contributions dans ce chapitre : un framework qui utilise un algorithme de classification dont nous souhaitons évaluer la qualité des groupes obtenus.

Tout d'abord, nous avons détaillé notre framework global de gestion de fiches d'anomalie pour la construction d'un système d'aide à la maintenance. Ce framework est basé sur le modèle de descripteur de fiche d'anomalie, présenté dans le chapitre précédent. Il exploite les propriétés de structuration et de qualité descriptive des attributs pour classifier les fiches d'anomalie d'une part et pour répondre à nos objectifs d'assistance à la maintenance corrective et préventive d'autre part.

Ensuite, nous nous sommes intéressés à l'algorithme de classification à utiliser dans notre framework. Nous nous sommes tournés vers les méthodes de clustering ascendant hiérarchique et avons proposé une solution pour diminuer l'importance du biais de l'ordre de sélection des documents induit par la méthode incrémentale. Nous autorisons la fusion de groupes au cours de l'exécution de l'algorithme. Pour cela, nous avons défini les conditions de déclenchement de la fusion de deux groupes ou plus et déterminons quels groupes sont suffisamment similaires pour être potentiellement fusionnés.

Enfin, nous avons sélectionné un ensemble d'indicateurs (de document, de groupe, de gain et de moyenne), pour évaluer la qualité de la classification obtenue. Ils peuvent donner des informations sur une classification ou bien être utilisés pour comparer différentes classifications.

Ce chapitre présentait un framework de classification de fiches d'anomalie sous forme de groupes de solutions à l'intérieur de groupes de problèmes. Cette classification met en évidence les patterns d'anomalies. En effet, chaque groupe de problèmes ainsi que les groupes de solutions qui lui sont associés peuvent être considérés comme un pattern. Dans notre contexte, un pattern n'est pas un couple {problème, solution} comme nous avons pu le voir dans le chapitre 2 mais un couple {problème, ensemble de solutions}, plusieurs solutions pouvant répondre à un même problème. Ce framework sert ainsi la maintenance corrective et la maintenance préventive. Pour tout nouveau problème, il est possible de connaître l'ensemble des solutions qui peuvent le résoudre si ce problème a déjà été rencontré par le passé. La maintenance corrective en est donc facilitée. Pour la maintenance préventive, les valeurs de certains indicateurs vont révéler les anomalies récurrentes.

Dans le chapitre suivant, nous présentons l'implémentation de ce framework ainsi que son évaluation en termes de qualité des résultats obtenus et de performance des algorithmes utilisés.

Chapitre 6

Mise en œuvre

6.1. Introduction

Le modèle de descripteur d'anomalie et le framework proposés ont été implémentés. Pour l'évaluation, les mêmes jeux de données réelles que ceux employés pour l'évaluation du modèle de descripteur de fiche d'anomalie ont été utilisés : une base de fiches d'anomalie logicielle et une base de fiches d'anomalie de production industrielle. Trois méthodes standards de l'algorithme de clustering ascendant hiérarchique ont été implémentées, les méthodes single-link, complete-link et incrémentale ainsi que deux méthodes incrémentales issues de l'algorithme proposé dans le chapitre précédent, §5.3.4.3. Les indicateurs ont aussi été implémentés.

Dans ce chapitre, nous présentons nos choix d'implémentation, avec notamment la manière dont le descripteur de fiche d'anomalie est construit, et le prototype logiciel développé, dont la propriété intellectuelle revient en intégralité à la société Intercim. Les évaluations menées sur les deux bases de fiches d'anomalie réelles concernent essentiellement deux points : la qualité des groupes obtenus et la performance des méthodes en termes de temps d'exécution. Les valeurs des indicateurs sur les groupes obtenus par les différentes méthodes sont aussi comparés.

6.2. Implémentation

6.2.1. Utilisation de techniques classiques en Recherche d'Information

Pour représenter les fiches d'anomalie, les mots significatifs (ou mots-clés) sont extraits de la valeur de leurs attributs. La propriété de structuration distingue les attributs contraints des attributs libres. Elle définit comment l'information contenue dans un attribut doit être extraite. Dans le cas d'un attribut contraint, un seul mot significatif est extrait, la valeur qu'il contient. Un mot significatif peut donc être composé de plusieurs mots. Cela trouve son intérêt dans les informations dont l'ensemble des mots a un sens plus fort que

chaque mot pris séparément (de la même manière que les mots-clés d'un article sont parfois composés de plusieurs mots). Dans le cas d'un attribut libre, la valeur est découpée en mots selon les espaces et les différents caractères spéciaux. Les mots obtenus sont testés pour vérifier qu'ils n'appartiennent pas à la liste de mots stop généraux de la langue anglaise ou à celle de mots stop spécifiques au domaine.

L'algorithme de suppression de suffixe de Porter (Porter, 2006) bien adapté à la Recherche d'Information, est appliqué sur les mots pour ne considérer que la racine des mots ce qui permet d'augmenter la similarité entre fiches d'anomalie puisque deux mots différents mais de même racine deviendront un seul et même mot significatif. Pour améliorer la similarité entre fiches d'anomalie, un algorithme de réduction de dimension aurait aussi pu être utilisé tel que la Décomposition en Valeurs Singulières (SVD), l'Analyse en Composantes Principales (PCA), l'Analyse Sémantique Latente (LSA), etc. Cela permettrait notamment d'améliorer le temps d'exécution de l'algorithme de regroupement et de travailler sur l'aspect sémantique plus que sur l'aspect syntaxique. Chaque fiche d'anomalie est donc représentée par un ensemble de mots significatifs appelé descripteur. Ce descripteur est composé de deux parties, une partie appelée descripteur de problème qui rassemble les mots significatifs issus des attributs descriptifs de problème et une autre partie appelée descripteur de solution qui rassemble les mots significatifs issus des attributs descriptifs de solution (cf §4.5).

Concernant le calcul de similarité, nous avons choisi d'utiliser le modèle vectoriel avec la mesure TF-IDF (Term Frequency – Inverse Document Frequency) (cf §2.4.3). Cela donne un poids à chaque mot significatif de chaque fiche d'anomalie qui est utilisé pour calculer la similarité entre descripteurs de fiches d'anomalie avec la mesure cosinus (cf, §2.4.3). Ces mesures de similarité sont utilisées dans une classification ascendante hiérarchique pour produire les groupes de fiches d'anomalie. Dans une catégorie, les fiches d'anomalie dont la similarité entre leurs descripteurs de problème est supérieure au seuil de similarité constituent un groupe de problèmes. Les groupes de solutions à l'intérieur de chaque groupe de problèmes sont construits de la même manière en employant les descripteurs de solution.

6.2.2. Le modèle implémenté en base de données

Le modèle implémenté en base de données peut être vu comme composé de trois parties :

- Le modèle de descripteur de fiche d'anomalie avec les classes *FicheAnomalie*, *Attribut*, *Renseigner*, *DescriptifProbleme*, *DescriptifSolution* et *NonDescriptif*.

- Les éléments nécessaires à la mise en place du framework représentés par les classes *Categorie*, *GroupeProblemes*, *GroupeSolutions* et *Prototype*.
- Les choix d'implémentation présentés dans le paragraphe précédent symbolisés par les classes *MotSignificatif*, *Extraire* et *Representer*.

Le modèle de descripteur de fiche d'anomalie a donc été instancié. Des éléments supplémentaires y ont été apportés. Ils répondent aux besoins de mise en place du framework tel que l'identification des catégories et des groupes. Ils font aussi suite aux choix d'implémentation que sont par exemple l'extraction de mots-clés ou la représentation du descripteur de fiche d'anomalie sous forme de mots-clés pondérés (Figure 35). Les éléments calculés tels que la similarité entre deux fiches d'anomalie ou la valeur du TF d'un mot significatif d'une fiche d'anomalie ou la valeur de l'IDF d'un groupe n'apparaissent pas sur le diagramme de classes. Ont donc été intégrés :

- Le fait qu'un attribut puisse être utilisé comme critère de catégories de fiches d'anomalie. Seuls les attributs contraints et descriptifs de problème peuvent avoir ce rôle. Cela apparaît dans la classe *DescriptifProbleme* avec la propriété *estCritereCategories*. Une note est associée à cette propriété booléenne pour indiquer qu'elle ne peut être à vrai que si la propriété booléenne *estContraint* de la classe *Attribut* est aussi à vrai.
- Les groupes et leur prototype. Une fiche d'anomalie n'appartient qu'à un seul groupe de problèmes (classe *GroupeProblemes*) et qu'à un seul groupe de solutions (classe *GroupeSolutions*). De plus, un groupe de solutions n'appartient qu'à un groupe de problèmes. Un groupe de solutions ne peut pas exister sans un groupe de problèmes auquel se rattacher. Pour chaque groupe, un prototype (classe *Prototype*) est créé, composé de mots significatifs pondérés de la même manière qu'un descripteur de fiche d'anomalie. C'est une fiche d'anomalie artificielle, il est logique que sa représentation soit la même que celle d'une fiche d'anomalie. Un prototype existe seulement si le groupe qu'il représente existe.
- L'extraction de mots significatifs à partir des valeurs des attributs. La propriété *valeur* dans la classe *Renseigner* est un texte contenant un mot significatif si l'attribut est contraint et plusieurs mots significatifs dans le cas d'un attribut libre. Un même mot significatif peut apparaître plusieurs fois dans une même valeur, cette information est conservée avec la propriété *nbOccurrences* de la classe *Extraire*.

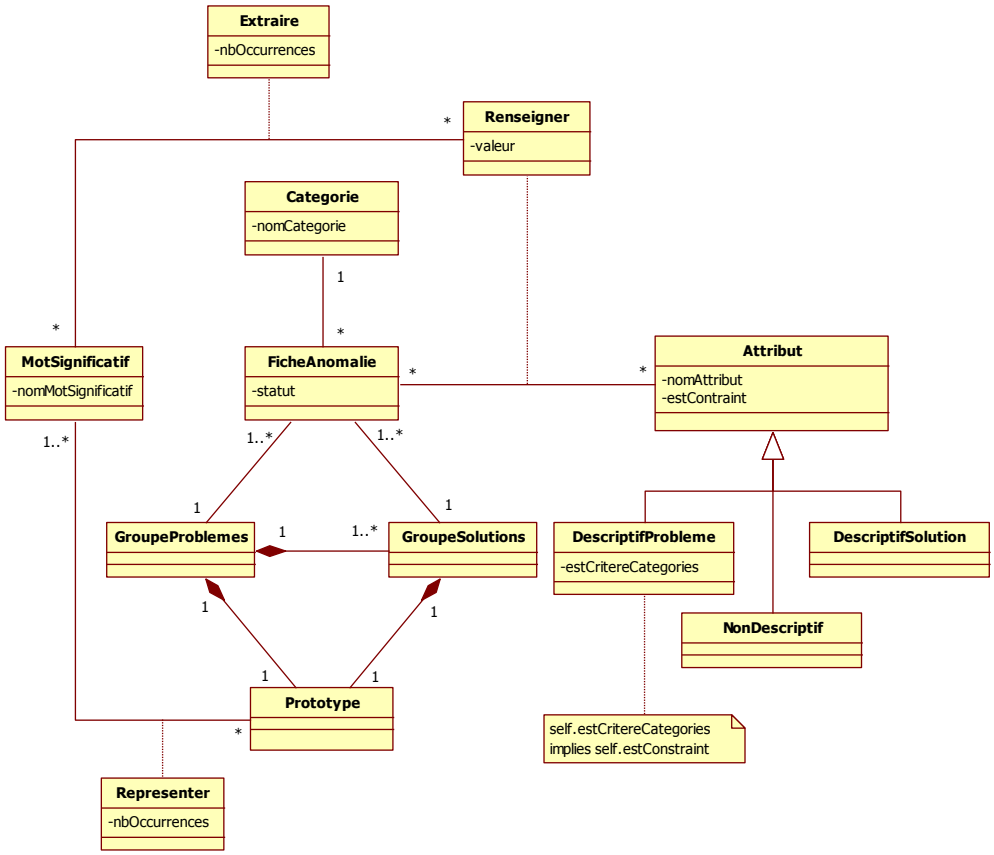


Figure 35 – Modèle de descripteur de fiche d'anomalie au coeur du framework proposé

Note 1 : La classe *Renseigner* est à la fois une classe d'association et une classe avec elle-même une classe d'association. Cela se justifie d'une part par le fait que la valeur d'un attribut dépend de l'attribut et de la fiche d'anomalie et d'autre part par le fait que le nombre d'occurrences dépend du mot et de la valeur de l'attribut considéré. Dans l'implémentation en base de données, la clé primaire de la classe *Extraire* est composée de celle de la classe *MotSignificatif* et celle de la classe *Renseigner*, elle-même composée de la clé primaire de la classe *FicheAnomalie* et de celle de la classe *Attribut*.

Note 2 : La classe *Prototype* possède deux liens de composition de type (1-1), l'un avec la classe *GroupeProblèmes*, l'autre avec la classe *GroupeSolutions*. Pour éviter ce type de relation, nous aurions pu utiliser une propriété *prototype* dans chacune des deux classes. Cependant, nous voulions insister sur cet aspect composition (un prototype n'existe pas sans un groupe auquel se rattacher) et nous voulions montrer qu'un prototype est un ensemble de mots significatifs comme un descripteur de fiche d'anomalie. Cette uniformisation dans leur représentation rend cohérents les calculs de similarités entre eux. En supprimant la classe Prototype pour n'en faire qu'un attribut, ce sont les groupes (*GroupeProblemes* et *GroupeSolutions*) qui sont un ensemble de mots significatifs.

6.2.3. Présentation du prototype logiciel

Le prototype logiciel, développé en Java, est composé de trois modules : un module de classification des fiches d'anomalie, un module de recherche de solution à un problème et un module d'information sur les groupes obtenus.

6.2.3.1. Le module de classification des fiches d'anomalie

Le module de classification des fiches d'anomalie est l'élément central du prototype. Les deux autres modules s'appuient sur le résultat qu'il produit pour générer leurs résultats. Il regroupe un ensemble de traitements définis dans le framework proposé. D'abord, pour chaque fiche d'anomalie, sa catégorie est identifiée grâce à la valeur des attributs contraints descriptifs de problèmes définis comme critères de catégories. Ensuite, les mots significatifs sont extraits de la valeur de chaque attribut de chaque fiche d'anomalie. Ce traitement est présenté dans en 6.2.1. L'ordre d'exécution de ces deux premiers traitements n'a pas d'importance. Ensuite, l'algorithme de clustering ascendant hiérarchique est exécuté. Pour les méthodes non-incrémentales (single-link et complete-link), une étape de calcul de similarité entre fiches d'anomalie deux à deux est effectuée avant l'exécution de l'algorithme et une étape de création des prototypes des groupes est effectuée après son exécution. Tous ces traitements sont exécutés automatiquement en exploitant la connaissance de l'expert apportée en amont (notamment concernant les propriétés de structuration et de qualité descriptive des attributs). L'utilisateur peut choisir parmi cinq méthodes de clustering ascendant hiérarchique, il peut lancer autant de classifications qu'il le souhaite (Figure 36).

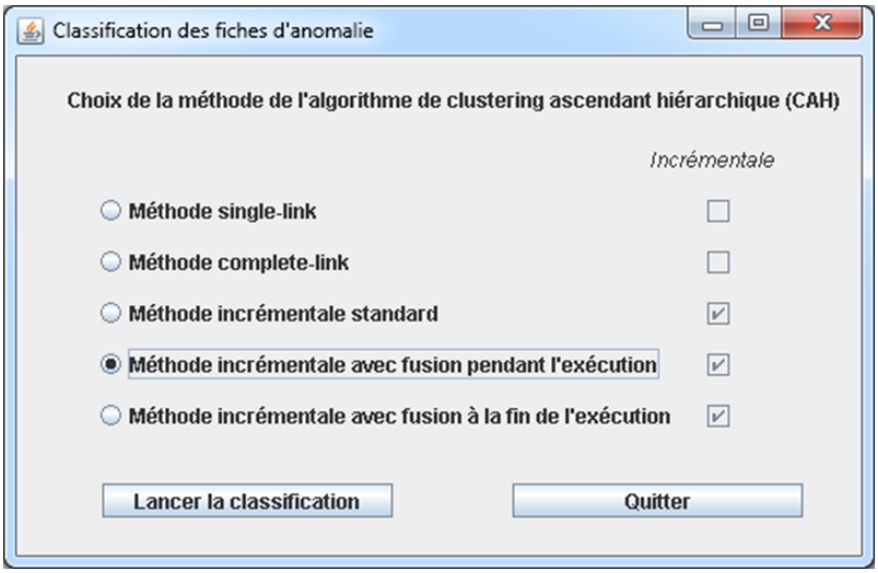


Figure 36 – Interface pour la classification des fiches d'anomalie

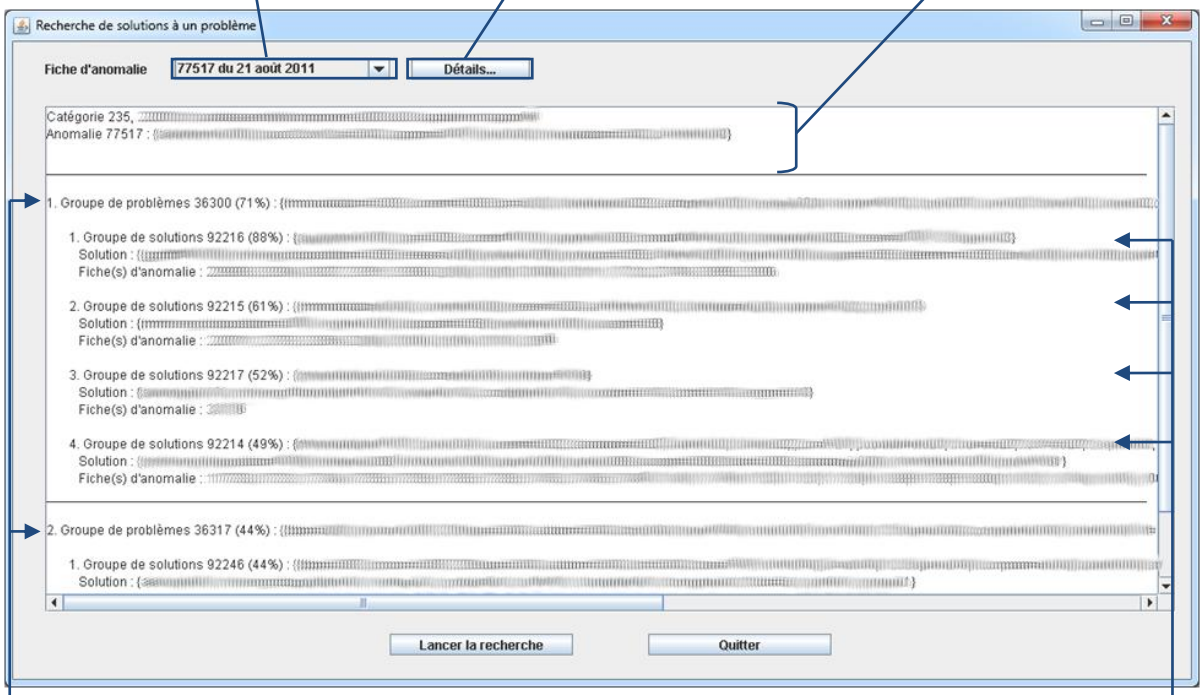
6.2.3.2. Le module de recherche de solutions

Le module de recherche de solutions a pour objectif de faciliter la maintenance corrective. Il permet de proposer des solutions utilisées par le passé pour répondre au problème d'une nouvelle anomalie (Figure 37). L'utilisateur sélectionne l'identifiant de la fiche d'anomalie qu'il veut résoudre (a.). Il peut la visualiser (b.). La catégorie à laquelle la fiche d'anomalie appartient est alors déterminée et les mots significatifs descriptifs de problème sont récupérés (c.). Ce descripteur est comparé à la partie problème des prototypes des groupes de problème de la même manière que lors de la phase de classification des fiches d'anomalie. Seuls sont conservés, présentés et ordonnés selon la similarité, les groupes de problèmes dont la similarité entre la partie problème de leur prototype et la partie problème du descripteur de l'anomalie courante est supérieure au seuil de similarité fixé (d.). Pour chaque groupe de problèmes retenu, les groupes de solutions sont présentés et ordonnés selon la similarité entre la partie problème de leur prototype de solution et la partie problème du descripteur de l'anomalie courante (e.). Les éléments pertinents de la partie solution des groupes de solutions sont affichés. Pour des raisons de confidentialité, les mots significatifs ont été floutés.

a. Visualisation de la fiche d'anomalie sélectionnée

b. Informations calculées sur la fiche d'anomalie

c. Sélection de la fiche d'anomalie dont on veut résoudre le problème



d. Tri des groupes de problèmes sur la similarité entre la partie problème du descripteur de chacun de ces groupes et la partie problème du descripteur de la fiche d'anomalie sélectionnée.

Affichage des groupes de problèmes : similarité et partie problème du descripteur.

e. Dans chaque groupe de problèmes, tri des groupes de solutions sur la similarité entre la partie problème du descripteur de chacun de ces groupes et la partie problème du descripteur de la fiche d'anomalie sélectionnée.

Affichage des groupes de solutions : similarité et partie problème du descripteur.
Affichage des solutions : partie solution du descripteur.
Affichage des fiches d'anomalies de chaque groupe de solutions.

Figure 37 – Interface pour la recherche de solutions à un problème

6.2.3.3. Le module d'information sur les groupes

Le module d'informations sur les groupes a pour objectif de faciliter la maintenance préventive. Pour chaque catégorie, groupe de problèmes et groupe de solutions, les indicateurs présentés en 5.4.3 sont calculés et présentés à l'utilisateur. Actuellement, il n'en existe qu'un affichage textuel mais des tests pour représenter les groupes sur deux dimensions à l'aide de l'algorithme MDS (cf. §2.4.2.2) ont été réalisés. Le jeu de données concerne un projet, nommé ActionPlanT, débuté en juin 2010, visant à développer une vision à court, moyen et long terme du rôle des TIC (Technologies de l'Information et des Communications) dans l'industrie de production européenne. Cela consiste dans un premier temps à analyser les plans d'action existants, les documents de stratégie et les documents de projet dans le domaine de la fabrication, des TIC pour la fabrication ainsi que des TIC théoriques (http://www.actionplant-project.eu/). Une partie de ces éléments, au nombre de 237, concernent des projets européen de recherche qui peuvent se représenter sous forme de documents ayant des attributs (*Source*, *Géographie*, *Objectif*, *Titre du sujet de recherche*, *Sujet de recherche*, *Impact* et *Commentaires*). Tous ces attributs ont le même apport sémantique, ils ne sont descriptifs que d'une nature d'information. Cependant, deux regroupements successifs ont été effectués. Le premier génère des groupes en utilisant un certain seuil de similarité (Figure 38 – seuls les groupes composés de plus d'un document sont représentés pour faciliter la lisibilité du graphique). Puis, à l'intérieur de chaque groupe obtenu, de nouveaux groupes plus précis ont été obtenus en effectuant un deuxième regroupement utilisant un seuil de similarité plus élevé (Figure 39 – ces groupes représentent un second regroupement à l'intérieur du groupe composé de 83 documents de la Figure 38). La taille des disques est proportionnelle au nombre de documents du groupe (nombre indiqué sur le groupe) et les trois mots significatifs les plus représentatifs de chaque groupe sont présentés (la valeur pour chaque mot est son poids calculé avec la formule TF-IDF, cf. §2.4.3).

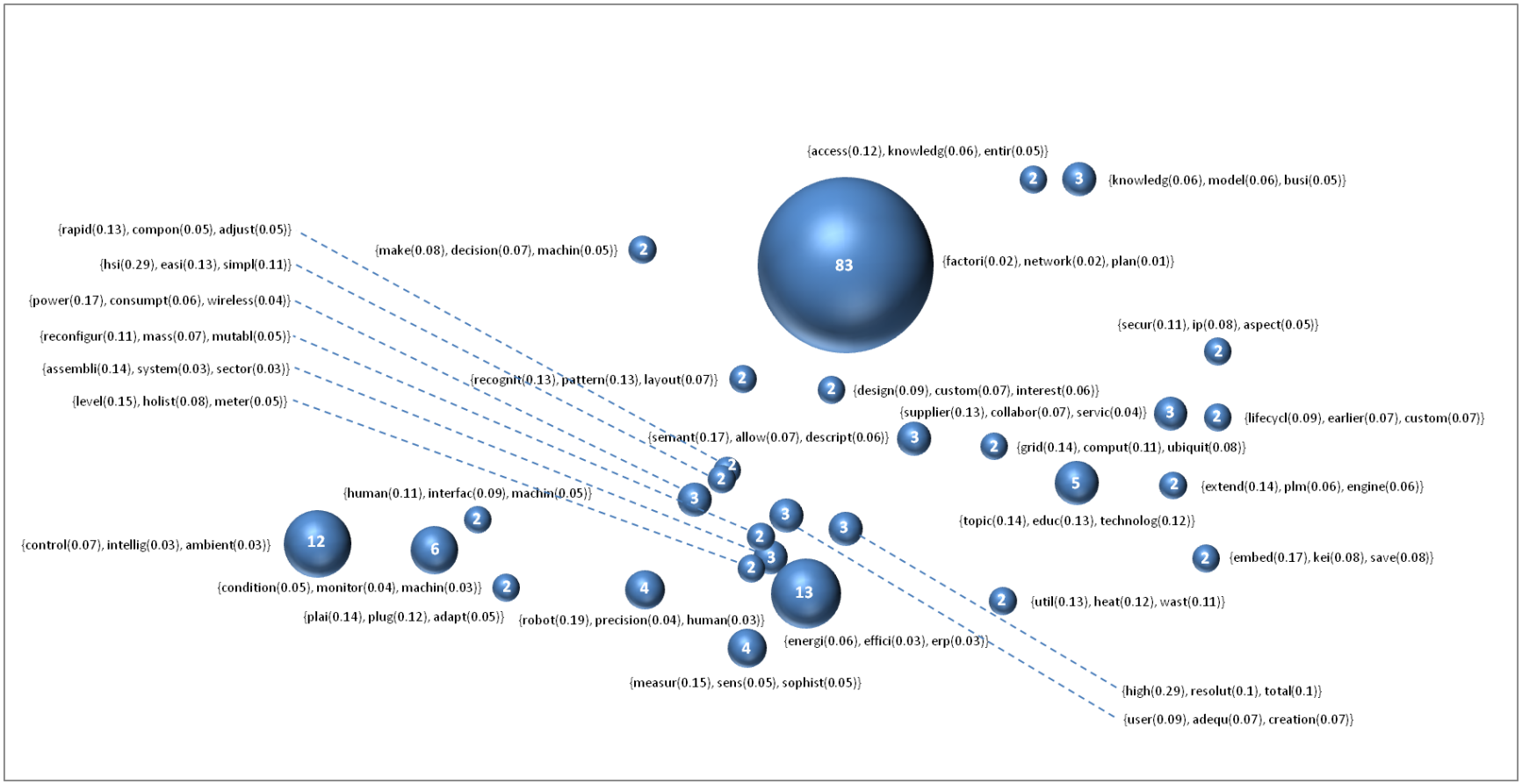


Figure 38 – Représentation de groupes à l'aide de l'algorithme MDS (1er regroupement)

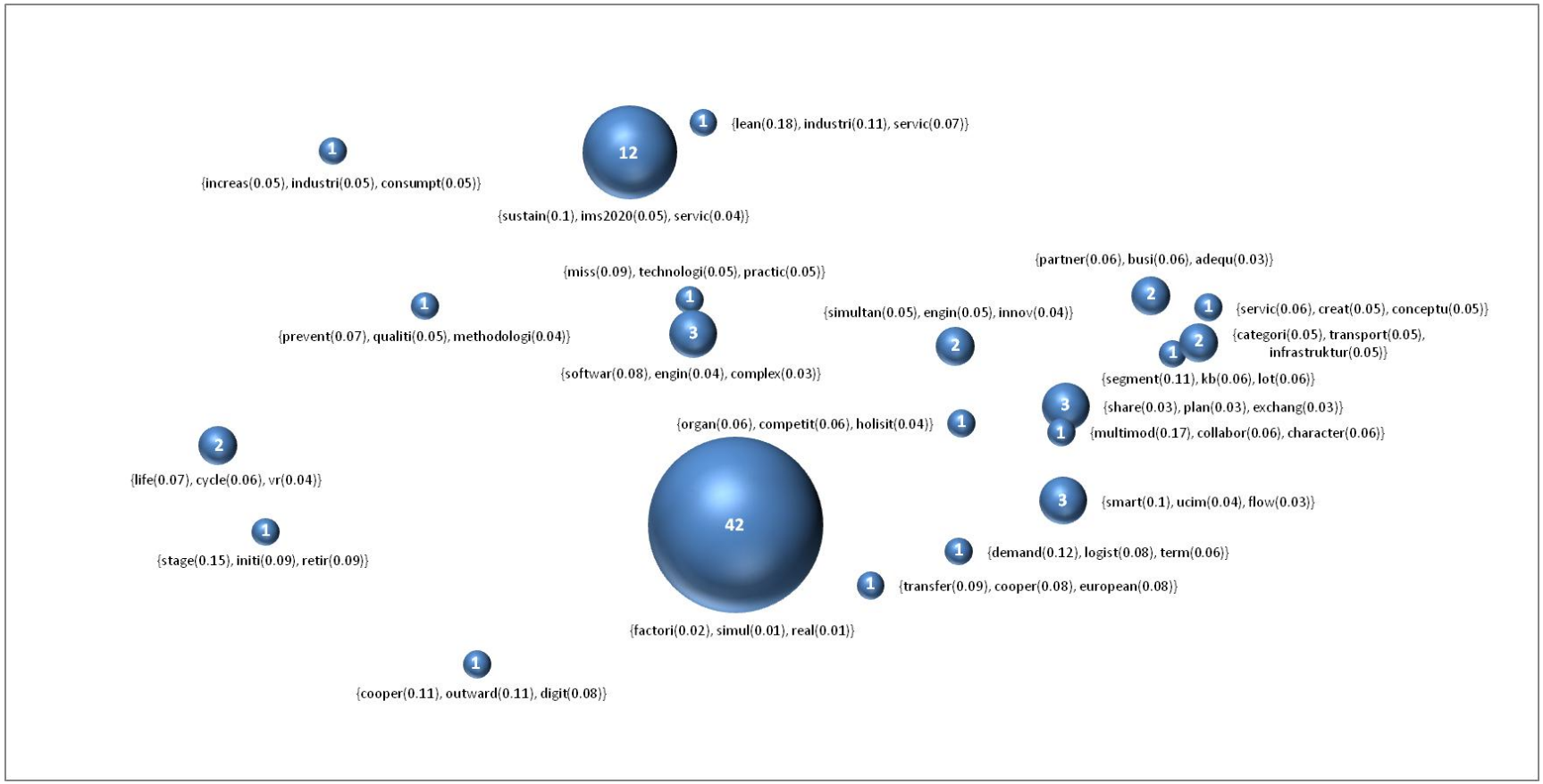


Figure 39 – Représentation de groupes à l'aide de l'algorithme MDS (2ème regroupement)

6.3. Expérimentations

Pour tester le framework proposé dans le chapitre 5, les jeux de données employés pour valider le modèle de descripteur de fiche d'anomalie ont été utilisés. Les premiers tests portent sur une base qui contient plus de 7000 fiches d'anomalie logicielle issues de la base d'anomalies logicielles d'Intercim. Les seconds tests portent sur une base qui contient près de 75000 fiches d'anomalie de production industrielle issues de la base d'un client d'Intercim.

6.3.1. 1er jeu de données : base de fiches d'anomalie logicielle réelles

6.3.1.1. Les données

Cette première base contient 7216 fiches d'anomalie logicielles réelles. Toutes ces fiches d'anomalie concernent plusieurs versions d'un seul logiciel. Elles ont été enregistrées par des responsables de l'assurance-qualité pendant plus de 3 ans, de septembre 2005 à décembre 2008.

Après avoir utilisé une liste de mots stop et appliqué un algorithme de suppression de suffixe (les détails sont donnés en 6.2.1), 2680 mots significatifs ont été extraits des attributs descriptifs de problème. Grâce à la définition de catégories de fiches d'anomalie, ces 2680 mots significatifs ne sont pas utilisés en même temps pour effectuer le calcul de similarité. Dans les 59 catégories que nous obtenons, l'une d'elles atteint 1098 mots significatifs impliqués dans le calcul de similarité, le nombre moyen étant de 213.

Comme il a été présenté dans le chapitre 4, §4.5.2.1, le partitionnement des attributs selon la propriété de qualité descriptive révèle qu'aucun attribut ne contient d'information relative à la solution. Cette évaluation a pour but d'obtenir des premiers résultats concernant la qualité des groupes obtenus avec l'algorithme de clustering ascendant hiérarchique, l'importance de l'ordre de sélection des documents pour la méthode incrémentale et la performance des différentes méthodes implémentées. Pour ce jeu de données, trois méthodes de l'algorithme de clustering ascendant hiérarchique ont été implémentées et testées (dans l'énoncé de la complexité de chaque méthode qui suit, n représente le nombre de fiches d'anomalie) :

- la méthode single-link (appelée aussi MIN), de complexité $O(n^2)$, qui produit de larges groupes potentiellement assez hétérogènes dû à l'effet de chaîne dont elle souffre

- la méthode complete-link (appelée aussi MAX), de complexité $O(n^2.\log(n))$, qui résout le problème de l'effet de chaîne et qui produit des groupes plus petits et plus homogènes
- la méthode incrémentale, de complexité $O(n.\log(n))$, qui résout le problème de complexité quadratique des deux précédentes mais qui est sensible à l'ordre de sélection des documents.

6.3.1.2. Evaluation qualitative

Une étude a été menée sur une catégorie contenant 92 fiches d'anomalie. Cette catégorie contient assez de fiches d'anomalie pour obtenir des résultats intéressants par notre approche automatique. En outre, il sera possible de comparer ces résultats avec ceux que nous voudrions obtenir manuellement. Au cours d'une première étape, les regroupements ont été effectués à la main par un expert pour obtenir la classification de référence. Celle-ci révèle que 22 appariements ont été effectués, les 92 fiches d'anomalie ont donc été classifiées en 70 groupes.

Ensuite, pour déterminer la valeur du seuil de similarité qui produit les groupes les plus pertinents, différentes classifications ont été obtenues en faisant varier le seuil de similarité entre 0,9 et 0,3 et comparées avec la classification de référence. Pour estimer la qualité de la classification, les courbes rappel-précision (Figure 40) et F-mesure (Figure 41) sont présentées. Les nombres sur les points indiquent la valeur du seuil de similarité.

La Figure 40 présente une courbe pour chaque méthode de l'algorithme. Chaque courbe indique la proportion de regroupements corrects effectués par l'algorithme parmi tous les groupes que l'algorithme a produit (la précision) par rapport à la proportion de regroupements corrects effectués par l'algorithme parmi tous les groupes que nous voudrions obtenir (le rappel). Cette figure montre que, quelle que soit la méthode de l'algorithme, la meilleure classification est obtenue avec un seuil de similarité compris entre 0,43 et 0,5. A l'extérieur de cet intervalle de valeurs, la précision diminue, il y a trop de bruit (trop de groupes incorrects), ou c'est le rappel qui diminue, il y a trop de silence (pas assez de groupes corrects).

L'objectif de la courbe rappel-précision est de maximiser la précision et le rappel. Cependant, il est très difficile d'estimer un bon seuil de similarité puisque lorsque la précision augmente, le rappel diminue, et inversement. La F-mesure permet de combiner la précision et le rappel. La Figure 41 montre une courbe qui représente une moyenne pondérée de la précision et du rappel pour chaque méthode de l'algorithme.

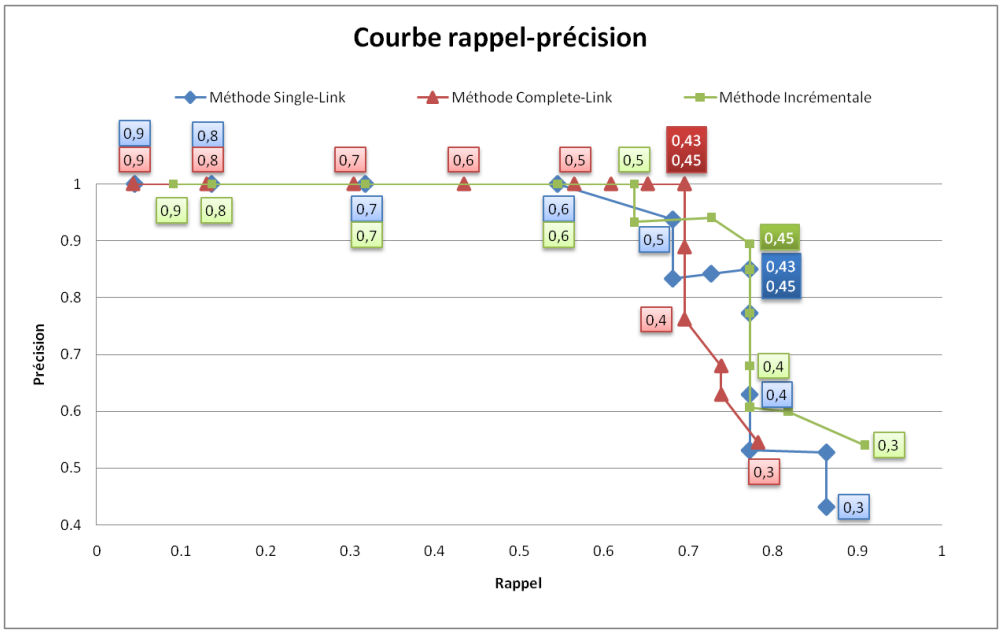


Figure 40 – Rappel-précision sur une catégorie de 92 fiches d'anomalie pour les méthodes single-link, complete-link et incrémentale (1er jeu de données)

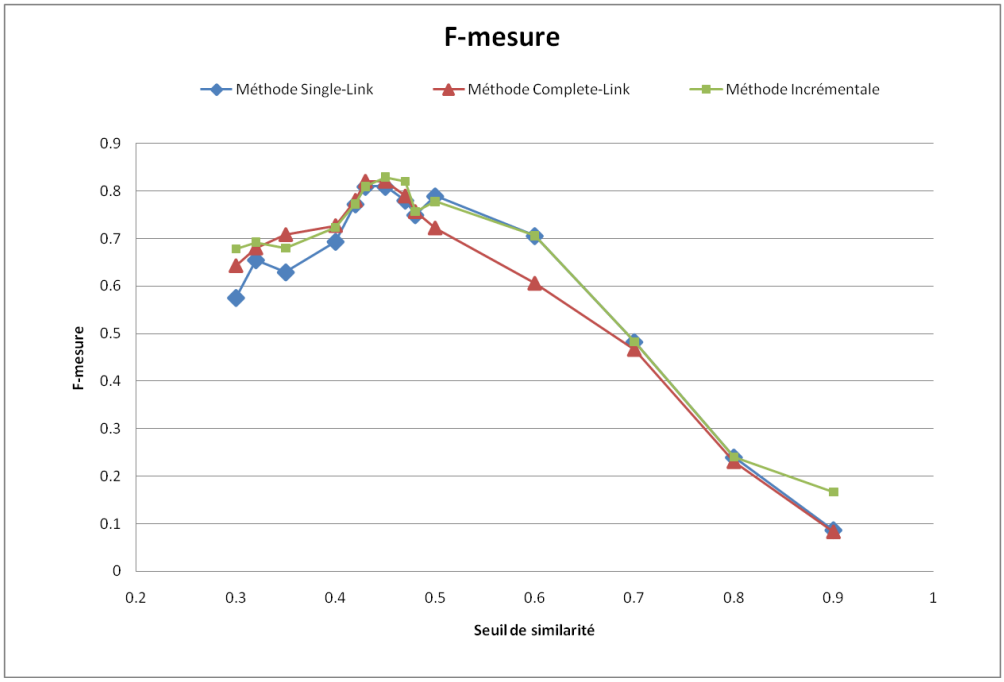


Figure 41 – F-mesure sur une catégorie de 92 fiches d'anomalie pour les méthodes single-link, complete-link et incrémentale (1er jeu de données)

Ces trois méthodes donnent des résultats très similaires. Il ne semble pas y avoir de réel effet de chaîne puisque la méthode single-link fournit des valeurs assez similaires aux deux autres méthodes. La valeur du seuil de similarité qui permet d'obtenir la classification la plus proche de la classification de référence est comprise entre 0,43 et 0,45 pour les méthodes single-link et

complete-link et est de 0,45 pour la version incrémentale. Aucun algorithme ne se démarque réellement des autres. Au niveau de la qualité des groupes obtenus sur notre échantillon de test, il semble que la méthode incrémentale soit un compromis entre les méthodes single-link et complete-link et même légèrement meilleure pour un seuil de similarité fixé à 0,45. Avec un seuil de similarité fixé à 0,45, nous obtenons :

- pour la méthode single-link, une précision et un rappel autour de 0,8
- pour la méthode incrémentale, une précision de 0,9 et un rappel de légèrement inférieur à 0,8
- pour la méthode complete-link, une précision de 1 et un rappel de 0,7
- pour toutes les méthodes, un score de F-mesure supérieur à 0,8.

6.3.1.3. Evaluation de la performance

Pour mener cette étude, les méthodes de l'algorithme ont été exécutées sur des catégories de fiches d'anomalie de tailles variées (0, 17, 37, 52, 78, 110, 170, 281, 462, 559, 911, 961, 1005 et 1232 fiches d'anomalie) pour déterminer les groupes de problèmes. Le seuil de similarité choisi est de 0,45 puisqu'il permet un ensemble de regroupements optimum pour toutes les méthodes. Chaque valeur obtenue du temps d'exécution d'une méthode pour un groupe donné est une moyenne de cinq de ses exécutions. Le temps d'exécution considéré concerne le temps mis par les méthodes pour déterminer les groupes mais aussi le temps nécessaire à la création des prototypes de groupe puisque la méthode incrémentale produit nécessairement pour sa propre exécution le prototype de chaque groupe.

Pour cette étude, le matériel utilisé est un ordinateur portable avec un processeur simple cœur cadencé à 1,20GHz et 2Go de RAM tournant sous Windows XP Professionnel SP3. La méthode incrémentale est nettement plus rapide que les méthodes non-incrémentales même si on ne considère pas la création du prototype de chaque groupe (Figure 42). En la considérant pour mettre les trois méthodes au même niveau de résultat produit, l'écart s'accentue grandement (Figure 43). De plus, il est logique de constater que la méthode complete-link est moins rapide que la méthode single-link car elle demande plus de comparaisons entre descripteurs de fiches d'anomalie lors de l'ajout potentiel d'une fiche d'anomalie dans un groupe (Figure 42) et elle crée plus de groupes ce qui demande la création de plus de prototypes de groupe (Figure 43).

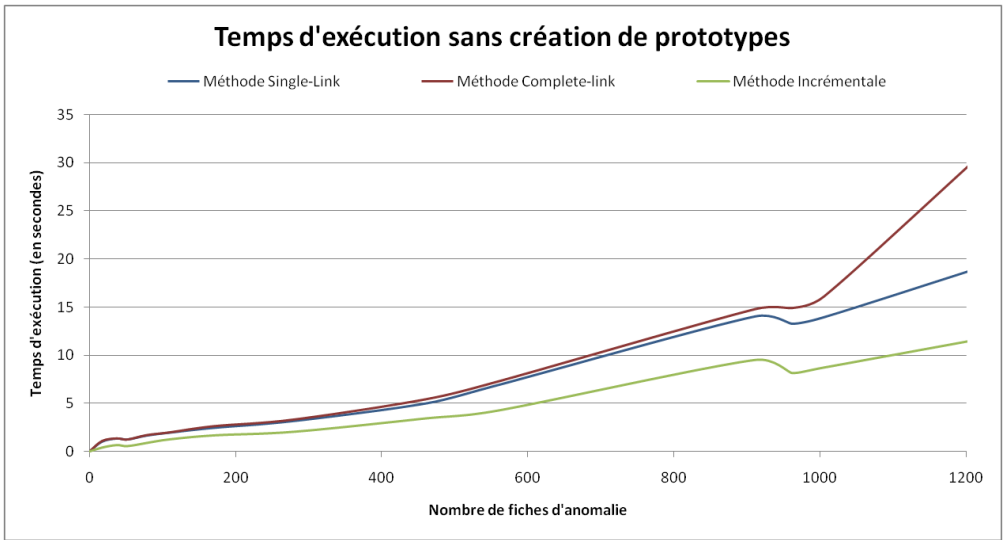


Figure 42 – Temps d'exécution sans création des prototypes (1er jeu de données)

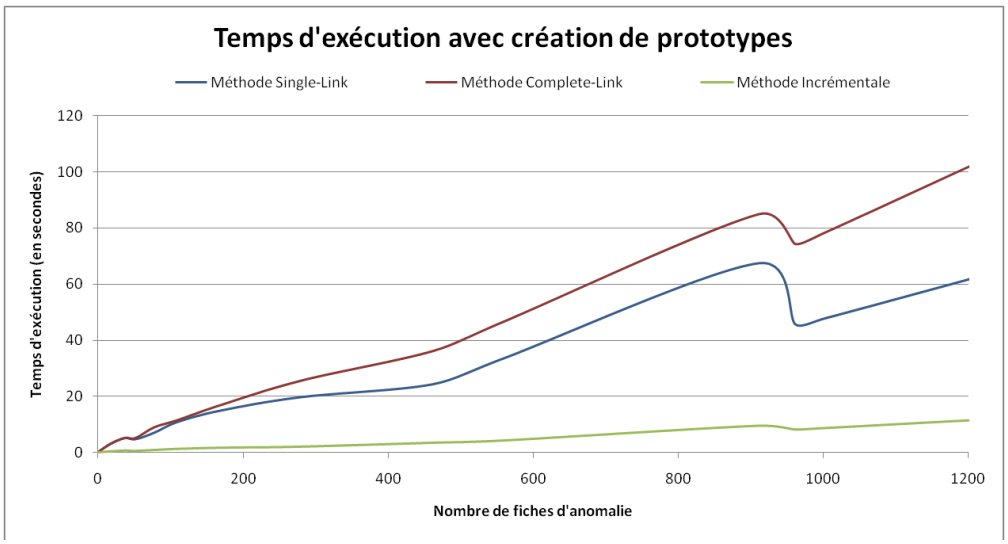


Figure 43 – Temps d'exécution avec création des prototypes (1er jeu de données)

Pour déterminer les groupes, le temps d'exécution des méthodes, incrémentale ou non, dépend de deux éléments (Table 26) : le nombre de fiches d'anomalie et le nombre de mots significatifs. Dans le tableau, les temps d'exécution des versions non-incrémentales est celui sans création des prototypes. Le pourcentage entre parenthèses mesure la différence de temps d'exécution par rapport à la version incrémentale. Plus le nombre de fiches d'anomalie à regrouper est grand, plus le nombre de comparaisons à effectuer entre descripteurs de fiches d'anomalie l'est aussi ; et plus le nombre de mots significatifs est grand, plus la comparaison entre deux descripteurs de fiches d'anomalie (ou entre un descripteur de fiche d'anomalie et un prototype) nécessite de temps (Table 26 – N°3 & 4).

N°	Nombre de fiches d'anomalie	Nombre de mots significatifs	Nombre de groupes obtenus avec la méthode incrémentale	Temps d'exécution de la méthode incrémentale (en secondes)	Temps d'exécution de la méthode single-link (en secondes)	Temps d'exécution de la méthode complete-link (en secondes)
1	0	0	0	0	0 *(0%)*	0 *(0%)*
2	17	109	17	0,40	1,02 *(+61%)*	1,08 *(+63%)*
3	37	210	36	0,65	1,34 *(+51%)*	1,35 *(+52%)*
4	52	134	38	0,55	1,24 *(+56%)*	1,24 *(+56%)*
5	78	224	57	0,88	1,64 *(+46%)*	1,69 *(+48%)*
6	110	361	90	1,26	1,95 *(+35%)*	1,98 *(+36%)*
7	170	491	135	1,66	2 ,45 *(+32%)*	2,62 *(+37%)*
8	281	464	188	2,05	3,17 *(+35%)*	3,33 *(+38%)*
9	462	657	279	3,44	5,01 *(+31%)*	5,45 *(+37%)*
10	559	661	341	4,25	6,90 *(+38%)*	7,28 *(+42%)*
11	911	1066	613	9,50	14,04 *(+32%)*	14,80 *(+36%)*
12	961	889	523	8,14	13,27 *(+39%)*	14,91 *(+45%)*
13	1005	967	529	8,71	13,95 *(+38%)*	16,12 *(+46%)*
14	1232	1045	646	11,85	19,44 *(+39%)*	31,72 *(+63%)*

Table 26 – Eléments intervenant dans le temps d'exécution des versions de l'algorithme

Outre ces deux éléments, le temps d'exécution de la méthode incrémentale dépend du nombre de groupes (ou de la valeur du seuil de similarité qui induit le nombre de groupes). Plus le nombre de groupes obtenus est limité (grâce à une valeur de seuil de similarité faible), moins il y a de comparaisons à effectuer entre un descripteur de fiche d'anomalie et les prototypes de groupe (Table 26 – N°11, 12 & 13).

Cette étude nous permet d'affirmer, comme nous l'attendions, que la méthode incrémentale est bien plus rapide que les méthodes non-incrémentales. Elle semble pouvoir supporter un gros volume de données dans un délai raisonnable et la création du prototype de chaque groupe au cours de l'exécution accentue la rapidité de cette méthode. Dans notre contexte, la faiblesse majeure des méthodes non-incrémentales de cet algorithme de clustering ascendant hiérarchique est leur complexité quadratique. Même en ne considérant qu'un peu plus de 7000 fiches d'anomalie, nous pouvons déjà relever des problèmes de performance pour une catégorie possédant 1232 fiches d'anomalie (c'est-à-dire 758296 calculs de mesure cosinus sur le problème). La méthode incrémentale implémentée permet de résoudre, ou au moins de grandement réduire les problèmes de performance.

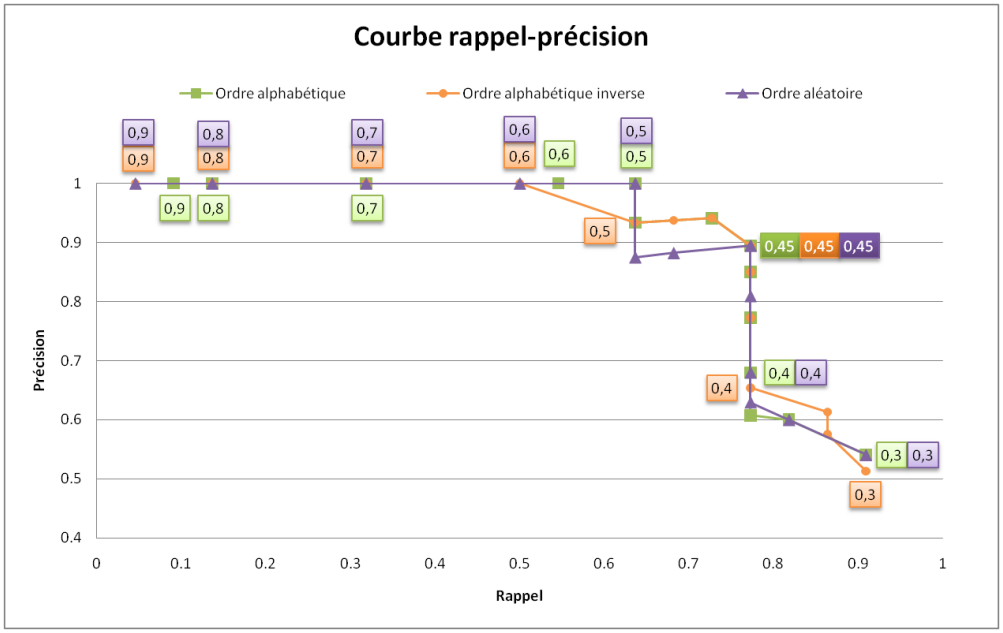


Figure 44 – Rappel-précision sur une catégorie de 92 fiches d'anomalie pour trois ordonnancements de sélection avec la méthode incrémentale (1er jeu de données)

Les trois ordonnancements de sélection des fiches d'anomalie fournissent des résultats très similaires. Sur ce jeu de données, lorsque le seuil de similarité est élevé (supérieur à 0,6), les trois courbes se comportent de la même manière. En revanche, lorsque l'on baisse ce seuil de similarité, de légères différences apparaissent ce qui traduit bien que l'ordre de sélection des fiches d'anomalie a un impact sur les résultats obtenus. Malgré tout, toujours en considérant ce jeu de données, il semble n'avoir qu'une très faible importance. En effet, les trois

courbes se suivent d'assez près et on peut remarquer que le seuil de similarité qui permet d'obtenir les meilleurs résultats pour toutes les courbes est une fois de plus fixé à 0,45.

Evidemment, aucun ordre n'est meilleur qu'un autre dans l'absolu, cela dépend du jeu de données considéré. Ces courbes montrent seulement qu'il peut avoir un impact sur les résultats produits et que celui-ci est plutôt faible. Pour l'évaluation sur le deuxième jeu de données, l'ordre utilisé est l'ordre alphabétique.

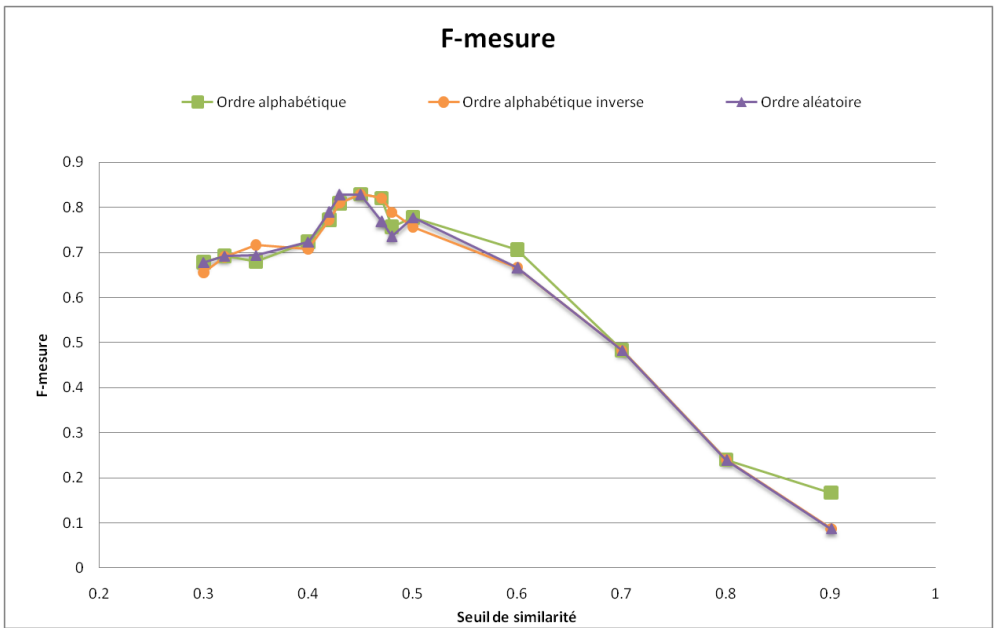


Figure 45 – F-mesure sur une catégorie de 92 fiches d'anomalie pour trois ordonnancements de sélection avec la méthode incrémentale (1er jeu de données)

6.3.2. 2ème jeu de données : base de fiches d'anomalie de production réelles

6.3.2.1. Les données

Cette deuxième base contient 74205 fiches d'anomalie de production industrielle réelles. Toutes ces fiches d'anomalie concernent la production d'un type de produit final d'un seul client. Plusieurs patrons de fiches d'anomalie présents sous plusieurs versions ont été pris en compte pour l'évaluation. Les informations relatives au problème ont été saisies par les personnes qui exécutent les différentes tâches sur la ligne de production. Celles relatives à la solution l'ont été par les personnes d'un service dédié à la résolution des anomalies. Ces fiches d'anomalie ont été enregistrées pendant près de 3 ans, d'octobre 2006 à septembre 2009.

Après avoir appliqué les mêmes traitements que sur le jeu de données précédent (liste de mots stop, algorithme de suppression de suffixe), 127120 mots significatifs ont été extraits, 104214 issus d'attributs descriptifs de problème (23826 d'attributs contraints et 82402 d'attributs libres) et 59101 issus d'attributs descriptifs de solution (3 d'attributs contraints et 59098 d'attributs libres). Dans les 456 catégories obtenues, l'une d'elles met en jeu 19198 mots significatifs, la moyenne par catégorie étant de 792 (seules 107 catégories sont au-dessus de cette moyenne).

Cette expérimentation a pour but de valider une nouvelle fois la qualité des groupes obtenus ainsi que les performances en temps d'exécution des différentes méthodes de clustering ascendant hiérarchique. Deux nouvelles méthodes incrémentales ont été ajoutées aux méthodes single-link, complete-link et incrémentale utilisées dans le jeu d'essai précédent. Ces deux méthodes incrémentales sont issues de l'algorithme proposé en 5.3.4.3. La première autorise la fusion de groupes pendant l'exécution de l'algorithme. La deuxième n'intègre les fiches d'anomalie qui provoquent la fusion de groupes qu'à la fin de l'algorithme. Les indicateurs présentés en 5.4.3 sont aussi comparés en utilisant les groupes obtenus par les méthodes incrémentales.

6.3.2.2. Evaluation qualitative

Une étude a été menée sur une catégorie contenant 145 fiches d'anomalie. De la même manière que lors de l'évaluation précédente, les regroupements ont été effectués à la main par un expert pour obtenir la classification de référence. Celle-ci révèle 31 groupes de problèmes dans lesquels 43 groupes de solutions ont été identifiés. Pour déterminer la valeur du seuil de similarité qui produit les groupes les plus pertinents, différentes classifications ont été obtenues en faisant varier le seuil de similarité entre 0,9 et 0,3 et comparées avec la classification de référence. Les courbes F-mesure pour les groupes de problèmes (Figure 46) et les groupes de solutions (Figure 47) présentent les résultats et indiquent que quelle que soit la méthode de l'algorithme, la meilleure classification est obtenue avec un seuil de similarité fixé à 0,4 sur le problème et fixé à 0,3 sur la solution. Les groupes de solutions ont été obtenus à partir de la classification de référence des groupes de problèmes, toutes les méthodes ont donc eu les mêmes données en entrée pour la classification sur le problème mais aussi pour la classification sur la solution.

Pour comparer plus finement les groupes issus de chacune des méthodes de clustering ascendant hiérarchique, les classifications obtenues par chaque méthode avec les seuils de similarité optimaux (0,4 sur le problème et 0,3 sur la solution) ont été analysées par rapport à la classification de référence.

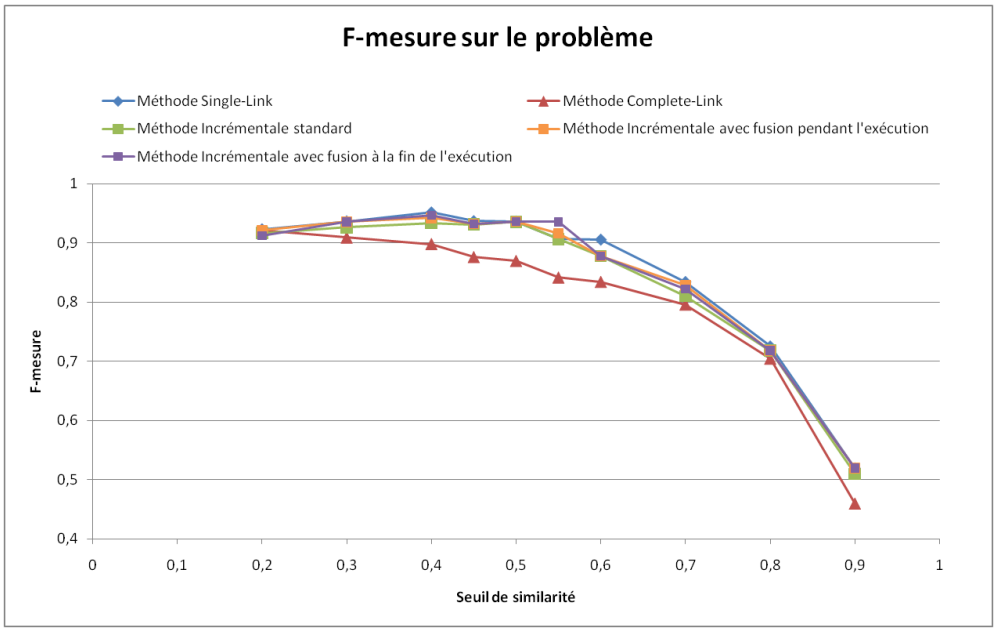


Figure 46 – F-mesure sur le problème sur une catégorie de 145 fiches d'anomalie pour les méthodes single-link, complete-link et incrémentales (2ème jeu de données)

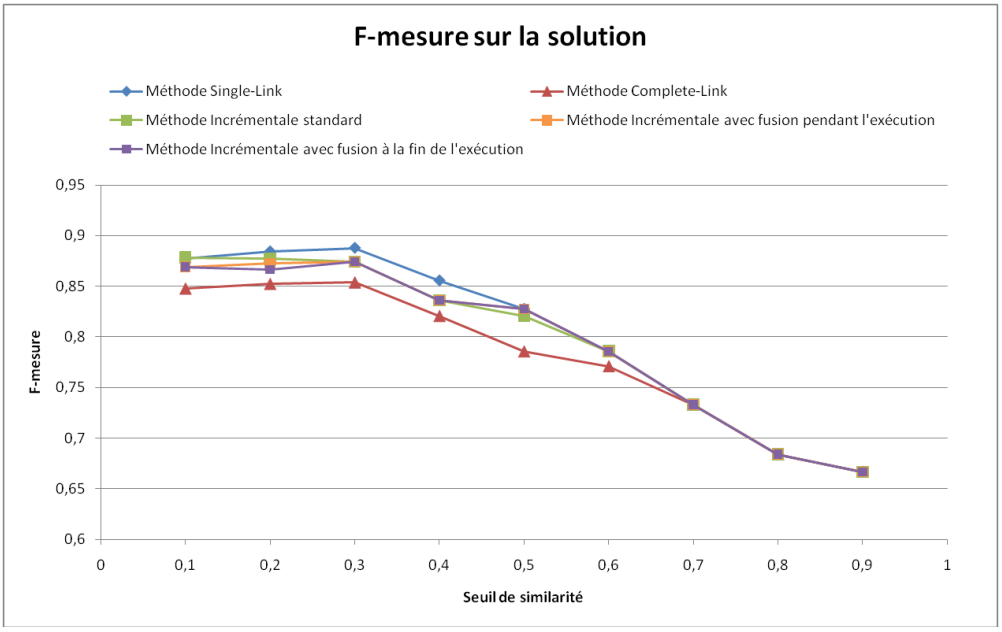


Figure 47 – F-mesure sur la solution sur une catégorie de 145 fiches d'anomalie pour les méthodes single-link, complete-link et incrémentales (2ème jeu de données)

Sur les 31 groupes de problèmes à obtenir (Figure 48) et les 43 groupes de solution à obtenir (Figure 49), les groupes issus des différentes méthodes peuvent être regroupés en 3 catégories (Table 27) :

- Ceux qui ont été parfaitement retrouvés par toutes les méthodes (F-mesure = 1)

- Ceux qui n'ont pas été parfaitement retrouvés mais dont toutes les méthodes produisent des groupes de pertinence égale (F-mesure ≠1 mais égale entre toutes les méthodes)
- Ceux qui sont différents selon les méthodes, et dont parmi eux, certaines méthodes peuvent produire les bons groupes (F-mesure différente entre les méthodes).

	Sur le problème	Sur la solution
Groupes parfaitement retrouvés	19 (61%)	27 (62%)
Groupes de pertinence égale	7 (23%)	8 (19%)
Groupes différents	5 (16%)	8 (19%)

Table 27 – Pertinence des groupes obtenus par les 5 méthodes sur une catégorie de 145 fiches d'anomalie

Cette étude fournit des résultats similaires entre les groupes de problèmes et les groupes de solutions. Etant donné que ce sont les mêmes algorithmes qui sont utilisés dans les deux cas sur des données issues de contextes identiques, cela semble tout à fait logique. Dans les deux cas, plus de 60% des groupes ont été parfaitement retrouvés, et environ 20% des groupes sont identiques pour toutes les méthodes sans être identiques à ceux de la classification de référence. L'application d'un autre algorithme de clustering ou le traitement moins statistique mais plus sémantique des informations pourraient améliorer les résultats des groupes de cette deuxième catégorie, voire même les faire passer dans la première catégorie. Néanmoins, la valeur de F-mesure de ces groupes est déjà satisfaisante pour les groupes de problèmes (au moins supérieure à 0,5 sauf pour un groupe qui n'a pu être identifié) et pour les groupes de solutions (dont la plupart se situe entre 0,6 et 0,7).

En ce qui concerne la troisième catégorie relative aux groupes différents selon les méthodes, plusieurs points sont à noter. Premièrement, sur ce jeu de données, les méthodes incrémentales standard et avec fusion pendant l'exécution fournissent des groupes de même pertinence. La troisième méthode incrémentale, avec fusion à la fin de l'exécution, est parfois légèrement différente mais reste très similaire aux deux autres méthodes incrémentales. Deuxièmement, on se rend compte des limites de la méthode single-link. Même si elle fournit parfois de meilleurs résultats que les autres méthodes (groupe s3), ce point positif est contrebalancé par le fait qu'elle peut aussi fournir des résultats nettement dégradés (groupes p23 et s32). Troisièmement, on peut relever que la méthode complete-link propose des résultats toujours satisfaisants par rapport aux autres méthodes mais cependant très souvent en retrait de la plupart d'entre elles. Enfin, les méthodes incrémentales semblent être celles avec lesquelles les groupes obtenus sont la plupart du temps de meilleure qualité et semblent être stables dans le sens où pour tous les groupes,

la valeur de la F-mesure est satisfaisante (exception faite du groupe p24 qui apparaît comme très particulier au vu des résultats de l'ensemble des méthodes).

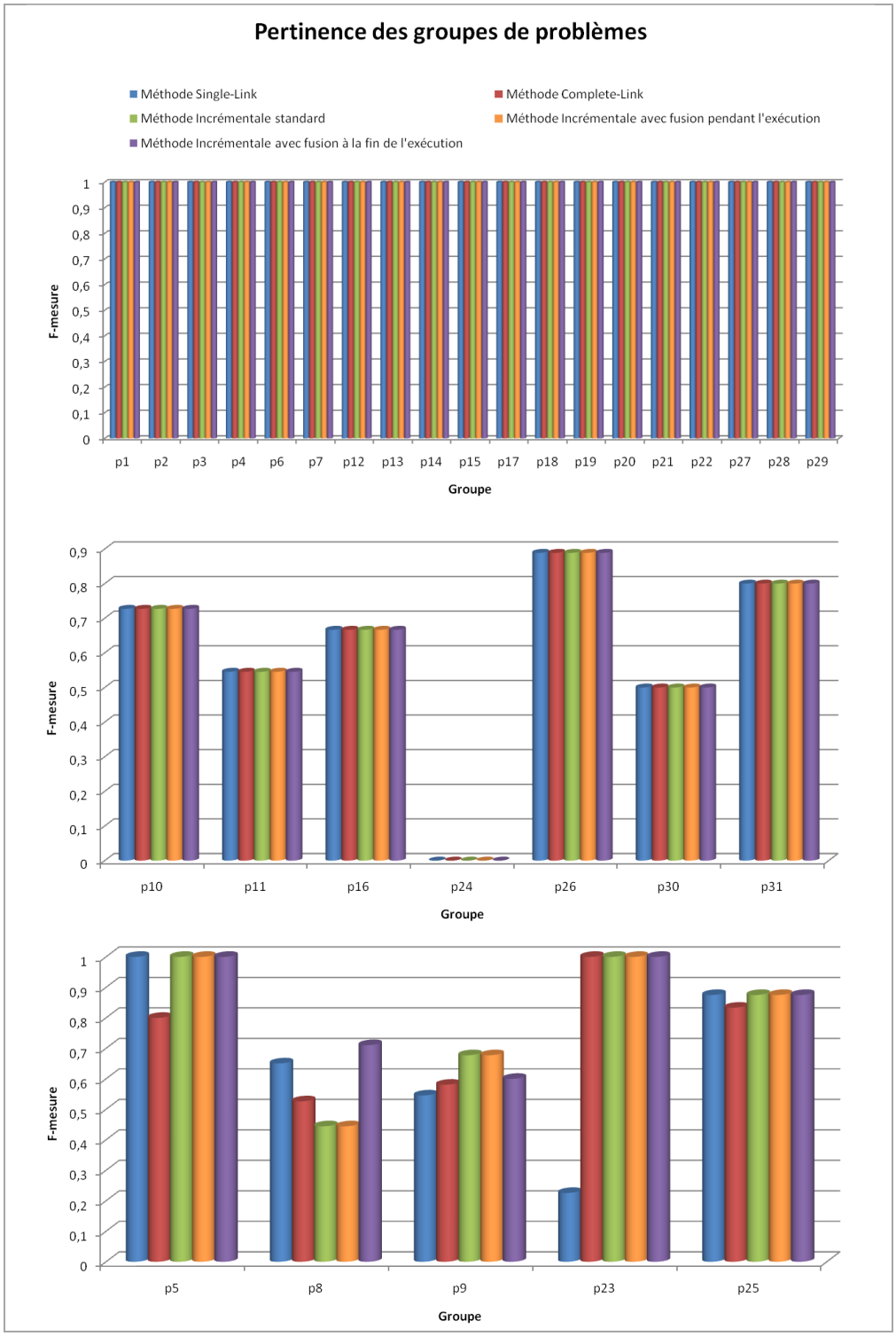


Figure 48 – Pertinence des groupes de problèmes obtenus par les 5 méthodes sur une catégorie de 145 fiches d'anomalie

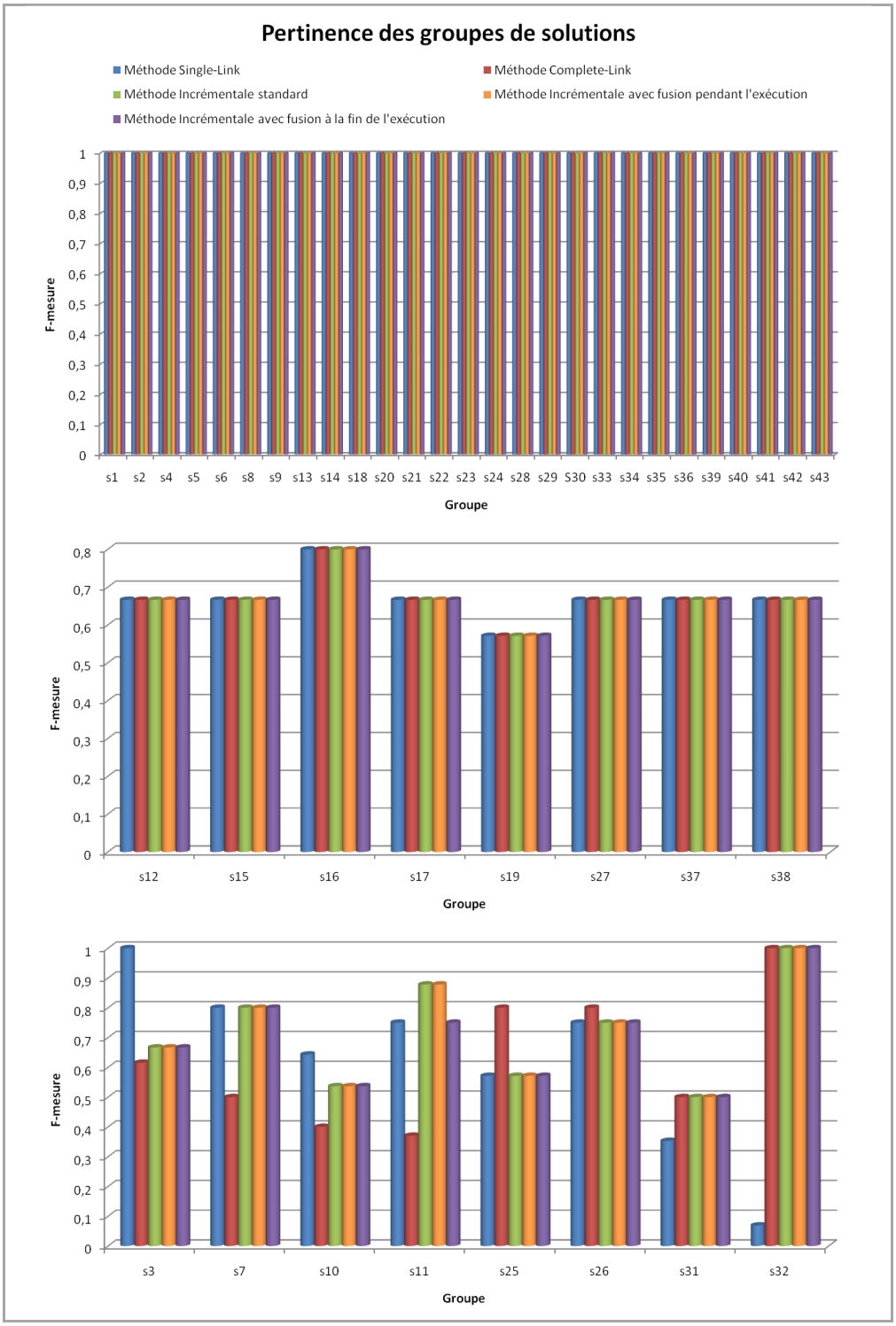


Figure 49 – Pertinence des groupes de solutions obtenus par les 5 méthodes sur une catégorie de 145 fiches d'anomalie

6.3.2.3. Evaluation de la performance

Pour mener cette étude, les méthodes de l'algorithme ont été exécutées sur des catégories de fiches d'anomalie de tailles variées (0, 47, 107, 179, 271, 476, 777, 978, 1972, 1351, 1936, 2290, 2751, 3101, 3674, 4138, 4519 et 6075 fiches d'anomalie) pour déterminer les groupes de problèmes avec un seuil fixé à 0,4 et les groupes de solutions avec un seuil fixé à 0,3. Le temps d'exécution d'une méthode pour un groupe donné inclut le temps mis par la méthode pour déterminer les groupes mais aussi celui nécessaire à la création des prototypes de groupe.

Pour cette étude, le matériel utilisé est un ordinateur portable avec un processeur quatre-cœurs cadencé à 2,67GHz et 8Go de RAM tournant sous Windows 7 Enterprise 64-bits. Il apparaît une nouvelle fois que les méthodes incrémentales sont bien plus rapides que les méthodes non incrémentales, que ce soit pour déterminer les groupes de problèmes (Figure 50) ou les groupes de solutions (Figure 51). D'une manière générale, il est logique que le temps de création des groupes de solutions soit plus court que celui de création des groupes de problèmes puisque que, même si toutes les fiches d'anomalie sont considérées dans les deux cas, moins de fiches d'anomalie sont impliquées en même temps dans le calcul de similarité sur la solution. Il faudrait un nombre de mots significatifs bien plus important sur la solution que sur le problème pour équilibrer les temps d'exécution.

Le temps d'exécution de la méthode single-link reste supérieur mais pas de manière excessive à celui des méthodes incrémentales (rares sont les catégories pour lesquelles elle est au moins deux fois plus longues que les méthodes incrémentales). En revanche, la méthode complete-link quant à elle montre ses limites avec l'augmentation du nombre de fiches d'anomalie. Son temps d'exécution étant parfois tellement élevé et afin qu'elle n'empêche pas la comparaison visuelle des autres méthodes, les figures n'en proposent qu'un affichage partiel (pour certaines catégories, le temps d'exécution se compte en heures). Cela est notamment vrai pour la création des groupes de problèmes puisque les données de départ sont les mêmes pour les deux méthodes. En revanche, pour la création des groupes de solutions, le discours est légèrement différent. En effet, les données de départ ne sont plus les mêmes, chaque méthode a produit ses propres groupes. Du fait de leur nature, la méthode single-link produit moins de groupes que la méthode complete-link. La différence de temps d'exécution sur la solution peut donc être réduite (catégories contenant 2751, 3101 et 6075 fiches d'anomalie par exemple), voire inversée (catégorie contenant 3674 fiches d'anomalie par exemple). En effet, à nombre de fiches d'anomalie égale, il est plus rapide de considérer un ensemble de petits groupes pris séparément plutôt qu'un seul groupe volumineux.

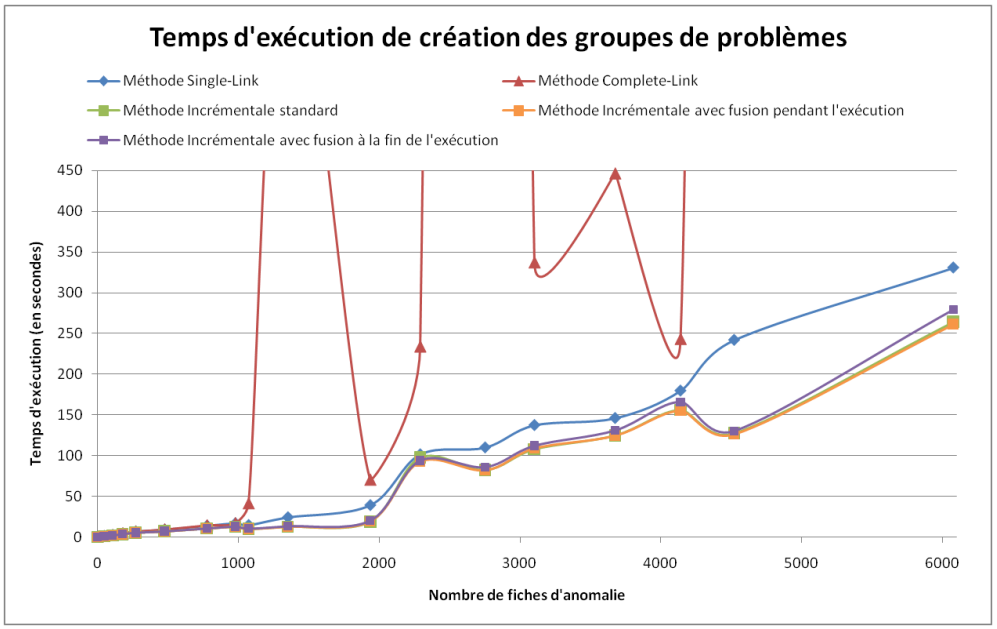


Figure 50 – Temps d'exécution de création des groupes de problèmes (2ème jeu de données)

En ce qui concerne les méthodes incrémentales, force est de constater que leurs temps d'exécution est très proche même si l'on peut noter que la méthode avec fusion pendant l'exécution est toujours plus rapide, ou au moins aussi rapide, que les deux autres (remarquable en Figure 51) alors que la méthode avec fusion à la fin de l'exécution est toujours la moins rapide, ou au moins aussi rapide, que les deux autres (remarquable en Figure 50). Cela ne s'explique que légèrement par les données et par l'ordre de sélection des fiches d'anomalie mais plutôt par la nature même des méthodes.

Dans la méthode avec fusion à la fin de l'exécution, si une fiche d'anomalie est similaire à plusieurs prototypes, alors celle-ci est mise de côté avec les autres fiches d'anomalie similaires à plusieurs prototypes. Celles-ci sont reconsidérées lorsque toutes les autres fiches d'anomalie ont été intégrées dans un groupe. Cela permet de proposer des fusions de groupes en considérant des groupes déjà bien formés. Cette méthode demande donc plus de comparaisons puisqu'au cours de l'algorithme, une même fiche d'anomalie peut être comparée deux fois avec l'ensemble des prototypes, la première fois ne faisant pas évoluer les groupes.

En revanche, dans la méthode avec fusion pendant l'exécution, si une fiche d'anomalie est similaire à plusieurs prototypes, alors des groupes peuvent être fusionnés et la fiche d'anomalie est ensuite intégrée dans le groupe pour lequel sa similarité avec le prototype est la plus élevée. Cette méthode peut donc demander moins de comparaisons puisque la fusion de deux groupes enlève une comparaison pour les fiches d'anomalie suivantes (il ne reste plus qu'un prototype avec lequel comparer au lieu de deux).

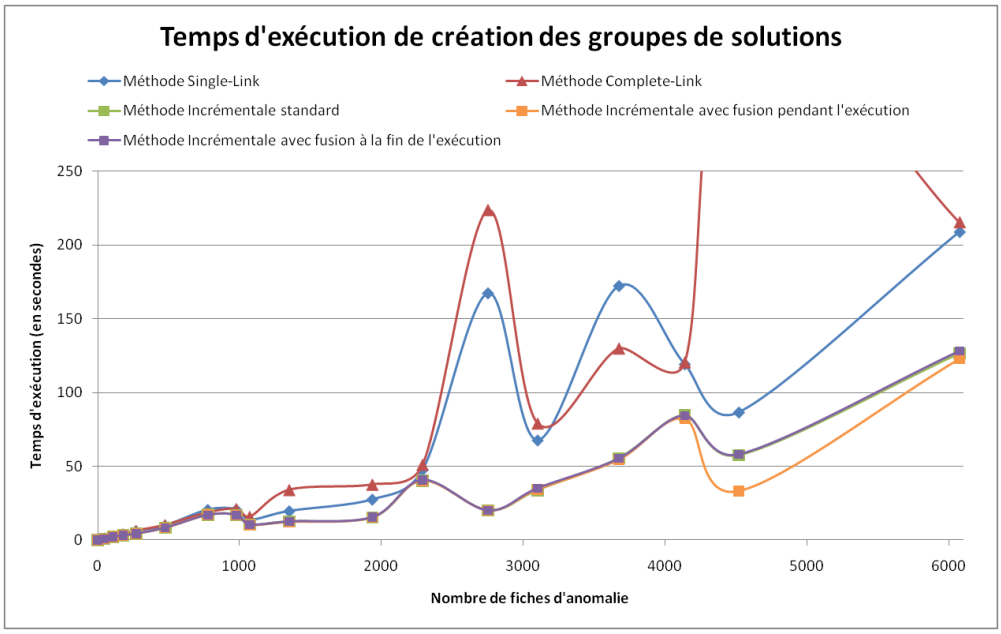


Figure 51 – Temps d'exécution de création des groupes de solutions (2ème jeu de données)

6.3.2.4. Les indicateurs

Pour évaluer la qualité des groupes obtenus, nous présentons ici les valeurs de quelques indicateurs sur plusieurs catégories, groupes de problèmes et groupes de solutions pour les trois méthodes incrémentales.

D'abord, nous nous intéressons à un groupe de problèmes et aux groupes de solutions qui le composent, tous construits de manière identique quelle que soit la méthode incrémentale.

Ensuite, nous considérons un groupe de problèmes et les groupes de solutions qui le composent, construits de manière différente selon la méthode incrémentale.

Enfin, les groupes de problèmes d'une catégorie entière sont comparés suivant les trois méthodes incrémentales.

Deux types d'indicateurs sont présentés : des indicateurs inter-groupes dont la valeur doit être maximisée puisque l'intérêt d'avoir les distances les plus grandes possibles entre les groupes et des indicateurs intra-groupe dont la valeur doit être minimisée puisqu'au contraire l'intérêt est que toutes les fiches d'anomalies d'un groupe soient très peu distantes entre elles. La valeur de tous ces indicateurs, distance minimum inter-groupes, distance moyenne inter-groupes, densité et radius, est comprise entre 0 (distance minimale) et 1 (distance maximale).

6.3.2.4.1. Groupe de problèmes et groupes de solutions identiques

Pour ce premier groupe de problèmes, les fiches d'anomalies qui y sont associées ainsi que les groupes de solutions qui le composent sont exactement les mêmes pour les trois méthodes incrémentales. Le groupe de problèmes retenu représente 14% des fiches d'anomalie de sa catégorie. A l'intérieur, les fiches d'anomalies sont réparties en 5 groupes de solutions (Figure 52).

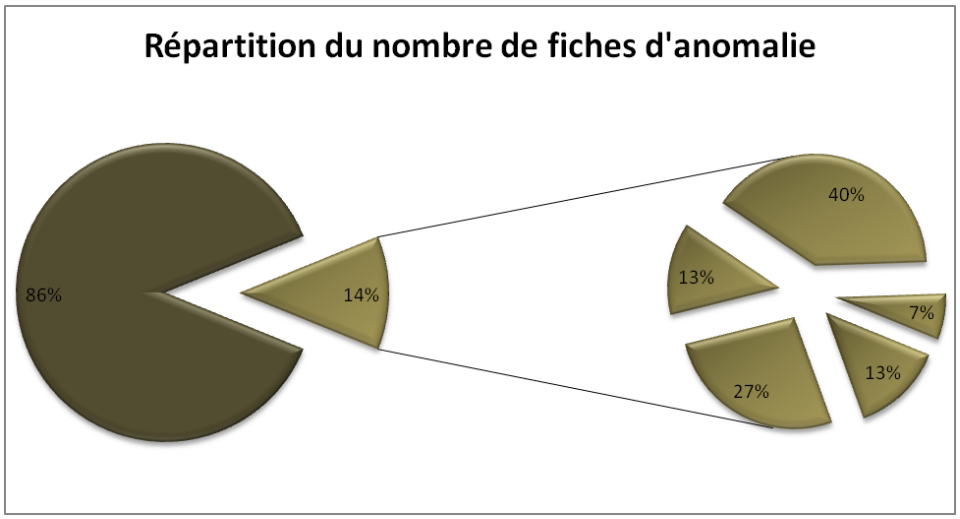


Figure 52 – Répartition du nombre de fiches d'anomalie d'un groupe de problèmes et des groupes de solutions qui le composent construits de manière identique avec les 3 méthodes incrémentales

Les indicateurs sont présentés en Table 28. Concernant le groupe de problèmes, nous pouvons relever le fait que celui-ci est très différent des autres groupes de problèmes de la catégorie (distance moyenne inter-groupes à 0,98). Même le groupe qui en est le plus proche reste bien éloigné (distance minimum inter-groupes à 0,76). C'est donc un groupe très isolé qui traduit un problème singulier dans la catégorie. De plus, la densité indique que le groupe est très compact, les fiches d'anomalies sont très similaires entre elles. Néanmoins, le radius montre qu'au moins une fiche d'anomalie se démarque des autres. Dans ce groupe très dense, cette fiche d'anomalie, même assez similaire aux autres, pourrait s'interpréter comme une fiche extrême.

En ce qui concerne les groupes de solutions, les mêmes remarques sur les indicateurs inter-groupes peuvent être formulées. Leurs valeurs élevées montrent que tous ces groupes sont bien distincts les uns des autres. Les indicateurs intra-groupes traduisent deux types de groupe. Les trois premiers ont une densité et un radius assez faible et très similaire. Contrairement au groupe de problème pour lequel on pouvait parler de fiche extrême, ces groupes semblent bien homogènes. Les deux derniers ont une densité et un radius nuls. Cela montre simplement que les fiches d'anomalie de chacun d'eux sont identiques (ou plus exactement ce sont les descripteurs qui en ont été extraits qui sont identiques). Généralement, ces valeurs sont obtenues pour

des groupes composés de très peu de fiches d'anomalie. C'est d'ailleurs nécessairement le cas pour des groupes composés d'une seule fiche d'anomalie, le prototype du groupe étant rigoureusement le même que le descripteur de la fiche d'anomalie.

	Distance minimum inter-groupes	Distance moyenne inter-groupes	Densité	Radius
	Groupe de problèmes			
14%	0,76	0,98	0,09	0,44
	Groupe de solutions			
40%	0,95	0,99	0,17	0,24
7%	0,80	0,91	0,25	0,31
13%	0,95	0,99	0,13	0,13
27%	0,83	0,95	0,00	0,00
13%	0,80	0,94	0,00	0,00

Table 28 – Indicateurs sur un groupe de problèmes et sur les groupes de solutions qui le composent construits de manière identique avec les 3 méthodes incrémentales

6.3.2.4.2. Groupe de problèmes différents

Pour ce deuxième groupe de problèmes, les fiches d'anomalie ainsi que les groupes de solutions qui le composent sont différents suivant la méthode incrémentale utilisée. Les méthodes incrémentales avec fusion pendant et à la fin de l'exécution produisent les mêmes groupes de problèmes et de solutions alors que la méthode incrémentale standard produit un groupe de problèmes dont le nombre de fiches d'anomalie est inférieur aux deux autres (Figure 53). Un des groupes de solutions est identique pour toutes les méthodes, l'autre possède moins de fiches d'anomalie avec la méthode incrémentale standard. Avec cette dernière, il existe des fiches d'anomalie qui sont isolées formant des groupes qui ne sont composés que d'une seule fiche d'anomalie alors que les autres méthodes incrémentales les intègrent dans le groupe de problèmes relevé.

Les indicateurs sont présentés en Table 29. Concernant le groupe de problèmes, tous les indicateurs donnent sensiblement les mêmes valeurs quelle que soit la méthode excepté l'indicateur de distance minimum inter-groupes qui est plus faible avec la méthode incrémentale standard. Cela montre que les deux autres méthodes sont plus pertinentes puisqu'en permettant d'avoir plus de fiches d'anomalie dans le groupe de problème, ses caractéristiques intra-groupes sont conservées et ses caractéristiques inter-groupes sont améliorées.

En ce qui concerne les groupes de solutions, encore une fois, les valeurs des indicateurs inter-groupes et intra-groupe sont sensiblement les mêmes. Les fiches d'anomalies ajoutées dans le deuxième groupe de solutions par les méthodes incrémentales avec fusion pendant et à la fin de l'exécution sont donc parfaitement à leur place.

Dans la méthode incrémentale standard, ces fiches d'anomalie qui n'ont pas été intégrées ont donné lieu à la création de deux groupes de problèmes. Leur distance minimum inter-groupes à 0,53 et 0,55 renforce l'intérêt de la fusion pendant ou à la fin de l'exécution. Leur distance moyenne inter-groupes est légèrement supérieure (0,93). Comme ce sont des groupes plus petits, leur densité et leur radius sont très faibles (aux alentours de 0,05).

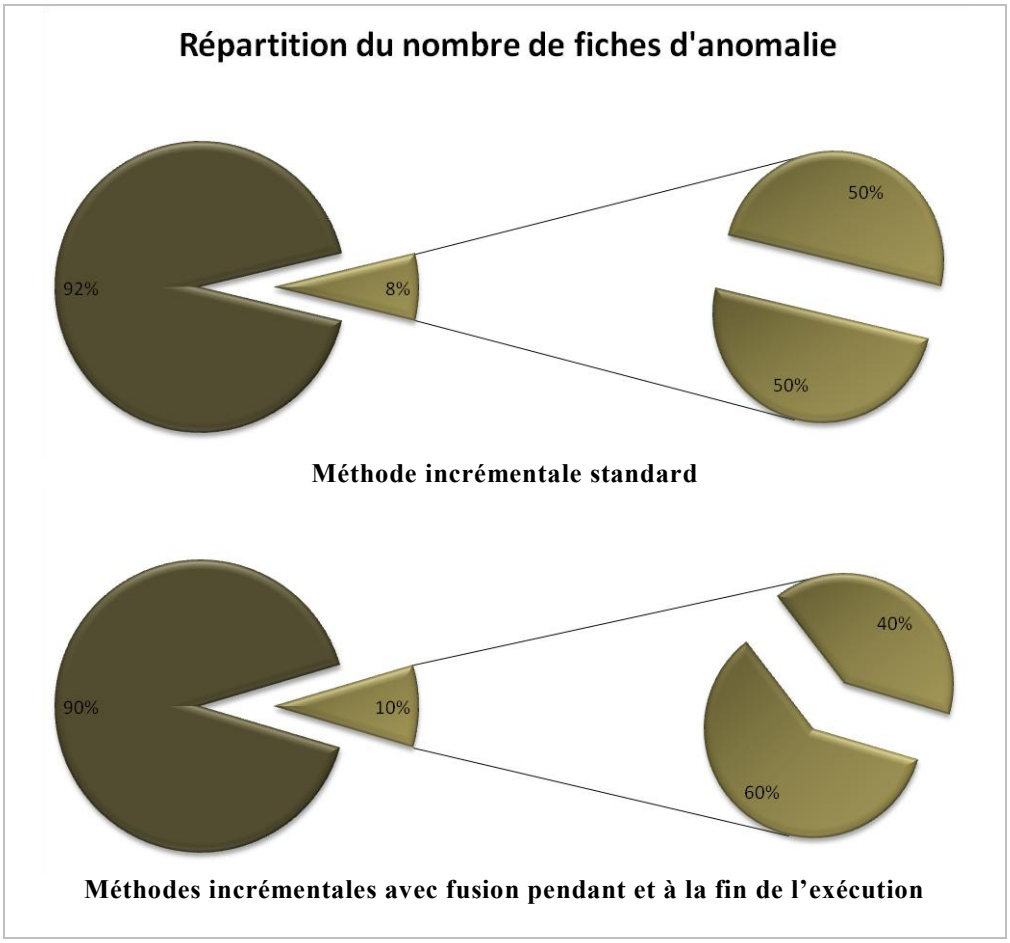


Figure 53 – Répartition du nombre de fiches d'anomalie d'un groupe de problèmes et des groupes de solutions qui le composent construits de manière différente avec les 3 méthodes incrémentales

	Distance minimum inter-groupes	Distance moyenne inter-groupes	Densité	Radius
Méthode incrémentale standard				
Groupe de problèmes	0,53	0,91	0,33	0,43
Groupes de solutions	1,00 & 1,00	1,00 & 1,00	0,40 & 0,25	0,43 & 0,31
Méthode incrémentale avec fusion pendant l'exécution				
Groupe de problèmes	0,61	0,92	0,36	0,47
Groupes de solutions	1,00 & 1,00	1,00 & 1,00	0,40 & 0,27	0,43 & 0,33
Méthode incrémentale avec fusion à la fin de l'exécution				
Groupe de problèmes	0,61	0,92	0,36	0,47
Groupes de solutions	1,00 & 1,00	1,00 & 1,00	0,40 & 0,27	0,43 & 0,33

Table 29 – Indicateurs sur un groupe de problèmes construit de manière différente avec les 3 méthodes incrémentales

6.3.2.4.3. Groupes de problèmes d'une catégorie complète

Pour cette dernière étude sur les indicateurs, nous nous intéressons à une catégorie complète. Le nombre de groupes de problèmes obtenus est le même quelle que soit la méthode incrémentale utilisée. Les méthodes autorisant la fusion ont produit exactement les mêmes groupes. La méthode standard diffère des deux autres sur trois groupes. L'un des groupes possède 8 fiches d'anomalie en moins. Ces 8 fiches d'anomalie sont présentes dans deux autres groupes, 5 dans l'un et 3 dans l'autre (Figure 54).

Pour chacun des 21 groupes de problèmes obtenus, nous présentons la distance minimum inter-groupes (Figure 55), la distance moyenne inter-groupes (Figure 56), la densité (Figure 57) et le radius (Figure 58).

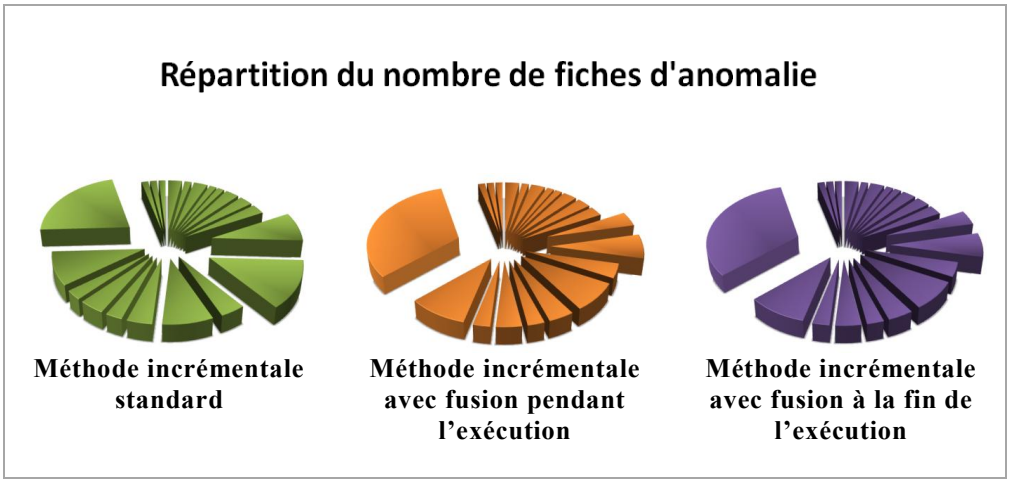


Figure 54 – Répartition du nombre de fiches d'anomalie par groupes de problèmes d'une catégorie avec les 3 méthodes incrémentales

Même si pour une grande partie des groupes, les trois méthodes produisent des groupes dont la valeur des indicateurs est la même, dans l'ensemble, nous pouvons affirmer que les méthodes incrémentales avec fusion produisent des groupes plus pertinents que la méthode incrémentale standard. En effet, à quelques exceptions près, tous les indicateurs de tous les groupes fournissent des valeurs égales ou meilleures avec ces méthodes. Les indicateurs inter-groupes sont plus élevés et les indicateurs intra-groupe sont plus faibles. De plus, lors des rares fois où la méthode incrémentale standard est meilleure, elle ne l'est que très légèrement (groupe 12 avec les indicateurs inter-groupes, groupe 18 avec le radius) alors que, lorsqu'elle est moins bonne, la différence est parfois assez importante (groupe 8 avec les indicateurs inter-groupes, goupes 9 et 10 avec les indicateurs intra-groupe).

C'est l'indicateur de distance moyenne inter-groupes qui renforce cette impression que les méthodes incrémentales avec fusion sont plus pertinentes que la méthode incrémentale standard. Pour les autres indicateurs, il n'y a de différence qu'avec quelques groupes. En revanche, seuls 33% des groupes ont la même distance moyenne inter-groupes avec les trois méthodes incrémentales. Pour les autres groupes, les valeurs obtenues sont soit meilleures avec la méthode standard (19%), soit meilleure valeur avec les méthodes autorisant la fusion (48%).

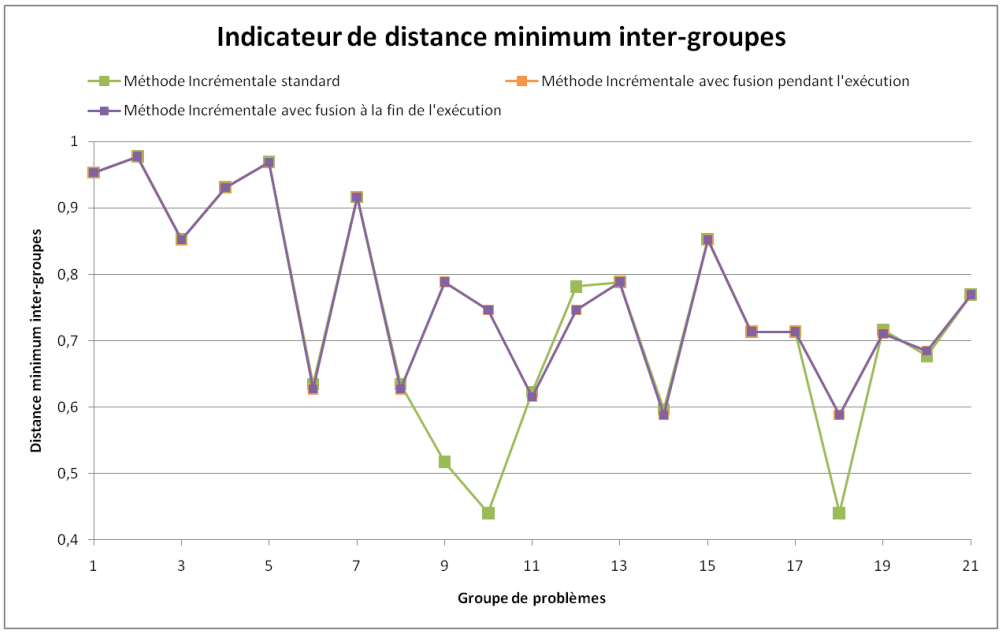


Figure 55 – Distance minimum inter-groupes par groupes de problèmes d'une catégorie avec les 3 méthodes incrémentales

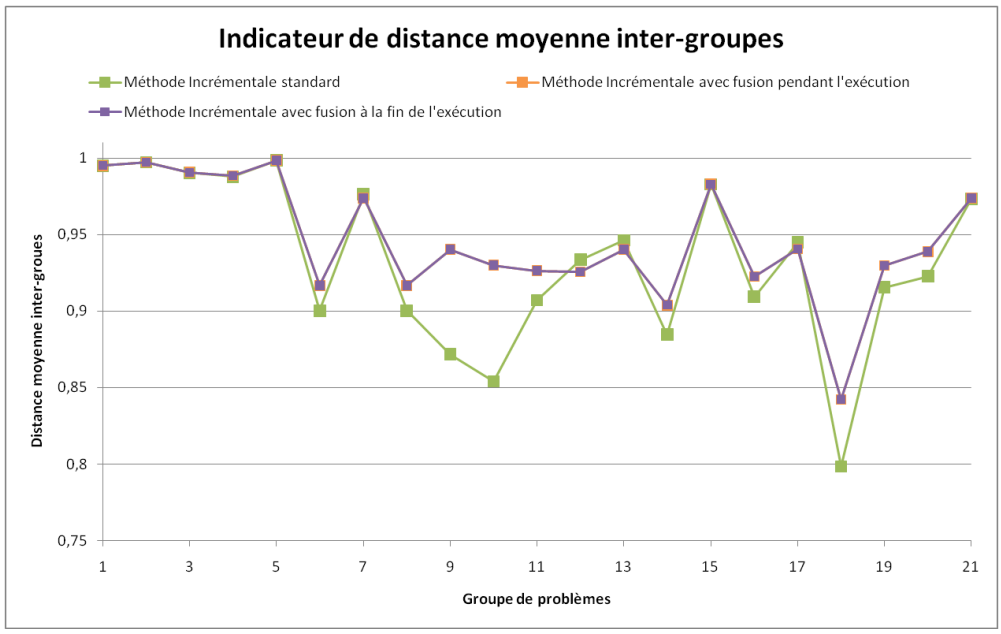


Figure 56 – Distance moyenne inter-groupes par groupes de problèmes d'une catégorie avec les 3 méthodes incrémentales

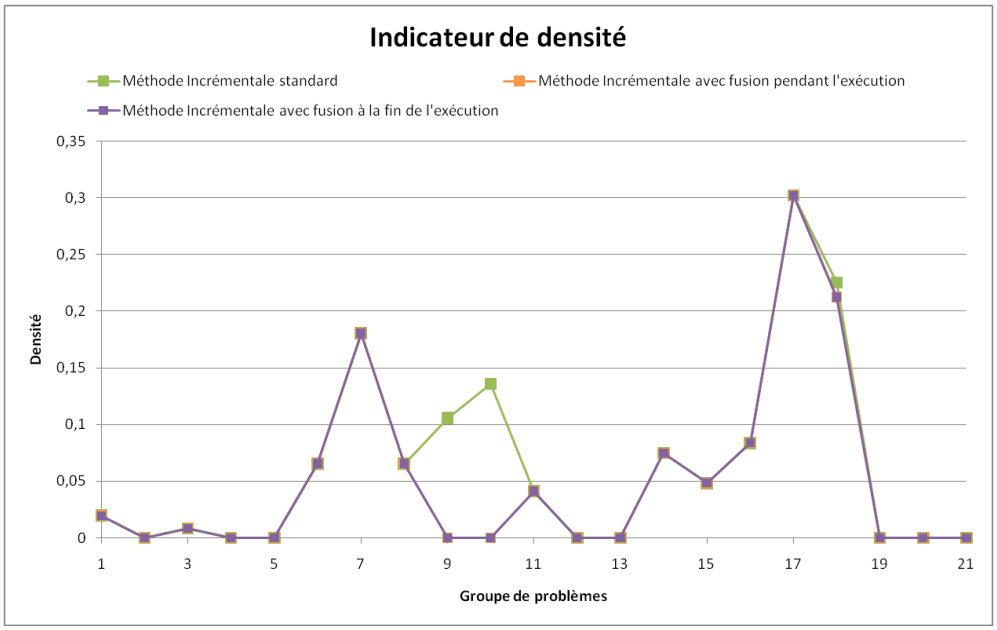


Figure 57 – Densité par groupes de problèmes d'une catégorie avec les 3 méthodes incrémentales

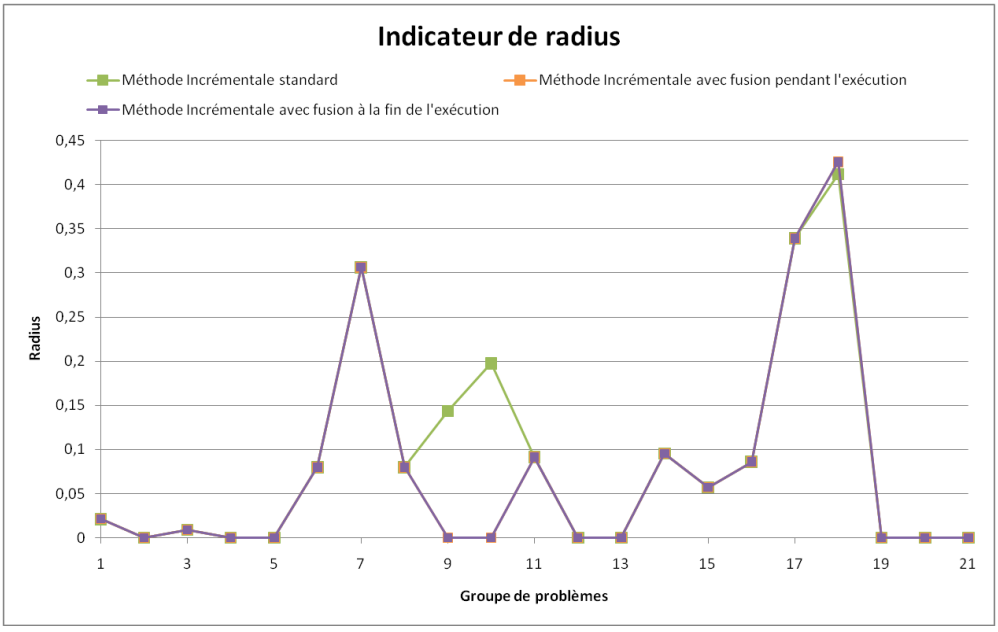


Figure 58 – Radius par groupes de problèmes d'une catégorie avec les 3 méthodes incrémentales

6.4. Conclusion

Ces deux évaluations montrent que le modèle de descripteur de fiche d'anomalie et le framework global de gestion de fiches d'anomalie exposés respectivement dans les chapitres 4 et 5 sont implémentables et fournissent des résultats très satisfaisants aussi bien en termes de qualité des groupes obtenus que de temps d'exécution de l'algorithme de classification. Le modèle de descripteur de fiche d'anomalie a été enrichi par les éléments du framework et par la manière dont nous avons choisi de construire les descripteurs. Le prototype logiciel est composé de trois modules :

- Le module de classification des fiches d'anomalie, exécuté en offline et qui peut durer plusieurs heures.
- Le module de recherche de solutions, utilisé en online et qui ne prend que quelques secondes.
- Le module d'informations sur les groupes qui peut être exécuté en mode online ou offline. Le mode online permet de manipuler des informations qui concernent exactement le moment où celles-ci sont demandées. Le mode offline permet de manipuler des informations qui concernent une version antérieure déjà calculée.

Lorsque l'on ne considère que les évaluations qualitatives avec notamment la pertinence des groupes de problèmes et des groupes de solutions (Figure 48 et Figure 49 du §6.3.2.2) celles-ci révèlent qu'au niveau de la qualité des groupes, toutes les méthodes présentent des résultats assez similaires. On peut tout de même relever deux points sur les méthodes non-incrémentale. Premièrement, les résultats de la méthode complete-link sont régulièrement inférieurs aux autres méthodes. Deuxièmement, bien que la méthode single-link ait les meilleurs résultats sur certains groupes, elle possède aussi les pires sur d'autres à cause de sa propension à créer des groupes volumineux. De plus, même si deux méthodes incrémentales qui s'attaquent au problème de l'ordre de sélection des documents ont été proposées, les résultats obtenus ne permettent pas d'affirmer qu'elles sont meilleures que la méthode incrémentale standard tant les résultats sur la qualité des groupes obtenus sont similaires. En revanche, lorsque l'on considère les indicateurs sur les groupes obtenus (Figure 55 à Figure 58 du §6.3.2.4.3), nous pouvons noter que les méthodes incrémentales avec fusion produisent dans l'ensemble des groupes de meilleure qualité que la méthode incrémentale standard. En ce qui concerne le temps d'exécution, ces évaluations montrent que les méthodes non-incrémentales single-link et complete-link ne supportent pas le passage à l'échelle et ne sont donc pas utilisables dans notre contexte industriel.

Les meilleurs résultats, que ce soit en termes de qualité des groupes construits ou de performance en temps d'exécution, sont obtenus en utilisant les méthodes incrémentales avec fusion. Le modèle de descripteur de fiche d'anomalie avait déjà été validé dans le chapitre 4, ce chapitre évaluait donc le framework proposé dans le chapitre 5. Le processus de classification de fiches d'anomalie utilisant l'algorithme de clustering ascendant hiérarchique permet une bonne structuration de la base de fiches d'anomalie. Le processus de recherche de fiches d'anomalie similaires utilisant le même algorithme puisqu'il cherche à classer une nouvelle fiche d'anomalie incomplète (on ne connaît que les informations sur le problème) est donc tout aussi pertinent. Il facilite la maintenance corrective en assistant l'utilisateur dans la recherche d'une solution à un problème déjà survenu dans le passé. La base structurée de fiches d'anomalie couplée aux indicateurs facilite l'analyse des groupes, analyse étant à la base de la maintenance préventive.

Conclusion et perspectives

Conclusion

Notre travail s'intéresse à la capitalisation de connaissances issues de fiches d'anomalie pour faciliter les maintenances corrective et préventive, particulièrement dans un contexte de production industrielle. Notre point de départ est le processus de Knowledge Discovery in Databases (Fayyad et al., 1996a). Il définit différentes étapes auxquelles nous avons donné pour chacune une dimension spécifique à la gestion d'anomalies afin de faciliter le travail de correction du mainteneur aussi bien en temps passé à la résolution de l'anomalie qu'en pertinence de la solution utilisée et de faciliter le travail de prévention en mettant en évidence les anomalies récurrentes. Les travaux réalisés peuvent être répartis en trois parties.

Le point de départ de ces travaux est l'étude des problématiques de la maintenance industrielle et de la maintenance logicielle pour dégager celles de notre maintenance de processus de production industrielle. Le problème de qualité de la maintenance, du retour d'expérience et de la formalisation du processus de maintenance sont souvent évoqués et des solutions existent mais elles ne permettent pas d'exploiter la connaissance capitalisable sur les anomalies passées. Nous n'avons pas relevé de travaux qui s'intéressent au document en tant que support à une anomalie, à sa structure, à son contenu. Pourtant nous pensons que c'est un élément essentiel pour extraire de la connaissance pertinente. L'objectif de ce travail consistait donc à capitaliser la connaissance des fiches d'anomalie. Mais, en fonction du domaine d'étude et des différents types d'anomalie d'un même domaine, la structure et le contenu des fiches varie. Notre contribution, la proposition et l'évaluation d'un modèle de descripteur de fiche d'anomalie, vise à résoudre ce problème. Nous avons aussi étudié comment est représentée la connaissance une fois qu'elle a été extraite. Les patterns sont le support à la connaissance. Ils regroupent la connaissance sous une forme structurée et sont spécifiques au domaine considéré. Dans celui qui nous concerne, les patterns d'anomalie (d'exception) présents dans la littérature possèdent deux éléments en commun particulièrement intéressants : le problème que le pattern tente de résoudre et la solution à appliquer pour résoudre le problème. Pour déterminer des patterns de manière automatique, il est nécessaire de regrouper les informations selon

un ou plusieurs critères de similarité. Pour cela, nous nous sommes intéressés à différents environnements de classification de documents afin de déterminer leurs forces et leurs faiblesses et choisir l'environnement le plus adapté dans le framework que nous proposons par la suite.

Nous nous sommes donc focalisés sur la modélisation du document qui sert de support à une anomalie, sa fiche d'anomalie. L'analyse de fiches d'anomalie industrielle a corroboré les deux éléments principaux d'un pattern d'anomalie. En effet, au-delà des informations organisationnelles ou de traçabilité qui ne permettent presque jamais d'expliquer une anomalie, une fiche d'anomalie est composée de deux natures d'informations, celles qui décrivent le problème et celles qui décrivent la solution mise en œuvre. Nous avons donc proposé un modèle de descripteur de fiche d'anomalie basé notamment sur ce constat. La propriété de qualité descriptive partitionne les informations sur le contenu (information descriptive de problème, de solution ou non-descriptive) alors que la propriété de structuration partitionne les informations sur la structure (information libre écrite en plein texte ou information contrainte beaucoup plus précise).

Nous avons ensuite cherché à placer le modèle de descripteur de fiche d'anomalie au cœur d'un environnement visant à faciliter les activités de maintenance corrective et préventive. Pour cela, nous avons proposé un framework pour classer les fiches d'anomalie de manière à pouvoir réutiliser leurs informations. L'approche consiste notamment en la classification des fiches d'anomalie sous forme de groupes de problèmes et de groupes de solution. Afin d'améliorer les performances et la qualité des groupes produits, deux versions supplémentaires de l'algorithme de clustering ascendant hiérarchique ont été présentées. Les méthodes non-incrémentales n'étant pas utilisables sur de larges volumes de données, nous nous sommes tournés vers la méthode incrémentale. Cependant, cette dernière est sensible à l'ordre de sélection des documents. Pour diminuer l'importance de ce problème, nous avons proposé deux méthodes incrémentales permettant la fusion de groupes au cours de l'exécution et à la fin de l'exécution de l'algorithme. De plus, pour évaluer la qualité des groupes obtenus, des indicateurs présents dans la littérature ont été sélectionnés. Ils permettent de juger de la pertinence du groupe lui-même, par rapport aux autres groupes et par rapport à l'absence de groupes.

Pour démontrer l'applicabilité du modèle de descripteur de fiche d'anomalie, celui-ci a été évalué sur deux bases de fiches d'anomalie réelle, une base contenant des fiches de bug logiciel et une autre contenant des fiches d'anomalie de production industrielle. Il a été aussi comparé au modèle utilisé par le logiciel de suivi de bugs logiciels BugZilla. De plus, le framework de gestion de fiches d'anomalie a lui aussi été évalué sur ces deux bases réelles, en termes de qualité des groupes obtenus et de performance d'exécution. Notre

principale contribution, le modèle de descripteur de fiche d'anomalie, ainsi que le framework dans lequel nous l'intégrons s'appliquent au processus de Knowledge Discovery in Databases pour lui donner une dimension spécifique à la gestion d'anomalies (Figure 59).

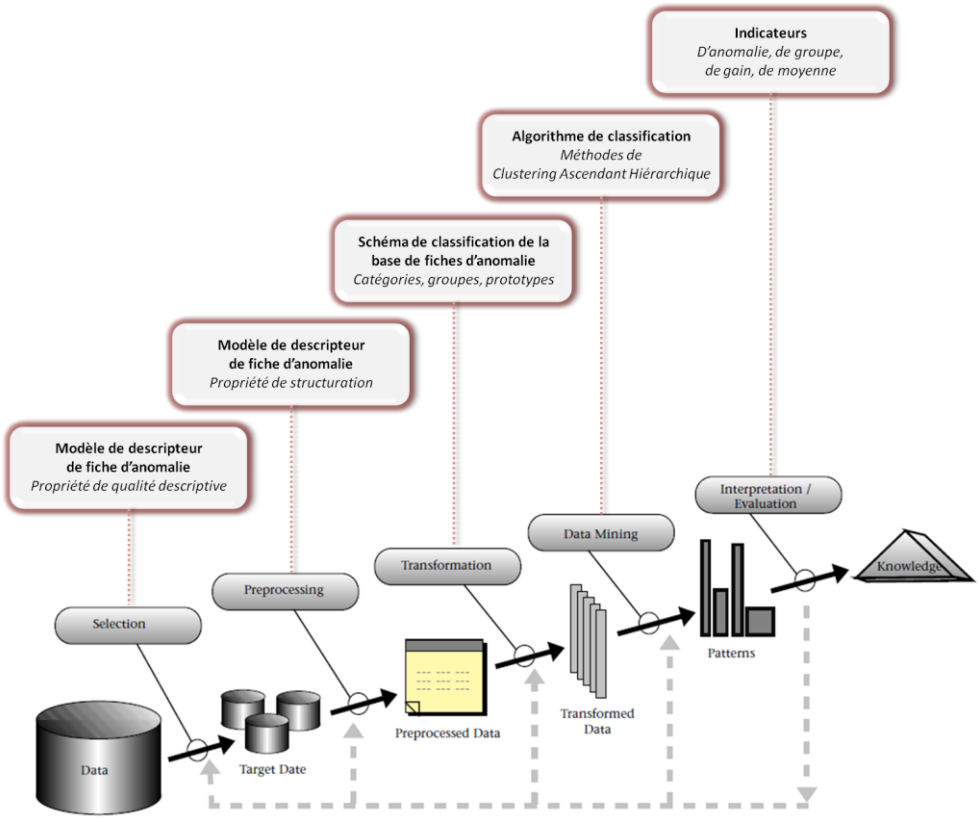


Figure 59 – Adaptation à la gestion d'anomalies du processus de KDD défini par (Fayyad et al., 1996a)

Les deux premiers traitements que sont la *selection* et le *preprocessing* sont respectivement remplis grâce aux propriétés de qualité descriptive et de structuration des attributs définis dans modèle de descripteur de fiche d'anomalie. Les traitements suivants de *transformation*, *data mining* et *interpretation/evaluation* sont respectivement réalisés grâce à l'approche de classification des fiches d'anomalie, à l'algorithme de clustering ascendant hiérarchique et aux indicateurs. Le modèle de descripteur de fiche d'anomalie ainsi que le framework ont été implémentés et évalués. Les *patterns* issus de l'étape de *data mining* correspondent aux groupes de problèmes et de solutions que nous obtenons, un pattern d'anomalie associant un problème à une solution à ce problème. Dans notre cas, plusieurs solutions peuvent être associées à un seul problème. L'application de ce framework permet de proposer automatiquement des solutions pertinentes à un nouveau problème (maintenance corrective) et d'analyser les groupes obtenus pour identifier les problèmes récurrents et faire remonter cette information aux équipes de conception (maintenance préventive).

Perspectives à court terme

Pour représenter une fiche d'anomalie, les mots significatifs sont extraits de ses données et pondérés. Ce choix a été fait car les domaines considérés sont toujours fermés avec un vocabulaire contrôlé. L'application de techniques supplémentaires telle que la suppression de suffixes renforce l'intérêt de l'utilisation de cette représentation simple à mettre en œuvre. Une autre approche est l'utilisation de la technique des n-grammes qui considère une fenêtre de longueur n de suite de lettres ou de suite de mots. Cette fenêtre parcourt l'ensemble du document par déplacements successifs d'une lettre ou d'un mot. Pour aller encore plus loin dans la représentation des documents, des techniques plus axées sur l'aspect sémantique peuvent être employées. Par exemple, les techniques de réduction de dimensions, notamment l'approche LSA (cf. §2.4.2.4) sont une possibilité, une ontologie du domaine hiérarchisant les concepts qui y sont manipulés peut être élaborée ou encore une base lexicale telle que WordNet (http://wordnet.princeton.edu/) peut être utilisée. Cette base regroupe les noms, verbes, adjectifs et adverbes de la langue anglaise en ensembles de synonymes, appelés synsets (synonym sets). Plusieurs relations entre ces synsets permettent de les classifier et de les organiser entre eux. Suivant le type du mot, différentes relations peuvent être appliquées. Seule la partie concernant les noms propose une organisation hiérarchique profonde, celle des verbes est plus à plat. Quant aux adjectifs et adverbes, cette hiérarchie est inexistante. WordNet n'est donc pas un outil d'extraction de termes d'un texte mais il doit être vu comme une aide à cette extraction. Il peut notamment servir à la mise en place des synonymes, qu'il propose automatiquement avec les synsets mais aussi au regroupement de termes sous des concepts plus généraux grâce aux hyperonymes. De plus, il fait la distinction entre les types de mots (nom, verbe, adjectif, adverbe), ce qui peut s'avérer très utile si l'on ne souhaite en traiter ou en ignorer certains.

Lors de la pondération des mots significatifs d'une fiche d'anomalie, tous les mots significatifs de son descripteur sont considérés de la même façon. On peut cependant imaginer que certains mots significatifs soient plus importants, soit par le mot lui-même soit par l'attribut duquel ils sont extraits. Un poids d'importance pourrait être donné a priori. Pour éviter de « noyer » un mot significatif extrait d'un attribut contraint au milieu de mots significatifs extraits d'attributs libres, un poids plus élevé pourrait être donné aux mots significatifs extraits d'attributs contraints. Ceux-ci sont souvent très descriptifs et représentatifs et il peut être intéressant de les mettre en avant. Différents poids pourraient même être donnés aux attributs contraints pour hiérarchiser l'importance des informations qu'ils contiennent. De la même manière, un expert peut indiquer qu'un (ou plusieurs) attributs libres sont très descriptifs et un poids différents des autres peut être donné à tous les mots significatifs qui

en sont extraits. On peut aussi imaginer effectuer cette pondération à un niveau de granularité encore plus fin en travaillant sur les mots significatifs eux-mêmes. Un expert pourrait donner des poids a priori sur les mots significatifs eux-mêmes s'il sait que certains expliquent mieux ou moins bien l'anomalie que d'autres.

Un autre point d'amélioration concerne la limitation actuelle du modèle de descripteur de fiche d'anomalie sur la propriété de structuration. Compte tenu du domaine, de l'étude des différentes fiches d'anomalie que nous avons effectuée, nous n'avons souhaité considérer que les informations textuelles. Cependant, il existe aussi des informations numériques, de quantité et d'importance souvent bien plus faibles, qu'il peut être intéressant de prendre en compte dans le descripteur des fiche d'anomalie. Dans ce travail, tous les mots significatifs sont extraits des valeurs renseignées dans les attributs et sont regroupés dans la partie problème ou solution du descripteur de la fiche. Ce choix a été fait car les éléments numériques apparaissaient moins importants pour notre étude et qu'un mot significatif présent dans des attributs différents de deux fiches d'anomalie devait améliorer la similarité entre elles. Pour traiter des éléments numériques cardinaux par exemple, il faudrait donner une granularité plus fine au descripteur. Les mots significatifs doivent être regroupés non seulement dans les parties problème et solution mais aussi, à l'intérieur d'elles, dans des parties spécifiques à chacun des attributs. Cela permettrait d'effectuer des comparaisons uniquement entre mots significatifs d'attributs (stockant des valeurs cardinales).

Perspectives à moyen et long terme

A plus long terme, d'autres perspectives sont envisageables. Le framework proposé est conditionné par le fait qu'il doit être adapté aux activités de maintenance corrective et préventive. La structuration de la base de fiches d'anomalie en divisions successives (catégories de problèmes, groupes de problèmes et groupes de solutions) qui en a été dégagée est issue de cet objectif. Pour chaque type de problème, représenté par un groupe de problèmes, plusieurs types de solutions, représentées par des groupes de solutions, sont associés. Il serait aussi intéressant de considérer cette structuration dans l'autre sens, c'est-à-dire trouver des problèmes différents ayant une solution similaire en commençant par déterminer les groupes de solutions puis à l'intérieur de ceux-ci, les groupes de problèmes. De cette manière, pour chaque type de solution, plusieurs types de problèmes seraient associés. Cela permettrait de définir une solution générale à un problème global plutôt qu'à plusieurs problèmes spécifiques. Une solution type pour un

ensemble de problèmes pourrait être définie au lieu de déterminer plusieurs solutions types très similaires pour résoudre différents types de problème.

Nous n'intégrons pas dans ce travail de connaissances expertes a posteriori. L'expert n'intervient qu'en amont du framework pour identifier la structuration et la qualité descriptive des attributs et pour déterminer les attributs critères de catégories de fiches d'anomalie. Après cela, les différents traitements sont exécutés pour structurer la base de fiches d'anomalie et pour proposer des solutions à un nouveau problème. L'expert pourrait aussi intervenir en aval du framework. Sans réaliser une analyse complète des groupes obtenus (irréalisable avec le nombre de fiches d'anomalie important qui est en jeu), il pourrait faire évoluer les groupes obtenus au cours du temps lorsqu'il se rend compte de lacunes de la classification automatique. Il pourrait vouloir fusionner deux groupes en un seul ou au contraire diviser un groupe. Il pourrait aussi vouloir travailler à un niveau encore plus fin en déplaçant une fiche d'anomalie d'un groupe à un autre ou en sortant une fiche d'anomalie d'un groupe pour créer un nouveau groupe. Dans tous les cas, il faudrait comprendre et garder l'action en mémoire pour en tenir compte lorsqu'on relancera la structuration de la base de fiches d'anomalie. Plusieurs cas justifient la réexécution de la structuration de la base de fiches d'anomalie : pour utiliser de nouveaux seuils de similarité, pour intégrer les nouvelles fiches d'anomalie depuis la dernière classification (dans le cas d'ajout par lots) ou pour repartir d'une base plus propre, l'ajout de fiches d'anomalie au fur et à mesure de leur arrivée pouvant dénaturer la pertinence des groupes (dans le cas d'un ajout au fil de l'eau). Ces cas d'ajouts sont particulièrement sérieux si l'algorithme de classification utilisé n'est pas incrémental. La réexécution devrait donc redéfinir les groupes de la manière la plus pertinente tout en réalisant les modifications appliquées par un expert auparavant.

Enfin, ce travail de thèse que nous avons placé dans un contexte spécifique de gestion d'anomalies de production industrielle peut se voir appliqué dans d'autres domaines. Ainsi, dans le domaine médical, partons du constat qu'une personne malade présente des symptômes caractéristiques d'une ou plusieurs maladies et qu'un traitement doit lui être prodigué pour la guérir. Une personne en bonne santé est dans un état standard. Les symptômes dont une personne malade souffre correspondent au problème et le traitement administré correspond à la solution. Il est donc possible de mettre en évidence des patterns associant un ensemble de symptômes et de caractéristiques patients à différents traitements, chacun guérissant d'une ou plusieurs maladies. Evidemment, le spécialiste est le seul à pouvoir déterminer le traitement adapté mais il est possible de l'assister en lui proposant les différents traitements répondant aux symptômes. Le cas d'une patiente d'une cinquantaine d'années qui a de la fièvre, des nausées et qui souffre de douleurs abdominales est présenté en annexe (Annexe 1).

En vidéosurveillance, la détection d'un comportement inhabituel peut donner lieu à une action en temps réel. Un comportement inhabituel peut se traduire de différentes façons selon le cadre d'application : (a) quelqu'un qui passe dans le champ d'une caméra alors que le lieu est interdit d'accès à cette heure-là, (b) quelqu'un qui s'approche d'une zone radioactive sans équipement adapté, (c) quelqu'un qui enjambe un portique de sécurité pour accéder au métro ou encore (d) plusieurs personnes devant un distributeur automatique de billets traduisant une agression. Les modèles de comportements standards peuvent être définis a priori (tout comme on définit un processus standard en production industrielle) et ce sont les comportements inhabituels, les exceptions, qu'il faut traiter de manière spécifique. Ceux-ci sont comparables à des problèmes auxquels différentes solutions sont applicables : l'envoi d'un agent de sécurité sur place (a, b, c, d), le déclenchement d'une alarme (a, b), le verrouillage de l'équipement (d), etc. Ce genre d'exceptions est donc modélisable par des patterns qui peuvent être appris par l'expérience. Ainsi, lors de la détection d'un comportement inhabituel, une action peut être effectuée immédiatement dans le cadre d'un système expert ou les différentes actions possibles peuvent être proposées à la personne responsable dans le cadre d'un système de recommandation. Le projet LINDO (Large scale distributed INDexation of multimedia Objects) s'intéresse à cette problématique (Brut et al., 2011a) et permet d'exécuter une succession d'algorithmes d'indexation comme solution au problème détecté, au comportement inhabituel (Brut et al., 2011b).

Toujours dans l'optique d'effectuer une action en temps réel, un autre domaine dans lequel appliquer ce travail de thèse est celui du socio-technique ambiant. Dans ce domaine, on va s'intéresser à des exceptions au sens comportement de la personne, par exemple dans le cadre du maintien à domicile. Les modèles de comportements standards peuvent être définis a priori. Considérant des personnes âgées, les exceptions peuvent concerner le fait qu'une personne soit à terre depuis trop longtemps, qu'elle soit dans une posture inadaptée ou encore que la lumière soit allumée de manière inhabituelle pendant la nuit. Ainsi, comme dans le contexte de la vidéosurveillance, des patterns d'exception associant un type de problème à des solutions pour assister la personne âgée peuvent être déterminés et lorsque le problème survient, une solution peut être immédiatement appliquée ou des solutions pertinentes peuvent être proposées à la personne en charge de leur suivi. La répétition d'un ou de plusieurs problèmes peut même dénoter une dégradation de la santé de la personne.

Références

Aamodt et al., 1994 — A. Aamodt, E. Plaza. *Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches*. AI Communication, vol. 7, pp. 39-59, 1994.

Abdi et al., 2010 — H. Abdi, L. Williams. *Principal Components Analysis*. Wiley Interdisciplinary Reviews: Computational Statistics, vol. 2, n°4, pp. 433-459, 2010.

Adkisson, 2003 — H. Adkisson. *Use of Faceted Classification*. http://www.webdesignpractices.com/navigation/facets.html, Oct. 2009.

Agost et al., 2006 — M. Agost, F. Romero, C. Vila, P. Company. *Use of Patterns for Knowledge Management in the Ceramic Tile Design Chain*. 3rd International Conference on Cooperative Design, Visualization, and Engineering, LNCS, vol. 4101, pp. 65-74, 2006.

Agrawal et al., 1996 — R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, I. Verkano. *Fast Discovery of Association Rules*. In: Advances in Knowledge Discovery and Data Mining, eds. U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, AAAI Press, pp. 307-328.

Ahuja et al., 2008 — I. Ahuja, J. Khamba. *Total Productive Maintenance: Literature Review and Directions*. International Journal of Quality & Reliability Management, Emerald, vol. 25, n°7, pp. 709-756, 2008.

Alexander et al., 1977 — C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King, S. Angel. *A Pattern Language*. Oxford University Press, New York, 1977.

Alloui, 2009 — I. Alloui. *Conciliating Property Stability and System Evolution through Software Model Analysis*. GDR Génie de la Programmation Logicielle 2009, pp. 224-231, 2009.

Alsyouf, 2007 — I. Alsyouf. *The Role of Maintenance in Improving Companies' Productivity and Profitability*. International Journal of Production Economics, Elsevier, vol. 105, n°1, pp. 70-78, 2007.

Alsyouf, 2009 — I. Alsyouf. *Maintenance Practices in Swedish Industries: Survey Results*. International Journal of Production Economics, Elsevier, vol. 121, n°1, pp. 212-223, 2009.

Assent et al. 2007 — I. Assent, R. Krieger, E. Müller, T. Seidl. *DUSC: Dimensionality Unbiased Subspace Clustering*. ICDM'2007, pp. 409-414, 2007.

Bacão et al., 2005 — F. Bacão, V. Lobo, M. Painho. *Self-Organizing Maps as Substitutes for K-means Clustering*. International Conference on Computational Science, n°5, pp. 476-483, 2005.

Balaban et al., 2009 E. Balaban, P. Bansal, P. Stoelting, A. Saxena, K. Goebel, S. Curran. *A Diagnostic Approach for Electro-Mechanical Actuators in Aerospace Systems*. IEEE Aerospace Conference, pp. 1-13, 2009.

Barbu et al., 2005 E. Barbu, P. Héroux, E. Trupin, *Classification Non Supervisée Hiérarchique Incrémentale Basée sur le Calcul de Dissimilarités*. Comptes rendus des 12èmes rencontres de la Société Francophone de Classification, pp. 51-54, 2005.

Barros, 1997 S. Barros. *Analyse a priori des Conséquences de la Modification de Systèmes Logiciels: de la Théorie à la Pratique*. Thèse de doctorat, Université Paul Sabatier – Toulouse 3, 1997.

Bennett, 2000 K. Bennett, V. Rajlich. *Software Maintenance and Evolution: A Roadmap*. Proceedings of the Conference on the Future of Software Engineering, pp. 73-87, 2000.

Bergmann et al., 1999a R. Bergmann, I. Vollrath. *Generalized Cases: Representation and Steps towards Efficient Similarity Assessment*. 23rd Annual German Conference on Artificial Intelligence, pp. 195-206, 1999.

Bergmann et al., 1999b R. Bergmann, I. Vollrath, T. Wahlmann. *Generalized Cases and their Application to Electronic Designs*. 7th German Workshop on Case-Based Reasoning, pp. 6-19, 1999.

Berkhin, 2002 P. Berkhin. *Survey of Clustering Data Mining Techniques*. Technical report, Accrue software, San Jose, California, 2002.

Bhandari et al., 1997. I. Bhandari, E. Colet, J. Parker, Z. Pines, R. Pratap, K. Ramanujam. *Advanced Scout: Data Mining and Knowledge Discovery in NBA Data*. Data Mining and Knowledge Discovery, vol. 1, n°1, pp. 121-125, 1997.

Bohner, 2002 S. Bohner. *Software Change Impacts - An Evolving Perspective*. ICSM'02, International Conference on Software Maintenance, pp. 263-272, 2002.

Bossy et al., 1997 M. Bossy, L. Fezoui, S. Piperno. *Comparaison d'une Méthode Stochastique et d'une Méthode Déterministe Appliquées à l'Equation de Burgers*. Rapports de recherche – INRIA, 1997.

Bowles, 2003 J. Bowles. A*n Assessment of RPN Prioritization in a Failure Modes Effects and Criticality Analysis*. Reliability and Maintainability Symposium, pp. 380-386, 2003.

Brakatsoulas et al., 2002 S. Brakatsoulas, D. Pfoser, Y. Theodoridis. *Revisiting R-Tree Construction Principles*. 6th East European Conference, ADBIS 2002, pp. 149-162, 2002.

Brezellec et al., 2001 P. Brezellec, G. Didier. *GIZMO : un Algorithme de Grille Cherchant des Clusters Homogènes*. 3ème Conférence Francophone sur l'Apprentissage Automatique, pp. 101-116, 2001.

Broughton, 2006 V. Broughton. *The Need for a Faceted Classification as the Basis of All Methods of Information Retrieval*. Aslib Proceedings – New Information Perspectives, vol. 58, n°1, pp. 49-72, 2006.

Broughton et al., 2007 V. Broughton, A. Slavic. *Building a Faceted Classification for the Humanities: Principles and Procedures*. Journal of Documentation, vol. 63, n°5, pp. 727-754, 2007.

Brut et al., 2011a M. Brut, D. Codreanu, S. Dumitrescu, A.-M. Manzat, F. Sèdes. *A Distributed Architecture for Flexible Multimedia Management and Retrieval*. International Conference on Database and Expert Systems Applications (DEXA 2011), Abdelkader Hameurlain, Stephen Liddle, Klaus-Dieter Schewe, Xiaofang Zhou (Eds.), Springer-Verlag, pp. 249-263, 2011.

Brut et al., 2011b M. Brut, D. Codreanu, A.-M. Manzat, F. Sèdes. *Adapting Indexation to the Content, Context and Queries Characteristics in Distributed Multimedia*

Systems. International Conference on Signal-Image Technology & Internet-based Systems (SITIS 2011), pp. 118-125, 2011.

Candillier et al., 2004 — L. Candillier, I. Tellier, F. Torre. *Tuareg : Classification Non Supervisée Contextualisée*. 6ème Conférence francophone sur l'Apprentissage automatique, pp. 159-174, 2004.

Candillier et al., 2005 — L. Candillier, I. Tellier, F. Torre, O. Bousquet. *SSC: Statistical Subspace Clustering*. MLDM'2005, pp. 100-109, 2005.

Candillier, 2006 — L. Candillier. *Contextualisation, Visualisation et Evaluation en Apprentissage Non Supervisé*. Thèse de doctorat, Université Charles de Gaulle – Lille 3, 2006.

Candillier et al., 2006 — L. Candillier, I. Tellier, F. Torre, O. Bousquet. *SuSE: Subspace Selection Embedded in an EM Algorithm*. 8ème Conférence Francophone sur l'Apprentissage Automatique, pp. 331-345, 2006.

Celeux et al., 1992 — G. Celeux, G. Govaert. *A Classification EM Algorithm for Clustering and Two Stochastic Versions*. Computational Statistics and Data Analysis, Vol. 14, pp. 315-332, 1992.

Chandola et al., 2009 — V. Chandola, A. Banerjee, V. Kumar. *Anomaly Detection: A Survey*. ACM Computing Surveys, vol. 41, n°3, pp. 1-58, 2009.

Chaudron et al., 2002 — L. Chaudron, N. Maille, M. Boyer. *Rule Induction based on Cubical-FCA*. European Conference on Artificial Intelligence, Workshop FCAKDD Advances in Formal Concept Analysis for Knowledge Discovery in Databases, pp. 7-14, 2002.

Chen et al., 2007 — Z. Chen, X.-S. Liu, X.-D. Zhuang. *A Fast Incremental Clustering Algorithm Based on Grid and Density*. 3rd International Conference on Natural Computation, pp. 207-211, 2007.

Chent et al., 2009 — J.-C. Chent, S.-J. Huang. *An Empirical Analysis of the Impact of Software Development Problem Factors on Software Maintainability*. Journal of Systems and Softwares, vol. 82, n°6, pp. 981-992, 2009.

Chitra, 2003 — T. Chitra. *Life Based Maintenance Policy for Minimum Cost*. Reliability and Maintainability Symposium, pp. 470-474, 2003.

Choi et al., 2004 — K. Choi, M. Shin, S. Bae, C. Kwon, I. Ra. *Similarity Retrieval Based on SOM-Based R*-Tree*. 4th International Conference on Computational Science, vol. 3, pp. 234-241, 2004.

Clark, 2011 — J. Clark. *Exceptions and other Rare and Irregular Events: Two Modes of Learning in Business Intelligence (research in progress)*. Proceedings of the 44th Hawaii International Conference on System Sciences, pp. 1-10, 2011.

Claude, 2009 — G. Claude. *Métamodélisation de documents et recherche de « points communs »*. Actes du XXVIIème Congrès INFORSID, pp. 449-450, 2009.

Claude et al., 2011 — G. Claude, M. Boyer, G. Durand, F. Sèdes. *A Framework to Manage Knowledge from Defect Resolution Process*. IEEE Conference on Commerce and Enterprise Computing, Birgit Hofreiter, Eric Dubois, Kwei-Jay Lin, Thomas Setzer, Claude Godart (Eds.), IEEE, pp. 10-17, 2011.

Claude et al., 2012 — G. Claude, M. Boyer, G. Durand, F. Sèdes. *From Simple Management of Defects to Knowledge Discovery to Optimize Maintenance*. Recent Trends in Information Reuse and Integration, Özyer, Kianmehr, Tan (Eds.), Springer-Verlag, pp. 115-141, 2012.

Cox et al., 2008 — M. Cox, T. Cox. *Multidimensional Scaling*. Handbook of Data Visualization, Springer Handbooks of Computational Statistics, pp. 315-347, 2008.

Curbera et al., 2003 F. Curbera, R. Khalaf, F. Leymann, S. Weerawarana. *Exception Handling in the BPEL4WS Language*. Proceedings of the 2003 International Conference on Business Process Mananagement, LNCS, vol. 2678, pp. 276-290, 2003.

De Lucia et al., 2008 A. De Lucia, F. Fasano, R. Oliveto. *Traceability Management for Impact Analysis*. Frontiers of Software Maintenance, pp. 21-30, 2008.

Deewerster et al., 1990 S. Deerwester, S. Dumais, G. Furnas, T. Landauer, R. Harshman. *Indexing by Latent Semantic Analysis*. Journal of the American Society for Information Science, Wiley, vol. 41, n°6, pp. 391-407,1990.

Denton, 2009 W. Denton. *How to Make a Faceted Classification and Put it on the Web*. http://www.miskatonic.org/library/facet-web-howto.html, Oct. 2009.

Despujols, 2004 A. Despujols. *Approche Fonctionnelle de la Maintenance*. Techniques de l'Ingénieur, n°AG4710, pp. 1-14, 2004.

Dowlatshahi, 2008 S. Dowlatshahi. *The Role of Industrial Maintenance in the Maquiladora Industry: An Empirical Analysis*. International Journal of Production Economics, Elsevier, vol. 114, n°1, pp. 298-307, 2008.

Dubes et al., 1979 R. Dubes, A. Jain. *Validity Studies in Clustering Methodologies*. Pattern Recognition, Elsevier, vol. 11, n°4, pp. 235-254, 1979.

Edelweiss et al., 1998 N. Edelweiss, M. Nicolao. *Workflow Modeling: Exception and Failure Handling Representation*. 18th International Conference of the Chilean Society of Computer Science, pp. 58-67, 1998.

Eder et al., 1996 J. Eder, W. Liebhart. *Workflow Recovery*. 1st International Conference on Cooperative Information Systems, pp. 124-134, 1996.

El Aoufir, 2007 H. El Aoufir, D. Bouami. *Proposition d'un Schéma d'Evolution des Structures Maintenance*. CPI'07, Conception et Production Intégrées, 2007.

Ergazakis et al., 2005 K. Ergazakis, K. Karnezis, K. Metaxiotis, I. Psarras. *Knowledge Management in Enterprises : A Research Agenda*. International Journal of Intelligent Systems in Accounting and Finance Management, Wiley, vol. 13, n°1, pp. 17-26, 2005.

Ester et al., 1996 M. Ester, H.-P. Kriegel, J. Sander, X. Xu. *A Density-Base Algorithm for Discovering Clusters in Large Spatial Database with Noise*. 2nd International Conference on Knowledge Discovery and Data Mining, pp. 226-231, 1996.

Fatudimu et al., 2008 I. Fatudimu, A. Musa, C. Ayo, A. Sofoluwe. *Knowledge Discovery in Online Repositories: A Text Mining Approach*. European Journal of Scientific Research, EuroJournals, vol. 22, n°2, pp. 241-250, 2008.

Fayyad et al., 1996a U. Fayyad, G. Piatetsky-Shapiro, P. Smyth. *From Data Mining to Knowledge Discovery in Databases*. AI Magazine, vol. 17, pp. 37-54, 1996.

Fayyad et al., 1996b U. Fayyad, D. Haussler, P. Storloz. *KDD for Science Data Analysis: Issues and Examples*. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, pp. 50-56, 1996.

Feldman et al., 1998 R. Feldman, M. Fresko, Y. Kinar, Y. Lindell, O. Liphstat, M. Rajman, Y. Schler, O. Zamir. *Text Mining at the Term Level*. In: Zytkow, J.M., Quafafou, M. (eds) PKDD, Second European Symposium, LNCS, vol. 1510, pp. 65-73, Springer, Heidelberg, 1998.

Fisher, 1987a D. Fisher. *Improving Inference Through Conceptual Clustering*. AAAI Press, Proceedings of the 6th National Conference on Artificial Intelligence, vol. 2, pp. 461-465, 1987.

Fisher, 1987b D. Fisher. *Knowledge Acquisition via Incremental Conceptual Clustering*. Kluwer Academic Publishers, Machine Learning, vol. 2, n°2, pp. 139-172, 1987.

Fleurquin et al., 2005 R. Fleurquin, C. Tibermacine, S. Sadou. *Le Contrat d'Evolution d'Architectures : Un Outil pour le Maintien de Propriétés non Fonctionnelles*.

Proceedings of LMO'05 Conference, Langages et Modèles à Objets, l'Objet, vol. 11, pp. 209-222, 2005.

Frakes et al., 1998 W. Frakes, R. Prieto-Diaz, C. Fox. *DARE: Domain Analysis and Reuse Environment*. Annals of Software Engineering, vol. 5, pp. 125-141, 1998.

Gallagher et al., 1991 K. Gallagher, J. Lyle. *Using Program Slicing in Software Maintenance*. IEEE Transactions on Software Engineering, vol. 17, n°8, pp. 751-761, 1991.

Ganter et al., 1999 B. Ganter, R. Wille. *Formal Concept Analysis*. Mathematical foundations. Springer, 1999.

Gatti, 2003 T. Gatti. *TPM: Total Productive Management*. Techniques de l'Ingénieur, n°AG4840, pp. 1-14, 2003.

Gennari et al., 1989 J. Gennari, P. Langley, D. Fisher. *Models of Incremental Concept Formation*. Artificial Intelligence, vol. 40, n°1-3, pp. 11-61, 1989.

Gharbi et al., 2000 A. Gharbi, J.-P. Kenne. *Production and Preventive Maintenance Rates Control for a Manufacturing System: An Experimental Design Approach*. International Journal of Production Economics, Elsevier, vol. 65, n°3, pp. 275-287, 2000.

Gibert et al., 2006 K. Gibert, M. Sanchez-Marre, I. Rodriguez-Roda. *GESCONDA: An Intelligent Data Analysis System for Knowledge Discovery and Management in Environmental Databases*. Environmental Modelling and Software, vol. 21, n°1, pp. 115-120, 2006.

Giess et al., 2007 M. Giess, P. Wild, C. McMahon. *The Use of Faceted Classification in the Organisation of Engineering Design Documents*. International Conference on Engineering Design, 12 pp., 2007.

Giess et al., 2008 M. Giess, P. Wild, C. McMahon. *The Generation of Faceted Classification Schemes for Use in the Organisation of Engineering Design Documents*. International Journal of Information Management, Elsevier, vol. 28, n°5, pp. 379-390, 2008.

Giovannini, 1989 S. Giovannini. *Controlling a Maintenance Effort: An Exercise in Software Project Management*. Conference on Software Maintenance, pp. 264-270, 1989.

Godfrey et al., 2008 M. Godfrey, D. German. *The Past, Present, and Future of Software Evolution*. FoSM 2008, Frontiers of Software Maintenance, pp. 129-138, 2008.

Goh et al., 2009 Y. Goh, M. Giess, C. McMahon, Y. Liu. *From Faceted Classification to Knowledge Discovery of Semi-Structured Text Records*. In: A. Abraham, A.-E. Hassanien, A. de L. de Carvalho, V. Snasel (eds) Foundations of Computational Intelligence Volume 6. Studies in Computational Intelligence, vol. 206, pp. 151-169, 2009.

Guigues et al., 1986 J.-L. Guigues, V. Duquenne. *Familles Minimales d'Implications Informatives Résultant d'un Tableau de Données Binaires*. Math. Sci. Hum., pp. 5-18, 1986.

Gurrutxaga et al., 2009 I. Gurrutxaga, O. Arbelaitz, J. Martín, J. Muguerza, J. Pérez, I. Perona. *SIHC: A Stable Incremental Hierarchical Clustering Algorithm*. International Conference on Enterprise Information Systems, pp. 300-304, 2009.

Guttman, 1984 A. Guttman. *R-Trees: A Dynamic Index Structure for Spatial Searching*. ACM SIGMOD International Conference on Management of Data, pp. 47-57, 1984.

Hall et al., 1996 J. Hall, G. Mani, D. Barr. *Applying Computational Intelligence to the Investment Process*. Proceedings of CIFER'96, Computational Intelligence in Financial Engineering, IEEE Computer Society.

Harjani et al., 1992 D.-R. Harjani, J.-P. Queille. *A Process Model for the Maintenance of Large Space Systems Software*. IEEE Conference on Software Maintenance, pp. 127-136, 1992.

Haziza et al., 1992 — M. Haziza, J. Voidrot, E. Minor, L. Pofelski, S. Blazy. *Software Maintenance: An Analysis of Industrial Needs and Constraints*. Conference on Software Maintenance, pp. 18-26, 1992.

He et al., 2006 — L. He, H. Bai, J. Sun, C. Jin. *A General Incremental Hierarchical Clustering Method*. Computational Methods, pp. 1303-1307, 2006.

IEEE, 2005 — Glossary of software engineering terms. IEEE, 2005.

Jin et al., 2009 — X. Jin, L. Li, J. Ni. *Option Model for Joint Production and Preventive Maintenance System*. International Journal of Production Economics, Elsevier, vol. 119, n°2, pp. 347-353, 2009.

Kao et al., 2010 — Y.-W. Kao, C.-F. Lin, K.-Y. Cheng, S.-M. Yuan, C.-T. Tsai. *A PDCA-Based Critical Exception Management System in Semiconductor Industry*. International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, pp. 417-420, 2010.

Karypis et al., 1999 — G. Karypis, E.-H. Han, V. News. *Chameleon: Hierarchical Clustering Using Dynamic Modeling*. Computer, vol. 32, n°8, pp. 68-75, 1999.

Kohonen, 1982 — T. Kohonen. *Self-Organizing Formation of Topologically Correct Feature* Maps. BiologicalCybernetics , vol. 43, n°1, pp. 59-69, 1982.

Komonen, 2002 — K. Komonen. *A Cost Model of Industrial Maintenance for Profitability Analysis and Benchmarking*. International Journal of Production Economics, Elsevier, vol. 79, n°1, pp. 15-31, 2002.

Kun-Seok et al., 2001 — O. Kun-Seok, F. Yaokai, K. Kaneko, A. Makinouchi, S. Bae. *SOM-Based R*-Tree for Similarity Retrieval*. 7th International Conference on Database Systems for Advanced Classification, pp. 182-189, 2001.

Lambolez, 1994 — P.-Y. Lambolez. *Recherche d'Informations pour la Maintenance Logicielle*. Thèse de doctorat, Université Paul Sabatier – Toulouse 3, 1994.

Lamontagne et al., 2002 — L. Lamontagne, G. Lapalme. *Raisonnement à Base de Cas Textuels : Etat de l'Art et Perspectives*. Revue d'intelligence artificielle ISSN 0992-499X, vol. 16, n°3, pp. 339-366, 2002.

Lampel et al., 2009 — J. Lampel, J. Shamsie, Z. Shapira. *Experiencing the Improbable : Rare Events and Organizational Learning*. Organization Science, vol. 20, n°5, pp. 835-845, 2009.

Leake, 1996 — D. Leake. *Case-Based Reasoning: Experiences, Lessons, and Future Directions*. AAAI Press/MIT Press, Menlo Park, CA, 1996.

Leake et al., 2001 — D. Leake, B. Smyth, D. Wilson, Q. Yang. *Introduction to the Special Issue on Maintaining Case-Based Reasoning Systems*. Computational Intelligence, vol. 17, n°2, pp. 193-195, 2001.

Lehman, 1980 — M. Lehman. *Programs, Life Cycles, and Laws of Software Evolution*. Proceedings of the IEEE, vol. 68, n°9, pp. 1060-1076, 1980.

Lehman et al., 1997 — M. Lehman, J. Ramil, P. Wernick, D. Perry, W. Turski. *Metrics and Laws of Software Evolution - The Nineties View*. 4th International Software Metrics Symposium, pp. 20-32, 1997.

Lerner et al., 2010 — B. Lerner, S. Christov, L. Osterweil, R. Bendraou, U. Kannengiesser, A. Wise. *Exception Handling Pattern for Process Modeling*. IEEE Transactions on Software Engineering, vol. 36, n°2, pp. 162-183, 2010.

Levner et al., 1998 — E. Levner, D. Zuckerman, G. Meirovich. *Total Quality Management of a Production-Maintenance System: A Network Approach*. International Journal of Production Economics, Elsevier, vol. 56-57, n°1, pp. 407-421, 1998.

Li et al., 2009 — Y.-H. Li, Y.-D. Wang, W.-Z. Zhao. *Bogie Failure Mode Analysis for Railway Freight Car based on FMECA*. 8th International Conference on Reliability, Maintainability and Safety, pp. 5-8, 2009.

Lui et al., 2004 — J. Liu, K. Strohmaier, W. Wang. *Revealing True Subspace Clusters in High Dimensions*. ICDM'2004, pp. 463-466, 2004.

Luxburg, 2007 — U. Von Luxburg. *A Tutorial on Spectral Clustering*. In Statistics and Computing, vol. 17, n°4, pp. 395-416, 2007.

Luxburg et al., 2004a — U. Von Luxburg, M. Belkin, O. Bousquet. *Consistency of Spectral Clustering*. Annals of Statistics, vol. 36, n°2, pp. 555-586, 2004.

Luxburg et al., 2004b — U. Von Luxburg, O. Bousquet, M. Belkin. *Limits of Spectral Clustering*. Advances in Neural Information Processing Systems (NIPS), Vol. 17, pp. 857-864, 2004.

Luxhoj et al., 1997 — J. Luxhoj, J. Riis, U. Thorsteinsson. *Trends and Perspectives in Industrial Maintenance Management*. Journal of Manufacturing Systems, vol. 16, n°6, pp. 437-453, 1997.

MacKay, 2003 — D. MacKay. *An Example Inference Task: Clustering*. Information Theory, Inference and Learning Algorithms, pp. 284-292, 2003.

Mahoui et al., 2006 — M. Mahoui, Z. Miled, A. Godse, H. Kulkarni, N. Li. *BioFacets: Faceted Classification for Biological Information*. 18th International Conference on Scientific and Statistical Database Management, pp. 225-234, 2006.

Maindorfer et al., 2008 — C. Maindorfer, T. Ottmann. *Is the Popular R*-Tree Suited for Packet Classification?*. Proceedings of 7th IEEE International Symposium on Network Computing and Applications, pp. 168-176, 2008.

Manago et al., 1996 — M. Manago, M. Auriol. *Mining for OR*. ORMS Today (Special Issue on Data Mining), pp. 28-32.

Mantaras et al., 2005 — R. De Mantaras, D. McSherry, D. Bridge, D. Leake, B. Smyth, S. Craw, B. Faltings, M. Maher, M. Cox, K. Forbus, M. Keane, A. Aamodt, I. Watson. *Retrieval, Reuse, Revision, and Retention in Case-Based Reasoning*. The Knwoledge Engineering Review, vol. 20, pp. 215-240, 2005.

May et al., 2003 — D. May, P. Taylor. Knowledge Management with Patterns. *Communication of the ACM*, vol. 46, n°7, pp. 94-99, 2003.

Mechin, 2007 — B. Mechin. Maintenance : Concepts et Définitions. Techniques de l'Ingénieur, n°MT9030, pp. 1-9, 2007.

Mens et al., 2005 — T. Mens, R. Van Der Straeten. *On the Use of Formal Techniques to Support Model Evolution*. 1ères journées sur l'Ingénierie Dirigée par les Modèles, pp. 115-124, 2005.

Merkl et al., 2003 — D. Merkl, S. He, M. Dittenbach, A. Rauber. *Adaptive Hierarchical Incremental Grid Growing: An Architecture for High-Dimensional Data Visualization*. WSOM03, 4th Workshop on Self-Organizing Maps, pp. 293-298, 2003.

Mili et al., 1997 — H. Mili, E. Ah-Ki, R. Godin, H. Mcheick. *Another Nail to the Coffin of Faceted Controled-Vocabulary Component Classification and Retrieval*. Symposium on Software Reusability, vol. 22, n°3, pp. 89-98, 1997.

Mills, 2004 — J. Mills. *Faceted Classification and Logical Division in Information Retrieval*. Library Trends, vol. 52, n°3, pp. 541-570, 2004.

Mougouie et al., 2003 — B. Mougouie, M. Richter, R. Bergmann. *Diversity-Conscious Retrieval from Generalized Cases: A Branch and Bound Algorithm*. 5th International Conference on Case-Based Reasoning, pp. 319-331, 2003.

Nguyen et al., 2008 — Q. Nguyen, V. Rayward-Smith. *Internal Quality Measures for Clustering in Metric Spaces*. International Journal Business Intelligence and Data Mining, inderscience Publishers, vol. 3, n°1, pp. 4-29, 2008.

Nonaka, 1994 — I. Nonaka. A Dynamic Theory of Organizational Knowledge Creation. Organization Science, vol. 5, n°1, pp. 14-37.

Nonaka et al., 1998 — I. Nonaka, N. Konno. *The Concept of « Ba »: Building a Foundation for Knowledge Creation*. California Management Review, vol. 40, n°3, pp. 40-54.

Parsons et al., 2004 — L. Parsons, E. Haque, H. Liu. *Evaluating Subspace Clustering Algorithms*. Workshop on Clustering High Dimensional Data and its Application, SIAM International Conference on Data Mining, pp. 48-56, 2004.

Persson et al., 2002 — A. Persson, J. Stirna. *Creating an Organisational Memory through Integration of Entreprise Modeling, Patterns, and hHpermedia : The HyperKnowledge Approach*. In Kirikova et al. (eds.), Information Systems Development – Advances in Methodologies, Components and Management, pp. 181-192.

Persson et al., 2003 — A. Persson, J. Stirna, H. Dulle, G. Hatzenbichler, G. Strutz. *Introducing a Pattern Based Knowledge Management Approach : The Verbundplan Case*. Proceedings of the 14th International Workshop on Database and Expert Systems Applications, IEEE Computer Society, 2003.

Persson et al., 2006 — A. Persson, J. Stirna. *How to Transfer a Knowledge Management Approach to an Organization: A Set of Patterns and Anti-Patterns*. 6th International Conference on Practical Aspects of Knowledge Management, LNCS, vol. 4333, pp. 243-252, 2006.

Porter, 1980 — M. Porter. *An Algorithm for Suffix Stripping*. Program: Electronic Library and Information Systems, vol. 14, n°3, pp. 130-137, 1980.

Porter, 2006 — M. Porter. *An Algorithm for Suffix Stripping*. Program: Electronic Library and Information Systems, vol. 40, n°3, pp. 211-218, 2006.

Prieto-Diaz, 1997 — R. Prieto-Diaz. *Implementing Faceted Classification for Software Reuse*. Communications of the ACM, vol. 34, n°5, pp. 88-97, 1991.

Prieto-Diaz, 2003 — R. Prieto-Diaz. *A Faceted Approach to Build Ontologies*. IEEE International Conference on Information Reuse and Integration, pp. 458-465, 2003.

Prieto-Diaz et al., 1987 — R. Prieto-Diaz, P. Freeman. *Classifying Software for Reusability*. IEEE Software, vol. 4, n°1, pp. 6-16, 1987.

Raskutti et al., 1999 — B. Raskutti, C. Leckie. *An Evaluation of Criteria for Measuring the Quality of Clusters*. Proceedings of the 16th Joint Conference on Artificial Intelligence, pp. 905-910, 1999.

Ribert et al., 1999 — A. Ribert, A. Ennaji, Y. Lecourtier. *An Incremental Hierarchical Clustering*. Proceedings of the Vision Interface Conference, pp. 586-591, 1999.

Salton et al., 1983 — G. Salton, M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983.

Salton et al., 1988 — G. Salton, C. Buckley. *Term-Weighting Approaches in Automatic Text Retrieval*. Information Processing and Management, Elsevier, vol. 24, n°5, pp. 513-523, 1988.

Savsar, 2004 — M. Savsar. *Effects of Maintenance Policies on the Productivity of Flexible Manufacturing Cells*. Omega, vol. 34, n°3, pp. 274-282, 2004.

Sèdes, 1998 — F. Sèdes. *Bases documentaires – Hyperbases – Proposition d'un Modèle Générique et Contribution à la Spécification d'un Langage pour l'Intégration et la Manipulation d'Informations Semi-Structurées*. Habilitation à Diriger les Recherches, Université Paul Sabatier – Toulouse 3, 1998.

Senator et al., 1995 T. Senator, H. Goldberg, J. Wooton, M. Cottini, A. Umar Khan, C. Klinger, W. Llamas, M. Marrone, R. Wong. *Financial Crimes Enforcement Network AI System (FAIS) Identifying Potential Money Laundering from Reports of Large Cash Transactions*. AI Magazine, vol. 16, n°4, pp. 21-39, 1995.

Serban et al., 2005 G. Serban, A. Campan. *Core Based Incremental Clustering*. Studia Universitatis Babes-Bolyai, Informatica, vol. L, n°1, pp. 89-96, 2005.

Shah et al., 2007 N. Shah, R. Iqbal, A. James, J. Siddiqi, B. Akhgar. *Ontological Model for Exception Management in Open Multi-Agent Systems*. 11th International Conference on Computer Supported Cooperative Work in Design, pp. 366-371, 2007.

Shaw et al., 2009 G. Shaw, Y. Xu. *Enhancing an Incremental Clustering Algorithm for Web Page Collections*. Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, pp. 81-84, 2009.

Slavic, 1998 A. Slavic. *Faceted Classification: Management and Use*. Axiomathes, vol. 18, n°2, pp. 257-271, 1998.

Sneed et al., 2008 H. Sneed, S. Opferkuch. *Training and Certifying Software Maintainers*. 12th European Conference on Software Maintenance and Reengineering, pp. 113-122, 2008.

Somekh et al., 2007 Y. Somekh, M. Peleg, D. Dori. *Classifying and Modeling Exceptions through Object Process Methodology*. Proceedings of the 2007 International Conference on Systems Engineering and Modeling, pp. 60-70, 2007.

Song, 2009 D.-P. Song. *Production and Preventive Maintenance Control in a Stochastic Manufacturing System*. International Journal of Production Economics, Elsevier, vol. 119, n°1, pp. 101-111, 2009.

Song et al., 2009 Y.-C. Song, H.-D. Meng, S.-L. Wang, M. O'Grady, G. O'Hare. *Dynamic and Incremental Clustering Based on Density Reachable*. 5th International Joint Conference on INC, IMS and IDC, pp. 1307-1310, 2009.

Soulé-Dupuy, 2001 C. Soulé-Dupuy. *Bases d'Informations Textuelles : des Modèles aux Applications*. Habilitation à Diriger des Recherches, Université Paul Sabatier – Toulouse 3, 2001.

Steinbach et al., 2000 M. Steinbach, G. Karypis, V. Kumar. *A Comparison of Document Clustering Techniques*. KDD workshop on Text Mining, 2000.

Stewart, 1993 G. Stewart. *On the Early History of Singular Value Decomposition*. SIAM Review, Society for Industrial and Applied Mathematics, vol. 35, n°4, pp. 551-566, 1993.

Swanson, 1976 B. Swanson. *The Dimensions of Maintenance*. 2nd International Conference on Software Engineering, IEEE Computer Society Press, pp. 492-497, 1976.

Waeyenbergh et al., 2004 G. Waeyenbergh, L. Pintelon. *Maintenance Concept Development: A Case Study*. International Journal of Production Economics, Elsevier, vol. 89, n°3, pp. 395-405, 2004

Waeyenbergh et al., 2007 G. Waeyenbergh, L. Pintelon. *CIBOCOF: A Framework for Industrial Maintenance Concept Development*. International Journal of Production Economics, Elsevier, vol. 121, n°2, pp. 633-640, 2007.

Wang et al., 1997 W. Wang, J. Yang, R. Muntz. *STING: A Statistical Information Grid Approach to Spatial Data Mining*. 23rd International Conference in very Large Data Bases, pp. 186-195, 1997.

Wang et al., 2004 M. Wang, H. Wang, K. Wan, D. Xu. *Knowledge-Based Exception Handling in Securities Transactions*. Proceedings of the 37th Hawaii International Conference on System Sciences, IEEE Computer Society, vol. 3, 10 pp., 2004.

Wei et al., 2007 X. Wei, H. Huang, S. Tian. *A Grid-Based Clustering Algorithm for Network Anomaly Detection*. 1st International Symposium on Data, pp. 104-106, 2007.

Widyantoro et al., 2002 D. Widyantoro, T. Ioerger, J. Yen. *An Incremental Approach to Building a Cluster Hierarchy*. Proceedings of the International Conference on Data Mining, pp. 705-708, 2002.

Wille, 1982 R. Wille. *Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts*. I. Rival (ed.): Ordered Sets, pp. 445-470, 1982.

Wilson, 2009 T. Wilson. *The Strict Faceted Cassification Model*. http://facetmap.com/pub/, Oct. 2009.

Xiang et al., 2008 X. Xiang, S. Tuo, P. Vaidya, J. Lee. *R-Tree: A Hardware Implementation*. International Conference on Computer Design, pp. 3-9, 2008.

Yaxiong et al., 2010 T. Yaxiong, X. Zhen, X. Huibin. *BPM Exception Monitoring based on Process Knowledge*. Conference on Cybernetics and Intelligent Systems, pp. 279-284, 2010.

Young et al., 2101 S. Young, I. Arel, T. Karnowski, D. Rose. *A Fast and Stable Incremental Clustering Algorithm*. 7th International Conference on Information Technology: New Generations, pp. 204-209, 2010.

Zhang et al., 2010 C. Zhang, Y. Huang, G. Zong. *Study on the Application of Knowledge Discovery in Data Bases to the Decision Making of Railway Traffic Safety in China*. International Conference on Management and Service Science, pp. 1-10, 2010.

Zhao, 1998 J. Zhao. *Applying Slicing Technique to Software Architectures*. ICECCS'98, 4th IEEE International Conference on Engineering of Complex Computer Systems, pp. 87-98, 1998.

Annexes

Annexe 1 : application du framework au diagnostic médical

Pour effectuer un diagnostic médical, les données d'un patient que l'on peut trouver dans son dossier sont analysées. Parmi celles-ci, nous pouvons relever les données concernant le patient lui-même, les symptômes et les analyses effectuées, ainsi que le traitement qu'il a suivi pour soigner sa maladie (Figure 60). En regroupant les dossiers patients similaires sur les caractéristiques des patients et leurs symptômes, des traitements déjà utilisées peuvent être proposés.

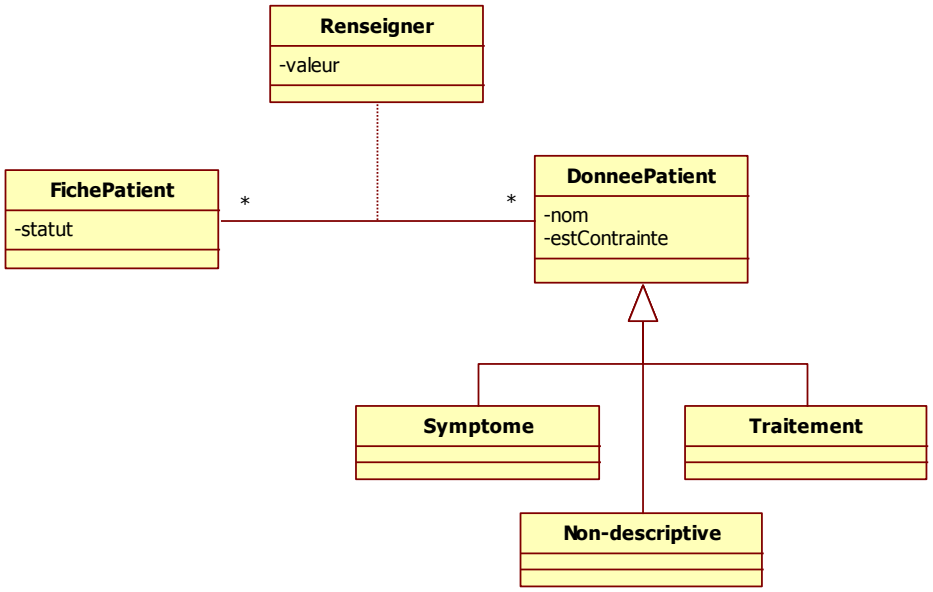


Figure 60 – Modèle de descripteur de fiche adapté au domaine médical

Considérons une patiente d'une cinquantaine d'années qui arrive à l'hôpital. Elle informe avoir de la fièvre, des nausées et souffrir de douleurs abdominales. A partir de ces symptômes communs (descriptibles par tout le monde) et avant d'effectuer un diagnostic plus poussé avec des examens plus approfondis, quatre traitements sont envisageables (Figure 61).

Premier traitement : *chirurgie (appendicectomie).*

La patiente souffre d'une appendicite mise en évidence en effectuant une échographie montrant un épaississement de l'appendice. Cela peut arriver à tout âge mais reste surtout fréquent entre 10 et 20 ans.

Deuxième traitement : *prise d'antalgiques et d'antibiotiques*.

La patiente souffre soit de sigmoïdite (inflammation du sigmoïde (partie basse du colon) d'origine infectieuse), soit de pyélonéphrite (infection rénale). La

sigmoïdite est révélée par une douleur spécifique de la fosse iliaque gauche et par un scanner. C'est une maladie rare avant 30 ans et la moitié des plus de 70 ans en font une. La pyélonéphrite se traduit par une douleur de la fosse lombaire et par la présence de globules blancs et de sang dans les urines. Ce sont principalement les femmes qui en sont atteintes.

Troisième traitement : *prise d'antibiotiques et d'anti-inflammatoires*.

La patiente souffre de salpingite (inflammation des trompes utérines). L'utérus est très douloureux à l'examen. Maladie touchant exclusivement les femmes.

Quatrième traitement : *prise d'antalgiques*.

La patiente souffre d'adénolymphite mésentérique (inflammation des ganglions abdominaux). Cela survient après une infection de la sphère ORL ou des poumons. Seuls les jeunes enfants en sont atteints.

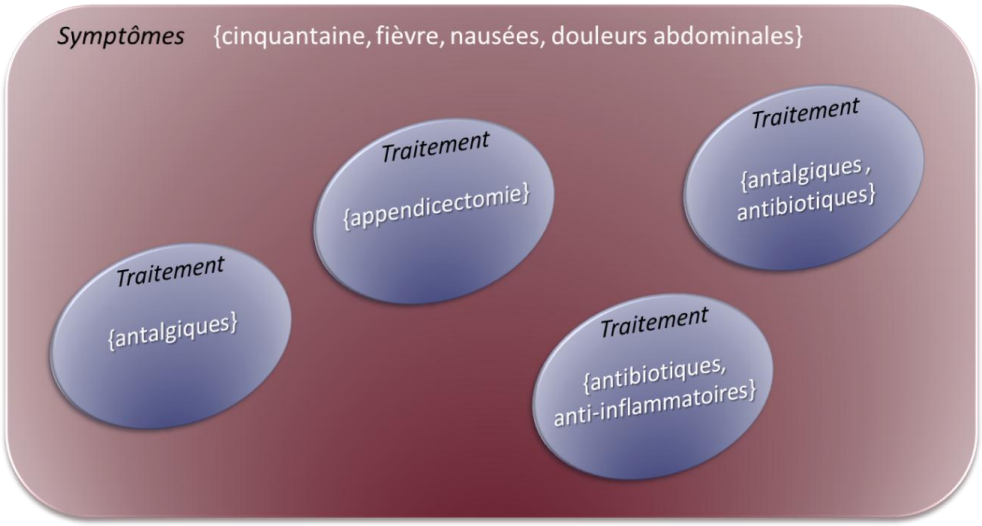


Figure 61 – Illustration de classification d'un cas d'utilisation dans le domaine médical

Différents traitements correspondant à des maladies variées peuvent donc être proposés. En considérant les caractéristiques de la patiente (femme d'une cinquantaine d'années), le quatrième traitement n'est pas adapté et il est assez peu probable que le premier traitement le soit, surtout si la patiente a déjà subi une appendicectomie. Pour déterminer le traitement adéquat, correspondant à la maladie de la patiente, le médecin peut choisir des examens plus approfondis orientés vers la mise en évidence de sigmoïdite, de pyélonéphrite ou de salpingite.