TABLE DES MATIÈRES

Page

Liste des tableaux **xiii**

Table des figures **xv**

1 Introduction **1**

1.1 Contexte et motivation 2

1.1.1 La recherche documentaire dans un corpus de documents numériques 2

1.1.2 Le projet PlaIR 2.0 3

1.2 Problématique et Contributions 5

1.3 Organisation 6

I État de l'art **9**

2 Recherche documentaire **11**

2.1 La recherche d'information 14

2.1.1 Document et requête 15

2.1.2 Pertinence 18

2.1.3 Principales phases du processus RI 20

2.1.4 Les modèles classiques de la RI 25

2.1.5 Limites de la recherche d'information classique 32
2.2 La recherche d'information personnalisée 33
2.2.1 Profil utilisateur 35
2.2.2 Approches de la personnalisation 36
2.3 Systèmes de recommandation 42
2.3.1 Recommandation versus Personnalisation 43
2.3.2 La recommandation à base de contenu 44
2.3.3 La recommandation à base de filtrage collaboratif 46
2.3.4 La recommandation à base de confiance 48
2.3.5 La recommandation hybride 49
2.4 Conclusion 50

3 Systèmes d'agents assistants 53

3.1 Assistance aux utilisateurs 55
3.1.1 Les agents assistants 56
3.1.2 Les caractéristiques d'un agent assistant personnel 56
3.1.3 Les types d'agent assistant personnel 58
3.1.4 Exemples 59
3.2 Assistance à la recherche d'information par système multi-agent 61
3.2.1 L'agentification du processus de la RI 62
3.2.2 L'assistance de l'utilisateur et la personnalisation de la RI par des systèmes multi-agents 63
3.3 Approche stigmergique des systèmes multi-agents 67
3.3.1 Le modèle A&A : Agent et Artefact 69
3.3.2 Les agents dans le A&A méta-modèle 70

3.3.3 Les artefacts dans le méta-modèle A&A 71
3.3.4 Les systèmes multi-agent à base d'A&A 74
3.4 Conclusion 75

II Contributions théoriques 77

4 Algorithmes d'assistance à la recherche documentaire 79
4.1 Composants d'un corpus documentaire 82
4.1.1 Les documents du corpus 82
4.1.2 Une terminologie 83
4.1.3 L'index 84
4.2 Navigation dans le corpus 86
4.2.1 Navigation 86
4.2.2 Recherche 89
4.2.3 Interface 92
4.2.4 Document 92
4.3 Reformulation de requêtes 93
4.3.1 Retour de pertinence et annotation 94
4.3.2 Profil 95
4.3.3 Reformulation du besoin informationnel 97
4.4 Recommandation communautaire 100
4.5 Recommandation à base de filtrage collaboratif 101
4.5.1 Regroupement des utilisateurs similaires 102
4.5.2 Filtrage Collaboratif 106
4.5.3 Exemple 108

4.6 Conclusion 109

5 Plateforme à base d'agents et d'artefacts pour la recherche documentaire 111

5.1 Architecture multi-agent 114

5.2 Couche navigation 117

5.2.1 Artefact Interface 118

5.2.2 Artefact Navigation 121

5.2.3 Artefact Document 122

5.2.4 Artefact Profil 124

5.2.5 Artefact Recherche 127

5.2.6 Artefact Annotation 128

5.3 Couche Décision 131

5.3.1 Agent Interface 131

5.3.2 Agent Reformulation 133

5.3.3 Agent Communautaire 136

5.3.4 Agent Recommandation 137

5.4 Configurations selon profil de l'utilisateur 141

5.4.1 Expert 142

5.4.2 Intermédiaire 143

5.4.3 Novice 144

5.5 Conclusion 146

III Expérimentations et mise en oeuvre **147**

6 Expérimentation et évaluation **149**

6.1 Plateforme JaCa 150

6.1.1 Le langage de programmation agent Jason 151

6.1.2 Plateforme CArtAgO 152

6.2 Expérimentation sur la base de l'IDIT 153

6.2.1 Base documentaire de L'IDIT 153

6.2.2 Première partie de l'expérimentation 154

6.2.3 Deuxième partie de l'expérimentation 156

6.2.4 Analyse des résultats 157

6.3 Conclusion 160

7 Conclusion et perspectives **161**

Bibliographie **167**

LISTE DES TABLEAUX

TABLEAU **Page**

3.1 Comparaison entre les agents et les artefacts dans un modèle A&A 75

4.1 Matrice des notes attribuées par des utilisateurs à des documents 104

4.2 Regroupement des utilisateurs 106

4.3 Remplissage de la matrice 106

4.4 Exemple de matrice 108

4.5 Matrice remplie 108

5.1 Exemple de matrice 140

5.2 Matrice remplie 140

6.1 Nombre moyen de documents consultés 159

TABLE DES FIGURES

FIGURE **Page**

1.1 Tâches du projet Plair 2.0 4

2.1 Le processus de la recherche d'information. 20

2.2 Les modèles classiques de la Recherche d'information [Ricardo & Berthier, 2011]. 26

2.3 Approches de la personnalisation 36

2.4 La recommandation vu comme une boite noire [Jannach *et al.* , 2010] 42

2.5 La recommandation à base de contenu vu comme une boite noire [Jannach *et al.* , 2010] 45

2.6 La recommandation à base de filtrage collaboratif vu comme une boite noire [Jannach *et al.* , 2010] 47

2.7 La recommandation hybride vu comme une boite noire [Jannach *et al.* , 2010] 49

3.1 Le fonctionnement d'un agent assistant personnel 58

3.2 Les éléments de base d'un modèle Agent & Artefact 69

3.3 Une abstraction d'un artefact 73

4.1 Processus d'indexation dans lucene 85

4.2 Processus de recherche dans lucene 91

4.3 L'evolution du besoin informationnel au cours d'une navigation dans le corpus 97

5.1 Architecture Agent-Artefact proposée 116
5.2 Illustration d'un artefact (adoptée de [Ricci *et al.* , 2011]) 118
5.3 Artefact Interface 119
5.4 Artefact Navigation 121
5.5 Artefact Document 122
5.6 Artefact Profil 125
5.7 Artefact Recherche 127
5.8 Artefact Annotation 129
5.9 Dynamique de l'agent Interface 132
5.10 Dynamique de l'agent Reformulation 133
5.11 Dynamique de l'agent Communautaire 137
5.12 Dynamique de l'agent Recommandation 138
5.13 Configuration pour un utilisateur expert 143
5.14 Configuration pour un utilisateur intermédiaire 144
5.15 Configuration pour un utilisateur novice 145

6.1 Expérimentation en phase 1 avec des experts 154
6.2 La moyenne de nombre de documents pertinents des experts ainsi que des utilisateurs novices selon les trois modes pour toutes les questions 158

CHAPITRE 1

INTRODUCTION

Sommaire

1.1 Contexte et motivation **2**

1.1.1 La recherche documentaire dans un corpus de documents numériques 2

1.1.2 Le projet PlaIR 2.0 3

1.2 Problématique et Contributions **5**

1.3 Organisation **6**

1.1 Contexte et motivation

1.1.1 La recherche documentaire dans un corpus de documents numériques

Le développement et la multiplication des systèmes et plateformes informatiques pour accéder à de l'information ne fait que s'accentuer depuis une trentaine d'années. Le grand volume d'information disponible a soulevé de nombreux défis scientifiques dans des domaines tel que la recherche d'information ou la personnalisation des moyens de visualisation, pour ne parler que de l'accès et laisser de côté dans cette thèse le traitement de ces masses de données. Pour accéder à des documents regroupés dans un corpus numérique, il faut être en mesure d'exprimer son besoin en information, souvent sous la forme d'une requête, d'y associer les documents pertinents et de les présenter de la meilleure manière possible aux utilisateurs.

Dans cette thèse, nous nous intéressons plus particulièrement à des corpus documentaires thématiques présentant un haut niveau de technicité dans la discipline concernée. Ces corpus ont pour particularité de s'adresser prioritairement à des experts du domaine, rendant plus difficile la recherche dès l'expression de la requête si l'utilisateur ne dispose pas du vocabulaire précis de la discipline mais aussi lors de l'évaluation de la pertinence des documents proposés.

La recherche documentaire dans de tels corpus numériques s'apparente à un processus de navigation guidé par un besoin d'information d'un utilisateur. Cette navigation nécessite l'usage d'outils classiques de recherche d'information pour sélectionner des documents pertinents en fonction d'une requête, mais ils doivent être complétés par des mécanismes de personnalisation et d'adaptation capable de faire évoluer la représen-

tation du besoin en fonction des spécificités d'un utilisateur, de sa navigation en cours ou du corpus considéré. En effet, l'accès aux documents d'un corpus numérique soulève des problèmes liés à la recherche d'information, à la visualisation des résultats d'une requête et à la navigation entre les documents. Certains de ces problèmes sont similaires à ceux rencontrés dans le domaine de la recherche d'information sur le Web (par exemple le calcul de la pertinence de documents par rapport à une requête) mais d'autres sont spécifiques au fait que nous nous intéressons à un corpus fermé concernant des types de documents, requêtes et utilisateurs restreints que nous pouvons représenter de manière plus fine que dans le cadre général du Web.

1.1.2 Le projet PlaIR 2.0

Le travail présenté dans cette thèse s'inscrit dans le cadre du projet régional PlaIR 2.0 (Plateforme d'Indexation Régionale) et a bénéficié d'une allocation doctorale régionale par le Grand Réseau de Recherche (GRR) normand "Logistique, Mobilité, Numérique".

L'ambition majeure du projet PlaIR 2.0 est de proposer des outils et plateformes pour la constitution de corpus documentaires numériques et leur exploitation au moyen de technologies avancées d'accès à l'information. Dans cet objectif, le projet s'est intéressé aux différentes étapes de ce processus : i) la **dématérialisation** de documents ; (ii) leur **indexation** multi-terminologique et multi-linguiste ; (iii) l'interaction avec les usagers pour un processus **personnalisé** d'accès à l'information ; (iv) l'aide à l'**interprétation** des documents. Plusieurs démonstrateurs sont réalisés dans PlaIR 2.0 pour l'accès à des documents dans les domaines de la santé, du droit ou du patrimoine.

Selon les tâches, des corpus documentaires thématiques de différentes natures ont été considérés. Dans la tâche 1, il s'agit de documents patrimoniaux, notamment issus

des archives du journal de Rouen, avec pour objectif, par une reconnaissance optique de caractères, de construire un corpus numérique à partie de documents papiers. La tâche 2 contribue au développement du portail CISMeF [1] en s'intéressant à l'indexation semi-automatique et à la découverte de documents dans le domaine de la santé. Enfin, les tâches 3 et 4 utilisent une base documentaire juridique mise à disposition par l'Institut du Droit International des Transports (IDIT [2]). Le travail de cette thèse contribue à la tâche 3 pour fournir des mécanismes d'assistance et de personnalisation pour la recherche de documents par des utilisateurs de niveau d'expertise variable dans le domaine du droit. La tâche 4, sur cette même base documentaire, propose une analyse linguistique comme support à l'interprétation sémantique de parties de documents.

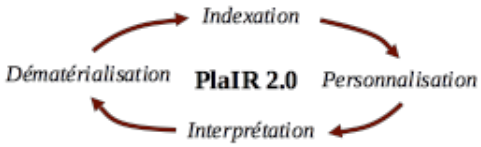

FIGURE 1.1 – Tâches du projet Plair 2.0

Comme l'illustre la figure 1.1, ces quatre tâches sont complémentaires et fournissent des modèles et algorithmes pour une chaîne complète de création et d'enrichissement d'un corpus numérique en commençant par la création d'un corpus numérique, l'indexation des documents créés, des mécanismes de découverte de documents et d'interprétation facilitant leur accès. La fermeture de la boucle signifiant que l'utilisation d'une telle plate-forme peut permettre l'amélioration du corpus par la correction d'erreurs lors de la numérisation (comme le permet la plateforme PIVAJ [3] développée dans le projet).

[1] http://www.chu-rouen.fr/cismef/
[2] http://www.idit.fr
[3] http://litis-pivaj.univ-rouen.fr/pivaj/

1.2 Problématique et Contributions

Au sein du projet PlaIR 2.0, nous nous sommes intéressés aux problématiques soulevées dans la tâche 3 pour proposer des mécanismes d'assistance personnalisée à des utilisateurs avec comme terrain d'application une base documentaire numérique dans le domaine du droit international des transports.

Les phases du processus de la recherche d'information nécessitent des améliorations et surtout l'intégration de l'utilisateur comme facteur principal à prendre en compte dans la recherche de satisfaction de son besoin informationnel. Nous considérons plusieurs approches pour aider un utilisateur dans sa recherche de documents. Une première assistance porte sur la reformulation de requêtes en visant un public d'utilisateurs peu familier avec les termes techniques du droit et en difficulté pour exprimer sous la forme d'une requête leur besoin. La deuxième approche que nous proposons consiste à ne pas considérer l'utilisateur isolément mais à le rapprocher de ceux ayant exprimé des recherches similaires pour retrouver les documents qu'ils avaient jugés pertinents. Enfin, nous incluons des travaux issus du domaine de la recommandation afin de mieux cerner le besoin informationnel de l'utilisateur et l'aider à trouver plus facilement ce qu'il cherche en lui recommandant des ressources documentaires

Nous nous appuyons dans nos travaux sur des modèles existants, développés notamment dans le domaine de la recherche d'information, avec comme originalité de les intégrer dans une plateforme modulaire, adaptative pour activer ou non les services nécessaires à un utilisateur selon son degré d'expertise. Ainsi, nous proposons de traiter cette diversité de facteurs de personnalisation par un système multi-agent, avec plusieurs agents par utilisateur, interagissant avec un environnement partagé représentant les navigations des utilisateurs. Pour cela, nous nous orientons vers une architecture dé-

couplant la gestion de la navigation des décisions liées à la personnalisation permettant une composition dynamique du contenu de ces deux couches.

1.3 Organisation

Ce manuscrit est composé de trois parties distinctes correspondant aux travaux de recherche accomplis. La première partie, bibliographique, est composé de 2 chapitres.

- **Chapitre 2 : Recherche documentaire**. Le chapitre 2 présente les fondements de la recherche d'information (RI) classique : document, requête (ou besoin d'information) et pertinence. Le processus de la RI et les modèles existants y sont décrits. La recherche d'information personnalisée est introduite. Plus particulièrement, nous décrivons la notion de profil utilisateur ainsi que les approches de personnalisation. Finalement, les systèmes de recommandation sont détaillés. Ils représentent une approche intéressante qui complète les facteurs de personnalisation pour cerner le besoin de l'utilisateur et lui proposer un contenu convenable.

- **Chapitre 3 : Systèmes d'agents assistants** : Le chapitre 3 commence par présenter le problème général de l'assistance aux utilisateurs. Il présente les agents assistants personnels en détaillant leur attributs ainsi que leur types. Nous détaillons les approches d'assistance à la recherche d'information par système multiagent. Plus particulièrement, nous décrivons les travaux utilisant les systèmes multi-agents pour la recherche d'information dans un but d'assistance aux utilisateurs en recherche d'information classique et personnalisée. Finalement, ce chapitre présente l'approche stigmergique du développement de systèmes multi-agents qui

sera suivie dans cette thèse. Sa mise en œuvre dans le modèle Agents & Artefacts (A&A) est présentée plus en détail.

La seconde partie est consacrée aux contributions proposées dans cette thèse :

- **Chapitre 4 : Algorithmes d'assistance à la recherche documentaire** : Le chapitre 4 présente un modèle de représentation d'un corpus documentaire composé d'une base de donnée contenant des documents, une terminologie représentant les mots clés pertinents ainsi qu'un index. Ce chapitre aussi présente une formalisation du processus de navigation au sein du corpus documentaire ainsi que les algorithmes d'assistance de l'utilisateur pendant sa navigation dans le corpus documentaire. Ensuite le processus d'indexation ainsi que le processus de recherche sont détaillé. De plus le chapitre détaille les étapes à suivre pour la reformulation de la requête, la recommandation communautaire à base de profils proches ainsi que les algorithmes qui permettent la recommandation collaborative.

- **Chapitre 5 : Plateforme à base d'agents et d'artefacts pour la recherche documentaire** : Le chapitre 5 présente notre modélisation multi-agent de la plateforme de navigation personnalisée décrite selon trois couches : une couche Navigation représentant les artefacts la composant, une couche Décision composée d'agents et une couche Ressources. Ce chapitre présente aussi un ensemble de configuration possibles de notre modèle agent et artefact en prenant en compte le niveau d'expertise des utilisateurs.

Une troisième partie permet de décrire l'architecture de notre système et de l'évaluer au travers de prototypes et de projets opérationnels.

- **Chapitre 6 : Expérimentation et évaluation** Le chapitre 6 présente une implémentation de l'architecture proposée. Une expérimentation avec des utilisateurs experts en droit et des novices permet d'illustrer empiriquement les bénéfices de la plateforme proposée.

Enfin,le chapitre 7 présente les conclusions de la thèse et expose quelques perspectives.

Première partie

État de l’art

CHAPITRE 2

RECHERCHE DOCUMENTAIRE

Sommaire

2.1 La recherche d'information **14**

2.1.1 Document et requête 15

2.1.2 Pertinence 18

2.1.3 Principales phases du processus RI 20

2.1.3.1 Indexation 21

2.1.3.2 L'appariement document-requête 24

2.1.3.3 La reformulation du besoin informationnel 25

2.1.4 Les modèles classiques de la RI 25

2.1.4.1 Le modèle booléen 27

2.1.4.2 Le modèle vectoriel 29

2.1.4.3 Le modèle probabiliste 31

2.1.5 Limites de la recherche d'information classique 32

2.2 La recherche d'information personnalisée **33**

2.2.1 Profil utilisateur 35

2.2.2 Approches de la personnalisation 36

2.2.2.1 Personnalisation individuelle 37

2.2.2.2 Personnalisation collaborative 37

2.2.2.3 Acquisition explicite 37

2.2.2.4 Acquisition implicite 38

2.2.2.5 Acquisition hybride 41

2.2.2.6 Personnalisation à partir d'un contexte courant . . . 41

2.2.2.7 Personnalisation à partir d'un contexte passé 42

2.3 Systèmes de recommandation **42**

2.3.1 Recommandation versus Personnalisation 43

2.3.1.1 La personnalisation 43

2.3.1.2 La recommandation 44

2.3.2 La recommandation à base de contenu 44

2.3.3 La recommandation à base de filtrage collaboratif 46

2.3.4 La recommandation à base de confiance 48

2.3.5 La recommandation hybride 49

2.4 Conclusion **50**

La recherche documentaire dans un corpus numérique s'apparente à un processus de navigation guidé par un besoin d'information d'un utilisateur. Cette navigation nécessite l'usage d'outils classiques de recherche d'information pour sélectionner des documents pertinents en fonction d'une requête, mais ils doivent être complétés par des mécanismes de personnalisation et d'adaptation capable de faire évoluer la représentation du besoin en fonction des spécificités d'un utilisateur, de sa navigation en cours ou du corpus considéré. En effet, l'accès aux documents d'un corpus numérique soulève des problèmes liés à la recherche d'information, à la visualisation des résultats d'une requête et à la navigation entre les documents. Certains de ces problèmes sont similaires à ceux rencontrés dans le domaine de la recherche d'information sur le Web (par exemple le calcul de la pertinence de documents par rapport à une requête) mais d'autres sont spécifiques au fait que nous nous intéressons à un corpus fermé concernant des types de documents, requêtes et utilisateurs restreints que nous pouvons représenter de manière plus fine que dans le cadre général du Web. Dans ce cadre, il est pertinent de s'intéresser d'abord au domaine de la recherche d'information en général avant d'étudier ceux de la recherche d'information personnalisée et des systèmes de recommandation dont l'approche cible plus précisément des cas d'utilisation spécifiques.

Ce chapitre est organisé comme suit : la section 2.1 présente les fondements de la recherche d'information (RI) classique : document, requête (ou besoin d'information) et pertinence. Le processus de la RI et les modèles existants y sont décrits. La section 2.2 introduit la recherche d'information personnalisée. Plus particulièrement, nous décrivons la notion de profil utilisateur ainsi que les approches de personnalisation. Finalement, les systèmes de recommandation, détaillés dans la section 2.3, représentent une approche de intéressante qui complète les facteurs de personnalisation pour cerner le besoin de

l'utilisateur et lui proposer un contenu convenable.

2.1 La recherche d'information

Le domaine de la recherche d'information, apparue dans les années soixante, étudie la manière de retrouver des informations dans un ensemble de documents appelé corpus documentaire. La recherche d'information peut être définie comme la science de la recherche d'informations dans les bases de données relationnelles, des documents, des textes, des fichiers multimédias, et le Web [Lauren & Becker, 1975].

Le principe de la recherche d'information classique est la correspondance entre un ensemble de mots qui représentent un document et un ensemble de mots qui représentent la requête de utilisateur pour avoir les documents pertinents qui répondent aux besoins de l'utilisateur.

Définition 1 (Recherche d'information [Van Rijsbergen, 1986])**.**

"The user expresses his information need in the form of a request for information. Information retrieval is concerned with retrieving those documents that are likely to be relevant to his information need as expressed by his request."

Un système de Recherche d'Information (SRI) intègre différents acteurs : le besoin informationnel d'un utilisateur ou requête, le corpus documentaire, ainsi qu'un ensemble de modèles et de processus qui permettent de sélectionner des résultats pertinentes répondant au besoin en information de l'utilisateur représenté par une requête. Selon Salton [Salton, 1989] "Un système de recherche d'information (RI) est un système qui permet de retrouver les documents pertinents à une requête d'utilisateur, à partir d'une base de documents volumineuse."

Ainsi, dans un contexte documentaire, l'objectif de la recherche d'information est de fournir les opérations et les procédures pour sélectionner les documents susceptibles de répondre à un besoin en information d'un utilisateur exprimé par une requête.

Trois notions clés représentent les concepts de base de la recherche d'information : document, requête et pertinence.

2.1.1 Document et requête

Les notions de documents et de requêtes sont définies comme suit :

Définition 2 (Document [Salton, 1989])**.**

Un document peut être un texte, un morceau de texte, une page Web, une image, une bande vidéo, etc. On appelle document toute unité qui peut constituer une réponse à une requête d'utilisateur.

Définition 3 (Requête [Salton, 1989])**.**

Une requête exprime le besoin d'information d'un utilisateur. Elle est en général de la forme suivante : "Trouvez les documents qui ..."

Généralement, une requête est construite par un utilisateur sous la forme d'un ensemble de mots clés (en langage naturel [Flurh & Debili, 1985; Salton, 1971], langage graphique [Bourne *et al.* , 1976], langage booléen [Lelu & François, 1992], etc.) en fonction de son besoin informationnel. Elle peut être exprimée par différentes manières tels que l'utilisation des mots non-contrôlés, des phrases courtes, sous forme de textes ou de documents en langage naturel.

Une requête est une formulation possible du besoin informationnel d'un utilisateur., des hypothèses théoriques intéressantes sur la façon dont un besoin d'information peut se développer dans l'esprit d'un utilisateur sont faits par Taylor [Taylor, 1968]. Il pose comme principe qu'un état psychologique particulier de l'esprit de l'utilisateur peut conduire à exprimer une demande d'information. Basé sur des entretiens avec les bibliothécaires universitaires, Taylor a suggéré quatre niveaux de formation de question (demande d'information). Trois sont intrinsèques et le quatrième constitue la demande d'information au système de RI :

- **Le besoin viscéral** : C'est le besoin réel inexprimé de l'information. Il est le besoin conscient ou même inconscient en information qui n'existe pas dans les souvenirs d'expérience du demandeur. Ce besoin est probablement inexprimable en termes linguistiques. Il peut en effet, changer en terme de forme, de qualité, d'aspect concret, ainsi que les critères dont l'information est ajoutée, car elle est influencée par analogie, ou que son importance croît avec l'enquête.

- **Le besoin conscient** : C'est un besoin dont la description dans le cerveau est consciente.

  C'est une description mentale consciente d'une zone mal définie de l'indécision. Ce sera probablement une déclaration ambiguë et décousue. Le demandeur peut, à ce stade, parler à quelqu'un d'autre pour aiguiser son attention.

- **Le besoin formalisé** : c'est l'expression formelle du besoin. A ce niveau, un demandeur peut former une expression qualifiée et rationnelle de sa question. Dans ce cas, il décrit son domaine de doute en termes concrets. En plus, il peut ou

pas avoir des réflexions sur le contexte ou sur les contraintes du système duquel il souhaite obtenir des informations.

- **Le besoin compromis** : C'est la demande présentée au système d'information, (la requête) Dans ce cas, la demande est reformulée en prévision de ce que les fichiers peuvent fournir. à moins que le demandeur connait bien le domaine d'information, il suffit alors de poser sa demande avec des termes positifs et bien définis.

En se basant sur ces hypothèses, Ingwersen définit trois types fondamentaux de besoins d'information[Ingwersen, 1992] :

- **Besoin vérificatif** (dit aussi problème de localisation d'information) : Dans ce cas, l'utilisateur veut vérifier ou localiser des éléments, par exemple certains articles spécifiques ou une liste d'adresses de clients. Un autre exemple peut être des données bibliographiques caractéristiques, par exemple une source, des pages, le nom d'un auteur. En terme de localisation, l'utilisateur cherche par exemple à localiser une ville par un code postal qu'il connait déjà. Tous ces exemples se joignent dans le fait qu'il sont connus de l'utilisateur. Un besoin vérificatif est stable. En effet il ne change pas pendant la recherche.

- **Besoin thématique connu** : Dans ce cas, l'utilisateur veut clarifier, examiner ou poursuivre ses recherches sur un sujet et domaine connus. Il veut en effet, améliorer ses informations sur ce qu'il connait déjà. Un besoin thématique connu est généralement évolutif : le besoin s'affine pendant la recherche.

- **Besoin thématique inconnu** (dit aussi problème d'information mal-défini) : Avec un besoin thématique inconnu, l'utilisateur cherche à explorer de nouveaux

concepts ou de relations conceptuelles en dehors des domaines de ses connaissances. Ce besoin est variable et mal défini du fait que l'utilisateur n'a pas suffisamment de connaissance pour pouvoir cerner son besoin d'information.

2.1.2 Pertinence

La pertinence est la notion centrale dans la RI puisque toutes les évaluations s'articulent autour de cette notion et de nombreuses études portent sur la notion de pertinence.

Définition 4 (Pertinence [Salton & McGill, 1970])**.**

La pertinence est :

- la correspondance entre un document et une requête, une mesure d'informativité du document à la requête ;
- un degré de relation (chevauchement, relativité, ...) entre le document et la requête ;
- un degré de la surprise qu'apporte un document, qui a un rapport avec le besoin de l'utilisateur ;
- une mesure d'utilité du document pour l'utilisateur.

Tout d'abord, il est convenable de faire la différence entre les notions de *pertinence* et *relevance*. D'après Simonnot [Simonnot, 2008], "La relevance concerne l'adéquation d'un document ou d'un objet informationnel à un besoin d'information ou à une demande d'information en général". Par contre la pertinence est considérée comme l'adéquation d'un document ou d'un objet informationnel à un besoin en information ou à une demande

d'information en général. De ce fait, pour qu'un document pertinent et relevant pour cet individu, il doit lui être compréhensible et lié à ses connaissances sur le sujet.

De nombreuses études portent sur la notion de pertinence nous citons par exemple : [Borlund, 2003; Cosijn & Ingwersen, 2000; Mizzaro, 1997; Salton & McGill, 1970; Saracevic, 2007; Simonnot, 2008], etc.

La notion la plus étudiée est la *nature* de la pertinence :

- **Pertinence sujet - pertinence thématique (Topicality)** : La pertinence sujet ou thématique concerne les documents jugés pertinents lorsqu'ils traitent le même sujet de la requête. Un document est pertinent s'il contient les mots clés de la requête tout en respectant le même thème.

- **Pertinence dynamique** : Une pertinence statique ou ponctuelle est le retour des documents pertinent pour une requête ponctuelle. Après l'introduction du principe de la rétroaction de pertinence (relevance feedback) par Rocchio en 1966 [Rocchio, 1966, 1971], une fonction de reformulation de la requête vient affiner le besoin informationnel de l'utilisateur en se basant sur les documents jugés pertinents ou non pertinents. Nous parlons alors d'une session de recherche et non simplement d'une requête ponctuelle. Au cours d'une session de recherche, les documents pertinents aident l'utilisateur à mieux cerner son besoin d'information. En plus, en utilisant la reformulation et le retour de pertinence, l'utilisateur peut poser des questions de manière différente. Finalement les documents pertinents représentent la source d'information qui répond au besoin de l'utilisateur.

- **Pertinence situationnelle** : La pertinence situationnelle s'intéresse à la manière dont les utilisateurs utilisent l'information d'une part et aux effets réels qu'a l'information sur le changement de vision de l'utilisateur sur un problème d'autre

part. La pertinence situationnelle implique l'utilité [Xu & Chen, 2006]. C'est la valeur pratique et utilitaire que possède un document pour une tâche à effectuer ou un problème à résoudre. Face à une même requête, formée par les mêmes mots-clés, les documents retournés ne sont pas forcément pertinents pour tous les utilisateurs qui ont des besoins d'information différents en posant cette requête. Un ensemble de document pour cette même requête peut ne pas être pertinent pour un seul utilisateur dans des situations différentes.

2.1.3 Principales phases du processus RI

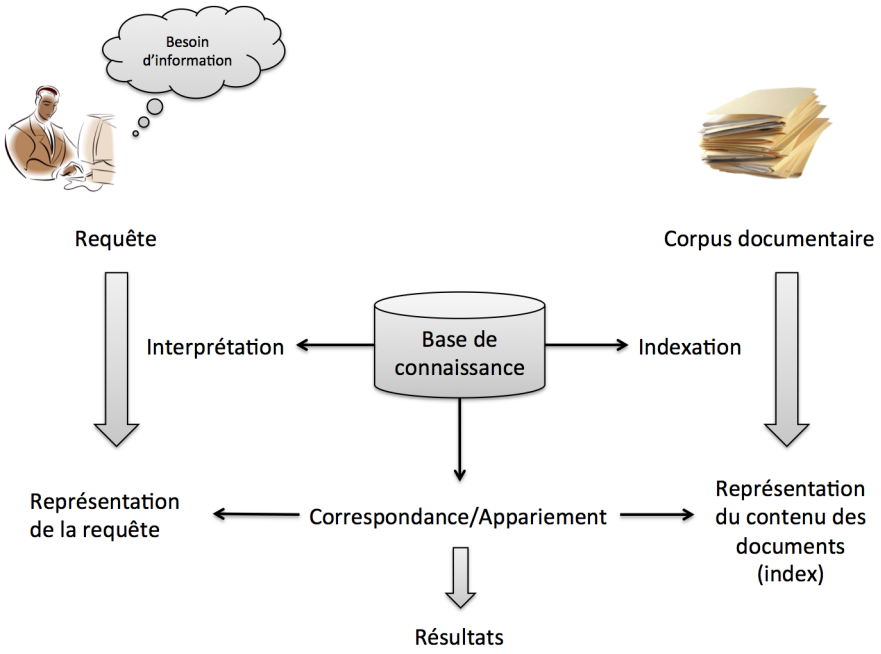

FIGURE 2.1 – Le processus de la recherche d'information.

Les phases du processus RI varient d'une définition à une autre. Cependant, nous trouvons des étapes fondamentales dans ces définitions : L'indexation, la correspondance entre la représentation de la requête et la représentation du document, appelée aussi

l'appariement, et finalement l'accès aux documents. La figure 2.1 illustre ces différents étapes. En effet, l'utilisateur formule son besoin en information sous la forme d'une requête. L'idée de base de la recherche d'information fait référence à la comparaison entre la requête de l'utilisateur et un ensemble de documents qui forment un corpus documentaire. Cette comparaison ou correspondance ne peut pas se faire directement : Le corpus documentaire ainsi que la requête sont transformés respectivement en une représentation du contenu des documents en utilisant un processus appelé indexation, et une représentation de la requête en utilisant un processus d'interprétation. Les deux présentations sont basées sur une base de connaissance ou un modèle de représentation. Finalement vient la correspondance (ou l'appariement) qui compare les deux présentations pour afficher un ensemble de résultats.

Différemment d'un processus de recherche d'information ponctuel, une navigation dans un corpus documentaire se réalise pendant une session de recherche. Lors de cette session, un processus de reformulation s'ajoute au processus de RI en se basant sur les documents pertinent ou pas pertinent retournés à l'utilisateur. De plus, un utilisateur peut naviguer entre les documents résultant d'une requête.

2.1.3.1 Indexation

Définition 5 (Indexation [Salton & McGill, 1986])**.**

L'indexation recouvre un ensemble de techniques visant à transformer les documents (ou requêtes) en substituts ou descripteurs capables de représenter leur contenu.

Le processus d'indexation vise à représenter le contenu sémantique des documents (ou requêtes) par un ensemble de fonctionnalités d'indexation. Dans le cas le plus courant,

ces unités d'indexation sont des mots pour les documents, les notes de musique pour la musique, les valeurs des couleurs pour les images, etc.[Savoy & Gaussier, 2010]

L'indexation fait référence à la tâche qui construit les descripteurs ou la présentation de chaque document d'une collection de document pour les recherches ultérieures. Ces descripteurs seront par la suite comparés avec la représentation de la requête dans le but de retourner les documents susceptibles de répondre au besoin de l'utilisateur. L'indexation sert alors à extraire les termes les plus représentatifs pour un document ou une requête. Pour cela, il est important de bien indexer un document puisque le résultat de la recherche se base en réalité sur les termes indexant le document.

Il existe différents modes d'indexation en recherche d'information : l'indexation manuelle, l'indexation automatique ou l'indexation semi-automatique. Nous détaillons dans ce qui suit chacun de ces modes.

- **L'indexation manuelle** : Dans ce mode d'indexation, l'indexation est réalisée d'une façon manuelle. En effet, chaque document est analysé par un spécialiste du domaine généralement un documentaliste. Pour chaque document, il synthétise l'idée ou les idées du contenu de ce document. Il choisit les termes les plus pertinent pour représenter le contenu d'un document. Ce mode est très couteux en temps. En plus, cette indexation dépend de l'état de connaissance de documentaliste sur le sujet du document. Par ailleurs, deux personnes peuvent indexer un même document différemment. C'est pour cette raison que les indexeurs se basent sur un thesaurus qui représente un ensemble de termes avec un nombre de relations bien défini entre les termes [Foskett & Kent, 1980]. C'est une façon de limiter les termes utilisables pour l'indexation manuelle.

- **L'indexation automatique** : L'indexation automatique vient résoudre le pro-

blème du coût de l'indexation manuelle. En effet, l'indexation automatique est réalisée par un processus entièrement automatisé. Il n'y a donc pas d'intervention d'expert pendant le processus. Une indexation automatique est réalisée sur deux étapes :

- Rechercher les termes descriptifs qui caractérisent le contenu d'un document ;
- Evaluer le pouvoir de caractérisation des termes sélectionnés dans l'étape précédente.

Généralement en recherche d'information, l'indexation automatique consiste à :

- sélectionner des termes simples (des groupes de mots dans d'autres approches) ;
- supprimer les mots vides (en se basant sur un anti-dictionnaire qui contient les mots outils de la langue) ;
- radicaliser ou normaliser les termes restants (en supprimant les variantes morphologiques. Une possibilité est utiliser de l'algorithme de Porter, [Porter, 1980]) ;
- pondérer les radicaux obtenus en utilisant leur fréquence relative dans le document courant ainsi que leur fréquence absolue dans la collection(*tf.idf* pour *term frequency-inverse.document frequency* est la pondération la plus connue[Robertson & Jones, 1976]) ;
- présenter le résultat de l'indexation pour chaque document généralement, en un ensemble de couple (terme d'indexation, poids).

- **L'indexation semi-automatique (mixte)** : C'est la combinaison de l'indexation manuelle et l'indexation automatique :

    - Premièrement, l'indexation automatique est appliquée : Elle permet d'extraire les termes de chaque document d'une manière automatique ;
    - Ensuite, l'utilisation de l'indexation manuelle se résume dans le fait que c'est le spécialiste du domaine, l'expert ou le documentaliste qui a le choix final pour choisir les termes significatifs et représentatifs ainsi qu'établir les relations entre les termes.

2.1.3.2 L'appariement document-requête

Le processus d'appariement document-requête (matching) permet d'évaluer la pertinence d'un document par rapport à une requête. Plus précisément, lorsque l'utilisateur introduit sa requête le système calcule un score de pertinence entre cette requête et les documents. Le score de pertinence est calculé en utilisant une fonction de correspondance dans le but de calculer un score de similarité $RSV(Q,D)$ (pour Relevance status Value) entre une requête indexée notée Q et un document indexé noté D. Cette similarité est exprimée en fonction du modèle d'indexation et de pondération des termes. Nous pouvons distinguer deux méthodes d'appariement :

- **Appariement exact** : Dans ce cas, la requête spécifie d'une manière exacte et précise les critères recherchés. Le résultat de la correspondance est une liste non ordonnée de documents qui respectent exactement la requête [Salton, 1971].

- **Appariement approché** : Contrairement à l'appariement exact, l'appariement rapproché fournit une liste ordonnée de documents qui sont sélectionnés en fonction

d'un degré de pertinence de la requête par rapport aux documents. La requête décrit d'une sorte les critères recherchés dans un document [Robertson & Jones, 1976].

2.1.3.3 La reformulation du besoin informationnel

La reformulation du besoin informationnel fait référence à l'étape où l'utilisateur modifie ou redéfinit son besoin informationnel pendant la session de recherche. Cette reformulation peut être réalisée d'une manière automatique où le système exploite les termes ayant le poids le plus important dans les documents pertinents ou les documents visités, ou alors d'une manière manuelle où l'utilisateur redéfinit son besoin en introduisant une nouvelle requête.

2.1.4 Les modèles classiques de la RI

Un modèle de RI est l'abstraction du processus de la recherche d'information. Il sert à formaliser le processus de la RI dans le but de modéliser la mesure de pertinence, l'utilisateur et son besoin informationnel. Il existe plusieurs modèles dans la littérature. Nous étudions ici les trois modèles classiques : Le modèle booléen, le modèle vectoriel et le modèle probabiliste. Ces modèles considèrent plusieurs éléments : Les besoins, les utilisateurs, les termes, les documents, les jugements de pertinence, etc. Baeza-Yates et al. [Baeza-Yates *et al.* , 1999] définissent un modèle de RI par un quadruplet $(D,Q,F,R(q,d))$ où :

- D est l'ensemble de documents
- Q est l'ensemble de requêtes

- F est le schéma du modèle théorique de représentation des documents et requêtes
- R(q,d) est la fonction de pertinence du document $d \in D$ à la requête $q \in Q$.

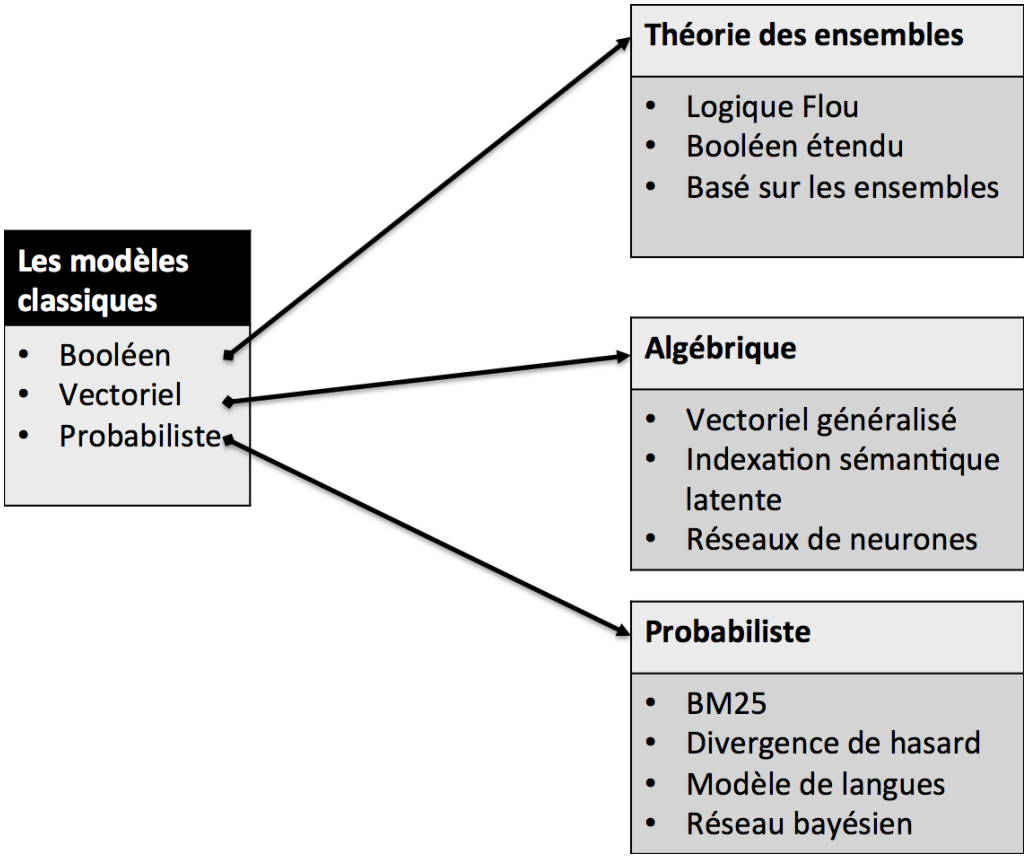

FIGURE 2.2 – Les modèles classiques de la Recherche d'information [Ricardo & Berthier, 2011].

La figure 2.2 illustre un extrait de la taxonomie des modèles de RI proposée dans [Ricardo & Berthier, 2011]. Le premier modèle classique est basé sur la théorie des ensemble. C'est le modèle booléen. Ils existent des extensions et d'autres modèles basés sur la théorie des ensembles : Le modèle flou (basé sur la logique floue), le modèle booléen étendu et le modèle basé sur les ensembles.

Le deuxième modèle classique est le modèle vectoriel. C'est un modèle algébrique. Il est étendu par le modèle vectoriel généralisé, le modèle à base des réseaux de neurones

et le modèle basé sur l'indexation sémantique latente. Le troisième modèle est le modèle probabiliste étendu par le modèle BM25, le modèle de langues, le modèle à base des réseaux bayésiens. Nous détaillons dans la suite les trois principaux modèles de RI.

2.1.4.1 Le modèle booléen

Le modèle booléen [Salton, 1969] est un modèle basé sur la théorie des ensembles. Dans ce modèle, la requête de l'utilisateur est exprimée en langage booléen. Les documents ainsi que les requêtes sont modélisés pars des ensembles de mots clés. Pour trouver les documents pertinents, l'idée est de créer pour chaque terme de la requête les documents indexés par ce terme. Par la suite, nous utilisons les opérateurs logiques (intersection, union ou différence) sur les ensembles selon les termes qui forment la requête.

Un exemple de représentation d'un document est comme suit :

$$d = t_1 \wedge t_2 \wedge t_3 ... \wedge t_n. \tag{2.1}$$

Un exemple de représentation d'une requête est comme suit :

$$q = (t_1 \wedge t_2) \vee (t_3 \wedge t_4). \tag{2.2}$$

La fonction de correspondance se base sur l'hypothèse de présence ou l'absence des termes de la requête dans le document. La fonction est une fonction binaire décrite comme suit :

$$Appariement(q,d) = RSV(q,d) = 1 \; ou \; 0. \tag{2.3}$$

Par exemple, soit la requête "recherche documentaire". L'utilisateur est intéressé par les résultats qui contiennent "recherche" et "documentaire". En langage booléen la

requête s'écrit de la manière suivante :

$$Q = Recherche \wedge documentaire. \tag{2.4}$$

La recherche d'information pour cette requête suit les étapes suivantes :

- **Etape 1** : chercher les documents dont le terme "recherche" fait partie de son index et mettre les résultats dans un ensemble $A1$.
- **Etape 2** : chercher les documents dont le terme "documentaire" fait partie de son index et mettre les résultats dans un ensemble $A2$.
- **Etape 3** : enregistrer l'intersection de $A1$ et $A2$ dans un nouveau ensemble $A3$.
- **Etape 4** : trouver les documents qui correspondent aux éléments de l'ensemble $A3$

Supposons que l'utilisateur souhaite avoir des documents qui contiennent soit le terme "recherche" OU le terme "documentaire". La requête devient :

$$Q = Recherche \vee documentaire. \tag{2.5}$$

Dans ce cas, une opération d'union vient remplacer l'opération d'intersection dans l'étape 3 entre les ensembles $A1$ et $A2$.

Pour l'opérateur de négation NON la requête peut être :

$$Q = Recherche \wedge \neg documentaire. \tag{2.6}$$

Dans ce cas, l'étape 1 et 2 restent inchangés. Ensuite l'étape 3 devient :

- **Etape 3** : supprimer de l'ensemble $A1$ les éléments qui sont aussi présents dans $A2$ et mettre les résultats dans un ensemble $A3$.

- **Etape 4** : trouver les documents qui correspondent aux éléments de l'ensemble $A3$

Le modèle booléen utilise l'appariement exact, les document pertinents correspondent exactement à la requête et sont retournés sans ordre de pertinence. Le modèle booléen étendu [Salton & McGill, 1986] vient rajouter un ordre de pertinence dans les résultats.

Un modèle booléen présente des inconvénients : Les documents sont sélectionnés en se basant sur une décision binaire. En plus, la formulation de la requête en langage booléen est difficile et n'est pas évidente pour tous les utilisateurs. L'appariement exact élimine les documents qui sont susceptibles d'être pertinents.

2.1.4.2 Le modèle vectoriel

Le modèle vectoriel a été développé dans le cadre du projet SMART (Salton's Magical Automatic Retriever of Text) par Salton [Salton, 1971; Salton & McGill, 1986].

Ce modèle utilise les bases mathématiques des espaces vectoriels comme par exemple le calcul de distances entre les vecteurs. En effet, le modèle vectoriel crée un espace où les documents et les requêtes sont représentés par des vecteurs. Soit t le nombre de termes d'indexation du corpus documentaire, l'espace est dit de dimensions t. Chaque requête est représenté par un vecteur de termes recherchés. De la même façon, chaque document est représenté par un vecteur de t nombre de 1 si le terme est présent dans l'index du document ou 0 s'il n'est pas présent.
Une extension du modèle vectoriel apporte une pondération sémantique des termes d'indexation. Cette pondération donne une signification aux termes en termes de représentativité du contenu de document.

Un document est représenté comme suit :

$$\overrightarrow{D_i} = (d_{i1}, d_{i2}, d_{i3}, ..., d_{it}) \tag{2.7}$$

Une requête est représentée comme suit :

$$\overrightarrow{Q} = (q_1, q_2, q_3, ..., q_t) \tag{2.8}$$

Le processus de recherche d'information avec une modélisation vectorielle consiste à chercher les vecteurs représentant les documents les plus proches au vecteur requête. Pour cela, il existe des mesures de similarité et de distances entre les vecteurs. Le produit scalaire ainsi que la mesure cosinus représentent les deux mesures les plus fréquemment utilisées.

- **Le produit scalaire** : C'est une fonction mathématique dans l'espace vectoriel qui utilise les cordonnées des vecteurs. Dans ce cadre, les cordonnées sont les poids sémantiques des termes. La mesure de similarité Sim entre $\overrightarrow{D_i}$ et $\overrightarrow{Q}$ en utilisant le produit scalaire est comme suit :

$$Sim(Q, D_i) = \sum_{t}^{j=1} q_j . d_{ij} \tag{2.9}$$

Où d_{ij} représente le poids du terme t_j dans le document i et q_j représente le poids du terme t_j dans la requête.

Soit N le nombre des documents dans le corpus documentaire.

$$d_{ij} = tf_{ij} . \log\left(\frac{N}{f_j}\right) \tag{2.10}$$

Où f_j représente le nombre de documents contenant le terme t_j et tf_{ij} représente la fréquence d'apparition du terme t_j dans le document i et $\log\left(\frac{N}{f_j}\right)$ représente la mesure de la fréquence inverse du terme t_j

$$\begin{cases} q_j = \log\left(\frac{N}{f_j}\right) \text{si le terme } t_j \text{apparaît dans la requête} \\ q_j \text{ Sinon} \end{cases} \tag{2.11}$$

- **La mesure cosinus** : C'est la mesure de similarité la plus utilisée pour le modèle vectoriel. Elle consiste à mesurer l'angle entre le vecteur de la requête et les vecteurs des documents par le produit scalaire entre des vecteurs normalisés [Salton, 1971] :

$$Sim(Q,D_i) = \frac{\sum_{j=1}^{t} q_j d_{ij}}{\left(\sum_{j=1}^{t} q_j^2\right)^{\frac{1}{2}} \left(\sum_{j=1}^{t} d_{ij}^2\right)^{\frac{1}{2}}} \tag{2.12}$$

La pertinence est alors calculée entre un document et une requête en se basant sur cette mesure de similarité :

$$\begin{cases} Sim(Q,D_i) = 1 \Leftrightarrow \overrightarrow{Q} = \overrightarrow{D_i} \\ Sim(Q,D_i) = Sim(D_i,Q) \end{cases} \tag{2.13}$$

Un degré de similarité est maximal s'il est égal à 1 c'est-à-dire que la requête et le document se correspondent parfaitement. Par contre, le degré de similarité est égal à 0 s'il n'y a pas de correspondances entre la requête et le document.

2.1.4.3 Le modèle probabiliste

Roberston [Robertson & Jones, 1976; Robertson *et al.* , 1995] a proposé un modèle probabiliste du processus de RI. La pertinence entre la requête et les documents est représentée par un degré de pertinence. Lorsque les représentations de la requête et des

documents dépassent un seuil de similarité, la probabilité de pertinence est suffisante pour renvoyer ce document comme résultat à cette requête.

Soient $D = (t_1, t_2, ..., t_t)$ avec $t_i = 1$ s'il indexe le document D et $t_i = 0$ sinon. La probabilité de pertinence d'un document D est calculée comme suit [Robertson & Jones, 1976] :

$$P(Pert/D) = \frac{P(D/Pert).P(Pert)}{P(D)} \tag{2.14}$$

$$P(NonPert/D) = \frac{P(D/NonPert).P(NonPert)}{P(D)} \tag{2.15}$$

avec

$$P(D) = P(D/Pert).P(Pert) + P(D/NonPert).P(NonPert) \tag{2.16}$$

Où $P(Pert/D)$ est la probabilité de pertinence d'un document sachant sa description ;
$P(D/Pert)$ est la probabilité de trouver D sachant qu'il est pertinent ;
$P(D/NonPert)$ est la probabilité de trouver D sachant qu'il est non pertinent ;
$P(Pert)$ est la probabilité à priori pour qu'un document soit pertinent ;
$P(NonPert)$ est la probabilité à priori pour qu'un document soit non pertinent.

2.1.5 Limites de la recherche d'information classique

La limite majeure de la RI classique est qu'elle est basée sur une approche généraliste qui évalue invariablement les requêtes des utilisateurs et délivrent des résultats sans tenir compte des critères spécifiques de l'utilisateur qui a émis la requête ou du contexte de la recherche.

Face à l'accroissement continu de ressources d'informations et de la diversité des utilisateurs et de leurs besoins et centres d'intérêt, les travaux en RI classique se sont orientés vers des approches adaptatives. Ces approches utilisent plusieurs sources

d'evidence (termes pertinents, document jugés, etc.) dans le but d'aider l'utilisateur à satisfaire son besoin informationnel : des techniques de reformulation de la requête ou pour desambiguïser la requête de l'utilisateur. Cependant, la RI adaptative ne résout pas la problématique de présentation de l'utilisateur et son besoin informationnel.

Prenons l'exemple du terme Java. Le terme Java est polysémique. Cette requête est alors ambiguë. Il peut faire référence au langage de programmation ou au nom d'une ile en Indonésie ou aussi des grains de café venant de cet ile. La RI doit prendre en considération le besoin utilisateur sous-jacent.

Dans ce même exemple, le contexte pourrait également être utile pour deux utilisateurs avec des besoins informationnels différents posant cette même requête. Le résultat fourni par le SRI risque de ne pas répondre au besoin d'un des deux utilisateurs comme par exemple un informaticien qui attend comme résultat des cours sur le langage JAVA, la description de l'ile java n'est pas pertinente.

Ces limitations ont ouvert la voie à la RI personnalisée. Ce domaine apporte principalement la prise en compte de l'utilisateur en tant que composante principale dans le processus de la recherche.

2.2 La recherche d'information personnalisée

La personnalisation peut être définie comme un processus qui change la fonctionnalité, l'interface, la teneur en information, ou l'aspect d'un système pour augmenter sa pertinence personnelle en fonction des caractéristiques de l'utilisateur. La personnalisation facilite et assiste l'utilisateur lors de sa recherche d'informations [Perenon, 2000].

Le besoin de la personnalisation apparaît en premier lieu face à la difficulté que rencontre l'utilisateur pour définir son besoin en information. Généralement, les requêtes posées par l'utilisateur sont constituées d'un à trois termes. En effet, pour affiner leurs besoins d'information les utilisateurs manquent d'efforts [Jansen *et al.* , 2000] et il reste souvent ambigu [Spink & Jansen, 2004]. Cette tâche d'expression de besoin est difficile même pour les utilisateurs expérimentés [Leake & Scherle, 2001].

D'autre part, la notion de pertinence qui représente le fondement et le cœur de la recherche d'information, est classée en plusieurs types et possède plusieurs natures [Borlund, 2003; Mizzaro, 1997; Saracevic, 2007]. Chaque type de pertinence est jugé par différents facteurs de jugement de pertinence [Saracevic, 2007].

Le choix du meilleur type de pertinence et du meilleur facteur de jugement face à un besoin informationnel difficile à exprimer semble être compliqué sans tenir compte de la diversité des intentions des utilisateurs pour une même requête, la diversité du langue, de culture, de lieu et des tâches de chaque utilisateur.

Un système de recherche d'information personnalisé est un système de recherche d'information qui prend en compte l'utilisateur tout au long du processus de la recherche. Il prend ainsi le contexte de l'utilisateur et répond d'une manière unique aux besoins informationnels de chaque utilisateur.

Nous commençons par une présentation générale de la RI personnalisée à savoir la notion de profil utilisateur. Ensuite, nous présentons d'une manière non exhaustive des

approches de personnalisation.

2.2.1 Profil utilisateur

Plusieurs définitions du profil ont été présentées dans la littérature en RI personnalisée : le profil cognitif [Leung *et al.* , 2006; Li & Belkin, 2008; Lieberman *et al.* , 1995; Pazzani *et al.* , 1996], le profil qualitatif [Harrathi & Calabretto, 2006] et le profil multidimensionnel [Kostadinov, 2003; Tamine *et al.* , 2007].
Le profil utilisateur peut être défini comme l'ensemble des données qui caractérisent l'utilisateur. Ces données englobent ses centres d'intérêts, son besoin informationnel et ses préférences. Un profil utilisateur peut aussi contenir des données personnelles et des préférences (sécurité ou qualité des données par exemple).

Le modèle de l'utilisateur est présenté par Höök et al. [Höök *et al.* , 1998] comme étant "une connaissance à propos de l'utilisateur explicitement ou implicitement codée, utilisée par le système afin d'améliorer son interaction"
On peut déduire de cette définition que la modélisation du profil de l'utilisateur s'articule au tour de la modélisation de ses préférences, de son besoin informationnel ou de ses intentions collectés d'une manière implicite ou explicite. En effet, construire un modèle de l'utilisateur nécessite de mettre en place une structure spécifique qui permet d'exploiter les données du profil utilisateur d'une manière optimale. Plusieurs techniques ont été utilisées afin de construire un profil utilisateur ainsi que le mettre à jour au fur et à mesure de l'évolution de l'utilisateur dans le temps.

Parmi les formalismes utilisés pour représenter le profil de l'utilisateur, on peut citer les vecteurs de mots clés [Santos & Vieira, 2004] les ontologies [Dempski, 2002], les

ensembles de prédicats pondérés [Koutrika & Ioannidis, 2005], les fonctions d'utilités [Cherniack *et al.* , 2003], les expressions logiques à la clause de Horn [Dell'Acqua *et al.* , 2002; Koutrika & Ioannidis, 2004] ou encore les approches statistiques [Tamine *et al.* , 2007].

2.2.2 Approches de la personnalisation

Avec l'apparition de la problématique de personnalisation dans la recherche d'information plusieurs travaux ont été développés. La figure 2.3 illustre les critères permettant de comparer les approches existantes. Le premier critère considère l'utilisateur comme cible de personnalisation en utilisant soit une personnalisation individuelle soit une personnalisation collaborative. Le deuxième couvre l'acquisition des données qui peut être explicite, implicite ou hybride. Un autre axe de personnalisation traite la durée de la session de recherche, à savoir la personnalisation basée sur le contexte courant (court-terme) et la personnalisation basée le contexte passé (long-terme).

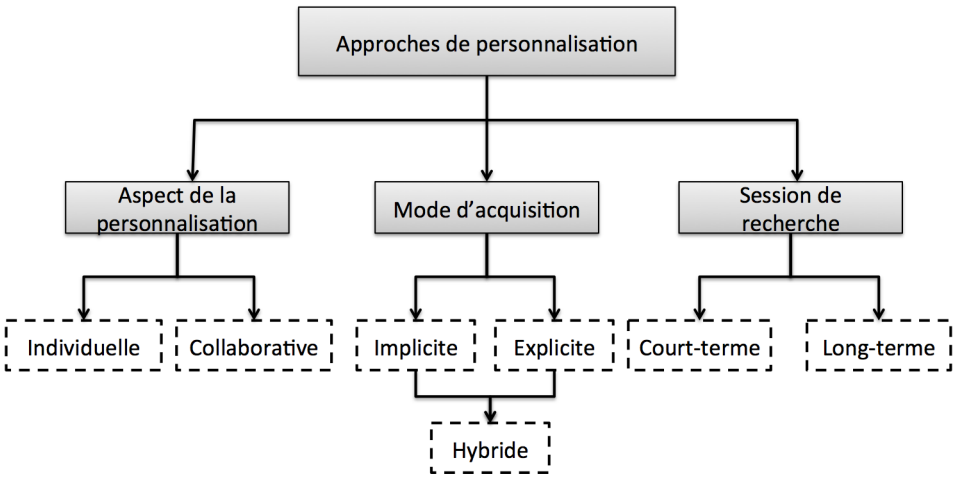

FIGURE 2.3 – Approches de la personnalisation

2.2.2.1 Personnalisation individuelle

La personnalisation individuelle [Liu *et al.* , 2004; Sieg *et al.* , 2007] où seul l'utilisateur est considéré s'effectue sur la base de ses préférences de contenu ou activités et mobilité via les applications qu'il utilise. La personnalisation individuelle est utilisée pour la recommandation d'objets ou ressources ou pour l'aider à accéder à des informations pertinentes en rapprochant les informations de son profil des informations décrivant les ressources considérées.

2.2.2.2 Personnalisation collaborative

La personnalisation collaborative [Morris *et al.* , 2008; Teevan *et al.* , 2009] considère l'utilisateur comme membre d'un groupe d'utilisateurs. Le modèle utilisateur est construit sur la base de ses préférences de contenu ou activités via les applications qu'il utilise ainsi que par les modèles des utilisateurs de son groupe. La personnalisation collaborative permet de recommander des ressources en exploitant l'expérience des autres utilisateurs.

2.2.2.3 Acquisition explicite

L'acquisition explicite est une approche simple de collecte de données puisqu'elle consiste en l'interrogation de l'utilisateur pour obtenir des informations par exemple en lui demandant de remplir un formulaire pour collecter les données personnelles ou démographiques telles que son âge, son sexe, sa langue maternelle [Hupfer & Detlor, 2006; Weber & Castillo, 2010] ou encore ses préférences concernant les documents retournés par exemple la langue ou le genre du document [Martin & Jose, 2004; Sieg *et al.* , 2007]) ainsi que le sujet d'intérêt de cet utilisateur [Cheverst *et al.* , 2000].

Par ailleurs, l'approche explicite peut être considérée comme un retour de pertinence

explicite (*feedback* explicite) de l'utilisateur utilisé le plus fréquemment dans les systèmes de filtrage ou de reformulation de la requête. L'utilisateur fournit alors, un jugement de pertinence en utilisant par exemple le jugement qualitatif sur l'information (j'aime, j'aime pas) [Chen & Sycara, 1998; Liu *et al.* , 2004] ou sous la forme de notes de pertinence, votes ou annotations.

Cette approche a été largement utilisée dans les systèmes de E-commerce dont le but est la personnalisation de l'interface des sites en respectant les préférences des utilisateurs. Plusieurs systèmes utilisent l'approche explicite pour collecter les données caractérisant l'utilisateur. On peut citer MyYahoo ! [Yahoo, 2016] qui demande explicitement aux utilisateurs de remplir des formulaires pour construire leur profils utilisateurs. Ainsi, le contenu affiché à l'utilisateur sur sa page contient que des informations pertinentes pour cet utilisateur puisqu'elles se basent sur les données qu'il a fournies.

L'acquisition explicite a un grand avantage du fait que les données sont fournies par l'utilisateur ce qui implique que les données sont facilement collectées et contiennent des informations correctes et certaines sur l'utilisateur. Par contre cette approche repose essentiellement sur l'effort fourni par l'utilisateur en remplissant des formulaires ou en donnant son jugement de pertinence. De ce fait, le désintéressement ou l'abandon de l'utilisateur implique la performance du système. En effet, dans le cas ou un utilisateur refuse de fournir des informations, la construction de son profil devient impossible.

2.2.2.4 Acquisition implicite

L'acquisition implicite est la collecte des données descriptives de l'utilisateur sans que celles-ci lui soient directement demandées. En effet, elle représente un retour implicite

de l'utilisateur qui peut être collecté en observant le comportement de l'utilisateur ainsi qu'en analysant ses activités.

L'activité de l'utilisateur est une source primordiale pour la personnalisation implicite. Cette activité peut être déterminée à partir :

- de l'utilisation des moteurs de recherche : les pages visitées, des requêtes passées [Liu *et al.* , 2004; Shen *et al.* , 2005],
- de la navigation sur le web : les données de navigation, les liens [Speretta & Gauch, 2005; Yau *et al.* , 2003],
- des applications utilisées : outils de messagerie éléctronique, les fichiers logs [Dumais *et al.* , 2003; Yau *et al.* , 2003],
- des favoris [Dumais *et al.* , 2003],
- de l'historique de localisations de l'utilisateur [Ying *et al.* , 2010] .

Diverses sources d'informations sont utilisées pour collecter les informations implicitement. Principalement, ces informations sont dérivées des interactions ou de navigation de l'utilisateur avec le système ou encore de son historique de recherche.

L'historique des interactions ou de navigation est généralement dérivé des actions de l'utilisateurs effectuées sur les pages consultés telles que :

- Les mouvements des yeux ; les données de clicks [Sun *et al.* , 2005] ;
- Les actions sur les documents tellesque l'ouverture, la fermeture, l'impression ou le temps de lecture etc.) [Pretschner & Gauch, 1999] ;
- Les messages (e-mails) envoyés ou reçus [Bellotti *et al.* , 2008] ;

- Les annotations sociales, et les bookmarks [Hecht *et al.* , 2012; Karweg *et al.* , 2011];

Les données démographiques peuvent aussi être dérivées des interactions et activités des utilisateurs [Hu *et al.* , 2007], des styles d'écriture [Argamon *et al.* , 2003] ou des requêtes [Jones *et al.* , 2007].

L'historique de recherche représente la deuxième source de données implicite pour construire le profil de l'utilisateur. Il regroupe les informations issues de ses requêtes, des documents consultés les données situationnelles qui dépendent du lieu et temps d'émission de la requête [Abowd *et al.* , 1997; Van Setten *et al.* , 2004]. Nous citons aussi, l'environnement social de l'utilisateur tels que ses amis [Church *et al.* , 2010; Hwang *et al.* , 2007] ou personnes physiquement proches [Rhodes, 2003].
Les données sociales de l'utilisateur telles que les annotations, les posts, les blogs, les messages, les signaux, etc.[Hecht *et al.* , 2012; Karweg *et al.* , 2011; Vallet *et al.* , 2010], représentent aussi une source d'acquisition implicite.

L'historique des interactions est collecté généralement en utilisant un serveur proxy qui observe le comportement et les interactions de l'utilisateur avec le système. L'avantage majeur de cette technique est qu'elle nécessite une implication minimale de l'utilisateur et ne nécessite aucune installation de logiciel pour collecter les informations. Mais la construction de profil semble être difficile puisque les utilisateurs se connectent d'une manière anonyme.
Pour faire face à cette limite, le sytème propose à l'utilisateur de s'identifier et à chaque fois qu'il se connecte son historique est alimenté.
D'autre systèmes utilisent des agents installés sur le système de l'utilisateur ou sur

les navigateurs pour capturer les activités de l'utilisateur. Letizia [Lieberman *et al.* , 1995] est l'un des premiers systèmes de personnalisation utilisant le feedback implicite collecté par des agents. En se basant sur les pages visitées précédemment ainsi que les annotations de l'utilisateur, il suggère des liens qui peuvent être pertinents l'utilisateur sur sa page courante.

2.2.2.5 Acquisition hybride

Enfin la personnalisation hybride utilise à la fois la personnalisation implicite et explicite. Cette personnalisation hybride peut commencer par la collecte explicite puis implicite (système WAIR,[Zhang & Seo, 2001]) ou la collecte implicite puis explicite. Dans le premier cas, une collecte explicite sert à initialiser le profil qui est ensuite raffiné par une collecte implicite de données. Dans le second, le profil est construit automatiquement par l'observation des activités de l'utilisateur, puis validé et complété par une intervention explicite de l'utilisateur.

2.2.2.6 Personnalisation à partir d'un contexte courant

La personnalisation basée sur le contexte courant (court-terme) s'appuie sur une mémoire de personnalisation qui est la durée de la session. Dans ce cas on ne tient pas compte de l'historique et des interactions de l'utilisateur dans les sessions précédentes. Pour la personnalisation à court terme on rencontre un problème de délimitation de la session de recherche. Certains travaux proposent une délimitation basée sur un intervalle de temps par exemple 30 min [Silverstein *et al.* , 1999]. Une délimitation basée sur la similitude est aussi proposée. Cette similitude peut être une similitude de requêtes successives [He *et al.* , 2002; Jones & Klinkner, 2008] ou sur une similitude des résultats de recherches successives [Daoud *et al.* , 2009].

2.2.2.7 Personnalisation à partir d'un contexte passé

La personnalisation basée sur le contexte passé (long-terme) s'appuie sur une mémoire de personnalisation qui est égale à la durée de vie du compte utilisateur. Dans ce cas, le comportement de l'utilisateur est observé tout au long de son interaction avec le système. Il tient compte de tout l'historique et toutes les interactions de l'utilisateur avec le système. Dans ces systèmes, on peut délimiter les sessions de recherche [Sieg *et al.* , 2007] ou ne pas les délimiter [Liu *et al.* , 2004].

2.3 Systèmes de recommandation

Un système de recommandation est une technique de filtrage d'information utilisée pour présenter les éléments d'information qui peuvent intéresser l'utilisateur.

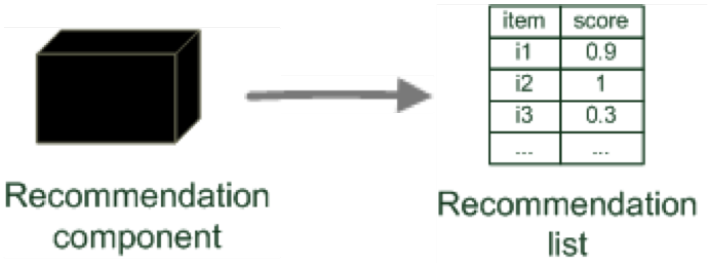

FIGURE 2.4 – La recommandation vu comme une boite noire [Jannach *et al.* , 2010]

Un système de recommandation (RS) aide les utilisateurs qui n'ont pas suffisamment de compétence ou l'expérience nécessaire pour évaluer le nombre, potentiellement très important, de ressources offertes par une source documentaire. La figure 2.4 montre la vision de Jannach [Jannach *et al.* , 2010] d'un service de recommandation dont le résultat attendu est d'attribuer un score à une liste d'objets.

Par exemple dans le cadre du commerce électronique, les sites commerçants recommandent à leurs utilisateurs des listes personnalisées et classées d'articles et fournissent aux consommateurs des renseignements pour les aider à décider quels articles acheter. Un article peut être un film, un livre, musique, etc. plus généralement on parle d'items.

2.3.1 Recommandation versus Personnalisation

Il est utile de faire la difference entre la personnalisation et la recommandation. En effet ces deux notions sont généralement confondues dans plusieurs domaines et surtout dans le domaine de la recherche d'information.

2.3.1.1 La personnalisation

La personnalisation peut être définie comme la manière d'affiner le besoin d'information de l'utilisateur. En effet, la personnalisation de la requête en tenant compte du profil de l'utilisateur consiste à rajouter des contraintes à cette requête suivant les préférences de l'utilisateur. Ces contraintes ajoutées limiteront le champs de recherche tout en restant dans le même cadre que la requête initiale. C'est la notion d'inclusion qui est à la base de la personnalisation. Les résultats d'une requête personnalisée sont généralement inclus dans les résultats de la requête initiale.

Par exemple, pour un utilisateur qui consulte des articles scientifiques sur la recherche d'information la personnalisation de la requête peut retourner comme résultats des articles sur la RI personnalisée ou même affiner le besoin jusqu'à arriver à la modélisation du profil utilisateur dans la recherche d'information personnalisée.

2.3.1.2 La recommandation

Contrairement à la personnalisation, la recommandation se base sur des items qui existent déjà et qui sont par exemple extraits d'un ensemble items consultés par d'autres utilisateurs. De ce fait, les éléments recommandés peuvent avoir des caractéristiques différentes des items initiaux.

Par exemple, pour un utilisateur qui consulte des articles scientifiques sur la recherche d'information, la recommandation peut proposer des articles parlant de la personnalisation ou la recommandation comme proposer des articles sur la découverte des documents, etc.

2.3.2 La recommandation à base de contenu

Pour les approches à base de contenu, détaillées par [Pazzani & Billsus, 2007], l'objectif du système est de recommander des items qui correspondent au profil de l'utilisateur d'après les informations décrivant les items. Généralement, le profil de l'utilisateur est basé sur des items que l'utilisateur a aimé par le passé ou sur des intérêts explicitement exprimés par l'utilisateur. Un système de recommandation basé sur le contenu compare les caractéristiques d'un item avec les caractéristiques du profil de l'utilisateur pour déterminer sa pertinence pour cet utilisateur. Le système de recommandation consiste alors à déterminer quels items correspondent le plus aux préférences de l'utilisateur. Ce système ne nécessite pas un grand nombre d'utilisateurs ni une longue session d'historique de l'utilisateur.

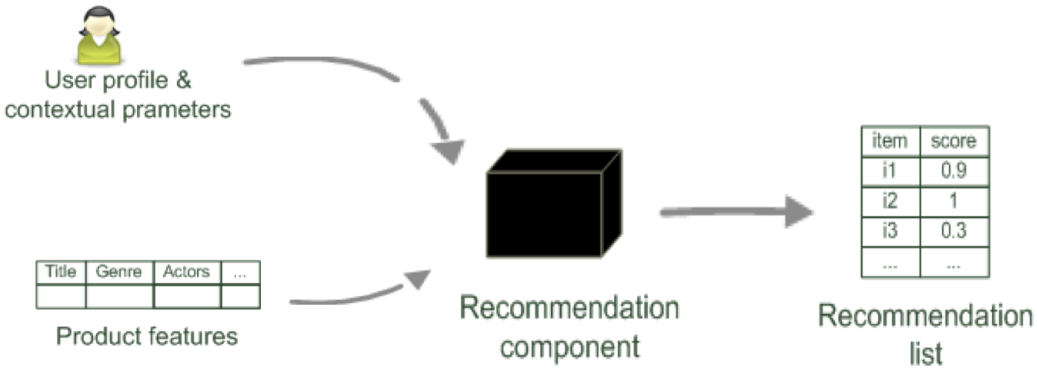

FIGURE 2.5 – La recommandation à base de contenu vu comme une boite noire [Jannach *et al.* , 2010]

Les étapes de la recommandation à base de contenu sont les suivantes :

- la sélection des documents pertinents vis-à-vis du profil ;
- la mise à jour du profil de l'utilisateur en fonction du retour de pertinence fourni par l'utilisateur sur les documents qu'il a reçus ; la mise à jour se fait par intégration des thèmes abordés dans les documents jugés pertinents.

Les avantages des approches à base de contenu sont :

- la recommandation d'items similaires à ceux que les utilisateurs ont aimé dans le passé ;
- la correspondance entre les préférences de l'utilisateur et les caractéristiques des items fonctionne aussi pour les données textuelles ;
- la possibilité de faire des recommandations à des utilisateurs avec des goûts "uniques" ;
- la possibilité de recommander de nouveaux items ou même des items qui ne sont pas populaires.

Cependant, ces systèmes possèdent des inconvénients :

- les items représentés par le même ensemble de mots-clés ne peuvent pas être distingués ;
- les utilisateurs avec un grand nombre de centres d'intérêt/items sont un problème ;
- l'utilisateur doit fournir un retour de pertinence sur les suggestions retournées pour produire des recommandations précises ;
- les jugements des autres utilisateurs ne sont pas exploitées.

2.3.3 La recommandation à base de filtrage collaboratif

Le paradigme du filtrage collaboratif apporte précisément une réponse à ces problèmes, en s'appuyant sur la communauté des utilisateurs du système. Les systèmes basés sur le filtrage collaboratif produisent des recommandations en calculant la similarité entre les préférences d'un utilisateur et celles d'autres utilisateurs. De tels systèmes n'ont pas besoin d'analyser ou de comprendre le contenu des items à recommander. Néanmoins, ils sont capables de suggérer de nouveaux items à des utilisateurs qui ont des préférences similaires à d'autres. La méthode consiste à faire des prévisions automatiques (filtrage) sur les intérêts d'un utilisateur en collectant des avis de nombreux utilisateurs (collaborateurs). L'hypothèse sous-jacente de cette approche est que ceux qui ont accepté un item particulier dans le passé ont tendance à accepter de continuer à aimer cet item de nouveau dans l'avenir.

Les approches collaboratives essaient de prédire l'opinion de l'utilisateur sur différents items et être en mesure de recommander le meilleur item à chaque utilisateur en

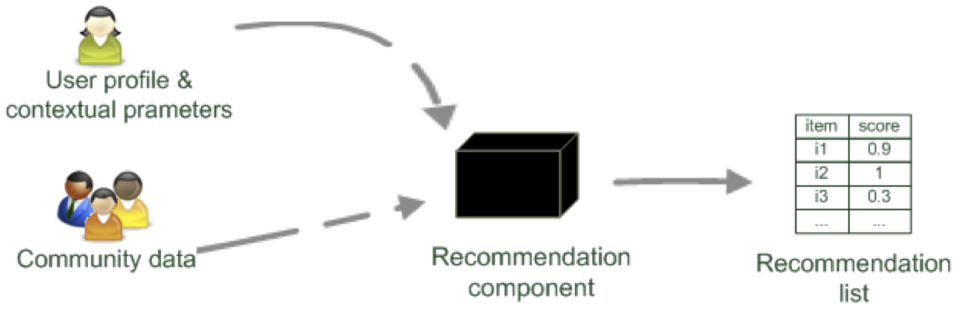

FIGURE 2.6 – La recommandation à base de filtrage collaboratif vu comme une boite noire [Jannach *et al.* , 2010]

fonction des goûts/avis précédents de l'utilisateur et des avis d'autres utilisateurs qui lui sont semblables. Cela est généralement réalisé suivant le mécanisme suivant :

- enregistrer un grand groupe de préférences d'utilisateurs ;
- repérer un sous-groupe d'utilisateurs dont les préférences sont similaires à celles de l'utilisateur qui cherche la recommandation ;
- calculer une moyenne des préférences pour ce groupe ;
- utiliser la fonction de préférence qui en résulte pour recommander des options à l'utilisateur qui cherche la recommandation.

Les avantages des approches collaboratives sont :

- l'utilisation des recommandations d'autres utilisateurs (score) pour évaluer l'utilité des items ;
- l'utilité du nombre des utilisateurs : Plus il y a d'utilisateurs, plus il y a de scores, meilleurs sont les résultats.

Cependant, ces systèmes possèdent des inconvénients :

- il est difficile de trouver des utilisateurs ou groupes d'utilisateurs similaires ;
- le système de recommandation doit gérer des faibles densités de la matrice utilisateur/item.

2.3.4 La recommandation à base de confiance

Les systèmes de recommandation à base de confiance sont aussi appelés recommandations sociales. La recommandation à base de confiance ressemble au filtrage collaboratif mais avec deux différences principales. D'une part, la notion de voisinage du filtrage collaboratif est remplacée par la notion de confiance. En effet, c'est l'utilisateur qui choisit ses voisins souvent désignés comme étant ses "amis".
D'autre part, une fonction de propagation de confiance vient se rajouter au système de recommandation. Cette fonction permet à l'utilisateur d'exploiter l'experience des amis de ses amis. [Golbeck & Hendler, 2006; Massa & Avesani, 2004; Walter *et al.* , 2008].

La notion de confiance a émergé dans plusieurs domaines comme par exemple le cloud computing [Li & Ping, 2009], le partage des fichiers [Kamvar *et al.* , 2003] et dans les domaines ou les technologies agent sont appliquées [Huynh *et al.* , 2006; Ramchurn *et al.* , 2004; Sabater & Sierra, 2005].

La notion de confiance ainsi que celle de réputation ont aussi émergés dans le domaine de recommandation. Elles ont été appliquées selon deux axes principaux : la confiance locale qui représente la relation entre deux individus dans le système et la réputation dite aussi confiance globale qui représente l'avis de la communauté concernant un individu [Ziegler & Lausen, 2004]. La confiance locale est la plus fréquente dans les

systèmes de recommandation. Plusieurs approches de confiance ont été utilisées dans le domaine de la recommandation. Nous pouvons citer quelques modèles proposés pour la recommandation à base de confiance : [Golbeck & Hendler, 2006; Massa & Avesani, 2004; Victor *et al.* , 2009; Walter *et al.* , 2008].

2.3.5 La recommandation hybride

La recommandation hybride utilise plusieurs types d'approches de recommandation. Par exemple, un système à base de recommandation hybride peut combiner un système de recommandation à base de contenu et d'un filtrage collaboratif. Il existe trois manières

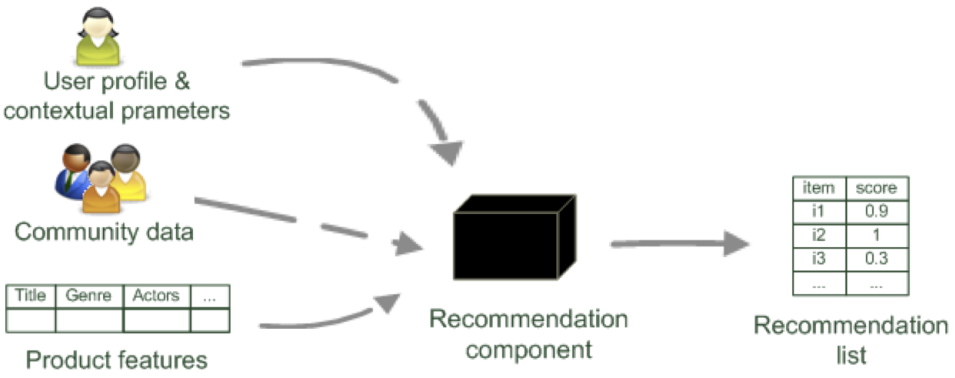

FIGURE 2.7 – La recommandation hybride vu comme une boite noire [Jannach *et al.* , 2010]

de concevoir l'hybridation des approches de recommandation à savoir [Jannach *et al.* , 2010] : l'hybridation monolithique, l'hybridation en pipeline et l'hybridation parallèle.

L'hybridation monolithique intègre différentes approches de recommandations en un seul algorithme. Différents systèmes de recommandation contribuent dans ce processus. Les deux autres approches de recommandation demandent au moins deux implémenta-

tions séparées de systèmes de recommandation qui seront ensuite combinées.

Les systèmes de recommandation hybride parallèles opèrent indépendamment en utilisant les mêmes entrées et produisent des listes de recommandation séparées. Ces sorties sont ensuite combinées après dans une étape d'hybridation qui fournira une liste finale de recommandation.

Dans le cas ou les systèmes de recommandation sont liés sous la forme de pipeline, la sortie d'un système de recommandation présente les entrés pour le système de recommandation suivant. Les paramètres d'entrée initiaux peuvent être utilisés comme supplément des autres systèmes de recommandation.

2.4 Conclusion

Nous avons présenté au cours de ce chapitre, les principaux éléments de la recherche d'information personnalisée qui vient compléter la recherche d'information classique. Nous avons détaillé les principales notions de la RI classique ainsi que le processus de la RI commençant par l'indexation jusqu'à l'appariement.

Les phases du processus de la RI nécessitent des améliorations et surtout l'intégration de l'utilisateur comme facteur principal à prendre en compte dans la recherche de satisfaction de son besoin informationnel. Plusieurs approches de personnalisations sont apparues et elles se distinguent selon plusieurs critères. Un premier critère est la prise en compte ou non de l'environnement de l'utilisateur (pris individuellement ou dans un groupe d'utilisateurs). Un autre porte sur les modes d'acquisition des données sur les utilisateurs : explicite, implicite et hybride. Enfin la durée considérée pour la session de

recherche peut aussi varier selon que l'on représente une session à court terme ou à long terme.

Des travaux dans le domaine de la recommandation viennent compléter les approches de personnalisation afin de mieux cerner le besoin informationnel de l'utilisateur et l'aider à trouver plus facilement ce qu'il cherche en lui recommandant des ressources documentaires.

Nous détaillons dans le chapitre suivant les systèmes qui ont exploité ces approches de personnalisation pour assister l'utilisateur dans ses parcours de navigation dans les corpus.

CHAPITRE 3

SYSTÈMES D'AGENTS ASSISTANTS

Sommaire

3.1 Assistance aux utilisateurs **55**

3.1.1 Les agents assistants 56

3.1.2 Les caractéristiques d'un agent assistant personnel 56

3.1.3 Les types d'agent assistant personnel 58

3.1.4 Exemples 59

3.2 Assistance à la recherche d'information par système multi-agent **61**

3.2.1 L'agentification du processus de la RI 62

3.2.2 L'assistance de l'utilisateur et la personnalisation de la RI par des systèmes multi-agents 63

3.3 Approche stigmergique des systèmes multi-agents **67**

3.3.1 Le modèle A&A : Agent et Artefact 69

3.3.2 Les agents dans le A&A méta-modèle 70

3.3.3 Les artefacts dans le méta-modèle A&A 71

3.3.4 Les systèmes multi-agent à base d'A&A 74

3.4 Conclusion **75**

La révolution numérique a multiplié les systèmes informatiques en interaction avec les utilisateurs. Malgré le fait que ces systèmes aient été conçus pour faciliter les tâches courantes de l'utilisateur, leur prise en main n'est pas forcément aisée pour les utilisateurs non experts. Face à ces difficultés, les utilisateurs ont parfois besoin d'une assistance à leur utilisation ainsi que de mécanismes de personnalisation pour leur présenter un système adapté à leurs besoins et compétences.

Ce chapitre est organisé comme suit : La section 3.1 représente le problème général de l'assistance aux utilisateurs. Elle présente les agents assistants personnels en détaillant leur attributs ainsi que leur types. La section 3.2 détaille les approches d'assistance à la recherche d'information par système multiagent. Plus particulièrement, nous décrivons les travaux utilisant les systèmes multi-agents pour la recherche d'information dans un but d'assistance aux utilisateurs en recherche d'information classique et personnalisée. Finalement, dans la section 3.3 nous présentons l'approche stigmergique du développement de systèmes multi-agents qui sera suivie dans cette thèse. Sa mise en œuvre dans le modèle Agents & Artefacts (A&A) est présentée plus en détail.

3.1 Assistance aux utilisateurs

L'assistance désigne l'action de porter aide ou secours à une personne. Il existe plusieurs types d'assistance et plusieurs domaine d'assistance. En informatique, nous parlons surtout du support aux utilisateurs dit aussi services d'assistance qui consistent à garantir que les utilisateurs d'un système puissent continuer à profiter de la disponibilité de l'ensemble de ses composants pour l'accomplissement de leurs tâches. L'assistance peut porter sur les applications, ou sur les composants techniques (plateformes informatiques, réseaux).

3.1.1 Les agents assistants

Un agent assistant personnel peut être défini comme étant un agent logiciel qui a pour fonction d'assistant dans le but de alléger et diminuer les charges de travail d'un individu pendant la réalisation des activités quotidiennes ou professionnelles qui demandent une interaction avec un ordinateur, un smartphone ou tout autre moyen informatique.

Définition 1 (Agent assistant personnel [Enembreck, 2003])**.**

"Un agent assistant personnel ou tout simplement IPA (Intelligent Personal Assistant) est un agent logiciel qui permet d'effectuer des tâches ou des services pour un individu. Cet agent a pour rôle d'aider son *maître* dans son travail quotidien".

Les tâches et services qu'exécute l'agent assistant personnel se basent sur les données ou les entrées de l'utilisateur, ses connaissances, ses activités, son emplacement et la possibilité d'accéder à ces informations en ligne. Un agent assistant personnel aide l'utilisateur à exécuter ses tâches ou même les exécuter à la place de l'utilisateur.

3.1.2 Les caractéristiques d'un agent assistant personnel

Les principales caractéristiques d'un IPA sont :

- **L'autonomie** : Un IPA doit être capable d'agir sans demande de l'utilisateur. Il doit être autonome pour pouvoir prendre des décisions tout seul sans l'intervention de l'utilisateur. En effet, l'agent doit observer l'activité de l'utilisateur ainsi que reconnaitre ses actions répétitives et ses préférences.

- **La personnalisation** : Un agent assistant personnel doit s'adapter à son *maître* et agir selon ses préférences. Par exemple, pendant la navigation de l'utilisateur, un agent assistant personnel, en connaissance des préférences de l'utilisateur, peut réduire l'ambiguïté dans les thèmes proposés à l'utilisateur en lui proposant le sites qui respectent ses centres d'intérêt.

- **La compétence** : Dans son domaine d'application, un agent assistant personnel doit être compétent. En effet un agent qui donne des mauvaises réponses à l'utilisateur perd sa confiance. La compétence est un attribut très important dans l'acquisition de la confiance de l'utilisateur. En outre, plus l'utilisateur a confiance plus il donnera des tâches à l'agent pour qu'il les exécute à sa place. Par exemple, un IPA qui enregistre mal un événement dans le calendrier de l'utilisateur pousse l'utilisateur à le faire tout seul. Plusieurs domaines de compétences d'un IPA existent à titre d'exemple nous pouvons mentionner :

  - dialoguer avec l'utilisateur, établir un discours ou une conversation avec l'utilisateur ;
  - répondre aux courriers électroniques de l'utilisateur d'une manière automatique ;
  - rechercher une information par thème sur internet ;
  - s'adapter aux changements d'activités de l'utilisateur ;
  - guider l'utilisateur pendant l'exécution de ses tâches ou les exécuter à sa place à sa demande ;
  - gérer et classer les courriers de l'utilisateur, les rendez vous, l'emploi de temps.

- **La communication** : L'agent doit être capable de tenir une discussion avec son utilisateur. En effet, cet attribut rejoint l'attribut de compétence dans son importance, plus l'utilisateur trouve des facilités en communiquant avec son agent assistant plus il lui confiera des tâches. Par exemple un agent assistant personnel conversationnel [Bersot *et al.* , 1998] qui utilise bien le langage naturel, facilite la tâche à l'utilisateur puisqu'il utilise son langage de communication.

- **La coopération** : Dans le cas de tâches complexes que l'agent assistant ne peut pas exécuter tout seul, il doit être capable de coopérer avec d'autres agents pour les exécuter. Par exemple, il peut coopérer avec un agent e-commerce pour effectuer des achat qui correspondent aux exigence de l'utilisateur avec les meilleurs coûts.

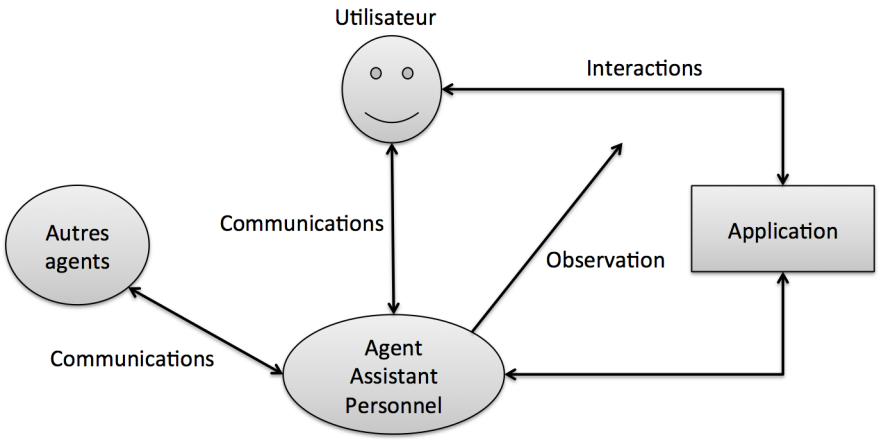

FIGURE 3.1 – Le fonctionnement d'un agent assistant personnel

3.1.3 Les types d'agent assistant personnel

Il existe différents types d'agent assistant personnel. Le type de l'agent est défini par les attributs qui le caractérisent [Maes *et al.* , 1994] :

- **Les agents semi-autonomes** [Lai *et al.* , 1988] : Ce type d'agent n'est pas *compétent* mais il est fiable. En outre, le fonctionnement de cet agent dépend de son script écrit par un programmeur. Il exécute que les tâches définies dans son code de programmation.

- **Les agents à base de connaissance** [Chin, 1990; Sullivan *et al.* , 1994] : Un IPA dit à base de connaissance est un agent qui possède une base de connaissance sur son utilisateur ainsi que sur ses objectifs. Il est capable d'assister l'utilisateur durant les tâches de ses plans d'actions. En effet, il est aussi capable de reconnaître les plans d'actions de l'utilisateur. Cependant, la conception et la programmation d'un agent à base de connaissance demande beaucoup de travail, notamment pendant la création de la base de connaissance ainsi que sa mise à jour.

- **Les agents apprenants** [Dent *et al.* , 1992] : Un agent apprenant possède une base de connaissance initiale qu'il enrichit au cours de son exécution. Un IPA apprenant observe les actions de son utilisateur de manière à améliorer sa compétence. Cette adaptation, en apprenant à partir des comportements de l'utilisateur, l'amène à faire évoluer son processus de décision et il est souvent souhaitable de pouvoir expliquer la cause de ces changements.

3.1.4 Exemples

Plusieurs exemples d'agents assistants existent. Parmi les premiers agents assistants nous pouvons citer Maxims [Metral & Maes, 1998]. C'est un agent qui aide l'utilisateur dans la gestion de son courrier électronique. Maxims apprend à mettre en prioritaire, supprimer, de transférer, trier et archiver les messages électroniques de l'utilisateur. La principale technique d'apprentissage utilisé par Maxims est le raisonnement à base de

mémoire [Stanfill & Waltz, 1986]. L'agent observe en continu l'utilisateur lorsqu'il traite de courrier électronique. Lorsque l'utilisateur effectue des actions, l'agent mémorise toutes les paires situation-action générés. Par exemple, si l'utilisateur enregistre un message de courrier électronique particulier après l'avoir lu, l'agent de messagerie ajoute une description de cette situation et les mesures prises par l'utilisateur à sa mémoire des exemples.

Cet agent est aussi utilisé pour assister l'utilisateur dans l'ordonnancement de ses réunions [Kozierok & Maes, 1993; Kozierok, 1993]. Il possède un fonctionnement semblable basé sur le raisonnement à base de mémoire où l'agent peut accepter ou rejeter une réunion, l'enregistrer dans le calendrier, planifier, négocier les temps de réunion, etc. Certains agents ont été également conçus avec ont pour fonctionnalité le filtrage des actualités [Sheth & Maes, 1993; Sheth, 1994]. Face à l'énorme flux d'information et d'actualités disponible sur internet, l'utilisateur à besoin d'un outil qui lui filtre les actualités les plus intéressantes.

Les applications précédemment citées sont mono-tâches et possèdent un seul objectif. Actuellement, les travaux scientifiques ont avancer pour proposer des applications multi-tâches comme par exemple :

Siri [Aron, 2011; inc., 2016] est un assistant personnel intelligent qui permet d'effectuer diverses opérations, déployé dans les téléphones d'Apple. Siri fonctionne grâce à la reconnaissance vocale avancée et le traitement du langage naturel oral et la synthèse vocale. Il est présenté à la presse dans une premiere version le 4 octobre 2011. Siri permet d'envoyer des messages, de passer des appels d'effectuer des réservations, d'afficher un itinéraire ou les commerces dans l'entourage, etc. Siri peut aussi effectuer des recherches d'information pour l'utilisateur. Mais cette recherche se limite dans la

tâche de conversion de la commande vocale de l'utilisateur en requête texte et lancer la recherche de cette requête dans un moteur de recherche. La limite majeure dans ce cas est que Siri apporte une assistance à l'utilisateur mais qui n'est ni personnalisée ni adaptée à l'utilisateur.

Google Now [Google, 2016] représente l'alternative de Siri sur les appareils tournant sous Android. C'est un assistant personnel intelligent basé sur la reconnaissance vocale, le traitement de langage naturel oral et écrit et la synthèse vocale pour répondre aux requêtes de l'utilisateur à l'oral et à l'écrit. Google Now profite des fonctionnalités de Google pour proposer plusieurs fonctionnalités : traduction de mots, les rendez vous (via google calender), les lieux (via google maps), la météo, les nouveaux articles des sites que l'utilisateur consulte fréquemment, etc.

D'autres agents personnels intelligents existent ayant un fonctionnement semblable à Google Now et Siri : *Amazon Echo*, *Cortana* de Microsoft, *S Voice* de Samsung, *Voice Mate* de LG etc.

3.2 Assistance à la recherche d'information par système multi-agent

Dans le domaine de la recherche d'information, nous pouvons trouver de nombreux travaux dans la littérature ayant adopté l'approche multi-agent au sein de problèmes de recherche d'information. Les premières approches multi-agents apparues dans le domaine de recherche d'information se placent dans le cadre de la modélisation des systèmes de recherche d'information. Généralement, les tâches du processus sont affectées

à différents agents qui peuvent les exécuter en séquence ou en concurrence. Avec l'émergence de la personnalisation de la recherche d'information les systèmes multi-agent se sont aussi adaptés pour réaliser la tâche de la personnalisation. Si nous revenons à la définition de l'assistance, la personnalisation rejoint l'assistance dans la proposition de services et fonctionnalités personnalisées et adaptées à l'utilisateur qui peuvent être encapsulés dans différents agents. Nous commençons ici par présenter les premières approches d'agentification de la recherche d'information classique. Ensuite nous détaillons les approches ayant opté pour l'assistance de l'utilisateur et la personnalisation de la RI par des systèmes multi-agents.

3.2.1 L'agentification du processus de la RI

L'agentification d'un processus de RI consiste en une modélisation qui représente chaque tâche du processus de la par un agent. En général, un système multi-agent pour la recherche d'information est composé d'un agent recherche, un agent visualisation, agent requête, etc. Pour des systèmes spécifiques viennent s'ajouter des agents ressources, des agents ontologie, etc.

Parmi les premiers systèmes multi-agents pour la recherche d'information nous pouvons citer à titre d'exemple :

- **RETSINA**[Sycara *et al.* , 1996] (Reusable Task Structure based Intelligent Network Agents) : C'est un ensemble réutilisable de composants logiciels pour la construction agents sous la forme d'une architecture multi-agent distribuée basée sur les interactions et la collaboration des agents. Cette architecture est mise en œuvre et développée dans diverses tâches complexes du monde réel à savoir le système WARREN qui intègre l'information trouvée et filtrée pour aider un

utilisateur dans le contrôle de son portefeuille financier et le système de prise de décision organisationnelle PLEIADES.

- Le système **NetSA** [Côté & Troudi, 1998] ajoute un agent courtier, au moins un serveur d'ontologie et au moins un agent d'exécution.

- **InfoSleuth** [Nodine *et al.* , 2000] qui est un système multi-agent pour la recherche coopérative d'informations dans des bases de données distribuées composé de trois types d'agents principaux : agents d'utilisateur, agents ressources et agents de noyau.

- **AgentSeek system** [Grey *et al.* , 2000] : un système d'agents mobiles qui se déplacent à travers le web d'une manière dirigée pour demander des renseignements au nom des utilisateurs.

- **UMDL**, le système de recherche des documents dans une librairie digitale [Chaib-Draa *et al.* , 2001], est un système d'informations coopératif pour la recherche des documents dans une librairie digitale. Il utilise des agents capables d'acheter et de vendre des services entre eux ou aux utilisateurs.

3.2.2 L'assistance de l'utilisateur et la personnalisation de la RI par des systèmes multi-agents

Avec l'apparition de la notion de profil autrement dit de la personnalisation, les systèmes multi-agents se sont adaptés pour mieux convenir et répondre aux besoins des utilisateurs. Généralement, les approches de personnalisation utilisent des agents pour collecter les informations des utilisateurs d'une manière interactive au cours de ses navigations.

Les agents assistants intégrant des mécanismes de personnalisation sont souvent implémentés soit en tant qu'un plug-in à installer dans navigateur de l'utilisateur ou sous la forme d'une application qui représente une sorte de navigateur internet, de manière à pouvoir collecter des traces de navigation de leur utilisateur. Un ensemble riche d'information sur l'utilisateur est aussi disponible grace aux historiques de navigation. De plus, l'agent peut collecter les urls visités par l'utilisateur ainsi que le temps passé sur une page web. Il peut aussi collecter les actions que l'utilisateur a effectué ainsi que les favoris et les téléchargements de l'utilisateur.

Letizia [Lieberman, 1997; Lieberman *et al.* , 1995] est l'un des premiers systèmes qui collecte les informations de l'utilisateur d'une manière interactive et exploite ses feedbacks implicites en se basant sur les pages précédemment visitées et les pages favorites. Il suggère des liens sur la page en cours qui pourraient être d'intérêts. Plusieurs systèmes à base d'agent assistants de navigation existent tel que **WebMate** [Chen & Sycara, 1998], **Vistabar** [Marais & Bharat, 1997], et **Personal WebWatcher** [Mladenic, 1996].

Certains travaux dans la littérature établissent une distinction entre un assistant de navigation et un agent de navigation. En effet, **Vistabar** [Marais & Bharat, 1997] est un prototype d'assistant de navigation, un outil qui aide les utilisateurs à suivre les urls vus, remplir des formulaires ou à extraire des pages sans ordre du jour précis. En revanche, **WebMate** [Chen et Sycara 1998] et **Personal WebWatcher** [Mladenic, 1996] sont des exemples d'agents de navigation qui effectuent des tâches plus critiques telles que mettre en évidence des liens hypertextes susceptibles d'intéresser l'utilisateur,

recommander des URLs, ou raffiner les mots-clés de la recherche.

Dans le même contexte de personnalisation de la recherche d'information, nous trouvons dans la littérature plusieurs travaux qui implémentent des services de personnalisation par un système multi-agent. Par exemple :

- Fung et al. [Fung & Wongthongtham, 2002] ont présenté deux propositions pour résoudre le problème de la recherche distribuée sur Internet. La première est basée sur l'utilisation d'un agent IPA pour aider l'utilisateur dans le processus de recherche d'information, et le second, traite des bases de connaissances distribuées situées à différents serveurs comme une connaissance intégrée du domaine.

- **BASAR** (Building Agents Supporting Adaptive Retrieval) [Thomas & Fischer, 1997] fournit aux utilisateurs une assistance lors de la gestion de leurs espaces d'informations personnelles. Cette assistance est spécifique à l'utilisateur et faite par des agents logiciels, appelés des assistants web, et les vues actives spécifiques à l'utilisateur. Les utilisateurs délèguent des tâches aux assistants web qui exécutent des actions sur leurs vues sur le web, sur le Web lui-même, et sur l'historique de toutes les actions de l'utilisateur.

- Ding et al. [Ding *et al.* , 2005] ont présenté un agent intelligent personnel pour la recherche sur le web (**PAWS** : Personal Agent for Web Search) conçu pour effectuer la recherche personnalisée sur le web pour chaque utilisateur en fonction de ses préférences individuelles. Les PAWS utilisent intelligemment une carte d'auto-organisation (SOM : Self-Organizing Map) comme étant le profil de l'utilisateur, et par conséquent, est capable de fournir une réponse de haute qualité mise à l'utilisateur.

- Zhang et al. [Zhang *et al.* , 2006] a proposé un modèle de recherche d'information intelligente basée sur le paradigme multi-agent et Graphes Conceptuels (GC). Le GC, développé par Sowa [Sowa, 1983], est un langage de représentation des connaissances initialement conçu pour saisir le sens du langage naturel. Le système représente des requêtes et des documents en GC, qui entraîne la sémantique dans certaines fonctions, telles que la recherche intelligente, l'auto-notification, le guide de navigation, et la gestion des informations personnelles.

- **SARIPOD** qui est un Système multi-Agent de Recherche Intelligente POssibiliste de Documents Web [Elayeb, 2009]. Ce modèle est un modèle multi-agent qui offre une collaboration entre les différents acteurs et la mise en oeuvre de toutes les fonctionnalités du système de recherche d'information. Il est à base de deux Réseaux Petits Mondes Hiérarchiques (RPMH) et d'un Réseau Possibiliste (RP) : Le premier RPMH consiste à structurer les documents retrouvés en zones denses de pages Web thématiquement liées les unes aux autres. Le second RPMH consiste à ne pas chercher seulement le mot-clé dans les pages Web mais aussi les substantifs qui lui sont sémantiquement proches. Les Réseaux Possibilistes combinent les deux RPMH afin d'organiser les documents recherchés selon les préférences de l'utilisateur.

- Dans le même cadre nous citons la plateforme **SWAPP** (Search based Web AdaPtive Platform). C'est la mise en place des systèmes interactifs auto-adaptatifs par systèmes multi-agents auto-organisateurs appliqués à la personnalisation de l'accès à l'information [Lemouzy, 2011]. Ce système utilise l'approche de AMAS (Adaptive Multi-Agent System) pour proposer une évaluation adaptative et personnalisée

du feedback implicite de l'utilisateur en utilisant l'UIM (User Interest Manager) ainsi que la construction adaptative de son profil à base de l'UPM (User Profile Manager) en utilisant des documents textuels représentant ses intérêts.

Chacun des systèmes indiqués ci-dessus, propose un modèle multi-agent pour un système de recherche d'information spécifique. Ainsi, ils sont conçus pour une utilisation et des ressources spécifiques et les agents et leurs interactions sont définies en fonction des composants des systèmes d'information. En outre, les premiers systèmes ont pour but de remplacer chaque étape du processus de la RI par un agent. L'inconvénient de ces approches se manifeste dans le fait qu'elles ne profitent pas et n'exploitent pas les capacités d'un agent intelligent puisqu'il ne fait que exécuter un processus. Une autre limite concernant l'utilisation des IPA est que ces approches utilisent un seul agent assistant personnel qui a pour rôle de collecter les informations sur les activités et le comportement de l'utilisateur, et qui réagit rarement avec l'environnement de l'utilisateur qui peut en soit être une source de personnalisation.

L'approche que nous avons suivi dans nos travaux adopte une approche stigmergique de la coordination entre agents qui à notre connaissance n'a pas été utilisée dans le cadre d'une recherche documentaire personnalisée. La section suivante présente ce principe d'approche stigmergique dans d'autres cas de développement de systèmes multi-agents.

3.3 Approche stigmergique des systèmes multi-agents

La stigmergie [Grassé, 1959] est généralement présentée comme un mécanisme social de coordination basée sur l'interaction par des modifications locales sur un environnement

partagé. Un exemple est le dépôt de phéromones par les fourmis, ou du mouvement des particules de bois par les termites [Camazine *et al.* , 2002].

"La stigmergie est un mécanisme de coordination indirecte entre agents ou actions. Le principe est qu'une trace laissée par une action dans l'environnement stimule l'accomplissement de l'action suivante, que ce soit par le même agent ou un agent différent. De cette façon, les actions suivantes tendent à se et renforcer et bâtir sur l'existant, ce qui conduit à l'émergence spontanée d'une activité d'apparence cohérente et systématique. La stigmergie est une forme d'auto-organisation. Elle produit des structures complexes sans avoir besoin de plan, de contrôle ou même de communication directe entre les agents." [wikipédia, 2016]

Les recherches sur la stigmergie dans les SMA a conduit à plusieurs travaux (Parunak et al. [Van Dyke Parunak *et al.* , 2002] et Valckenears et al. [Valckenaers *et al.* , 2001] parmi de nombreux autres), mais présentant deux biais principaux : les modèles d'agent sont souvent très simples, et le modèle de l'environnement est souvent tout à fait élémentaire [Gardelli *et al.* , 2008].

Basée sur l'adoption d'artefacts comme fondement pour la modélisation de l'environnement SMA, Ricci et al. [Ricci *et al.* , 2006b] définit la stigmergie cognitive comme la généralisation de la stigmergie pour un mécanisme de coordination pour les sociétés d'agents cognitifs. La stigmergie cognitive repose sur l'utilisation des artefacts comme des outils peuplant et structurant l'environnement de travail des agents, et que les agents perçoivent, partagent et utilisent rationnellement pour leurs objectifs individuels. Cette

approche étend le concept d'artefact initialement introduit pour assurer une coordination via l'environnement [Omicini *et al.* , 2004].

3.3.1 Le modèle A&A : Agent et Artefact

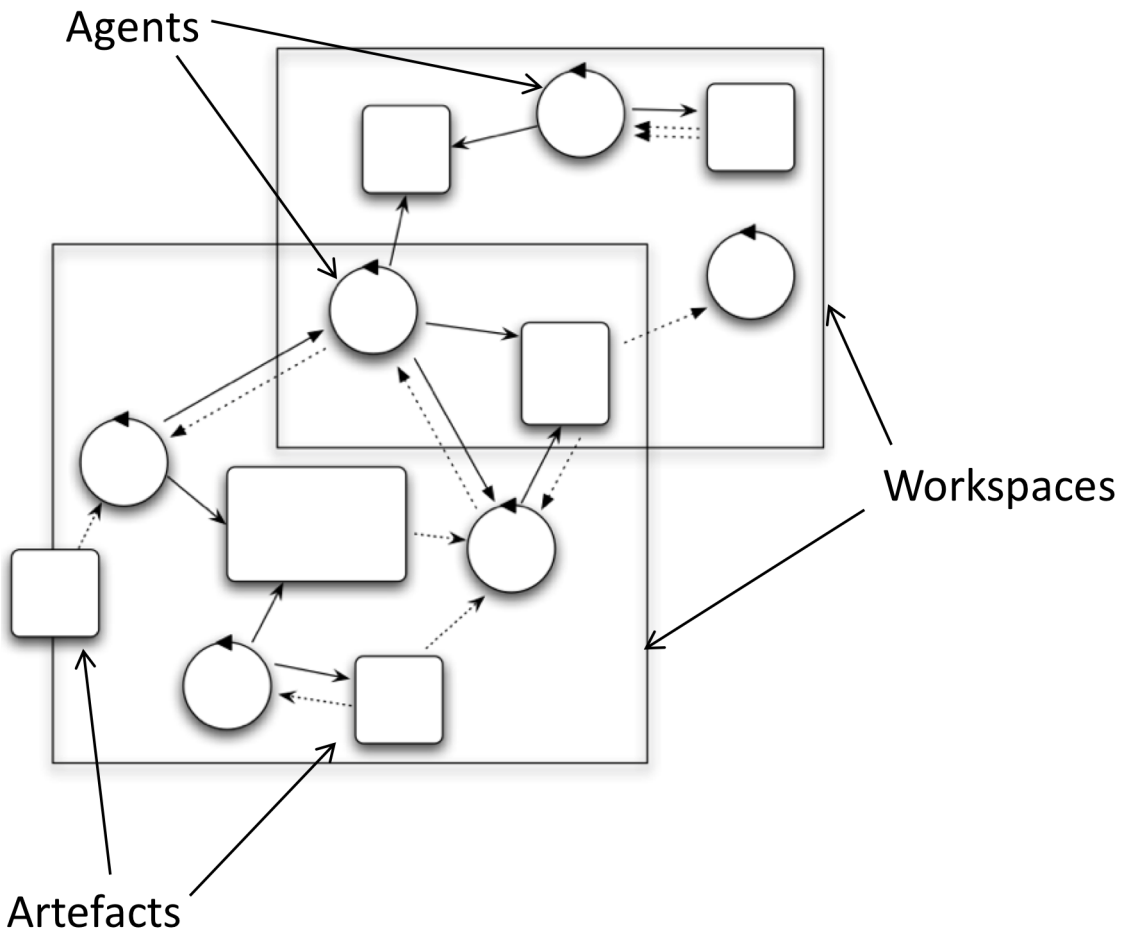

FIGURE 3.2 – Les éléments de base d'un modèle Agent & Artefact

Le méta-modèle A&A est un cadre conceptuel caractérisé par trois abstractions [Omicini *et al.* , 2008] :

- Agents, pour représenter des composants pro-actifs du système, encapsulant l'exé-

cution autonome d'une sorte d'activités à l'intérieur d'une sorte d'environnement ;

- Artefacts, pour représenter des composants passifs des systèmes tels que les ressources et les médias qui sont intentionnellement construites, partagées, manipulées et utilisées par les agents pour soutenir leurs activités, soit en coopération ou en compétition ;

- Workspaces, comme des conteneurs conceptuels des agents et des artefacts, utiles pour définir la topologie de l'environnement et fournir un moyen de définir une notion de localité.

3.3.2 Les agents dans le A&A méta-modèle

Omicini et al.[Omicini *et al.* , 2008] définissent un agent dans le méta-modèle A&A comme suit :

Définition 2 (A&A Agent [Omicini *et al.* , 2008])**.**

"An A&A agent is an autonomous computational entity.

genus *Agents are computational entities*

differentia *Agents are autonomous, in that they encapsulate control along with a criterion to govern it."*

Un agent est caractérisé dans un méta-modèle A&A essentiellement par son *autonomie*. Cette autonomie est représentée selon plusieurs points de vues.
Premièrement, du point de vue calcul, l'autonomie signifie que les agents encapsulent le contrôle. Dans le même contexte, un agent ne possède pas une interface pour qu'il soit utilisé, invoqué ou contrôlé. Pour être autonome, un agent doit alors être auto-controlé,

auto-organisé et auto-déterminé en possédant des buts à atteindre et des tâches à réaliser d'une manière indépendante.

Deuxièmement, pour être autonome, un agent doit faire partie d'une société. En effet, nous pouvons parler de l'autonomie que lorsqu'un individu fait partie d'une société puisque l'autonomie n'a pas de sens pour un individu isolé. Ce qui est au coeur de la définition d'un agent et sa distinction par rapport a un simple programme.

Finalement, d'un point de vue littéraire, le terme "agent" du Latin "agens" signifie "celui qui agit". Les agents sont alors des entités actives. Ils sont pro-actifs puisqu'ils encapsulent du contrôle et sont auto-gouvernés. De ce fait, un agent agit pour changer. Vu qu'un agent fait partie d'une société d'agents, il possède un environnement [Viroli *et al.* , 2007; Weyns *et al.* , 2007] sur lequel il peut agir et faire des changement.

D'autres caractéristiques sont attribuées à un agent dans un A&A méta-modèle. Par exemple un agent est intelligent, mobile, peut apprendre, etc.

3.3.3 Les artefacts dans le méta-modèle A&A

Après la définition des agents, nous présentons ici la définition des artefacts proposée par Omicini et al. [Omicini *et al.* , 2008] :

Définition 3 (A&A Artefact [Omicini *et al.* , 2008])**.**

"An A&A artifact is a computational entity aimed at the use by A&A agents.

genus *Artifacts are computational entities*

differentia *Artifacts are aimed to be used by agents"*

La notion d'artefact

Les artefacts représentent tout type de ressource ou d'outil que les agents peuvent créer, partager, utiliser, manipuler de manière dynamique. Ils existe différentes sortes et types d'artefacts avec éventuellement la possibilité de créer plusieurs instances pour chaque type d'artefact.

Les artefacts sont passifs conçus pour encapsuler une sorte de fonction qui est le but de la création de cet artefact.

Ainsi, la fonctionnalité d'un artefact est structurée sous la forme d'opérations.

La notion d'utilisation d'artefact

Contrairement à interaction basée sur la communication (entre agents), les agents ne communiquent pas avec les artefacts mais leur interaction est basée sur une notion d'usage. En effet, les agents utilisent des artefacts pour exploiter leur fonctionnalités. Un agent peut activer une fonctionnalité d'un artefact au moyen d'*opérations*. Ces opérations changent l'état d'un artefact et produisent des effets sur l'environnement. L'ensemble des opérations fournies par un artefact sont représentées par son *interface d'utilisation*.

L'interface d'utilisation est un ensemble de contrôleurs qui permettent à l'agent de déclencher et contrôler l'exécution de l'opération par l'artefact.

Les artefacts possèdent aussi des états et des évènements observables. En effet, ces états et évènements peuvent être perçu par les agents qui observent ou utilises ces artefacts.

La notion de manuel

Chaque artefact est équipé d'un manuel. C'est la description formelle de ses fonctionnalités et leurs modalités d'utilisation. Le contenu d'un manuel peut se résumer en deux questions :

- Pourquoi utiliser l'artefact ? : la réponse à cette question décrit le fonctionnement de l'artefact.

- Comment utiliser l'artefact ? : La réponse à cette question décrit le mode d'emploi des opérations de l'artefact.

Le manuel permet l'utilisation cognitive d'artefacts. En effet, la description et le mode d'emploi permettent la découverte dynamique, la sélection et l'apprentissage des artefacts par les agents cognitifs. Cette découverte renforce la notion d'ouverture d'un système à base d'artefact.

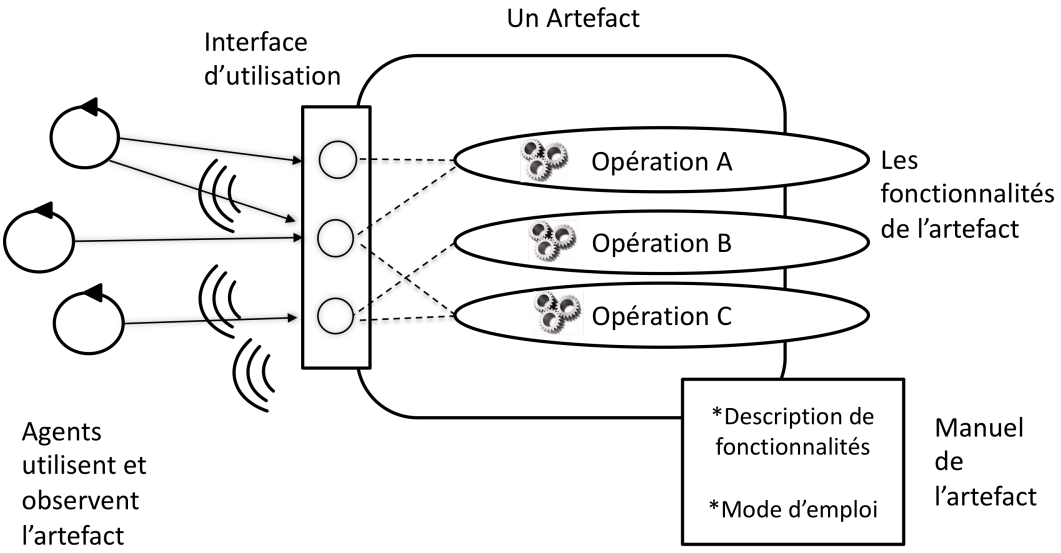

FIGURE 3.3 – Une abstraction d'un artefact

Exemples

Plusieurs types d'artefacts ont été définis pour remplir des fonctions génériques fréquemment rencontrées dans le développement de systèmes multi-agents, tels que l'artefact de coordination [Omicini *et al.* , 2004], l'artefact de réputation [Hübner *et al.* , 2009] ou l'artefact cognitif [Omicini *et al.* , 2008].

3.3.4 Les systèmes multi-agent à base d'A&A

Un système multi-agent selon le modèle A&A est défini comme étant composé d'agents et d'artefacts. Il est défini par Omicini et al. [Omicini *et al.* , 2008] comme suit :

Définition 4 (A&A MAS [Omicini *et al.* , 2008])**.**

"An A&A MAS is a computational system made of agent and artifacts

genus *MAS are computational systems*

differentia *MAS basic components are agents and artifacts"*

Dans un système multi-agent à base d'A&A nous pouvons trouver quatre sortes d'interactions possibles entre les agents et les artefacts à savoir :

- **Communication** : Les agents "parlent" avec des agents. Les agents dans un SMA communiquent soit directement en échangeant des messages en utilisant des ACL(Agent Communication Language) ou indirectement via leur environnement en agissant sur les artefacts.
- **Opération** : Les agents utilisent des artefacts : les agents dynamiquement créent, partagent et exécutent les opérations proposées par les artefacts pour réaliser leurs activités individuelles ou collectives.

- **Composition** : Les artefacts sont liés à des artefacts : une tâche peut être divisée sur plusieurs opérations pour être réaliser. Cette tâche est alors modélisé par plusieurs artefacts qui sont lies.

- **Présentation** : Les artefacts se manifestent aux agents : grâce à leurs propriétés observables les artefacts émettent un signal aux agents pour leurs informer de l'accomplissment d'une opération par exemple.

Tableau 3.1 – Comparaison entre les agents et les artefacts dans un modèle A&A

Agent	Artefact
Proactif	Passif
Orienté objectif / tâche / activité	Axé sur la fonction
Encapsulation du contrôle	Pas d'encapsulation du contrôle
Plusieurs degrés d'autonomie	Pas d'autonomie

Malgré son apparition relativement récente, la notion d'artefact a été appliquée dans différents domaines utilisant des SMAs, spécifiquement dans le domaine de génie logiciel orienté agent (AOSE : agent-oriented software engineering) [Bernon *et al.* , 2005], le domaine de la simulation à base de système multi-agent (MABS :multi-agent based simulation) [Montagna *et al.* , 2008], le domaine des systèmes à bas d'agent auto-organisé (agent-based SOS : self-organising systems) [Ricci *et al.* , 2007] le domaine des langages et infrastructure SMA [Ricci *et al.* , 2006a] ainsi que dans autres domaines de recherches à base de systèmes multi-agents.

3.4 Conclusion

Nous avons présenté dans ce chapitre les travaux existants dans le domaine des agents autonomes et des systèmes multi-agents pour réaliser une tâche d'assistance aux utilisa-

teurs. Nous avons cité les premiers travaux s'intéressant à l'assistance pour la recherche d'information et la navigation avant d'aborder des approches plus récentes intégrant dans un agent personnel intelligent (IPA) des mécanismes de personnalisation. Nous avons mis en avant le fait que ces travaux n'exploitent pas les capacités d'un agent intelligent et se limite la plupart du temps à implémenter au sein d'un seul agent une méthode de personnalisation, parfois intégré à un système où d'autres agents implémentent des tâches du processus de recherche d'information. Une autre limite est que ces approches utilisent un seul agent assistant personnel qui a pour rôle de collecter les informations sur les activités et le comportement de l'utilisateur, et qui réagit rarement avec l'environnement de l'utilisateur qui peut en soit être une source de personnalisation. Nous avons détaillé par la suite la notion de stigmergie ainsi que le modèle A&A (Agents et Artefacts) qui est un nouveau paradigme pour la modélisation et l'ingénierie des systèmes logiciels. Il est composé par des agents des artefacts et des workspaces en respectant l'approche de coordination indirecte offerte par la stigmergie.

Nous détaillons dans la partie suivante les contributions théoriques de cette thèse à savoir la spécification des algorithmes d'assistance à la recherche documentaire que nous proposons ainsi que leur utilisations dans un premier chapitre suivi par la présentation, la modélisation et et les détails de la plateforme à base d'agent et d'artefact pour la recherche documentaire dans un deuxième chapitre.

Deuxième partie

Contributions théoriques

CHAPITRE 4

ALGORITHMES D'ASSISTANCE À LA RECHERCHE DOCUMENTAIRE

Sommaire

4.1 Composants d'un corpus documentaire **82**

4.1.1 Les documents du corpus 82

4.1.2 Une terminologie 83

4.1.3 L'index 84

4.2 Navigation dans le corpus **86**

4.2.1 Navigation 86

4.2.2 Recherche 89

4.2.3 Interface 92

4.2.4 Document 92

4.3 Reformulation de requêtes **93**

4.3.1 Retour de pertinence et annotation 94

4.3.2 Profil 95

4.3.3 Reformulation du besoin informationnel 97

4.4 Recommandation communautaire **100**

4.5 Recommandation à base de filtrage collaboratif **101**

4.5.1 Regroupement des utilisateurs similaires 102

4.5.2 Filtrage Collaboratif 106

4.5.3 Exemple 108

4.6 Conclusion **109**

L'accès aux documents d'un corpus numérique fermé soulève des problèmes liés à la recherche d'information et à la navigation entre les documents. Certains de ces problèmes sont similaires à ceux rencontrés dans le domaine de la recherche d'information sur le web (par exemple le calcul de la pertinence de documents par rapport à une requête) mais d'autres sont spécifiques au fait que l'on s'intéresse à un corpus documentaire fermé concernant des types de documents, requêtes et utilisateurs restreints à un domaine spécifique que l'on peut représenter de manière plus fine que dans le cadre général du web. La recherche documentaire (RD) a lieu dans un corpus documentaire (CD) fermé qui indexe des documents présélectionnés à partir de sources d'information pertinentes par rapport à une thématique prédéfinie. Un corpus documentaire est un ensemble de documents ayant un sujet spécifique en commun. Un corpus est indexé en utilisant une terminologie de termes spécifique du domaine. Un index du corpus est important et nécessaire pour pouvoir interroger le corpus et représenter les termes les plus signifiants dans un document. Dans un corpus fermé, les utilisateurs possèdent des profils spécifiques à la thématique ciblé des documents de ce corpus.

Dans ce cadre, il est possible d'identifier et de chercher à reconnaître des pratiques ou usages spécifiques à l'accès à un corpus documentaire donnée et d'adapter en conséquence le comportement de la plateforme logicielle.

Dans ce chapitre nous définissons l'ensemble de constituants d'un corpus documentaire. Nous proposons ensuite un modèle de représentation de la navigation dans un corpus de documents. Les facteurs à prendre en compte dans cette présentation sont multiples : le contenu d'un corpus documentaire, un profil propre à l'utilisateur, un cas d'usage reconnu pour la navigation, une proximité sémantique entre documents ou un retour de pertinence construit sur les annotations effectués par les utilisateurs.

Ce chapitre présente ainsi, les bases du modèle de navigation utilisé par la suite pour l'assistance de l'utilisateur pendant sa navigation dans le corpus documentaire.

Ce chapitre est organisé comme suit : La section 4.1 présente notre corpus documentaire composé d'une base de donnée contenant des documents, une terminologie représentant les mots clés pertinents ainsi qu'un index. La section 4.2 présente une formalisation du processus de navigation au sein du corpus documentaire. La section 4.3 présente les algorithmes d'assistance de l'utilisateur pendant sa navigation dans le corpus documentaire. Nous détaillons le processus d'indexation ainsi que le processus de recherche. Ensuite, les étapes à suivre pour la reformulation de la requête sont détaillés. La section 4.4 détaille la recommandation communautaire à base de profils proches. La section 4.5 présente des algorithmes qui permettent la recommandation collaborative plus sophistiqué de documents à l'utilisateur en utilisant l'algorithme X-means pour classifier les utilisateurs et le filtrage collaboratif pour prédire la pertinence d'un document pour un utilisateur.

4.1 Composants d'un corpus documentaire

4.1.1 Les documents du corpus

La recherche documentaire (RD) a lieu dans un corpus documentaire (CD) fermé qui indexe des documents présélectionnés à partir de sources d'information pertinentes par rapport à la thématique du corpus. Un corpus documentaire est un ensemble de documents ayant un sujet spécifique en commun. Un corpus est indexé en utilisant une terminologie de termes spécifique du domaine. Un index du corpus est important et nécessaire pour pouvoir interroger le corpus et représenté les termes les plus signifiants

dans un document.

Les documents constituant un corpus documentaire fermé doivent traiter la même thématique. Par exemple, nous avons comme cadre applicatif dans cette thèse le corpus documentaire constitué par l'Institut International du Droit du Transport (IDIT [1]). Ce corpus contient autour de 50 000 références incluant des documents classés en quatre groupes : jurisprudence, articles, règlementation, fonds documentaires. Ces documents sont indexés à l'aide d'une terminologie du domaine et par une analyse plein texte des documents. La personnalisation en fonction des utilisateurs et des navigations est très intéressante dans ce type de corpus. D'une part les utilisateurs n'ont pas les mêmes besoins pour une même requête du fait de niveaux d'expertises différents. D'autre part, des navigations "typiques" ont lieu, sans être précisément formalisées, selon des usages fréquents comme des études comparatives de cas, de jurisprudence, des recherches précises, ...

Définition 1 (Corpus documentaire)**.**

Nous notons l'ensemble des documents du corpus $\mathscr{D} = \{d_1, d_2, ..., d_{|\mathscr{D}|}\}$.

4.1.2 Une terminologie

La terminologie du domaine dans un corpus documentaire est la représentation du contenu des documents formant ce corpus. Une terminologie est considérée comme la clé pour accéder au contenu d'un corpus. Le but de la constitution d'une terminologie de domaine est la normalisation du vocabulaire utilisé dans un domaine bien défini. Elle facilite l'élaboration de vocabulaire de référence dans un domaine spécialisé dans le but

[1] http://www.idit.asso.fr/

d'être utilisé par des systèmes d'indexation.

Définition 2 (Terminologie)**.**

Soit $\mathcal{T}$ un ensemble de termes distincts propres au corpus. $\mathcal{T}$ est la terminologie qui regroupe l'ensemble des termes utilisés pour indexer les documents du corpus. Un document d_i du corpus, est alors associé à une séquence de termes. Un sous-ensemble $\mathcal{K} \subset \mathcal{T}$ représente l'ensemble de mots-clés utilisés dans les requêtes sur le corpus.

4.1.3 L'index

Dans un corpus documentaire nous trouvons un index qui est réalisé selon deux manières :

- Une indexation automatique : Nous utilisons pour notre plateforme le moteur *Lucene* [2] qui est une application open source gratuite pour la recherche plein texte et l'analyse de contenu textuel. Le corpus est alors indexé par le moteur d'indexation de Lucene pour nous permettre de réaliser des recherches dans ce corpus.

- Une indexation semi-automatique : L'indexation semi-automatique ou humaine est réalisée par des experts du domaine du corpus qui jugent quels termes sont le plus pertinents pour décrire un document parmi plusieurs termes proposés automatiquement. En effet, l'expert commence par la lecture du document, il introduit ensuite l'ensemble de termes, les plus représentatifs du document, qu'il choisit en

[2] http ://lucene.apache.org/core/

se basant sur la terminologie $\mathcal{T}$. Ces termes sont enregistrés automatiquement comme index de ce document en les identifiant dans la terminologie et les associant à chaque document en précisant sa position dans le document pour nous permettre de les extraire par la suite.

Le processus d'indexation automatique est composé de quatre classes illustrés par le diagramme 4.1 suivant :

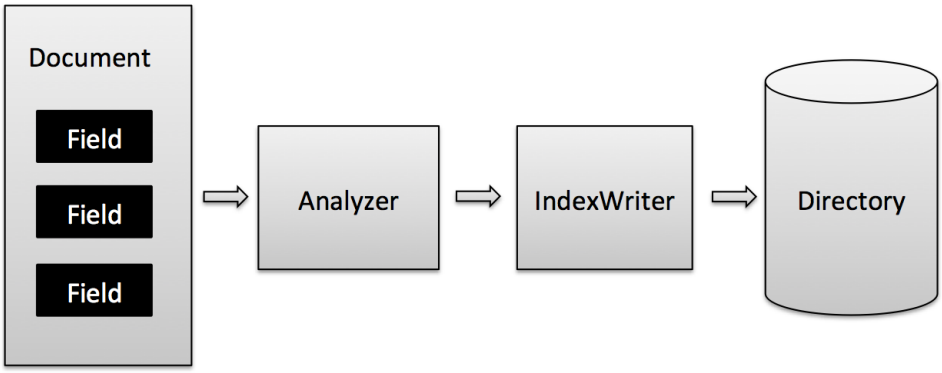

FIGURE 4.1 – Processus d'indexation dans lucene

Chaque étape du processus d'indexation représente une classe :

- **IndexWriter** est le composant le plus important et le noyau du processus d'indexation. Cette classe agit comme un élément de base qui crée / met à jour des index pendant le processus d'indexation.
- **Directory** : l'annuaire représente l'emplacement de stockage des index.
- **Analyzer** est responsable d'analyser un document et obtenir les jetons / mots du texte qui doivent être indexé. Sans l'analyse faite, IndexWriter ne peut pas créer un index.

- ***Document*** représente un document virtuel avec les champs où le terrain est un objet qui peut contenir le contenu du document physique, ses méta données et ainsi de suite. la classe Analyzer peut comprendre ce document virtuel uniquement. Il est utile de faire la différence entre les documents physique du corpus et la représentation virtuelle *document*.

- **Field** : Le champ est l'unité la plus basse ou le point du processus d'indexation de départ. Chaque champ correspond à une donnée qui est interrogée ou récupérée à partir de l'index lors de la recherche.

4.2 Navigation dans le corpus

L'accès à l'information s'apparente plus à une navigation au sein d'un corpus fermé plutôt qu'à une recherche ponctuelle d'information, par exemple sur le web, et la personnalisation doit tenir compte de plusieurs aspects (profil de l'utilisateur, historique de la session, retour de pertinence).

Une *navigation* représente une session de recherche où l'utilisateur exprime son besoin informationnel à l'*interface* de la plateforme et navigue dans les *documents* susceptibles d'être pertinents résultats d'une *recherche*. Son besoin d'information est affiné par son *profil* utilisateur.

4.2.1 Navigation

Une navigation encapsule un besoin informationnel d'un utilisateur. Soit $\mathcal{N} = \{\mathrm{N}_1, ..., \mathrm{N}_{|\mathcal{N}|}\}$ l'ensemble de toutes les navigations, pour chaque navigation N_i, un utilisateur u_{N_i} a

une requête courante K_{N_i} composée de plusieurs termes $k^j_{N_i}$.

$$\mathrm{N}_i = \{\mathrm{K}_{N_i}, \mathrm{u}_{N_i}\} \tag{4.1}$$

Avec

$$\mathrm{K}_{N_i} = \left\{\mathrm{k}^1_{N_i}, ..., \mathrm{k}^{|\mathrm{K}_{N_i}|}_{N_i}\right\} avec\ \mathrm{k}^i_{N_i} \in \mathscr{K}. \tag{4.2}$$

Un utilisateur u commence une navigation en saisissant une requête $\mathrm{K}_u = \left\{\mathrm{k}^1_u, ..., \mathrm{k}^{|\mathscr{K}_\sqcap|}_u\right\}$ composée d'un ensemble de mots-clés. Si l'utilisateur n'avait pas de navigation en cours, une nouvelle session de navigation est créée avec cette requête.

S'il existe déjà des navigations pour cet utilisateur, Il faut déterminer s'il s'agit du début d'une nouvelle navigation ou si l'insertion de nouveaux termes correspond à la continuité d'une session en cours. En effet il est difficile de déterminer la limitation d'une session. Quand changer de session et quand continuer dans la session courante. Plusieurs travaux s'intéressent à cette problématique et choisissent par exemple un intervalle de temps comme critère de limitation, ou à chaque connexion de l'utilisateur une nouvelle session est créée. Pour résoudre cette problématique, nous avons opté pour l'utilisation d'une mesure de similarité $Sim(\mathrm{K}_u, \mathrm{K}_{N_i})$ entre les termes de la requête et ceux de toutes les navigations précédentes de l'utilisateur. Nous utilisons l'indice et la distance de *Jaccard* [Jaccard, 1902] pour calculer cette similarité. L'indice de Jaccard dit aussi coefficient de Jaccard est le rapport entre le cardinal de l'intersection des ensembles considérés et le cardinal de l'union de ces ensembles. Il permet d'évaluer la similarités entre les ensembles. Dans notre corpus les ensembles à comparer sont la requête courante d'un coté et les termes des requêtes précédentes. De telle sorte que nous exploitons les recherches précédentes de l'utilisateur et lui permettre de continuer

ses sessions non achevés.

$$Sim(\mathrm{K}_u, \mathrm{K}_{N_i}) = J(\mathrm{K}_u, \mathrm{K}_{N_i}) = \frac{\left|\mathrm{K}_u \cap \mathrm{K}_{N_i}\right|}{\left|\mathrm{K}_u \cup \mathrm{K}_{N_i}\right|} \tag{4.3}$$

La distance de Jaccard mesure la dissimilarité entre les ensembles. Elle consiste simplement à soustraire l'indice de Jaccard à 1.

$$J_\delta(\mathrm{K}_u, \mathrm{K}_{N_i}) = 1 - J(\mathrm{K}_u, \mathrm{K}_{N_i}) = \frac{\left|\mathrm{K}_u \cup \mathrm{K}_{N_i}\right| - \left|\mathrm{K}_u \cap \mathrm{K}_{N_i}\right|}{\left|\mathrm{K}_u \cup \mathrm{K}_{N_i}\right|} \tag{4.4}$$

Pour déterminer si la requête courante est la suite d'une navigation existante, nous calculons la distance entre l'ensemble des termes de cette requête est les termes de chacune des navigations précédentes. Le calcul de similarité est effectué par l'algorithme 1.

Algorithm 1 Similarité de navigations en utilisant la distance de Jaccard

```
1: ∃N_i ∈ 𝒩/u = u_{N_i}
2: for all N_i do
3:    if J_δ(K_u, K_{N_i}) > θ then
4:       K_{N_i} = K_{N_i} ∪ K_u
5:    else
6:       N_{|𝒩|+1} = (K_u, u)
7:    end if
8: end for
```

Si la similarité est en dessous d'un seuil θ pour toutes les navigations passées, Nous considérons qu'une nouvelle navigation commence, sinon nous considérons être dans la continuité des plus similaires.

$$\exists \mathrm{N}_i \in \mathcal{N}/\mathrm{u} = \mathrm{u}_{N_i}, Sim(\mathrm{K}_u, \mathrm{K}_{N_i}) > \theta, \tag{4.5}$$

$$\mathrm{K}_{N_i} = \mathrm{K}_{N_i} \cup \mathrm{K}_u \tag{4.6}$$

Sinon, une nouvelle navigation commence :

$$\nexists \mathrm{N}_i \in \mathcal{N} / \mathrm{u} = \mathrm{u}_{N_i}, Sim(\mathrm{K}_u, \mathrm{K}_{N_i}) < \theta, \tag{4.7}$$

$$\mathrm{N}_{|\mathcal{N}|+1} = (\mathrm{K}_u, u) \tag{4.8}$$

4.2.2 Recherche

Nous utilisons pour notre plateforme le moteur *Lucene* comme nous l'avons précédemment précisé. L'objectif de la recherche est de prendre en entrée un ensemble de termes issus d'une requête et de fournir les références de documents jugés pertinents en sortie.

$$Recherche : P(\mathrm{K}) \mapsto P(\mathrm{D}) \tag{4.9}$$

Les classes les plus importantes pour réaliser une recherche avec lucene sont les suivantes :

- **IndexSearcher** :C'est la classe la plus importante. Elle est le noyau du processus de recherche. Elle recherche dans les index créés après processus d'indexation.
- **Term** : Cette classe est la plus petite unité de recherche. c'est l'équivalent au champ *field* dans le processus d'indexation.
- **Query** : C'est une classe abstraite et contient diverses méthodes utilitaires. Elle représente le parent de tous les types de requêtes que lucene utilise au cours du processus de recherche.
- **TermQuery** : C'est l'objet de la requête la plus couramment utilisée. Elle est le fondement de nombreuses requêtes complexes que lucene peut utiliser.

- **TopDocs** : Cette classe pointe vers les résultats de la recherche. Ce sont N top documents qui correspondent aux critères de la recherche. TopDocs est un simple conteneur de pointeurs pour pointer vers des documents qui sont le résultat de la recherche.

Le processus de recherche commence par la création de l'annuaire qui contient les indexes en utilisant le processus d'indexation. *IndexSearcher* récupère cet annuaire et le lit en utilisant *IndexReader*. Ensuite, une requête est crée avec la classe *term*. Une recherche est et effectuée en utilisant *IndexSearcher* en passant la requête au *Searcher*. Finalement, *IndexSearcher* renvoie un objet *TopDocs* qui contient les détails de la recherche ainsi que les identifiants des document ID qui représente le résultat de l'opération de recherche.

Une étape très importante précède la recherche est la manipulation de la requête. En effet, la requête doit être transformée en un format compréhensible par lucene. Il s'agit d'utiliser le même analyseur utilisé pendant l'indexation pour analyser les termes de la requête et la transformer.

Ainsi nous avons d'un coté la représentation du corpus sous forme d'index et la représentation de la requête qui respecte les même format que l'index.
Le processus de recherche est composé de trois étapes illustrés par le diagramme 4.2 suivant :

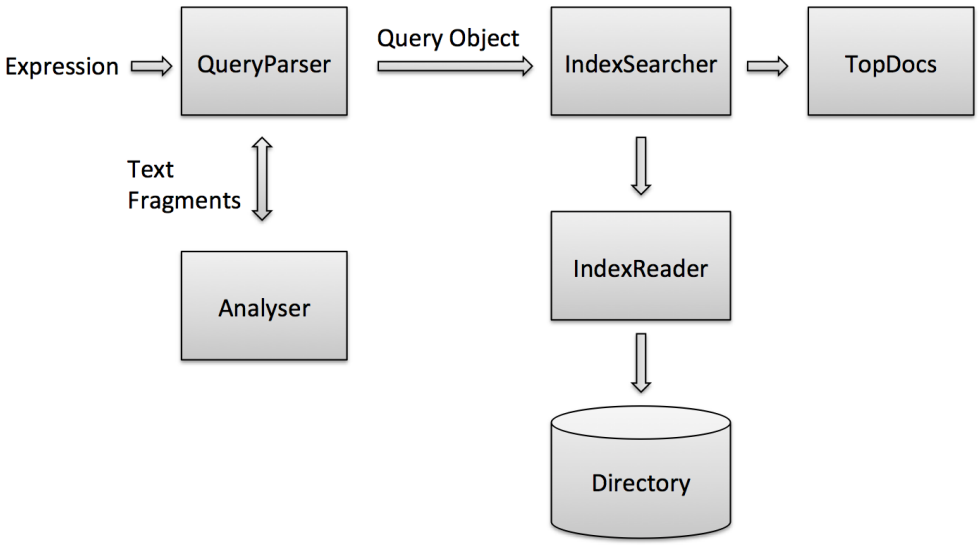

FIGURE 4.2 – Processus de recherche dans lucene

Lors de la première navigation d'un utilisateur, celui-ci introduit sa requête sous la forme d'un ensemble de termes du langage naturel. Cette requête est analysée pour en extraire les termes significatifs répertoriés dans la terminologie du domaine. Ensuite la base d'index est parcourue et les documents pertinents sont sélectionnés. Ces documents sont triés par ordre croissant de pertinence et affichés à l'utilisateur. Ce processus est une recherche par mot clés réalisé lors de la première requête de l'utilisateur.

Pour cette première recherche, Les actions de l'utilisateur sur le résultat présenté sont tracés pour enrichir les données du Profil. Les termes de la requête et les documents consultés alimentent le profil créé pour la session de navigation courante.

Un deuxième type de recherche est réalisé par la suite. C'est la recherche avec reformulation basée sur le profil de l'utilisateur. Elle est utilisée après reformulation de la requête en se basant sur les préférence de l'utilisateur à savoir les documents qu'il

juge pertinents.

4.2.3 Interface

Le rôle de l'interface est de gérer les interactions avec l'utilisateur. Celui-ci utilise l'interface pour exprimer son besoin informationnel. Il consulte ensuite les résultats de la recherche en réponse à sa requête. L'interface permet aussi à l'utilisateur de naviguer entre les documents et d'introduire son jugement de pertinence par rapport à un document consulté. Elle permet aussi à l'utilisateur d'effectuer des annotations sur les documents qu'il juge pertinents. Nous détaillerons par la suite la notion d'annotation.

Pendant sa navigation entre les documents, l'utilisateur peut modifier sa requête via l'interface en essayant de la réduire ou l'élargir ou la reformuler dans le but est de bien cerner son besoin informationnel qui évolue pendant la navigation et après la consultation des documents susceptibles d'être pertinents pour lui. En effet, un utilisateur peut avoir un besoin en information mais n'arrive pas à trouver les termes exacts qui décrivent son besoin informationnel. Au fur et à mesure de sa navigation dans les documents retournés comme résultats de recherche il raffine son besoin informationnel en ajoutant des termes qui décrivent exactement ce qu'il cherche.

4.2.4 Document

Un document est identifié par une référence aux documents jugés pertinents pour une navigation donnée N_i. Cet ensemble est amené à évoluer en cours de navigation en fonction des actions de l'utilisateur et des résultats de recherche.

L'ensemble de documents D_i correspondant à la navigation N_i est l'union des documents résultant d'une recherche documentaire $D_{i,RD}$:

$$D_i = \{D_{i,RD}\} \tag{4.10}$$

Avec

$$D_{i,RI} = \left\{d^1_{i,RD}, ..., d^{|D_{i,RD}|}_{i,RD}\right\}, \tag{4.11}$$

4.3 Reformulation de requêtes

La plateforme d'accès au corpus documentaire, de part la nature des documents et des utilisateurs visés, doit accorder une place prépondérante à la personnalisation et au mode d'accès. Les utilisateurs visés ont des niveaux d'expertise variables allant de l'expert au citoyen non averti en passant par le professionnel du domaine.

L'objectif de notre plateforme est de permettre à des utilisateurs de naviguer dans un corpus fermé en visualisant séquentiellement différents documents et en affinant ou adaptant leur requête au fur et à mesure de leur session de navigation. Pour cela, il est nécessaire dans un premier temps d'utiliser des mécanismes classiques de recherche d'information pour sélectionner des documents pertinents en fonction d'une requête. Nous proposons de compléter ces outils par des mécanismes de personnalisation et d'adaptation pour faire évoluer la représentation du besoin. Cette évolution est dynamique car réalisée pendant une navigation en fonction du profil des utilisateurs, de leurs actions et des navigations précédemment observées. Pour cette raison, nous utilisons le terme de *besoin*

informationnel pour désigner l'objectif global que la plateforme doit satisfaire plutôt que celui de *requête*, propre à une recherche ponctuelle d'information.

Dans cette section, nous décrivons les différents types d'assistance possible à l'utilisateur pendant ses navigations dans un corpus documentaire.

4.3.1 Retour de pertinence et annotation

Lorsque des documents sont proposés à l'utilisateur après exécution d'une requête, l'intérêt qu'il porte à chaque document est estimé par un retour de pertinence. Nous ne présupposons pas de la manière (explicite ou implicite) dont ce retour est capté. Il est représenté par une valeur numérique entre 0 (non pertinent) et 1 (parfaitement pertinent).

De plus, un utilisateur u_i a la possibilité d'annoter une ou plusieurs partie(s) $\S \subset D$ qu'il juge pertinente(s) dans un document jugé pertinent D_i^{REL}. Il y associe alors un commentaire appelé An. L'ensemble des annotations constitue l'ensemble A^D

Les annotations $\mathrm{A}_{N_i}^D$ d'un utilisateur impliqué dans une session de navigation N_i sont composées par son retour de pertinence $\mathrm{RF}_{N_i} \in \{0,1\}$, et ses commentaires An_{N_i}.

$$\mathrm{A}_{N_i}^D = \left(\mathrm{RF}_{N_i}, \mathrm{An}_{N_i}\right) \tag{4.12}$$

Avec

$$\mathrm{RF}_{N_i} \in \{0,1\} \tag{4.13}$$

et

$$\mathrm{An}_{N_i} = Message \tag{4.14}$$

Les annotations des documents de chaque utilisateur sont alors stockées. Ces modifications des documents annotés sont enregistrées et mises à jour. Ainsi, lors de l'affichage des résultats de recherche, les documents annotés auront le poids le plus important par rapport à d'autres documents pertinents à afficher.

Ces annotations peuvent être utilisées pendant la recommandation puisque les annotations contiennent le retour de pertinence de l'utilisateur. De plus, nous faisons l'hypothèse qu'un document annoté est implicitement pertinent.

4.3.2 Profil

Le profil P_i d'un utilisateur impliqué dans une session de navigation N_i est composé par sa requête initiale $K_{P_i}^{INIT} \subset \mathscr{K}$, les termes proposés en reformulation $T_{P_i}^{ACC} \subset \mathscr{T}$ (acceptés par l'utilisateur) et $T_{P_i}^{REJ} \subset \mathscr{T}$ (rejetés par l'utilisateur) et enfin les références des documents jugés pertinents par l'utilisateur.
Il est pertinent de préciser qu'un utilisateur peut avoir plusieurs profils. Chaque navigation possède un profil propre. Ce sont les différentes pratiques d'usage qu'un utilisateur peut avoir en utilisant la plate-forme.

$$P_i = \left(K_{P_i}^{INIT}, T_{P_i}^{ACC}, T_{P_i}^{REJ}, D_{P_i}^{REL}\right) \tag{4.15}$$

Avec

$$K_{P_i}^{INIT} = \left\{k_{P_i}^1, ..., k_{P_i}^n\right\} \tag{4.16}$$

$$T_{P_i}^{ACC} = \left\{t_{P_i}^{ACC,1}, ..., t_{P_i}^{ACC,p}\right\} \tag{4.17}$$

$$T_{P_i}^{REJ} = \left\{t_{P_i}^{REJ,1}, ..., t_{P_i}^{REJ,q}\right\} \tag{4.18}$$

$$\mathrm{D}_{P_i}^{REL} = \left\{ \mathrm{d}_{P_i}^{REL,1}, ..., \mathrm{d}_{P_i}^{REL,r} \right\} \tag{4.19}$$

et

$$\mathrm{K}_{P_i}^{INIT} \cap \mathrm{T}_{P_i}^{ACC} \cap \mathrm{T}_{P_i}^{REJ} = \emptyset \tag{4.20}$$

Au départ d'une navigation, le profil de l'utilisateur est initialisé avec les mots clés qui représentent la requête initiale de l'utilisateur.

$$\mathrm{P}_i = \left(\mathrm{K}_{N_i}, \emptyset, \emptyset, \emptyset \right) \tag{4.21}$$

Tout au long de la navigation de l'utilisateur, son profil est enrichi en rajoutant les documents qu'il consulte et juge pertinents. Dans notre système nous avons choisi le feedback explicite en demandant à utilisateur de donner un score de pertinence. Celcui-ci est représenté par une valeur numérique entre 0 (non pertinent) et 1 (parfaitement pertinent). L'ensemble des documents sélectionnés comme pertinents (c'est-à-dire situé au dessus d'un seuil proposé comme paramètre sur la plateforme) constitue l'ensemble $\mathrm{D}_{P_i}^{REL}$ qui est ajouté à P_i.

$$\mathrm{P}_i = \left(\mathrm{K}_{P_i}^{INIT}, \emptyset, \emptyset, \mathrm{D}_{P_i}^{REL} \right) \tag{4.22}$$

L'utilisateur peut aussi reformuler sa requête d'ou l'ajout des termes acceptés et termes rejetés dans le profil.

$$\mathrm{P}_i = \left(\mathrm{K}_{P_i}^{INIT}, \mathrm{T}_{P_i}^{ACC}, \mathrm{T}_{P_i}^{REJ}, \mathrm{D}_{P_i}^{REL} \right) \tag{4.23}$$

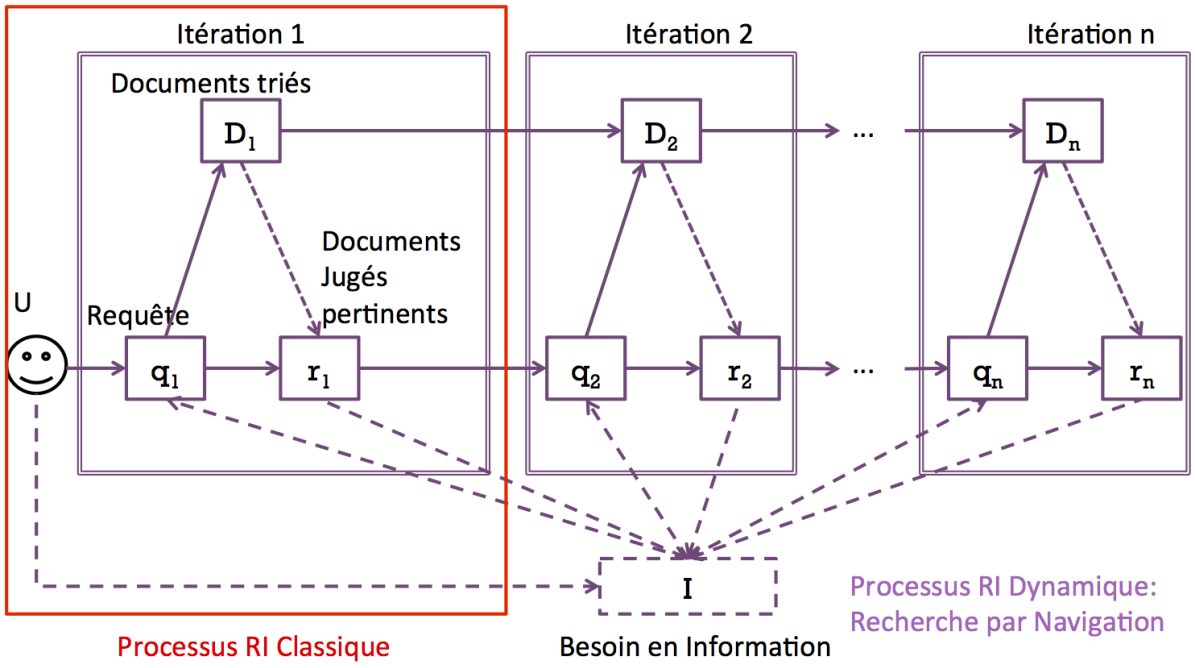

FIGURE 4.3 – L'evolution du besoin informationnel au cours d'une navigation dans le corpus

4.3.3 Reformulation du besoin informationnel

Rappelons qu'un utilisateur u commence une navigation en saisissant une requête $K_u = \left\{k_u^1, ..., k_u^{|K_u|}\right\}$ composée d'un ensemble de mots-clés. Si l'utilisateur n'avait pas de navigation en cours, une nouvelle session de navigation est crée avec cette requête. S'il existe déjà des navigations pour cet utilisateur, une distance est calculée entre le besoin courant de l'utilisateur et les navigation existantes. La reformulation de la requête est basé sur le déroulement de la navigation de l'utilisateur. Le profil de l'utilisateur contient initialement sa premiere requête.

$$P_i = \left(K_{N_i}, \emptyset, \emptyset, \emptyset\right) \tag{4.24}$$

Suite à un retour de pertinence de l'utilisateur. L'ensemble des documents sélectionnés comme pertinents constitue l'ensemble $D_{P_i}^{REL}$ qui est ajouté à P_i.

La proposition de reformulation se base sur les termes pertinents indexant les documents jugés pertinents qui sont obtenus par

$$GetIndex(\mathrm{D}_{P_i}^{REL}). \tag{4.25}$$

Cette fonction renvoie l'ensemble de termes indexant les document jugés pertinents par l'utilisateur. Pour la reformulation de sa requête, nous proposons à l'utilisateur l'ensemble $Proposition(\mathrm{N}_i)$. Cet ensemble est construit tel que :

$$\forall \mathrm{d} \in \mathrm{D}_{P_i}^{REL}, \forall \mathrm{t} \in GetIndex(d)/\mathrm{t} \notin \mathrm{K}_{P_i}^{INIT} \cup \mathrm{T}_{P_i}^{ACC} \cup \mathrm{T}_{P_i}^{REJ}, \tag{4.26}$$

$$\mathrm{t} \in Proposition(\mathrm{N}_i) \tag{4.27}$$

Pour ne proposer que les termes pertinents à l'utilisateur nous calculons la mesure Tf-idf pour ces termes. Pour un terme t et un document d, Tf(t,d) est la fréquence d'occurrence de ce terme t dans le document d. Idf est l'inverse du nombre de documents contenant ce terme.

Cet ensemble est proposé à l'utilisateur qui peut choisir de les accepter (ensemble T_{acc}) ou de les rejeter (ensemble T_{rej}). Le profil est mis à jour avec

$$\mathrm{T}_{P_i}^{ACC} = \mathrm{T}_{P_i}^{ACC} \cup \mathrm{T}_{acc} \tag{4.28}$$

$$\mathrm{T}_{P_i}^{REJ} = \mathrm{T}_{P_i}^{REJ} \cup \mathrm{T}_{rej} \tag{4.29}$$

Si $\mathrm{T}_{P_i}^{ACC}$ a été modifié, une nouvelle recherche en utilisant la nouvelle requête :

$$Recherche\left(\mathrm{K}_{P_i}^{INIT} \cup \mathrm{T}_{P_i}^{ACC}\right) \mapsto \mathrm{D}^{'} \tag{4.30}$$

puis met à jour l'ensemble des documents pertinents à base de la recherche d'information.

$$\mathrm{D}_i = \left(\mathrm{D}_{i,RI} \cup \mathrm{D}^{'}, \mathrm{D}_{i,REC}\right) \tag{4.31}$$

Ce processus de reformulation, ici décrit après le premier résultat d'une requête, sera réitéré à chaque fois que l'utilisateur sélectionnera des documents proposés comme étant pertinents (ceux-ci pouvant avoir été issus d'une requête déjà re-formulée).

Algorithm 2 Reformulation de requête

for all d_i in D_{rel} **do**
  for all t_j in $index(d_i)$ **do**
    $score(t_j) \leftarrow tf(t_j\ in\ d_i) * idf(t_j)$
    $ScoreTri[j] \leftarrow score(t_j)$
    $TermTri[j] \leftarrow t_j$
  end for
end for
for $k \leftarrow 0 to 10$ **do**
  $Proposition[k] \leftarrow TermTri[k]$
end for
for all t_m in $Proposition[]$ **do**
  if $t_m \notin K_{n_i} \cup T_{acc} \cup T_{rej}$ **then**
    $q' \leftarrow q \cup \{t_j\}$
  end if
end for
$q'' \leftarrow userFeedback(q')$
for all t_i in q' **do**
  if $t_i \notin K_{n_i} \cup q''$ **then**
    $T_{rej} \leftarrow T_{rej} \cup \{t_i\}$
  end if
  if $t_i \notin K_{n_i}\ and\ t_i \in q''$ **then**
    $T_{acc} \leftarrow T_{acc} \cup \{t_i\}$
    $K_{n_i} \leftarrow K_{n_i} \cup \{q''\}$
  end if
end for

L'algorithme 2 représente la reformulation d'une requête en se basant sur les termes indexant les documents jugés pertinents dans une navigation avec $score(t_j)$ est le poids d'un terme dans le document qu'il indexe ainsi que les autres documents jugés pertinents. Nous précisons que :

$$Tf = sqrt(freq) \tag{4.32}$$

$$Idf = 1 + Log\left(\frac{numDocs}{DocFreq + 1}\right) \tag{4.33}$$

En plus, $ScoreTri[j]$ est un tableau trié qui contient les poids des termes. et $TermTri[j]$ contient les termes trié selon leur poids le plus important.

4.4 Recommandation communautaire

La recommandation communautaire consiste à proposer des documents qui ont été jugés pertinents pour des profils de navigation proches. À la création d'un profil P_i, nous parcourons les profils existants Profils P_j pour comparer leurs termes initiaux et acceptés à ceux de la requête de P_i. Si tous les termes de P_i sont inclus dans ceux d'un profil P_j, les documents jugés pertinents pour P_j seront recommandés pour P_i. La construction de l'ensemble D_{AJ} représentant l'ensemble des documents à ajouter à ceux recommandés se fait comme suit :

$$\forall \mathrm{P}_j \in \mathscr{P} / \mathrm{K}_{P_i}^{INIT} \subset \mathrm{K}_{P_j}^{INIT} \cup \mathrm{T}_{p_j}^{ACC}, \mathrm{D}_{P_j}^{REC} \subset \mathrm{D}_{AJ} \tag{4.34}$$

$$\mathrm{D}_j = \left(\mathrm{D}_{j,RI}, \mathrm{D}_{j,REC} \cup \mathrm{D}_{AJ}\right) \tag{4.35}$$

Algorithm 3 Recommandation communautaire

```
for all P_j in P do
    for all t_i in K_{P_i}^{INIT} do
        if t_i ∈ (K_{P_i}^{INIT} ∪ T_{p_j}^{ACC}) then
            Equal ← True
        else
            Equal ← False
        end if
    end for
    if Equal then
        D_{P_i}^{REC} ← D_{P_j}^{REL}
    end if
end for
```

4.5 Recommandation à base de filtrage collaboratif

Le troisième type d'assistance de notre modèle est la recommandation collaborative. Ce type d'assistance utilise le filtrage collaboratif pour recommander des documents qui peuvent être pertinents aux utilisateurs.

Nous avons formulé la recommandation collaborative de la manière suivante :

$$Recommandation(\mathcal{N}, \mathrm{N}_i, \mathrm{P}_j, \mathrm{A}^D_{u_i}) :\mapsto \{(\mathrm{D}_i, Note)\} \tag{4.36}$$

Cette fonction prend en paramètres comme nous avons précédemment défini : $\mathcal{N} = \{\mathrm{N}_1, ..., \mathrm{N}_m\}$ est l'ensemble de toutes les navigations, pour chaque navigation N_j, un utilisateur u_{N_j} a un profil P_j et peut avoir annoté des documents $\mathrm{A}^D_{N_j}$.

La fonction de recommandation renvoie un ensemble de paires de documents et de leurs notes à recommander à l'utilisateur dans la session de navigation actuelle si la note est comprise entre 0,5 et 1.

L'ensemble de documents $\mathrm{D}_i = \left(\mathrm{D}_{i,RI}, \mathrm{D}_{i,REC}\right)$ correspondant à la navigation N_i est l'union des documents résultant d'une recherche d'information $\mathrm{D}_{i,RI}$ et de ceux d'une recommendation communautaire $\mathrm{D}_{i,REC}$. Les documents résultant de la fonction Recommandation sont ajoutés aux documents pour mettre à jour l'ensemble de documents recommandés $\mathrm{D}_{i,REC}$ de la manière suivante :

$$\mathrm{D}_{i,REC} = \left(\mathrm{D}_{i,RECC}, \mathrm{D}_{i,RECR}\right) \tag{4.37}$$

Avec

$$\mathrm{D}_{i,RECC} = \left\{\mathrm{d}^1_{i,RECC}, ..., \mathrm{d}^p_{i,RECC}\right\} \tag{4.38}$$

$$D_{i,RECR} = \left\{ d^1_{i,RECR}, ..., d^p_{i,RECR} \right\} \tag{4.39}$$

Les documents $D_{i,RECC}$ sont les documents issues d'une recommandation communautaire et $D_{i,RECR}$ sont ceux issues d'une recommandation collaborative.

Nous avons choisi de faire la différence entre les deux classes de documents à afficher à l'utilisateur d'une part venant de la recherche documentaire et de l'autre part de la recommandation pour pouvoir montrer à l'utilisateur lors de l'affichage que nous lui produisons deux types de documents pour son besoin informationnel. C'est donc à l'utilisateur de décider quels documents consulter.

Le fonctionnement de la recommandation peut se résumer en deux étapes :

- La première étape consiste en la recherche des utilisateurs avec des comportements similaires à l'utilisateur auquel nous voulons faire des recommandations, en utilisant les profils de ces utilisateurs ainsi que leurs sessions de navigations.

- La deuxième étape consiste à utiliser les notes et les annotations de ces utilisateurs similaires pour calculer une liste de recommandations pour l'utilisateur courant.

4.5.1 Regroupement des utilisateurs similaires

Étant donné le nombre très élevé d'utilisateurs, nous avons choisi de les regrouper dans des groupes d'utilisateurs similaires pour faciliter les comparaisons.

Pour parvenir à ce regroupement nous utilisons un algorithme de clustering.

Étant donné que nous ne disposons pas d'un nombre prédéfini de groupes d'utilisateurs, le X-Means[Pelleg *et al.* , 2000] est approprié pour regrouper les utilisateurs.

X-Means est une extension de K-Means [MacQueen *et al.* , 1967] dans le but de trouver le meilleur K.

L'algorithme de K-means se déroule selon les étapes suivantes :

Algorithm 4 L'algorithme K-Means

1: Choisir K
2: Sélectionner aléatoirement K points qui représente le centroïde de cluster
3: **repeat**
4: **for all** point x **do**
5: trouver le centroïde C le plus proche de x
6: $C\ cluster \leftarrow x$
7: **end for**
8: Re-estimer la localisation du centroïde C' en calculant le centre de masse de point associés au précédent centroïde C.
9: **until** Les centroïdes ne changent pas après les itérations.

L'algorithme K-means (alg-4) consiste à fixer au départ le nombre de cluster K. Ensuite, à partir des éléments à regrouper, K éléments sont sélectionnés pour représenter les K centroïdes C_i des clusters. L'algorithme cherche pour chaque élément X le centroïde C le plus proche. X est alors associé à ce centroïde. La localisation d'un nouveau centroïde C' est ré-estimée en calculant le centre de masse de point associés au précédent centroïde C. Les deux dernières étapes sont répétées jusqu'à ce que les centroïdes ne changent pas après les itérations. L'algorithme se termine alors avec K cluster définis par leurs centroïdes.

Le principe de X-Means est de commencer par un K_{min} un nombre minimal de classe (borne inférieure) et de continuer à ajouter des centroïdes jusqu'au K_{max} nombre maximal de classe (borne supérieure). Au cours de ce processus, les centroïdes qui ont le meilleur score sont enregistrés et seront à la fin de l'algorithme affichés.

Algorithm 5 L'algorithme X-Means

1: Initialiser K = K_{min}
2: Exécuter K-Means
3: **for** $i \leftarrow 1$ *to* k **do**
4: Remplacer chaque centroïde C_k par deux centroïdes C'_k et C''_k
5: **end for**
6: Exécuter 2-Means sur le cluster K
7: Remplacer ou retenir chaque centroïde en se basant sur le test de modèle de selection BIC
8: $BIC_k = -2 * logL + k * logN$
9: **if** $K > Kmax$ **then**
10: Stop
11: Retourner le meilleur modèle de notation trouvé lors de la recherche
12: **else**
13: Aller à 2
14: **end if**

Dans l'étape (4) les deux centroïdes sont obtenus par le changement d'un centroïde original en deux directions opposées du centroïde choisi aléatoirement dans le cluster.

L'algorithme utilise le modèle de sélection BIC pour déterminer si les deux nouveaux clusters sont meilleurs que le cluster original. $BIC_k = -2 * logL + k * logN$ avec N est le nombre d'observations, k le nombre de clusters et logL est le log-likelihood.

Tableau 4.1 – Matrice des notes attribuées par des utilisateurs à des documents

	d_1	d_2	d_3	...	d_m
u_1	$N_{1,1}$	$N_{1,2}$	$N_{1,3}$	...	$N_{1,m}$
u_2	$N_{2,1}$	$N_{2,2}$	$N_{2,3}$	...	$N_{2,m}$
u_3	$N_{3,1}$	$N_{3,2}$	$N_{3,3}$	...	$N_{3,m}$
u_n	$N_{4,1}$	$N_{4,2}$	$N_{4,3}$	...	$N_{n,m}$

Le tableau 4.1 représente les notes $N_{i,j}$ attribuées par chaque utilisateur u_i à un

document d_j, n le nombre total d'utilisateurs et m le nombre total de documents.

$$N_{i,j} = \begin{cases} 0 & \text{si l'utilisateur juge le document non pertinent,} \\ 1 & \text{si l'utilisateur juge le document pertinent,} \\ - & \text{sinon.} \end{cases} \tag{4.40}$$

Algorithm 6 L'algorithme X-Means

sélectionner les utilisateurs U ={U_1,U_2,...,U_m}
sélectionner les documents D ={d_1,d_2,...,d_n}
Initialiser K = K_{min}
sélectionner les K utilisateur qui ont plus noter comme centoides : UC={UC_1,UC_2,....UC_k}
les K cluster sont initialement vides : c={c_1,c_2,....c_k}
repeat
  for all Utilisateur $U_i \in U$ **do**
    for all Centroide $UC_j \in UC$ **do**
      Calculer $sim(U_i, UC_j$
    end for
    $sim(U_i, UC_m) = max\{sim(U_i, CU_1), sim(U_i, UC_2), ..., sim(U_i, UC_k)\}$
    $cm \leftarrow U_i$
  end for
  for all cluster $c_i \in c$ **do**
    for all Utilisateur $U_j \in U$ **do**
      $UC_i = moyenne(c_i, U_j)$
    end for
  end for
until Les centroïdes ne changent pas après les itérations.
for $i \leftarrow 1$ *to* k **do**
  Remplacer chaque centroïde UC_i par deux centroïdes UC'_i et UC''_i
end for
Exécuter 2-Means sur le cluster i (Répéter de (4) à (19) avec k=2)
Remplacer ou retenir chaque centroïde en se basant sur le test de model de selection BIC
$BIC_k = -2 * logL + k * logN$
if $K > Kmax$ **then**
  Stop
  Retourner le meilleur modèle de notation trouvé lors de la recherche
else
  Aller à 4
end if

Tableau 4.2 – Regroupement des utilisateurs

	d_1	d_2	d_3	...	d_m
C_1	$M_{1,1}$	$M_{1,2}$	$M_{1,3}$	...	$M_{1,m}$
C_2	$M_{2,1}$	$M_{2,2}$	$M_{2,3}$	...	$M_{2,m}$
C_3	$M_{3,1}$	$M_{3,2}$	$M_{3,3}$	...	$M_{3,m}$
C_t	$M_{4,1}$	$M_{4,2}$	$M_{4,3}$	...	$M_{t,m}$

Le tableau 4.2 représente la moyenne des notes $M_{i,j}$ attribuées par les utilisateurs du cluster de centroïde C_i à un document d_j, t le nombre total cluster et m le nombre total de documents.

Après le regroupement des utilisateurs dans X groupes définis par centroïdes C comme le montre le tableau 4.2. Nous comparons l'utilisateur courant avec différents centroïdes pour déterminer à quel groupe il est similaire.

4.5.2 Filtrage Collaboratif

La première étape avant d'appliquer le filtrage collaboratif, après avoir sélectionné le cluster à lequel appartient l'utilisateur courant, est d'essayer de remplir la matrice pour chaque cas lorsque l'utilisateur effectue une annotation et ne note pas le document. Nous considérons qu'une annotation d'un document représente une pertinence implicite. $An_{i,j}$ est l'annotation de l'utilisateur u_i du document d_j.

Tableau 4.3 – Remplissage de la matrice

u_i	d_1	d_2	d_3	d_m	Moyenne
Note	0	-	-	1	0.5
Annotation	-	$An_{i,2}$	-	$An_{i,m}$	
u_i	d_1	d_2	d_3	d_m	Moyenne
Note	0	1	-	1	0.66
Annotation	-	$An_{i,2}$	-	$An_{i,m}$	

La recommandation commence lorsque nous avons un utilisateur actuel et son groupe

d'utilisateurs similaires. Généralement, l'entrée du filtrage collaboratif est une simple matrice V de notes de taille $n \times m$; où n est le nombre total d'utilisateurs U et m est le nombre total d'items (document) D.

L'élément v_{ij} de la matrice représente la note attribuée par l'utilisateur u_i au document d_j. S'il se trouve que u_i n'a pas attribué une note à d_j, la valeur v_{ij} est considérée comme valeur manquante.

L'objectif du filtrage collaboratif est donc réduit à simplement estimer les valeurs manquantes de la matrice V.

Nous utilisons les valeurs existantes dans la matrice pour calculer les valeurs manquantes. Pour calculer une valeur v_{ij}, nous utilisons les voisins de u_i qui ont déjà noté d_j.

Pour les extraire, la similitude entre u_i et chaque utilisateur ayant noté le document d_j est calculée.

La mesure de similarité le plus fréquemment utilisée est le coefficient de correlation de Pearson[Resnick *et al.* , 1994].

Voici la formule de Pearson pour calculer la corrélation entre deux utilisateurs u_i et u_j [Shardanand & Maes, 1995] :

Pearson$(u_i, u_j) =$

$$\frac{\sum_{d \in D_{i,j}} (v_{(u_i,d)} - \overline{v_{u_i}})(v_{(u_j,d)} - \overline{v_{u_j}})}{\sqrt{\sum_{d \in D_{i,j}} (v_{(u_i,d)} - \overline{v_{u_i}})^2 \sum_{d \in D_{i,j}} (v_{(u_j,d)} - \overline{v_{u_j}})^2}} \tag{4.41}$$

Avec $D_{i,j}$ est le sous ensemble de documents notés par aussi bien u_i que u_j, $v_{(u_i,d)}$ est la note donnée par u_i au document d et $\overline{v_{u_i}}$ est la moyenne de note donnée par u_i. Afin de prédire la notation de l'utilisateur actuel de ce document, le FC exploite les notes des autres utilisateurs du même groupe qui ont noté ce document.

$$v(u_i, d_j) = \overline{v_{u_i}} + \frac{\sum_{u_k \in U_{d_j}} pearson(u_i, u_k) \times (v_{(u_k, d_j)} - \overline{v_{u_k}})}{card(U_{d_j})} \tag{4.42}$$

Avec : U_{d_j} est le sous-ensemble d'utilisateurs ayant noté d_j.

Le FC renvoie la matrice remplie avec des notes prédites des documents à recommander à l'utilisateur actuel.

4.5.3 Exemple

Nous commence d'abord par le remplissage des valeurs manquantes où il y a une annotation. Soit la matrice dans le tableau 5.1 des notes des utilisateurs.

Tableau 4.4 – Exemple de matrice

	d_1	d_2	d_3	d_4	Moyenne
u_1	0	1	-	1	0.66
u_2	0	1	1	0	0.5
u_3	1	1	0	-	0.66
u_4	0	-	-	1	0.5

Tableau 4.5 – Matrice remplie

	d_1	d_2	d_3	d_4	Moyenne
u_1	0	1	0.79	1	0.69
u_2	0	1	1	0	0.5
u_3	1	1	0	0.72	0.68
u_4	0	-	-	1	0.5

Supposons que nous voulons prédire la valeur de $v_{1,3}$.

Nous commençons par la selection de voisins de $U1$ qui ont noté d_3, D_{d_3} = {u_3, u_2} Par la suite, il calcule la similarité entre u_1 et chaque membre de D_{d_3} en utilisant le mesure Pearson.

- $pearson(u_1,u_2) = \frac{(0-0,66)\times(0-0,5)+(1-0,66)\times(1-0,5)+(1-0,66)\times(0-0,5)}{\sqrt{((0-0,66)^2+(1-0,66)^2+(1-0,66)^2)\times((0-0,5)^2+(1-0,5)^2+(0-0,5)^2)}}$

- $pearson(u_1,u_2) = 0,46$
- $pearson(u_1,u_3) = \frac{(0-0,66)\times(1-0,66)+(1-0,66)\times(1-0,66)}{\sqrt{((0-0,66)^2+(1-0,66)^2)\times((1-0,66)^2+(1-0,66)^2)}} = 0,3.$
- $v_{1,3} = 0,66 + \frac{0.46\times(1-0,5)+0,3\times(0-0.66)}{2} = 0.79.$
- $pearson(u_1,u_2) = 0,46$
- $pearson(u_1,u_3) = 0,3$
- $v_{1,3} = 0.79.$
- $pearson(u_3,u_1) = -0.3,$
- $pearson(u_3,u_4) = (-0.46),$
- $pearson(u_3,u_2) = -0.28,$
- $v_{3,4} = 0.72.$

Etant donné $v_{1,3}$ et $v_{3,4}$ sont tous deux supérieurs à 0.5, le document $\{(d_3,0.79)\}$ est ajouté à la liste de documents à recommander à u_1 et $\{(d_4,0.72)\}$ à u_3 en plus de la liste de document à proposer par recommandation communautaire pour u_1 et u_3 s'il y a un autre utilisateur ayant le même besoin d'information.

4.6 Conclusion

Nous avons présenté dans ce chapitre une formalisation de la navigation au sein d'un corpus documentaire. Ce modèle représente les ressources de corpus et les processus de

navigation. L'intérêt de ce formalisme se manifeste dans son utilité pour réaliser une personnalisation et assister l'utilisateur en utilisant plusieurs algorithmes d'assistance à la recherche documentaire tels que reformulation de requêtes, ou les recommandations à partir de l'historique des explorations passées.

CHAPITRE 5

PLATEFORME À BASE D'AGENTS ET D'ARTEFACTS POUR LA RECHERCHE DOCUMENTAIRE

Sommaire

5.1 Architecture multi-agent **114**

5.2 Couche navigation **117**

5.2.1 Artefact Interface 118

5.2.2 Artefact Navigation 121

5.2.3 Artefact Document 122

5.2.4 Artefact Profil 124

5.2.5 Artefact Recherche 127

5.2.6 Artefact Annotation 128

5.3 Couche Décision **131**

5.3.1 Agent Interface 131

5.3.2 Agent Reformulation 133

5.3.3 Agent Communautaire 136

5.3.4 Agent Recommandation 137

5.4 Configurations selon profil de l'utilisateur **141**

5.4.1 Expert 142

5.4.2 Intermédiaire 143

5.4.3 Novice 144

5.5 Conclusion **146**

Nous avons détaillé dans le chapitre précédent la formalisation de la navigation dans un corpus documentaire ainsi que les algorithmes d'assistance à la recherche documentaire.

Nous avons présenté dans le chapitre 3 de nombreux travaux dans la littérature ayant adopté l'approche multi-agent dans le domaine de la recherche d'information. Ces systèmes mettent en œuvre chaque tâche de la RI dans un agent distinct. Ainsi, ils sont conçus pour une utilisation et des ressources spécifiques. En plus, les agents et leurs interactions sont définies en fonction de la composante des systèmes d'information.

Dans ce chapitre, nous proposons une plateforme de navigation personnalisée dans un corpus numérique de documents reposant sur un système multi-agent. Une approche multi-agent a été adoptée de manière à représenter l'hétérogénéité des facteurs de personnalisation à appliquer. Chaque agent exerce ainsi une influence différente sur la sélection de documents à présenter à l'utilisateur représentant un facteur de personnalisation ou de recommandation. Plusieurs agents sont activés pour chaque utilisateur, interagissant avec un environnement partagé représentant leurs navigations. L'architecture globale de la plateforme découple la gestion de la navigation des décisions liées à la personnalisation permettant une composition dynamique du contenu de ces trois couches. Nous suivons une approche stigmergique dans laquelle les agents mettent en œuvre différents facteurs de personnalisation et modifient leur environnement partagé pour influencer la représentation des besoins des utilisateurs et de la navigation. Cet environnement, implémenté dans une couche d'artefacts, est ainsi un objet construit et adapté par l'activité collective des agents et des utilisateurs, qui représentent la couche décisionnelle.

Contrairement aux travaux existants, l'utilisation de la stigmergie augmente l'au-

tonomie du processus de décision. Ainsi, notre architecture est ouverte et extensible. Elle permet l'ajout et/ou la suppression de processus de recherche d'information tels que divers facteurs de personnalisation. Les contributions décrites dans ce chapitre sont :

- une modélisation sous la forme d'agents et d'artefacts d'un système multi-agent pour réaliser une navigation personnalisée dans un corpus documentaire ;
- une définition de plusieurs configurations selon le profil d'expertise de l'utilisateur.

Ce chapitre est organisé comme suit : La section 5.1 présente notre modélisation multi-agent de la plateforme de navigation personnalisée décrite selon trois couches : une couche Navigation détaillée dans la section 5.2 en représentant les artefact la composant, une couche Décision détaillée dans la section 5.3 où chaque agent est détaillé et sont fonctionnement est illustré par un exemple d'application et une couche Ressources. La section 5.4 présente un ensemble de configuration possibles de notre modèle agent et artefact en prenant en compte le niveau d'expertise des utilisateurs.

5.1 Architecture multi-agent

L'objectif de notre plateforme est de permettre à des utilisateurs de naviguer dans un corpus fermé en visualisant séquentiellement différents documents et en affinant ou adaptant leur requête au fur et à mesure de leur session de navigation. Comme indiqué précédemment, nous avons opté pour une modélisation à base d'agents et d'artefacts entrainant une coordination indirecte via l'environnement virtuel. Cette coordination est obtenue par le mécanisme de la stigmergie favorisant l'ajout indépendant de facteurs de personnalisation.

Le principe est qu'une trace laissée par une action dans l'environnement stimule l'accomplissement de l'action suivante, que ce soit par le même agent ou un agent différent. De cette façon, les actions suivantes tendent à se renforcer et bâtir sur l'existant, ce qui conduit à l'émergence spontanée d'une activité d'apparence cohérente et systématique. La stigmergie est une forme d'auto-organisation. Elle produit des structures complexes sans avoir besoin de plan, de contrôle ou même de communication directe entre les agents.

La coordination directe utilise beaucoup de temps et de ressources. Dans un système stigmergique, tous les agents ont une autonomie complète pour agir comme ils le souhaitent. Dans ce système basé sur l'action, ce qui compte est l'action sur l'environnement, autrement dit la trace laissée par un agent sur l'environnement qui entraine d'autres actions de cet agent ou des autres.

Nous représentons donc le besoin informationnel au sein d'un environnement virtuel, partagé par les agents. Ce besoin, initialement exprimé par une requête composée d'un ensemble de termes, est modifié par le biais d'actions des agents sur cet environnement. L'évolution du besoin informationnel est ainsi le résultat d'un processus de co-construction impliquant les agents et l'utilisateur dans le but d'intégrer différentes sources de personnalisation et un contrôle par l'utilisateur. De plus, l'environnement partagé des agents inclut des outils nécessaires pour la navigation (index, moteur de recherche d'information, interface avec l'utilisateur), ainsi que l'ensemble des documents jugés pertinents à l'instant courant de la navigation.

Afin de distinguer la couche décisionnelle et la couche opérationnelle de la plateforme, nous avons opté pour une approche Agent-Artefact qui est au coeur de la stigmergie. Les décisions d'évolution du besoin informationnel sont prises par les agents, parfois

en interaction avec l'utilisateur, ou à partir de l'analyse des traces de l'utilisateur. Le stockage des informations liées à la navigation et l'exécution de requêtes ponctuelles sont à la charge d'artefacts de l'environnement des agents. Cette architecture est représentée par la figure 5.1.

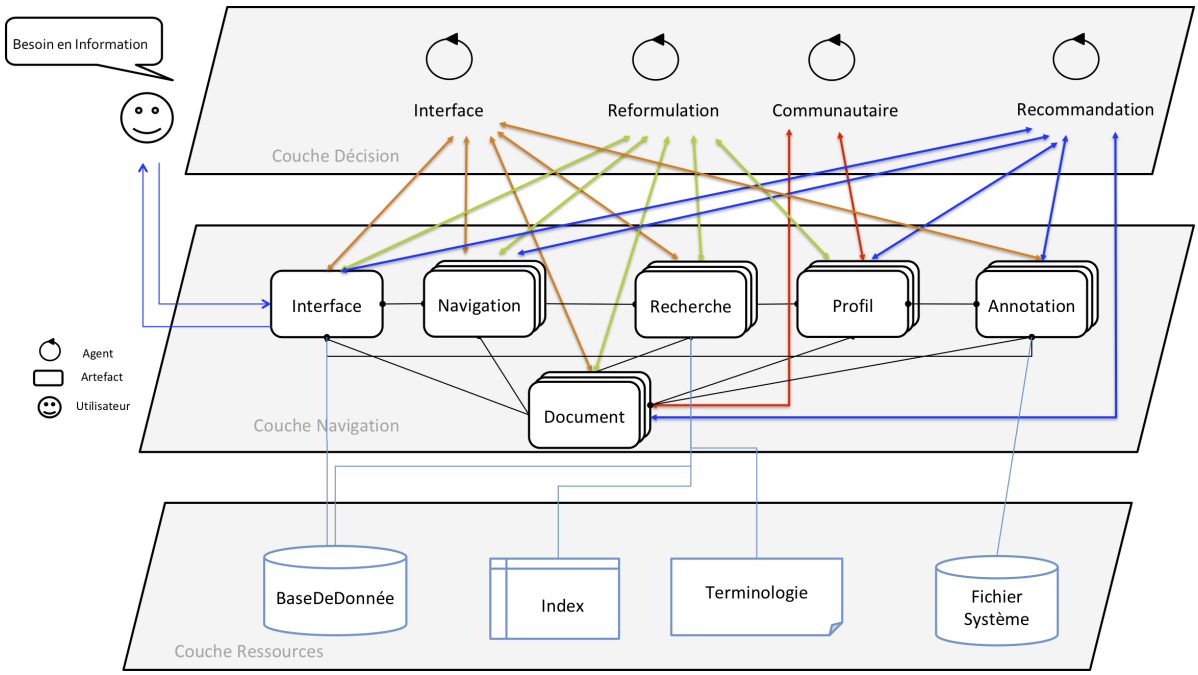

FIGURE 5.1 – Architecture Agent-Artefact proposée

Au niveau de la couche *Navigation*, six types d'artefacts sont utilisés. Un artefact *Interface* encapsule les fonctions d'interaction directe avec l'utilisateur. Chaque instance d'artefacts de type *Navigation* représente le besoin informationnel d'une session de navigation pour laquelle une instance d'artefact *Recherche* est créée pour exécuter une recherche d'information sur ce besoin. Le résultat d'une recherche est stocké dans un artefact de type *Document*, les artefacts de type *Profil* collectent des informations sur le comportement des utilisateurs. Enfin les artefacts de type *Annotation* pour traiter les annotations fournies par les utilisateurs.

La couche *Décision* contient les agents qui vont faire évoluer le besoin informationnel et les documents présentés pendant la navigation. L'agent *Interface* sert de point d'entrée à l'utilisateur pour créer une requête initiale et la modifier pendant la navigation. L'agent *Reformulation* propose l'ajout de termes au besoin informationnel sur la base des documents sélectionnés par les recherches précédentes. Une instance de l'agent *Interface* et l'agent *Reformulation* est mise en œuvre pour chaque utilisateur. Par contre un seul agent *Communautaire* est partagé par tous les utilisateurs du système. Il propose d'ajouter des documents jugés pertinents d'après l'historique des navigations passées similaires. L'agent *Recommandation* utilise le filtrage collaboratif pour recommander des documents qui peuvent être pertinents aux utilisateurs.

5.2 Couche navigation

Les six artefacts de la couche navigation sont spécifiés dans cette section.

Comme le montre la figure 5.2, les artefacts sont définis par des propriétés observables, des opérations et des descriptions de fonctionnement.

Les opérations sont les moyens par lesquels les agents interagissent avec les artefacts qui composent leur environnement. Il existe aussi un type spécial d'opération (opération de liaison) avec lequel des artefacts peuvent manipuler d'autres artefacts pour fournir des ressources complexes ou composées.

En complément, les artefacts fournissent un mécanisme par lequel les agents peuvent être informés de l'état de l'artefact sans effectuer aucune opération, ce mécanisme est implémenté en utilisant des propriétés observables.

Enfin, les artefacts sont enrichis avec une description fonctionnelle qui est utilisée comme un manuel par lequel un agent peut découvrir de nouveaux artefacts et raisonner

sur leurs descriptions, afin de déterminer s'ils présentent un intérêt pour la réalisation de ses objectifs.

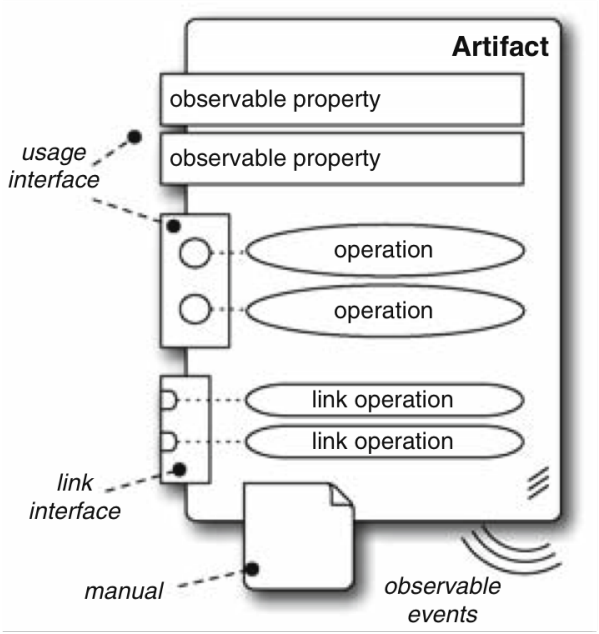

FIGURE 5.2 – Illustration d'un artefact (adoptée de [Ricci *et al.* , 2011])

5.2.1 Artefact Interface

Le rôle de l'artefact Interface est de gérer les interactions avec l'utilisateur. Il récupère les requêtes ainsi que les retours de l'utilisateur pour le compte des agents Interface, Reformulation, Communautaire et Recommandation et affiche les résultats des recherches effectuées par les agents.

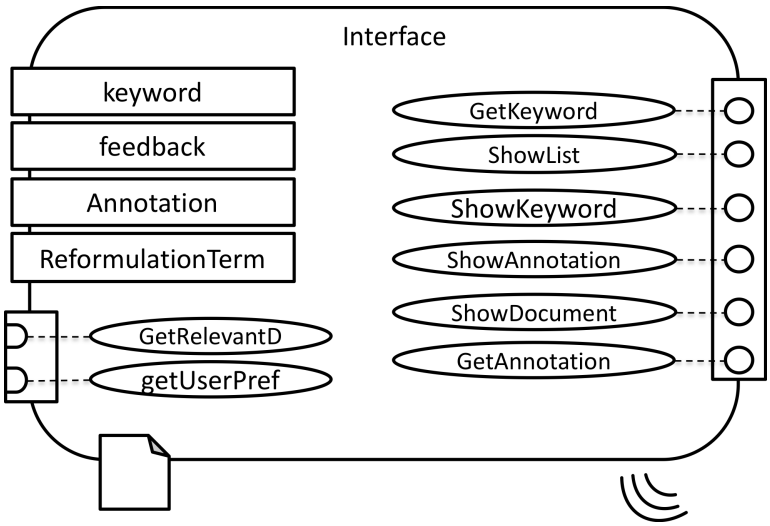

FIGURE 5.3 – Artefact Interface

Les propriétés de l'Interface

Les propriétés de l'artefact interface sont ses propriétés observables. En effet, elles sont utilisées pour exprimer un état courant de l'environnement. En observant ces propriétés, les agents mettent à jour leur base de connaissance. L'artefact Interface possède plusieurs propriétés observables :

- Le besoin informationnel : c'est le besoin courant de l'utilisateur K_{u_i}.Ce sont les mots clés qu'il a introduit. Ils sont traités pour extraire les termes signifiants,

- Le retour de pertinence RF_{u_i} : C'est la note que donne l'utilisateur sur les documents affichés et consultés, cette note peut être comprise entre 1 pour les non ou peu pertinents et 5 pour les plus pertinents,

- Les termes acceptés T_{acc} et les termes rejetés T_{rej} : Ce sont les termes sélectionnés par l'utilisateur lors de la reformulation de son besoin informationnel,

- Les annotations An : ce sont les messages d'annotations faites par les utilisateurs qui ont consulté et annoté le document courant.

Les évènements de l'Interface

Les évènements de l'artefact interface sont des signaux envoyés par l'artefact pour alerter les agents qui l'utilisent. Par exemple, l'artefact interface alerte l'agent interface quand l'utilisateur introduit son besoin informationnel, l'agent reformulation quand l'agent sélectionne des termes pour la reformulation etc.

Les opérations de l'Interface

Les opérations proposés par l'artefact Interface sont :

- la récupération des termes du besoin informationnel $GetKeyword$,
- l'affichage des résultats d'une recherche $ShowList$,
- l'affichage d'un document $ShowDocument$,
- l'affichage des annotations faites par d'autres utilisateurs $ShowAnnotation$,
- la récupération des annotations faites par les utilisateurs $GetAnnotation$,
- la récupération du retour de pertinence de l'utilisateur $GetFeedback$.

5.2.2 Artefact Navigation

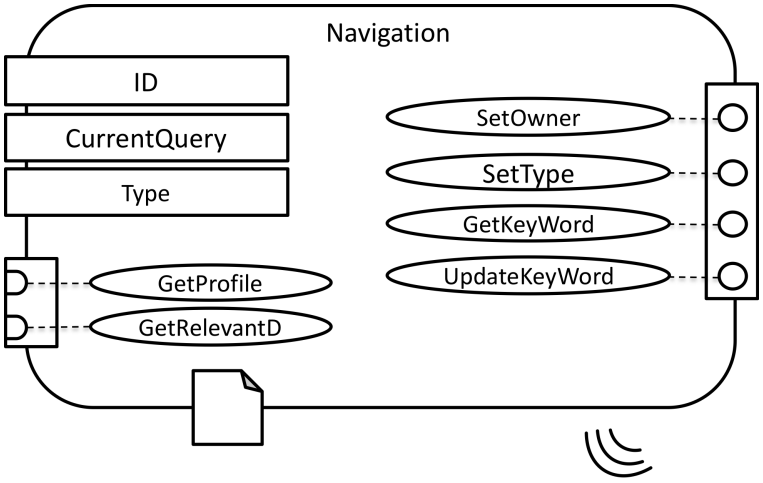

FIGURE 5.4 – Artefact Navigation

Un artefact Navigation encapsule un besoin informationnel d'un utilisateur. Rappelons notre formalisation de la navigation : Soit $\mathcal{N} = \{N_1, ..., N_m\}$ l'ensemble de toutes les navigations, pour chaque navigation N_i, un utilisateur u_{N_i} a une requête courante K_{N_i} composée de plusieurs termes $k^j_{N_i}$.

$$N_i = \{K_{N_i}, u_{N_i}\} \tag{5.1}$$

Avec

$$K_{N_i} = \left\{k^1_{N_i}, ..., k^l_{N_i}\right\} avec\ k^i_{N_i} \in K. \tag{5.2}$$

Les propriétés de la Navigation

Comme montre la figure 5.4, l'artefact Navigation possède comme propriétés observables :

- un identifiant de la navigation ID,
- le type de la navigation $Type$ qui représente si c'est une nouvelle navigation ou c'est une continuation d'une navigation précédente,

- une requête courante K_{N_i}.

Les opérations de la Navigation

Les opérations de l'artefact Navigation sont exploitables par l'agent Interface et l'agent Reformulation. Ces agents peuvent exécuter les opérations suivantes :

- $GetKeyWord$: pour avoir les mots clés de la requête introduite par l'utilisateur ; cette opération renvoie K_{N_i},
- $UpdateKeyWord$: pour mettre à jour les mots clés après une reformulation,
- $SetOwner$: cette opération permet de fixer l'utilisateur en cours u_{N_i},
- $SetType$: cette opération permet de déterminer le type de la navigation en cours si c'est une nouvelle navigation ou une suite d'une navigation existante.

5.2.3 Artefact Document

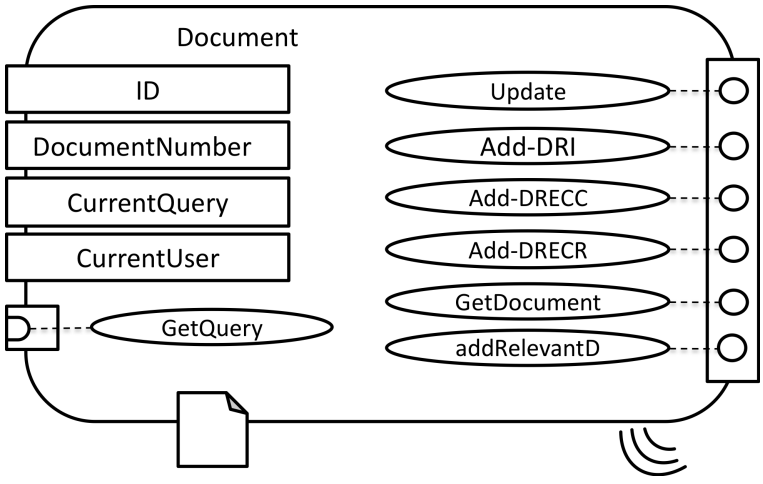

FIGURE 5.5 – Artefact Document

Un artefact Document contient une référence aux documents jugés pertinents pour une navigation donnée N_i. Cet ensemble est amené à évoluer en cours de navigation en fonction des actions des agents et de l'utilisateur. L'ensemble de documents D_i correspondant à la navigation N_i est l'union des documents résultant d'une recherche d'information $D_{i,RI}$ et de ceux recommandés par l'agent Communautaire $D_{i,RECC}$ et par l'agent Recommandation $D_{i,RECR}$ Rappelons le formalisme de représentation de documents :

$$D_i = \{D_{i,RI}, D_{i,REC}\} \tag{5.3}$$

Avec

$$D_{i,RI} = \left\{d^1_{i,RI}, ..., d^n_{i,RI}\right\}, \tag{5.4}$$

$$D_{i,REC} = \left(D_{i,RECC}, D_{i,RECR}\right) \tag{5.5}$$

Avec

$$D_{i,RECC} = \left\{d^1_{i,RECC}, ..., d^p_{i,RECC}\right\} \tag{5.6}$$

$$D_{i,RECR} = \left\{d^1_{i,RECR}, ..., d^p_{i,RECR}\right\} \tag{5.7}$$

Les propriétés de Document

Comme montre la figure 5.5, l'artefact Document possède comme propriétés observables :

- un identifiant ID,

- le nombre de documents pertinents $DocumentNumber$,
- l'utilisateur en cours $CurrentUser$,
- la requête posée $CurrentQuery$ pour avoir ces documents pertinents.

Les opérations de Document

Les agents peuvent lancer les opérations proposés par l'artefact document :

- $Update$: mettre à jour la liste des documents pertinents,
- $Add-DRI$: rajouter les documents $\mathrm{D}_{i,RI}$ résultant d'une recherche d'information à l'ensemble de documents,
- $Add-DRECC$: ajouter les documents résultant d'une recommandation communautaire $\mathrm{D}_{i,RECC}$,
- $Add-DRECR$: ajouter les documents de la recommandation collaborative $\mathrm{D}_{i,RECR}$,
- $GetDocument$: récupérer un document,
- $AddRelevantD$: ajouter des documents pertinents,
- $GetQuery$: récupérer la requête associée aux documents retournés par l'artefact recherche.

5.2.4 Artefact Profil

Le profil P_i d'un utilisateur impliqué dans une session de navigation N_i est composé par sa requête initiale $\mathrm{K}_{P_i}^{INIT} \subset \mathscr{K}$, les termes proposés en reformulation $\mathrm{T}_{P_i}^{ACC} \subset \mathscr{T}$

(acceptés par l'utilisateur) et $T_{P_i}^{REJ} \subset \mathcal{T}$ (rejetés par l'utilisateur) et enfin les références des documents jugés pertinents par l'utilisateur

$$P_i = \left(K_{P_i}^{INIT}, T_{P_i}^{ACC}, T_{P_i}^{REJ}, D_{P_i}^{REL}\right) \tag{5.8}$$

Avec

$$K_{P_i}^{INIT} = \left\{k_{P_i}^1, ..., k_{P_i}^n\right\} \tag{5.9}$$

$$T_{P_i}^{ACC} = \left\{t_{P_i}^{ACC,1}, ..., t_{P_i}^{ACC,p}\right\} \tag{5.10}$$

$$T_{P_i}^{REJ} = \left\{t_{P_i}^{REJ,1}, ..., t_{P_i}^{REJ,q}\right\} \tag{5.11}$$

$$D_{P_i}^{REL} = \left\{d_{P_i}^{REL,1}, ..., d_{P_i}^{REL,r}\right\} \tag{5.12}$$

et

$$K_{P_i}^{INIT} \cap T_{P_i}^{ACC} \cap T_{P_i}^{REJ} = \emptyset \tag{5.13}$$

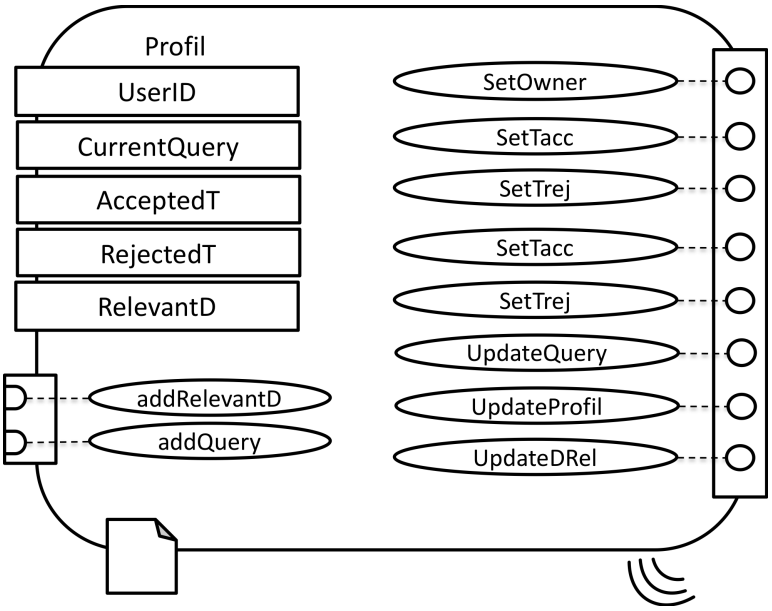

FIGURE 5.6 – Artefact Profil

Les propriétés de Profil

La figure 5.6 montre la définition de l'artefact Profil avec ses opérations et ses propriétés observables.

Par ailleurs, l'agent Reformulation peut avoir des informations grâce aux propriétés observables de l'artefact.

- $UserID$: l'identifiant de l'utilisateur,
- $CurrentQuery$: le besoin informationnel de l'utilisateur,
- $AcceptedT$: les termes acceptés,
- $RejectedT$: les termes rejetés,
- $RelevantD$: les documents jugés pertinents par l'utilisateur.

Les opérations de Profil

Nous citons quelques opérations de l'artefact Profil qui permettent aux agents de mettre à jour les informations qu'il contient :

- $SetOwner$: permet de définir l'utilisateur,
- $SetTacc$: permet de définir les termes acceptés,
- $SetTrej$: permet de définir les termes rejetés,
- $GetTacc$: permet de récupérer les termes acceptés,
- $GetTrej$: permet de récupérer les termes rejetés,
- $UpdateQuery$: permet de mettre à jour les termes de besoin informationnel courant de l'utilisateur,

- $UpdateDRel$: permet de mettre à jour la liste de documents pertinents,
- $UpdateProfil$: permet d'initialiser ou de mettre à jour le profil de l'utilisateur.

La notion de liens entre les artefacts (Links dans Cartago) permet le partage des opérations entre les artefacts.

Dans l'artefact Profil, l'agent peut accéder et ajouter les documents pertinents ainsi que les besoins d'informations de l'utilisateur $AddRelevantD$, $AddQuery$.

5.2.5 Artefact Recherche

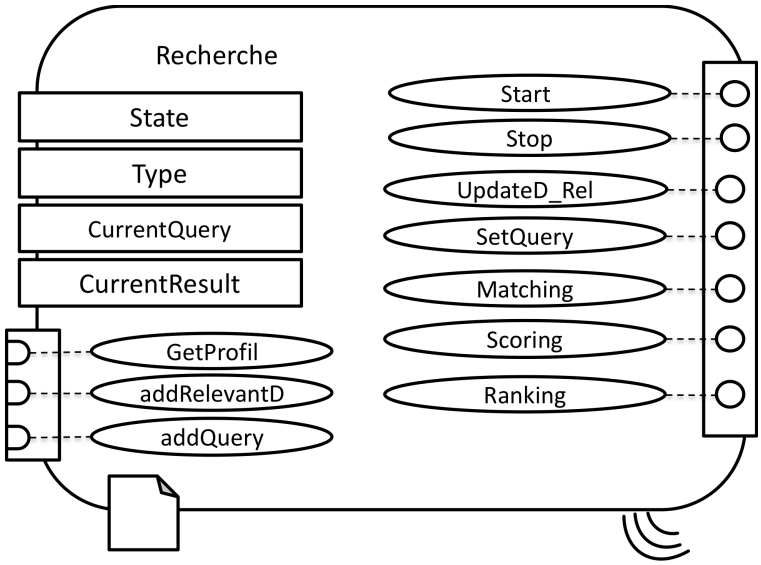

FIGURE 5.7 – Artefact Recherche

L'artefact Recherche sert de moteur de recherche d'information. Nous utilisons pour notre plateforme le moteur *Lucene* [1] qui est une application open source gratuite pour la recherche plein texte et l'analyse de contenu textuel. L'objectif de l'artefact est de prendre en entrée un ensemble de termes issus d'une requête et de fournir les références

[1] http ://lucene.apache.org/core/

de documents jugés pertinents en sortie.

$$Recherche : (\mathrm{K}) \mapsto P(\mathrm{D}) \tag{5.14}$$

L'artefact Recherche est illustré par la figure 5.7. Les processus de la recherche et d'indexation sont encapsulés dans des actions internes de l'artefact ou le corps de l'action interne représente le code en Java de la recherche par lucene.

Pour lancer la recherche, l'opération $Recherche$ permet de lancer l'action interne précédemment présenté.

Les propriétés observables de l'artefact Recherche sont :

- $CurrentResult$: affiche les résultats associés à la requête et la recherche en cours. Cette propriété est exploitée pour alimenter le Profil de l'utilisateur en terme d'historique.
- $Type$: retourne le type de la recherche. En effet on peut avoir des recherches à base de profil et des recherches classiques,
- $CurrentQuery$: la requête associée à la recherche en cours.
- $State$: retourne l'état en cours de la recherche. Cette propriété renvoie l'étape en cours dans le processus de recherche $SearchEngine$ qui peut être les opérations de l'artefact $Matching$ ou $Scoring$ ou $Ranking$.

5.2.6 Artefact Annotation

Le rôle de l'artefact annotation est de représenter les annotations de l'utilisateur. En effet, un utilisateur u_i peut annoter une ou plusieurs partie(s) $\S \subset D$ qu'il juge pertinente(s) dans un document jugé pertinent D_i^{REL}. Il sélectionne un passage du document pertinent

et y associe un commentaire appelé annotation An. L'ensemble des annotations constitue l'ensemble $\mathrm{A}_{u_i}^{D}$

Les Annotations $\mathrm{A}_{u_i}^{D}$ d'un utilisateur impliqué dans une session de navigation N_i est composé par son retour de pertinence $\mathrm{RF}_{u_i} \in \{0,5\}$, une ou plusieurs partie(s) $\S \subset D$ qu'il juge pertinente(s) dans un document jugé pertinent D_i^{REL} ainsi que les annotations An.

$$\mathrm{A}_{u_i}^{D} = \left(\mathrm{RF}_{u_i}, \S, \mathrm{An}_i\right) \tag{5.15}$$

Avec

$$\mathrm{RF}_{u_i} \in \{0,5\} \tag{5.16}$$

$$\S \subset D \tag{5.17}$$

et

$$\mathrm{An}_i = Message \tag{5.18}$$

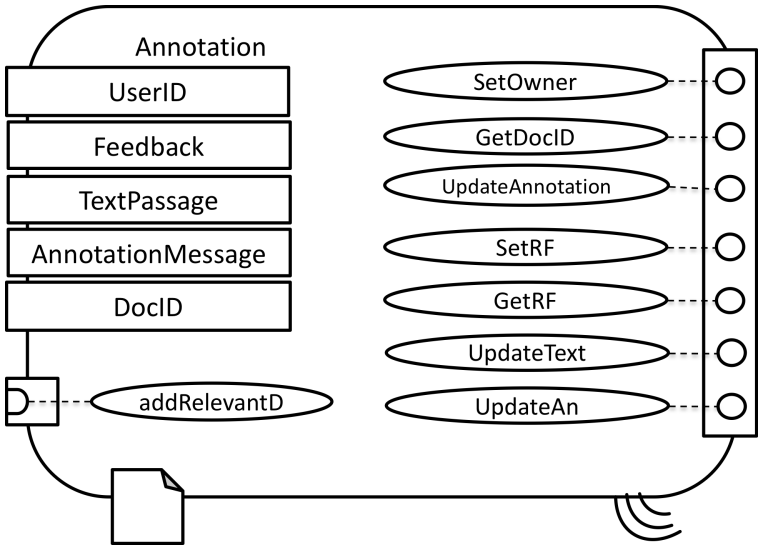

FIGURE 5.8 – Artefact Annotation

Les annotations des documents de chaque utilisateur sont stockées dans l'artefact Annotation. Pour intégrer les annotations dans la couche Navigation, nous avons besoin de mettre à jour l'artefact Interface pour permettre à l'utilisateur tout d'abord d'effectuer ses annotations en sélectionnant des passages dans le document et introduire son message. Ces modifications des documents annotés sont enregistrées et mises à jour par l'agent Interface dans l'artefact Document. Ainsi, lors de l'affichage des résultats de recherche, les documents annotés auront le poids le plus important par rapport à d'autres documents pertinents à afficher.

Ces annotations peuvent être utilisées par l'agent Recommandation puisque l'artefact Annotation contient le retour de pertinence de l'utilisateur. Comme nous le détaillons ci-dessous, nous faisons l'hypothèse qu'un document annoté est implicitement pertinent.

Les propriétés de Annotation

La figure 5.8 montre la définition de l'artefact Annotation avec ses opérations et ses propriétés observables.

- $UserID$: l'identifiant de l'utilisateur courant u_i,
- $Feedback$: le retour de pertinence RF_{u_i},
- $TextPassage$: le passage qualifié comme pertinent dans le document $\S \subset D$,
- $AnnotationMessage$: le texte de l'annotation An,
- $DocID$: les termes acceptés D.

Les opérations de Annotation

Nous citons quelques opérations de l'artefact Annotation qui permettent aux agents de mettre à jour les informations qu'il contient :

- $SetOwner$: permet de définir l'utilisateur,
- $SetRF$: permet de définir le retour de pertinence de l'utilisateur,
- $GetRF$: permet de récupérer le retour de pertinence de l'utilisateur,
- $GetDocID$: permet de récupérer l'identifiant du document courant,
- $UpdatePassage$: permet de mettre à jour le passage qualifié comme pertinent dans le document,
- $UpdateAn$: permet de mettre à jour le texte d'annotation de l'utilisateur,
- $UpdateAnnotation$: permet d'initialiser ou de mettre à jour l'artefact annotation.

5.3 Couche Décision

La couche décision est composée de quatre agents qui ont la capacité de percevoir et d'agir sur leur environnement partagé représentant un processus de navigation en cours.

5.3.1 Agent Interface

Il existe un agent Interface par utilisateur, en charge du suivi de sa navigation. Un utilisateur u commence une navigation en saisissant une requête $\mathrm{K}_u = \{\mathrm{k}_u^1, ..., \mathrm{k}_u^n\}$ composée d'un ensemble de mots-clés. Si l'utilisateur n'avait pas de navigation en cours, une nouvelle session de navigation est créée avec cette requête.

S'il existe déjà des navigations pour cet utilisateur, l'agent interface doit déterminer s'il s'agit du début d'une nouvelle navigation ou si l'insertion de nouveaux termes correspond à la continuité d'une session en cours. Dans ce but, l'agent calcule une mesure de similarité entre les termes de la requête et ceux de toutes les navigations précédentes de l'utilisateur cette mesure est détaillée dans le chapitre 4 section 4.2 .

Sinon, un nouvel artefact de navigation est créé. À chaque fois qu'un artefact de navigation est créé ou mis à jour, l'agent effectue une recherche par l'artefact correspondant $Recherche(N_j)$ avec pour résultat de créer ou de mettre à jour l'artefact Document avec un ensemble $D_{N_j,RI}$.

Enfin, l'agent Interface envoie l'ensemble de documents D_j à l'artefact Interface pour les visualiser.

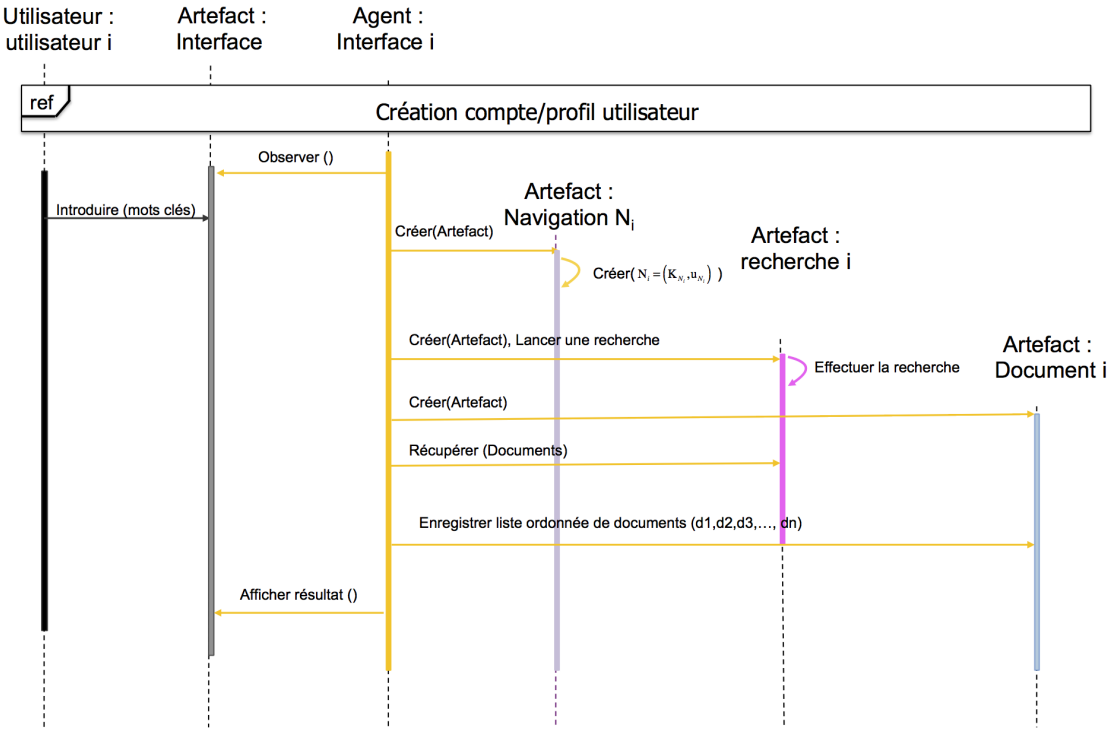

FIGURE 5.9 – Dynamique de l'agent Interface

Exemple de recherche par mot clés

Le fonctionnement de chaque agent est illustré dans cette section par un exemple de navigations dans un corpus documentaire sur le droit international du transport.

Supposons qu'un utilisateur, Paul, initie une navigation en cherchant toutes les décisions juridiques ayant impliquées des camions. Sa requête initiale est $\mathrm{K}_{N_1}^{INIT} = \{camions\}$. L'agent Interface crée un artefact Navigation défini par $\mathrm{N}_1 = (\{camions\}, Paul)$.

L'agent Interface lance une première requête : $Recherche(camions)$ via l'artefact Recherche, qui renvoie deux documents $\mathrm{D}_{N_1,RI} = \{Decision_1, Decision_{10}\}$. Il s'ensuit la création d'un artefact Document $D_1 = \{\{Decision_1, Decision_{10}\}, \emptyset\}$.

5.3.2 Agent Reformulation

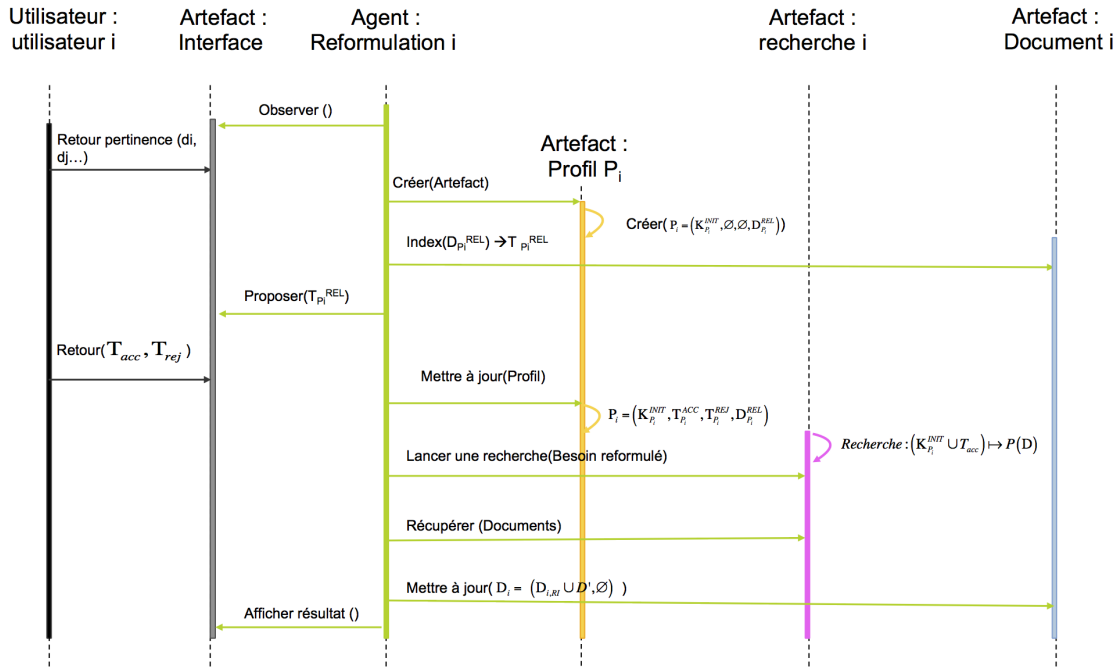

FIGURE 5.10 – Dynamique de l'agent Reformulation

Un agent Reformulation est associé à chaque utilisateur de la plateforme. Le fonctionnement de l'agent Reformulation est basé sur l'observation des artefacts. Il réagit à la perception qu'une navigation N_i ainsi qu'un ensemble de documents D_i ont été créés. Dans ce cas la première action de l'agent Reformulation est de créer un artefact Profil.

La deuxième partie du fonctionnement de l'agent Reformulation se base sur le retour de pertinence de l'utilisateur depuis l'artefact Interface. Dans notre système nous avons choisi le retour de pertinence explicite en demandant à utilisateur de donner un score de pertinence aux documents par rapport à son besoin informationnel. Par seuillage, ce retour nous permet de classer les documents comme étant pertinents ou non. L'ensemble des documents sélectionnés comme pertinents constitue l'ensemble $D_{P_i}^{REL}$ qui est ajouté à P_i.

La proposition de reformulation se base sur les termes fréquents indexant les documents jugés pertinents. L'agent reformulation extrait alors les termes de poids fort de l'index des documents jugés pertinents et construit un ensemble $Proposition(N_i)$.

Cet ensemble est proposé à l'utilisateur qui peut choisir de les accepter (ensemble T_{acc}) ou de les rejeter (ensemble T_{rej}). Le profil est alors mis à jour par l'agent Reformulation.

Si $T_{P_i}^{ACC}$ a été modifié, l'agent Reformulation lance une nouvelle recherche en utilisant la nouvelle requête. Ce processus de reformulation, ici décrit après le premier résultat d'une requête, sera réitéré à chaque fois que l'utilisateur sélectionnera des documents proposés comme étant pertinents (ceux-ci pouvant avoir été issus d'une requête déjà re-formulée).

Exemple de reformulation

L'utilisateur choisit de visualiser les deux documents. Il juge comme un résultat pertinent $Decision_{10}$. Ce retour de pertinence, ainsi que les premiers résultats, entraîne la création d'un artefact Profil par l'agent Reformulation initialisé à $P_1 = (\{camions\}, \emptyset, \emptyset, \{Decision_{10}\})$.

L'agent Reformulation extrait les termes de l'index de la décision$_{10}$ $Index(Decision_{10})$ qui renvoie {véhicules motorisés, fruits, camions, aliments, accidents, motocyclettes, automobiles}.

Les termes ayant le poids le plus important et ne contenant pas les termes du besoin d'information initial sont : *véhicules motorisés*, *accidents* et *fruits* . L'agent Reformulation propose alors ces termes à l'utilisateur qui a le choix de les accepter ou les rejeter. Supposons, que Paul n'accepte que les deux derniers, son profil de navigation est alors devenu : $P_1 = \{\{\text{camions}\}, \{\text{accidents , fruits}\}, \{\text{véhicules motorisés}\}, \{Decision_{10}\}\}$.

L'agent Reformulation lance une nouvelle recherche : $Recherche(camions, accidents, fruits)$ qui renvoie comme résultats les décisions (1, 10, 15, 20 et 19). L'utilisateur consulte les décisions (20, 19 et 15) et juge les décisions 15 et 19 comme pertinentes. L'agent Reformulation met alors à jour le profil de l'utilisateur $Paul$. $P_1 = (\{\text{camions}\}, \{\text{accidents, fruits}\}, \{\text{véhicule motorisés}\}, \{Decision_{10}, Decision_{15}, Decision_{19}\})$.

Les termes de l'index de la décision$_{15}$ et la décision$_{19}$ sont : *fruits*, *camions*, *dommages*,

marchandises, *poids lourd*, *pourcentage de dommages* et *délai de livraison*. À partir de la proposition de ces termes et qui ne font pas partie du besoin d'information initial et des termes déjà acceptés ou rejetés, l'utilisateur sélectionne les termes qui lui semblent utiles pour affiner son besoin d'information, ce qui amène à la proposition de nouveaux documents que l'utilisateur pourra à nouveau juger comme pertinents. Ce processus continue jusqu'à ce que l'utilisateur n'accepte plus de termes pour affiner son besoin d'information ou s'il exprime un nouveau besoin qui caractérise une nouvelle navigation.

5.3.3 Agent Communautaire

La couche décision contient un unique agent Communautaire dont le rôle est de proposer des documents qui ont été jugés pertinents pour des profils de navigation proches. L'agent Communautaire observe la création d'un profil P_i. Il analyse tous les autres Profils P_j pour comparer leurs termes initiaux et acceptés à ceux de P_i. Si tous les termes de P_i sont inclus dans ceux d'un profil P_j, les documents jugés pertinents pour P_j seront recommandés pour P_i. L'agent Communautaire propose alors ces documents à l'utilisateur.

Exemple de recommandation à base de profils proches

Supposons qu'un utilisateur, Jean, utilise la plateforme pour une navigation ultérieure à Paul. L'artefact de navigation associé est $N_2 =$ ({fruits, délai de livraison}, $Jean$). Parallèlement aux documents proposés par les agents Interface et Reformulation, l'agent Communautaire compare ce besoin d'information initial avec les autres profils connus.

Comme résultat il trouve : {fruits, délai de livraison} $\subset$ {$K^{INIT} \cup T^{ACC}$, $Paul$}.

Il propose comme documents recommandés les documents jugés pertinents par $Paul$

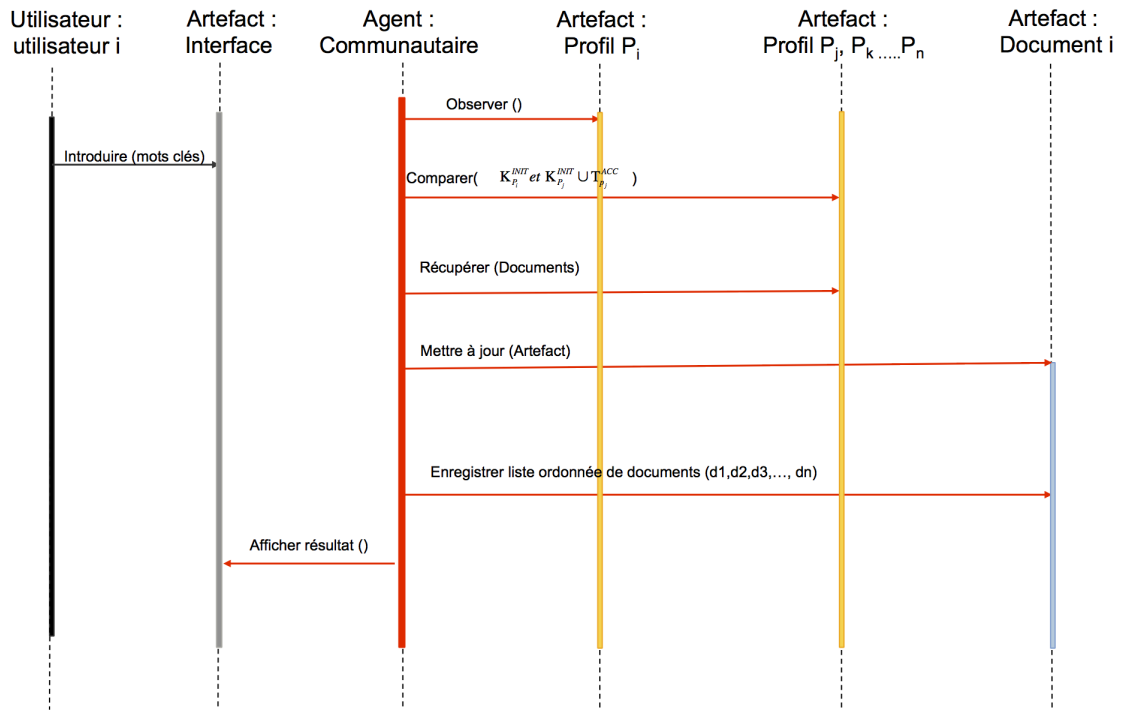

FIGURE 5.11 – Dynamique de l'agent Communautaire

pour ce besoin d'information à savoir les décisions (27, 15, 19 et 10). Le résultat à afficher à l'utilisateur est composé des résultats de l'agent Interface et des recommandations de l'agent Communautaire : Décision_1, Décision_{51}, Décision_{11}, Décision_{19} et Décision_{27}, Décision_{15}, Décision_{10}, Décision_{19}.

5.3.4 Agent Recommandation

Un agent Recommandation utilise le filtrage collaboratif pour recommander des documents qui peuvent être pertinents aux utilisateurs.

Nous avons formulé la recommandation de l'agent recommandation de la manière suivante :

$$Recommandation(\mathcal{N}, \mathrm{N}_i, \mathrm{P}_j, \mathrm{A}_{u_i}^{D}) :\mapsto \{(\mathrm{D}_i, Note)\} \tag{5.19}$$

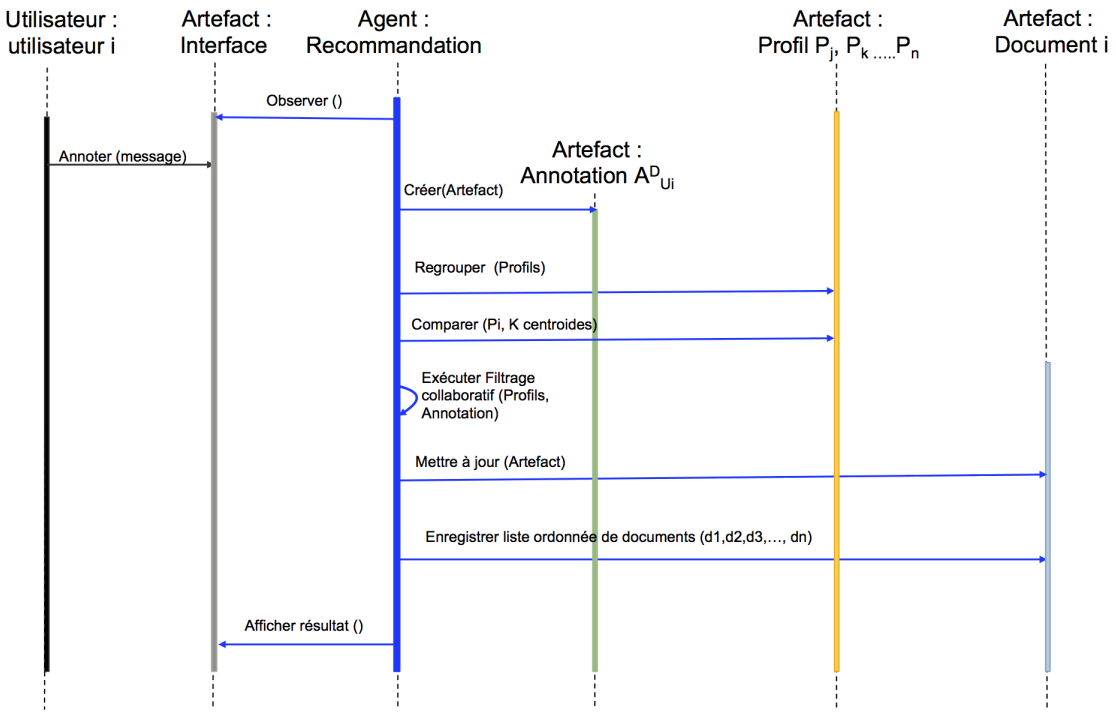

FIGURE 5.12 – Dynamique de l'agent Recommandation

Le fonctionnement de l'agent Recommandation peut se résumer en deux étapes :

- La première étape consiste en la recherche des utilisateurs avec des comportements similaires à l'utilisateur auquel nous voulons faire des recommandations, en utilisant les profils de ces utilisateurs ainsi que leurs sessions de navigations.
- La deuxième étape consiste à utiliser les notes et les annotations de ces utilisateurs similaires pour calculer une liste de recommandations pour cet utilisateur.

L'agent Recommandation observe les retours de pertinence de l'utilisateur. À l'introduction d'une annotation, Il crée un artefact Annotation pour sauvegarder le message saisi par l'utilisateur, le passage sélectionné et le retour de pertinence de cet utilisateur pour le document concerné. D'autre part, l'agent Recommandation regroupe les profils des utilisateurs similaires. Il exécute l'algorithme X-means et compare ensuite le profil

de l'utilisateur en cours avec les centroides des groupes construits. Une fois que l'agent Recommandation connait à quel groupe de profils appartient le profil de l'utilisateur actuel, il exécute le filtrage collaboratif. Comme nous avons détaillé dans le chapitre précédent, l'agent Recommandation remplie la matrice de notes en utilisant les annotations de l'utilisateur. Il utilise les valeurs existantes dans la matrice pour calculer les valeurs manquantes. L'agent Recommandation obtient la matrice remplie avec des notes prédites des documents à recommander à l'utilisateur actuel. Il met à jour l'artefact Document et affiche les documents à recommander en utilisant l'artefact Interface.

Exemple de recommandation à base d'annotation et de filtrage collaboratif

Reprenons notre scénario d'exécution. Le profil de $Paul$ est le suivant.

$P_1 = (\{\text{camions}\}, \{\text{accidents}, \text{fruits}\}, \{\text{véhicule motorisés}\}, \{Decision_{10}, Decision_{15}, Decision_{19}\})$.

$Paul$ annote le Décision 19 en affirmant que le passage de la décision de la ligne de 3 à la ligne 9 est la partie la plus pertinente. L'agent Interface met ensuite à jour l'artefact Annotation de l'utilisateur $Paul$. $A_{Paul}^{D_{19}} = (1, L3 - L9, \text{"cette partie détaille les raisons de la non-livraison de la marchandise"})$.

Les annotations sont mises en évidence en les affichant dans les premiers résultats si le document contenant l'annotation répond au besoin d'information.

L'agent Recommandation commence d'abord par le remplissage des valeurs manquantes où il y a une annotation. Soit la matrice dans le tableau 5.1 des notes des utilisateurs.

Tableau 5.1 – Exemple de matrice

	d_1	d_2	d_3	d_4	Moyenne
$Paul$	0	1	-	1	0.66
$Julien$	0	1	1	0	0.5
$Jean$	1	1	0	-	0.66
u_4	0	-	-	1	0.5

Tableau 5.2 – Matrice remplie

	d_1	d_2	d_3	d_4	Moyenne
$Paul$	0	1	0.79	1	0.66
$Julien$	0	1	1	0	0.5
$Jean$	1	1	0	0.72	0.66
u_4	0	-	-	1	0.5

Supposons que nous voulons prédire la valeur de $v_{1,3}$.

Il commence par la selection de voisins de $Paul$ qui ont noté d_3, D_{d_3} = {$Jean$, $Julien$} Par la suite, il calcule la similarité entre $Paul$ et chaque membre de D_{d_3} en utilisant le mesure Pearson.

$$pearson(Paul, Julien) = \frac{(0-0{,}66)\times(0-0{,}5)+(1-0{,}66)\times(1-0{,}5)+(1-0{,}66)\times(0-0{,}5)}{\sqrt{((0-0{,}66)^2+(1-0{,}66)^2+(1-0{,}66)^2)\times((0-0{,}5)^2+(1-0{,}5)^2+(0-0{,}5)^2)}} = 0{,}46.$$

$$pearson(Paul, Jean) = \frac{(0-0{,}66)\times(1-0{,}66)+(1-0{,}66)\times(1-0{,}66)}{\sqrt{((0-0{,}66)^2+(1-0{,}66)^2)\times((1-0{,}66)^2+(1-0{,}66)^2)}} = 0{,}3.$$

$$v_{1,3} = 0{,}66 + \frac{0.46\times(1-0{,}5)+0{,}3\times(0-0.66)}{2} = 0.79.$$

$pearson(Paul, Julien) = 0,46$ and $pearson(Paul, Jean) = 0,3$ and $v_{1,3} = 0.79$. $pearson(Jean, Paul) = -0.3$, $pearson(Jean, u_4) = (-0.46)$, $pearson(Jean, Julien) = -0.28$ et $v_{3,4} = 0.72$.

Etant donné $v_{1,3}$ et $v_{3,4}$ sont tous deux supérieurs à 0.5, l'agent Recommandation ajoute $\{(d_3, 0.79)\}$ à la liste de documents à recommender à $Paul$ et $\{(d_4, 0.72)\}$ à $Jean$ en plus de la liste de document à proposer par l'agent communautaire pour $Paul$ et $Jean$ s'il y a un autre utilisateur ayant le même besoin d'information.

Nous avons déjà les résultats à afficher à $Jean$ qui comprend les résultats de l'agent Interface $D_{i,RI}$ ={Décision$_1$, Décision$_{51}$, Décision$_{11}$, Décision$_{19}$} et les recommandations de l'agent communautaire : $D_{i,RECC}$ ={ Décision$_{27}$, Décision$_{15}$, Décision$_{10}$ et Décision$_{19}$ } à laquelle vient s'jouter $D_{i,RECR}$ ={Décision$_4$} la recommandation de l'agent Recommandation.

5.4 Configurations selon profil de l'utilisateur

Dans les sections précédentes, nous avons présenté notre modèle pour la recherche de documents à base d'agent et un exemple d'application. L'objectif de l'approche stigmergique que nous avons adoptée est de permettre une adaptation dynamique du système en fonction des ressources et des processus spécifiques d'un corpus numérique donné. Ainsi, nous montrons dans cette section que le système introduit dans la section 5.1 peut être configuré avec des ressources de corpus spécifiques. Nous pouvons ajouter/supprimer des agents et des artefacts selon les attentes de l'utilisateur. Nous montrons ainsi dans les trois sections suivantes trois configurations utilisant des combinaisons différentes

d'agents et d'artefacts selon le niveau d'assistance requis.

Pour cette question, le prochain paragraphe présente différentes configurations de notre architecture Agent-Artefact selon le degré d'expertise de l'utilisateur de la plateforme.

5.4.1 Expert

Une utilisateur est dit expert lorsqu'il est capable de préciser son besoin informationnel sans difficulté. Dans une recherche experte l'utilisateur n'a pas besoin de service de reformulation de sa requête. Généralement, il effectue des recherches ponctuelles. La recommandation communautaire est jugée utile pour un utilisateur expert vu qu'elle fournit un ensemble de résultats pertinents sans faire une deuxième recherche (les documents qui ont été jugés pertinents pour des requêtes proches). Les annotations d'un utilisateur expert sont enrichissantes pour les recommandations futures des autres utilisateurs.

La configuration proposée à un utilisateur expert est donc composée d'un agent interface et d'un agent communautaire. Pour les artefacts, la configuration comporte un artefact Interface, un artefact profil, un artefact recherche et un artefact document et un artefact annotation.

Une variante peut être d'y rajouter l'artefact navigation si nous voulons enregistrer les navigations de l'utilisateur. La configuration est illustrée par la figure 5.13.

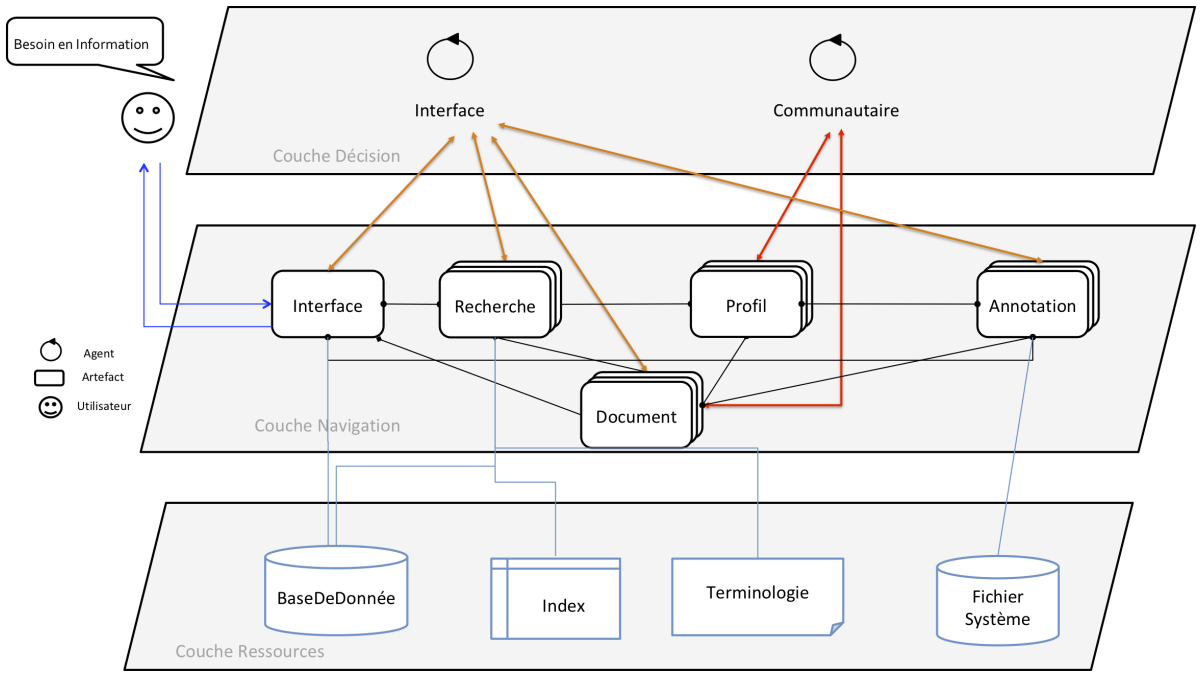

FIGURE 5.13 – Configuration pour un utilisateur expert

5.4.2 Intermédiaire

Un utilisateur est dit intermédiaire lorsqu'il a une connaissance du domaine mais sans pouvoir être qualifié d'expert. Il peut avoir besoin de l'aide de la plateforme pour affiner son besoin informationnel. Pour cette raison la configuration pour un utilisateur intermédiaire comporte un agent reformulation. Ainsi le profil de l'utilisateur est mis à jour au fur et à mesure de la reformulation du besoin informationnel de l'utilisateur. Les navigations de cet utilisateur sont aussi conservées et les sessions de recherche sont comparées pour déterminer si l'on est dans la meme navigation ou dans une nouvelle.

Des recommandations communautaires sont proposées à l'utilisateur ainsi que des recommandations collaboratives. En effet ces recommandations permettent d'élargir le champs de recherche et de connaissance de l'utilisateur intermédiaire en lui proposant ce qui peut être pertinent pour lui en comparant ses besoins avec les utilisateurs similaires.

La configuration proposée à un utilisateur intermédiaire est ainsi composée d'un agent interface, un agent communautaire, un agent reformulation et un agent recommandation. Pour les artefacts, la configuration comporte un artefact Interface, un artefact navigation, un artefact profil, un artefact recherche, un artefact annotation et un artefact document.

La configuration pour un utilisateur intermédiaire est illustrée par la figure 5.14.

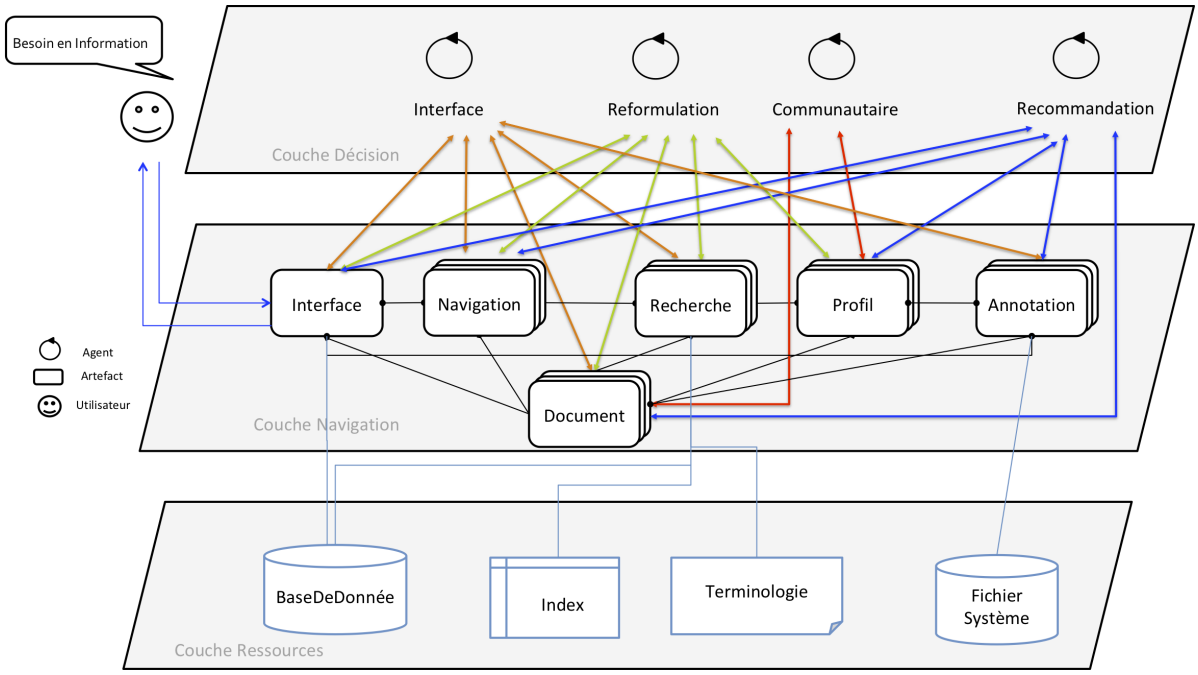

FIGURE 5.14 – Configuration pour un utilisateur intermédiaire

5.4.3 Novice

Une utilisateur est dit novice lorsqu'il n'a aucune ou peu de connaissance du domaine. Pour exprimer son besoin informationnel, il a besoin d'être aidé. Pour cette raison nous rajoutons à la configuration pour utilisateur novice un agent reformulation. Un agent communautaire est aussi pertinent vu qu'il propose les documents qui ont été déjà jugés pertinents.

L'utilisation de recommandation collaborative pour un utilisateur novice n'est pas pertinente. En effet un utilisateur novice aura probablement des recherches aux profils très différents et la recommandation en se basant sur ces profils peut donner des résultats très divergents.

La configuration proposée à un utilisateur novice est ainsi composée d'un agent interface, un agent communautaire et un agent reformulation. Pour les artefacts, la configuration comporte un artefact Interface, un artefact navigation, un artefact profil, un artefact recherche et un artefact document.

La configuration pour un utilisateur novice est illustrée par la figure 5.15.

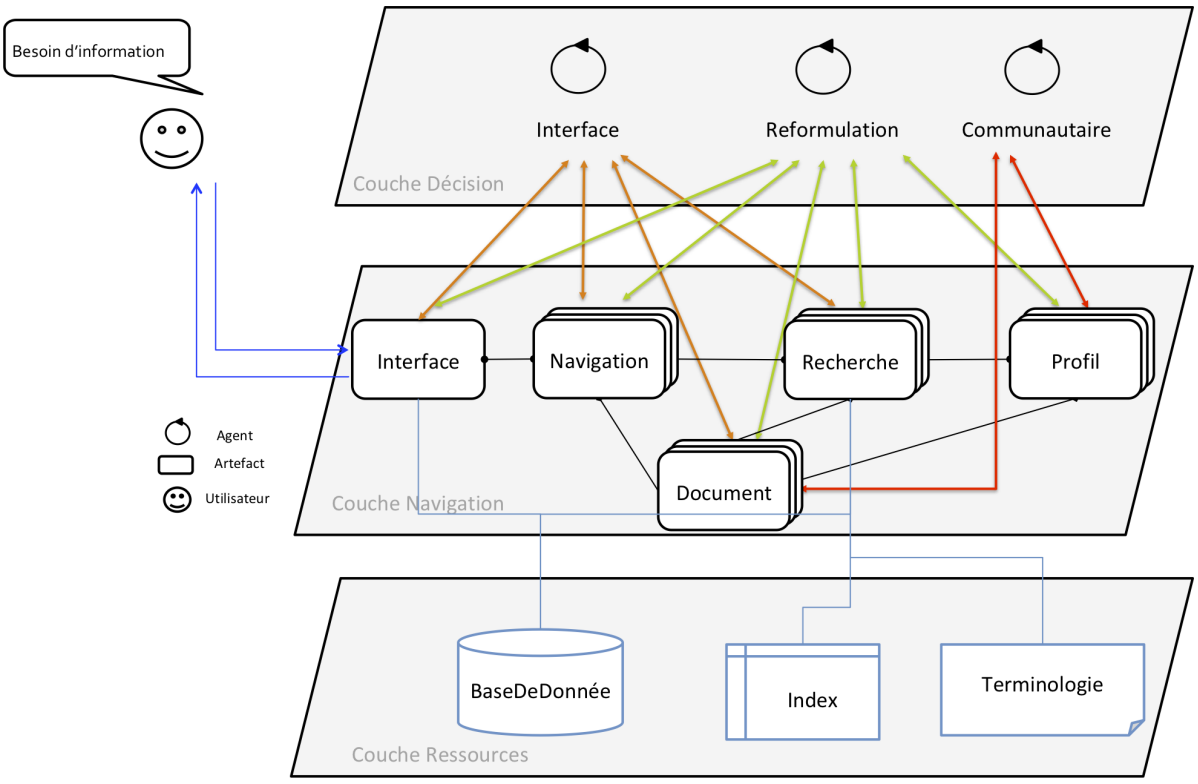

FIGURE 5.15 – Configuration pour un utilisateur novice

5.5 Conclusion

Nous avons présenté dans ce chapitre un modèle à base d'agents pour la recherche de document. Contrairement aux travaux existants en utilisant des systèmes multi-agents pour la recherche de documents, nous avons proposé un modèle général suivant une approche stigmergique afin de rester indépendant de tout corpus numérique spécifique. Deux couches ont été définies pour cette question : une couche de navigation qui représente les processus de navigation, mis en œuvre par le biais d'artefacts ; et une couche de décision composée par des agents qui agissent sur la couche de navigation pour influencer la navigation avec divers facteurs tels que reformulation de requêtes, ou les recommandations à partir de l'historique des explorations passées. Ces deux couches viennent s'ajouter à celle regroupant les ressources du système d'information.

Un scénario d'utilisation de notre architecture a été montré sur un corpus numérique sur les documents de droit des transports. L'ouverture de l'approche utilisée est illustrée par la définition de plusieurs configurations représentant autant de combinaisons des services et ressources proposés par les agents et artefacts. Même si la définition de ces configurations a été réalisée par une sélection "manuelle" d'agents et d'artefacts, elle démontre la flexibilité apportée par le découplage des différentes fonctionnalités et ouvre la voie à de possibles re-configurations dynamiques.

Troisième partie

Expérimentations et mise en oeuvre

CHAPITRE 6

EXPÉRIMENTATION ET ÉVALUATION

Sommaire

6.1 Plateforme JaCa **150**

6.1.1 Le langage de programmation agent Jason 151

6.1.2 Plateforme CArtAgO 152

6.2 Expérimentation sur la base de l'IDIT **153**

6.2.1 Base documentaire de L'IDIT 153

6.2.2 Première partie de l'expérimentation 154

6.2.3 Deuxième partie de l'expérimentation 156

6.2.4 Analyse des résultats 157

6.3 Conclusion **160**

Ce chapitre est dédié aux expérimentations menées durant cette thèse. Dans une première partie, nous présentons la plateforme JaCa qui permet le développement de notre plateforme sous trois couches Navigation, Décision et Ressources. Dans une deuxième partie nous présentons une expérimentation réalisée en conditions quasi-réelles pour tester l'efficacité des outils proposés.

6.1 Plateforme JaCa

Dans ce chapitre, nous présentons JaCa, une plate-forme de programmation polyvalente qui peut être utilisée pour développer des systèmes multi-agents en général, en exploitant à la fois les dimensions de programmation de l'agent et de l'environnement en synergie.

En fait, JaCa n'est pas une plate-forme entièrement nouvelle, mais l'intégration de deux technologies existantes de programmation d'agents que nous allons introduire dans la section suivante : (i) le langage de programmation agent Jason, pour programmer les agents BDI, et (ii) la plateforme CArtAgO, pour la programmation et l'exécution des environnements endogènes où les agents travaillent.

La nouveauté principale de la plate-forme ainsi que sa caractéristique principale est l'intégration synergique des deux technologies d'agents considérées, et en particulier les hypothèses et les simplifications que cela permet, tant du côté de l'agent que de l'environnement, de programmer un système multi-agent.

Le modèle de programmation JaCa repose sur l'intégration synergique des concepts définis dans le modèle conceptuel A & A et le modèle d'agent BDI. En conséquence, un programme JaCa est conçu comme un ensemble dynamique d'agents BDI Jason autonomes travaillant en coopération dans un environnement de travail CArtAgO partagé, dont la topologie est structurée en termes d'espaces de travail éventuellement distribués.

L'environnement est composé d'un ensemble dynamique d'artefacts, en tant qu'entités de calcul que les agents peuvent créer et supprimer dynamiquement, en plus de les utiliser et de les observer.

La programmation de l'application signifie ensuite programmer les agents Jason d'un côté, en encapsulant la logique de contrôle des tâches à exécuter, et l'environnement de travail CArtAgO de l'autre côté, comme une abstraction de première classe fournissant les actions et les fonctionnalités exploitées par les agents pour faire leurs tâches.

6.1.1 Le langage de programmation agent Jason

Jason [Bordini & Hübner, 2005][Bordini *et al.* , 2007] est une plate-forme pour le développement de systèmes multi-agents qui incorpore un langage de programmation orienté agent BDI et un environnement de développement intégré (Integrated Development Environment - IDE). Le langage vient d'une version étendue de AgentSpeak (L), un APL basé sur la logique introduit par Rao dans [Rao, 1996], qui a été plus tard beaucoup étendu dans une série de publications par Bordini, Hübner, et collègues, afin de le rendre approprié en tant que langage de programmation d'agent pratique.

Jason est disponible en open-source sous GNU LGPL sur le site officiel de Jason [Bordini & Hubner., 2018]. L'interprète a été développé pour être modulaire et facilement personnalisable. En effet, il permet aux développeurs MAS d'implémenter leurs propres personnalisations - par exemple, base de croyances personnalisées, fonctions de mise à jour et de récupération de croyances personnalisées, modifications du fonctionnement de base du cycle de raisonnement des agents, etc.

L'IDE Jason fournit une interface graphique pour l'édition d'un fichier de configuration système multi-agent, ainsi que du code AgentSpeak pour les agents individuels.

Grâce à l'IDE, il est possible d'exécuter et de contrôler l'exécution de systèmes multi-agents.

L'IDE fournit également un autre outil, appelé "Mind Inspector", un débogueur orienté agent qui permet à l'utilisateur d'inspecter les états internes des agents lorsque le système s'exécute en mode de débogage.

6.1.2 Plateforme CArtAgO

CArtAgO (Common ARtifact infrastructure for AGent Open environments) est une plateforme et une infrastructure pour la programmation et l'exécution d'environnements basés sur des artefacts pour les systèmes multi-agents, implémentant le modèle conceptuel A & A détaillé dans le chapitre 3 section 3.3.1.

Conçue de manière à être orthogonal à la dimension de programmation d'agent, elle est potentiellement intégrable avec n'importe quel langage et plateforme de programmation d'agent existant. En détail, la plateforme comprend :

- Une API basée sur le langage Java pour la programmation d'artefacts, pour la définition (programmation) de nouveaux types d'artefacts suivant le modèle de programmation A & A.

- Une API d'agent côté agent pour l'intégration de langages de programmation orientés agents dans les environnements de travail CArtAgO. Cette API fournit les moyens d'exploiter toutes les actions requises pour créer et interagir avec les artefacts, gérer et rejoindre les espaces de travail locaux et distants, etc.

- Un environnement d'exécution et des outils connexes, prenant en charge la distribution et l'exécution des environnements de travail, la gestion de l'espace de

travail et des cycles de vie des artefacts.

CArtAgO est une technologie open-source implémentée sur la plate-forme Java et publiée sous licence GNU LGPL. Les sources complètes sont librement disponibles pour téléchargement et modifications sur le site officiel de CArtAgO [Alessandro & Andrea, 2018].

6.2 Expérimentation sur la base de l'IDIT

Afin d'illustrer l'utilisation de notre plateforme dans des configurations différentes ainsi que l'intérêt des mécanismes d'assistance, une expérimentation à été menée en utilisant la base documentaire de l'Institut du Droit International du Transport (IDIT [1]) pour répondre à plusieurs question posées.

6.2.1 Base documentaire de L'IDIT

Le corpus, constitué par l'IDIT, est une base documentaire contenant autour de 40 000 références incluant des documents classés en quatre groupes : jurisprudence, articles, règlementation, fonds documentaires. Ces documents sont indexés à l'aide d'une terminologie du domaine et par une analyse plein texte des documents. La personnalisation en fonction des utilisateurs et des navigations est très intéressante dans ce type de corpus. D'une part les utilisateurs n'ont pas les mêmes besoins pour une même requête du fait de niveaux d'expertises différents. D'autre part, des navigations "typiques" ont lieu, sans être précisément formalisées, selon des usages fréquents comme des études comparatives de cas, de jurisprudence, des recherches précises, ...

[1] `http://www.idit.fr`

Nous avons utilisé une centaine des documents issus de cette base documentaire pour mettre en place une expérimentation impliquant des utilisateurs ayant des connaissances approfondies en droit et des utilisateurs néophytes mis en situation de recherche documentaire. L'objectif d'observer le comportement des experts dans leurs recherches et utilisations de la plateforme. Les algorithmes d'assistance élaborés dans le projet vont ainsi apprendre de ces observations pour les exploiter lors d'une assistance à des utilisateurs novices dans la reformulation de requêtes avec des termes plus techniques ou dans la recommandation de documents.

6.2.2 Première partie de l'expérimentation

Nous avons sollicité des étudiants en master 2 de droit de l'Université de Rouen pour la réalisation de la première phase de l'expérimentation le vendredi 6 avril 2018. Ce profil d'étudiant a été choisi pour leur connaissance experte des termes précis employés dans le domaines du droit et pour leur capacité à juger la pertinence de documents répondant à une question donnée. Ils ont accédé aux documents issus de la base de l'IDIT via notre plateforme configurée en mode Expert (cf section 5.4.1).

FIGURE 6.1 – Expérimentation en phase 1 avec des experts

Six questions leur ont été posées. Pour chacune d'entre elles, nous leur avons demandé

d'essayer de trouver via la plateforme des documents pertinents traitant de ce sujet. Pour leur recherche, les étudiants saisissent une première requête qu'ils peuvent modifier à leur gré autant de fois qu'ils le souhaitent. Dès qu'une requête est lancée, dix documents leur sont proposés par le moteur de recherche. Ils ont la possibilité de consulter chaque document sous sa forme originale en pdf ou retranscrit en format texte. Dès qu'un document est consulté il leur est demandé de noter sa pertinence dans une échelle allant de 1 (non pertinent) à 5 (très pertinent) par rapport à la question posée.

Pour chaque question un nombre approximatif de documents pertinents (c'est-à-dire évalués de +3 à +5) sont attendus. Si la requête initiale ne permet pas de ressortir suffisamment de documents pertinents, ils peuvent modifier la requête autant de fois que souhaité. Les six questions posées sont les suivantes :

1. Question 1 : A partir de la jurisprudence, quelles sont les règles établissant la responsabilité d'un transporteur ?
   Estimation du nombre de documents pertinents attendus : autour de 5

2. Question 2 : Quels arguments un transporteur peut-il évoquer pour ne pas être responsable de dégâts provoqués lors de la livraison d'un conteneur ?
   Estimation du nombre de documents pertinents attendus : autour de 1 ou 2

3. Question 3 : A partir de la jurisprudence, quels sont les principes de la responsabilité du transporteur maritime si les délais de livraison ne sont pas respectés ?
   Estimation du nombre de documents pertinents attendus : autour de 1 ou 2

4. Question 4 : Expliquez les principales caractéristiques de la responsabilité du transporteur routier national et international (régime juridique, textes applicables, principes régissant sa responsabilité, prescription). Notamment, précisez les condi-

tions qui doivent être réunies pour que sa responsabilité soit jugée illimitée en donnant des exemples issus de la jurisprudence de la base IDIT.

Estimation du nombre de documents pertinents attendus : autour de 5

5. Question 5 : Trouver des exemples de prescription en transport routier international issus de la jurisprudence.

   Estimation du nombre de documents pertinents attendus : autour de 1 ou 2

6. Question 6 : Trouver des exemples de rupture brutale d'une relation commerciale établie pour des contrats chargeur/transporteur routier ou déménageur/loueur de VL avec conducteur.

   Estimation du nombre de documents pertinents attendus : 4 ou 5

6.2.3 Deuxième partie de l'expérimentation

Dans cette deuxième phase, nous avons eu recours à des utilisateurs néophytes : six collègues du laboratoire n'ayant pas de connaissances expertes en droit. L'objectif est d'étudier si la reformulation de requête ou la recommandation utilisant les parcours de navigation des experts améliore la performance de la plateforme. Les mêmes six questions leur ont été posées mais séparées en trois groupes de deux questions. Pour chacun des groupes, la plateforme a été utilisée dans un mode différent : novice, intermédiaire ou expert, ci-dessous désignés par les lettres A, B et C utilisées dans les tableaux qui suivent.

1. Mode A : Essayer de répondre aux questions sans assistance : Une recherche classique avec les propres expressions des utilisateurs. Ils peuvent reformuler leur requête autant de fois qu'ils le souhaitent s'ils ne sont pas satisfaits des résultats.

2. Mode B : Essayer de répondre aux questions avec l'assistance d'un service de reformulation de requête : une recherche basée sur des recommandations de termes plus techniques pour la reformulation du besoin informationnel.

3. Mode C : Essayer de répondre aux questions avec l'assistance d'un service de recommandation de documents : Une recherche basée sur des recommandations de documents. Les utilisateurs peuvent consulter les documents que le système recommande en se basant sur des navigations précédentes des utilisateurs experts.

6.2.4 Analyse des résultats

Les résultats ont été analysés en étudiant, pour chaque type de participant, le nombre moyen de documents jugés pertinents et le nombre moyen de documents consultés. Pour le premier critère, nous suivons la consigne qui a été donnée aux participants de juger qu'un document est pertinent dès lors que la note octroyée est supérieure ou égale à 3. Nous avons opté pour un critère sur le nombre de documents pertinents plutôt qu'une moyenne des scores de pertinence car, après étude des données obtenues lors de la deuxième phase de l'expérimentation, nous avons constaté qu'il était difficile pour des utilisateurs novices de graduer entre 3 et 5 l'intérêt d'un document jugé pertinent. De ce fait les novices ont eu tendance à utiliser les notes extrêmes +1 ou +5 ce qui tends à exagérer les résultats obtenus dans une moyenne.

La figure 6.2 montre la moyenne des notes attribuées par les experts ainsi que des utilisateurs novices selon les trois modes pour toutes les questions.

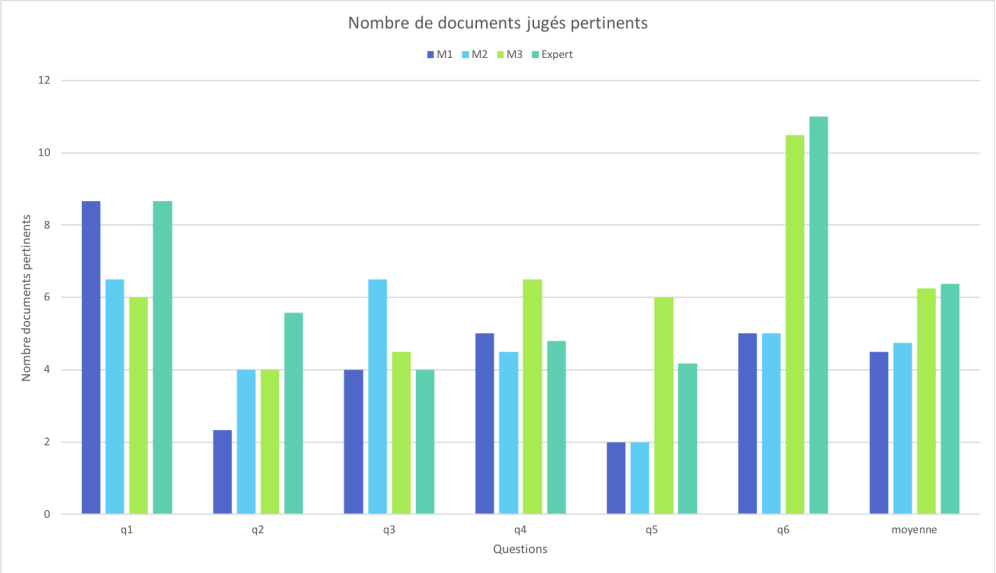

FIGURE 6.2 – La moyenne de nombre de documents pertinents des experts ainsi que des utilisateurs novices selon les trois modes pour toutes les questions

Nous restons prudent sur les conclusions à tirer de ces résultats car les petits nombres de participants, de questions et de documents ne permettent pas d'avoir une validation expérimentale des performances de notre plateforme. Néanmoins, nous en ressortons quelques tendances qui seraient à approfondir par une expérimentation à plus grande échelle en perspective de ce travail.

Nous constatons en premier lieu que les utilisateurs novices sans assistance (mode A) trouvent, en moyenne, le plus petit nombre de documents pertinents. Lorsqu'une reformulation de requête est proposée (mode B), ce nombre progresse légèrement mais sans que cela ne soit significatif. La valeur de ce critère varie selon les questions. Le mode B améliore le nombre de documents pertinents trouvés pour certaines questions (q2 et q3), le diminue pour d'autres (q1 et q4) et est identique pour les deux questions restantes (q5 et q6). Nous supposons que cela est lié à la nature des questions posées. Pour certaines questions, il est possible que des utilisateurs, même novices, connaissent

les termes juridiques à employer et qu'une reformulation de requêtes ne les aide pas, voire les incitent à utiliser des mots-clés moins intéressants.

L'utilisation de la plateforme avec le service de recommandation (mode C) fournit de meilleurs résultats en augmentant quasiment de 2 par rapport au mode A le nombre de documents jugés pertinents. Cela rapproche le résultat de la recherche documentaire de celui réalisé par les experts en phase 1.

Le deuxième critère d'évaluation que nous avons étudié est le nombre moyen de documents consultés. Le tableau 6.1 donne ce nombre moyen par mode d'utilisation de la plateforme.

Tableau 6.1 – Nombre moyen de documents consultés

	Mode A	Mode B	Mode C	Expert
Documents consultés	27,00	28,50	37,50	38,20
Moyenne	4,5	4,75	6,25	6,36

Ce tableau montre que les utilisateurs novices et sans assistance sont ceux qui consultent le moins de documents (4,5 en moyenne), pertinents ou non. Cela ne correspond pas au comportement des experts qui en consultent presque deux de plus par question (6,36 en moyenne). Cela s'explique peut-être par leur faculté à trouver plus facilement des documents à l'aide de requêtes plus appropriées. En mode B, la reformulation de requêtes augmente légèrement le nombre de documents accédés mais là encore ce n'est pas une hausse significative. Par contre, le mode C est ici aussi plus intéressant en permettant aux utilisateurs de trouver plus de documents à consulter ce qui les rapproche du comportement des experts (6,25 documents consultés en moyenne).

6.3 Conclusion

Nous avons présenté dans ce chapitre les technologies Jason et Cartago pour l'implémentation d'un système de recherche documentaire à base d'agents et d'artefacts. Dans un deuxième temps, nous avons mis en place une plateforme de recherche d'information qui nous a permis de réaliser une expérimentation avec des vrais utilisateurs. Dans une première phase, elle a eu lieu avec des étudiants en droit qui ont fourni une base d'apprentissage pour notre plate-forme. Nous avons utilisé cette base pour assister les utilisateurs qui ont fait des recherches dans la deuxième phase.

Même si l'échelle de l'expérimentation ne permet pas de parler de validation, elle nous a permis de dégager certaines tendances. La première est le besoin d'assistance pour la recherche documentaire dans un corpus thématique, les utilisateurs novices sans assistance obtenant le plus faible nombre de documents pertinents trouvés. La seconde est que l'utilisation d'un service de recommandation ayant appris sur des experts donne de meilleurs résultats que le service de reformulation de requêtes dont l'intérêt varie selon les questions.

CHAPITRE 7

CONCLUSION ET PERSPECTIVES

Nous nous sommes intéressés dans cette thèse à l'assistance à des utilisateurs impliqués dans une recherche sur une base documentaire à forte spécificité thématique. La recherche a lieu dans un corpus documentaire fermé qui rassemble des documents présélectionnés à partir de sources d'information pertinentes par rapport à la thématique du corpus. Dans ce cadre, il est possible d'identifier et de chercher à reconnaître des pratiques ou usages spécifiques à l'accès à un corpus documentaire donné et d'adapter en conséquence le comportement de la plateforme logicielle pour répondre au mieux au besoin informationnel de l'utilisateur.

Les étapes propres à une recherche documentaire nécessitent des améliorations et une prise en compte du profil de l'utilisateur dans le processus de recherche. Nous avons proposé des mécanismes de personnalisation et d'assistance pour aider l'utilisateur à mieux cerner son besoin informationnel grâce à un processus de reformulation de sa requête, de considérer l'utilisateur comme un membre d'une communauté ou un groupe en lui proposant des document en se basant sur les profils proches de son profil. Une autre des formes d'assistance proposées consiste en un processus de recommandation comparant les centre d'intérêts de l'utilisateur et les autres utilisateurs experts du système.

L'approche suivie dans cette thèse repose sur le postulat que plusieurs formes d'assistance sont utiles dans une recherche documentaire mais qu'elles ne le sont pas forcément toutes en même temps et quelque soit le profil de l'utilisateur. Cet état de fait est encore plus marqué lorsque l'on considère une base documentaire thématique consultée par des utilisateurs dont l'expertise est variée par rapport à son thème. Certains auront des difficultés à formuler une requête adéquate par manque de vocabulaire technique alors que d'autres préfèreront être guidés plus rapidement vers les documents jugés

pertinents par les précédents utilisateurs confrontés à une même situation de recherche.

Pour cela, nous avons proposé une plateforme modulaire et adaptative reposant sur le langage JaCa pour le développement de systèmes multi-agents composés d'agents logiciels pour la prise de décision et d'artefacts pour encapsuler les ressources partagées. Une approche multi-agent a été adoptée de manière à représenter l'hétérogénéité des facteurs de personnalisation à appliquer. Chaque agent exerce ainsi une influence différente sur la sélection de documents à présenter à l'utilisateur représentant un facteur de personnalisation ou de recommandation. Plusieurs agents sont activés pour chaque utilisateur, interagissant avec un environnement partagé représentant leurs navigations. L'architecture globale de la plateforme découple la gestion de la navigation des décisions liées à la personnalisation permettant une composition dynamique du contenu de ses trois couches. Nous avons montré en chapitre 5 que la simple activation/désactivation d'agents permet de déployer d'adapter la plateforme à des utilisateurs novices, de niveau intermédiaire ou experts.

Notre proposition de plateforme pour la recherche documentaire a pu être utilisée lors d'une expérimentation réalisée avec des utilisateurs experts et novices. Pour cela, nous avons utilisé une partie de la base documentaire construite par l'Institut du Droit International des Transports (IDIT), partenaire du projet PlaIR 2.0 dans lequel s'inscrit cette thèse. La recherche et la navigation dans une telle base nécessitent notamment une bonne connaissance du vocabulaire juridique employé. De cette expérimentation, nous avons constaté une amélioration dans l'efficacité de l'accès à la base à l'aide des services d'assistance que nous proposons. Nous avons également pu observer que les comportements des utilisateurs étaient différents selon qu'il s'agisse d'expert ou de novice, ce qui plaide pour une assistance personnalisée selon leur profil.

Nous avons identifié plusieurs perspectives de développement de nos travaux de thèse.

La première porte sur l'expérimentation réalisée dont l'objectif est d'illustrer les apports de l'utilisation de notre plateforme mais qui ne constitue pas encore une validation de résultats car elle ne porte pas sur un nombre suffisant de données. Une perspective est donc de la reproduire à grande échelle pour obtenir des résultats significatifs. La principale difficulté qui nous a empêché de réaliser une expérimentation à grande échelle dans le cadre de cette thèse est l'absence de jeux de données existants adaptés au suivi d'une navigation. A notre connaissance, le seul jeu de donnée qui représentent des parcours de navigation est celui constitué au MIT par Asarina et Simek [Asarina & Simek, 2015] mais que nous n'avons pu exploiter à cause de l'absence de certaines informations requises par notre plateforme.

Une deuxième perspective est d'automatiser l'adaptation de la plateforme et l'activation des agents logiciels. Même si elle est simple à réaliser, cette adaptation est pour le moment effectuée manuellement par la personne qui déploie le système ou par une sélection d'un mode par l'utilisateur. Une adaptation dynamique soulève plusieurs défis qu'il serait intéressant d'aborder : la définition de profils-types, la détection de l'appartenance d'un utilisateur à un profil-type à partir de ces traces d'utilisation et la sélection des agents logiciels à activer.

Enfin, nous avons identifié une troisième perspective qui est déjà en cours d'exploration dans le projet FEDER PlaIR 2018 faisant suite à PlaIR 2.0. Il s'agit d'enrichir le mécanisme d'annotations pour qu'il participe plus activement à l'amélioration de la recherche documentaire. Le fait d'annoter un document est notamment un marqueur d'intérêt pour ce document et nous pourrions envisager que le retour de pertinence expli-

cite utilisé dans cette thèse soit complété par un retour implicite quand un document est annoté. De plus, une analyse sémantique du contenu de l'annotation pourrait servir à quantifier ce retour. Les travaux menés sur cette perspective ont aussi pour objectif d'intégrer le système d'annotation à un système de gestion de la confiance pour évaluer la qualité des utilisateurs/annotateurs et donc de leur production.

BIBLIOGRAPHIE

Abowd, Gregory D, Atkeson, Christopher G, Hong, Jason, Long, Sue, Kooper, Rob, & Pinkerton, Mike. 1997. Cyberguide : A mobile context-aware tour guide. *Wireless networks*, **3**(5), 421–433.

Alessandro, Ricci, & Andrea, Santi. 2018 (MAI). *Official CArtAgO Website.*

Argamon, Shlomo, Koppel, Moshe, Fine, Jonathan, & Shimoni, Anat Rachel. 2003. Gender, genre, and writing style in formal written texts. *Text*, **23**(3), 321–346.

Aron, Jacob. 2011. How innovative is Apple's new voice assistant, Siri? *New Scientist*, **212**(2836), 24.

Asarina, Alya, & Simek, Olga. 2015. Recommending Documents for Complex Question Exploration by Analyzing Collective Browsing Behavior. *Pages 1248–1255 of : System Sciences (HICSS), 2015 48th Hawaii International Conference on*. IEEE.

Baeza-Yates, Ricardo, Ribeiro-Neto, Berthier, *et al.* . 1999. *Modern information retrieval*. Vol. 463. ACM press New York.

Bellotti, Victoria, Begole, Bo, Chi, Ed H, Ducheneaut, Nicolas, Fang, Ji, Isaacs, Ellen, King, Tracy, Newman, Mark W, Partridge, Kurt, Price, Bob, *et al.* . 2008. Activity-based serendipitous recommendations with the Magitti mobile leisure guide. *Pages 1157–1166 of : Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM.

Bernon, Carole, Cossentino, Massimo, & Pavón, Juan. 2005. An overview of current trends in european aose research. *Informatica*, **29**(4).

Bersot, Olivier, El Guedj, P-O, Godéreaux, Christophe, & Nugues, Pierre. 1998. A conversational agent to help navigation and collaboration in virtual worlds. *Virtual Reality*, **3**(1), 71–82.

Bordini, Rafael H, & Hübner, Jomi F. 2005. BDI agent programming in AgentSpeak using Jason. *Pages 143–164 of : International Workshop on Computational Logic in Multi-Agent Systems*. Springer.

Bordini, Rafael H., & Hubner., Jomi F. 2018 (MAI). *Rafael H. Bordini and Jomi F. Hubner. Official Jason Website.*

Bordini, Rafael H, Hübner, Jomi Fred, & Wooldridge, Michael. 2007. *Programming multi-agent systems in AgentSpeak using Jason*. Vol. 8. John Wiley & Sons.

Borlund, Pia. 2003. The concept of relevance in IR. *Journal of the American Society for information Science and Technology*, **54**(10), 913–925.

Bourne, Charles P, Anderson, Barbara, & Robinson, Jo. 1976. *DIALOG Lab Workbook*. DIALOG Information Retrieval Service.

Camazine, Scott, Deneubourg, Jean-Louis, Franks, N, Sneyd, James, Theraulaz, Guy, & Bonabeau, Eric. 2002. *Self-organization in biological systems*. Princeton University Press.

Chaib-Draa, Brahim, Jarras, Imed, & Moulin, Bernard. 2001. Systèmes multi-agents : principes généraux et applications. *Edition Hermès*, 1030–1044.

Chen, Liren, & Sycara, Katia. 1998. WebMate : a personal agent for browsing and searching. *Pages 132–139 of : Proceedings of the second international conference on Autonomous agents*. ACM.

Cherniack, Mitch, Galvez, Eduardo F, Franklin, Michael J, & Zdonik, Stan. 2003. Profile-driven cache management. *Pages 645–656 of : Data Engineering, 2003. Proceedings. 19th International Conference on*. IEEE.

Cheverst, Keith, Davies, Nigel, Mitchell, Keith, Friday, Adrian, & Efstratiou, Christos. 2000. Developing a context-aware electronic tourist guide : some issues and experiences. *Pages 17–24 of : Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM.

Chin, David Ngi. 1990. *Intelligent interfaces as agents*. University of Hawaii, Depertment of Information and Computer Sciences.

Church, Karen, Neumann, Joachim, Cherubini, Mauro, & Oliver, Nuria. 2010. SocialSearchBrowser : a novel mobile search and information discovery tool. *Pages 101–110 of : Proceedings of the 15th international conference on Intelligent user interfaces*. ACM.

Cosijn, Erica, & Ingwersen, Peter. 2000. Dimensions of relevance. *Information Processing &amp ; Management*, **36**(4), 533–550.

Côté, Marc, & Troudi, Nader. 1998. NetSA : Une architecture multiagent pour la recherche sur Internet. *Expertise Informatique*, **3**(3).

Daoud, Mariam, Lechani, Lynda-Tamine, & Boughanem, Mohand. 2009. Towards a graph-based user profile modeling for a session-based personalized search. *Knowledge and Information Systems*, **21**(3), 365–398.

Dell'Acqua, Pierangelo, Pereira, Luís Moniz, & Vitória, Aida. 2002. User preference information in query answering. *Pages 163–173 of : Flexible Query Answering Systems*. Springer.

Dempski, Kelly L. 2002. Real time television content platform : Personalized programming over existing broadcast infrastructures. *Pages 171–179 of : Proceedings of the AH'2002 Workshop on Personalization in Future TV*.

Dent, Lisa, Boticario, Jesus, McDermott, John P, Mitchell, Tom M, & Zabowski, David. 1992. A personal learning apprentice. *Pages 96–103 of : AAAI*.

Ding, Chen, Patra, Jagdish C, & Peng, Fu Cheng. 2005. Personalized web search with self-organizing map. *Pages 144–147 of : e-Technology, e-Commerce and e-Service, 2005. EEE'05. Proceedings. The 2005 IEEE International Conference on*. IEEE.

Dumais, Susan, Cutrell, Edward, Cadiz, Jonathan J, Jancke, Gavin, Sarin, Raman, & Robbins, Daniel C. 2003. Stuff I've seen : a system for personal information retrieval and re-use. *Pages 72–79 of : Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM.

Elayeb, Bilel. 2009. *SARIPOD : Système multi-agent de recherche intelligente possibiliste de documents web*. Ph.D. thesis, Institut National Polytechnique de Toulouse-INPT ; Université de Manouba ; Ecole Nationale des Sciences de l'Informatique.

Enembreck, Fabricio. 2003. *Contribution à la conception d'agents assistants personnels adaptatifs*. Ph.D. thesis, Compiègne.

Flurh, C, & Debili, F. 1985. Interrogation en langue naturelle de données textuelles et factuelles. *Intelligent Multimédia Information System and Management*.

Foskett, Douglas J, & Kent, A. 1980. *Encyclopedia of Library and Information Science*.

Fung, Chun Che, & Wongthongtham, Pornpit. 2002. A distributed environment for effective internet search using intelligent personal agent and distributed knowledge base. *Pages 379–382 of : TENCON'02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering*, vol. 1. IEEE.

Gardelli, Luca, Viroli, Mirko, Casadei, Matteo, & Omicini, Andrea. 2008. Designing self-organising environments with agents and artefacts : a simulation-driven approach. *International Journal of Agent-Oriented Software Engineering*, **2**(2), 171–195.

Golbeck, Jennifer, & Hendler, James. 2006. Inferring binary trust relationships in web-based social networks. *ACM Transactions on Internet Technology (TOIT)*, **6**(4), 497–529.

Google. 2016. *Google Now*.

Grassé, Plerre-P. 1959. La reconstruction du nid et les coordinations interindividuelles chezBellicositermes natalensis etCubitermes sp. la théorie de la stigmergie : Essai d'interprétation du comportement des termites constructeurs. *Insectes sociaux*, **6**(1), 41–80.

Grey, D, Dunne, P, & Ferguson, RI. 2000. A Mobile Agent Architecture for Searching the WWW. *In : Proc. Workshop on Agents in Industry, 4th International Conference of Autonomous Agents*. Citeseer.

Harrathi, Rami, & Calabretto, Sylvie. 2006. Un modèle de qualité de l'information. *Pages 299–304 of : EGC*.

He, Daqing, Göker, Ayşe, & Harper, David J. 2002. Combining evidence for automatic web session identification. *Information Processing & Management*, **38**(5), 727–742.

Hecht, Brent, Teevan, Jaime, Morris, Meredith Ringel, & Liebling, Daniel J. 2012. SearchBuddies : Bringing Search Engines into the Conversation. *ICWSM*, **12**, 138–145.

Höök, Ph Lic Kristina, Karlgren, Ph Lic Jussi, Wærn, Tech Lic Annika, Dahlbäck, Nils, Jansson, Carl Gustaf, Karlgren, Klas, & Lemaire, Benoit. 1998. *A glass box approach to adaptive hypermedia*. Springer.

Hu, Jian, Zeng, Hua-Jun, Li, Hua, Niu, Cheng, & Chen, Zheng. 2007. Demographic prediction based on user's browsing behavior. *Pages 151–160 of : Proceedings of the 16th international conference on World Wide Web*. ACM.

Hübner, Jomi Fred, Vercouter, Laurent, & Boissier, Olivier. 2009. Instrumenting multi-agent organisations with artifacts to support reputation processes. *Pages 96–110 of : Coordination, Organizations, Institutions and Norms in Agent Systems IV*. Springer.

Hupfer, Maureen E, & Detlor, Brian. 2006. Gender and Web information seeking : A self-concept orientation model. *Journal of the American Society for Information Science and Technology*, **57**(8), 1105–1115.

Huynh, Trung Dong, Jennings, Nicholas R, & Shadbolt, Nigel R. 2006. An integrated trust and reputation model for open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, **13**(2), 119–154.

Hwang, Amy, Ahern, Shane, King, Simon, Naaman, Mor, Nair, Rahul, & Yang, Jeannie. 2007. Zurfer : mobile multimedia access in spatial, social and topical context. *Pages 557–560 of : Proceedings of the 15th international conference on Multimedia*. ACM.

inc., Apple. 2016. *Apple inc.*

Ingwersen, Peter Emil Rerup. 1992. *Information Retrieval Interaction*. Taylor Graham.

Jaccard, Paul. 1902. Lois de distribution florale dans la zone alpine. *Bull Soc Vaudoise Sci Nat*, **38**, 69–130.

Jannach, Dietmar, Zanker, Markus, Felfernig, Alexander, & Friedrich, Gerhard. 2010. *Recommender systems : an introduction*. Cambridge University Press.

Jansen, Bernard J, Spink, Amanda, & Saracevic, Tefko. 2000. Real life, real users, and real needs : a study and analysis of user queries on the web. *Information processing & management*, **36**(2), 207–227.

Jones, Rosie, & Klinkner, Kristina Lisa. 2008. Beyond the session timeout : automatic hierarchical segmentation of search topics in query logs. *Pages 699–708 of : Proceedings of the 17th ACM conference on Information and knowledge management*. ACM.

Jones, Rosie, Kumar, Ravi, Pang, Bo, & Tomkins, Andrew. 2007. I know what you did last summer : query logs and user privacy. *Pages 909–914 of : Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM.

Kamvar, Sepandar D, Schlosser, Mario T, & Garcia-Molina, Hector. 2003. The eigentrust algorithm for reputation management in p2p networks. *Pages 640–651 of : Proceedings of the 12th international conference on World Wide Web*. ACM.

Karweg, Bastian, Huetter, Christian, & Böhm, Klemens. 2011. Evolving social search based on bookmarks and status messages from social networks. *Pages 1825–1834 of : Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM.

Kostadinov, D. 2003. *La personnalisation de l'information, définition de modèle de profil utilisateur. rapport de dea*. Ph.D. thesis, Master's thesis, Université de Versailles, France.

Koutrika, Georgia, & Ioannidis, Yannis. 2004. Rule-based query personalization in digital libraries. *International Journal on Digital Libraries*, **4**(1), 60–63.

Koutrika, Georgia, & Ioannidis, Yannis. 2005. Personalized queries under a generalized preference model. *Pages 841–852 of : Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*. IEEE.

Kozierok, Robyn, & Maes, Pattie. 1993. A learning interface agent for scheduling meetings. *Pages 81–88 of : Proceedings of the 1st international conference on Intelligent user interfaces*. ACM.

Kozierok, Robyn Arlene Edelson. 1993. *A learning approach to knowledge acquisition for intelligent interface agents*. Ph.D. thesis, Massachusetts Institute of Technology.

Lai, Kum-Yew, Malone, Thomas W, & Yu, Keh-Chiang. 1988. Object lens : a "spreadsheet" for cooperative work. *ACM Transactions on Information Systems (TOIS)*, **6**(4), 332–353.

Lauren, Doyle, & Becker, Joseph. 1975. *Information Retrieval and Processing*. Melville.

Leake, David B, & Scherle, Ryan. 2001. Towards context-based search engine selection. *Pages 109–112 of : Proceedings of the 6th international conference on Intelligent user interfaces*. ACM.

Lelu, A, & François, C. 1992. Automatic generation of hypertext links in information retrieval systems. *In : Communication of colloque ECHT'92*.

Lemouzy, Sylvain. 2011. *Systèmes interactifs auto-adaptatifs par systèmes multi-agents auto-organisateurs : application à la personnalisation de l'accès à l'information*. Ph.D. thesis, Université de Toulouse, Université Toulouse III-Paul Sabatier.

Leung, Cane Wing-ki, Chan, Stephen Chi-fai, & Chung, Fu-lai. 2006. A collaborative filtering framework based on fuzzy association rules and multiple-level similarity. *Knowledge and Information Systems*, **10**(3), 357–381.

Li, Wenjuan, & Ping, Lingdi. 2009. Trust model to enhance security and interoperability of cloud environment. *Cloud Computing*, 69–79.

Li, Yuelin, & Belkin, Nicholas J. 2008. A faceted approach to conceptualizing tasks in information seeking. *Information Processing &amp ; Management*, **44**(6), 1822–1837.

Lieberman, Henry. 1997. Autonomous interface agents. *Pages 67–74 of : Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*. ACM.

Lieberman, Henry, *et al.* . 1995. Letizia : An agent that assists web browsing. *IJCAI (1)*, **1995**, 924–929.

Liu, Fang, Yu, Clement, & Meng, Weiyi. 2004. Personalized web search for improving retrieval effectiveness. *Knowledge and Data Engineering, IEEE transactions on*, **16**(1), 28–40.

MacQueen, James, *et al.* . 1967. Some methods for classification and analysis of multivariate observations. *Pages 281–297 of : Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1. Oakland, CA, USA.

Maes, Pattie, *et al.* . 1994. Agents that reduce work and information overload. *Communications of the ACM*, **37**(7), 30–40.

Marais, Hannes, & Bharat, Krishna. 1997. Supporting cooperative and personal surfing with a desktop assistant. *Pages 129–138 of : Proceedings of the 10th annual ACM symposium on User interface software and technology*. ACM.

Martin, Innes, & Jose, Joemon M. 2004. Fetch : A Personalised Information Retrieval Tool. *Pages 405–419 of : RIAO*, vol. 4.

Massa, Paolo, & Avesani, Paolo. 2004. Trust-aware collaborative filtering for recommender systems. *Pages 492–508 of : On the Move to Meaningful Internet Systems 2004 : CoopIS, DOA, and ODBASE*. Springer.

Metral, Y Lashkari M, & Maes, Pattie. 1998. Collaborative interface agents. *Readings in agents*, 111.

Mizzaro, Stefano. 1997. Relevance : The whole history. *JASIS*, **48**(9), 810–832.

Mladenic, Dunja. 1996. *Personal WebWatcher : design and implementation*. Tech. rept. Technical Report IJS-DP-7472, Department of Intelligent Systems, J. Stefan Institute, Slovenia.

Montagna, Sara, Ricci, Alessandro, & Omicini, Andrea. 2008. A&A for modelling and engineering simulations in Systems Biology. *International Journal of Agent-Oriented Software Engineering*, **2**(2), 222–245.

Morris, Meredith Ringel, Teevan, Jaime, & Bush, Steve. 2008. Enhancing collaborative web search with personalization : groupization, smart splitting, and group hit-highlighting. *Pages 481–484 of : Proceedings of the 2008 ACM conference on Computer supported cooperative work*. ACM.

Nodine, Marian, Fowler, Jerry, Ksiezyk, Tomasz, Perry, Brad, Taylor, Malcolm, & Unruh, Amy. 2000. Active information gathering in infosleuthTM. *International Journal of Cooperative Information Systems*, **9**(01n02), 3–27.

Omicini, Andrea, Ricci, Alessandro, Viroli, Mirko, Castelfranchi, Cristiano, & Tummolini, Luca. 2004. Coordination artifacts : Environment-based coordination for intelligent agents. *Pages 286–293 of : Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*. IEEE Computer Society.

Omicini, Andrea, Ricci, Alessandro, & Viroli, Mirko. 2008. Artifacts in the A&A meta-model for multi-agent systems. *Autonomous agents and multi-agent systems*, **17**(3), 432–456.

Pazzani, Michael J, & Billsus, Daniel. 2007. Content-based recommendation systems. *Pages 325–341 of : The adaptive web*. Springer.

Pazzani, Michael J, Muramatsu, Jack, Billsus, Daniel, *et al.* . 1996. Syskill & Webert : Identifying interesting web sites. *Pages 54–61 of : AAAI/IAAI, Vol. 1*.

Pelleg, Dan, Moore, Andrew W, *et al.* . 2000. X-means : Extending K-means with Efficient Estimation of the Number of Clusters. *Pages 727–734 of : ICML*, vol. 1.

Perenon, Pascal. 2000. Réalisation d'un prototype de système de recherche d'informations scientifiques : indexation non thématique sous forme de métadonnées et développement d'une interface de consultation prenant en compte les profils des utilisateurs. *Mémoire de DEA, Sous la direction de Sylvie Lainé-Cruzel, Laboratoire RECODOC, Université Claude Bernard Lyon*, **1**.

Porter, Martin F. 1980. An algorithm for suffix stripping. *Program*, **14**(3), 130–137.

Pretschner, Alexander, & Gauch, Susan. 1999. Ontology based personalized search. *Pages 391–398 of : Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on*. IEEE.

Ramchurn, Sarvapali D, Huynh, Dong, & Jennings, Nicholas R. 2004. Trust in multi-agent systems. *The Knowledge Engineering Review*, **19**(01), 1–25.

Rao, Anand S. 1996. AgentSpeak (L) : BDI agents speak out in a logical computable language. *Pages 42–55 of : European Workshop on Modelling Autonomous Agents in a Multi-Agent World*. Springer.

Resnick, Paul, Iacovou, Neophytos, Suchak, Mitesh, Bergstrom, Peter, & Riedl, John. 1994. GroupLens : an open architecture for collaborative filtering of netnews. *Pages 175–186 of : Proceedings of the 1994 ACM conference on Computer supported cooperative work*. ACM.

Rhodes, Bradley. 2003. Using physical context for just-in-time information retrieval. *Computers, IEEE Transactions on*, **52**(8), 1011–1014.

Ricardo, BY, & Berthier, RN. 2011. Modern Information Retrieval the concepts and technology behind search second edition. *Addision Wesley*, **84**, 2.

Ricci, Alessandro, Viroli, Mirko, & Omicini, Andrea. 2006a. CArtAgO : A framework for prototyping artifact-based environments in MAS. *Pages 67–86 of : Environments for Multi-Agent Systems III*. Springer.

Ricci, Alessandro, Omicini, Andrea, Viroli, Mirko, Gardelli, Luca, & Oliva, Enrico. 2006b. Cognitive stigmergy : Towards a framework based on agents and artifacts. *Pages 124–140 of : Environments for Multi-Agent Systems III*. Springer.

Ricci, Alessandro, Buda, Claudio, & Zaghini, Nicola. 2007. An agent-oriented programming model for SOA &amp ; web services. *Pages 1059–1064 of : Industrial Informatics, 2007 5th IEEE International Conference on*, vol. 2. IEEE.

Ricci, Alessandro, Piunti, Michele, & Viroli, Mirko. 2011. Environment programming in multi-agent systems : an artifact-based perspective. *Autonomous Agents and Multi-Agent Systems*, **23**(2), 158–192.

Robertson, Stephen E, & Jones, K Sparck. 1976. Relevance weighting of search terms. *Journal of the American Society for Information science*, **27**(3), 129–146.

Robertson, Stephen E, Walker, Steve, Jones, Susan, Hancock-Beaulieu, Micheline M, Gatford, Mike, *et al.* . 1995. Okapi at TREC-3. *NIST SPECIAL PUBLICATION SP*, **109**, 109.

Rocchio, Joseph John. 1966. *Document retrieval systems – Optimization and evaluation*. Ph.D. thesis, Harvard University.

Rocchio, Joseph John. 1971. Relevance feedback in information retrieval. *The SMART retrieval system : experiments in automatic document processing*, 313–323.

Sabater, Jordi, & Sierra, Carles. 2005. Review on computational trust and reputation models. *Artificial intelligence review*, **24**(1), 33–60.

Salton, G, & McGill, M. 1970. *The concept of" relevance" in information science : A historical review*.

Salton, Gerard. 1969. A comparison between manual and automatic indexing methods. *American Documentation*, **20**(1), 61–71.

Salton, Gerard. 1971. The SMART retrieval system—experiments in automatic document processing. *Englewood Cliffs*.

Salton, Gerard. 1989. Automatic text processing : The transformation, analysis, and retrieval of. *Reading : Addison-Wesley*.

Salton, Gerard, & McGill, Michael J. 1986. Introduction to modern information retrieval. *McGraw-Hill*.

Santos, C, & Vieira, N. 2004. Use reformulated profile in information filtering. *In : Proceedings of the AAAI Workshop on SemanticWeb Personalization, San Jose, California*.

Saracevic, Tefko. 2007. Relevance : A review of the literature and a framework for thinking on the notion in information science. Part III : Behavior and effects of relevance. *Journal of the American Society for Information Science and Technology*, **58**(13), 2126–2144.

Savoy, Jacques, & Gaussier, Éric. 2010. Information Retrieval. *Pages 455–484 of : Handbook of Natural Language Processing, Second Edition.* CRC Press.

Shardanand, Upendra, & Maes, Pattie. 1995. Social information filtering : algorithms for automating "word of mouth". *Pages 210–217 of : Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co.

Shen, Xuehua, Tan, Bin, & Zhai, ChengXiang. 2005. Context-sensitive information retrieval using implicit feedback. *Pages 43–50 of : Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM.

Sheth, Beerud, & Maes, Pattie. 1993. Evolving agents for personalized information filtering. *Pages 345–352 of : Artificial Intelligence for Applications, 1993. Proceedings., Ninth Conference on*. IEEE.

Sheth, Beerud Dilip. 1994. *A learning approach to personalized information filtering*. Ph.D. thesis, Massachusetts Institute of Technology.

Sieg, Ahu, Mobasher, Bamshad, & Burke, Robin. 2007. Web search personalization with ontological user profiles. *Pages 525–534 of : Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM.

Silverstein, Craig, Marais, Hannes, Henzinger, Monika, & Moricz, Michael. 1999. Analysis of a very large web search engine query log. *Pages 6–12 of : ACm SIGIR Forum*, vol. 33. ACM.

Simonnot, Brigitte. 2008. La pertinence en sciences de l'information : des modèles, une théorie ? *Problématiques émergentes dans les Sciences de l'Information*, 161–182.

Sowa, John Florian. 1983. *Conceptual structures : information processing in mind and machine*. Addison-Wesley Pub., MA.

Speretta, Micro, & Gauch, Susan. 2005. Personalized search based on user search histories. *Pages 622–628 of : Web Intelligence, 2005. Proceedings. The 2005 IEEE / WIC / ACM International Conference on*. IEEE.

Spink, Amanda, & Jansen, Bernard J. 2004. Web Search : Public Searching of the Web (Band 6). *Information Science and Knowledge Management, Kluwer, Dordrecht*.

Stanfill, Craig, & Waltz, David. 1986. Toward memory-based reasoning. *Communications of the ACM*, **29**(12), 1213–1228.

Sullivan, Joseph, Sullivan, Joseph W, Tyler, Sherman W, *et al.* . 1994. *Intelligent user interfaces*. Citeseer.

Sun, Jian-Tao, Zeng, Hua-Jun, Liu, Huan, Lu, Yuchang, & Chen, Zheng. 2005. Cubesvd : a novel approach to personalized web search. *Pages 382–390 of : Proceedings of the 14th international conference on World Wide Web*. ACM.

Sycara, Katia, Pannu, Anandee, Williamson, Mike, Zeng, Dajun, & Decker, Keith. 1996. Distributed intelligent agents. *IEEE Intelligent Systems*, 36–46.

Tamine, Lynda, Zemirli, Nesrine, & Bahsoun, Wahiba. 2007. Approche statistique pour la définition du profil d'un utilisateur de système de recherche d'information. *Information-Interaction-Intelligence*, **7**(1), 5–25.

Taylor, Robert S. 1968. Question-negotiation and information seeking in libraries. *College &amp ; research libraries*, **29**(3), 178–194.

Teevan, Jaime, Morris, Meredith Ringel, & Bush, Steve. 2009. Discovering and using groups to improve personalized search. *Pages 15–24 of : Proceedings of the Second ACM International Conference on Web Search and Data Mining*. ACM.

Thomas, Christoph G, & Fischer, Gerhard. 1997. Using agents to personalize the Web. *Pages 53–60 of : Proceedings of the 2nd international conference on Intelligent user interfaces*. ACM.

Valckenaers, Paul, Van Brussel, Hendrik, Kollingbaum, Martin, & Bochmann, Olaf. 2001. Multi-agent coordination and control using stigmergy applied to manufacturing control. *Pages 317–334 of : Multi-agent systems and applications*. Springer.

Vallet, David, Cantador, Iván, & Jose, Joemon M. 2010. Personalizing web search with folksonomy-based user and document profiles. *Pages 420–431 of : Advances in Information Retrieval*. Springer.

Van Dyke Parunak, H, Brueckner, Sven, & Sauter, John. 2002. Digital pheromone mechanisms for coordination of unmanned vehicles. *Pages 449–450 of : Proceedings of*

the first international joint conference on Autonomous agents and multiagent systems : part 1. ACM.

Van Rijsbergen, Cornelis J. 1986. A non-classical logic for information retrieval. *The computer journal*, **29**(6), 481–485.

Van Setten, Mark, Pokraev, Stanislav, & Koolwaaij, Johan. 2004. Context-aware recommendations in the mobile tourist application COMPASS. *Pages 235–244 of : Adaptive hypermedia and adaptive web-based systems*. Springer.

Victor, Patricia, Cornelis, Chris, De Cock, Martine, & Da Silva, Paulo Pinheiro. 2009. Gradual trust and distrust in recommender systems. *Fuzzy Sets and Systems*, **160**(10), 1367–1382.

Viroli, Mirko, Holvoet, Tom, Ricci, Alessandro, Schelfthout, Kurt, & Zambonelli, Franco. 2007. Infrastructures for the environment of multiagent systems. *Autonomous Agents and Multi-Agent Systems*, **14**(1), 49–60.

Walter, Frank Edward, Battiston, Stefano, & Schweitzer, Frank. 2008. A model of a trust-based recommendation system on a social network. *Autonomous Agents and Multi-Agent Systems*, **16**(1), 57–74.

Weber, Ingmar, & Castillo, Carlos. 2010. The demographics of web search. *Pages 523–530 of : Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM.

Weyns, Danny, Omicini, Andrea, & Odell, James. 2007. Environment as a first class abstraction in multiagent systems. *Autonomous agents and multi-agent systems*, **14**(1), 5–30.

wikipédia. 2016. *Wikipédia*.

Xu, Yunjie, & Chen, Zhiwei. 2006. Relevance judgment : What do information users consider beyond topicality ? *Journal of the American Society for Information Science and Technology*, **57**(7), 961–973.

Yahoo. 2016. *My Yahoo !*

Yau, Stephen S, Liu, Huan, Huang, Dazhi, & Yao, Yisheng. 2003. Situation-aware personalized information retrieval for mobile internet. *Pages 639–644 of : Computer*

Software and Applications Conference, 2003. COMPSAC 2003. Proceedings. 27th Annual International. IEEE.

Ying, Josh Jia-Ching, Lu, Eric Hsueh-Chan, Lee, Wang-Chien, Weng, Tz-Chiao, & Tseng, Vincent S. 2010. Mining user similarity from semantic trajectories. *Pages 19–26 of : Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks*. ACM.

Zhang, Byoung-Tak, & Seo, Young-Woo. 2001. Personalized web-document filtering using reinforcement learning. *Applied Artificial Intelligence*, **15**(7), 665–685.

Zhang, Yong, Hou, Li-li, Zhou, Zhen-long, & Ding, Hong-chang. 2006. Multi-agent paradigm and conceptual graphs in information retrieval model. *Pages 875–880 of : Intelligent Systems Design and Applications, 2006. ISDA'06. Sixth International Conference on*, vol. 2. IEEE.

Ziegler, Cai-Nicolas, & Lausen, Georg. 2004. Spreading activation models for trust propagation. *Pages 83–97 of : e-Technology, e-Commerce and e-Service, 2004. EEE'04. 2004 IEEE International Conference on*. IEEE.