وزارة التعليم العالي و البحث العلمي

BADJI MOKHTAR-ANNABA UNIVERSITY
UNIVERSITÉ BADJI MOKHTAR-ANNABA

جامعة باجي مختار - عنابة

Faculté des sciences de l'ingénieur

Département d'informatique

Année : 2010

MÉMOIRE

Présenté en vue de l'obtention du diplôme de **MAGISTER**

Intitulé

Recherche d'information dans le Web ubiquitaire

Option

Génie logiciel

Par

Hakim Ghers

DIRECTEUR DE MÉMOIRE : D[r] Mohamed Taher Kimour, MC en informatique, Univ. Annaba

DEVANT LE JURY

PRÉSIDENT:	D[r] Rachid Boudour,	MC en informatique, Univ. Annaba
EXAMINATEURS:	D[r] Hayet Merouani,	MC en informatique, Univ. Annaba
	D[r] Yamina Tlili,	MC en informatique, Univ. Annaba

Remerciements

Je tiens à exprimer mes remerciements les plus vifs à M. KIMOUR Mohamed Taher, Maître de conférences à l'université de Badji Mokhtar – Annaba, qui m'a fait l'honneur d'être le directeur de mon mémoire de magister.

J'ai appris de lui toute une philosophie de travail dans le domaine de la recherche scientifique. La liberté et la confiance qu'il m'a accordées ont beaucoup contribué au développement de mon autonomie dans le travail. Malgré que je me fusse longtemps loin de lui, il m'a apporté lors de nos rencontres des conseils judicieux, des encouragements, un soutien moral, beaucoup de sympathie et de bonne humeur.

Je tiens à remercier, également, les membres de Jury, Dr Rachid Boudour, Dr Hayette Merouani, et Dr yamina Tlili. Je les remercie infiniment pour le temps qu'ils ont consacré à l'évaluation de mon travail.

Merci individuellement à mes collègues de l'équipe CAS2I pour leur sympathie et pour l'ambiance amicale qui règne au sein du travail.

Je ne dois pas oublier de remercier tous mes enseignants depuis l'école primaire, c'est eux qui ont éveillé en moi mon amour et ma passion pour la science et pour le savoir.

Sans doute je ne vais pas trouver les mots pour remercier les personnes qui me sont les plus chères: mes parents, pour leur sacrifices, leur patience, et tous ce qu'ils ont fait pour m'apporter le bonheur.

Enfin, je tiens à remercier tous mes amis et tous ceux qui m'ont soutenu surtout pendant les moments délicats, en particulier les membres de ma grande famille qui m'ont accompagné par la pensée tout au long de ce mémoire. J'ai également une pensée toute particulière pour mon oncle décédé. Par son exemple et ses valeurs, il a contribué à sa façon à l'aboutissement de ce travail que je lui dédie aujourd'hui.

ملـــخص

الإعلام الآلي الإبكيتاري أصبح اليوم حقيقة، بفضل وضع عدد متزايد من الأجهزة الرقمية الذكية المترابطة فيما بينها بتقنيات الشبكات الرقمية و الإنترنت. الواب الإبكيتاري يرى إذا، على أنه دمج بين الأنترنت و الإعلام الآلي الإبكيتاري، الذي يضيف للواب الحالي، الترابط بين مختلف الأجهزة الرقمية المتحركة، الحساسية لوضعية و حالة المستعملين، التكيف مع اللاتجانس الموجود بين الشبكات الرقمية وأيضا مصادر المعلومات. وبشكل دقيق، لغرض وضع التكنلوجيات الرقمية الجديدة في خدمة المستعملين للحصول على المعلومات في أي وقت وفي أي مكان و باستعما ل أي جهاز رقمي ذكي. و لكن البحث عن المعلومات المهمة، في حجم كبير للواب الإبكيتاري ، يعد عملية غاية في تطلعات المستعملين وفي نفس الوقت معقدة جدا.

هذه المذكرة تهتم أساسا بمظهرين: من جهة، ما يتعلق بحساسية نظام البحث عن المعلومات في الواب الإبكيتاري ، هذا المظهر يتعلق بالبحث عن المعلومات المهمة حسب سياق البحث، أي بما يناسب رغبات المستعملين، حالتهم، مكان تواجدهم وكذلك خواص الأجهزة الرقمية المستعملة. ومن جهة أخرى، ما يتعلق بالتزايد المطرد في حجم المعلومات المنقولة عبر الشبكات الرقمية، والتي يضاف إليها معطيات سياق البحث.

من أجل تحقيق هده الأهداف، أخذنا بعين الاعتبار سياق البحث عن المعلومات على أنه متعدد مستويات الخصائص، الذي يؤثر مباشرة على نوعية وسرعة استخراج المعلومات المهمة. كذلك قمنا باقتراح هندسة تعتمد على مفهوم دور البحث و العميل المتحرك، قمنا أيضا بتقديم نموذج لعميل متحرك حساس للسياق بطريقة دينامكية، من أجل التأقلم مع تغيرات محيط البحث الإبكيتاري.

تصرف عميل البحث المتحرك مبني على أساس تجميع أدوار البحث صغيرة الحجم، إنه عبارة عن أجزاء دقيقة قابلة للتبديل، حيث يتم تشكيلها بما يناسب حركية وتغير سياق البحث، وهكذا العميل باستطاعته الانتقال عبر الشبكات بمجموعة مصغرة من أدوار البحث، مع العلم أن في كل خادم يوجد هناك قاعدة خاصة بأدوار البحث.

بهذا الشكل، العميل الذي يزور الخادم يختار الأدوار الملائمة على حسب وصائف و سياق البحث، وهذا لإثراء قدراته البحثية، كما بإمكانه المشاركة في تطوير قاعدة الأدوار هذه وذلك بوضع أدوار جديدة أو بتطوير أدوار قديمة. يسمح هذا بتقليص الحمولة على مستوى العميل المتحرك مما يسهل عليه التنقل وهذا يؤدي إلى تقليص الحمولة الإجمالية للشبكات.

كلمات مفاتيح
الإعلام الآلي الإبكيتاري، الواب الإبكيتاري ، نظام البحث عن المعلومات، العميل المتحرك، الدور، الجزء، التأقلم، دينامكية.

Résumé

L'informatique ubiquitaire devient aujourd'hui une réalité, grâce à la mise en réseau d'un nombre croissant de dispositifs informatiques par le biais de technologies réseaux et de l'Internet. Le Web ubiquitaire est vu alors, comme une synthèse de l'Internet et de l'informatique ubiquitaire qui étend le Web existant, la coordination de dispositifs mobiles, la sensibilité au contexte, l'adaptation à l'hétérogénéité des réseaux et des sources d'information et également, la mise en disponibilité des nouvelles technologies de l'information et de la communication au service des utilisateurs, à n'importe quel moment, à partir de n'importe quel endroit et avec n'importe quel dispositif mobile. Cependant, rechercher de l'information pertinente, dans un volume important du Web ubiquitaire est une tache à la fois cruciale et très complexe.

Ce mémoire s'intéresse principalement à deux aspects : d'une part, celui de la sensibilité au contexte, quoi doit fournir un système de recherche d'information dans le Web ubiquitaire. En fait, cet aspect inclut, l'adaptation de l'information retournée à l'utilisateur en fonction de la mobilité et de changement très important du contexte de la recherche dans l'environnement ubiquitaire d'exécution. Et d'autre part, celui de l'augmentation de la charge du réseau. Cet aspect englobe, la croissance massive des informations véhiculées à travers les réseaux, à cela, vient s'ajouter un surcroît par les informations contextuelles qu'un système de recherche d'information dans le Web ubiquitaire doit les traiter.

Pour cela, nous avons pris en compte le contexte de la recherche comme étant une multitude de niveau de caractéristiques qui influence directement la qualité et la vitesse d'extraction de l'information pertinente. De plus, nous avons proposé une architecture basée rôle et agent mobile. Nous avons présenté, également, un modèle d'agent mobile à base de rôle adaptable, sensible au contexte dynamiquement, pour s'adapter aux variations de son environnement de recherche ubiquitaire. Le comportement de l'agent mobile de recherche proposé est construit principalement à partir d'un assemblage de rôles de recherche de granularité faible, il s'agit, de micro-composants remplaçables dont leur configuration se réalise en fonction de la variabilité du contexte de la recherche. Ainsi, l'agent peut se déplacer à travers les réseaux avec un ensemble restreint de rôles de recherche étant donné qu'à tout serveur d'information, il existe une base spécifique de rôles de recherche. De cette façon, l'agent visitant un serveur sélectionne sur la base de ses paramètres de recherche les rôles adéquats, à fin d'enrichir son comportement et de même, il peut contribuer à l'évolution de cette base de rôle par déposition de nouveaux rôles, ou bien par adaptation des rôles existants. Ceci réduit la charge au niveau de l'agent mobile et donc le trafic de son déplacement, et par conséquent, réduction de la charge totale des réseaux.

Mots clés

Informatique ubiquitaire, web ubiquitaire, recherche d'information, agent mobile, rôle, sensibilité au contexte, adaptation, composant, flexibilité.

Abstract

Ubiquitous computing becomes today a reality, by the setting in network of an increasing number of computer devices and by the slant of network technologies and the Internet. The Ubiquitous Web is seen then, like a synthesis of Internet and ubiquitous computing that spreads the existing Web, mobile devices coordination, context-awareness, adaptation to the heterogeneity of networks and sources of information and also, put available new information and communication technologies to the service of users, at any time, from any where and with any mobile device. However, information retrieval, in an important volume of the ubiquitous Web is at a time a crucial and very complex process.

This memory is interested mainly two aspects: on the context-awareness, what must provide an information retrieval system in the ubiquitous Web. In fact, this aspect includes, the adaptation of information returned to the user according to the mobility and context change in the ubiquitous execution environment. The other hand, the growing in the total load of networks, this aspect includes, the massive growth of the information transported through networks, to it, comes to be added an addition by the contextual information.

For this, we took in account the context of research as being a multi-level characteristic that influences the quality and the speed of extraction of information directly. Besides, we proposed an architecture based role and mobile agent. We presented, also, an adaptable mobile agent model based role witch is dynamically context aware, to adapt to the variations of its ubiquitous retrieval environment.

The behavior of the proposed retrieval mobile agent is constructed mainly from an assembly of retrieval roles with weak granularity; it is about, replaceable micro-components whose is realized according to the variability of the research context. Thus, the agent can be moved across networks with a restricted set of retrieval roles, given that all servers, there is a specific base of retrieval roles.

In this way, the agent visiting a server selects on the basis of its search parameters the adequate roles, in order to enrich its behavior and in the same way, he can contribute to the evolution of this role basis by deposition of new roles, or by adaptation of the existing roles. This reduces the load on the level of the mobile agent, therefore the traffic of its displacement, and also, the total load of the networks.

Keys words

Ubiquitous computing, ubiquitous Web, information retrieval, context- awareness, mobile agent, role, adaptation, component, flexibility.

Table des matières

Table des Figures

Introduction générale

Il est connu de nos jours que le savoir est un pilier très important dans le développement des peuples. En effet, les grandes nations n'hésitent pas à réserver une partie de plus en plus importante de leur budget à l'enseignement et la recherche scientifique. Grâce aux moyens humains et financiers mis en place, la science est en constante évolution, le domaine qui à profité le plus de ce fait c'est l'informatique, en effet, des avancés étonnantes que ce soient sur le plan matériel que sur le plan logiciel ont permit le développement de l'Internet comme étant une gigantesque source d'information qui est devenu un outil considérable et même indispensable dans la vie quotidienne, grâce à l'email, au chat, bibliothèques en ligne et forum, etc.

Le volume de données présente sur Web est en augmentation extrêmement rapide, et en faisant une moyenne des estimations trouvées, on peut dire que le nombre de documents accessibles est de l'ordre de dizaines de milliards. Ces documents peuvent être de nature scientifique, industrielle ou commerciale. Ces documents peuvent contenir à un moment donné des informations stratégiques nécessaires à la résolution des problèmes dans la vie quotidienne. Cette taille colossale et la demande des utilisateurs posent un défi à la communauté scientifique qui doit être en mesure de proposer des outils efficaces de recherche d'information.

Les efforts continus des chercheurs ont permis jusqu'à présent d'améliorer sans cesse les performances et la qualité des services fournis par les outils de recherche d'information. Les premiers travaux en RI qualifiés d'approche *classique*, sont apparus dans les années soixante, ils ont été focalisés à résoudre des problèmes principalement liés à la représentation de l'information, l'évaluation de requêtes ainsi que l'évaluation des performances de recherche. Toutefois, la RI *classique*, étant une approche orientée système, la différence de vocabulaire utilisé pour l'expression des contenus des documents et des besoins en information n'est pas prise en compte dans le processus d'appariement document-requête. Ce défaut d'appariement engendre une dégradation des performances de recherche. Ainsi, le problème qui se pose actuellement n'est plus tant la disponibilité de l'information mais la capacité d'accès et de sélection de l'information répondant aux besoins précis d'un utilisateur, à partir des représentations qu'il perçoit. Ces facteurs ont soulevé des défis majeurs pour les tâches de collecte et de gestion de l'information, le stockage, la transmission et la recherche efficace de l'information [1].

Des moteurs de recherche sont apparus. À partir d'une requête simple, ils parcourent leur mémoire pour trouver le plus rapidement possible, les documents correspondant aux souhaits de l'utilisateur. Google est certainement une réussite américaine en ce domaine puisqu'il possède en mémoire l'image de plusieurs milliards de documents sous forme d'index. Cependant, Il fournit souvent en réponse à une requête des milliers de documents en un temps inférieur à la seconde. Ce sont là précisément les points forts et les points faibles de ce type de méthode : l'utilisateur est submergé par des milliers de réponses présentées "en vrac", dont une partie ne correspond pas à la requête, une autre correspond à des pages qui n'existent plus, sans parler de nombreux types de documents qui ne sont pas pris en compte dans la mémoire du moteur. Dans bien des cas, l'utilisateur passe un temps assez long à analyser les résultats du moteur, sans garantie de résultats.

Le changement du Web tant dans le volume d'information que dans le nombre d'utilisateurs amène de nouveaux problèmes et ouvre de nouvelles perspectives. D'un côté, on

peut affirmer que la communauté scientifique en recherche d'information a apporté des solutions avérées pour améliorer sans cesse l'ensemble des fonctions d'un système de recherche d'information. On cite plus particulièrement les résultats probants obtenus dans le domaine de la modélisation de l'information, de l'évaluation de requêtes et de la visualisation des résultats des requêtes. D'un autre côté, force est de constater que les solutions traditionnelles connues ne peuvent conserver leur degré d'efficacité, voire leur faisabilité, dans le contexte actuel où la problématique de la recherche d'information a pris une nouvelle dimension. Cette inadéquation mérite des investigations qui permettraient, d'une part, de transposer en partie des acquis considérables en la matière (modèles et mécanismes théoriques développés en recherche d'information) au contexte actuel et, d'autre part, de dresser une étude prospective qui éclairerait les voies de recherche à investir pour faire face à ce nouveau « *passage à* l'échelle » [2].

L'apparition des terminaux mobiles intelligents a fait évoluer considérablement le rapport de l'humain à l'outil informatique. Les assistants personnels ont longtemps cherché leur voie, mais maintenant que leurs capacités commencent à être comparables à celles de véritables ordinateurs en modèles réduits, on imagine bien la place prépondérante qu'ils pourront prendre dans quelques années.

Ces nouvelles conditions d'usage, existantes ou à venir, forment le socle des systèmes *informatique ubiquitaire*.

L'apparition du *Web ubiquitaire* marche de paire avec le besoin idéal de l'utilisateur en information désirant l'obtenir dans n'importe quel moment, avec n'importe quel terminal (ordinateur, PDA, Mobile,…), et dans n'importe quel endroit le souhait (à la maison, sur un bateau, sur un avion). L'objectif premier du *Web ubiquitaire* est de permettre de répondre au besoin de l'utilisateur en fonction de son contexte. Les systèmes ubiquitaires peuvent être vus comme étant l'étape suivante après les systèmes mobiles et les systèmes distribués.

Les avancées en réseaux sans fil et mobiles d'une part, et en terminaux mobiles, d'autre part, ont ouvert une perspective nouvelle pour les systèmes de recherche d'information. Le *Web ubiquitaire* est ainsi né de l'ambition de fournir un accès aux informations en tout lieu, à tout moment, et sous une forme et un contenu correspondant à la situation de l'utilisateur. *Mark Weiser* [3] a donné sa vision de *l'informatique ubiquitaire* qui va encore plus loin en associant à tout objet de la vie quotidienne un outil de calcul et de communication, et rendant son intelligence transparente à l'utilisateur.

Le *Web ubiquitaire* cherche donc à élargir les capacités des navigateurs Web afin de permettre de nouveaux types d'applications Web, notamment celles nécessitant une coordination avec d'autres terminaux et une adaptation dynamique au contexte d'utilisation. Ces applications seront capables d'exploiter les services en réseau pour élargir les capacités des terminaux. Les utilisateurs pourront alors se concentrer sur ce qu'ils font et non sur les terminaux eux-mêmes. La situation est actuellement paradoxale : la masse d'informations est telle que l'accès à une information pertinente, adaptée aux besoins spécifiques d'un utilisateur donné, devient à la fois difficile et nécessaire. En clair, le problème n'est pas tant la disponibilité de l'information mais sa pertinence relativement à un contexte d'utilisation particulier.

Les besoins sont ainsi grandissants. Dans ce cadre, la *RI contextuelle* émerge comme un domaine à part entière : elle pose des problématiques nouvelles allant de la modélisation du

contexte jusqu'à la modélisation de la pertinence cognitive en passant par la modélisation de l'interaction entre un utilisateur et un Système de Recherche d'Information (SRI). De ce fait, au delà de la mise en œuvre des techniques d'*adaptation au contexte*, les travaux s'orientent actuellement vers la révision de la chaîne d'accès à l'information dans la perspective d'intégrer l'utilisateur dans l'ensemble des phases de recherche, dans le but de lui délivrer l'information pertinente adaptée à son contexte et ses préférences, répondant à ses besoins précis. Dès lors, l'accès à l'information tend vers une nouvelle définition [4]: "*Combine search technologies and knowledge about query and user context into a single framework in order to provide the most appropriate answer for a user's information needs*".

La tendance dominante du *Web ubiquitaire* alors, est de développer une vision de la disponibilité et l'accessibilité de l'information via l'Internet du futur et de proposer des solutions pour le construire et le maîtriser.

Problématique

Les systèmes de RI sur le *Web ubiquitaire* soulèvent de nouvelles problématiques informatiques, que l'on peut classer dans deux niveaux.

- Les questions relevant du réseau constituent le premier niveau. Il faut définir de nouveaux protocoles capables de gérer les caractéristiques spécifiques des terminaux mobiles, il faut gérer de nouvelles notions comme les réseaux de capteurs, ou encore permettre l'interconnexion de systèmes hétérogènes en termes de support du signal ou de protocole. Á ce niveau, les solutions qui ont été apportées par l'approche agent mobile s'adaptent considérablement aux questions relevées par les réseaux, Surtout, avec la récente explosion des *applications Web ubiquitaire*, nous assistons à l'émergence de nouveaux modèles pour la structuration d'applications réparties et le calcul mobile. Cependant, la principale faiblesse de ces solutions à base d'agent mobile est la complexité de la structuration de modèle de comportement d'agent mobile, mais aussi de l'augmentation sans cesse croissante des informations véhiculées à travers les réseaux, à cela, vient s'ajouter un surcroit par les informations du contexte d'utilisation qu'un *système de recherche d'information dans le Web ubiquitaire* doit les traiter.

- Le second niveau, concerne la pertinence des informations retournées. Il faut permettre l'accès aux informations dans un environnement d'exécution ubiquitaire fortement dynamique et avec un rendu efficace dans le contexte de l'utilisateur. Face à ceci, la RI *sensible au contexte* constitue un domaine indispensable pour assurer l'adaptation des informations retournées à l'utilisateur en fonction de la variabilité du contexte de recherche.

Toutes ces problématiques ont reçu des solutions avérées dans des environnements fixes et potentiellement distribués, mais les contraintes de mobilité et de faiblesse des terminaux mobiles demandent de reconsidérer les solutions existantes. De plus, il est impossible de proposer des solutions qui satisfassent tout type de système ubiquitaire. Il est nécessaire de définir des mécanismes d'adaptation.

Ce mémoire s'intéresse principalement à deux aspects : d'une part, celui de la *sensibilité au contexte*, quoi doit fournir un système de recherche d'information dans le *Web ubiquitaire*, en fait, cet aspect inclut, l'adaptation de l'information retournée à l'utilisateur en fonction de la mobilité et de changement très important de l'environnement ubiquitaire d'exécution, y compris, bien sure, le profil de l'utilisateur (composé spécifiquement des préférences de

l'utilisateur qui peuvent être applicables en tenant compte du *contexte de recherche* de la session en cours) ainsi que les caractéristiques et contraintes techniques de son dispositif d'accès. Et d'autre part, celui de l'augmentation de la charge du réseau. Cet aspect englobe d'une part, la croissance massive des informations véhiculées à travers les réseaux, à cela, vient s'ajouter un surcroit par les informations du contexte d'utilisation qu'un *système de recherche d'information dans le Web ubiquitaire* doit les traiter.

Contribution

Le travail effectué au cours de ce mémoire s'inscrit dans le cadre de *la recherche d'information dans le Web ubiquitaire*, nous proposons une architecture de recherche à base de rôle et agent mobile. En partant du principe très général suivant : plutôt qu'un agent mobile de recherche se déplace tout le temps, à travers différents réseaux, avec ces outils de recherche, il serait envisageable de déployer certains outils sur les sources d'informations. Nous proposons une approche qui peut simuler cette situation par la structuration de modèles de comportements d'agents mobiles, nous adoptons le concept de rôle pour organiser le comportement interne de l'agent, lui permettant de se déplacer à travers les réseaux avec un ensemble restreint de rôles de recherche étant donné qu'à tout nœud il existe une base spécifique de rôles de recherche. De cette façon, l'agent visitant un nœud sélectionne sur la base de ses paramètres de recherche les rôles adéquats, à fin d'enrichir son comportement et de même, il peut contribuer à l'évolution de cette base de rôles de recherche par déposition de nouveaux rôles, ou bien par adaptation des rôles existants. Ceci réduit la charge au niveau de l'agent mobile et donc le trafic de son déplacement, et par conséquent, réduction de la charge totale des réseaux.

Nous proposons l'utilisation de l'agent mobile comme outil de génie logiciel et plus précisément, Nous présentons un modèle d'agent mobile à base de rôle adaptable, sensible au contexte dynamiquement, pour s'adapter aux variations de son environnement de recherche ubiquitaire. En fait, ce modèle d'agent nourri des travaux qui ont été réalisés dans ce domaine et qui ont des succès notable dans le domaine de la recherche d'information. Vu que l'agent se voie comme étant un composant, le comportement de l'agent mobile proposé est construit principalement à partir d'un assemblage de rôles de recherche de granularité faible, il s'agit, en fait, de micro-composants remplaçables en fonction de la variabilité du contexte de la recherche. Ce style d'architecture d'agent flexible doit faire face à la forte dynamicité de son environnement ubiquitaire de recherche, dans lequel, différentes stratégies d'assemblages de rôles permettent d'engendrer un comportement sensible au contexte de la recherche. En plus, nous préconisons de prendre en compte le contexte de la recherche comme étant une multitude de niveau de caractéristiques qui influence directement la qualité et la vitesse d'extraction des informations.

De cette façon, ce nouveau modèle de SRI dans le *Web ubiquitaire* va pouvoir être sensible au contexte dynamique de l'environnement d'exécution ubiquitaire, En se basant essentiellement à l'utilisation du concept abstrait du rôle de recherche, comme brique constrictrice de l'algorithme de recherche.

Organisation du mémoire

Ce mémoire est structuré de la manière suivante :

- *Chapitre 1* : l'objectif de ce chapitre est d'apporter une présentation et une étude du domaine du *Web ubiquitaire*, ses spécificités, ses particularités et ses technologies, ainsi

que, les verrous technologiques à lever et les défis du domaine. Pour cela, après avoir donné la définition de *l'informatique ubiquitaire* et les spécificités du *Web ubiquitaire*, nous présentons les technologies du *Web ubiquitaire*. Nous proposons ensuite un panorama des *applications Web ubiquitaire* existantes en mettant particulièrement l'accent sur les *applications Web ubiquitaire sensibles au contexte*. En montrant l'importance des *intergiciels* dans la résolution de l'hétérogénéité matérielle et logicielle des dispositifs informatiques dans un environnement ubiquitaire.

- *Chapitre 2* : Le but de ce chapitre est de faire un état de l'art des techniques de recherche d'information usuelles et l'émergence de la recherche *d'information contextuelle*. Pour cela, nous présentons les techniques de RI *classiques*, nous détaillons les principes de base de l'architecture des systèmes de recherche d'information SRI en général, avec une présentation des différents modèles de SRI, puis nous examinons en particulier, les systèmes de recherche d'information distribuée SRID. En suite, Nous détaillerons une variété d'approches et algorithmes de recherche d'information sur Internet utilisés à grande échelle dans les outils de recherche. Nous abordons ici les techniques utilisées dans les approches *multi-agents* et des *heuristiques* utilisées dans les *algorithmes génétiques*, plus précisément, les algorithmes basés sur le comportement des *fourmis*, Pour en suite, les utiliser sous forme de fragments de code de recherche, dans le but, également, de profiter de l'apport de ces *heuristiques* algorithmique dans le processus de la RI sur des serveurs d'information distribués et hétérogènes. Nous examinons également des exemples d'adaptation de ces *heuristiques* au Web et à la recherche d'information.

Ce chapitre apporte aussi un éclairage sur les problématiques de la recherche d'information *contextuelle*. Nous présentant les diverses motivations qui ont conduit à l'émergence de la RI *contextuelle*, les principaux travaux portant sur les approches, modèles et techniques de RI *contextuelle*.

- Dans le *Chapitre 3* : nous discutons les challenges actuels dans Web, en visant principalement les *problématiques des grands volumes d'information*, l'explosion de l'utilisation de la Toile. Dans ce cadre, nous visons l'identification des verrous technologiques liés à la croissance et à la diversification des contenus des systèmes de recherche d'information et aux processus de recherche d'information associés.

- Le C*hapitre 4* : développe une *approche de recherche d'information dans le Web ubiquitaire à base de rôle et d'agent mobile*. Pour cela, nous présentons différentes technologies à la base de construction d'applications *Web ubiquitaire* sensible au contexte: *intergiciel*, *composant logiciel*, *agent mobile*, En effet, une adaptation de ces technologies peut apporter une avance pour concevoir un modèle de SRID sensible au contexte fortement dynamiques dans le *Web ubiquitaire*.

À la fin de ce document, une conclusion générale fait le bilan sur l'ensemble de ces travaux de recherche et indique des perspectives et des défis de la recherche d'information dans le domaine du *Web ubiquitaire*.

Ce mémoire a fait l'objet d'une communication dans la conférence internationale ICAI09 « International Conference on Applied Informatics, 2009 », intitulée : « *Recherche d'information Distribuée sur le Web : Approche à base de rôles et agents mobiles* ».

Chapitre 1

Le Web ubiquitaire

1.1 Introduction

En 1991, Mark Weiser présentait sa vision de l'informatique du 21ème siècle en établissant les fondements de ce que nous appelons aujourd'hui *l'informatique ubiquitaire* [3]. *L'informatique ubiquitaire* se définit comme un environnement saturé de dispositifs informatiques susceptibles de coopérer de façon autonome et transparente afin d'améliorer l'interactivité et/ou l'expérience de l'utilisateur [5]. L'importance de cette vision consiste à mettre en disponibilité les nouvelles technologies de l'information et de la communication au service des utilisateurs. L'objectif est de permettre à ces derniers d'accéder aux différentes fonctionnalités offertes par les divers dispositifs informatiques hétérogènes présents dans leur environnement immédiat, à partir de n'importe quel moment, n'importe quel dispositif informatique comme par exemple leur ordinateur portable, leur téléphone portable ou leur PDA (*Personal Digital Assistant*).

Actuellement, l'Internet est utilisé de manière intensive pour accéder et échanger de l'information. À chaque accès, les utilisateurs souhaitent obtenir l'information la plus pertinente par rapport à leurs caractéristiques et à celles de leur(s) dispositif(s) d'accès. Cependant, l'utilisateur peut obtenir lors d'un accès, une grande quantité d'information qui n'est pas toujours pertinente, ni toujours supportée par son *Dispositif Mobile* (*DM*). Au cours de la dernière décennie, l'accès aux informations par les *Systèmes de recherche d'Information dans le Web* (*SRIW*) a beaucoup changé en raison de nombreux facteurs [6]:

- La mobilité intrinsèque de l'utilisateur;
- Les avancées techniques dans le domaine des *DM*, par exemple *PDA*, téléphones, ordinateurs portables ;
- Le besoin de prendre en compte les capacités matérielles et logicielles réduites des *DM* (par exemple, taille de l'écran, mémoire, disque dur) ;
- La nature multimédia des données échangées.

Le *Web ubiquitaire* est vu alors, comme une synthèse de l'Internet et *l'informatique ubiquitaire* qui étend le Web existant l'importance de la coordination de *dispositifs mobiles*, la *sensibilité au contexte*, l'adaptation à l'hétérogénéité des réseaux et des sources d'information. *Le Web ubiquitaire* cherche donc à élargir les capacités des navigateurs Web afin de permettre de nouveaux types d'applications Web, notamment celles nécessitant une coordination avec d'autres terminaux et une adaptation dynamique au contexte de l'utilisateur. Ces applications seront capables d'exploiter les services en réseau pour élargir les capacités des terminaux. Les utilisateurs pourront alors se concentrer sur ce qu'ils font et non sur les terminaux eux-mêmes. La mobilité des applications Web ubiquitaire tend à permettre à l'utilisateur de continuer à naviguer ou à consulter l'information tout en passant en douceur d'un terminal à l'autre et d'un réseau à l'autre.

Actuellement, les *DM* peuvent être utilisés afin d'accéder au Web mais aussi pour stocker des petites quantités d'information (par exemple, des fichiers) ou encore pour exécuter des applications simples. Afin de permettre l'accès à l'information et le partage de ressources entre des utilisateurs dans des environnements nomades[1], il est nécessaire de disposer d'architectures et de technologies ayant un grand potentiel communicatif. Ce potentiel est une des caractéristiques principales des *systèmes ubiquitaires.*

Par ailleurs, les concepteurs de *SRIW* doivent disposer de mécanismes et d'architectures afin de stocker, récupérer et délivrer efficacement l'information la plus pertinente, Le

1- Un *environnement nomade* peut être représenté en termes d'information contextuelle, telle que le moment de connexion, la localisation, les caractéristiques du *DM*, et les tâches de l'utilisateur, entre autres.

changement sous-jacent consiste à fournir aux utilisateurs de l'information utile basée sur une *recherche sensible au contexte* et un filtrage en fonction du *contexte de la recherche* et un *affichage approprié e*n proposant un accès à travers des *DM*, quels que soient le lieu et le moment. Cela est l'idée sous-jacente de du *Web ubiquitaire*. Les concepteurs doivent également tenir compte des capacités réduites de ce type de dispositifs (par exemple, la taille de l'écran, la mémoire, le disque dur) pour un affichage approprié sur le *DM* de l'utilisateur.

Les aspects à considérer lors de l'adaptation de l'information à l'utilisateur dans l'environnement du *Web ubiquitaire* sont:

- Tout d'abord, les changements de localisation de l'utilisateur qui peuvent induire des changements en termes d'accès et de besoins d'information. La localisation de l'utilisateur peut être obtenue à l'aide d'un dispositif *GPS* ou d'une des méthodes proposées par Nieto-Carvajal *et al.* [8] : le « *Signal Strength* », le « *SNMP* (*SimpleNetwork Management Protocol*) », etc;
- Ensuite, les préférences d'un utilisateur, relatives aux activités qu'il souhaite accomplir dans le système, aux résultats attendus de ces activités et à la manière dont il souhaite les voir affichés sur son dispositif d'accès, peuvent être appliquées en fonction des caractéristiques contextuelles de la session en cours. Pour cette raison, un Système de Recherche d'Information *SRI* doit disposer de données qui portent d'une part sur le *contexte d'utilisation* et d'autre part, sur les *préférences de l'utilisateur*. Les données sur le *contexte d'utilisation* permettent notamment (mais pas seulement) d'obtenir une description des conditions (temporelles, spatiales, matérielles, *etc*.) dans lesquelles l'utilisateur accède au *SRI*. Les données sur les préférences visent à traduire ce que l'utilisateur souhaiterait obtenir du système par rapport à différents axes (les fonctionnalités, le contenu, l'affichage, *etc*.).
- Finalement, les caractéristiques et contraintes techniques du dispositif d'accès de l'utilisateur nomade peuvent produire des problèmes d'affichage de l'information. Il faut donc également tenir compte des préférences de l'utilisateur par rapport à l'affichage de l'information sur son dispositif d'accès [7].

Ce mémoire s'intéresse principalement à deux aspects : d'une part, celui de la *sensibilité au contexte* (context-awareness), quoi doit fourni un système de recherche d'information dans le *Web ubiquitaire*, en fait, cet aspect inclut, l'adaptation de l'information retournée à l'utilisateur en fonction de la mobilité et de changement très important de l'environnement ubiquitaire d'exécution, y compris, bien sure, le profil de l'utilisateur (composé spécifiquement des préférences de l'utilisateur qui peuvent être applicables en tenant compte du *contexte d'utilisation* de la session en cours) ainsi que les caractéristiques et contraintes techniques de son dispositif d'accès. Et d'autre part, celui de l'augmentation de la charge du réseau. Cet aspect englobe d'une part, la croissance massive des informations véhiculées à travers les réseaux, à cela, vient s'ajouter un surcroit par les informations du contexte d'utilisation qu'un *système de recherche d'information dans le Web ubiquitaire* doit les traiter.

Pour cela, nous présentons tous d'abord, dans la section 2 la vision de *l'informatique ubiquitaire* par le W3C et tel qu'elle a été décrite par Mark Weiser, le propriétaire de l'idée.

Dans la section 3, nous citons de manière synthétique les définitions du Web 3.0 par les pointures de l'Internet, ses principaux concepts et les technologies qui feront le Web de demain. Pour ensuite,

Dans la section 4, on peut tirer les spécificités, tendances actuelles et les technologies du *Web ubiquitaire*.

La section 5 est consacrée à l'étude des *applications Web ubiquitaire sensibles au contexte*, Ensuite, nous présentons les mécanismes utilisés dans les différentes phases de prise en compte du contexte à savoir l'acquisition des informations de contexte, leur modélisation et l'adaptation au contexte. Enfin, nous dressons un état de l'art de travaux de recherche réalisés sur l'adaptation au contexte. Nous montrons dans la conclusion l'importance des *intergiciels* sensibles au contexte dans la simplification du développement des SRI adaptables pour le *web ubiquitaire*.

1.2 Informatique ubiquitaire

De nos jours, les applications informatiques doivent permettre à leurs utilisateurs de consulter des données n'importe quand, n'importe où, à travers leurs dispositifs d'accès. C'est l'idée sous-jacente de l'*informatique ubiquitaire* telle qu'elle a été décrite par Mark Weiser [3].

Nous présentons un scénario illustratif du principe de l'informatique ubiquitaire, dans lequel un utilisateur Mohamed interagit avec différents services de son environnement :

- Mohamed participe à une réunion qu'il doit malheureusement quitter plus tôt pour se rendre à un rendez-vous au bureau de Youcef situé à l'autre bout de la ville.
- Dès que Mohamed quitte la réunion, le flux audio de cette dernière est automatiquement transféré sur son téléphone mobile.
- Mohamed prend alors sa voiture. L'ordinateur de bord détecte la conversation téléphonique en cours et la rediffuse immédiatement sur les haut-parleurs du véhicule jusqu'à ce que la conversation se termine. Grâce à un service d'information trafic auquel le véhicule a accès via une infrastructure réseau du réseau routier,
- Un message s'affiche sur l'écran du tableau de bord indiquant à Mohamed l'existence d'un accident situé à 3 km. Ne pouvant pas se rendre en voiture à son rendez-vous,
- Mohamed décide alors de se garer et de poursuivre à pieds. Ne connaissant pas le quartier, il consulte via son PDA un service d'information local, accessible dans son environnement immédiat grâce au point d'accès Wi-Fi de la ville, qui lui affiche le plan du quartier et l'itinéraire le plus court pour se rendre à son rendez-vous.
- Arrivé à temps à ce dernier, Mohamed sort son ordinateur portable et le connecte au réseau local, afin de découvrir un service permettant de projeter sur un écran la présentation de son projet et termine son rendez-vous en synchronisant des documents de son portable avec ceux d'un dispositif de stockage de données appartenant à Youcef.

Ce scénario met en valeur plusieurs concepts de base de *l'informatique ubiquitaire*, dont celui de permettre aux utilisateurs d'accéder aux différents services offerts par l'environnement ubiquitaire, n'importe où et à tout instant et à travers différents dispositifs mobiles.

Un point important à souligner est que le développement de ces différents concepts de *l'informatique ubiquitaire* met en valeur une condition importante est que les différents dispositifs, terminaux ou services offerts par l'environnement de l'utilisateur soient capables d'interagir indépendamment de leurs hétérogénéités matérielle et logicielle. Dans notre exemple, le PDA, le téléphone mobile, les dispositifs informatiques embarqués, le dispositif

de stockage de données sont soumis potentiellement à des contraintes matérielles différentes et sont supportés par des systèmes logiciels hétérogènes. Ainsi, un des défis de l'informatique ubiquitaire est donc d'intégrer, de façon transparente pour l'utilisateur, des technologies hétérogènes [9].

Le *W3C*[2] définit l'*informatique ubiquitaire*, comme un paradigme émergeant de l'*informatique personnelle* (« *Personal Computing* ») caractérisé par :

- L'utilisation de dispositifs légers, manipulables et sans fil ;
- La nature pervasive[3] et sans fil des dispositifs qui demande des architectures réseaux supportant une configuration automatique;
- La haute distribution, hétérogénéité, mobilité et autonomie de l'environnement. Koch *et al.* [10] mentionnent plusieurs changements introduits par *l'informatique ubiquitaire*. En premier lieu, on trouve un *changement d'infrastructure* qui nécessite la construction de technologies robustes tant de matérielles que logicielles facilitant la connectivité mobile, l'identification de la localisation, la découverte de services, la tolérance aux fautes, *etc*.

Également, on peut évoquer le *changement de services* qui concerne la manière dont l'infrastructure disponible est utilisée afin de fournir de nouveaux services à l'utilisateur. À propos des dispositifs d'accès propres à l'*informatique ubiquitaire* (c'est-à-dire, des *Dispositifs Mobiles, DM, voir la figure 1.1*),

Fig.1.1 Diversité des dispositifs mobile (*DM*)

Rahwan *et al.* [11] soulignent certaines de leurs caractéristiques et contraintes techniques des *DM* telles que :

- le stockage limité ;
- la puissance de traitement limitée (sur un *DM*, ne sont possibles que l'utilisation de petites bibliothèques et l'exécution de petites applications) ;
- l'hétérogénéité de la représentation des données ;
- le manque de standards de spécification de plans d'exécution de tâches (ce qui réduit les possibilités de partager les données et de communiquer entre des applications) ;
- le besoin d'adjoindre des composants matériels embarqués (pour les dispositifs d'accès, par exemple, les capteurs de lumière, de son, de localisation, des lecteurs de code barres) pour la prise en compte des caractéristiques contextuelles de l'utilisateur (l'information sur les changements des caractéristiques contextuelles doit être fournie par l'utilisateur ou doit être détectée d'une manière automatique, par exemple, la localisation peut être détectée à travers un capteur *GPS*).

---

2 - Le terme « *pervasif* » fait référence à la définition d'*informatique pervasive*. Celle-ci fournit un accès approprié à l'information et aux applications à travers des dispositifs qui possèdent la capacité de fonctionner au moment et à l'endroit nécessaires.

3 - http://www.w3.org/

Compte tenu des caractéristiques des *DM*, Hristova *et al.* [12] donnent quelques conseils pour concevoir et développer des applications exécutées sur des *DM* : concevoir des clients légers, utiliser des logiciels dont l'exécution mobilise peu de ressources mémoire, utiliser des mécanismes minimisant la latence et améliorant la performance de la transmission des données sur des réseaux sans fil, définir un plan d'exécution des applications (et leurs tâches) sur le *DM*, concevoir des interfaces simples, *etc*.

Une application exécutée sur des *DM* doit posséder la capacité d'une part, d'interagir avec l'utilisateur et, potentiellement, avec d'autres applications et, d'autre part, de raisonner sur les objectifs en termes d'utilisation de l'utilisateur nomade et la manière dont il peut les atteindre [11]. De telles applications demandent des architectures réseaux capables de supporter une configuration automatique et compatible avec les caractéristiques de l'*environnement de l'informatique ubiquitaire* telles que l'hétérogénéité, la mobilité, l'autonomie, la haute distribution, *etc*. De tels *environnements* sont définis par Pirker *et al.* [13] comme étant un réseau dynamique distribué de dispositifs et systèmes embarqués qui peuvent interagir avec des humains afin de satisfaire leurs besoins et de leur fournir une variété de services d'information, de communication, et de collaboration.

Dans un environnement ubiquitaire, Les utilisateurs peuvent changer d'endroit ce qui éventuellement peut modifier l'exécution de leurs tâches en fonction des caractéristiques contextuelles de l'endroit où ils se trouvent (par exemple, l'utilisateur peut disposer de conditions de réseau différentes, ou il peut changer de dispositif d'accès selon sa localisation). La connectivité du réseau ne peut pas être assurée à tout moment et à tout endroit, et ses caractéristiques contextuelles peuvent modifier les besoins et préférences de l'utilisateur (par exemple, l'utilisateur peut préférer obtenir l'information demandée sous forme d'image à la place d'une vidéo à cause de la vitesse de transmission). Dans ces conditions, Calisti *et al.* [14] proposent d'exploiter des paramètres d'accès de réseau tels que la bande passante, les délais et le temps moyen d'accès, pour déterminer la « *meilleure* » chaîne d'accès possible compte tenu des politiques des fournisseurs d'accès, des préférences de l'utilisateur et des caractéristiques du dispositif d'accès, des besoins de l'application s'exécutant sur le dispositif d'accès, *etc*.
Plusieurs applications de l'*informatique ubiquitaire* tiennent compte des caractéristiques contextuelles et de celles de l'utilisateur afin de leur adapter l'information. Quelques unes de ces applications de l'*informatique ubiquitaire* sont présentées dans la suite.

Des applications telles que *WAY* (« *Where Are You* ») [15], *Ad-Me* (« *Advertising for mobile e-commerce user* ») [12] et *Gulliver's Genie* [16] ont été développées afin d'adapter l'information à la localisation de l'utilisateur, compte tenu des caractéristiques de son *DM* (en général, des *PDA* et des téléphones portables). *WAY* [15] est un système pour la synchronisation et la prise de rendez-vous entre utilisateurs nomades à l'aide de leurs *PDA*. *Ad-Me* [12] est un système créé pour l'office du tourisme de Dublin qui fournit aux utilisateurs nomades des services de publicité (des magasins, des restaurants) à partir de leur localisation et de leurs préférences. *Gulliver's Genie* [16] est un guide touristique qui se configure selon la localisation et l'orientation de déplacement de l'utilisateur et personnalise l'information tenant compte de son profil (par exemple, *Gulliver's Genie* fait un plan de la ville en soulignant tous les musées si les sites touristiques les plus visités par l'utilisateur sont des musées).

L'*informatique ubiquitaire* a besoin d'architectures et de technologies dotées de fortes capacités de communication, d'un réseau ubiquitaire qui sert à relier les différents réseaux et

serveurs d'informations hétérogènes existantes et également les différents dispositifs mobiles à la main des utilisateurs. La section suivante sera consacrée, à la vision des technologies en émergences considérable, ses principaux concepts qui feront le Web de demain, il s'agit du Web 3.0, la source d'information de futur indispensable dans la vie quotidienne des utilisateurs.

1.3 Émergence du Web

Voici de manière synthétique les définitions du Web 3.0 par les pointures de l'Internet, ses principaux concepts et les technologies qui feront le Web de demain. Le Web 3.0 en un clin d'œil... sur la figure ci-dessus.

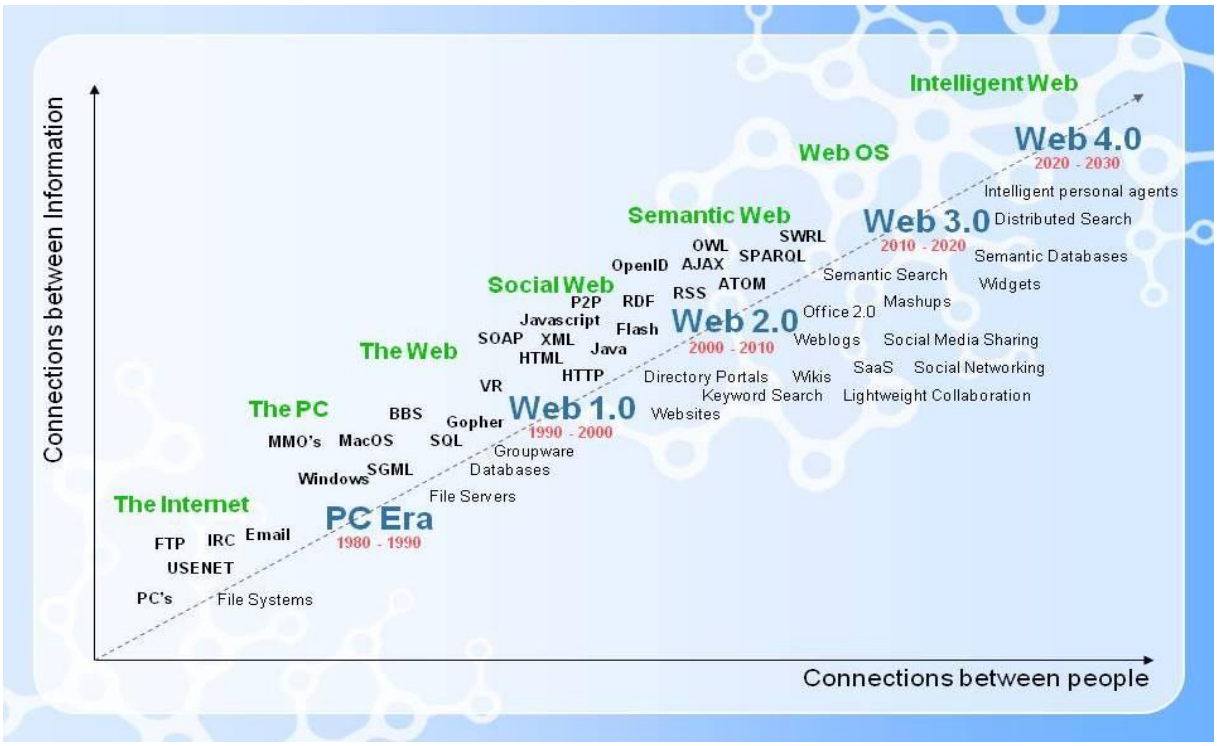

Fig.1.2 Chronologie de l'évolution du Web.

Tim Berners-Lee, co-inventeur du World Wide Web, dirigeant du W3C	Le Web 3.0 est le web sémantique - TBL pose donc des problèmes de savoir-vivre autant que de savoir-faire.
Vinton Cerf, inventeur d'Internet, Chef Internet chez Google	Internet (3.0) des objets. L'internet des objets permettra de masquer la complexité des technologies à l'œuvre. Tout se passera dans les coulisses
Tim O'Reilly, co-inventeur et propriétaire de l'expression "Web 2.0"	C'est la rupture du paradigme clavier/écran et le monde dans lequel l'intelligence collective émerge non pas des gens en train de taper sur un clavier mais de la mise en œuvre de nos activités par des instruments.

Eric Schmidt, PDG de Google, administrateur de Apple	Le Web comme une grille de calcul universelle remplaçant le système d'exploitation et le disque du PC.
Nova Spivack, patron de Radar Networks et co-auteur de la définition du web 3.0 sur Wikipedia	Le web 3.0 c'est la troisième décade du web (2010-2020). Il n'est pas synonyme du web sémantique (il y aura des évolutions technologiques importantes). Mais il est largement caractérisé par cette sémantique. Il représente une époque au cours de la quelle les couches profondes du web seront améliorées.
Nicholas Carr, auteur de "The Big Switch" et de "Does IT matter?"	Le web 3.0 est la désintégration des données numériques et des logiciels dans des composants modulaires qui, grâce à l'utilisation d'outils simples, peuvent être réintégrés à la volée dans de nouvelles applications ou fonctions, autant par les machines que par les hommes.
Fred Cavazza, consultant indépendant	C'est une expérience immersive étendue. Le futur du web, c'est l'internet 3.0. Il est possible d'identifier de nouveaux maillons pour une chaîne de valeur qui ne se limitera plus au web. L'expérience d'achat de l'internaute sera d'une part plus immersive mais surtout plus étendue à d'autres domaines.
Olivier Ertzscheid, maître de conférence en sciences de l'information	Le World Life Web, le troisième âge documentaire, pose comme principale question celle de la sociabilité et du caractère indexable, remixable de notre identité numérique et des traces qu'elle laisse sur le réseau. En quoi suis-je un document ?
Sramana Mitra, entrepreneur et consultant	WEB 3.0 = (4C+P+VS). Content, Commerce, Community, Context + Personalization + Vertical Search

Les mots-clés du web 3.0

Web sémantique	Caractérise un ensemble de technologies visant à rendre le contenu des ressources du web accessible et utilisable par les programmes et les agents logiciels, grâce à un système de méta-données formelles
Web symbiotique	Désigne un web accessible à partir d'une infinité de moyens dont il n'est pas nécessaire d'avoir conscience. Il devient notre environnement, il est partout, nous ne sommes plus tributaire du triptyque "*laptop, desktop, palmtop*"
Web ubiquitaire	On parle ici aussi d'informatique ubiquitaire, de réseau pervasif, d'intelligence ambiante. Cela fait référence à l'utilisation de minuscules systèmes numériques communiquant spontanément les uns avec les autres et qui, grâce à leurs dimensions très réduites, seront intégrés dans les objets de la vie quotidienne, favorisant ainsi l'accès aux informations dont on a besoin partout et à tout moment
Internet d'objets	désigne une extension d'Internet à des objets et à des lieux dans le monde réel. Il prolonge l'Internet au monde réel en fixant des étiquettes munies de codes ou d'URL aux objets et aux lieux. Ces étiquettes pourront être lues par un dispositif mobile. M2M (machine to machine) désigne les communications d'objet à objet, de machine à machine.
Web OS	Plate-forme logicielle utilisant un navigateur pour interagir mais ne dépendant pas d'un système d'exploitation particulier. Se présente comme une imitation d'un bureau ou de l'environnement graphique d'un OS.

1.4 Le Web ubiquitaire

Le succès du World Wide Web sur l'ordinateur de bureau, nous n'en sommes qu'au début dans l'exploitation du potentiel que procure un éventail de terminaux toujours croissant. Les avancées en réseaux sans fil et mobiles d'une part, et en terminaux mobiles, d'autre part, ont ouvert une perspective nouvelle en systèmes de recherche d'information. Le Web ubiquitaire est ainsi né de l'ambition de fournir un accès aux informations en tout lieu, à tout moment, et sous une forme et un contenu correspondant à la situation de l'utilisateur. L'informatique ubiquitaire [3] va encore plus loin en associant à tout objet de la vie quotidienne un outil de calcul et de communication, et rendant son intelligence transparente à l'utilisateur.

Le Web ubiquitaire est vu alors, comme synthèse de l'Internet et l'informatique ubiquitaire qui étend le Web existant l'importance de la coordination de dispositifs mobiles, la sensibilité au contexte, l'adaptation à l'hétérogénéité des sources d'information. Le Web ubiquitaire cherche donc à élargir les capacités des navigateurs Web afin de permettre de nouveaux types d'applications Web, notamment celles nécessitant une coordination avec d'autres terminaux et une adaptation dynamique au contexte de l'utilisateur. Ces applications seront capables d'exploiter les services en réseau pour élargir les capacités des terminaux. Les utilisateurs pourront alors se concentrer sur ce qu'ils font et non sur les terminaux eux-mêmes.

Ce concept en émergence traduit un accès permanent, ubiquitaire et auto-configurable à un Internet plus large, ambiant, qui connectera des milliards de composants divers. Ces réseaux diffus auront la capacité de se configurer automatiquement, pour créer un réseau spécifique ou servir les besoins d'une communauté dynamique dans le temps et l'espace. Ils devront donc s'adapter aux conditions physiques et aux performances de l'environnement ; de même, les équipements pourront choisir leur réseau d'accès en fonction de paramètres tels que le débit, la qualité ou la consommation électrique.

L'objectif du Web ubiquitaire est de développer une vision de la disponibilité et l'accessibilité de l'information via l'Internet du futur et de proposer des solutions pour le construire et le maîtriser.

1.4.1 Tendance actuelle du Web ubiquitaire

Les applications doivent respecter les caractéristiques spécifiques aux environnements mobiles. Ces appareils devront permettre de fournir un accès à des services et des ressources avec des entrées et sorties limitées, des capacités de stockage réduites mais qui nécessite une grande puissance de calcul. Cela oblige donc à adapter la conception des applications aux caractéristiques des réseaux et à l'adaptation au contexte géographique.

1.4.1.1 Le Web des objets communicants issus de la vie quotidienne

Une autre vision de cette omniprésence du Web. Appelé aussi l'ubiquité numérique. Cette notion fait référence à l'utilisation de minuscules systèmes numériques intégrés dans les objets de la vie quotidienne, favorisant là aussi l'accès aux informations partout et à tout moment.

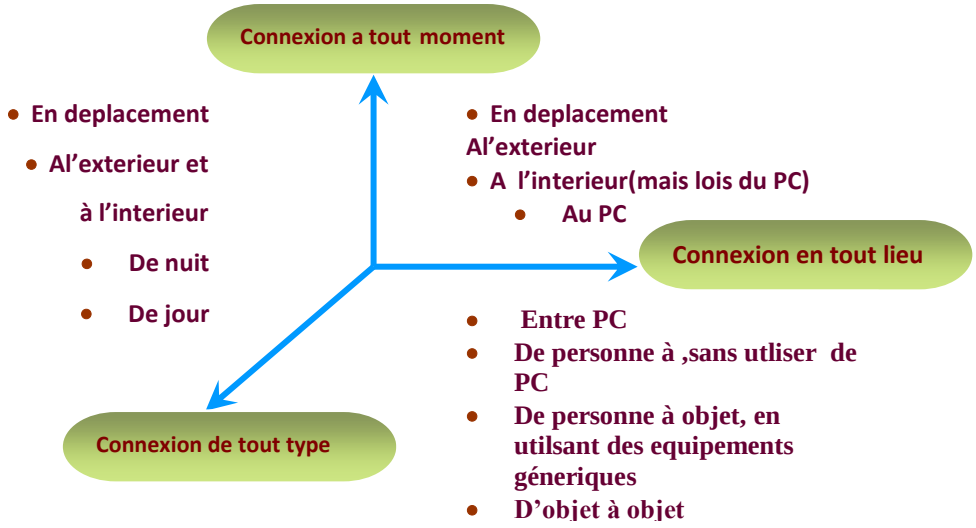

Fig.1.3 Connectivité des objets dans le Web ubiquitaire

Ainsi, par exemple, un réfrigérateur serait capable de proposer un menu à partir des ingrédients qu'il contient, d'avertir le propriétaire des dates de péremption des aliments et de commander directement ceux en rupture de stock… Autre scénario : un environnement ubiquitaire domestique relierait les contrôles domotiques grâce à des capteurs biométriques individuels intégrés dans les vêtements, modulant l'éclairage et la température de l'habitat, sans interruption et de manière imperceptible. Plus que le web, c'est l'infrastructure d'Internet qui est concernée par ces innovations, ce qu'on définit parfois sous le terme d'Internet 3.0

L'informatique ubiquitaire réunit trois conditions : l'intelligence des objets par la maîtrise de la sémantique, l'omniprésence des infrastructures réseaux (réseaux sans fil, capteurs biométriques…) et les comportements de dissémination de la vie privée sur les réseaux sociaux. On peut trouver l'informatique ubiquitaire dans de nombreux domaines tels que l'informatique distribuée, mobile, les réseaux de capteurs, l'interaction homme-machine, l'intelligence artificielle…

1.4.1.2 L'environnement ubiquitaire

Grâce aux progrès réalisés ces dernières années dans la miniaturisation de composants électroniques et à l'émergence des technologies réseaux sans fil, un nombre croissant d'objets de notre quotidien intègre des dispositifs électroniques grâce auxquels ils deviennent communicants [17]. Ces objets communicants peuvent être des dispositifs informatiques, des périphériques, ou des équipements de domaines aussi variés que l'électronique grand public, l'électroménager, les télécommunications, la domotique ou l'automobile. Ces objets sont, selon le cas, soit mobiles ou immobiles. Ils sont mobiles lorsqu'ils sont par exemple portés (téléphones, PDAs, ordinateurs portables, vêtements, etc.) ou conduits (véhicules) par leur(s) utilisateur(s). Ils sont en revanche immobiles et invisibles lorsqu'ils sont intégrés ou enfouis dans des objets fixes de l'environnement de l'utilisateur (télévisions, lampes, machines à laver, réfrigérateurs, etc.). Tout objet physique de l'environnement de l'utilisateur peut être potentiellement communicant et être ainsi capable d'interagir avec l'utilisateur et/ou, de façon autonome, avec d'autres objets.

Cette coopération entre objets est dynamique et spontanée dans le sens où : elle n'est pas prévue à l'avance, et elle doit s'adapter aux changements qui peuvent survenir dans l'environnement de l'utilisateur par l'apparition ou la disparition d'objets [18].

L'environnement numérique de l'utilisateur se définit comme un environnement ubiquitaire formé d'une fédération spontanée et dynamique d'objets communicants. Cette fédération est possible si l'on est capable de faire abstraction de l'hétérogénéité physique et matérielle des objets numériques pour ne considérer que les fonctionnalités qu'ils sont susceptibles de fournir à leur environnement. Pour faire abstraction de leur hétérogénéité, les objets communicants peuvent être perçus comme des entités abstraites qui requièrent et/ou fournissent un ou plusieurs services [18].

1.4.1.3 Architecture de services ubiquitaires

L'architecture de services permet de standardiser l'accès aux ressources et/ou aux fonctionnalités des objets communicants représentées sous formes de services [19]. Un service est défini par un contrat (appelé aussi interface) qui est une spécification abstraite de ses fonctionnalités. Ce contrat décrit :

- Ce que le service fournit,
- Comment y accéder, et
- Éventuellement, quelles sont ses propriétés non fonctionnelles.

L'architecture se compose de consommateurs de services (ou clients) qui interagissent avec des fournisseurs de services, mettant à disposition un ou plusieurs services. Un client et un service communiquent via des interactions synchrones ou asynchrones suivant le contrat du service, exprimé dans un langage déclaratif indépendant de tout langage de programmation, ce qui permet de s'abstraire de l'implémentation du service. Les consommateurs et les fournisseurs de services ne connaissent rien de leur implémentation respective. Les clients ne référencent pas directement des instances particulières de services qui leur sont nécessaires mais leur font indirectement référence par l'intermédiaire de la description de leurs caractéristiques. A l'aide de cette dernière, les clients sont en mesure de localiser/découvrir dynamiquement les instances de services disponibles dans leur environnement en interrogeant un service de découverte via un protocole de découverte de services (*SDP pour Service Discovery Protocol*) [20]. La découverte de service est soit centralisée ou distribuée [21]. Dans le premier cas, la découverte se fait à l'aide d'un annuaire qui centralise les descriptions des services, tandis que dans le second cas, les fournisseurs de services participent à la découverte de services suivant un modèle de découverte de service passif ou actif. La découverte est passive lorsque ce sont les fournisseurs de services qui envoient périodiquement des annonces dans l'environnement des services qu'ils mettent à disposition, tandis qu'elle est active lorsque ce sont les clients qui diffusent des requêtes décrivant les caractéristiques des services dont ils ont besoin. Ainsi, les interactions entre clients et services se font en plusieurs étapes [18] (Figure 1.4):

Le fournisseur de services publie la description de ses services auprès du service de découverte (voir Figure 1.4, étape 1).

Le consommateur de services interroge le service de découverte en lui soumettant la description (partielle) du ou des services requis (voir Figure 1.4, étape 2).
Le service de découverte renvoie le contrat du service et la référence d'une ou plusieurs instances de services correspondant (voir Figure 1.4, étape 3).

Le consommateur de services initie les interactions avec le fournisseur de service suivant les termes du contrat du service (voir Figure 1.4, étape 3).

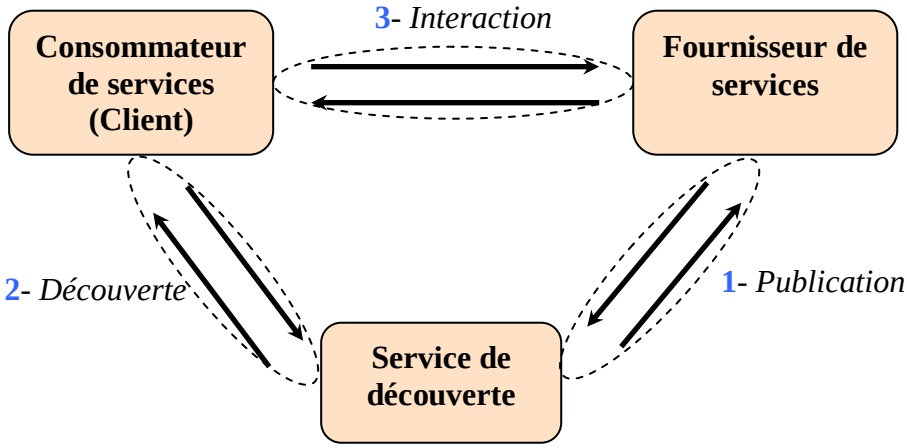

Fig.1.4 Architecture de services ubiquitaires

Une architecture de services ubiquitaires représente un environnement ubiquitaire où les objets communicants se comportent aussi bien comme un consommateur et/ou un fournisseur de service. Au niveau du réseau, les seules parties visibles de leur implémentation sont leurs protocoles de communication respectifs. L'interaction est possible si le comportement du service est connu, c'est-à-dire si son contrat/interface est standardisée et si le consommateur et le fournisseur de services utilisent un même protocole de découverte de services et d'interaction [18].

1.4.2 Les technologies du Web ubiquitaire

Ce qui rend le Web si utile pour les développeurs d'applications est la facilité avec laquelle les applications sont créées en utilisant des combinaisons de balises, de graphiques, de feuilles de style et de scripts.

Le Web ubiquitaire doit faciliter le développement d'applications distribuées en présentant des abstractions claires aux développeurs Web, leur permettant ainsi d'accéder aux capacités des terminaux et aux services de communication. La découverte et la description de ressources seront essentielles à la création d'applications du Web ubiquitaire.

L'utilisation des URI (adresses Web) pour nommer les terminaux, les services et les sessions permettra l'emploi de riches méta-données (le Web sémantique) pour la découverte de ressources, intervenant à travers les différents réseaux et exploitant la nature distribuée du World Wide Web.

TCP/IP	Le passage à IPv6 devrait résoudre pas mal de problèmes techniques et permettre à une foultitude d'appareils d'avoir une adresse IP
JavaScript	Passage à ECMAScript 4 : optimisations des performances
CSS	émergence de CSS 3
HTML	HTML 5 devrait apporter le stockage local, du contenu audio et vidéo natif, la 3D avec OpenGL
RIA	en attendant HTML5, les clients riches Internet ont de beaux jours devant eux : Adobe AIR, Microsoft Silverlight 3, Google GWT...

RDA	Java Web Start, Eclipse RCP, NetBeans RCP, AIR, Silverlight 3
APML	Attention Profiling Markup Language : format pour partager nos intérêts personnels à travers les réseaux sociaux
OPML	format d'échange de flux RSS, APML
XBRL	pour les documents de reporting financier
hListing	pour le commerce en ligne
RDF	Resource Description Framework : cadre défini pour encadrer les méta-données du web sémantique
OWL	Ontology Web Language : décrit les ontologies (liaisons logiques entre les métadonnées)
GRDDL	Gleaning Resources Description from Dialects of Languages : extrait les micro-formats XML d'un document vers le RDF

Le réseau ubiquitaire doit offrir des possibilités d'utilisation élargie pour des usages en situation de mobilité de handicap ou de domotique. Cette valeur ajoutée passe par l'utilisation de plusieurs modes de communication venant compléter ou enrichir les interfaces existantes.

De la multimodalité à l'interaction homme machine en passant par les IHM, la compréhension des enjeux du domaine passe par son étude préalable. La définition d'un ensemble de termes, de mécanismes et de concepts, l'étude des protocoles, langages et composantes logicielles sont les pré-requis nécessaires à une description plus fine des modèles sous tendant la conception d'une plate-forme de services ubiquitaires.

1.4.2.1 Interaction, interfaces et dialogue homme machine

La miniaturisation des terminaux mobiles a provoqué à la fois une modification des interfaces et l'introduction de nouveaux media, utilisant des modalités jusqu'alors peu exploitées dans les interfaces traditionnelles (vocales, gestuelles, tactiles).

1.4.2.1.1 Interaction vocale

Les technologies vocales recouvrent plusieurs activités complémentaires : la synthèse vocale, consistant à produire un signal sonore ; la reconnaissance vocale, consistant à faire interpréter par une machine une commande vocale produite par le locuteur afin de réaliser une action précise ; l'enregistrement vocal, consistant à acquérir et stocker un signal afin de le stocker ou de le transmettre à une machine ; la vérification du locuteur, activité dans laquelle on cherche authentifier le locuteur par l'analyse de son empreinte vocale.

L'utilisation de la voix pour communiquer avec une machine, présente plusieurs avantages. Elle s'impose dans le cas o`u les autres modalités de communication de l'utilisateur sont occupés. Par exemple dans le *cockpit* d'un avion de chasse, un pilote pourra utiliser sa voix pour marquer une position [22]. Dans le cadre de l'accessibilité et de la prise en compte du handicap, le mode vocal permet de rendre accessible, aux personnes non-voyantes, les écrans d'ordinateur ou le contenu des bibliothèques [23]. Dans le cadre de la réalisation de taches sensibles, on peut utiliser le mode vocal pour confirmer une action ou pour authentifier l'utilisateur. Pour les utilisateurs non spécialistes d'un système, le mode vocal offre une assistance guidant l'utilisateur dans sa démarche. Enfin, pour l'utilisateur souhaitant accéder à distance à un système, le mode vocal permet d'interagir à distance de manière directe, dans une maison par exemple, ou de manière médiatisée, par un accès téléphonique.

Le langage VoiceXML [24] est un dialecte XML spécifiant un dialogue homme machine vocal. VoiceXML permet de décrire un scénario de services vocaux. De la même manière que la visualisation d'une page web est l'interprétation d'un script HTML par un navigateur web, un service vocal VoiceXML est l'interprétation d'un script VoiceXML par un navigateur VoiceXML

Exemple simple d'utilisation de VoiceXML. L'interprétation de ce script déclenche la lecture du fichier jingle.wav, ensuite le texte Hello World ! est synthétisé puis joué dans le combiné de l'utilisateur et enfin la session se termine par la déconnexion du service.

```
<vxml version="2.0">
   <form>
        <block>
           <audio src="jingle.wav"/>
           <prompt>Hello World !</prompt>
           <disconnect/>
        </block>
   </form>
</vxml>
```

1.4.2.1.2 Interaction multimodale

Une modalité est une forme particulière d'un mode de communication. Une photo, un graphique et une carte, ce sont trois modalités du mode graphique. Considérons un media comme un dispositif physique avec lequel interagit l'utilisateur. Un magnétophone est un média, au même titre qu'un téléphone, un microphone, ou une caméra.

L'interaction multimodale est caractérisée par l'utilisation de plusieurs modes, comme par exemple la voix, le geste ou le mode graphique, pour accéder à un service, utiliser un logiciel. Les modes de communication font référence aux sens et aux moyens d'expression humains : le mode graphique fait référence à la vue ; le mode tactile fait référence au toucher ; le mode sonore fait référence à l'oreille; les modes gustatifs et odorants font référence au goût et à l'odorat. À ces références, s'ajoute la sensibilité du corps, la prise en compte de la conscience de l'utilisateur permet de prototyper des prothèses manipulables par la pensée et l'analyse de l'activité électrique des terminaisons nerveuses [25].

Selon le W3C[4] l'interaction multimodale vise à étendre le Web, permettant aux utilisateurs de sélectionner de manière dynamique le mode le plus approprié d'interaction pour leurs besoins actuels, y compris les handicapés afin de permettre aux développeurs d'applications Web à offrir une interface utilisateur efficace pour n'importe quel mode de communication. Avec les applications Web ubiquitaire, les utilisateurs peuvent apporter leur contribution par le biais de la parole, l'écriture et les frappes de touche, avec une production présentée par affiche, discours pré-enregistré et synthétiques, audio, et des mécanismes tactiles tels que des Vibrateurs de téléphonie mobile et des bandes en braille.

1.4.2.1.3 Interaction verbale et dialogue homme machine

L'interaction verbale se situe dans le paradigme « *computer as tool* », les commandes verbales permettent la manipulation d'une interface. Le dialogue homme machine adresse dans le paradigme « *computer as a partner* » c'est-à-dire qu'il reconnaît à la machine des compétences langagières et qu'elle coopère à la tache. L'interaction verbale reste centrée sur l'interaction et a pour but la réalisation de la tache. Le dialogue homme machine est un champ d'exploration de l'intelligence artificielle. Il confère aux agents logiciels des intentions, des émotions [26]. Sa mise en œuvre repose à la fois sur une représentation formelle des connaissances et sur le raisonnement d'agents rationnels utilisant des modèles de planification pour résoudre le problème de la coordination d'actions. L'étude du dialogue homme machine

4- Disponible en anglais sur la page (http://www.w3.org/UbiWeb/).

nous révèle une adaptation forte du dialogue au métier et au but que le concepteur de service cherche à adresser. Par exemple, la modélisation d'un service d'information voyageur en dialogue naturel nécessite une représentation des concepts d'espace, de lieu, de chemin, d'une définition des buts du dialogue et la spécification d'un ensemble de méthodes permettant de l'atteindre [27].

1.4.2.2 Identification et gestion des ressources

Les applications du *Web ubiquitaire* permettront d'identifier des ressources et de gérer celles-ci dans le cadre de sessions temporaires ou permanentes.

Les ressources peuvent être distantes comme dans le cas d'une imprimante ou d'un projecteur réseau, ou bien alors locales, comme dans le cas de l'autonomie d'utilisation d'un terminal, l'intensité du signal réseau reçu et son niveau sonore, ou bien des ressources matériel comme la bande passante de la connexion réseau, la disponibilité de la mémoire et la charge du processeur...etc.). Les ressources ne se limitent pas aux matériels, mais peuvent être également des services, tels que la reconnaissance de la parole, la traduction des langages naturels et l'identification de la localisation géographique d'un terminal.

Pour cela, il faudra une infrastructure plus souple que les solutions de rechange actuelles reposant sur les URL et les informations des sessions et les cookies.

1.5 Applications Web ubiquitaire

Au cours de ces dernières années, de nombreux travaux se sont intéressés à la conception d'applications Web adaptatives en fonction du contexte d'interaction des utilisateurs. Les éléments de contexte peuvent inclure, par exemple, des informations relatives à l'utilisateur (buts, préférences, historique des navigations, etc.), l'environnement (lieu, luminosité, bruit, temps, etc.), la plate-forme d'interaction (téléphone portable, PDA, PC, etc.) ou encore toute information pertinente pouvant être utilisée pour caractériser les conditions d'interaction. L'acquisition, la modélisation et le traitement de ces contextes d'interaction jouent un rôle fondamental dans le développement des systèmes adaptatifs en général et des applications web personnalisées.

Les maîtres mots des applications dans le Web ubiquitaire sont [28] : *sensibilité au contexte*, *smartness*, *scalabilité*, *invisibilité et pro-action*.
La *sensibilité au contexte* concerne la perception de l'environnement pour interagir plus "naturellement" avec l'utilisateur. Cette perception passe par l'utilisation de capteurs de l'environnement physique, de matériels auto-descriptifs, de description des personnes, etc. Le contexte fournit un grand éventail d'informations qui permet au système ubiquitaire d'agir de façon adaptée.

La contrainte de *smartness* est difficile à traduire en un seul mot ; elle signifie à la fois l'intelligence et la réactivité, mais aussi les bonnes manières ou le fair-play. L'objectif ici est de bien utiliser les informations de contexte pour décider de l'action la plus adéquate à la situation ; cela nécessite de définir des mécanismes d'adaptation.

La *scalabilité*, ou passage à l'échelle, est nécessaire dans les systèmes ubiquitaire, car on peut rencontrer des cas où le nombre de composants entrant en jeu est très important. On pense notamment aux réseaux de capteurs.

L'*invisibilité* est une contrainte forte; il ne faut pas encombrer l'utilisateur avec des considérations qui ne le concernent pas, il faut qu'il se concentre sur la tâche à réaliser. L'adaptation doit être transparente.

La *pro-action* a pour objectif de préparer le traitement d'une demande utilisateur avant même que cette demande ne soit explicite. Cela requiert une historisation des situations et actions, une analyse des situations et actions antérieures et leur rapprochement avec la session actuelle.

Le terme "*context-awareness*", souvent traduit par "sensibilité au contexte", inclut en fait souvent deux thématiques :

- *Sentir le contexte* : interagir avec des capteurs de contexte, interpréter les informations de bas niveau captées et ainsi construire une représentation de la situation dans laquelle on se trouve,
- *Réagir aux changements de contexte* : en fonction de la situation perçue, améliorer le fonctionnement du système logiciel pour qu'il puisse traiter ou utiliser les ressources et informations à disposition de façon optimale.

La conception des applications *Web ubiquitaire* doit concentrer sur l'adaptation logicielle et nos travaux se situent à ce niveau. Il s'agit de définir et mettre en œuvre des actions d'adaptation permettant au système logiciel ubiquitaire d'agir et interagir de la façon la plus optimale et transparente possible. Les changements de contexte sont les déclencheurs des actions d'adaptation. Le contexte est la principale source d'information pour l'adaptation de l'application.

La vision présentée ci-dessus a pour objet d'explorer les verrous scientifiques à long terme et donc de se positionner au premier plan des défis du domaine de *Web ubiquitaire*.

1.5.1 Caractéristiques de l'environnement du Web ubiquitaire

Le *Web ubiquitaire* se fonde en premier lieu sur une connaissance de la situation, appelée communément contexte. L'étude de l'adaptation au contexte nous amène à étudier les travaux de recherche qui se sont intéressés au contexte à fin d'en dégager les concepts essentiels.

Cet environnement se divise en trois catégories :

- l'environnement matériel qui représente les ressources du matériel comme la bande passante de la connexion réseau, la disponibilité de la mémoire et la charge du processeur...etc.) ;
- l'environnement de l'utilisateur qui représente la situation de l'utilisateur comme son emplacement, les personnes qui l'entourent, sa situation sociale ou son état émotionnel ;
- l'environnement physique ; telle que, la localisation géographique, la lumière ou le bruit.

Dey [29] a défini le contexte comme l'information pouvant être utilisée pour caractériser l'état de plusieurs entités (personne, endroit, objet) qui sont considérées comme pertinentes pour l'utilisateur, l'application et l'interaction entre l'utilisateur et l'application. Dey a étudié les définitions des autres auteurs, que nous venons de citer, et a fait une synthèse en

définissant le contexte comme étant tout élément pouvant influencer le comportement d'une application. Nous utilisons cette définition pour le reste du document.

1.5.2 Caractéristiques des informations de contexte

Les informations de contexte ont des caractéristiques variées. Elles proviennent de sources variées, ce qui amène à une large hétérogénéité en termes de modélisation, de traitement et de qualité. Elles peuvent être imparfaites, ambiguës et parfois même incohérentes ou incomplètes.

Ceci est dû au fait que ces informations sont généralement collectées à partir de capteurs physiques (ou sondes) qui peuvent être l'origine d'une haute dynamicité, des erreurs de capture et d'un bruitage des informations de contexte.

La nature dynamique de ces informations rend leur gestion difficile parce que la description de leur état devient rapidement obsolète. Cette dynamicité est d'autant plus difficile à gérer lorsque les sources de contexte sont distribuées et que les contextes nécessitent un traitement supplémentaire comme leur abstraction.

La complexité des informations de contexte nécessite des moyens de modélisation et d'interprétations spécifiques ainsi que des mécanismes d'adaptation qui permettent de prendre en compte leur nature. Nous définissons dans la section suivante les mécanismes utilisés pour collecter les informations de contexte, la modélisation de ces informations ainsi que les mécanismes utilisés pour adapter une application au contexte.

1.5.3 L'adaptation au contexte

Les termes informatiques "*sensible au contexte*", "*adaptable au contexte*" ou "*qui prend en compte le contexte* ", proviennent tous du terme anglais " *context-aware computing* " qui a été introduit la première fois par Schilit et al en 1994 [30], et un sujet commun entre les chercheurs dans le *Web ubiquitaire*, Schilit a défini ce terme comme étant un logiciel qui s'adapte en fonction de sa localisation d'utilisation, de l'ensemble des objets et des personnes qui l'entourent ainsi que de la variation de ces objets à travers le temps.

Une autre définition a été donnée par Dey [29] qui définit un système comme sensible au contexte s'il utilise le contexte pour offrir des informations ou des services pertinents pour l'utilisateur.

Toute adaptation au contexte nécessite d'abord l'acquisition des informations de contexte qui seront traitées et analysées afin de réaliser l'adaptation. Nous présentons dans cette section les différents mécanismes utilisés pour les acquérir, les interpréter, les modéliser et adapter une application aux changements du contexte.

1.5.4 Acquisition des informations de contexte

Diverses méthodes sont utilisées pour collecter les informations de contexte. Mostéfaoui [31] distingue trois types de contextes du point de vue de leur acquisition :

- **1) le contexte capturé** : ce type de contexte est acquis par le moyen de capteurs matériels ou logiciels appelés aussi " sondes " telles que les capteurs de température, de pression, de niveau d'éclairage et de bruit. Les informations de contextes capturées

les plus fréquemment citées dans les travaux de recherche sont la localisation des utilisateurs ou des objets et les informations sur le réseau. La localisation à l'extérieur des bâtiments est généralement capturée à l'aide d'un système GPS alors que la capture de la bande passante du réseau nécessite généralement l'utilisation de systèmes spécifiques.

- **2) le contexte explicitement fourni** : par exemple, les préférences de l'utilisateur sont explicitement communiquées par l'utilisateur à l'application ;
- **3) Le contexte dérivé ou interprété** : contrairement au contexte capturé et au contexte explicitement fourni qui représentent des informations de contexte de bas niveau (qui proviennent directement de leur source), le contexte dérivé représente un contexte de haut niveau qui peut être calculé à la volée à partir d'autres informations de contexte. Par exemple, le pays où se trouve l'utilisateur est un contexte de haut niveau déduit à partir du contexte de bas niveau qui représente ses coordonnées collectées à partir d'un GPS.

1.5.5 Modélisation des informations de contexte

Les caractéristiques variées des informations de contexte comme leur hétérogénéité, leur dynamicité et leur imperfection nécessitent des modèles abstraits de description de contexte. Plusieurs travaux ont défini des modèles de contexte. Nous en citons quelques uns dans ce qui suit.

- **Paires de clé-valeur** : L'information de contexte est modélisée par une paire clé-valeur.
- **Schmidt et al. [32]** fournissent un modèle de traitement du contexte basé sur des couches dans lesquels la sortie des capteurs est transformée en un ou plusieurs signaux. Ces signaux subissent un traitement pour former une description abstraite du contexte comportant un ensemble de valeurs, chacune liée à une mesure de certitude qui estime sa précision.
- **Modélisation orientée objet** : basée sur l'approche orientée objet. Dans cette approche, les informations de contexte sont regroupées en un ensemble d'entités. Chaque entité représente un objet conceptuel ou physique tel qu'une personne, un dispositif ou un réseau. Les propriétés des entités telles que le nom d'une personne sont représentées par des attributs. Les entités sont liées à leurs attributs à travers des associations.
- **Modèle d'objets sensibles** : appelé " **Sentient Object** " [33], ce modèle a proposé une manière plus formelle pour la présentation du contexte. Le contexte est décrit en utilisant un langage basé sur le modèle entité-association et l'information de contexte est stockée au moment de l'exécution dans une base de données relationnelle.
- **Modélisation par une ontologie** : une ontologie est une spécification explicite d'une conceptualisation. Dans une ontologie, les connaissances d'un certain domaine sont représentées formellement comme un ensemble d'objets ayant des relations entre eux. La modélisation du contexte par une ontologie a un réel intérêt ; elle permet le partage des informations de contexte dans un système distribué et avec une sémantique bien définie elle permet l'utilisation d'agents intelligents pour faire un raisonnement sur le contexte. Une ontologie nécessite un langage de représentation basé sur un modèle sémantique. Dans les dernières années, plusieurs langages d'ontologies basés sur les technologies web ont été proposés.

La modélisation des informations de contexte est primordiale pour leur gestion à savoir leur transport, leur échange et leur stockage mais vu la diversité des informations de contexte, cette modélisation est étroitement liée au domaine de leur utilisation.

1.5.6 Requête basée sur la localisation

Dans cette section, nous soulignons l'importance pour le Web ubiquitaire des requêtes basées sur la localisation et des mécanismes utilisés afin de capturer la localisation.

Cao et al. [34] définissent les requêtes dépendantes de la localisation comme celles dont le résultat est lié à une localisation qui n'est pas fournie explicitement dans la requête. Afin de construire la réponse à la requête, il faut tout d'abord connaître la localisation de l'utilisateur expéditeur de la requête. En fonction de la localisation de l'utilisateur, les applications peuvent configurer leurs fonctionnalités (par exemple, pour une même activité d'un utilisateur, les applications peuvent exécuter des ensembles de fonctionnalités différents selon sa localisation) et les résultats des requêtes sensibles à cette localisation changent (par exemple, si l'utilisateur demande des restaurants situés dans la rue dans laquelle il se trouve, les résultats peuvent changer selon sa localisation) [7].

Le travail de Thilliez et al. [35] concerne également les « requêtes basées sur la localisation ». Ces auteurs distinguent les « requêtes sensibles à la localisation » (« *location-aware queries* ») et les « requêtes dépendantes de la localisation » (« *location dependent queries* »). Les premières sont exécutées indépendamment de la localisation de l'utilisateur (par exemple, quels sont les sites touristiques les plus visités à Paris ?). Les secondes sont évaluées en tenant compte de la localisation physique courante de l'utilisateur (par exemple, quels sont les restaurants situés dans la rue où se trouve l'utilisateur ?). Dans ce cas, les réponses visent à aider l'utilisateur à trouver facilement et précisément l'information localisée dont il a besoin [7].

Différents mécanismes pour obtenir la localisation de l'utilisateur. Nous considérons que cet aspect est essentiel pour l'adaptation de l'information, surtout si les requêtes de l'application sont dépendantes de la localisation et/ou si l'utilisateur considère explicitement la localisation comme un critère à prendre en compte pour adapter ou chercher l'information. Il est important de noter que les changements de localisation de l'utilisateur peuvent entraîner la modification de ses activités dans le système.

1.5.6.1 Mécanismes de capture de la localisation

Les applications s'exécutant sur des DM peuvent obtenir la localisation de l'utilisateur ou de son dispositif d'accès de plusieurs manières : à travers une interface de saisie proposée à l'utilisateur ou à travers des mécanismes ou méthodes tels que ceux proposés par Nieto-Carvajal et al. [8] comme le « Signal Strength », le « SNMP (Simple Network Management Protocol) » .

Indulska et al. [36] proposent l'utilisation de capteurs comme mécanismes d'obtention de la localisation de l'utilisateur ou de son dispositif d'accès. Ils distinguent les capteurs physiques des capteurs virtuels.

Les capteurs physiques fournissent l'information sur la position d'un dispositif physique.

Les récepteurs GPS, des cameras et des lecteurs de code barres sont des exemples de ce type de capteurs. Ces capteurs fournissent de l'information sur la position ou la proximité du dispositif par rapport au système avec différents degrés de précision (c'est-à-dire, avec une précision de quelques centimètres, mètres ou centaines de mètres) [7].

1.5.7 Applications sensibles au contexte

Le premier système sensible au contexte issu des travaux de recherche dans ce domaine est *Active Badge* qui a été conçu et développé entre 1989 et 1992. Depuis, plusieurs travaux de recherche ont été réalisés donnant lieu à des applications sensibles à des contextes variés dans plusieurs domaines [37]. Nous en citons quelques exemples dans ce qui suit.

- Le ***Shopping Assistant guide*** utilise la localisation de l'utilisateur pour le guider dans un magasin. Il lui offre des détails sur les articles du rayon où il se trouve et lui indique où se trouvent les produits soldés.

- ***CyberGuide*** offre des informations pour les touristes sur une carte interactive et leur propose des endroits à visiter selon leur emplacement courant et l'historique des endroits déjà visités.

- ***Conference Assistant*** utilise une variété de contextes pour guider les participants à une conférence. L'assistant examine le programme de la conférence, les sujets des présentations, la localisation de l'utilisateur et l'intérêt de l'utilisateur et lui suggère une présentation à laquelle assister. Lorsque l'utilisateur entre dans une salle de conférences, l'assistant lui affiche automatiquement tous les détails liés à la présentation comme son titre et le nom de celui qui la présente.

- ***Adaptive GSM phone and PDA*** a utilisé une plus grande variété de contextes comme l'activité de l'utilisateur, le niveau de luminosité, la pression atmosphérique et la proximité des personnes. Il permet à un utilisateur de PDA de changer la taille d'écriture en fonction de son activité (par exemple une grande taille lorsque l'utilisateur est en train de marcher et une petite lorsque l'utilisateur est en position stable). Il permet aussi de sélectionner des profils de téléphones portables en fonction du contexte. Le téléphone choisit automatiquement lors d'un appel de sonner, vibrer ou de rester silencieux selon son emplacement, c'est à dire dans un sac, sur une table, dans la main ou à l'extérieur.

- ***Office Assistant*** est un agent qui interagit avec les visiteurs d'un bureau dès leur entrée par la porte pour gérer l'agenda du possesseur du bureau. L'entrée des visiteurs est automatiquement détectée et suivie par une demande d'identité. Selon l'agenda et la disponibilité du possesseur du bureau un rendez-vous est automatiquement fixé.

- ***Call Forwarding*** est basé sur le système de localisation d'*Active Badge*, le réceptionniste utilise la localisation de l'utilisateur pour faire suivre les appels téléphoniques au téléphone le plus proche de l'utilisateur.

Toutes ces applications sont construites selon l'approche conduite par les capteurs, tous les mécanismes d'acquisition et d'adaptation au contexte sont inclus dans le code de l'application, ce qui ne permet pas la réutilisation de ces mécanismes par une autre application. Par ailleurs, les mécanismes utilisés par ces applications pour interagir avec le

contexte sont spécifiques à une technologie d'acquisition donnée et ne supportent pas des technologies hétérogènes. Plusieurs versions de méthodes d'interaction avec les capteurs doivent donc être développées pour interagir avec chaque type de technologie d'acquisition de contexte. Par exemple, il existe plusieurs technologies pour acquérir la localisation de l'utilisateur : le GPS à l'extérieur, les fréquences radio et l'infrarouge à l'intérieur. Il est important que l'application puisse utiliser facilement l'un ou l'autre des capteurs. L'approche conduite par les capteurs ne le permet pas. L'utilisation d'un intergiciel sensible au contexte permet de développer de manière plus souple les applications sensibles au contexte [37].

1.5.8 Vers la standardisation : *middlewares*

Les développeurs de systèmes sensibles au contexte ont besoin de suivre des procédures uniformes et standard pour construire des applications adaptables au contexte. Ces procédures peuvent être fournies des Framework. Un Framework est un ensemble de classes abstraites collaborant entre elles pour faciliter la création de tout ou d'une partie d'un système logiciel. Un Framework fournit un guide architectural en partitionnant le domaine visé en classes abstraites et en définissant les responsabilités de chacune ainsi que les collaborations entre classes [37].

Selon Xu *et al.* [38] un « *middleware* » est une collection de services informatiques distribués qui permettent de faire communiquer des clients et des serveurs, s'exécutant sur des plates-formes hétérogènes, d'une manière flexible et correcte. Le rôle d'un « *middleware* » est de faciliter les tâches de conception, de construction et de gestion d'applications distribuées en fournissant des environnements de construction des applications réparties qui soient simples, cohérents et intégrés. L'élément clé d'un *middleware* est l'interopérabilité.

Plusieurs architectures et modèles d'applications sensibles au contexte ont été proposés et développés pour offrir des cadres de développement standards et proposer des interfaces uniformes et réutilisables pour interagir avec le contexte. Nous allons présenter certains de ces travaux.

CARISMA

CARISMA (*Context-aware Reflective middleware System for Mobile Applications*) [39] est un intergiciel personnalisable en fonction des besoins d'une application. Cette personnalisation est réalisée par l'intermédiaire de profils d'applications. Un profil d'application représente une liste d'associations entre un service que l'intergiciel propose, une politique d'utilisation et une configuration de contexte dans laquelle la politique peut être utilisée. Ce profil est ensuite passé à l'intergiciel pour qu'il puisse déterminer le comportement à adopter lors de l'exécution d'un service pour des conditions particulières du contexte. Dans le cadre de la réflexivité, les profils d'application définissent des méta-données qui déterminent le comportement du système réflexif. Ces méta-données sont accessibles via une API réflexive qui permet l'introspection et la modification des associations (service, politique, contexte) prédéfinies.

K-Components

K-Components [40] est un cadre qui utilise la réflexivité pour construire des logiciels adaptables. La logique de l'adaptation dans K-components est basée sur la spécification de contrats qui décrivent le comportement adaptatif du logiciel à l'aide du langage ACDL (Adaptation Contracts Declarative Language). L'adaptation se produit en réponse à des

événements adaptatifs envoyés par les composants de l'application. Si une adaptation s'avère nécessaire, un composant peut être retiré du graphe qui représente la configuration du système et remplacé par un autre composant ayant la même interface.

ReMMoC

ReMMoc (Reflective Middleware for Mobile Computing) [41] est un intergiciel configurable et reconfiguration qui a été conçu pour offrir une solution aux problèmes de l'hétérogénéité du réseau. Il est capable de communiquer avec tout service présent dans le réseau, quelle que soit l'infrastructure qui implémente ce service. La conception de ReMMoc se base sur un modèle réflexif de composants appelé OpenCOM qui est construit au-dessus d'un sous-ensemble de fonctions COM.

ReMMoC est défini par un cadre de composants qui contient à son tour trois autres cadres de composants.

- Le cadre *liaison* assure l'interopérabilité entre les divers services disponibles sur le réseau qui sont implémentés sur d'autres types d'intergiciels. Ce cadre est basé sur un mécanisme de greffe de différents types d'implémentations de liaisons ;
- le cadre *service de recherche* qui permet la découverte de services qui sont habituellement découverts par une diversité de protocoles de recherche de services. Ce cadre est également basé sur un mécanisme de greffe de différents protocoles de découverte de services ;
- le cadre *exécution* est responsable de l'exécution d'une application. Il est capable de créer des composants, de les supprimer ou bien de les assembler.

Grâce aux cadres de liaison, de recherche et d'exécution, un utilisateur mobile qui utilise un service sur un réseau et qui entre dans une zone couverte par un autre type de réseau offrant le même service peut utiliser ce service même si les protocoles diffèrent.

OpenORB

OpenORB [42] est un intergiciel réflexif qui a été conçu à l'université de Lancaster. Au moment du chargement, des composants appropriés sont sélectionnés pour former une instance de l'intergiciel. En utilisant la réflexion, des composants peuvent être modifiés ou chargés durant l'exécution. Chaque objet dans OpenORB est associé à un méta-espace qui peut être accédé à travers une des interfaces du méta-modèle.

Chisel

Chisel [43] est un cadre d'adaptation dynamique sensible au contexte réflectif basé sur la description de politiques d'adaptation. L'approche de Chisel consiste à décomposer les différents aspects d'un service en plusieurs comportements possibles. Chaque comportement sera utilisé dans un contexte donné. Le changement de comportement du service est conduit par un script basé sur des politiques d'adaptation accessibles à l'utilisateur. L'avantage de Chisel est qu'il permet de rajouter de nouveaux comportements non anticipés au moment de l'exécution du service. Cela est réalisé en définissant les différents comportements comme méta-types.

CORTEX

CORTEX (CO-operating Real-time senTient objects) a des capacités de configuration et de reconfiguration afin de répondre aux exigences imposées par des environnements ad hoc. Cette nouvelle architecture utilise, comme ReMMoc, les techniques de la réflexivité et des cadres de composants. L'implémentation de l'intergiciel est basée sur OpenCOM. CORTEX contient quatre cadres de composants :

- le cadre de composants *contexte* : il permet de collecter et gérer les informations de contexte. Il est basé sur le modèle des "objets sensibles" [44]. Les objets sensibles sont définis comme des entités intelligentes capables de consommer des événements depuis des sondes (capteurs) et de produire des événements réfléchis vers des déclencheurs (actuators). L'intelligence du cadre de composants *contexte* est due à sa capacité de transformation des événements en informations de contexte, à sa capacité de stockage de ces informations dans une mini-base de données et à son moteur d'inférence. ;
- le cadre de composants *Publish/Subscribe* : il est basé sur un modèle d'événements, appelé STEAM et permet la dissémination des événements dans un réseau ad hoc ;
- le cadre de composants *gestionnaire de ressources* : permet de répartir les tâches par rapport aux ressources disponibles ;
- le cadre de composants *service de recherche* : ce cadre de composants est le même que celui présenté dans ReMMoC, il donne une solution aux problèmes posés par l'hétérogénéité du réseau, puisqu'il permet de découvrir des services annoncés par différents protocoles de découverte de services.

CORTEX a les mêmes objectifs que CARISMA puisqu'il met la réflexivité au service d'applications orientées vers la connaissance de contexte. Néanmoins CORTEX est utilisé avec des applications à petite échelle.

Plusieurs intergiciels ont également été réalisés pour offrir des solutions aux problèmes de l'hétérogénéité des technologies d'interaction avec le contexte. Parmi eux nous pouvons citer ceux qui supportent des technologies de localisation différentes comme Oracle iASWE [45]. D'autres intergiciels ont été réalisés pour supporter les opérations de déconnexions et de partage des données pour les utilisateurs mobiles comme Xmiddle [46]. Le but de ces intergiciels est de maximiser la disponibilité des données.

1.6 Conclusion

Nous avons présenté dans ce chapitre les définitions et les visions de l'informatique ubiquitaire et en particulier, les spécificités du *Web ubiquitaire*, les challenges qui peut apporter en termes de technologies. Également, le *changement de services* qui concerne la manière dont l'infrastructure disponible est utilisée, afin de fournir de nouveaux services à l'utilisateur. À propos des dispositifs d'accès propres au Web ubiquitaire (c'est-à-dire, des *Dispositifs Mobiles, DM*), Le *Web ubiquitaire* cherche donc à élargir les capacités des navigateurs Web afin de permettre de nouveaux types d'applications Web, notamment celles nécessitant une coordination avec d'autres terminaux et une adaptation dynamique de l'utilisateur, des capacités des *dispositifs mobiles* et du contexte.

Nous avons analysé les caractéristiques des informations de contexte et les mécanismes utilisés dans les différentes phases d'adaptation au contexte qui sont l'acquisition des informations de contexte, leur analyse et l'adaptation au contexte. L'analyse des informations de contexte nécessite leur modélisation d'une façon abstraite. Nous avons ensuite étudié différents systèmes sensibles au contexte existants qui utilisent ces mécanismes.

Les informations de contexte sont généralement collectées à partir de capteurs différents. Ces informations se caractérisent par leur hétérogénéité et leur imperfection. Ces informations peuvent nécessiter plusieurs traitements tels que leur analyse, leur interprétation et leur présentation abstraite avant d'être utilisées dans le processus d'adaptation. Il est préférable de séparer ces traitements du système à adapter afin de dissimuler la complexité de ces traitements et permettre leur réutilisation par d'autres systèmes : c'est le rôle des intergiciels sensibles au contexte.

Les intergiciels sont de différents types. Ils peuvent ne tenir que le rôle de collecteur d'informations de contexte comme ils peuvent intégrer plusieurs fonctionnalités de traitements de contexte voir même des mécanismes d'adaptation. Le placement d'applications au dessus de tels intergiciels raccourcit et facilite les différentes phases du cycle de développement de des systèmes de recherche d'information dans le Web ubiquitaire, puisque le système va directement réutiliser les mécanismes offerts par l'intergiciel sans avoir à les redévelopper.

Les protocoles et les principes d'architectures pour de tels systèmes de RI communicants sensibles au contexte restent à définir. La disponibilité et la sécurité de ces systèmes de devront être intégrée dans l'architecture. Enfin, des mécanismes et des techniques distribuées spécifiques au traitement de très grandes dimensions de systèmes doivent être abordés, tant pour les problèmes de distribution que d'accès au contenu à grande volume d'informations.

Chapitre 2

Recherche d'information dans web

2.1 Introduction

Depuis son apparition, le Web a connu un grand effort de recherche et de développement au niveau mondial, grâce à cet effort, le Web est devenu la source d'information privilégiée à quiconque recherche des informations en relation avec ses besoins. En effet, le service de l'Internet met à la disposition de tout Internaute, tout type d'information organisée sous la forme de pages contenant des documents et permettant le passage d'une page vers une autre, et d'un site à un autre. La recherche d'une information particulière, parmi cet ensemble, est compliquée par la distribution de ces données sur un grand nombre de sites et de serveurs d'information. Cependant, cette distribution de l'information pose le problème de leur localisation, pour leur exploitation par l'utilisateur, chaque document étant noyé dans un énorme fond documentaire -appelé également corpus- en constante évolution. En effet, il n'est donc facile de retrouver, de manière pertinente, l'information recherchée, à moins de l'analyser.

Il n'est donc plus possible de se passer de l'utilisation d'outils automatiques de recherche d'information.

Les systèmes de recherche d'information SRI, sont conçus à l'origine, pour répondre à ce besoin. Ces systèmes de recherche utilisent certaines méthodes plus anciennes dites classique ayant trait à l'analyse de textes. La plupart de différents modèles de SRI réalisés demandent à l'utilisateur d'exprimer son besoin en utilisant le langage de requête du modèle. À partir de la requête, la fonction de correspondance du modèle extrait de la source d'informations, les informations qui sont susceptibles de répondre au besoin. Habituellement, cette fonction n'utilise pas les informations de la source, mais les indexations, qui sont des représentations des informations, dont le but est d'améliorer les performances de la fonction de correspondance (temps et qualité des résultats).

Les systèmes de recherche d'information distribuée (SRID) ont le même but qu'un SRI. A savoir, satisfaire le besoin d'information de l'utilisateur. Et principalement, couvrir au maximum la masse d'information existante sur Web est un premier pas vers ce but. En effet, les SRIDs ont été conçu pour assurer une grande couverture pour pallier les limites rencontrées par les SRI dites centralisés. Les SRIDs possèdent l'avantage d'éviter à l'utilisateur de consulter indépendamment chaque source d'information ou serveur Web. Un SRID est constitué d'un courtier qui communique avec un ensemble de serveurs. Chaque serveur correspond à un SRI. Chaque serveur contient donc, un index et un moteur de recherche qui exploite cet index. En clair, un SRID est un ensemble de serveurs qui coopèrent à fin d'atteindre un but commun, celui de répondre au mieux aux besoins d'information des utilisateurs. Le fonctionnement interne de cet ensemble de serveur reste transparent à l'utilisateur qui garde l'impression de travailler avec un seul serveur c'est-à-dire le courtier.

Les moteurs de recherche implémentant un SRID permettent à l'utilisateur de formuler une requête de recherche par l'intermédiaire d'un sac de mots-clés. La recherche consiste à sélectionner dans un corpus - l'index - l'ensemble des documents dans lesquels figurent ces mots et à les trier suivant une mesure définissant leur pertinence à la requête.

Les méta-moteurs de recherche offrent à l'utilisateur une interface d'interrogation beaucoup plus élaborée que les moteurs de recherche classiques. La recherche consiste à interroger plusieurs sources de documents - issues d'autres moteurs de recherche - en adaptant la requête pour chacun d'eux. Les résultats obtenus sont fusionnés et analysés de manière

beaucoup plus précise en fonction de la requête initiale afin de les trier de façon plus pertinente pour l'utilisateur.

Pour chacun de ces types d'outils de recherche, plusieurs composantes peuvent être définies. Chacune d'elles représente un domaine de recherche indépendant des autres. Ce chapitre aborde également les aspects d'un moteur de recherche.

Plusieurs familles d'algorithmes multi-agents ont vu le jour. La plus grande source d'inspiration reste cependant l'observation de la nature. Chacune de ces méthodes essaye ainsi de tirer parti d'un comportement observé dans les groupes sociaux. Ce chapitre aborde plusieurs méta-heuristiques trouvant principalement leurs sources d'inspiration dans le courant des algorithmes évolutionnaires, ainsi que leur adaptation à des problèmes de recherche d'information sur Web. Ces méta-heuristiques vont nous servir de base à l'élaboration des rôles de recherche (inspirés du comportement des fourmis et les systèmes multi-agents) permettant d'enrichir notre base de rôle de recherche par une variété de techniques de RI.

Les besoins en information sont ainsi énormes, et plus précisément, les informations pertinentes à l'utilisateur en fonction de son contexte de la recherche, Dans ce cadre, la RI contextuelle émerge comme un domaine à part entière : sans remettre en cause ses origines, elle pose des problématiques nouvelles allant de la modélisation du contexte jusqu'à la modélisation de la pertinence cognitive en passant par la modélisation de l'interaction entre un utilisateur et un système de recherche d'information (SRI) [47].

Ce chapitre est organisé en deux parties de la façon suivante :
La *Première Partie* est consacrée à l'étude des techniques de recherche d'information usuelles, pour cela, nous présentons dans la section 1, les techniques de RI classiques, nous détaillons les principes de base de l'architecture des systèmes de recherche d'information SRI en général, avec une présentation des différentes modèles de SRI, puis nous examinons en particulier les systèmes de recherche d'information distribuée SRID. En suite, Nous détaillerons deux algorithmes de recherche d'information sur Internet, il s'agit, de l'algorithme HITS et l'algorithme PageRank utilisés à grande échelle dans les outils de recherche. La suite aborde l'aspect matériel en analysant les différentes architectures des moteurs de recherche.

Dans la section 2, présente le domaine de la recherche d'information dans le Web à base des Systèmes Multi-Agents, avec quelque systèmes de RI dans le Web à base des SMA, présentons aussi le projet Marvin et le projet AGATHE.
La section 3 est consacrée à l'étude de la RI à base de colonie de fourmis, ses caractéristiques ainsi que l'apport des méta-heuristiques inspirées du comportement des fourmis à la recherche locale, nous détaillons par la suite l'algorithme fondateur : ACO, et l'algorithme API.

La section 4 présente les algorithmes génétiques, ses caractéristiques ainsi que l'apport à la RI dans Web, en suite, une étude particulière du phénomène de croisement et les techniques de parallélisation des algorithmes génétiques à gros grains et à grains fins.

La section 5 est consacrée à l'étude de l'approche génétique et à base d'agents pour la RI dans le Web, en présentant Webnaut et InfoSpiders.

Deuxième partie Est consacrée à la présentation de la RI contextuelle, L'objectif de cette partie est d'apporter un éclairage sur les problématiques posées par la RI contextuelle, sur

leurs origines, les diverses motivations qui ont conduit à l'émergence de la RI contextuelle puis nous présentons, les définitions des principaux concepts qui en traduisent les dimensions fondamentales, sur les verrous posés par la RI classique dans un tel cadre, ainsi que sur les solutions apportées dans la communauté.
La section 1 est consacrée à la présentation des principaux verrous technologique portant sur les approches, modèles et techniques de RI contextuelle. La section 2 présente les notions et concepts de base de la RI contextuelle, en suite, dans la section 3 on détaille les différents modèles de la RI contextuelle. Dans la section 4, on cite quelques systèmes opérationnels notables nés de ces travaux. Enfin la section 5 conclut le chapitre.

2.2 Techniques de recherche d'information usuelles

Devant la révolution du WEB, plusieurs travaux ont été menés afin de faciliter l'accès aux informations disponibles dans ce gigantesque espace d'information. Les moteurs de recherche représentent une aide inestimable pour la recherche et l'accès aux documents du WEB. Nous pouvons noter que les moteurs de recherche actuels ne tiennent pas compte des spécificités du WEB et sont basés sur des modèles de RI qui ont été développés pour des documents textuels classiques depuis les années 70 [48]. Ils fonctionnent actuellement sur le principe de page HTML. Le document est l'atome d'information que l'internaute recherche. L'indexation se réalise au niveau de granularité de la page HTML et le réseau de liens entre les pages est peu exploité.

2.2.1 La RI classique

De manière générale, la recherche dans un SRI consiste à comparer la représentation interne de la requête aux représentations internes des documents. La requête est formulée, par l'utilisateur, dans un langage de requêtes qui peut être le langage naturel, un langage à base de mots clés ou le langage booléen. Elle sera transformée en une représentation interne équivalente, lors d'un processus d'interprétation. Un processus similaire, dit indexation, permet de construire la représentation interne des documents de la base documentaire [49].

2.2.1.1 Système de recherche d'information SRI

La Recherche d'Information (RI) concerne les mécanismes qui facilitent l'accès à une base d'informations. C'est une démarche faite par un utilisateur pour obtenir, à l'aide du système de recherche d'informations (SRI), les informations (ou les références vers les informations) qui peuvent répondre à son besoin.

- *Un Système de Recherche d'Information (SRI)* : est un système informatique qui facilite l'accès à un ensemble de *documents* (corpus), pour retrouver ceux dont le contenu *correspond* le mieux à un *besoin* d'information d'un utilisateur.

Les SRI et les modèles sous-jacents se basent donc sur trois notions clés : le document, le besoin et la correspondance.

- *Document:* Un document peut être un texte, une page WEB, une image, une bande vidéo, etc. Dans notre contexte, nous appelons document toute unité qui peut constituer une réponse à une requête d'utilisateur.
- *Requête*: Une requête exprime le besoin d'information d'un utilisateur.
- *Correspondance*: Le but de la RI est de retrouver seulement les documents pertinents (qui correspondent le mieux à la requête).

Nous distinguons donc les deux tâches principales d'un SRI :

- *L'indexation automatique*, c'est-à-dire l'extraction et le stockage du contenu sémantique des documents du corpus. Cette phase nécessite un modèle de représentation de ce contenu sémantique, appelé modèle de documents.
- *L'interrogation*, c'est-à-dire l'expression du besoin d'information de l'utilisateur sous la forme d'une requête, la recherche dans le corpus, et la présentation des résultats. Cette phase nécessite un modèle de représentation du besoin de l'utilisateur, appelé modèle de requête, ainsi qu'une fonction de correspondance qui doit évaluer la pertinence des documents par rapport à la requête.

À l'origine de l'utilisation d'un système de recherche d'information (SRI) se base sur la fonction de correspondance qui n'utilise pas les informations de la base de documents, mais les indexations, qui sont des représentations des informations, dont le but est d'améliorer les performances de la fonction de correspondance (temps et qualité des résultats). Nous distinguons donc quatre composants principaux (cf. Figure 2.1)

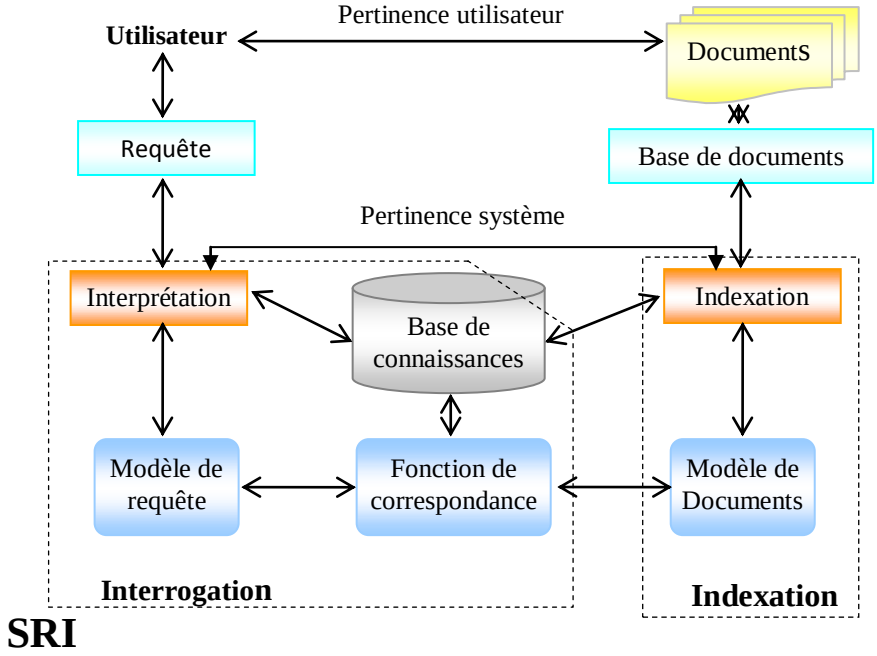

Fig. 2.1 : Schéma général d'un modèle de système de recherche

- *Modèle de documents* : correspond à la modélisation du contenu sémantique des documents, dans le formalisme de représentation de connaissances. Le choix du formalisme utilisé est crucial, mais il est toujours difficile, sinon impossible, d'obtenir une modélisation exprimant parfaitement l'idée initiale de l'auteur.
- *Modèle de requête* : correspond à la modélisation du besoin d'information de l'utilisateur, dans le formalisme de représentation de connaissances. Ce formalisme limite souvent la précision de définition du besoin. De plus, la "qualité" de la requête exprimée par l'utilisateur varie considérablement avec sa connaissance du domaine et avec son aptitude à définir son besoin. Il y a donc souvent une importante perte d'information entre le besoin et son expression.
- *Fonction de correspondance* : le système évalue la pertinence (la valeur de correspondance) des documents par rapport à la requête. La fonction de correspondance est un élément clé d'un SRI, car la qualité des résultats dépend de l'aptitude du système à calculer une pertinence des documents la plus proche possible du jugement de l'utilisateur.
- *Base de connaissances* : un thésaurus, composé de concepts apparaissant dans le corpus, reliés entre eux par diverses relations (Spécificité/Généricité, Synonymie, voir aussi, etc.).

2.2.1.1.1 Modèles de SRI

Dans un système de recherche d'information (SRI) la formulation d'une requête impose une représentation du document et un modèle d'extraction d'informations. Dès 1958, Lunh,

un des pionniers de la recherche en SRI, a établi dans [50] les bases de l'hypothèse fondamentale des travaux sur l'extraction et la sélection d'informations :
«*Le contenu textuel d'un document discrimine le type et la valeur des informations qu'il véhicule*».

Le quasi totalité des SRI actuels se base sur ce principe. L'analyse de la présence de mots dans un corpus de texte permet de déterminer les documents susceptibles de répondre aux souhaits d'un utilisateur du SRI.
Nous allons ainsi examiner dans cette section les différents systèmes permettant de spécifier la présence, l'absence ou encore la proximité de termes dans un document.
Nous étudierons dans un premier temps le système booléen et le système booléen étendu à l'origine des premiers moteurs de recherche. Nous détaillerons ensuite d'autres méthodes autorisant la spécification des requêtes sous une forme différente comme le modèle vectoriel, le modèle probabiliste qui permet de mesurer le degré de similarité entre deux documents. Et en fin le modèle logique. Nous nous intéresserons également dans la suite de cette section aux Algorithmes de base de RI dans le web qui s'appuient sur ces modèles.

2.2.1.1.1.1 Le modèle booléen

Dès l'apparition des premiers SRI, le modèle booléen [51] s'est imposé grâce à la simplicité et à la rapidité de sa mise en œuvre. L'interface d'interrogation de la plupart des moteurs de recherche est basée sur ce principe et n'est composée que d'une liste de mots-clés. Ces termes peuvent être combinés à des opérateurs logiques ($\wedge, \vee, \neg$) afin de déterminer au mieux le souhait d'un utilisateur.
Dans ce modèle, un document (d) est représenté par son ensemble de termes (ti), et une requête (q) comme une expression logique de termes. Un document ne correspondra à une requête que si l'implication $d \Rightarrow q$ est valide. Cette correspondance $C(d, q)$ est déterminée comme suit :

$$\begin{aligned} &C(d, ti) = 1 \text{ si } ti \in d \text{ ; 0 sinon} \\ &C(d, q_1 \wedge q_2) = 1 \text{ si } C(d, q_1) = 1 \text{ et } C(d, q_2) = 1 \text{ ; 0 sinon} \\ &C(d, q_1 \vee q_2) = 1 \text{ si } C(d, q_1) = 1 \text{ ou } C(d, q_2) = 1 \text{ ; 0 sinon} \\ &C(d, \neg q) = 1 \text{ si } C(d, q) = 0 \text{ ; 0 sinon} \end{aligned} \qquad (1.1)$$

En pratique, les termes de la totalité des documents sont identifiés et stockés en conservant les liaisons d'appartenance à chaque texte. On désigne cet ensemble sous le terme index inversé. La recherche des documents dans lesquels figure un terme est ainsi fortement accélérée.
Pour permettre à l'utilisateur de spécifier au mieux sa requête, certains moteurs de recherche ont ajouté d'autres opérateurs logiques au modèle initial adapté plus particulièrement au contexte et à la problématique textuelle. Ce sont les opérateurs de proximité - ou d'adjacence - et les opérateurs de troncature. Ils permettent de préciser la position de deux termes l'un par rapport à l'autre pour les premiers et la recherche de termes syntaxiquement proches pour les derniers. Ils sont généralement utilisés sous la forme de guillemets (") ou d'astérisques (*) dans les formulations de requêtes.

Une des principales faiblesses du modèle booléen est de distinguer uniquement l'adéquation d'un document à une requête et de ne pas établir de relation d'ordre de pertinence entre les différents éléments du corpus. C'est pour cette raison que ce modèle est

généralement utilisé dans la première étape d'un SRI pour pré-sélectionner les documents résultant de la requête initiale afin d'y appliquer un algorithme de tri par pertinence.

2.2.1.1.1.2 Le modèle booléen étendu

Le modèle booléen étendu a été introduit par Salton [51]. C'est une extension du modèle précédent qui vise à tenir compte d'une pondération des termes dans le corpus. Cela permet de combler le manque du modèle standard en ordonnant les documents retrouvés par le SRI.

La requête reste inchangée et est composée d'une expression booléenne classique. Par contre, la représentation d'un document se voit augmentée par l'ajout de pondération pour chaque terme (*ti*,*wi*). En général ce poids est principalement basé sur le nombre d'occurrences d'un terme dans le document mais dans certains systèmes la typographie - taille du texte, forme de la police de caractère - est également prise en compte.

La détermination de la correspondance d'un document à une requête *C(d, q)* peut prendre plusieurs formes. Suivant le cadre classique des ensembles flous proposé par Zadeh [52], on obtient les relations suivantes :

$$\begin{aligned} &C(d, ti) = ai \\ &C(d, q_1 \wedge q_2) = \min(C(d, q_1), C(d, q_2)) \\ &C(d, q_1 \vee q_2) = \max(C(d, q_1), C(d, q_2)) \\ &C(d, \neg q) = 1 - C(d, q) \end{aligned} \qquad (1.2)$$

Les opérateurs logiques $\wedge$ et $\vee$ sont évalués respectivement par min et max. Cependant, cette évaluation n'est pas parfaite. On n'obtient pas les relations :
$C(d, q \wedge \neg q) \equiv 0$ *et* $C(d, q \vee \neg q) \equiv 1$.
Ce qui signifie que lorsqu'on évalue une requête sous forme de conjonction, on ne s'intéresse qu'à la partie la plus difficile et lorsqu'on évalue une requête sous forme de disjonction, c'est la partie la plus facile qui domine. C'est pour cette raison que d'autres formes de correspondances ont été proposées. Les relations les plus utilisées ont été introduites par Lukasiewicz [**7**] sous la forme suivante :

$$\begin{aligned} &C(d, ti) = a_i \\ &C(d, q_1 \wedge q_2) = C(d, q_1) * C(d, q_2) \\ &C(d, q_1 \vee q_2) = C(d, q_1) + C(d, q_2) - C(d, q_1) * C(d, q_2) \\ &C(d, \neg q) = 1 - C(d, q) \end{aligned} \qquad (1.3)$$

Les deux parties d'une conjonction ou d'une disjonction contribuent en même temps à l'évaluation de la correspondance du document à la requête. Cependant, ce modèle n'est pas parfait (on n'obtient toujours pas C(d, q $\wedge$ $\neg$q) $\equiv$ 0 et C(d, q $\vee$ $\neg$q) $\equiv$ 1) mais il reste convenable.

Le modèle p-norme introduit par Salton mesure les correspondances de la conjonction et de la disjonction [51]. L'idée de base réside dans l'observation de la table de vérité (voir figure 2.2). Dans la colonne A $\wedge$ B la meilleure correspondance est atteinte dans le cas de la dernière ligne. Dans la colonne A $\vee$ B la pire correspondance correspond à la première ligne.

A	B	A ∧ B	A ∨ B
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Fig.2.2 : Table de vérité.

On peut ainsi considérer que l'évaluation d'une conjonction ou d'une disjonction consiste à calculer une sorte de distance entre le point à atteindre ou à éviter. Ce concept est illustré par la figure 2.3. Dans ces schémas, les axes des abscisses correspondent à l'évaluation du document A et les axes des ordonnées à l'évaluation du document B.

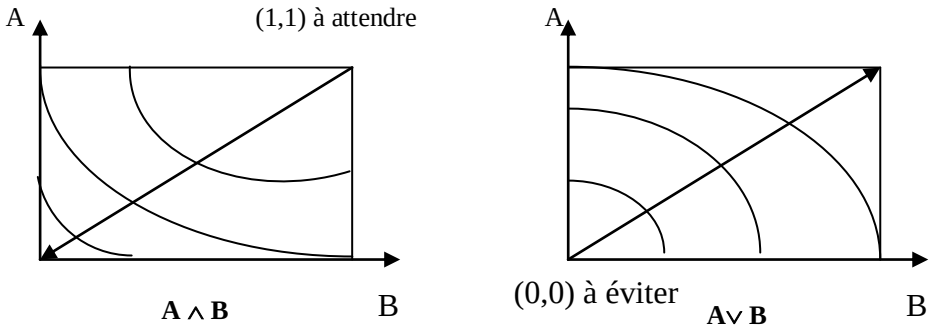

Fig.2.3 – Évaluation d'une conjonction ou d'une disjonction.

Pour la conjonction, on cherche à évaluer dans quelle mesure le point c défini par l'évaluation d'un document A et d'un document B est proche de (1,1), le point à atteindre. Ce rapprochement peut être mesuré par le complément de la distance entre le point c et le point (1,1). Plus cette distance est grande, moins A ∧ B est satisfaite. Les points qui se situent sur une même courbe, ont la même distance avec (1,1) et ainsi correspondent à la même évaluation. Dans le cas de A ∨ B, on recherche plutôt à éviter le point (0,0). Plus on est loin de (0,0), plus A ∧ B est satisfaite. Ainsi Salton propose les relations de correspondance normalisées suivantes :

$$C(d, t_i) = a_i$$

$$C(d, q1 \wedge q2) = 1 - \sqrt{\frac{(1 - C(d, q1))^2 + (1 - C(d, q2))^2}{2}} \qquad (1.4)$$

$$C(d, q1 \vee q2) = 1 - \sqrt{\frac{C(d, q1))^2 + C(d, q2))^2}{2}}$$

$$C(d, \neg q) = 1 - C(d, q)$$

Le principal avantage de ce modèle sur le modèle booléen classique réside dans la mesure du degré de correspondance entre un document et une requête dans [0, 1]. On peut ainsi ordonner les documents dans l'ordre décroissant de leur correspondance avec la requête.

Ce modèle de tri n'est pratiquement plus implémenté en tant que tel dans les moteurs de recherche actuels. L'ordre établi est maintenant remplacé par le tri de pertinence associé à la topologie des liens hypertextes dans un réseau. Nous examinerons cette technique dans la suite ci-dessus.

2.2.1.1.1.3 Le modèle vectoriel

L'interface d'interrogation d'un moteur de recherche par des listes de mots-clés montre ses limites dès lors qu'un utilisateur n'assimile pas le système booléen et n'a pas une idée précise des mots les plus pertinents caractérisant son souhait. Certains méta-moteurs de recherche ont commencé à explorer une autre voie en proposant une nouvelle interface d'interrogation permettant à l'utilisateur de spécifier des pages ressemblant aux résultats qu'il attend. Par exemple, le système Webnaut [53] scrute les pages visitées par un utilisateur et les utilise pour guider ses recherches. Cette démarche à l'avantage de cerner d'une manière beaucoup plus précise les souhaits d'un utilisateur. Cependant, elle nécessite l'établissement de méthodes permettant de déterminer la similarité entre deux documents.

La notion de similarité entre documents est fortement liée au choix de la méthode de représentation des textes. Dans le cas de collections de textes de grande taille, la représentation admettant actuellement les meilleurs résultats et la plus utilisée est la représentation vectorielle. Elle a été mise en œuvre dès 1971 par Salton dans le système de recherche documentaire SMART [54]. Dans cette méthode, un document est représenté par un vecteur dans un espace vectoriel dont les dimensions sont associées à des unités linguistiques spécifiques (mot, lemmes, etc.) que l'on désigne dans la littérature par terme d'indexation. Ce choix de terme d'indexation n'est pas aisé. Il représente un facteur prépondérant dans la détermination de l'efficacité de la mesure de similarité entre documents. Il repose principalement sur des systèmes d'analyse de texte et de détermination du sens lexical des documents.

Les premiers travaux conséquents dans ce domaine ont été proposés par Luhn. Dans un de ses premiers papiers [50], il a énoncé que la fréquence d'apparition d'un mot dans un document établit une mesure efficace de la signification des documents. Il a également proposé que la position relative de mots apportant le plus d'information dans une phrase fournisse une bonne mesure pour déterminer la signification des phrases. La détermination du facteur de signification d'un document est ainsi basée sur ces deux mesures.

Dans le domaine de l'analyse automatique de textes, Luhn a donc contribué à établir que la fréquence des termes d'un document peut être utilisée pour extraire les mots ou phrases qui représentent ce document.

L'analyse statistique de nombreux textes a montré que la fréquence d'apparition d'un mot dans un corpus en fonction de son rang dans le document - fonction décroissante du nombre d'occurrences du mot - forme une courbe similaire à une hyperbole. La courbe de fréquence de la figure 1.4 illustre cette situation. Cette courbe suit en réalité la loi de Zipf [55] qui établit que le produit de la fréquence d'un terme par son rang est approximativement constant. Cette loi a été vérifiée par Zipf sur la base de journaux américains en anglais mais s'applique sur tout corpus linguistique. Or dans un texte, la valeur informative d'un mot peut s'exprimer sous la forme d'une gaussienne en fonction du rang des termes d'un document comme montré dans le figure 2.4 [56]. En effet, les mots les plus fréquents d'un texte représentent généralement des mots de liaison ou d'usage courant et ne véhiculent pas ou peu d'information sur la signification d'un document, et les mots les moins fréquents n'apportent que rarement un sens primordial à la compréhension d'un texte.

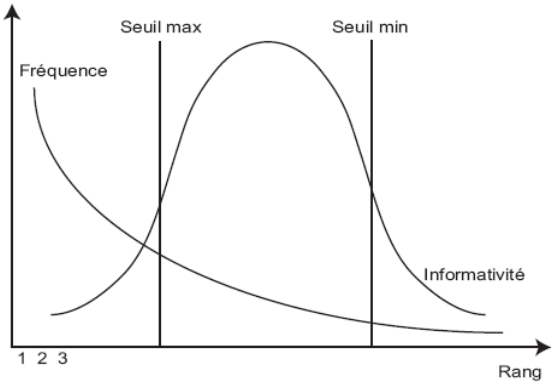

Fig. 2.4 – Relation entre le rang et la fréquence d'apparition d'un terme dans un document.

Luhn a utilisé cette constatation comme hypothèse de base pour spécifier deux seuils de coupure (seuil max et seuil min sur la figure) déterminant les mots apportant le plus d'information pour caractériser le sens d'un document et éliminant les mots sans signification. Les mots au-delà du seuil maximum sont considérés comme trop communs et ceux en deçà du seuil minimum comme trop rares. La probabilité d'obtenir un mot discriminant le contenu d'un document est maximale entre les deux seuils et tend à diminuer dans toutes les directions atteignant des valeurs proches de zéro aux points de coupure. Il n'y a pas de solution idéale pour établir ces seuils et la meilleure méthode consiste à procéder par essais successifs par rapport à un corpus donné. Cependant, dans le cas général, l'intervalle $\left[\frac{|C|}{100}, \frac{|C|}{10}\right]$, où |C| est le nombre de mots dans le corpus, est considéré comme adéquat pour fournir des termes avec un bon pouvoir discriminant.

Ces idées simples sont à la base de la plus grande partie des travaux réalisés dans le domaine de la recherche d'information. Luhn les a lui-même utilisées pour construire une méthode de génération automatique de résumés [50]. L'algorithme consiste à déterminer la proportion de mots significatifs ou non dans chaque portion de texte puis à les trier en fonction de leur score. Seules les meilleures phrases sont alors insérées dans le résumé.

Dans le cadre du modèle vectoriel, chaque document d est ainsi représenté au moyen d'un vecteur $\langle d \rangle = (wt_1, ..., wt_{t|T|})$ dénommé dans la littérature *profil lexical*, où *wti* représente le poids, dans le document d, du terme d'indexation *ti* associé à la $i^{ème}$ dimension de l'espace vectoriel.

Le poids d'un profil lexical *ti* peut être représenté par plusieurs mesures. La plus connue et utilisée à ce jour est certainement la mesure *tf*idf*. Elle regroupe un ensemble de schémas de pondération et de sélection de termes.
tf signifie 'term frequency' et *idf* 'inverted document frequency'.
On retrouve dans la littérature plusieurs formules *tf* et *idf*. Certaines introduisent une normalisation, mais celle-ci n'est pas nécessaire dans le cas de l'utilisation d'une mesure de similarité indépendante de la norme. Généralement on utilise les formules suivantes pour calculer les éléments *tf*idf* d'un document :

$$tf = log\ (f(ti, d) + 1) \quad (1.5)$$

$$idf = log(1/n) \quad (1.6)$$

Avec *f(ti, d)* correspondant à la fréquence d'occurrence du terme *ti* dans le document d et n représentant le nombre de documents du corpus dans lequel figure le terme *ti*.

Une formule *tf*idf* combine les deux critères précédents. La composante *tf* indique l'importance du terme pour un document tandis que *idf* indique le pouvoir de discrimination de ce terme. Ainsi, un mot ayant une valeur de *tf*idf* élevée doit être à la fois important dans ce document et apparaître peu dans les autres documents.

$$wti = tf * idf = log\,(f(ti, d) + 1) - log(1/n) \quad (1.7)$$

Cette représentation des documents sous forme de vecteurs de termes d'indexation nous permet de déterminer la similarité entre les différents éléments de notre corpus. La mesure cosine [**48**] est utilisée dans la plupart des travaux recensés dans ce domaine. Cette méthode consiste à calculer le cosinus de l'angle entre le vecteur représentant la requête de l'utilisateur et chaque document du corpus. Elle se caractérise par la formule suivante :

$$\cos(d_i, d_j) \frac{\sum_n \mathrm{d_i, nd_{j,n}}}{\sqrt{\sum_n d_{i,n}^2 \sum_n d_{j,n}^2}} \quad (1.8)$$

D'autres mesures de similarité peuvent être utilisées, citons par exemple les méthodes basées sur le nombre de concepts communs entre la requête et le document, l'utilisation de la somme des produits *tf*idf* ou encore la mesure Okapi [57].Mais c'est la mesure cosine qui apporte dans la plus grande majorité des cas les meilleurs résultats [58].
Nous verrons par la suite des exemples de moteurs de recherche utilisant ces mesures de similarités afin de proposer à l'utilisateur de formuler sa requête à l'aide de documents ressemblant à ceux qu'il désire retrouver.

2.2.1.1.1.4 Le modèle probabiliste

Basé sur la théorie des probabilités, considère la RI comme un espace d'évènement possible, un évènement peut être le jugement de pertinence porté par l'utilisateur sur un document par rapport à une requête ou l'association d'un descripteur à un document. La représentation des documents est généralement un index pondéré. Les poids des descripteurs correspondent a la probabilité que le descripteur soit pertinent pour le document, aussi appelée degré de croyance. Dans le cas du modèle binaire indépendant, seule la représentation de la requête est un index pondéré .les documents y sont représentés par un index à plat [59].

La fonction de comparaison calcule la probabilité qu'un document D reprendre à un besoin d'information d'un utilisateur, formulée par une requête Q qui peut être interprétée par la probabilité que l'évènement pertinent R arrive sachant D et Q : $P(R/D,Q)$ .les documents résultats sont ordonnés. Pour calculer la probabilité de l'évènement $(R/D,Q)$), il doit être décomposé en évènements plus simples par la formule de bayes, en tenant compte des dépendances entre les différents évènements. Plusieurs paramètres sont pris en compte : les documents, les requêtes, les représentations des documents, les termes d'indexation, etc. Afin d'obtenir une formule de comparaison calculable, les modèles probabilistes font des hypothèses restrictives, comme l'indépendance entre les mots. En revanche, le système INQUERY [60] tient compte des dépendances entre certains évènements pour calculer $P(R/D,Q)$ en parcourant un réseau d'inférence bayesienne.

L'avantage des modèles probabilistes est de tenir compte des imprécisions, en particulier entre le document et sa représentation. Pour évaluer les différentes probabilités du système il est nécessaire de disposer d'un jeu de données initiales. Ces systèmes fonctionnent en deux étapes. Une première étape d'apprentissage calcule les probabilités des évènements à partir

d'un jeu de données et une seconde étape de test répond à une nouvelle requête. Les données nécessaires au calcul des probabilités pouvant être :

- La fréquence du mot dans le document.
- Un ensemble de jugement de pertinence de documents par rapport à des requêtes, généralement obtenues par retour de pertinence (*relevance feedback*) permettent de faire évoluer le système au cours de son utilisation.
- Un corpus de documents préalablement indexés manuellement, un jeu de test contenant des requêtes et leurs documents résultats, etc.

Ces systèmes sont utilisables autant pour l'indexation automatique que pour l'indexation manuelle. En indexation automatique; la probabilité qu'un descripteur soit représentatif du document est évaluée à partir d'un jeu de données. En indexation manuelle, l'évènement d'attribution d'un descripteur à un document est connu, donc, sa probabilité d'apparition n'a pas besoin d'être évoluée. Dans ce cas, la représentation des documents est une indexation à plat.

Les systèmes de type probabiliste peuvent autant utiliser une indexation en langage contrôlé qu'en langage libre, tout dépend du jeu de données utilisé au départ pour évaluer les probabilités.

2.2.1.1.1.5 Le modèle logique

Van rijsbergen [**61**] modélise la pertinence d'un document répondant à une requête par une implication logique. Soit $x(d)$ l'information contenue dans le document d et $x(d)$ le besoin d'information formulée par la requête q, tous deux, sont des formules logiques. Ce genre de système cherche à évaluer l'ajout minimal d'information nécessaire pour obtenir l'implication $x(d) \rightarrow x(q)$, permettant de classer les documents résultats. Cette approche améliore l'utilisation des connaissances dans le SRI, car, les éléments d'information (et non plus les termes) sont les descripteurs du document. Le problème majeur est d'extraire les éléments d'information automatiquement. La proposition a été appliquée à plusieurs théories logiques pour déterminer $x(d) \rightarrow x(q)$ [62]

Une des théories, souvent utilisée est la théorie des situations. Les modèles logiques développés à partir de la théorie des situations considérants qu'un document est identifié à une situation et que les éléments d'information sont des types. Un type possède la valeur *vrai* dans une certaine situation, et fausse dans une autre situation. Ces valeurs sont déterminées dans la phase d'indexation. Des contraintes sont définies entre ces types provenant par exemple de relations sémantiques trouvées dans un thésaurus. Ces contraintes définissent la nature du flot d'information existant entre deux situations. La formule de comparaison évalue l'incertitude du flot d'information circulant entre la situation du document et celle de la requête [63].

Les modèles logiques développés actuellement ont permis de mieux comprendre fondamentalement la RI en donnant un cadre théorique pour la comparaison entre les modèles existants. En revanche, l'implémentation de ces modèles semble difficile du fait de leur complexité. Les systèmes développent à partir de ce modèle n'ont pas donné de résultats très satisfaisants comparés aux autres systèmes [64].

2.2.1.2 Systèmes de recherche d'information distribuée (SRID)

Définissons d'abord les termes qui vont nous servir à présenter le domaine de la RID.

2.2.1.2.1 Terminologie

1. Une **source d'informations** est un ensemble de documents à disposition par un particulier ou une organisation désirant les publier.
2. Une **collection** est un ensemble de documents appartenant à une ou plusieurs sources d'information.
3. Une **liste de résultats** est une liste triée de références vers des documents qui sont jugés pertinents par le système à une requête donnée. Ces listes peuvent avoir des syntaxes très différentes et peuvent contenir des informations variées telles :
    - les degrés de pertinence des documents ; le degré de pertinence peut être valeur numérique ou une valeur symbolique telles que des graphiques ou des étoiles etc.
    - les passages qui contiennent le plus de termes communs avec la requête ;
    - les résumés des documents ;
    - etc.
4. **Un serveur** est une machine qui offre à un utilisateur le service de recherche de l'information dans une collection qu'il héberge. Un serveur contient un SRI local qui effectue cette tache. L'utilisateur peut interroger directement le serveur ou depuis une autre machine en passant par un réseau.
5. Un **courtier** est un module qui joue le rôle d'intermédiaire entre l'utilisateur et les différentes sources d'information. Le courtier agit comme un pseudo-moteur de recherche qui reçoit en entrée une requête et fournit en sortie une liste de résultats. Le fonctionnement du courtier sera présenté en détails dans la section suivante.
6. Une **représentation de serveur** est un ensemble d'informations détenues par le courtier et décrivant certaines caractéristiques du serveur comme par exemple les termes apparaissant dans la collection du serveur, leur fréquences d'apparition, ainsi que des informations utiles à son interrogation (sa localisation, son coût, ses droit d'accès, etc.).
7. **L'utilité d'un serveur** à une requête se traduit, par sa capacité à contenir et retourner des documents pertinents à cette requête.
8. La **fréquence document** d'un terme t_j est le nombre de documents, dans une collection, contenant t_j.
9. La **fréquence collection** d'un terme t_j est le nombre de collections, dans un SRID, contenant t_j.

2.2.1.2.2 Principes de fonctionnement d'un SRID

Un SRID simple est compose de plusieurs serveurs et un courtier (figure 2.5).
Le courtier est le coeur d'un SRID. Le courtier contient cinq composants logiciels : un modèle de gestion, un modèle frontal (une interface utilisateur), un modèle de sélection de serveurs, un modèle de communication et un modèle de fusion de résultats.

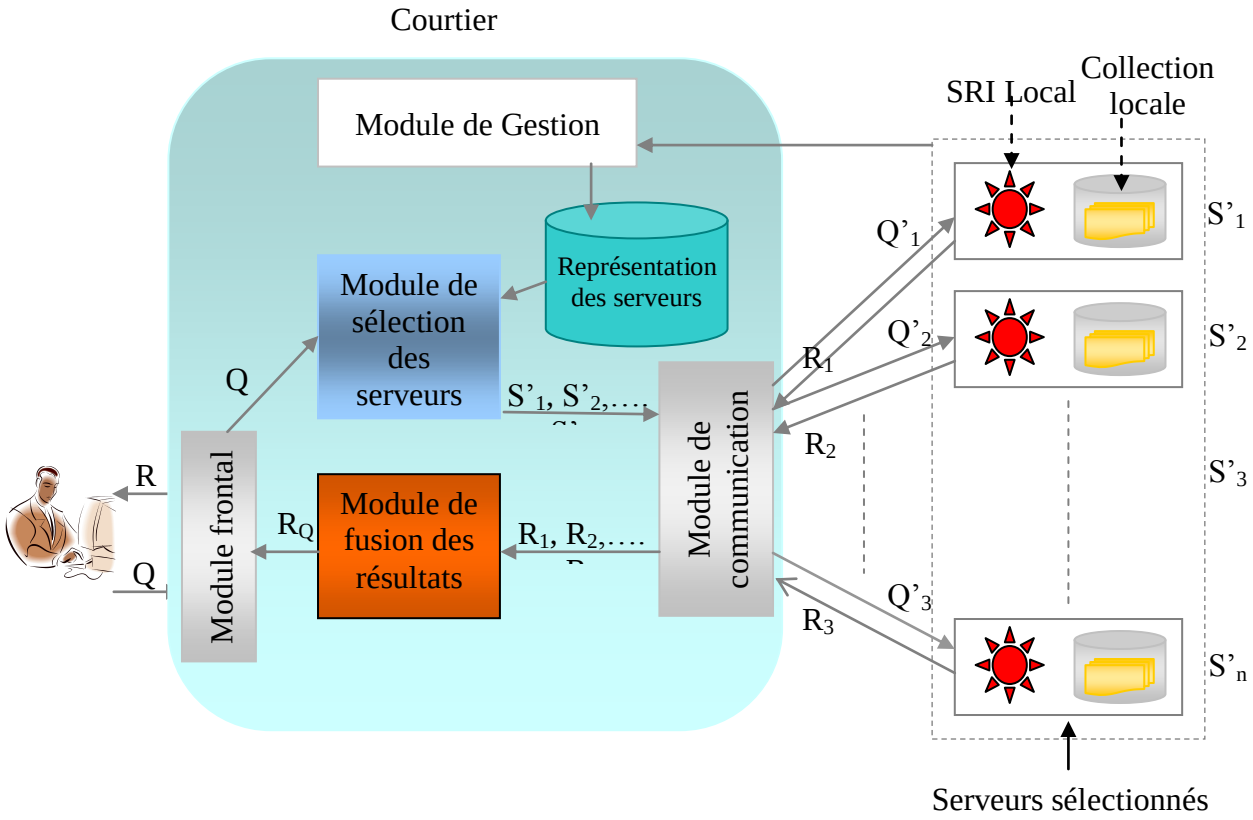

Fig.2.5 Les modules fonctionnels d'un SRID

D'une manière globale, le module de gestion construit les représentants de serveur du système dont il connaît les spécificités. L'utilisateur soumet sa requête par le biais du module frontal (interface utilisateur), le module de sélection de serveurs choisit les serveurs qui sont susceptibles de répondre à cette requête par des documents pertinents, le module de communication adapte alors la requête au langage d'interrogation de chaque serveur sélectionné, leur achemine la requête et réceptionne leur réponse. Enfin, le module de fusion des résultats groupe les réponses reçues par le module de communication et rend via l'interface utilisateur une liste unique de résultats à l'utilisateur. Nous détaillons dans ce qui suit le fonctionnement de chaque module. Pour cela nous adoptons quelques notations, soient :

- Q une requête soumise par un utilisateur ;
- S l'ensemble des n serveurs connus par le système ;
- Rep l'ensemble des représentants des serveurs connus par le système ;
- S'_Q l'ensemble $\{S'_1, S'_2,\ldots,S'_{xQ}\}$, le sous ensemble de S des serveurs sélectionnés pour Q et xQ son cardinal ;
- Q'_i est la requête traduite dans le langage d'interrogation du serveur S'_i ;
- R_i la réponse de S'_i, R_i est la liste de couples < D, info> provenant de S'_i, où D est une référence vers un document de la collection de S'_j jugé pertinent par S'_i pour Q et info des informations relatives à ce document ;

2.2.1.2.2.1 Le module de gestion

Ce module s'occupe essentiellement de construire Rep (voir la figure 2.5) l'ensemble des représentants des serveurs et le maintenir à jour. L'automatisation de cette tache est très

importante vu la rapidité avec laquelle les documents paraissent, sont modifies et disparaissent. En outre, les approches qui se basent sur une construction manuelle [65] ou une construction des représentants par apprentissage [66] son difficiles et coûteuses à appliquer dans un environnement en continuel changement.

2.2.1.2.2.2 Le module frontal

Ce module se charge d'établir la communication entre l'utilisateur et le système. Ses fonctionnalités varient d'un système à un autre, il peut offrir une aide à l'utilisateur pour formuler son besoin d'informations, pour choisir les sources d'informations à rechercher, pour définir le nombre de documents qu'il voudrait retrouver, pour affiner sa requête de manière manuelle ou automatique.

2.2.1.2.2.3 Le module de sélection de serveurs

A partir de Rep, le module de sélection détermine S'_Q (voir la figure 2.5), l'ensemble des serveurs qui sont les plus susceptibles de documents pertinents à la requête. Certes la sélection manuelle est possible. Dans un tel cas, les utilisateurs débutants ont tendance à sélectionner tous les serveurs.
Certains courtiers ne procèdent pas à une sélection et interrogent tous les serveurs disponibles. Cette approche est la plus répondue dans les méta-moteurs de recherche d'information sur Internet.
La plupart des méthodes de sélection existantes procèdent au classement des serveurs selon leur utilité à la requête qui est calculée par un score. Le score d'un serveur peut être par exemple le nombre présumé de documents pertinents que contient sa collection. Après le classement, la sélection proprement dite, peut s'effectuer selon plusieurs scénarios :

- *La sélection fixe* : consiste à sélectionner un prédéfini *n_serv* de serveur ;
- *La sélection par seuil* : consiste à sélectionner tous le serveur dont le score dépasse un certain seuil *seuil* ;
- *La sélection économique* : consiste à choisir les collections qui satisfont certaines métrique de coût de façon à ce que l'on obtienne le maximum de documents pertinents au moindre coût. Fuhr [67] propose des métriques basées par exemple sur le coût de traitement d'une requête par une collection et le temps de réponse de cette collection.

Concernant la sélection fixe et la sélection par seuil, la définition de *n_serv* et de seuil soulève des difficultés. Car, si l'une de ces valeurs est trop petite des serveurs contenant des documents pertinents peuvent être écartés, ce qui favorise l'augmentation du silence dans la réponse ; et si elle est trop grande, des serveurs ne contenant aucun document pertinent sont aussi interrogés, augmentant ainsi le bruit dans la réponse.
La sélection a plusieurs buts qui peuvent être regroupés en deux catégories. Premièrement, le but d'*efficience*, en d'autre terme, réduire le coût de la recherche en diminuant le nombre de serveur à interroger. Il est important aussi de réduire la quantité d'information nécessaire au processus de sélection, pour que le système soit extensible. Deuxièmement, le but de performance en interrogeant le plus petit nombre de serveurs, sans détériorer la performance du système voire en augmentant sa performance si l'on écarte uniquement les serveurs ne possédant aucun document pertinent.

Un autre type de sélection existe, dans lequel il faut faire le choix des serveurs dupliqués. Dans ce cas la sélection va se faire de plusieurs manières :

- D'une façon aléatoire [68] ;
- Selon la topologie du réseau, en choisissant le serveur le plus proche [69]
- En choisissant le serveur le moins encombre dynamiquement [70] ;
- En se basant sur des critères concernant les serveurs, notamment leur temps de réponse, la vitesse de transfert avec ces serveurs, etc. [71].

2.2.1.2.2.4 Le module de communication

Une fois l'ensemble S'_Q de serveur à interroger établi, et avant de procéder à l'émission de la requête vers chaque serveur S'_i, le module de communication, en se basant sur des informations sur le langage d'interrogation de serveur à contacter, convertit la requête dan un format compréhensible par chaque serveur. Pour résoudre le problème de traduction de requêtes, des protocoles de standardisation ont été suggérés comme, par exemple,

Z39.50 [72], *Common Command language* (CCL) [73] et *ISO 8777*[74].

La deuxième fonction du module de communication consiste à extraire, à partir des R_i, les informations nécessaires à la fusion de ces réponses.
Sur le Web pour le moins, la syntaxe du langage d'interrogation ou la présentation des résultats varient assez fréquemment, nécessitant une mise à jour du module de communication.

2.2.1.2.2.5 Le module de fusion de résultats

Le module de fusion regroupe les R_i provenant des serveurs S'_i pour constituer une seule et unique liste de résultats R_Q qui sera présentée à l'utilisateur en réponse de sa requête Q (voire la figure 2.5).
Idéalement, les éléments de R_Q doivent être triés par ordre décroissant de pertinence. Cependant, et moins de charger tous les documents référencés par les R_i sont pour chaque document, son rang et éventuellement son score.

Si l'on se trouve dans le cas où le score de chaque document retourné est disponible, nous confrontés à l'incompatibilité des scores. En réalité, chaque serveur a ses propres données, son propre modèle de pondération, et bien d'autres caractéristiques qui lui sont propres, qui de ce fait, rendent incomparables les scores de ces documents.

Chaque moteur de recherche emploie son propre système de requête, du plus simple au plus complexe. Cependant, Quel que soit le type de requête proposé, un algorithme de recherche tente de déterminer les documents les plus pertinents répondant à la question posée. Dans la section suivante, nous allons étudier les différentes techniques utilisées et leur implémentation dans des systèmes opérationnels. Ces systèmes se basent sur différentes notions pour établir le degré de pertinence d'un document, allant de l'analyse linguistique et statistique des termes contenus dans le document à la nature des liens existants entre plusieurs documents.

2.2.1.3 Algorithmes de RI dans le Web

Les moteurs de recherche sur Web doivent faire face à de très fortes contraintes. Les outils de recherche doivent répondre à des millions de requêtes par jour alors que la quantité de documents qu'ils doivent analyser est gigantesque. La détection d'une mise à jour de ces documents est elle aussi source de grandes difficultés. Il n'y a pas de centralisation de ces données et un moteur de recherche doit par conséquent scruter en permanence le réseau en

vue de détecter ces changements. Cette problématique est bien illustrée par une citation de Sergey Brin, un des inventeurs de Google et de l'algorithme PageRank [75] : «*Le Web est une vaste collection de documents hétérogènes complètement incontrôlés*».
Pour toutes ces raisons, des recherches ont été menées afin d'alléger la charge des machines composant les systèmes de recherche. Les meilleurs résultats ont été obtenus par des méthodes utilisant les propriétés déduites de l'analyse de la topologie d'Internet.

2.2.1.3.1 Topologie d'Internet

L'examen de la topologie de connexion des pages Web entre elles amène des informations très importantes sur la manière de rechercher de l'information dans les documents présents sur Internet. La vaste distribution des pages Web à travers le monde entier oriente vers une première problématique : comment les documents sont-ils reliés entre eux et comment les atteindre tous ? Le diamètre de notre graphe est ainsi une variable prépondérante du problème.

Il existe principalement deux manières de modéliser la topologie d'Internet sous forme d'un graphe. La première consiste à considérer les nœuds de routage du réseau comme les nœuds du graphe et les liaisons entre les routeurs comme des arcs non orientés. Cette modélisation a un intérêt dans l'optimisation des algorithmes de routage d'un réseau [76] mais peu dans l'optique de la recherche d'information.

La seconde s'intéresse non pas aux connexions physiques du réseau mais aux relations qui existent entre les documents d'Internet. Cette modélisation représente Internet par un graphe orienté dont les nœuds et les liens sont respectivement représentés par les documents (contenu textuel, structure, . . .) et les URL des liens hypertextes qui relient un document à un autre. La topologie de ce graphe détermine la connectivité d'Internet et par conséquent nous renseigne sur la meilleure manière de rechercher de l'information dans ce réseau. Cependant, sa taille gigantesque, estimée à plusieurs milliards de nœuds et sa perpétuelle évolution tant au niveau des arcs que des nœuds rendent impossible l'analyse de l'ensemble de ces éléments.

Par conséquent la plupart des travaux réalisés dans ce domaine ont tenté de déduire les lois régissant ce réseau à partir d'un sous-graphe d'Internet. Ces lois sont ensuite vérifiées et validées par des simulations de réseaux à plus grande échelle afin de déduire les propriétés et les caractéristiques du graphe complet d'Internet.
Ces analyses statistiques nous permettent de mieux comprendre l'organisation des documents web afin de rendre des algorithmes de recherche les plus efficaces possible. Deux algorithmes d'analyse des liens contenus dans les différents documents se sont imposés grâce à leur grande efficacité. Il s'agit de HITS et de Page Rank.

2.2.1.3.2 L'algorithme HITS

Kleinberg fut un des premiers à s'intéresser aux propriétés de connectivité du graphe représentatif d'Internet et de son apport dans la détection de la pertinence d'une page à une requête [77]. Quelques constatations simples sont à l'origine de ses travaux dans ce domaine.

Comme nous avons pu le voir, l'importance d'une page peut être extraite de la structure des liens du Web. Dans cette approche, deux types de page sont identifiés en fonction de la nature de leurs connexions avec les autres documents. On retrouve d'une part les pages qui semblent être très importantes et jouent le rôle d'autorité sur un sujet donné et d'autre part les documents possédant un grand nombre de liens vers des pages faisant autorité sur un sujet. On distingue ainsi les pages autorités ayant un grand nombre de liens entrants et les pages *hubs* ayant un grand nombre de liens sortants et regroupant les autorités d'un même sujet. Le but de

l'algorithme HITS est de déterminer les *hubs* et les autorités qui renforcent leurs relations mutuellement sur un sujet donné. Ainsi Kleinberg dénombre les bons *hubs* comme des pages pointant vers beaucoup de bonnes autorités et les bonnes autorités comme des pages pointées par beaucoup de bons *hubs.*

Cette dénotation de bons *hubs* et de bonnes autorités fait apparaître une troisième catégorie de pages ayant un grand nombre de liens entrants provenant de documents n'ayant aucune particularité. Ces pages, que nous nommons pages indépendantes, sont considérées comme universellement populaires et n'apportent que peu ou pas d'intérêt [78]. Par exemple la page de Google est extrêmement référencée mais n'apporte que peu d'intérêt sur la plupart des sujets rencontrés ou une publicité aura de nombreux liens vers elle. Cette situation est illustrée dans la fiure 2.6**.**

Fig. 2.6 – Répartitions des pages en bon *Hubs*, bonnes autorités et pages indépendantes.

Une justification intuitive de l'autorité conférée à une page en fonction de la structure des liens l'entourant peut être donnée en considérant qu'un fort taux de jugement humain entoure l'ajout d'un lien hypertexte dans un document. En quelque sorte, l'auteur du document estime que la page vers laquelle il construit un lien évoque un sujet similaire à son souhait et paraît intéressante.

Pour déterminer les *hubs* et les autorités d'un sujet σ donné, l'algorithme HITS se base sur un sous-graphe d'Internet *Sσ* qui doit répondre aux conditions suivantes :

– *Sσ* est relativement petit,
– *Sσ* est riche en pages pertinentes,
– *Sσ* contient la totalité (ou la plupart) des plus importantes autorités.

En gardant Sσ petit, l'application d'algorithmes non triviaux peut s'effectuer sans s'occuper du temps de calcul nécessaire à la réalisation de la tâche. Les deux derniers points nous permettent de nous assurer d'avoir de bonnes chances de déterminer les bonnes autorités correspondant à la requête *σ*.

Pour construire un tel graphe, l'algorithme utilise une requête de mots-clés (*σ*) afin de prendre en compte un petit nombre de pages - environ 200 - depuis un moteur de recherche traditionnel à base d'index inversé. Cependant, cet ensemble de pages, noté *Rσ*, ne contient pas nécessairement l'ensemble des autorités du sujet σ. Par exemple, il y a de fortes chances pour que les réponses à la requête du moteur de recherche ne correspondent pas aux grands sites de moteurs de recherche : ces pages ne contiennent en effet que très rarement les mots-clés recherchés. On peut toutefois espérer que ce sous-graphe contienne des liens vers des autorités ou des *hubs* importants. *Rσ* est alors étendu en ajoutant les nœuds pointés par le sous-graphe et les nœuds pointant le sous-graphe afin d'obtenir un graphe augmenté Sσ. L'auteur affirme alors que les trois conditions requises par le sous-graphe Sσ sont respectées.

Les bons *hubs* et les bonnes autorités peuvent être extraits de ce graphe en donnant une définition numérique à la notion de *hub* et d'autorité. L'algorithme HITS associe un vecteur de potentiels d'autorités non négatifs *hxi* et un vecteur de potentiels de *hubs* non négatifs *hyi* pour toutes les pages appartenant à *Sσ*. Ainsi, une page p possédant un fort potentiel d'autorité *xp*, respectivement un fort potentiel de *hub yp*, sera vue comme étant une bonne autorité, respectivement un bon *hub*.

Initialement, aucune hypothèse n'est faite sur les potentiels, hub et autorité de chaque document. Ainsi les vecteurs originaux *hxi* et *hyi* sont uniformes. Les poids sont ensuite mis à jour de la façon suivante : si une page est pointée par plusieurs bons hubs, nous incrémentons son potentiel d'autorité. Ainsi, pour une page p donnée, la valeur de son poids *xp* est mise à jour par la somme des potentiels *yq* de toutes les pages q pointant sur p. D'une manière symétrique, les potentiels de hub sont mis à jour en fonction des potentiels d'autorités des pages vers lesquelles notre document en cours pointe. Ces opérateurs sont décrits dans la figure 2.7.

La sortie de l'algorithme HITS est alors composée de deux listes ordonnées décroissantes de pages en fonction des potentiels respectifs de hub et d'autorité. L'originalité vient du fait que seule la topologie des liens entre les documents est prise en compte afin de déterminer les pages les plus pertinentes sur un sujet donné. Les expérimentations réalisées par Kleinberg montrent que sur une requête problématique pour un moteur de recherche basé uniquement sur un index inversé,

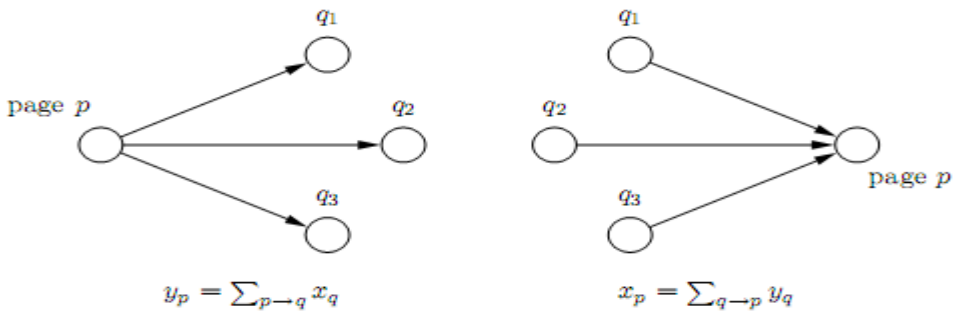

Fig. 2.7 – Opérateurs de mise à jour des accumulateurs Hub (*yp*) et Autorité (*xp*).

Les résultats obtenus sont très satisfaisants : la requête search engine -posée en 1999 - produit les sites Yahoo !, Excite, Magellan, Lycos et Altavista comme autorités bien que leur texte ne contienne pas la requête initiale.

La première grande implémentation de HITS fut réalisée au travers du système ARC et de l'algorithme CLEVER conçus au centre de recherche IBM Almaden et décrits dans [78]. Les auteurs ont tenté d'apporter certaines modifications en vue d'améliorer les résultats.

L'étape d'initialisation de ce dernier algorithme est essentiellement la même que celle de HITS. Le noyau de pages *Rσ* est cependant augmenté en suivant deux liens entrants et sortants afin de produire un sous-graphe Sσ plus important. La mise à jour des potentiels est elle aussi revue en prenant en compte une nouvelle pondération sur les sommes en fonction du texte des pages concernées. Ces poids sont basés sur le nombre d'apparitions des termes n(t) de la requête dans les alentours de chaque lien exploré. La zone d'exploration autour de chaque lien a été déterminée de manière statistique à 50 caractères de part et d'autre du lien. Ainsi, les arcs du graphe Sσ sont dorénavant pondérés par :

$$w(p, q) = 1 + n(t) \quad (1.9)$$

D'autres auteurs ont tenté d'améliorer encore l'algorithme de départ, notamment [79] qui propose une correction sur le calcul des potentiels de hub. En effet, si on considère un graphe possédant $M + 1$ pages *hubs* et $M + 1$ pages autorités avec $M \gg 0$, et si les premiers M *hubs* pointent uniquement sur la première autorité, le *hub* restant pointant sur toutes les $M+1$ autorités, on devrait s'attendre à ce que le dernier *hub* obtienne un plus faible score que les autres du fait qu'il pointe vers beaucoup de mauvaises autorités. Cependant, l'algorithme de base fera de lui le meilleur *hub*. La modification proposée altère la mise à jour des *hubs* en leur donnant pour valeur la moyenne des scores des autorités vers lesquelles ils pointent.

Toutes ces méthodes reposent sur un sous-graphe d'Internet afin de réduire les temps d'exécution des algorithmes. Ce sous-graphe est obtenu à partir d'une requête donnée. Le calcul de pertinence est alors réalisé pour chaque requête parvenue au moteur de recherche. Il existe cependant des méthodes indépendantes de la requête proposée par l'utilisateur qui peuvent déterminer une pertinence absolue des documents du réseau. La plus célèbre repose sur l'algorithme du PageRank [80] et est à l'origine du moteur de recherche Google.

2.2.1.3.3 L'algorithme PageRank

L'idée principale de cette méthode est de simuler le comportement d'un internaute naviguant de manière aléatoire sur Internet. La probabilité qu'il visite une page donnée est d'autant plus grande que cette page est pointée par beaucoup d'autres pages au travers de leurs liens hypertextes. En considérant qu'une page confère une certaine autorité à une autre page en établissant un lien vers elle, la probabilité de passage de cet internaute aléatoire sur une page indique le degré de pertinence de ce document.

Il existe deux manières d'accéder à une page Web sur Internet. On peut d'une part l'atteindre directement en connaissant son adresse et d'autre part suivre un lien hypertexte d'un autre document. Le calcul du PageRank - et donc de la pertinence - d'une page intègre ces deux éléments au travers d'une probabilité d. d représente en quelque sorte la probabilité que l'internaute aléatoire s'ennuie sur une page et décide de choisir une autre page au hasard. L'équation 1.10 montre la récursion permettant de calculer le PageRank pour tous les nœuds du graphe représentant Internet.

$$PR(pj)_t = (1-d) + d \sum_{\substack{i=1 \\ pi \to pj}}^{n} \frac{PR(p_i)_{t-1}}{C(p_i)} \quad (1.10)$$

Dans cette formule, $PR(pj)t$ représente la valeur du PageRank à l'itération t pour la page pj . $C(pi)$ est défini comme le nombre de liens sortants de la page pi Le paramètre d prend ses valeurs dans l'intervalle [0−1] et est généralement placé à d = 0.85 d'après des études statistiques menées par Lary Page dans [80]. Ce dernier paramètre permet de faire converger l'algorithme de manière plus ou moins rapide. En effet, plus d est élevé, plus l'effet de l'ajout d'un lien entrant vers une page est accru et plus celui-ci se propagera dans toutes les pages d'un même site.

Le PageRank forme ainsi une distribution de probabilités des pages Web. Le calcul peut s'effectuer de manière itérative et converge vers une valeur asymptotique de manière assez rapide. En effet, le calcul effectué dans [80] sur un graphe de 26 millions de nœuds en considérant $d = 0.85$, converge en seulement 52 itérations. On constate dans la formule que pour chaque itération toutes les pages distribuent leur PageRank en fonction de leurs liens sortants. Cette distribution s'effectue de manière plus ou moins importante en fonction du nombre de liens $C(pi)$ de chaque page pi (distribution uniforme). Cela peut aisément

s'interpréter en considérant qu'une page possède un certain potentiel de pertinence et que l'auteur de la page donne une certaine crédibilité - son PageRank - aux pages pour lesquelles il inscrit un lien.

La formule initiale suppose que la probabilité de sauter sur n'importe quelle page est uniforme. Mais une des plus importantes variations de cet algorithme consiste à attribuer une distribution de probabilité non uniforme afin de mieux caractériser la réalité d'Internet [80]. La première partie de l'équation 1.10 devient alors $(1 - d). Epj$.
L'équation 1.10 peut être reformulée de manière matricielle. Pour cela, on considère deux vecteurs colonne PR et E correspondant respectivement au rang de chaque page et à la loi de distribution de l'ensemble des nœuds du graphe - loi de distribution uniforme dans la formule initiale mais personnalisable comme vu par exemple dans le paragraphe précédent -, et une matrice A telle que l'entrée $A[p, q] = 0$ s'il n'existe pas de lien sortant de la page q vers la page p et $A[p, q] = 1/ C(p)$ sinon. A est une matrice sous-stochastique. Cette équation prend ainsi la forme suivante :

$$PR = (1 - d)E + d \, . \, A^t \, . \, PR \qquad (1.11)$$

Où A^t désigne la transposée de la matrice A. La sommation de l'équation 1.10 peut être représentée par la multiplication matricielle $A^t \, . \, PR$. Ainsi, PR correspond au vecteur propre principal de la matrice normalisée des liens du Web.

Afin d'améliorer les performances de cet algorithme, une autre variation a été introduite conduisant à modifier, dans l'évaluation du PageRank d'une page, l'importance de l'apport de chaque lien au calcul final. Des heuristiques de modifications du poids de chaque lien ont notamment été étudiées dans [78] et [81]. Ces heuristiques sont principalement basées sur l'analyse des textes entourant et présents sur les différents liens hypertextes contenus dans les pages Web et sur l'utilisation de mesures de similarités basées sur le modèle vectoriel et la mesure *tf*idf* (voir section 2.2.1.1.1.3).

Deux problèmes avec l'approche du PageRank sont soulignés par Kleinberg dans [77]. Premièrement, plusieurs pages peuvent être considérées comme des autorités particulièrement importantes sur des sujets donnés mais ne contenant pas les termes de la requête correspondante. Cela pose un problème de par la méthode de sélection des pages utilisée et décrite dans cette section : la sélection des réponses pertinentes se fait simplement à l'aide d'un index inversé et le tri de ces pages est effectué grâce au score obtenu par le PageRank. Les solutions obtenues contiennent alors forcément les termes de la requête.

Un autre problème est souligné concernant les pages qui ont un nombre de liens entrants très important comme par exemple www.yahoo.com. Ces pages auront un PageRank très élevé. Cependant, si la requête proposée contient un mot compris dans ces pages, en reprenant l'exemple précédant avec le terme Automobiles, ces sites vont être retournés à l'utilisateur en tête de liste bien qu'ils ne fassent pas du tout autorité sur le sujet donné. Kleinberg conclut d'ailleurs en disant que l'utilisation unique de la structure des liens entrants ne permet pas d'obtenir un équilibre correct entre la notion de popularité et la notion de pertinence.

Afin de corriger ces éventuels problèmes, plusieurs auteurs ont tenté d'effectuer une hybridation entre les algorithmes PageRank et HITS. Citons par exemple HubRank [82] qui modifie la matrice de personnalisation de distribution de probabilité E de l'équation 1.11 en prenant en compte à la fois la distribution des liens sortants et des liens entrants du graphe de

connexion des pages Web. Le surfer aléatoire va alors préférer aller sur une page ayant un fort degré de liens sortants lorsqu'il s'ennuie - avec la probabilité *(1 − d)*.

Un autre exemple d'amélioration intéressante est donné dans [83]. Les auteurs ont cherché à combiner les avantages de l'algorithme HITS avec l'algorithme PageRank. Il utilise la notion de parcours aléatoire du graphe d'Internet présent dans l'algorithme HITS permettant de détecter les pages aux caractéristiques de *hubs* et d'autorités avec l'initialisation du surfeur aléatoire présent dans la première partie de l'équation du PageRank 1.10. Les résultats obtenus par les auteurs montrent alors qu'un nouvel algorithme est plus robuste aux perturbations rencontrées dans les différents graphes de test.

Il s'agit de l'algorithme alternatif, SALSA, a été proposé par Lempel et Moran en 2000 [39]. Son but est similaire à celui de l'algorithme HITS décrit dans la section 2.2.1.3.2: il cherche à déterminer les meilleurs pages correspondant à un sujet donné en les caractérisant en hubs et autorités. Cet algorithme SALSA a été généralisé par Mendelzon [84]. Il s'agit en fait d'une hybridation de cet algorithme et de l'algorithme du Pagerank afin d'établir la réputation de chaque page. Les auteurs ont adapté le facteur de zap de Brin et Page en considérant qu'avec une probabilité d, l'algorithme de Mendelzon saute sur une nouvelle page suivant une loi de probabilité uniforme et avec une probabilité *1−d*, il effectue une itération de l'algorithme SALSA.

D'autres approches de détermination de *hubs* et autorités ont également été testées. Cohn et Chang ont proposé un algorithme statistique PHITS afin de déterminer ces deux catégories [85]. Le modèle que les auteurs ont construit tente d'expliquer deux types de variable, les citations *c* d'un document d en fonction d'un petit nombre de variables communs z qui sont appelées les aspects ou les facteurs. Ces variables communes peuvent être considérées comme des sujets ou des communautés de pages. Le modèle peut alors être décrit statistiquement : un document d $d \in D$ est généré avec une probabilité *P(d)*, le facteur, ou sujet $z \in Z$ associé à d est choisi en fonction d'une probabilité *P(z|d)*, et étant donné ce facteur, des citations $c \in C$ sont générées en fonction de la probabilité *P(c|z)*. La probabilité de chaque paire (document, citation) *(d, c)* est alors décrite par :

$$P(d,c) = P(d)P(c/d)$$
$$P(c/d) = \sum_z P(c/z)\,P(z/d) \qquad (1.12)$$

Comme nous avons pu le voir, les techniques de recherche d'informations sont très variées et prennent en compte des éléments très différents. À l'heure actuelle, l'analyse de la topologie d'Internet est un élément prépondérant dans la détection de la pertinence d'un document. Ce système permet en effet de récupérer un certain jugement humain inscrit dans les liens hypertextes : lorsqu'une personne décide d'inscrire un lien hypertexte dans une page Web, elle considère que le document pointé par ce lien apporte une information utile.

Ces techniques de recherche d'informations sont abordées par les différents moteurs existants. Dans la section suivant, nous présentons les architectures des moteurs de recherche ainsi que ses principes de fonctionnement.

2.2.1.4 Architectures des moteurs de recherche

Les architectures utilisées dans les systèmes de recherche d'information ont beaucoup évolué au cours de la dernière décennie, passant de systèmes monolithiques (SRI décrit dans la section 2.2.1.1) particulièrement peu adaptatifs à des systèmes fortement distribués (SRID

décrit dans la section 2.2.1.2) s'ajustant en permanence à la quantité d'information à analyser. Cette évolution a été rendue nécessaire - et même obligatoire - car le nombre d'internautes ne cesse de progresser. De plus, les moteurs de recherche sont devenus le point de départ de beaucoup de sessions de navigation sur Internet et il a donc fallu trouver des solutions pour répondre en des temps raisonnables aux innombrables requêtes formulées à chaque instant.

L'architecture originale utilisée par Altavista représente la première catégorie de systèmes. Son fonctionnement est décrit dans la figure 2.8**.** Il s'agit d'une architecture très simple qui se divise en deux parties distinctes. On retrouve d'une part un crawler et d'autre part l'interface d'interrogation du moteur de recherche et le système d'analyse des requêtes proposés par les utilisateurs du système. Le crawler peut être considéré comme un robot chargé de rapatrier tous les documents Web contenus sur Internet dans un index centralisé en suivant les liens hypertextes rencontrés dans les pages analysées [86].

Le cœur du système repose sur un index inversé permettant d'associer des mots à un ou plusieurs documents. La demande de l'utilisateur est traitée en interrogeant l'index inversé pour connaître les documents dans lesquels apparaissent le plus souvent les mots de la requête.

Ce système est très peu évolutif et nécessite un matériel très avancé. Par exemple, en 1998, Altavista utilisait un système comprenant 20 processeurs auxquels étaient alloués 130 Go de mémoire vive et 500 Go de disque dur. Dans ce modèle, le moteur de requête

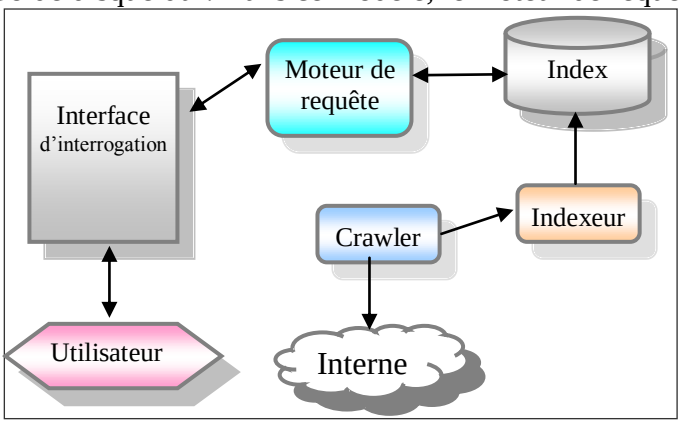

Fig.2.8 Architecture originale du moteur de recherche Altavista

Consommait à lui seul 75% des ressources mises à disposition par le système. Cette architecture monolithique ne permet pas de faire évoluer le système dynamiquement. Or, Internet est en perpétuelle expansion ce qui rend ce système obsolète aujourd'hui.

Depuis ces années - et surtout depuis l'apparition du moteur de recherche Google - l'architecture a évolué vers les modèles décrits dans la suite de cette section.

2.2.1.4.1 Vers un modèle distribué et adaptatif

Des variantes de l'architecture précédente, basées sur le modèle indexeur-crawler, ont été imaginées afin de gommer les défauts inhérents à sa conception. L'une d'entre elle, appelée Harvest s'est révélée très innovante en matière de distribution des ressources. Cette architecture est décrite également dans [86] et a été utilisée par de nombreux organismes comme la NASA, l'Académie des Sciences Nationale des Etats-Unis d'Amérique.

Comme le montre la figure 2.9, cette architecture se développe autour de deux composantes principales : le récolteur et le broker. Chacun de ces éléments a un rôle particulier à jouer dans la chaîne de traitement du moteur de recherche. Le récolteur est

chargé de collecter et d'extraire périodiquement des informations d'indexation (textes, images, vidéo,…), depuis plusieurs sites Web. Le broker, quant à lui, fournit le mécanisme d'indexation et l'interface d'interrogation sur les données amassées par le récolteur. On retrouve ici, le mécanisme indexeur-crawler identifié dans la section précédente. Cependant, plusieurs brokers et plusieurs récolteurs peuvent communiquer ensemble, chacun se spécialisant dans un domaine précis. Lorsqu'une requête est émise sur un broker dont le domaine traité ne correspond pas à ses capacités, celui-ci transmet la requête à une autre entité capable de la gérer.

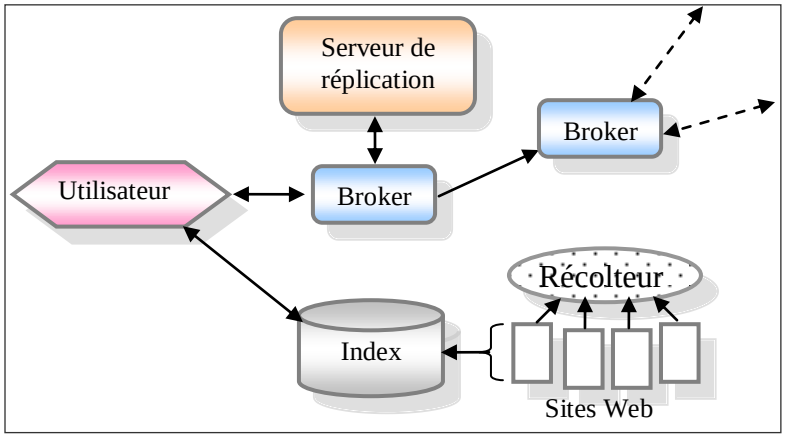

Fig. 2.9 – Architecture du système Harvest.

C'est un système totalement adaptatif dans lequel il est possible de configurer les brokers et les récolteurs de manière à répartir le besoin en ressources sur un ou plusieurs domaines particuliers. Un système de réplication permet de plus de garantir une qualité de service relativement fiable.

Lancé en 1998, Google a révolutionné le monde des moteurs de recherche. Son système de classement des résultats, mais aussi sa grande rapidité d'exécution des requêtes, l'ont rapidement placé en tête des outils de recherche d'information sur Internet.

2.2.1.4 Architecture moderne d'un moteur de recherche

L'architecture du moteur de recherche Google est certainement une des plus efficaces actuellement. Elle ne repose pas sur un système monolithique mais sur un grand nombre de machines classiques coopérant ensemble. La figure 1.10 montre l'organisation des différents éléments qui composent cette architecture.

Ce système peut se décomposer en plusieurs parties comprenant un sous système d'exploration d'Internet, un indexeur, un analyseur de la topologie d'Internet formée par les liens hypertextes et un sous système de présentation et d'exécution de requêtes.

Un serveur d'URL garde la mémoire des liens des pages à visiter. Des robots chargés d'explorer le Web récupèrent ces liens afin de télécharger les documents correspondant et les stocker dans une base de données recensant la totalité des pages indexées. Cette opération est réalisée continuellement et alimente et met à jour en permanence la base de documents du moteur. Périodiquement, cette base est analysée afin de réaliser un index inversé reliant des termes aux documents les contenant. D'autres informations sur les termes sont extraites comme leur position dans le document, la taille de la police utilisée ou sa fonte. Cet index inversé est distribué sur une multitude d'ordinateurs désignés par le terme barrel sur le la figure 2.10.

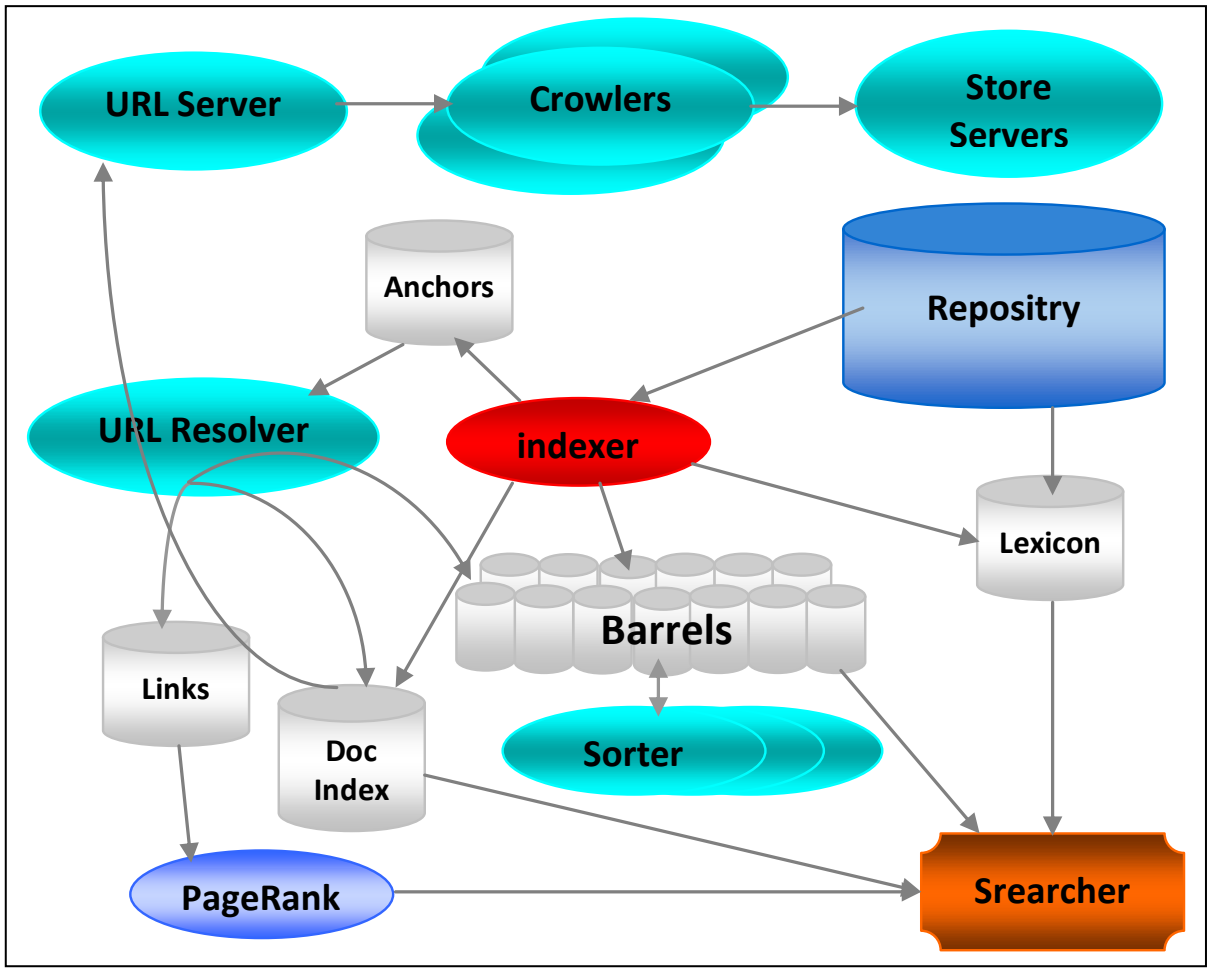

Fig. 2.10 – Architecture du moteur de recherche Google.

Cette analyse permet également d'extraire tous les liens hypertextes des documents rencontrés afin d'alimenter le serveur d'URL. Cette base de liens est utilisée afin de calculer le PageRank (voir section 2.2.1.3.3) permettant de trier les documents de l'index par pertinence décroissante.

Cette architecture est distribuée sur un grand nombre de machines standard et permet d'obtenir des temps de réponse inférieurs à la seconde pour une requête donnée. C'est actuellement le moteur de recherche le plus utilisé et il a montré, depuis son lancement en 1998 une très grande adaptabilité à l'évolution de la taille d'Internet jusqu'à indexer une dizaine de milliards de pages Web.

Les trois architectures de moteurs de recherche présentées montrent une évolution constante dans la rapidité et l'adaptabilité des outils de recherche s'adressant à un grand nombre d'utilisateurs. Ce type de système est devenu le point de départ d'un grand nombre de sessions de navigation sur Internet et a du s'adapter à l'afflux de requêtes toujours plus important.

Mais aussi fiables que peuvent l'être ces techniques, un outil de recherche doit présenter les résultats à un utilisateur de manière organisée. En effet, devant le nombre impressionnant de liens retournés par un moteur de recherche pour une requête donnée, on peut rapidement se sentir submergé. Par conséquent, il est nécessaire d'étudier la meilleure manière de présenter les résultats d'une recherche afin de faciliter leur lecture et leur compréhension.

2.2.1.4.3 Les méta-moteurs de recherche

Les méta-moteurs présentent des stratégies de recherche beaucoup plus hétérogènes que ce que l'on peut trouver dans les moteurs de recherche classiques. Cependant, tous ont pour point commun d'utiliser les résultats produits par ces mêmes moteurs de recherche.

La plupart de ces outils ne fait que trier les liens récupérés de plusieurs sources en utilisant ses propres algorithmes de détection de pertinence. Dans ce cas, leur but est de tirer parti de la complémentarité et de la spécialisation de plusieurs outils de recherche. Dans ce cas ces outils se distinguent par les fonctions de tri utilisées qui sont propres à chaque méta-moteur.

Cependant, il existe d'autres méta-moteurs que l'on peut qualifier de plus évolués. Ils ne se contentent pas de compiler les résultats d'autres moteurs de recherche mais parcourent également Internet à la recherche de documents pertinents. Il n'existe pas dans ce cas d'organisation type et chacun utilise ses propres stratégies afin de se distinguer des autres méta-moteurs.
Nous examinerons dans la section 2.2.5.2 un méta-moteur particulier : I*nfoSpiders*. Il a la particularité d'utiliser plusieurs agents de recherche spécialisés et capables d'apprendre en fonction des souhaits de l'utilisateur.

D'autres ressemblent plus à des partages de ressources en regroupant des communautés d'utilisateurs. Par exemple, *AntWorld* [87] est un outil qui prend en compte le jugement que portent des utilisateurs sur les documents qu'ils parcourent. Pour chaque page qu'il juge intéressante, un utilisateur donne les mots-clés correspondant à la page ainsi qu'un avis documenté sur celle-ci. Une base de données recense et centralise tous les avis. Si la communauté est importante, le nombre de documents accessibles devient significatif. Les résultats obtenus par l'intermédiaire d'un tel moteur sont alors très pertinents car classés par des experts.

Dans un moteur de recherche, les systèmes d'interrogation et de sélection des pages jouent un grand rôle. Mais l'énorme quantité de données à traiter et l'évolution permanente de l'information qui augmente sans cesse imposent d'avoir une architecture du système de recherche particulièrement élaborée.

2.2.2 RI à base des Systèmes Multi-Agents

D'après Koch *et al.* [88], le *paradigme agent* offre des méthodologies et des mécanismes pour la création d'applications distribuées, intelligentes, intégrées et coopératives.

Un agent peut fournir des informations adaptées au système de RI auquel il accède. Dans le Web, le contexte de la recherche de l'utilisateur est dynamique puisque les utilisateurs peuvent se déplacer d'un site à un autre, d'un endroit à un autre. Les sources d'informations sont multiples et hétérogènes, le volume d'information est en évolution considérable. Ces changements de l'environnement du Web provoquent des changements dans les tâches et les besoins d'information de l'utilisateur. En conséquence, le paradigme agent peut fournir un apport important à la RI dans le Web.

2.2.2.1 Définition des Systèmes Multi-Agents et caractéristiques des agents

Rahwan *et al.* [89] considèrent un *agent* comme une entité qui possède un comportement autonome et flexible où flexible signifie à la fois :

- *réactif,* pour percevoir l'environnement et répondre d'une manière opportune ;
- *proactif,* pour exhiber un comportement dirigé vers les buts à atteindre et pour prendre des initiatives grâce à sa connaissance (définie, acquise, inférée) ;
- *communicatif,* pour interagir avec d'autres agents ou humains.

Wooldridge *et al.* [90] ont aussi mis en valeur le comportement flexible (*proactif*) et autonome des agents pour résoudre des problèmes, la richesse de l'interaction des agents (*caractéristiques de communication* avec d'autres agents ou avec l'utilisateur) en vue d'accomplir un objectif commun ou propre objectifs, et la complexité de la structure d'organisation d'un système d'agents (*sociabilité*). Un agent accomplit ses actions lorsqu'il se situe dans un environnement particulier (par exemple, informatique, physique, ubiquitaire) et il décide par lui-même du moment auquel doivent être accomplies ces actions.

Pour qu'un agent soit une entité réactive, proactive et communicative, Lopez *et al.*[91] le décrivent à travers une représentation composée d'un ensemble d'attributs (représentant ses caractéristiques permanentes), de l'ensemble des buts à accomplir, de l'ensemble de ses capacités d'exécution, et de l'ensemble de ses préférences. Cette représentation considère les agents comme des entités sociales capables d'éviter ou de résoudre des conflits, de concrétiser des accords, de réduire la complexité des tâches, et en général, d'accomplir des tâches communes. Les agents doivent être contrôlés et coordonnés pendant l'exécution de leurs tâches, et si toutes les tâches de l'agent aident l'utilisateur dans ses activités, c'est bien l'utilisateur qui doit toujours avoir le contrôle.

Lopez *et al.* [91] soulignent le besoin d'établir des normes et des règles afin de contrôler le comportement de l'agent. Ils considèrent que les agents doivent être dotés de capacités à adopter de nouvelles normes. En matière de contrôle des activités des agents, Weib *et al.* [92] proposent le concept de « *RNS* » (« *Roles, Norms, Sanctions* »), un schéma spécifiant les limites du comportement autonome de l'agent en termes de rôles à travers lesquels l'agent peut accomplir les buts et tâches assignés. Du point de vue de l'autonomie de l'agent, Wooldridge *et al.* [93] considèrent qu'un agent est une entité « *stand-alone* » capable de représenter une personne ou une organisation. Bien qu'autonome l'agent est amené à évoluer au sein d'une communauté. A ces fins, l'*OMG* (*Object Management Group*) a établit des normes de comportement et d'exécution des agents.

- Pour J.Ferber **[94].** Un système multi-agents est une communauté d'agents autonomes travaillant en commun, selon des modes parfois complexes de coopération, conflit, concurrence, pour aboutir à un objectif global : la résolution d'un problème, l'établissement d'un diagnostic. Les systèmes multi-agents ou SMA forment une branche de l'Intelligence Artificielle dans laquelle des métaphores sociologiques ou biologiques sont employées pour la conception et la mise en œuvre de systèmes artificiels intelligents.

Il existe cependant de nombreuses catégories d'agents (agent bureautique, agent de notification…) qui sont notamment présentées dans [95]. Dans ce document, nous mettons l'accent sur les agents de recherche, les agents de recommandation puisqu'ils interviennent dans le contexte de la Recherche d'Information sur le Web mais également sur les Systèmes Multi-Agents SMA qui peuvent apporter de la valeur ajoutée dans l'aide à la recherche.

2.2.2.2 Les agents de recherche

La catégorie des agents de recherche comprend les méta-moteurs de recherche d'information de dernière génération jusqu'aux outils de recherche off-line (Copernic par exemple). Ces derniers permettent à l'utilisateur d'initier des recherches d'informations qui s'exécuteront même lorsque l'utilisateur ne sera plus connecté à Internet.

2.2.2.3 Les agents de recommandation

Les agents de recommandation, visent à optimiser la recherche d'information de l'utilisateur en lui proposant automatiquement de nouveaux documents au regard de ses besoins ou de ses actions. Ils reposent essentiellement sur une approche Push proposant des informations à l'utilisateur et une caractérisation des besoins au moyen d'un profil utilisateur. Nous présentons différentes catégories d'agents de recommandation en fonction des buts recherchés.

Dans un premier temps, nous soulignons les agents qui proposent les liens qui mènent vers des informations répondant aux besoins de l'utilisateur. Ceci permet notamment à l'utilisateur d'optimiser sa navigation en se consacrant à des documents potentiellement pertinents. Les systèmes Letizia [96], WebWatcher [97] ou encore BroadWay [98] sont de bons représentants de cette catégorie.

Letizia utilise les documents visités comme base du profil utilisateur. Celui-ci est construit à partir des termes issus des documents visités et des actions de l'utilisateur (ajout dans les signets par exemple). Lors de la visite d'un document, Letizia télécharge le contenu de tous les documents liés au document courant pour en évaluer l'appariement avec le profil utilisateur. Les liens menant vers des documents pertinents sont présentés à l'utilisateur au moyen d'une fenêtre annexe.

Webwatcher demande à l'utilisateur de formuler ses besoins explicitement (au travers d'une requête). Chacun des documents visités est annoté par les termes de la requête de l'utilisateur ainsi que par un jugement de pertinence de la part de l'utilisateur. A chaque visite d'un document, Webwatcher recherche les liens pointant vers des documents similaires aux besoins de l'utilisateur. Cet appariement est réalisé grâce aux annotations laissées par les utilisateurs ayant précédemment visité le document et la requête de l'utilisateur qui navigue. Les liens pointant vers des documents susceptibles d'intéresser l'utilisateur sont présentés directement au sein du document visité au moyen d'icônes.

Broadway poursuit le même but mais s'appuie sur une démarche légèrement différente car il repose sur l'idée que si plusieurs utilisateurs suivent le même parcours au sein de l'hypertexte, c'est vraisemblablement qu'ils recherchent les mêmes informations. Le profil utilisateur repose cette fois-ci sur une représentation de la navigation en cours. La représentation de la navigation est construite à partir de séries temporelles correspondant à quatre variables : le vecteur descripteur du document visité, l'évaluation explicite faite par l'utilisateur, le temps de lecture et l'URL du document. L'échelle de temps appliquée à ces séries temporelles correspond à une visite. Pour identifier les documents pertinents, Broadway utilise le concept de raisonnements par cas sur les représentations des navigations antérieures. Les documents ainsi retrouvés sont présentés à l'utilisateur au moyen d'une fenêtre annexe.

Du point de vue du résultat, la principale différence entre ces trois agents est la profondeur de recommandation. Dans Letizia ou Webwatcher, les recommandations ne concernent que les documents directement liés au document courant, tandis que Broadway anticipe en quelque sorte la navigation en proposant des documents liés par transitivité au document courant.

Les systèmes précédents n'indiquent à l'utilisateur que les documents potentiellement pertinents pour la navigation en cours. Or, il est tout aussi intéressant d'informer l'utilisateur sur le fait que le document qu'il est en train de visiter est pertinent (ou non pertinent) pour les thèmes qui l'intéressent. Le système Syskill & Webert [99] vise à remplir cette tâche. L'utilisateur peut créer autant de thèmes que nécessaire avec la restriction que les thèmes sont organisés sous forme de liste non hiérarchique. A chaque fois qu'il trouve un document pertinent (ou non pertinent) pour les thèmes qu'il a créé, l'utilisateur le signale au système qui met à jour un profil correspondant au thème. Lors de la navigation, le système indique à l'utilisateur si le document qu'il visite est pertinent (ou non pertinent) pour les thèmes qu'il possède.

2.2.2.4 Systèmes de RI dans le Web à base des SMA

Les SMA et les agents logiciels ont tout d'abord été utilisés afin de résoudre des problèmes intrinsèquement distribués et de simuler des phénomènes complexes. Il sont dorénavant aussi utilisés afin de développer des systèmes de recherche et de classification de l'information. De tels systèmes donnent la possibilité de concevoir des outils robustes et adaptatifs. La FIPA (Foundation For Physical Agents) propose de nombreux standards et recommandations facilitant l'utilisation du paradigme agent dans des applications réelles. Plusieurs méthodologies ont aussi été proposées permettant d'accompagner les phases de conception et d'implémentation contribuant ainsi à la constitution d'un génie logiciel orienté agents [100].

Fab [101] repose également sur une approche multi-agents qui rapproche les individus par centres d'intérêt. Le système collecte ensuite des documents sur le web grâce à un agent spécifique qui recherche les thèmes intéressant des utilisateurs, pour les leur proposer. Un document est recommandé s'il correspond bien au profil de l'utilisateur ou s'il correspond au profil d'un utilisateur proche. Le profil est construit à partir des jugements exprimés par les utilisateurs concernant les documents recommandés.

Le système **Watson** [102] apporte une innovation concernant l'identification d'expressions régulières dans les documents visités (comme les adresses par exemple) afin de proposer des services contextuels à l'utilisateur. Pour une adresse identifiée, Watson propose à l'utilisateur la génération d'un plan urbain permettant de localiser celle-ci. L'interface de Watson est présentée dans une fenêtre indépendante.

Dans le contexte de l'étude de l'environnement de travail de l'utilisateur pour la construction de son profil, l'approche de **Suitor** [103] innove par différentes fonctionnalités. Il utilise notamment une fenêtre déroulante permettant d'afficher les informations proposées par Suitor en base de l'écran, mais il propose surtout un système permettant de suivre le regard de l'utilisateur afin d'identifier les parties de l'écran qu'il visionne.

Dans une toute autre approche, **SiteSeer** [104] est un agent permettant de proposer à un utilisateur des signets directement dans la hiérarchie de signets qu'il possède. Ce système

repose sur la couverture entre les différents répertoires des hiérarchies de signets des utilisateurs du système. En effet, plus la couverture entre le contenu de deux répertoires de deux utilisateurs respectifs est importante, plus les deux répertoires traitent d'un thème similaire. Ainsi, les documents n'entrant pas dans la couverture composée par le contenu des deux répertoires sont jugés pertinents et sont proposés pour le répertoire de l'utilisateur qui ne les possède pas.

2.2.2.4.1 Le projet Marvin

Les SMA visent à faire coopérer une série d'agents afin d'obtenir un résultat. Un modèle général de coopération peut être vu dans [105]. Ainsi, dans le cadre de la recherche d'information, il peut être intéressant de faire coopérer les agents pour répondre à des besoins précis en information d'un utilisateur.
Par exemple, l'approche multi-agents proposée au travers du projet Marvin (http://www.hon.ch/Project/Marvin_project.html) permet de construire des collections thématiques de documents (Figure 2.11). Chaque agent parcours le Web à la recherche de documents pour un thème donné (exemple la médecine).

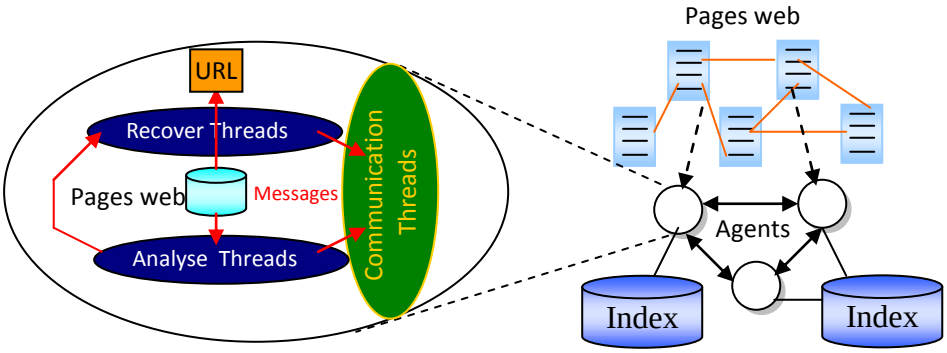

Fig 2.11 - Architecture multi-agents de Marvin

L'intérêt des approches multi-agents peut également être vu au travers du projet Abrose [106] qui permet de construire et de maintenir un profil utilisateur pour le commerce électronique mais qui pourrait être adaptée à la recherche d'information. Ce profil utilisateur contient l'ensemble des préférences d'utilisation d'Internet d'un utilisateur qui peut être utilisé pour répondre à des requêtes de façon plus fine ou proposer à l'utilisateur de nouveaux documents ou offres plus intéressantes. L'évolution des préférences est adaptative et est caractérisée par un ensemble d'agents.

2.2.2.4.2 Le projet AGATHE

Le système AGATHE [100]. (Agent information GATHEring) est une architecture logicielle générique permet le développement de systèmes de RI sur le Web sur un ou plusieurs domaines restreints. AGATHE met en œuvre une collecte coopérative d'information à base d'agents et d'ontologies. Ce système prend en compte des contextes de recherche en considérant des regroupements de pages Web relatifs à des domaines spécifiques (par exemple la recherche académique, le tourisme, …).

Architecture général d'AGATHE

L'architecture générale du système AGATHE, illustrée à la figure 2.12, s'articule autour de trois principaux sous-systèmes en interaction :

- 1) Le *sous-système de Recherche* (SSR) est chargé de l'interrogation des moteurs de recherche externes sur le Web (comme Google) afin d'obtenir des pages Web qui seront traitées par le sous-système d'Extraction.
- 2) Le *sous-système d'Extraction* (SSE) composé de différents « cluster d'extraction » (CE), chacun spécialisé dans le traitement de pages Web sur un domaine spécifique (comme celui de la recherche académique, ou celui du tourisme).
- 3) Le *sous-système d'utilisation* (SSU) assure le stockage des données extraites à partir des pages Web traitées par le sous-système d'Extraction, et fournit une interface d'interrogation pour des utilisateurs, pouvant être des humains ou d'autres agents logiciels.

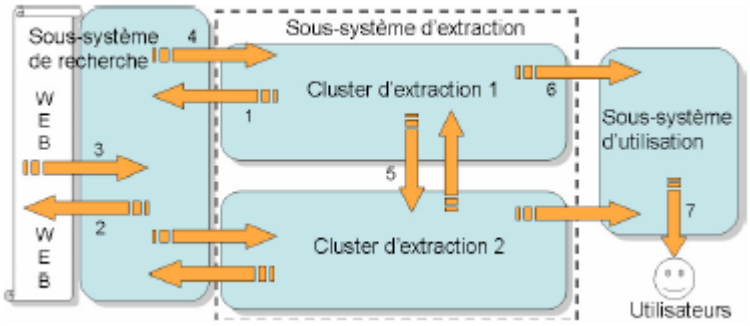

Fig 2.12 : Architecture générale d'AGATHE.

Ces trois sous-systèmes principaux du système AGATHE sont des systèmes multi-agents composés d'agents logiciels (agents informationnels) plus ou moins intelligents. Certains d'entre eux utilisent des ontologies pour réaliser les tâches pour lesquelles ils sont conçus.
Nous intéressons, en particulier, au *sous-système de Recherche* (SSR), pour l'étude de fonctionnement de trois types d'agents : agent de recherche ainsi qu'un agent superviseur et un agent ressource.

Le système AGATHE est déployé dans l'environnement ECLIPSE en utilisant la plate-forme multi-agents Jade (***Jade,2006***). Les agents informationnels sont développés avec le moteur d'inférences Jess (***Jess, 2006***). Le développent multi-agents respecte les recommandations de la FIPA (***FIPA, 2000***), notamment le langage de communication retenu est ACL-FIPA. Pour la saisie et la manipulation d'ontologies, l'environnement Protégé (***Protégé, 2006***) a été retenu et l'exploitation des ontologies par les agents informationnels est faite via le composant JessTab (***Eriksson, 2003***).

Le sous-système de recherche (SSR)

Comme illustré par la figure 2.13, le sous-système de recherche (SSR) reçoit des requêtes provenant des clusters d'extraction (CE) du sous système d'extraction (SSE). Ces requêtes permettent, via des moteurs de recherche, de récupérer des pages Web en HTML. Par exemple, un agent d'extraction "article" du cluster d'extraction "science", effectue une requête afin d'acquérir des pages contenant les termes tel que "introduction", "related work", "conclusion". Le SSR transfert cette requête aux différents moteurs de recherche et rassemble les pages Web obtenus.

Ces dernières sont retournées vers le cluster approprié qui s'occupe ensuite de les transmettre aux agents demandeur.

Trois types d'agent contribuent à la récupération des informations :

- les *agents de recherche* ciblant le Web dans son ensemble au travers des moteurs de recherche traditionnels (Google, Yahoo, Altavista, …),
- (ii) les *agents ressources* effectuant des recherches à travers des moteurs de recherche spécifiques comme DBLP et CITESEER,
- et enfin (iii) *l'agent superviseur* qui supervise les deux premiers types d'agents. Chacun de ces agents est décrit en détail dans les sous-sections suivantes.

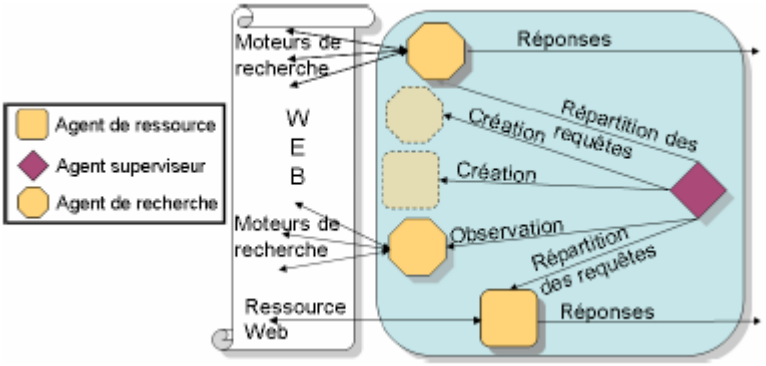

Fig 2.13 : L'architecture interne du Sous Système de Recherche SSR.

L'agent de recherche

L'agent de recherche transmet aux moteurs de recherche traditionnels les requêtes qui lui proviennent des clusters d'extraction du SSE. Après avoir fusionné les résultats obtenus par les différents moteurs sollicités, cet agent retourne les résultats aux clusters d'extraction concernés du SSE.

L'agent de ressource

L'agent de ressource est relativement similaire à l'agent de recherche si ce n'est qu'il transmet les requêtes issues du SSE vers des ressources spécialisées du Web. Par exemple, pour des requêtes concernant la recherche académique de telles ressources peuvent être des sites comme CITESEER dans le cas de publications, DBLP dans le cas de recherche à propos d'auteurs où même des services Web spécifiques.

L'agent superviseur

L'agent superviseur assure la supervision du SSR. Plusieurs rôles lui sont dévolus. Tout d'abord, il est là pour recevoir les requêtes en provenance SSE. En fonction de la charge de chacun des agents de recherche et de ressources, il va gérer l'allocation des requêtes. Ensuite, il est capable en fonction des besoins et de la charge des agents de recherche et de ressource, d'en créer ou d'en supprimer. Enfin, pour des raisons de performance, il est aussi capable de réaliser la migration d'agents vers d'autres unités de calcul moins surchargée. L'agent superviseur gère aussi des inscriptions donnant ainsi la possibilité au SSE de recevoir périodiquement les résultats d'une requête spécifique sans avoir à en reformuler la demande. Cette méthode est appelée "push".

Plusieurs familles d'algorithmes multi-agents ont vu le jour. La plus grande source d'inspiration reste cependant l'observation de la nature. Chacune de ces méthodes essaye ainsi de tirer parti d'un comportement observé dans les groupes sociaux. Les algorithmes biomimétiques s'inspirant du comportement social observé chez les fourmis en sont un bon exemple [107].

2.2.3 RI à base de colonie de fourmis

Les algorithmes à base de fourmis artificielles s'inspirent du comportement collectif des fourmis pour résoudre des problèmes d'optimisation combinatoire ou de classification dans la recherche d'information. On peut considérer que les fourmis sont des agents qui agissent sur un environnement donné. Par exemple, dans une colonie de fourmis, une ouvrière est autonome et aide la communauté en rapportant de la nourriture. Elle communique avec d'autres ouvrières ne trouvant pas de nourriture en laissant une trace de *phéromone* menant à la source de nourriture afin de transmettre son savoir. Enfin, elle est également capable d'émettre un signal d'alarme lorsqu'elle rencontre une fourmi d'une autre colonie pour alerter d'autres fourmis spécialisées dans le combat afin de protéger la colonie [107].

Les fourmis sont notamment une parfaite illustration : individuellement, une fourmi n'est pas considérée comme une créature très intelligente mais prise dans sa globalité, une colonie de fourmis est capable d'accomplir des tâches complexes et faire preuve d'une intelligence étonnante. Elles sont capables d'identifier rapidement le chemin le plus court menant à la nourriture ou encore de s'entraider spontanément afin de pouvoir traîner un très gros morceau de nourriture qu'une seule fourmi est incapable de transporter. Les fourmilières sont d'ailleurs très bien organisées et sont des constructions assez élaborées qui peuvent comporter plusieurs millions d'individus. La plupart des interactions entre les individus d'une colonie de fourmis sont réglées par des substances chimiques : les *phéromones*. Ces éléments chimiques interviennent à plusieurs stades dans la vie de la communauté des fourmis. Grâce à elles, les fourmis peuvent retrouver leur chemin, ou encore émettre une alarme. Les phéromones permettent aussi de changer la spécialité d'une fourmi. Chacune possède un seuil de phéromone pour chaque spécialité et n'effectue une tâche donnée qu'à partir du moment où le taux de phéromones correspondant à cette tâche dépasse ce seuil. Ainsi, lorsque les larves ont besoin de fourmis nourricières pour les nourrir, elles émettent des phéromones qui vont attirer d'autres fourmis se spécialisant à leur tour dans la nourriture des larves.
Phéromones et mémoire L'utilisation de phéromone est cruciale pour les algorithmes de colonies de fourmis. Le choix de la méthode d'implémentation des pistes de phéromone est donc important pour obtenir les meilleurs résultats. Ce choix est en grande partie lié aux possibilités de représentation de l'espace de recherche, chaque représentation pouvant apporter une façon différente d'implémenter les pistes. En effet, les pistes de phéromone décrivent à chaque pas l'état de la recherche de la solution par le système, les agents modifient la façon dont le problème va être représenté et perçu par les autres agents. Cette information est partagée par le biais des modifications de l'environnement des fourmis, grâce à une forme de communication indirecte : la stigmergie.

L'information est donc stockée, un certain temps dans le système, ce qui a amené certains auteurs à considérer ce processus comme une forme de mémoire adaptative, où la dynamique de stockage et de partage de l'information va être cruciale pour le système.
***Représentation du problème* :** Le problème est représenté par un *jeu de solutions*, *une fonction objectif* assignant une valeur à chaque solution et *un jeu de contraintes*. L'objectif est de trouver l'optimum global de la fonction objectif satisfaisant les contraintes. Les différents états du problème sont caractérisés comme une séquence de composants.

Fig.2.14 graphe représentant la phéromone
{Dans un algorithme de colonie de fourmis, les pistes de phéromone peuvent être associées aux composants (a) ou aux connexions (b) du le problème à résoudre.}

On peut noter que, dans certains cas, un coût peut être associé à des états autres que des solutions.

Dans cette représentation, les fourmis construisent des solutions en se déplaçant sur un graphe *G = (C; L),* où les nœuds sont les composants de *C* et où l'ensemble L connecte les composants de *C*. Les contraintes du problème sont implémentées directement dans les règles de déplacement des fourmis (soit en empêchant les mouvements qui violent les contraintes, soit en pénalisant de telles solutions).

2.2.3.1 Recherche locale et méta-heuristiques

Du point de vue de la recherche locale, utiliser des algorithmes de colonies de fourmis pour engendrer une solution initiale est un avantage indéniable. Les méta-heuristiques de colonies de fourmis sont souvent plus efficaces quand elles sont hybridées avec des algorithmes de recherche locale. Ceux-ci optimisent les solutions trouvées par les fourmis, avant que celles-ci ne soient utilisées pour la mise à jour des pistes de phéromone.

Une autre possibilité pour améliorer les performances est d'injecter une information heuristique plus pertinente. Cet ajout a généralement un coût élevé en termes de calculs supplémentaires.

Il faut noter que ces deux approches sont similaires de par l'emploi qu'elles font des informations de coût pour améliorer une solution. La recherche locale le fait de façon plus directe que l'heuristique, cependant que cette dernière est peut être plus naturelle pour utiliser des informations a priori sur le problème.

Une description élégante a été proposée dans [108], Ce cas, bien que restrictif, permet de dégager les apports originaux de ces méta-heuristiques, Nous en avons traduit ci-dessous un extrait :

- Une méta-heuristique de colonie de fourmis est un processus stochastique construisant une solution, en ajoutant des composants aux solutions partielles. Ce processus prend en compte une heuristique sur l'instance du problème des pistes de phéromone changeant dynamiquement pour refléter l'expérience acquise par les agents.
- Une formalisation plus précise existe [108]. Elle passe par une représentation du problème, un comportement de base des fourmis et une organisation générale de la méta-heuristique. Plusieurs concepts sont également à mettre en valeur pour comprendre les principes de ces algorithmes, notamment la définition des pistes de

phéromone en tant que mémoire adaptative, la nécessité d'un réglage intensification/diversification et enfin l'utilisation d'une recherche locale.

Parallélisme D'une manière générale, les solutions de bonnes qualités émergentes du résultat des interactions indirectes ayant cours dans le système, pas d'un codage explicite d'échanges. En effet, chaque fourmi ne prend en compte que des informations locales de son environnement (les pistes de phéromone) ; il est donc facile de paralléliser un tel algorithme. Il est intéressant de noter que les différents processus en cours dans la méta-heuristique (i.e. le comportement des fourmis, l'évaporation et les processus annexes) peuvent également être implémentés de manière indépendante, l'utilisateur étant libre de décider de la manière dont ils vont interagir, par conséquent, La structure des méta-heuristiques de colonies de fourmis comporte un parallélisme intrinsèque.

Extensions Devant le succès rencontré par les algorithmes de colonies de fourmis, de nombreuses pistes autres que celle de l'optimisation combinatoire commencent à être explorées : par exemple, l'utilisation de ces algorithmes dans des problèmes continus et/ou dynamiques, ou encore la mise en relation de ce type d'algorithmes dans un cadre d'intelligence en essaim et avec d'autres méta-heuristiques.

Problèmes d'adaptation Les méta-heuristiques sont bien souvent élaborées pour des problèmes combinatoires, mais il existe une classe de problèmes souvent rencontrée en ingénierie, où la *fonction objectif* est à variables continues et pour lesquels les méta-heuristiques peuvent être d'un grand secours (fonction non dérivable, multiples minimums locaux, grand nombre de variables, non-convexité, etc.). Plusieurs tentatives pour adapter les méta-heuristiques de colonies de fourmis au domaine continu sont apparues.

Outre les problèmes classiques d'adaptation d'une méta-heuristique, les algorithmes de colonies de fourmis posent un certain nombre de problèmes spécifiques. Ainsi, le principal problème vient si l'on se place dans le formalisme ACO avec une construction de la solution composant par composant. En effet, un problème continu peut selon la perspective choisie présenter une infinité de composants, le problème de la construction est difficilement soluble dans ce cas. La plupart des algorithmes s'inspirent donc des caractéristiques d'auto-organisation et de mémoire externe des colonies de fourmis, laissant de coté la construction itérative de la solution ; cependant, depuis peu, le caractère probabiliste du formalisme ACO commence à être employé [**107**].

Comportement des fourmis : Les fourmis artificielles peuvent être caractérisées comme une procédure de construction stochastique construisant des solutions sur le graphe $G =(C; L)$. En général, les fourmis tentent d'élaborer des solutions faisables mais, si nécessaire, elles peuvent produire des solutions infaisables. Les composants et les connexions peuvent être associés à des pistes de phéromone (mettant en place une mémoire adaptative décrivant l'état du système) et à une valeur heuristique (représentant une information a priori sur le problème, ou venant d'une source autre que celle des fourmis ; c'est bien souvent le coût de l'état en cours). Les pistes de phéromone et la valeur de l'heuristique peuvent être associées soit aux composants, soit aux connexions (figure 2.14).

Dans les sections suivantes, nous examinerons les algorithmes ayant prouvé le potentiel et l'efficacité d'algorithmes s'inspirant du comportement des individus d'une colonie de fourmis. Ces exemples nous serviront de base pour l'élaboration du mode de fonctionnement

par déposition et absorbation de rôles de recherche dans notre approche proposée pour la structuration du modèle de comportement au sein de nos agents de recherche.

2.2.3.2 L'algorithme fondateur : ACO

La première démonstration de la capacité et de la performance des algorithmes à base de fourmis artificielles a été réalisée par Dorigo en 1992 au travers de l'heuristique Ant Colony Optimization (ACO) [109]. Cette méta-heuristique s'inspire du comportement de fourragement observé chez les fourmis réelles en tenant compte des interactions physiologiques entre les fourmis. Ce comportement est à la base de la coopération et la communication menant à la résolution de problèmes classiques comme le problème du plus court chemin ou le problème du voyageur de commerce.

Le PVC, ou Traveling Salesman Problem se définit de la manière suivante :
Un voyageur de commerce doit visiter un ensemble $\{v1, \ldots, vn\}$ de n villes dont on connaît les distances respectives $d(vi,vj),\forall(i,j)\in\{1,\ldots,n\}^2$. Le problème consiste à trouver la permutation de $\sigma\{v1, \ldots, vn\}$ telle que la séquence $S=(v_{\sigma(1)},\ldots,v_{\sigma(n)})$ minimise la distance totale $D(\sigma)$ parcourue par le voyageur :

$$D(\sigma)=\sum_{i=1}^{n-1} d(v_{\sigma(i)},v_{\sigma(i+1)}+d(v_{\sigma(n)},v_{\sigma(1)}) \qquad (1.13)$$

En effet, l'espace de recherche correspond à l'ensemble des combinaisons possibles des n villes, soit n! Combinaisons. Ce problème peut également être considéré comme la recherche d'un circuit hamiltonien de longueur minimale dans un graphe complet pouvant être anti-symétrique dans le cas général ($\forall$ (i, j) tel que $d(vi, vj) \neq d(vj , vi)$)

Dans [110], les auteurs montrent que l'organisation même de la colonie et les interactions opérant entre les fourmis amènent à la résolution de ce type de problème. En déposant des phéromones sur les chemins qu'elles empruntent, les fourmis établissent un moyen efficace de résolution du problème du plus court chemin. En effet, lorsque deux chemins de longueurs différentes s'offrent à une colonie de fourmis pour aller du nid à une source de nourriture, les fourmis choisissent initialement avec une probabilité uniforme le chemin à emprunter. Par la suite, les fourmis déposant toutes des phéromones, le plus court des deux chemins va avoir tendance à être davantage marqué incitant les fourmis suivantes à suivre ce même chemin. Ce comportement autoentretenu entraîne la majorité des fourmis à suivre le plus court chemin.

Pour résoudre le PVC, l'algorithme AS étend ces principes en permettant aux fourmis artificielles de sauvegarder quelques données en mémoire et de percevoir une partie de leur environnement. Ces nouvelles facultés vont permettre d'adapter le comportement des fourmis artificielles au problème : chaque fourmi doit se souvenir des villes dans lesquelles elle est déjà passée afin de ne pas y retourner. Dans cette optique, une liste des villes à ne pas visiter est maintenue au sein de la mémoire de la fourmi et réinitialisée une fois qu'un parcours complet a été effectué (i.e. toutes les villes ont été visitées).

(1) **tant que** Condition de terminaison non rencontrée faire
(2) Construction de solutions par les fourmis
(3) Mise à jour des phéromones
(4) Exécuter des actions sur l'environnement
(5) **fin tant que**

Fig.2.15 Cadre d'exécution de l'algorithme ACO

La modélisation utilisée considère le graphe orienté *G (A, V)* représentant l'ensemble des villes considérées dans le problème (les sommets *V*) et des chemins les reliant (les arcs *A*). Les fourmis sont disposées sur les sommets du graphe et peuvent voyager de sommet en sommet en suivant les arcs. À chaque voyage d'une ville i à une ville j, elles déposent une trace de phéromone sur l'arête *(i, j)*. Le choix d'une nouvelle ville à parcourir se fait alors en fonction des villes précédemment visitées et selon une probabilité dépendant de la quantité de phéromone présente sur l'arête ainsi que de la distance entre cette ville et la position courante de la fourmi. Une fois que toutes les fourmis ont effectué un parcours complet, le taux de phéromone présent sur chaque arête est mis à jour. Pour cela, le taux présent à l'itération précédente est diminué proportionnellement à une constante d'évaporation définie à l'origine puis renforcée en fonction du parcours des fourmis sur l'arête prise en compte.

L'algorithme se termine au bout d'un temps défini et la solution retenue correspond au chemin de moindre coût - le plus court - parcouru par les fourmis au cours de l'exécution de l'algorithme.

Plusieurs extensions à l'algorithme ACO ont vu le jour afin d'améliorer la répartition des phéromones sur le graphe ainsi que d'ajuster plus finement la probabilité de choix d'une ville par les fourmis. Nous ne détaillerons pas ces mises à jour dans ce document mais nous pouvons cependant citer les travaux suivants : [111] introduit un système d'apprentissage par renforcement, [112] propose le système MAX −MIN AS introduisant des seuils minimum et maximum à la trace de phéromone et [113] décrit l'algorithme Ant Colony System (ACS). Ce dernier introduit principalement plus de choix probabiliste dans la décision d'exploration d'une ville et correspond à l'heuristique à base de fourmis artificielles apportant les meilleurs résultats pour le PVC.

ACO rassemble l'ensemble de ces algorithmes et peut, d'une manière plus générale, être décrit par l'algorithme dans la figure 2.15. Les trois opérations peuvent s'exécuter de manière non synchronisée et leur ordonnancement est laissé à la discrétion du programmeur.

La première opération listée consiste à construire une solution au problème grâce à une fourmi en la déplaçant sur les nœuds du graphe représentatif du problème. Ce mouvement est soumis à des décisions locales probabilistes qui tiennent compte à la fois de la connectivité du graphe et de l'intensité des traces de phéromones déposées par les fourmis dans les précédentes itérations. Au fur et à mesure de son déplacement, la fourmi mémorise la solution qu'elle est en train de construire en termes de chemin parcouru sur le graphe.

La deuxième opération permet d'annoter les arcs du graphe représentatif du problème pour aider les autres fourmis à trouver la meilleure solution. La mise à jour des phéromones peut se faire suivant deux méthodes distinctes : à chaque pas effectué par une fourmi d'une part - on parle alors de mise à jour pas à pas - et une fois la solution construite en suivant le chemin inverse sauvegardé dans la mémoire de la fourmi d'autre part - on parle alors de mise à jour retardée. Il est également important de procéder à l'évaporation de l'ensemble des phéromones déposées sur les arcs du graphe. En effet, l'intensité des traces de phéromones doit décroître au cours du temps afin d'éviter une convergence trop rapide de l'algorithme dans un extremum local. Cette évaporation permet à la colonie de fourmis d'oublier peu à peu ce qu'elle a appris ce qui favorise de plus une meilleure adaptation à tout changement dans l'environnement.

Enfin, il peut être nécessaire d'exécuter certaines actions centralisées qui ne peuvent être effectuées par les fourmis elles-mêmes de manière individuelle. Il peut être par exemple nécessaire de collecter l'ensemble des solutions établies par les fourmis afin de décider s'il est judicieux d'ajouter des phéromones supplémentaires pour aider le processus de recherche en soulignant davantage le chemin menant à la construction de la meilleure solution actuelle. Cette mise à jour de phéromone est appelée mise à jour en différé et permet dans certains cas d'augmenter les performances globales d'un algorithme.

L'algorithme ACO est à la base d'un grand nombre d'applications s'inspirant du comportement collectif des fourmis afin de résoudre un problème donné. Celui-ci s'appuie sur un modèle du comportement de certaines espèces de fourmis. Cependant, toutes les colonies de fourmis n'utilisent pas les mêmes armes et l'évolution des espèces a permis de tester plusieurs stratégies permettant la survie de chaque espèce. Il existe ainsi tout naturellement d'autres modèles algorithmiques utilisés en optimisation et s'inspirant d'espèce de fourmis différentes aux caractéristiques distinctes de celles reprises dans ACO.

Nous présentons dans la section suivante l'algorithme API permettant de ne pas avoir à gérer un système à base de phéromones qui peut se révéler très lourd à maintenir lorsque l'on traite des graphes de très grande dimension comme c'est le cas dans une représentation du Web ubiquitaire.

2.2.3.3 L'algorithme API

Chaque fourmi s'attribue des sites particuliers qu'elle va explorer aléatoirement en mémorisant toutefois préférentiellement ceux sur lesquels elle a pu rencontrer un certain succès. Au fur et à mesure de leur sortie, les ouvrières chargées de récolter de la nourriture s'éloignent de plus en plus du nid couvrant petit à petit une grande partie de leur espace de recherche. Plus généralement, trois règles décrivent le comportement de ces fourmis :

- La découverte d'une proie entraîne toujours le retour sur le même site de chasse lors de la sortie suivante,
- La découverte d'une proie pèse sur la décision de sortie des fourrageuses en réduisant l'intervalle de temps passé au nid,
- Les fourrageuses semblent apprendre progressivement une association entre une direction de recherche opposée au nid et l'augmentation de la probabilité de succès.

Pour modéliser algorithmiquement ce comportement, deux opérateurs ont été définis afin de déterminer le déplacement et l'exploration des fourmis dans l'espace de recherche S :

- l'opérateur O*rand* qui génère un point de S de manière aléatoire et uniforme,
- l'opérateur O*explo*(s, A) qui génère un point s′ dans le voisinage d'un point s.

Ce dernier opérateur admet deux paramètres : le point de départ de l'exploration locale et l'amplitude maximale du voisinage à prendre en compte. Cet opérateur peut effectuer une simple exploration aléatoire ou être personnalisé avec une heuristique inspirée par le domaine de la recherche.

Pour adapter le comportement décrit ci-dessus, il est nécessaire d'examiner deux cas représentés sur la figure 2.16. D'un point de vue global, les fourmis explorent aléatoirement le voisinage d'un point central de l'espace de recherche représentant le nid des fourmis grâce à l'opérateur O*rand*. Elles cherchent des points de bonne qualité – maximisant la fonction d'évaluation - en mémorisant des sites de chasse localisés dans une amplitude A_{site} déterminée autour du nid N à l'aide de l'opérateur O*explo*(N,A*site*).

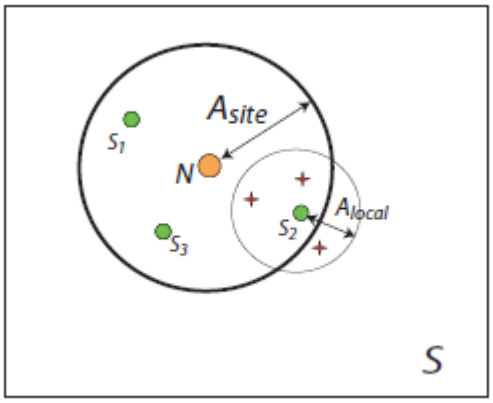

Fig 2.16 Exploration locale de la fourmi autour de son nid de rattachement.
Trois sites de chasses si sont identifiés autour du nid N.
La fourmi effectue une exploration locale depuis le site s2.

Au bout d'un temps donné, le nid est déplacé à un autre endroit et les fourmis oublient les sites de chasse mémorisés. D'un point de vue local, chaque fourmi se déplace autour du nid et mémorise p sites de chasse indépendamment des autres fourmis. Le voisinage de chaque site est exploré de manière aléatoire sur une zone délimitée par une certaine amplitude A*local* (avec généralement A*local* < A*site*) en utilisant l'opérateur O*explo*(si,A*local*) où si représente le point de départ de l'exploration. La capture d'une proie est représentée par une amélioration locale de la fonction d'optimisation. Lorsqu'une exploration mène à la capture d'une proie, la fourmi va systématiquement visiter le voisinage de ce site à la prochaine exploration. Et de manière symétrique, si un site se révèle peu fructueux, il est oublié et remplacé au sein de la mémoire par un nouveau site de chasse choisi aléatoirement dans le voisinage du nid.

L'automate présenté sur la figure 2.17 illustre le comportement individuel d'une fourmi pendant une recherche de nourriture. Comme indiqué dans le paragraphe précédent, une fourmi se crée initialement p sites de chasse en mémoire et décide d'en explorer un. À partir de ce site, la fourmi cherche des proies dans une amplitude A*local* autour du site de chasse *sj* . À chaque exploration, si une proie n'est pas trouvée - i.e. si la fonction d'optimisation n'est pas améliorée -, un compteur d'échec *ej* est incrémenté et est réinitialisé dans le cas contraire.

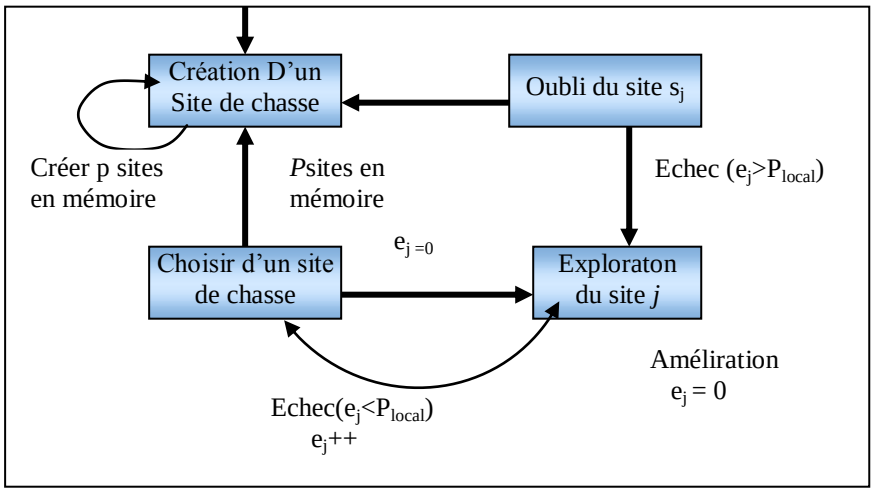

Fig 2.17 Automate représentant le comportement individuel de fourragement d'une fourmi.

On considère que chaque fourmi peut effectuer un certain nombre de tentatives de chasses infructueuses avant d'abandonner son site de chasse. Ainsi, elles possèdent un paramètre P*local* indiquant le nombre maximum d'échecs avant l'oubli du site de chasse. À chaque itération de l'algorithme, toutes les fourmis du nid sont simulées en parallèle de manière indépendante. Ainsi, cet algorithme peut facilement être implémenté sur un système multiprocesseur afin d'établir une indépendance supplémentaire entre chaque individu.

Quelques extensions à cet algorithme ont été proposées dans [55]. Par exemple, une fourmi pourrait recruter un autre individu de la colonie si celui-ci explore une zone très fructueuse, augmentant ainsi les chances d'obtenir rapidement l'optimum recherché. Ou encore les paramètres pilotant le comportement des fourmis pourraient être fixés indépendamment pour chaque individu.

Les algorithmes inspirés par les colonies de fourmis sont des algorithmes à base d'agents particuliers. Cependant, il n'existe pas à notre connaissance d'outil de recherche d'information basé sur le comportement des fourmis. Nous nous inspirerons très fortement de cet algorithme afin de concevoir un modèle algorithmique relativement simple par assemblage de rôles de recherche, tandis qu'il se trouve une base de rôle déployée à chaque serveur d'information donnant lieu de l'évolution de cette base de rôle par le passage d'autre agent de recherche donne notre cas.

Dans la suite, nous allons présenter des systèmes de recherche d'information conçus à partir d'algorithmes génétiques. Ces algorithmes ont montré leur grande adaptabilité dans beaucoup de domaines de recherche et il est donc naturel de penser à eux pour étudier les différentes techniques usuelles de la recherche d'information dans le Web.

2.2.4 Algorithmes génétiques

La famille des approches évolutionnaires englobe l'ensemble des méthodes basées sur des concepts inspirés de l'évolution naturelle des espèces vivantes [115]. Bien qu'à l'origine, ces approches ne fussent pas nécessairement destinées à la résolution de problèmes d'optimisation [116], elles sont maintenant fréquemment utilisées pour cette tâche, et constituent des alternatives très performantes aux approches de recherche d'informations de grands volumes.

Selon les grands principes de l'évolution des espèces et de la sélection naturelle [117], l'évolution d'une population est caractérisée par le degré d'adaptation des individus à leur environnement : en favorisant la survie et la reproduction des individus les mieux adaptés à leur milieu, la nature assure la pérennité des meilleures caractéristiques de la population et permet ainsi, par la recombinaison de ces caractéristiques entre elles, de faire évoluer l'espèce en formant des nouveaux individus qui tendent, au fil des générations, à être toujours mieux adaptés à l'environnement qui les entoure (puisqu'héritant des bonnes caractéristiques de leurs deux parents). Considérant la qualité d'une solution (selon le critère à optimiser) comme un degré d'adaptation au milieu, les approches évolutionnaires s'appuient sur ces principes pour obtenir des solutions proches de l'optimum du problème. Typiquement, un algorithme évolutionnaire est alors composé de trois éléments essentiels :

- Une population constituée d'individus représentant des solutions potentielles (configurations) du problème à optimiser ;
- Une fonction d'évaluation de l'adaptation (fonction de fitness) des individus à leur environnement (objectif du problème) ;

- Un mécanisme d'évolution permettant la reproduction des individus et l'application du phénomène de sélection naturelle.

D'un point de vue opérationnel, un algorithme évolutionnaire typique fait évoluer une population en suivant un cycle composé de 3 étapes séquentielles :

1. Mesure de la qualité des individus de la population ;
2. Sélection d'une partie de la population ;
3. Production et/ou mutations d'individus ;

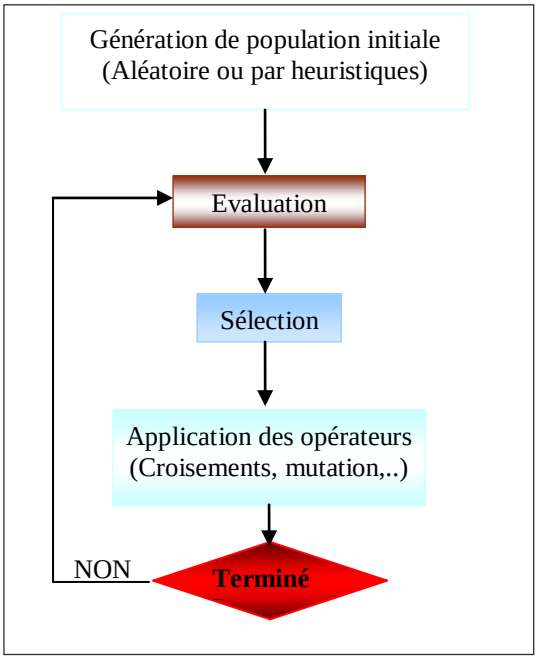

Fig 2.18 Organigramme d'un algorithme génétique.

À la fin des années 80, Goldberg a contribué à populariser les AG en définissant l'AG canonique où les individus sont décrits par l gènes binaires et sont évalués numériquement par une fonction d'évaluation *f* [118]. Il se décompose en trois étapes principales décrites ci-dessous.

1. **Générer** aléatoirement et uniformément une population initiale *P(0)* de n individus, $t \leftarrow 0$,
2. **Sélectionner** *n* individus dans *P(t + 1)* avec remise suivant une distribution de probabilité proportionnelle à l'évaluation des individus par *f* et leur appliquer les opérateurs de croisement et de mutation avec des probabilités respectives de p_{cross} et p_{mut}.
3. **Évaluer** les individus de *P(t + 1)*, $t \leftarrow t + 1$,

Recommencer le processus de sélection ou arrêter.

Cet algorithme se veut valide pour tout domaine étudié tant qu'il est possible de modéliser les individus de la population par un codage binaire. Cependant, le codage binaire n'est pas toujours bien adapté à la résolution de problèmes par un AG notamment lorsqu'on traite des espaces de recherche assez grands [119].

Sous réserve que le problème soit bien codé et que les opérateurs utilisés soient bien adaptés, la qualité des individus de la population devrait alors tendre à s'améliorer au fur et à mesure de ce processus d'évolution. Les différentes approches évolutionnaires peuvent se classer en 3 grandes catégories :

- 1) Les stratégies d'évolution [120] qui font évoluer une population en utilisant des opérateurs de sélection et de mutation déterministes ;
- 2) La programmation évolutionnaire [121] qui ne réalise pas de sélection mais fait évoluer la population en appliquant des mutations successives aux individus
- 3) Les algorithmes génétiques [122] qui, contrairement aux deux autres types d'approches, utilisent un opérateur de croisement des individus.

Ces trois types d'approches se différencient surtout par leur manière de représenter les données et de faire évoluer les populations de solutions qu'elles manipulent. Nous nous limitons ici à la présentation des algorithmes génétiques, qui sont les algorithmes évolutionnaires les plus répandus.

Les algorithmes génétiques classiques, initialement introduits par Holland [122], s'appuient sur un codage de l'information qu'ils manipulent ainsi que sur un ensemble d'opérateurs génétiques permettant de produire des configurations du problème à partir de configurations parentes. Le codage de l'information, qui doit être défini de manière adaptée pour permettre des recombinaisons génétiques simples et efficaces, peut se faire de multiples façons. Le codage le plus courant utilise des chaînes.

2.2.4.1 Phénomène de croisement

Le phénomène de croisement est une propriété naturelle de l'ADN. C'est par analogie qu'ont été conçus les opérateurs de croisement dans les AG.

Cette famille d'opérateurs a pour but de répartir aléatoirement les individus sélectionnés en couples afin d'engendrer une nouvelle génération. Cette génération est obtenue en copiant et recombinant les deux chromosomes parents de manière à générer deux chromosomes enfants possédant des caractéristiques issues des deux parents.

L'opérateur de croisement favorise l'exploitation de l'espace de recherche en examinant de nouvelles solutions à partir de deux solutions actuelles. En effet, cet opérateur assure le brassage du matériel génétique de la population et l'accumulation des gènes favorables en les multipliant. Ce brassage permet alors de créer de nouvelles combinaisons de gènes qui peuvent se révéler potentiellement favorables.

Le croisement à un point est l'opérateur de croisement historique des AG. Il consiste à choisir au hasard un point de croisement pour chaque couple d'individus sélectionné. Ce point divise le génome en deux sous-chaînes. On échange ensuite les deux sous chaînes terminales de chacun des deux chromosomes, ce qui produit deux enfants. La figure 2.6 illustre ce fonctionnement.

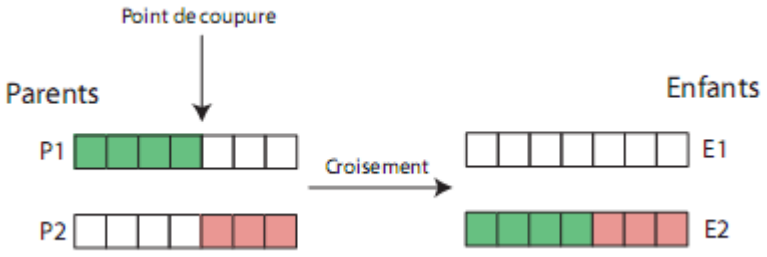

Fig. 2.19 Opérateur de croisement à 1 point.

L'opérateur de mutation consiste à modifier un gène sélectionné dans un chromosome et à le remplacer par une valeur aléatoire. La probabilité de mutation d'un gène est généralement très faible dans un environnement stable afin de favoriser la capitalisation des gènes favorables tout en permettant d'élargir l'espace des solutions évaluées. Cependant, dans un environnement dynamique subissant une évolution rapide, il peut être judicieux d'employer un taux de mutation relativement élevé permettant une adaptation rapide de la population. Le taux de mutation est ainsi à adapter en fonction de l'application et du type de mutation employé.

Les AG se prêtent facilement à la parallélisation du fait de leur conception. Nous allons aborder dans la section suivante les différentes méthodes de parallélisation des AG présentes dans la littérature. À ce sujet, on peut consulter deux états de l'art des techniques existantes de parallélisation dans les méta-heuristiques [107].

2.2.4.2 Parallélisation des algorithmes génétiques

Le développement des réseaux et des machines parallèles a amené plusieurs auteurs à s'intéresser à la parallélisation des AG. L'intérêt de cette évolution réside principale ment dans le gain en temps de calcul, mais également dans la définition de nouveaux algorithmes. De nombreux modèles ont été proposés dans la littérature depuis le début des recherches sur les AG mais on recense principalement deux types de parallélisation.

La première concerne la parallélisation de l'évaluation des individus. Elle a été introduite pour accélérer les traitements des calculs qui peuvent s'avérer coûteux dans une méta-heuristique, sans essayer d'explorer d'avantage l'espace de recherche. Généralement cette méthode correspond à une méthode séquentielle dont les traitements ont été accélérés.

Dans l'autre cas, la recherche est divisée en plusieurs entités avec différents niveaux de synchronisation et de coopération. Le but de cette méthode est d'explorer plus activement l'espace de recherche en initialisant l'algorithme sur plusieurs points de cet espace. Les recherches s'effectuent ensuite simultanément à partir de ces points initiaux [107].

2.2.4.2.1 Parallélisme à gros grains (coarse grain)

Dans cette approche, on utilise généralement le même AG pour chaque sous-population d'individus mais des recherches ont été menées (notamment dans [123]) afin d'étudier différentes stratégies d'exploration.

Le parallélisme à gros grain ajoute des opérateurs et des règles gérant la communication entre les différentes sous-populations. Un opérateur de migration est ajouté afin de permettre l'échange d'informations entre les sous-populations. Une politique d'émigration détermine quelle stratégie (hasard, meilleur individu, etc) employer pour sélectionner les individus candidats à la migration. Une politique d'immigration gère le remplacement d'individus dans la population d'accueil à l'arrivée d'un nouvel élément. D'autres paramètres gèrent la fréquence d'échange ou encore le nombre de départs simultanés d'une population.

Les études menées par Tanese [124] et Belding [125] ont montré d'une part que sans la connaissance des meilleures combinaisons de paramètres possibles, des méthodes parallèles utilisant différentes valeurs de paramètres permettent d'obtenir de meilleurs résultats. D'autre part, ils ont montré qu'un fort taux de migration diminuait sensiblement le temps nécessaire à la résolution d'un problème.

Le modèle Island (parallélisme par îlots), initié par Cohoon a permis d'étudier le rôle de la migration au niveau de l'évolution de la qualité de la population [126].

Dans ce modèle les individus peuvent migrer dans n'importe quelle autre population. Il a été observé que les nouvelles solutions apparaissent majoritairement rapidement après l'introduction de nouveaux individus dans une population. Les AG parallèles par îlots ne sont pas nécessairement équivalents aux AG séquentiels décrits précédemment car l'application des opérateurs de croisement et de mutation est localisée dans un voisinage donné de la population totale. Mühlenbein introduit en 1991 un nouveau modèle, le stepping-stone, dans lequel les échanges ne peuvent se faire qu'entre populations d'individus voisines dans l'espace de recherche [127]. Son originalité vient de l'utilisation d'une heuristique de type hill-climbing lorsque la qualité d'une population ne s'améliore pas au bout d'un certain nombre d'itérations. Cette méthode a montré sa grande efficacité et a beaucoup contribué au développement des AG parallèles.

Dans un autre registre n'utilisant pas de migration entre les sous-populations, Whitley a constaté qu'en adaptant le paramètre de mutation en fonction de ceux obtenus par la sous-population obtenant les meilleurs résultats, les performances s'améliorent sensiblement [128].

2.2.4.2.2 Parallélisme à grains fins (fine grain)

Ce sont des méthodes asynchrones qui divisent la population en un nombre conséquent de sous-éléments. Ces sous-éléments peuvent aller du cas limite (décrit dans [129]) pour lequel un seul individu est représenté par un processeur. Chaque entité est connectée à plusieurs autres entités dans son voisinage formant un groupe et les opérateurs génétiques sont appliqués à ce groupe grâce à des échanges entre les individus le composant.

Ces méthodes sont souvent identifiées comme des méthodes massivement parallèles car chaque individu est représenté idéalement par un seul processeur. Whitley qualifie même ce type d'algorithme dans [130] de modèle cellulaire car on retrouve un parallèle avec les automates cellulaires avec des règles de réécritures probabilistes et un alphabet composé des éléments de l'espace de recherche.

La parallélisation est extrêmement efficace pour accélérer les temps de résolution des AG. Mais il convient d'étudier le type de problème rencontré afin de bien sélectionner la méthode à utiliser. Le ratio entre le temps mis pour l'évaluation d'un individu et l'exécution d'un opérateur génétique est un bon indicateur permettant d'effectuer un choix judicieux. Dans le cas d'un temps d'évaluation important, la parallélisation des calculs est préférable à une parallélisation par îlots, et réciproquement [107].

De très nombreux algorithmes génétiques ont été proposés pour la résolution de problèmes d'optimisation. Il est communément admis que, pour être réellement efficace, un algorithme doit être adapté au problème considéré. Ainsi, les différents opérateurs peuvent être définis différemment selon les applications. Le fonctionnement général décrit dans la figure 2.18 peut lui aussi être remis en question, notamment pour ce qui est de la manière dont on manipule les différentes populations (par exemple le mode de remplacement de la population courante). Un aperçu des algorithmes génétiques et de leur mise en œuvre pour la résolution de problèmes d'optimisation est donné dans [131]. Le problème crucial du paramétrage des algorithmes génétiques, notamment pour le choix des probabilités appliquées aux différents opérateurs génétiques, est exploré en détail dans [132].

2.2.5 Approches génétiques et à base d'agents pour la RI sur le Web

Quelques auteurs se sont penchés sur la résolution de problèmes liés au Web et plus particulièrement au problème de recherche d'information par algorithmes biomimétiques. On peut citer [133] utilisant l'apprentissage grâce à un AG, [134] utilisant un AG interactif afin de générer des styles de sites Web ou encore [135] dont l'AG gère un cache permettant d'accélérer l'accès à des documents provenant de sources diverses.

Dans la section suivantes, nous allons détailler une approche génétique « Webnaut » et une approche multi-agents apportant des solutions intéressantes pour la RI sur le Web.

Il existe plusieurs types de systèmes de recherche d'information sur Web. Mais généralement, seuls les méta-moteurs de recherche utilisent des algorithmes évolutionnaires afin d'améliorer les résultats produits par une méthode plus classique à base d'index inversé. Les méta-moteurs ont en effet de plus grandes ressources matérielles à disposition pour effectuer une recherche parce qu'ils se présentent généralement comme une application mono-utilisateur et ne doivent par conséquent pas répondre à des milliers d'utilisateurs en parallèle. Sur ce principe, des agents intelligents ont vu le jour, agissant comme des assistants personnels de recherche en étant capables de s'adapter à l'utilisateur les manipulant.

2.2.5.1 Webnaut

Webnaut, décrit dans [136], est un bon représentant de ce type d'agents personnels. Il combine un méta-moteur interrogeant les moteurs de recherche classiques (Google, . . .) avec un algorithme génétique permettant de générer de nouvelles requêtes afin de trouver de meilleurs résultats, le tout étant contrôlé par l'utilisateur au travers d'un système de relevance feedback. Cinq étapes se succèdent ainsi afin d'obtenir les meilleurs résultats possibles :

- Collecter des données sur les préférences de l'utilisateur (pages de résultats visitées du méta-moteur),
- Extraire des informations de ces données pour créer un profil de recherche (mots les plus représentatifs des documents),
- Découvrir de nouveaux documents en accord avec le profil créé précédemment,
- Évaluer et ne garder que les meilleurs documents découverts,
- Demander à l'utilisateur d'indiquer les meilleurs résultats retournés.

La première étape consiste donc classiquement à demander à l'utilisateur de poser une question au système de recherche à l'aide de mots-clés. Le méta-moteur interroge plusieurs moteurs de recherche et en extrait les résultats, élimine les doublons, et les présente à l'utilisateur sous forme de résumés. Ce dernier peut alors visiter les pages lui paraissant convenir le mieux à sa question. Pendant ce temps, le système enregistre les pages visitées et les propose en entrée du système génétique afin de créer de nouvelles requêtes. Le système génétique se compose de deux populations d'individus distincts représentés par des vecteurs de dimensions égales : une composée de termes et une composée d'éléments de la logique booléenne (et, ou, . . .). Chacune est gérée et modifiée par deux algorithmes génétiques distincts l'un de l'autre. Les requêtes booléennes sont alors générées en combinant un individu de chaque population comme décrit ci-dessous.

Pour cela, les documents pris en compte sont représentés sous forme vectorielle -décrite dans la section 2.2.1.1.3 - dans laquelle les poids des mots sont calculés à partir de leur fréquence d'apparition et de la fréquence du mot le plus représenté dans le document. La

méthode classique *tf*idf* n'est pas utilisée ici parce qu'elle nécessite la prise en compte de l'ensemble des documents du corpus utilisé. Les individus de la première population de l'algorithme génétique sont alors représentés par une sélection aléatoire d'un nombre défini de termes, choisis parmi les mots les plus fréquents des documents concernés. Tandis que les individus de la seconde population sont générés aléatoirement à partir d'éléments de la logique booléenne.

Deux opérateurs génétiques sont utilisés dans chaque AG : un opérateur de croisement à un point de coupure agissant sur deux individus, et un opérateur de mutation. Cet opérateur consiste à choisir aléatoirement un élément d'un vecteur représentant un individu et à le remplacer par un autre, tiré au hasard dans le cas de la population de vecteurs de booléens, et issu des documents initiaux dans l'autre population.

Les requêtes sont alors générées en associant éléments à éléments les termes aux booléens. Elles sont ensuite triées comme indiqué sur l'illustration 2.7, puis évaluées. Cette évaluation consiste à interroger le méta-moteur grâce aux requêtes générées et à mesurer la similarité - par la mesure *cosine* - des nouveaux documents retournés avec ceux ayant servi initialement au système génétique.

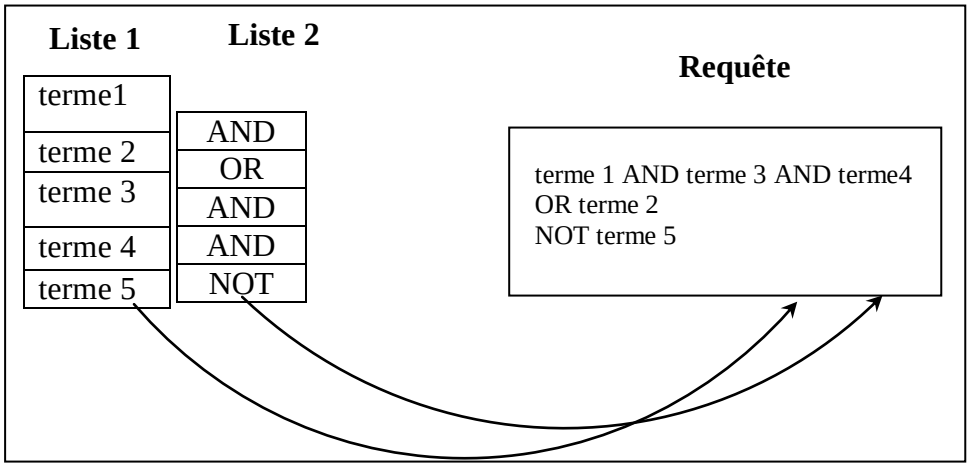

Fig 2.20 Génération d'une requête à partir de deux populations d'individus (une de termes et une d'éléments de la logique booléenne) dans le système Webnaut.

Les résultats produits par la meilleure requête sont alors présentés à l'utilisateur afin que celui-ci sélectionne les meilleures pages. Le processus peut alors recommencer dans le but d'améliorer encore les résultats obtenus.

2.3.4.2 InfoSpiders

Menczer a conçu le modèle InfoSpiders décrit dans [137] .Ce système est composé de plusieurs agents évoluant de leur naissance à leur mort et chargés de retrouver en permanence les documents correspondant à une requête donnée. Un tel système, au contraire d'une approche classique à base d'index, a l'avantage de prendre en compte naturellement le caractère de plus en plus dynamique des documents du réseau évoluant de jour en jour. Le modèle InfoSpiders présente un algorithme à base d'agents capables d'évoluer grâce à un système d'apprentissage par renforcement.

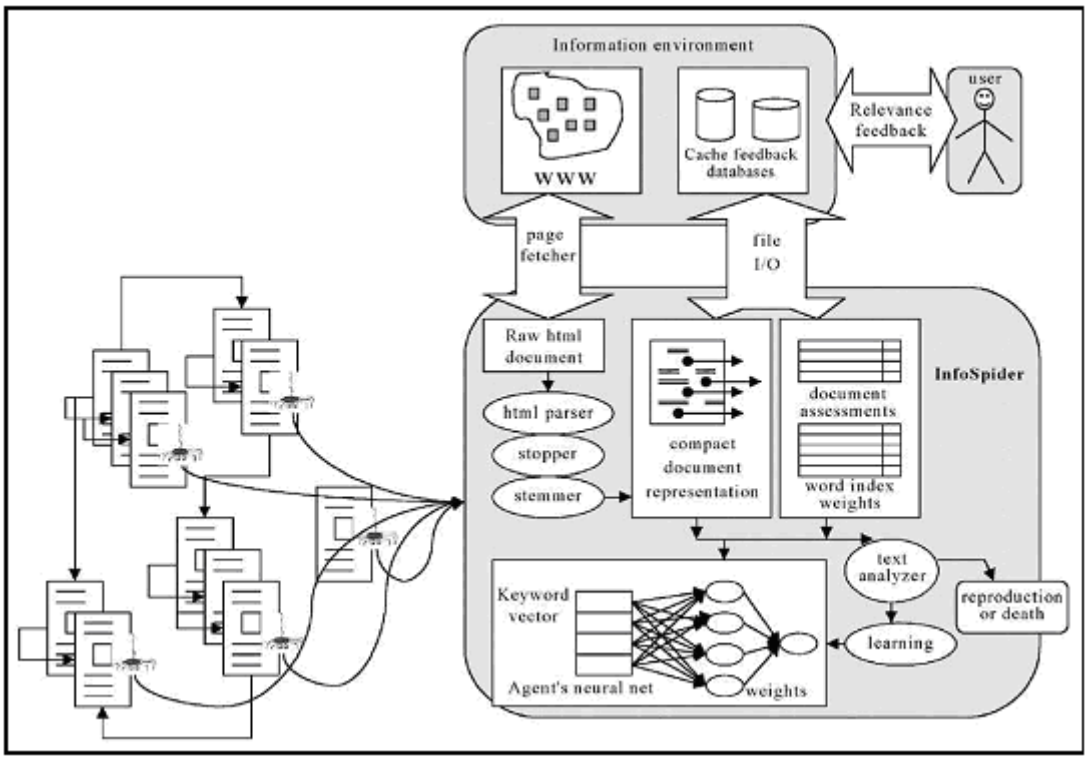

Fig.2.21 Architecture d'un agent dans le modèle InfoSpiders

L'architecture d'un agent dans le système InfoSpiders est représentée par la figure 2.21. Afin de pouvoir évaluer chaque agent de manière à sélectionner efficacement les plus performants d'entre eux, tous les individus possèdent une certaine énergie fixée uniformément à l'initialisation. Chaque agent est alors placé à l'origine sur une page considérée comme un bon élément de départ, une page relativement pertinente par rapport à une requête donnée. Afin de garantir de bons points de départ, ceux-ci sont issus de moteurs de recherche classiques, interrogés à l'aide de la requête de l'utilisateur, ou de signets définis par l'utilisateur lui-même [107].

On affecte à chaque agent un réseau de neurones dont les entrées correspondent à la liste des mots-clés formulés par l'utilisateur dans la requête. À chaque itération – dès qu'un agent se retrouve sur un nouveau document - les araignées chercheuses analysent le document qui leur est attribué afin de déterminer les meilleurs liens à suivre par la suite. Cette analyse consiste à examiner la présence des mots-clés de la requête dans le voisinage d'un lien. Le poids de chaque entrée du réseau de neurones – correspondant à un mot-clé - est mis à jour en fonction de son éloignement à un lien. Ce score est d'autant plus élevé que le mot-clé est proche d'un lien. Le réseau de neurones fait ensuite la somme de toutes les activités de ses entrées afin de calculer une fonction d'activation pour chaque lien. Le lien hypertexte que l'agent va par la suite suivre est sélectionné aléatoirement en favorisant les liens ayant obtenu le meilleur score au travers du réseau de neurones.

L'énergie de chaque agent est ensuite mise à jour en fonction de la pertinence à la requête du document analysé et du coût de téléchargement de la nouvelle page à visiter. Ces éléments étant à définir pour chaque implémentation du modèle. Le réseau de neurones est également mis à jour en fonction de la pertinence de la page analysée et de l'estimation que le réseau en avait faite à l'itération précédente en analysant le lien que l'agent a suivi. Cette adaptation est réalisée à l'aide d'une version adaptée à un réseau de neurones de l'algorithme Q-Learning

[138] réalisant un apprentissage par renforcement. Les poids utilisés dans le réseau de neurones sont alors mis à jour par rétro-propagation de l'erreur.

Ce modèle détermine une approche très intéressante qui démontre que l'utilisation de nouveaux algorithmes de recherche n'utilisant pas une indexation peuvent apporter une information plus approfondie et pertinente que les moteurs de recherche traditionnels.

2.2.5.3 Apprentissage par AG dans un agent

La capacité d'apprentissage des algorithmes génétiques a également été mise à contribution dans [139]. Le but est également de créer un agent chargé d'assister les utilisateurs dans leurs problèmes de recherche d'information. Cependant, ce système a la particularité d'être conçu à la manière d'un outil de veille stratégique : il est chargé de scruter en permanence une zone prédéfinie d'Internet afin d'y détecter l'apparition de documents pertinents pour un utilisateur. L'originalité de ce système réside dans sa manière de récupérer les souhaits de l'utilisateur et de les adapter au fur et à mesure de son utilisation.

Cet agent accepte des requêtes sous forme de documents exemples dont le contenu doit être représentatif du souhait de l'utilisateur. Le système va ainsi scruter un site donné - la zone d'Internet prise en compte dans la recherche et indiquée par l'utilisateur - afin de détecter les pages les plus similaires aux exemples fournis en entrée à l'agent de recherche.

Afin de mesurer la similarité d'une page à une autre, les auteurs utilisent le modèle vectoriel de représentation des documents. Une sorte de liste noire de termes est utilisée afin d'éliminer les mots les plus courants du langage. Les poids associés à chaque terme sont alors déterminés par la mesure *tf*idf*. Le corpus pris en compte pour le calcul de la composante *idf* correspond à l'ensemble des documents indiqués en entrée par les utilisateurs. Cet ensemble est alors désigné par le terme requête. Par contre, au cours de la recherche, il n'est pas possible de connaître l'ensemble du corpus pris en compte. Par conséquent, le calcul du vecteur représentatif d'un document en cours d'analyse ne pourra prendre en compte la composante *idf* et les poids des termes seront égaux à la valeur *tf*. La similarité peut alors être calculée grâce à la mesure *cosine* en utilisant les vecteurs de documents déterminés.

À partir de cette requête, l'agent analyse périodiquement l'ensemble des documents contenus dans le site visé. Lorsque des pages similaires au corpus représenté par la requête sont déterminées, elles sont présentées à l'utilisateur afin que ce dernier puisse établir une note. La requête initiale composée de documents choisis par l'utilisateur peut ainsi évoluer afin de prendre en compte les modifications de souhaits des utilisateurs. Cette adaptation, sorte d'apprentissage, se fait conjointement à l'aide d'un système de relevance feedback et d'un AG.

La population prise en compte dans l'AG est composée d'individus représentés par un vecteur de termes représentant un document correspondant à la requête. La fonction de fitness utilisée calcule une valeur représentant l'adaptation de cette requête à l'intérêt de l'utilisateur. Un opérateur de croisement à deux points est utilisé afin de mélanger les gènes correspondant aux termes représentés dans les vecteurs de documents, dimensionnés de manière similaire. L'opérateur de mutation consiste à sélectionner aléatoirement des poids de termes dans les vecteurs de documents et à les modifier en fixant une valeur choisie au hasard entre 0 et 1 [107].

2.3 RI contextuelle

Le développement du Web au niveau mondial a profondément transformé la gestion des documents. Cette révolution technologique a engendré de nouvelles problématiques documentaires pour la RI du fait de l'accroissement incessant et quasi exponentiel [140] du volume d'information disponible sur le Web ainsi que du nombre d'utilisateurs inexpérimentés dans la manipulation des moteurs de recherche. Parmi ces problématiques, l'intégration du contexte, qui constitue une priorité dans le traitement de grands volumes d'information. Surtous avec l'émergence du *Web ubiquitaire*, où un grand volume d'informations accessibles via différents réseaux hétérogènes et avec différents dispositifs informatique. Les résultats de recherche doivent s'adapter aux utilisateurs possédant différents besoins en information et en fonction des informations de leur contexte. Alors, la recherche d'information sur le *Web ubiquitaire* ajoute un surcroît important par les informations contextuelles.

Les prochaines générations de moteurs de recherche doivent faire face à un usage croissant du contexte, soit en utilisant des informations du contexte explicite ou implicite de l'utilisateur, soit en implémentant des fonctionnalités additionnelles au sein de contextes restreints. Une meilleure utilisation du contexte dans les moteurs de recherche permet d'aider à accroître la compétition et la diversité sur le Web. En outre, aux bases de données utilisées dans les SRI traditionnels, le Web se différencie en termes de contenu et de structure. En effet, il se caractérise par une forte hétérogénéité des sources d'information : hétérogénéité des langues (on y compte actuellement plus de cent langues différentes sur le Web), hétérogénéité des médias (texte, image, vidéo), hétérogénéité des structures, etc. La conceptualisation de la recherche permet d'adapter le contenu et la structure du document au contexte de l'utilisateur, et de diminuer ainsi l'hétérogénéité des sources d'information. Enfin, l'émergence du domaine de la RI contextuelle provient également de l'évolution de l'environnement physique de l'utilisateur : PDA, téléphonie mobile, etc. Plusieurs études ont considéré les implications de ce nouvel environnement en RI [140] .Les conclusions de ces travaux ont montré la nécessité de prendre en compte le contexte physique dans les modèles de RI.

Les besoins sont ainsi énormes. Dans ce cadre, la RI contextuelle émerge comme un domaine à part entière : sans remettre en cause ses origines, elle pose des problématiques nouvelles allant de la modélisation du contexte jusqu'à la modélisation de la pertinence cognitive en passant par la modélisation de l'interaction entre un utilisateur et un système de recherche d'information (SRI) [141].

L'objectif de cette section est d'apporter un éclairage sur ces problématiques, sur leurs origines, sur les verrous posés par la RI classique dans un tel cadre, ainsi que sur les solutions apportées dans la communauté.

2.3.1 Verrous technologiques

L'accès contextuel à l'information est confronté à des verrous que l'on peut classer sur trois niveaux : conception, évaluation et mise en œuvre.

- *Conception :* le contexte constitue une nouvelle dimension à modéliser puis décliner dans le modèle de RI. Ceci pose alors des problèmes liés à la définition, la formalisation, la mise en relation et exploitation conjointe de différents éléments potentiels : centres d'intérêts et préférences de l'utilisateur, son but, sa tâche, les

caractéristiques de son dispositif mobile *DM*, l'expression de sa requête, sa perception de la pertinence, ses interactions te sa localisation etc.

- *Evaluation :* les méthodologies d'évaluation classiques des systèmes d'accès à l'information devraient être révisées dans le sens de l'intégration du contexte. Cette révision portera essentiellement sur la définition de nouveaux protocoles d'évaluation basés sur des utilisateurs spécifiques, de nouvelles métriques d'évaluation tenant compte de différents niveaux de pertinence, de nouvelles collections de tests décrites par des méta-données descriptives des contextes de recherche.
- *Mise en œuvre :* l'exploitation de systèmes d'accès personnalisé pose un problème fondamental, d'ordre technologique, qui porte sur la protection de la vie privée. En effet, la définition, l'utilisation et la dissémination des profils constituent à la fois un atout et une contrainte pour assurer la portée des systèmes qui les supportent. C'est tout d'abord un atout dans le sens où les différents profils sont maintenus et sont accessibles, pouvant donc contribuer à mettre en œuvre une recherche d'information collective et dynamique par l'introduction de techniques d'apprentissage. Cependant, c'est ensuite une contrainte dans le sens où elle doit être impérativement soutenue par une réflexion sur les droits des personnes pour assurer la sécurité globale des profils [141].

La contextualisation de la RI doit donc s'accompagner d'une réflexion sur les architectures des systèmes de traitement d'informations et de la définition de techniques de protection de ces informations (méthode de cryptographie, modèles d'octroi ou révocation de droits).

2.3.2 Concepts et notions de base

Définissons tout d'abord les concepts et les notions de base qui interviennent dans le courant de la RI contextuelle.

2.3.2.1 Contexte

Les notions de *contexte* et *situation* sont en amont de l'interaction utilisateur-SRI. Ces notions ont été initialement introduites, sans distinction de sens, par les travaux de Saracevic [142] et Ingerwersen [143]. Le contexte (ou situation) y est défini(e) comme l'ensemble des facteurs cognitifs et sociaux ainsi que les buts et intentions de l'utilisateur au cours d'une session de recherche. Une tentative de distinction entre ces notions a fait l'objet d'autres travaux [144] qui précisent que le contexte couvre des aspects larges tels que l'environnement cognitif, social et professionnel dans lesquels s'inscrivent des situations liées à des facteurs tels que le lieu, le temps et l'application en cours. C'est le sens générique du contexte qui a été largement exploré cette dernière décennie en RI contextuelle [145].

Même si les auteurs ne convergent pas vers une même définition, on retrouve toutefois des dimensions descriptives communes telles que l'environnement cognitif, le besoin mental en information, l'interaction liée à la recherche d'information, la tâche de recherche, le type du besoin véhiculé par la requête, les ressources disponibles. Vu sous l'angle de la RI, le contexte possède, selon N. Fuhr [146], trois principales dimensions : social, application et temps. La dimension sociale définit la composante d'appartenance de l'utilisateur : individuel, groupe ou communauté. La dimension application définit le contexte applicatif du besoin exprimé : recherche *ad hoc*, résolution de problème ou *workflow*.

La dimension temps permet de décrire la circonscription temporelle du besoin exprimé : temps passé *(batch)*, instant courant ou à court terme (interactive), intention ou long terme (personnalisation). Sous l'angle de la dimension temps, on distingue deux types de contexte avec des démarches de personnalisation appropriées. Le contexte courant ou à court terme décrit les besoins et préférences de l'utilisateur lors d'une session d'une recherche. Le contexte persistant décrit les besoins à long terme de l'utilisateur sur diverses sessions de recherche [141].

2.3.2.2 Profil

Le concept de profil est directement lié à l'utilisateur. L'utilisation de ce concept a été introduite par les travaux en filtrage d'information, pour décrire une structure représentative de l'utilisateur, plus particulièrement de ses centres d'intérêts. Cette notion est réutilisée en RI contextuelle pour cibler les éléments du contexte dépendant directement de l'utilisateur : centres d'intérêts, familiarité avec le sujet de la recherche, domaine professionnel, expertise, etc.

2.3.2.3 RI contextuelle ou RI personnalisée

L'objectif de la RI contextuelle est de délivrer une information pertinente et appropriée au contexte de l'utilisateur qui a émis la requête. La RI contextuelle traduit précisément l'exploitation des éléments du contexte de la recherche dans l'une des principales phases de l'évaluation de requête : reformulation, calcul du score de pertinence de l'information, présentation des résultats de recherche. La RI personnalisée est, à notre connaissance, un type de RI contextuelle où l'accent est mis sur l'utilisation d'un modèle de l'utilisateur préalablement construit [147, 148]. La littérature du domaine ne fait pas état, cependant, d'une distinction franche entre RI contextuelle et RI personnalisée.

2.3.2.4 Pertinence contextuelle

La pertinence est incontestablement la question fondamentale posée en RI. Cette notion subjective, dépendant essentiellement du point de vue de l'utilisateur, a de nouveau été l'objet d'investigations dans le cadre de la RI orientée utilisateur de manière générale, de la RI contextuelle de manière particulière. Dans [149], on rapporte que la pertinence est un concept multidimensionnel. On distingue principalement, quatre types de pertinence :

- *pertinence algorithmique* : la pertinence est traduite par une mesure algorithmique dépendant des caractéristiques des requêtes d'une part et des documents d'autre part. C'est le seul type de pertinence qui est indépendant du contexte ;
- *pertinence thématique* : la pertinence traduit le degré d'adéquation de l'information à couvrir, en partie, le thème évoqué par le sujet de la requête. C'est le type de pertinence adressé par les assesseurs de la campagne d'évaluation TREC ;
- *pertinence cognitive* : c'est la pertinence liée au thème de la requête, « pondérée » par la perception ou les connaissances de l'utilisateur sur ce même thème ;
- *pertinence situationnelle* : c'est la pertinence liée à la tâche de recherche. Ce type de pertinence traduit essentiellement l'utilité de l'information relativement au but de recherche de l'utilisateur [141].

La RI contextuelle explore essentiellement la pertinence cognitive et la pertinence situationnelle.

2.3.3 Modèles de RI contextuelle

Les modèles classiques ont été définis en supposant un utilisateur unique, un besoin d'information pour chaque requête, une localisation, un temps, un historique et un profil.

Des techniques de capture; pour capturer le temps, l'espace, l'historique et les profils sont injectés dans les modèles. Des capteurs pour récupérer la localisation, des fichiers *logs* pour implémenter l'historique, des méta-données pour décrire les profils, des horloges et des calendriers pour récupérer le temps sont alors nécessaires.

2.3.3.1 Modélisation du contexte dans les SRI

Les approches et techniques de modélisation du contexte sont au cœur du processus d'accès contextuel à l'information. L'objectif du système d'accès contextuel à l'information est de délivrer une information pertinente en fonction du contexte de l'utilisateur. Le contexte peut être exploité à différentes phases du processus d'accès à l'information : dans la formulation de la requête, dans l'accès, dans l'ordonnancement des résultats. Les éléments du contexte peuvent être utilisés pour reformuler une requête. La reformulation de requête consiste à augmenter la requête avec des informations du contexte avant de lancer le processus d'appariement. Le contexte peut également intervenir dans la définition de la fonction de pertinence. Dans ce sens, [150] a proposé l'adaptation des paramètres de la fonction de pertinence au contexte de l'utilisateur, en utilisant les techniques de programmation génétique. Jeh et Widom [151] ont proposé une variante personnalisée de l'algorithme PageRank (détaillé dans la section 2.2.1.3.3) en l'occurrence PPV (*Personnalized PageRank Vector*). Son principe fondamental consiste à privilégier les pages reliées aux pages préférées de l'utilisateur ou pages citées par ces dernières au cours du processus de calcul des scores de sélection. L'ordonnancement des résultats constitue la phase ultime du processus d'accès à l'information. Cette phase peut également prendre en compte le contexte pour réordonner les résultats fournis par le processus de sélection. De ce fait, l'ordre final des documents à présenter à l'utilisateur est une combinaison de l'ordre produit par le processus de sélection et celui fourni par le contexte de l'utilisateur *via* un calcul de similarité [152] ou des jugements explicites de la pertinence [153].

Plusieurs modèles peuvent être utilisés pour prendre en compte le contexte en RI : basé sur des *logs* de requêtes, Wen [154] construit un modèle probabiliste par expansion des termes de la requête pour la recherche contextuelle. Melucci [155] propose une extension du modèle vectoriel intégrant le contexte de l'utilisateur. Rode et Hiemstra [156] proposent un modèle basé sur un langage statistique pour incorporer des informations du contexte dans le processus de recherche. Finkelstein [157] développe un système de recherche qui utilise le contexte entourant le texte pour améliorer l'extension de requête. [158] exploitent des profils basés sur des ontologies pour des recherches personnalisées. [159] étudient l'amélioration de la recherche en se basant sur des *web log data*. SearchPad [160] explicite des contextes de recherche sur le Web

Nous allons présenter plus en détail deux modèles de RI contextuelle qui sont des extensions des modèles classiques de RI, en l'occurrence les modèles probabiliste et vectoriel.

2.3.3.2 Extension du modèle probabiliste

Dans les travaux sur la RI contextuelle, requête, contexte et document sont représentés par le même type de composant : le mot. Il est ainsi aisé d'adapter les données du contexte dans le cadre de la RI textuelles. Dans de nombreux travaux, le contexte est simplement combiné avec la requête pour effectuer une expansion de requête. Cependant, le contexte peut contenir des informations non compatibles avec les requêtes et les documents d'où une impossibilité d'incorporer le contexte dans une requête. La méthode proposée dans [155] fait appel aux *logs* de requêtes. Ces *logs* de requête enregistrent les sessions de requêtes précédentes. Une session requête en RI contextuelle enregistre une séquence requête-contexte-document sous la forme suivante :

Session requête := <requête, contexte> [*clicked documents*]

Chaque session contient une requête, son contexte et un ensemble de documents que l'utilisateur a sélectionnés (par un *click*) (appelés *clicked* documents). L'idée générale du modèle est que si un ensemble de documents est souvent sélectionné pour des requêtes similaires dans des contextes similaires alors les termes de ces documents sont strictement liés aux termes des requêtes et aux éléments du contexte. De même, si des requêtes similaires et des contextes similaires sont fréquemment co-occurrents dans les *logs*, les termes de la requête sont bien corrélés aux éléments du contexte. Ainsi des corrélations probabilistes entre les termes de la requête, les éléments du contexte et les termes des documents peuvent être établies sur la base des *logs* de requêtes [141], comme on peut le constater sur la figure 2.22.

L'information mutuelle est utilisée pour déterminer les degrés de corrélation entre les termes de la requête, les éléments contextuels et les termes des documents (figure 2.23). L'information mutuelle est une mesure de dépendance statistique entre deux variables aléatoires basée sur l'entropie de Shannon. Il s'agit d'un score d'association de deux mots *(x, y)* noté *IM* qui permet de comparer la probabilité d'observer ces deux mots ensemble avec la probabilité de les observer séparément. Selon [161],

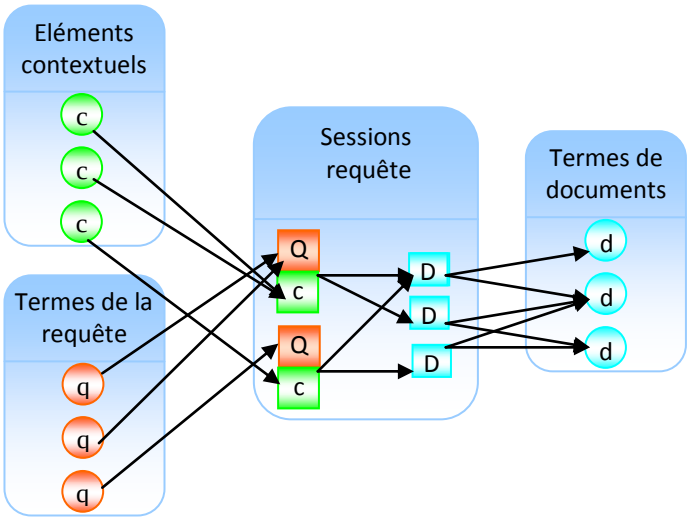

Fig. 2.22 Les sessions de requête construisent les corrélations entre les termes de la requête, les éléments contextuels et les termes des documents

La définition du score IM est la suivante : $IM(x, y) = log2(P(x, y)/(P(x)P(y)))$ où P(x) et $P(y)$ sont les probabilités d'observer les mots x et y, et P(x, y) est la probabilité de les

observer simultanément. Si *IM(x, y)* est fortement positive, cela signifie que *x* et *y* apparaissent très souvent ensemble. Si *IM(x, y)* est proche de 0, alors *x* et *y* n'ont aucun rapport et enfin, si *IM(x, y)* est fortement négative, alors *x* et *y* ont des distributions complémentaires [141].

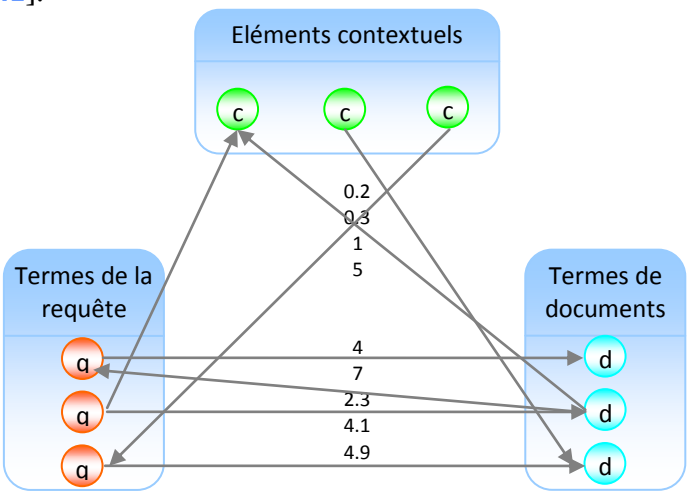

Fig 2.23 Information mutuelle entre les termes de la requête, les éléments contextuels et les termes des documents

Dans la suite et sur la base de ces corrélations, nous présentons quatre modèles probabilistes qui ont été proposés pour générer des expansions de termes pour la recherche contextuelle.

2.3.3.3 Modèle orienté contexte

Le premier modèle est assez simple et intuitif : les termes des documents bien corrélés au contexte sont choisis comme expansion de termes pour modifier la requête initiale :

$$M1(d \triangleright Q,C >) = I(d,C) = \sum_i I(d, ci) \quad (1.14)$$

Où *I(d,C)* est l'information mutuelle entre le contexte et les termes du document. Du point de vue de la combinaison de la requête et du contexte, ce modèle est similaire à la solution traditionnelle en recherche contextuelle. Cependant, ici, les termes utilisés pour l'expansion de requête ne proviennent pas directement du contexte mais des documents corrélés au contexte.

2.3.3.4 Modèle indépendant requête-contexte

Le principal défaut du premier modèle est que les termes de l'expansion sont générés uniquement à partir du contexte et ne sont donc pas corrélés à la requête. Le deuxième modèle utilise la requête et le contexte pour contrôler le processus d'expansion de requête :

$$M2(d \triangleright Q,C >) = I(d,< Q,C >) \quad (1.15)$$
$$= I(d,C) + I(d,Q)$$
$$= \sum_i I(d, ci) + \sum_i I(d, qi)$$

Où *I(d,Q)* est l'information mutuelle entre la requête et les termes du document. L'avantage du deuxième modèle par rapport au premier réside dans le fait qu'il affecte des poids plus élevés aux termes des documents qui sont corrélés à la fois à la requête et au contexte [141].

2.3.3.5 Modèle dépendant requête-contexte

Pour diminuer l'effet de l'hypothèse d'indépendance entre la requête et le contexte, un troisième modèle a été introduit pour prendre en compte les relations de dépendance entre la requête et le contexte. Ce troisième modèle est défini par :

$$M3(d \triangleright Q,C >) = I(d,< Q,C >) = \sum i\, I(d, ci) + \sum j\, I(d, qj) + \sum ij\, I(d,< qj , ci >) \quad (1.16)$$

Où $P\ I(d,< qj , ci >)$ est l'information mutuelle entre un terme de document et une paire requête-contexte. C'est ce facteur qui introduit une dépendance requête-contexte. Le paramètre est introduit pour ajuster le poids de ce facteur de dépendance requête-contexte [141].

2.3.3.6 Modèle filtrant le contexte

Un problème commun entre les trois premiers modèles est que le bruit dans un contexte n'est pas traité. Manifestement, le bruit du contexte peut facilement produire des expansions de termes hors de propos. Ainsi, le quatrième modèle utilise l'information mutuelle entre requête et contexte pour éliminer le bruit dans les éléments du contexte :

$$M4(d \triangleright Q,C >) = I(d,< Q,C >) = I(d,< Q,C'>) = \sum i\, I(d, ci) + \sum j\, I(d, qj) + \sum ij\, I(d,< qj , ci >)$$

$$\text{Où } C' = \{c/c \in C, I(c,Q) \geq \alpha\}. \quad (1.17)$$

$I(c,Q)$ est l'information mutuelle entre la requête et un élément contextuel. Elle est utilisée pour filtrer des éléments contextuels non corrélés à la requête courante, α représente le seuil pour le processus de filtrage [141].

La méthode d'expansion de requête est basée sur les modèles probabilistes décrits ci-dessus. Lorsqu'une nouvelle requête avec contexte est soumise, les termes des documents corrélés sont sélectionnés et classés selon leur probabilité conditionnelle obtenue par les modèles. Enfin les meilleurs termes classés peuvent être sélectionnés comme termes d'expansion.

2.3.3.7 Extension du modèle vectoriel

M. Melucci [156] propose de modéliser le contexte par un vecteur. Plus précisément, le contexte est modélisé par des bases d'espaces de vecteurs et son évolution est modélisée par des transformations linéaires d'une base à une autre. Chaque document ou requête peut être associé à une base distincte, qui correspond à un contexte.

Avant de présenter la contribution de M. Melucci [156], nous rappelons ci-dessous les éléments principaux du modèle vectoriel.

Dans le cadre du modèle vectoriel classique, un contexte unique est considéré pour le document et la requête. Soit $\{t1, \ldots, tn\}$ une liste d'autorités, une base T est modélisée telle qu'il existe un vecteur de T pour chaque terme. Chaque document et chaque requête est vectorisé par $T : Z = \sum_{i=1}^{n} a_i t_i$ où Z est le document ou la requête, ti est le terme et ai le poids de ti pour générer Z. Le classement des documents trouvés est basé sur le calcul suivant où x représente un document, y une requête et $T^x \cdot T$ la matrice de corrélation :

$$x \cdot y = (T \cdot a)^T \cdot (T \cdot b) = a^T \cdot (T^T \cdot T) \cdot b \quad (1.18)$$

Les hypothèses du modèle vectoriel contextuel sont les suivantes : une base de vecteurs modélise un descripteur de document ou de requête. La sémantique d'un descripteur de document ou de requête dépend du contexte. Une base peut être dérivée d'un contexte. De ce fait, une base de vecteurs modélise le contexte. En d'autres termes, chaque document et chaque requête sont associés à des bases distinctes et les liens d'une base à une autre sont gouvernés par une transformation linéaire. Par exemple, soit $n = 2$ et $t1$, $t2$ deux descripteurs, disons *Padoue* et *Venise*, respectivement. Si un utilisateur recherche des informations sur *Padoue* et voyage autour de cette ville, une requête q contient $t1$ avec le poids 1 ; si les vecteurs représentant les descripteurs sont supposés orthonormaux, alors des vecteurs peuvent être utilisés et $q = [1, 0]^T$. Si une information contextuelle est disponible relativement à l'espace,

Par exemple, l'utilisateur s'est déplacé jusqu'à *Venise*, les descripteurs ne peuvent plus être considérés comme orthonormaux et l'ensemble de vecteurs $[1, 2]^T$, $[3, 1]^T$ peut être utilisé comme base, et ainsi $q = [1, 3]T$. Toute base peut être projetée dans une autre par une transformation linéaire. Ainsi, le contexte influence non seulement le choix d'un descripteur mais aussi sa sémantique et la manière dont il se réfère aux autres descripteurs.

Le vecteur x du document x écrit dans son propre contexte est généré par la base T qui n'est pas nécessairement égal à la base U qui génère un vecteur requête y ou à la base T' qui génère un autre vecteur document. Ainsi, x est représenté *par* $x = T \cdot a$ alors que y est représenté par $y = U \cdot b$ où a et b sont les coefficients utilisés pour combiner les bases de vecteurs de T et U respectivement. Si la pertinence est estimée par le produit scalaire classique, les documents sont rangés grâce à la formule :

$$x^T \cdot y = a^T \cdot (T^T \cdot U) \cdot b. \qquad (1.19)$$

Cette dernière formule montre que la relation entre les descripteurs utilisée pour exprimer les documents et la requête dépendent de deux contextes : le premier est impliqué dans l'indexation du document et le deuxième dans la formulation de la requête.

M. Melucci [156] propose également un modèle pour l'évolution du contexte basé sur le principe suivant : tout changement de contexte peut être modélisé par des transformations linéaires d'une base vers une autre.

2.3.4 Systèmes de RI contextuelle

Les moteurs de recherche du *web* traitent généralement des requêtes isolées. Les résultats pour une requête donnée sont identiques, indépendants de l'utilisateur, ou du contexte dans lequel l'utilisateur pose sa requête. Les informations du contexte peuvent être fournies par l'utilisateur sous la forme de mots-clés ajoutés à une requête, par exemple un utilisateur recherchant la page web personnelle d'un individu devrait ajouter des mots-clés comme *home* ou *home page* à la requête. Cependant, fournir un contexte sous cette forme est difficile et limitée. Une solution pour ajouter des informations bien définies du contexte à une requête consiste pour le moteur de recherche à demander spécifiquement de telles informations. On peut classer les systèmes de recherche d'information contextuelle selon que le contexte est explicitement demandé à l'utilisateur ou automatiquement inféré.

2.3.4.1 Le projet INQUIRUS 2

Le projet INQUIRUS 2 [162] requiert les informations du contexte sous la forme d'une catégorie d'informations désirées. Les utilisateurs doivent choisir une catégorie telle que «

pages personnelles », « papiers de recherche », « événements actuels » ou « introduction générale » lorsqu'ils posent leur requête sous forme de mots-clés. INQUIRUS 2 est un méta-moteur de recherche qui exploite la requête exprimée explicitement par l'utilisateur ainsi qu'un ensemble de préférences pour réécrire la requête et identifier le moteur de recherche à utiliser. Les informations du contexte sont utilisées pour sélectionner les moteurs de recherche à qui envoyer les requêtes, pour modifier les requêtes et pour sélectionner la politique d'ordonnancement des résultats. Le contexte correspond donc à une requête reformulée, adaptée à la catégorie de recherche. Par exemple, une requête qui recherche des articles sur *machine learning* pourra envoyer des requêtes multiples à des moteurs de recherche. Les termes ajoutés à la requête si on recherche des papiers de recherche seront *abstract, keywords, introduction*.

2.3.4.1.1 Génération automatique des informations du contexte

INQUIRUS 2 a permis d'améliorer de manière satisfaisante la précision de la recherche, mais impose à l'utilisateur de rentrer explicitement les informations du contexte. L'objectif du projet Watson [163] est d'inférer automatiquement les informations du contexte. Watson vise à modéliser le contexte de l'utilisateur en se basant sur le contenu des documents édités par Microsoft Word, ou visualisés par Internet Explorer. Les documents que les utilisateurs éditent ou visualisent sont analysés à l'aide d'un algorithme de termes pondérés, qui vise à identifier les mots qui donnent une indication sur le contenu du document. Des informations comme la taille de la police sont également utilisées pour assigner un poids aux termes. Si un utilisateur exprime une requête explicite, Watson modifie la requête en se basant sur le contenu du document édité ou visualisé, et envoie la requête modifiée vers des moteurs de recherche, ajoutant ainsi automatiquement des informations du contexte à la recherche sur le *web*. Les autres projets similaires sont nombreux : Margin Notes [164] qui réécrit les pages *web* pour intégrer des liens vers des fichiers personnels ; le projet *Haystack* [165] dont le but est de créer une communauté de *Haystack* interactif ou des entrepôts d'informations personnelles ; et le programme Automy's Kenjin (www.kenjin.com) qui suggère automatiquement du contenu du *web* ou de fichiers locaux, basé sur les documents qu'un utilisateur lit ou édite. Nous pouvons citer également les systèmes suivants qui apprennent les profils d'intérêt de l'utilisateur pour recommander des pages *web* : Letizia [166], WebWatcher [167], Siskill and Weber [168], Web Personne et SIS [169].

2.3.4.2 Deviner les besoins de l'utilisateur

Une technique en croissance sur le *web* consiste à deviner le contexte de la requête de l'utilisateur. Les moteurs de recherche Excite : (www.excite.com), Lycos : (www.lycos.com), Google : (www.google.com) et Yahoo : (www.yahoo.com) intègrent des fonctionnalités spéciales pour certains types de requêtes. Par exemple, des requêtes posées à Excite et Lycos sur le nom d'un artiste ou une entreprise produisent des résultats additionnels qui orientent directement aux informations sur l'artiste ou l'entreprise. Yahoo a ajouté de telles fonctionnalités en 2000, et fournit des résultats spécialisés pour de nombreux types différents de requêtes - par exemple, des noms d'équipes de sport sont reliés à des informations d'équipes ou de ligues. D'autres exemples avec Yahoo incluent des modèles de voiture, des célébrités, des musiciens, des villes importantes, des noms de médicaments ou de maladies, des signes du zodiaque, des lignes d'aviation, des shows de télévision et des parcs nationaux. Google identifie des requêtes qui ressemblent une adresse de rue américaine, et fournit des liens directs avec des plans de ville ou de quartier. Plutôt que de demander explicitement à l'utilisateur des informations telles que *je cherche un plan de ville* ou *je cherche le site web*

d'une équipe de football, cette technique devine lorsque de tels contextes peuvent être pertinents. Les utilisateurs peuvent relativement aisément identifier des contextes d'intérêt. Cette technique est limitée aux cas où des contextes potentiels peuvent être identifiés à partir de la requête sous forme de mots-clés [141].

Une autre solution pour ajouter du contexte à une recherche sur le Web est de restreindre le contexte des moteurs de recherche. Des milliers de ces moteurs de recherche existent déjà. Nous pouvons citer www.invisibleweb.com et www.completeplanete.com.

2.4 Conclusion

Comme nous l'avons vu, dans ce chapitre, de nombreux travaux ont été menés sur la problématique de la recherche d'information dans le Web. Il s'agit en fait, d'un problème crucial dans une époque ou la quantité d'information accessible devient extrêmement importante. Il est en effet, impossible de se passer d'outils de recherche automatiques.

Dans ce chapitre, nous avons fait le point - dans la première partie - de l'état de l'art sur les techniques de recherche d'informations usuelles. La plupart des systèmes de recherche d'information classiques se basent sur une indexation par termes simples. Cependant, ces derniers délivrent beaucoup de résultats en réponse aux requêtes des utilisateurs. Ceci est du en partie au fait que le contenu sémantique d'un document (ou d'une requête) ne peut pas être capturé précisément par un simple ensemble de mots clés indépendants. La plupart des systèmes de recherche d'information (SRI) privilégient la minimisation de temps de réponses par rapport à la qualité des documents retournés à l'utilisateur. En effet, ces derniers délivrent de grandes quantités de documents en réponse aux requêtes des utilisateurs, ce qui génère ainsi, une surcharge informationnelle dans laquelle il est difficile de distinguer l'information pertinente.

Les modèles de recherche booléens, vectoriels ou probabilistes présentent différentes solutions performantes, robustes et relativement simples à mettre en œuvre. Sur le Web, les outils de recherche correspondent aux moteurs de recherche. Il existe toute une panoplie de moteurs de recherche sur le Web qui se différencient notamment par la taille de leur base d'indexation, leurs langages d'interrogation, le type d'indexation utilisé…

L'architecture des moteurs classiques est aujourd'hui caractérisée par l'utilisation d'un système de recherche d'information distribuée SRID qui se base sur un ensemble de machines fonctionnant en parallèle, à l'instar de Google. La pertinence des réponses est liée à un système de tri de pertinence construit sur la notion de lien existant entre les pages. Dans ce cadre, nous pensons que ces nouvelles stratégies de recherche pourront accéder plus rapidement aux documents recherchés, en permettant de plus l'utilisation d'une requête plus riche, associée par le contexte de la recherche, que ce soit, des préférences de l'utilisateur, de sa localisation et des caractéristiques du terminal utilisé.

Ce chapitre nous a permis d'aborder des heuristiques de recherche d'information sur le Web. Les approches à base de colonies de fourmis et les algorithmes génétiques utilisés classiquement dans le domaine de l'optimisation s'adaptent particulièrement bien aux problèmes de la recherche d'information dans des sources d'informations distribués et hétérogènes. Bien qu'elles soient très utilisées en optimisation et ont montré des qualités pour la résolution de multiples problèmes, Les propriétés de parallélisation de chacune des

méthodes peuvent permettre de réaliser un outil efficace de recherche d'information sur Web ubiquitaire. Evolutif et adaptable aux conditions de dynamicité très forte de l'environnement ubiquitaire d'information.

À partir de ces éléments, nous avons décidé d'élaborer une approche de RI sur Web ubiquitaire. Après avoir présenté une modélisation du comportement de nos agents de recherche : il s'agit de transformer le problème de la structuration du modèle de comportement des agents de recherche en un problème d'assemblage de rôles de recherche : composants de granularité fine, évoluant dans le temps. Il s'agit de tirer des fragments des algorithmes génétiques, ainsi que des fragments des algorithmes à base de colonies de fourmis. L'organisation de ces rôles de recherche se fait dans une base de rôles de recherche bien conçus en vu de les faire évoluer et pourquoi pas de les faire standardiser. Cette base de recherche sera déployée aux seins des serveurs d'information.

Par conséquent, Les ressources à la disposition de notre approche de RI seront alors exploitées avec le potentiel des algorithmes qui implémentent des méta-heuristiques trouvant principalement leurs sources d'inspiration de la nature.

La deuxième partie montre que la RI contextuelle constitue actuellement un domaine de recherche très actif qui vise l'amélioration de la RI sur le Web ubiquitaire, Il s'agit d'un domaine de recherche récent dont l'émergence provient d'une part de l'évolution des supports physiques (PDA, téléphonie mobile) et d'autre part de l'accroissement du volume d'information sur le Web. Un point à approfondir porte sur la modélisation du contexte. Cependant, Il existe une multitude de définitions du contexte dans la littérature. En effet, Une définition standardisée des éléments du contexte pourrait être proposée. Une autre point concerne les modèles de RI contextuelle. De nombreux modèles de RI contextuelle ont été définis ces dernières années dont des extensions des modèles classiques (vectoriel, probabiliste) permettent déjà la prise en compte de plusieurs éléments contextuels. Les nouveaux modèles doivent prendre en compte toutes les dimensions du contexte (préférence utilisateur, temps, espace, localisation, caractéristiques du terminal, etc.).

Chapitre 3

Recherche d'information volumineuse

3.1 Introduction

La croissance continuelle et exponentielle des volumes d'information numérique affecte principalement des domaines comme celui de la Recherche d'Information (RI). Le World Wide Web fournit un environnement riche pour la recherche d'information. Jour après jour, l'Internet se développe et la quantité d'information disponible devient si importante que les utilisateurs peuvent facilement se perdre dans cette grande source d'information malgré les outils d'aide à la recherche tel que les moteurs de recherche.

L'explosion du volume d'informations sur le Web est à l'origine d'une croissance très significative des applications destinées à aider la majorité des utilisateurs à produire de l'information et à y accéder facilement. Cette explosion, tant du volume d'information que du nombre d'utilisateurs amène de nouveaux problèmes et ouvre de nouvelles pistes de réflexion. Le traitement de grands volumes d'informations est souvent désigné par l'expression « *passage à l'échelle* » [2]. Plus précisément, le passage à l'échelle d'une technique à une autre et la capacité à traiter des masses considérables d'informations tout en conservant une complexité du même ordre de grandeur réelle.

D'un côté, des solutions avérées ont été apportées par la communauté scientifique en recherche d'information pour améliorer sans cesse l'ensemble des fonctions d'un système de recherche d'information, plus particulièrement les résultats importants obtenus dans le domaine de la modélisation de l'information, de l'évaluation de requêtes et de la visualisation des résultats des requêtes. D'un autre côté, La recherche d'information, n'a cessé d'évoluer dans le but de rationaliser le processus complexe permettant l'identification, au sein de volumes de plus en plus importants d'informations, celles qui sont potentiellement intéressantes pour l'utilisateur.

Force est de constater que les solutions traditionnelles connues ne peuvent conserver leur degré d'efficacité, voire leur faisabilité, dans le contexte actuel où la problématique de la recherche d'information a pris une nouvelle dimension. Cette inadéquation mérite des investigations qui permettraient, d'une part, de transposer en partie des acquis considérable [2].

Pour répondre à cette problématique des grands volumes d'informations,

La section 2 présente les facteurs à l'origine de la croissance du WWW, une caractérisation sous l'angle des dimensions *espace/temps* ainsi que les conséquences engendrés.

La section 3 présente le problème du passage à l'échelle à travers sa projection sur le processus de base de la RI. Plus précisément, nous abordons les contraintes imposées par les grands volumes d'informations sur les phases de préparation des collections, d'évaluation des requêtes et de visualisation des résultats d'une recherche.

La section 4 analyse la viabilité des protocoles d'évaluation de la recherche d'information dans un contexte de grande échelle. Enfin, la section 5 dresse un bilan des réflexions menées dans ce cadre. Ce bilan se décline essentiellement dans la définition de directions de recherche permettant de s'affranchir en partie des verrous liés au passage à l'échelle.

3.2 Croissance du volume d'information

À l'époque, plusieurs giga-octets (10^9) de données ont été considérés comme des grands et des dizaines de giga-octets ont été considérées comme énormes. Avec les progrès récents dans la technologie de stockage et de la baisse des prix de stockage, un téra-octet (10^{12}) de données n'est pas rare, et de grands volumes de données sont maintenant mesurés en péta-octets (10^{15}). Alors il est évident, que l'analyse des ensembles de très grands volumes continuera à être un problème grandissant.

Bien que les progrès des techniques de stockage ont été importants, les progrès dans le traitement des données d'analyse a été marginal. Il est possible de trouver un morceau de données dans un système de stockage péta-octet de taille (Google, Yahoo! Et d'autres fournisseurs de moteur de recherche de technologie sont un témoignage de cela), mais l'analyse de ces données pour trouver des corrélations et des tendances significatives à un analyste reste un énorme défi.

On peut s'attendre à une continuité dans cette croissance car le nombre d'utilisateurs est sans cesse grandissant et leurs activités s'intensifient avec la mise à disposition de nombreux services (bibliothèques numériques, moteurs de recherche, annuaires, bases de données, ...). Plusieurs autres raisons justifient cette croissance d'information numérique. Dans le domaine de la recherche d'information, la dimension du problème et le contexte d'utilisation ont changé dans des proportions considérables depuis quelques années. En effet, l'espace de stockage augmente continuellement puisque :

- Les techniques de digitalisation sont de plus en plus performantes et il y a un besoin plus pressant d'échanger rapidement et efficacement de l'information;
- Les documents électroniques contiennent de plus en plus souvent des informations multimédias (images et graphiques sont courants, mais audio et vidéo tendent à se généraliser) ;
- Des méta-informations sont générées et associées aux données de base afin de faciliter les accès ultérieurs ;
- Les utilisateurs accèdent à des sources de plus en plus vastes et disséminées. Le cas extrême étant la Toile. La question de la structuration et de l'accès aisé à cette masse d'information alors un enjeu important pour la RI

Cette croissance du volume se constate également à plusieurs niveaux de granularité :

- Nombre de collections ;
- Nombre de documents au sein d'une collection ;
- Nombre des granules au sein des documents (semi) structurés ;
- Nombre de descriptions ou méta-données associées aux granules ;
- Nombre d'index associés aux données et méta-données.

Ainsi, la recherche d'information étant le processus permettant de renvoyer à des utilisateurs le sous-ensemble des documents pertinents contenus dans une collection, il est ainsi clair qu'aujourd'hui apparaissent de nouveaux enjeux. Cette section présente les principaux facteurs de la problématique des grands volumes d'informations puis on tente de les projeter sur les dimensions d'espace et de temps.

3.2.1 Facteurs de croissance

L'analyse de tout phénomène nécessite une étude préliminaire de ses origines, partant de l'information où la donnée élémentaire jusqu'aux collections de très grands volumes d'informations, La Web constitue la collection géante et la source d'information accessible à un très large public et tendant à devenir la première référence.

3.2.1.1 Donnée et information

Trop souvent, les mots «données» et «information» sont utilisés de façon interchangeable, quand il y a une distinction très importante entre les deux. Un moyen facile de penser à la différence est la suivante:

- une donnée est l'entrée pour le système d'analyse,
- l'information est la sortie.

L'analyse est le processus de transformation des données en information. Bien qu'il puisse sembler de base, cette distinction est importante, car elle ouvre une autre façon d'aborder le problème de l'analyse de grands volumes de données, plutôt que de s'appuyer sur des approches traditionnelles de base de données. Examinons un scénario simple de flux de données d'analyse de réseau: un flux de données réseau est capturé pour chaque bloc unique (par paquets) de données qui se déplace sur le réseau. La simple opération d'une personne regardant une page sur un site Web va générer un bon nombre des transactions de débit réseau qui permet d'appréhender la demande allant de l'utilisateur vers le site Web, et la réponse va du site Web à l'utilisateur. Un réseau unique de transaction de débit est composé d'une source (une adresse IP source), une cible (une adresse IP de destination), et une certaine informations sur la taille des données déplacées. A ce niveau de granularité, la transaction de débit du réseau est un bon exemple de données - pas l'information - avec la valeur analytique d'une transaction individuelle proche de zéro. Mais dès que plusieurs opérations de flux sur le réseau sont associés à une recherche Web, on peut avoir accès à certaines informations de base opérationnelle, tels que:

- Quelle quantité de données a été transférée pour une page Web en particulier ?
- Combien de temps a-t-il pris?
- Y a-t-il des erreurs générées dans le processus?

La recherche d'information sur la Toile se singularise vis-à-vis d'une recherche d'information classique par un degré plus accentué de nombreux paramètres : taille, hétérogénéité et dynamique des collections, variété des langues utilisées, interconnexion des documents, nombre et variété des profils des utilisateurs [170]. Il est ainsi clair que le passage à l'échelle se pose avec d'avantage d'acuité dans le cas de la recherche sur la Toile, si bien que des techniques proposées pour traiter de très larges collections de documents ont déjà été proposées dans ce contexte [171].

On peut dégager les deux principaux facteurs suivants de la problématique des grands volumes d'informations :

- Croissance du nombre des utilisateurs.
- Croissance de la taille des collections ;

3.2.1.2 Croissance du nombre des utilisateurs

Mediametrie[1] vient de publier ses chiffres sur l'usage et l'audience d'Internet en décembre 2009.

1- Mediametrie, Mesure d'audience medias audiovisuels Television, Radio, Cinema, Internet, cable, satellite, multimedia, media audiovisuel, TNT, ADSL, **...**www.**mediametrie**.fr

Utilisateurs d'Internet

- 1,73 milliards d'utilisateurs au niveau mondial (Septembre 2009), soit une augmentation de 18% par rapport à 2008
- 738,257,230 utilisateurs en Asie.
- 418,029,796 utilisateurs en Europe.
- 252,908,000 utilisateurs en Amérique du Nord.
- 179,031,479 utilisateurs en Amérique Latin et Caraïbes.
- 67,371,700 utilisateurs en Afrique.
- 57,425,046 utilisateurs au Moyen-Orient.
- 20,970,490 utilisateurs en Océanie et Australie.

Une étude réalisée sur 211 063 utilisateurs [172] révèle une tendance à l'interrogation à l'aide de requêtes courtes ; plus précisément plus de 48,4 % de requêtes se limitent à un unique terme, 20,8 % ont deux termes et 31 % ont trois termes ou plus. Cette étude rapporte en outre que les utilisateurs consultent rarement les résultats de recherche figurant au-delà de la deuxième page. Le facteur de croissance du nombre d'utilisateurs pose alors un double impératif :

- Sur les délais des réponses, issues d'interrogations *simultanées* ;
- Sur la qualité de la fonction d'évaluation des scores de pertinence des documents puisque seuls les quelques premiers documents sont consultés par les utilisateurs alors que des centaines et milliers, voire des millions, de documents peuvent s'apparier avec des requêtes *peu expressives*

3.2.1.3 Croissance de la taille des collections

La croissance de la taille des collections est le facteur important. En effet, nous sommes actuellement dans une phase de croissance *exponentielle* de la taille des collections interrogées en termes de nombre de documents accessibles. La taille de certains documents, ou plutôt multi-documents, comme les encyclopédies en ligne, nécessitent de faire apparaître de très nombreux granules d'informations (phrases, paragraphes, etc., mais aussi concepts, thèmes, etc.) qui ne se ramènent pas nécessairement à la page. Enfin, le nombre et le volume des lexiques, *thesaurus* ou ontologies manipulés augmentent également. Notons également, les influences sur les coûts en espaces de stockage occupés par de tels volumes d'information, qui demeurent d'actualité malgré la baisse des coûts des technologies de stockage, ainsi que les coûts de fonctionnement (ex. : consommation électrique).

Une incidence immédiate porte sur le temps d'évaluation des requêtes des utilisateurs puisqu'elle dépend en grande partie de la qualité et de la taille des index mis en œuvre. Dans ce contexte, on rapporte qu'un compromis généralement adopté entre la taille des collections indexées et la qualité des réponses en termes de délais de réponse a conduit de nombreux concepteurs de moteurs de recherche sur la Toile à limiter l'espace d'indexation à quelques pour cents seulement de la totalité de la Toile, de l'ordre de 20 % [2].

3.2.2 Compromis temps/espace

Dans la section précédente, deux facteurs identifiés concernent des problèmes de performances, liés aux volumes et aux temps de traitement respectivement. La problématique des grands volumes d'information est une réalité qui peut être effectivement appréhendée de ces façons-là. Elles ne s'excluent pas ; Or, ces facteurs entraîne un problème de temps de

calcul : temps pour indexer l'information et temps pour retrouver l'information. Formellement, il existe une relation fonctionnelle entre le temps et l'espace :

$$t = f(e) \tag{3.1}$$

Le volume des données influence directement les temps de traitement. Il est évident qu'un algorithme lent ne peut pas être appliqué sur une collection aussi vaste que celle affectée à un algorithme rapide. Un algorithme rapide rencontrera lui-même une frontière. Cette dernière pourra être repoussée si l'algorithme est parallélisable (et les données réparties). Elle pourra même être repoussée indéfiniment, du moins en théorie, si l'algorithme est extensible. Mais n'oublions pas qu'il doit aussi être incrémental, de nouvelles données venant s'ajouter régulièrement à la collection d'information.

Il nous faut lever la supposition toute simple et implicite de l'existence même d'un algorithme ! Certaines applications nécessitent une indexation en grande partie manuelle.
La problématique de grands volumes se révèle donc être tout autant un problème de volume que de temps de calcul, le premier influant directement et fortement sur le second mais n'étant pas le seul facteur de complexité.

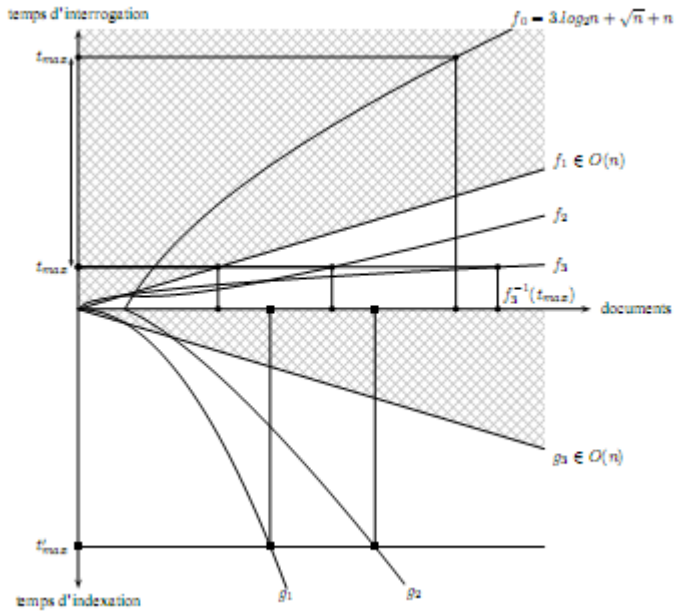

Fig.3.1 Compromis temps/espace et temps d'indexation / temps d'interrogation

Cela est traduit sur la figure 1 où l'on distingue :

- un axe horizontal correspondant à l'espace, c'est-à-dire à la taille des données (collections et objets) ;
- un axe vertical qui porte :
- dans sa partie supérieure le temps d'interrogation ;
- dans sa partie inférieure le temps d'indexation.

La fonction *f* qui lie le temps et l'espace n'est rien d'autre qu'une courbe de complexité asymptotique. Sur l'axe du temps, il est possible de placer des contraintes *t*max plus ou moins fortes :

3.2.3 Verrous technologiques et scientifiques

Au de-là des problèmes algorithmiques, le passage à l'échelle se heurte aussi à la « malédiction de la dimensionnalité » [173]. Ce terme traduit l'extrême difficulté, et à la limite l'impossibilité, de gérer efficacement et simultanément un très grand nombre de paramètres aussi bien en interrogation qu'en indexation. Si l'interrogation textuelle a pu éviter ce problème dans toute son ampleur, notamment en limitant arbitrairement le nombre maximum de termes intervenant dans une requête, ce n'est pas le cas de la recherche de données multimédias par le contenu où, de par le manque de sémantique associée aux méta-données, il est nécessaire de fournir systématiquement un très grand nombre de valeurs de descripteurs.

Dans les deux cas, cela entraîne à aborder peu ou prou non seulement les questions d'efficience et mais aussi d'efficacité d'un processus de recherche d'information [174].

3.2.3.1 Efficience

« Efficience : capacité de rendement, performance » (Larousse)

Sous l'angle de l'*efficience* (c'est-à-dire de la rapidité), des travaux montrent que le temps d'indexation moyen de la collection et le temps de traitement moyen des requêtes augmentent de manière très significative en fonction de la taille des collections ([175]**.** Cela est essentiellement dû, d'une part, à l'accroissement des index et donc de l'espace de stockage face à une évolution quasi-linéaire des ressources matérielles et, d'autre part, à une complexité trop importante des algorithmes d'indexation et d'évaluation de requêtes. Ce verrou a été abordé en proposant des solutions qui portent essentiellement sur la compression physique des informations, améliorant donc le premier point et par contre-coup le second ([176]**;** [177]**;** [178]). Cependant, le problème de l'échelle n'étant pas abordé dans sa globalité et dans sa diversité, l'impact sur l'efficience de la recherche reste peu significative ; l'amélioration des performances est réelle mais ne change pas la complexité asymptotique.

3.2.3.2 Efficacité

« Efficace : qui produit l'effet attendu [. . .] dont l'action aboutit à des résultats utiles » (Larousse)

Sous l'angle de l'*efficacité* (c'est-à-dire de la qualité des résultats), force est de constater que les travaux en recherche d'information ont peu considéré le paramètre du volume. La mesure classique d'évaluation des performances moyennant les taux de rappel et précision est en effet compromise dans un contexte où la procédure de sélection et de tri des documents pertinents agit sur un volume considérable d'informations. Cela signifie que l'ancien objectif « idéal » d'équilibre entre rappel et précision devrait s'orienter davantage vers de la *haute précision* de manière à cibler plus précisément les documents plus pertinents parmi les milliers et plus de documents candidats. Par ailleurs, les protocoles d'évaluation de la pertinence des résultats de recherche demeurent à ce jour indépendant de la taille et de la diversité des *corpus* de test. Cela engendre des biais d'évaluation qui peuvent être relativement conséquents lors de la comparaison des performances entre différents systèmes [179]. Ce verrou constitue l'enjeu d'une tâche récente, *TeraByte*, introduite à juste titre dans la campagne d'évaluation TREC (*text retrieval conference*, http://trec.nist.gov).

3.2.3.3 Pertinence Multivaluée

Pour les campagnes d'évaluation et de comparaison de SRI comme TREC, on utilise la technique du *pooling* pour former un ensemble de jugements de pertinence. Cette pertinence est généralement binaire [180]. Cependant [181], [182] proposent des cadres de classifications des notions de pertinence, suggérant que la pertinence est un phénomène social et cognitif complexe et qu'il n'y a pas une seule pertinence. Différents degrés de pertinence ont été étudiés dans des travaux antérieurs. Pour Kekäläinen et al. [183], dans les environnements larges et modernes de RI, il est désirable d'avoir des SRI qui retournent des documents en fonction de leur niveau de pertinence. L'évaluation en RI doit donc pouvoir "récompenser" les SRI qui retournent les documents ayant le plus haut niveau de pertinence en tête des autres documents. Pour ce faire, il est nécessaire de prendre en compte les différents niveaux de pertinence d'un document par rapport à un besoin d'information. Ces auteurs utilisent une collection avec une échelle à 4 niveaux de pertinence : (hautement pertinent, suffisamment pertinent, marginalement pertinent et non pertinent). Chacun de ces niveaux est quantifié par une valeur numérique et une des questions est le choix de ces valeurs et le sens dont elles peuvent être porteuses : exemple : le niveau de pertinence "hautement pertinent" a la valeur 3 et le niveau "marginalement pertinent" a la valeur 1 ; doit-on interpréter cela comme suit : "un document hautement pertinent est 3 fois plus pertinent qu'un document marginalement pertinent" ? Ces auteurs proposent les métriques *generalised non-binary recall and precision*(précision et rappel non binaire notées *gP* et *gR*) qui sont des extensions des métriques classiques (précision et rappel) prenant en compte plusieurs niveaux de pertinence : soit *R* l'ensemble des *n* documents retournés et appartenant à l'ensemble des documents ayant un niveau de pertinence pour une requête donnée, $R \subseteq D$; chaque document *di* a le niveau de pertinence *r(di)* qui est un nombre réel de l'intervalle [0..1], on pose $gP = Pd \in R\ r(d)/n$ et $gR = Pd \in R\ r(d)/Pd \in D\ r(d)$. Comme les métriques de précision et de rappel classiques, ces métriques permettent des moyennes sur l'ensemble des requêtes, des moyennes de précision à des niveaux de rappel, des courbes de performances. Pour des environnements où les niveaux de pertinence ne sont pas compris dans l'intervalle [0..1], ils peuvent être ramenés à cet intervalle par une normalisation (en faisant le rapport avec le plus grand niveau de pertinence par exemple).

Les métriques *Cumulative Gain et Discounted Cumulative Gain* sont également des métriques qui prennent en compte plusieurs niveaux de pertinence et proposées par Kekäläinen et al. [184]. Pour une collection donnée et un SRI, ces métriques calculent le gain cumulé d'information pertinente qu'on réalise au fur et à mesure qu'on parcourt la liste des résultats retournés par ce SRI sur cette collection ; dans le cas de la métrique *Discounted Cumulative Gain*, l'information pertinente à un rang donné est pondérée par une fonction décroissante du rang, avant d'être cumulée. Pour chacune de ces métriques, on obtient ainsi un vecteur qui donne pour chaque rang l'information pertinente cumulée du premier rang jusqu'à ce rang. Ces vecteurs peuvent être comparés à des vecteurs de gain d'information cumulée pour le cas d'un SRI idéal (SRI qui retourne les documents dans le meilleur ordre de niveau de pertinence, pour chaque requête).

Les métriques *Cumulative Gain et Discounted Cumulative Gain* ont été adaptées à l'évaluation des SRI qui travaillent sur des documents structurés (XML). Dans le cadre de la campagne INEX, la métrique XCG (Xml Cumulative Gain) est utilisée [185].

La prise en compte du volume d'information lors de l'évaluation des systèmes en termes d'efficience et d'efficacité a été la principale motivation de ce travail. Dans cette perspective,

l'objectif est tout d'abord de cerner les différentes facettes de cette problématique puis de dresser des pistes de réflexion qui abordent les verrous évoqués précédemment.

3.3 Influence du volume d'information sur le processus de la RI

Le processus de recherche d'information consiste à fournir, en réponse à une demande de l'utilisateur (requête) les documents qui répondent au mieux à son besoin d'information (documents pertinents). Il se décline en plusieurs phases : construction de la collection, indexation, interrogation et évaluation de la requête (décrits dans la figure 3.2) :

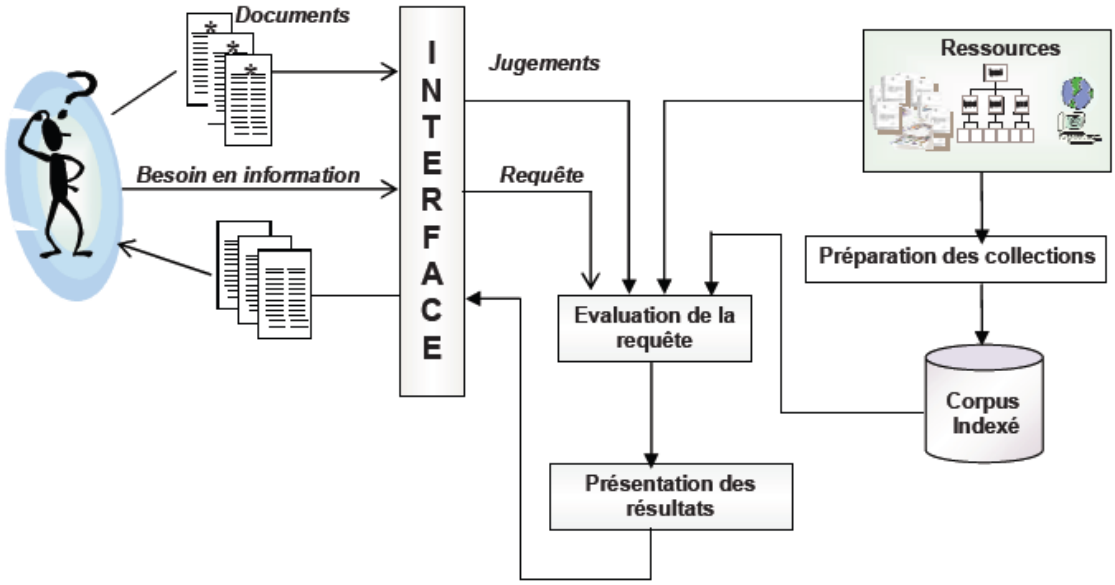

Fig.3.2 Processus de base de la recherche d'information

Imafouo et al. [**180**] donne une vue d'ensemble des travaux de RI associés à ces différentes phases qui se sont intéressés au passage à l'échelle. Pour ce qui concerne la phase d'évaluation, les travaux existants portent soit sur des métriques d'évaluation de SRI dans des collections volumineuses soit sur l'évaluation des performances des SRI quand la taille de collections augmente.

Concernant l'évaluation des performances de SRI face à la croissance de collections, les participants de la tâche *Very large Collection* (VLC) de TREC-6 ont noté une augmentation significative de la précision dans les premiers documents quand on passe d'un échantillon de la collection à la collection totale. Sur cette base, Hawking et al. [186] ont établi différentes distributions de scores de documents pertinents et de documents non pertinents et les échantillons analytiques de collection construites à partir de ces distributions de score confirment les observations sus-cités ; puis de façon expérimentale, la haute précision est observée sur différents types d'échantillons de taille croissante : une amélioration de la précision sur les premiers documents retournés quand la taille d'échantillons croît. Cette amélioration est justifiée par le nombre de documents pertinents plus élevé dans les échantillons plus grands, mais aussi dans la capacité du couple (requête, SRI) à classer les documents pertinents avant les documents non pertinents (puisque la précision diminue avec la taille de l'échantillon même dans le cas où elle n'est pas limitée par le nombre de documents pertinents présents dans l'échantillon) [180].

Les travaux de Imafouo et al. [180] fournissent une méthodologie d'échantillonnage d'une grande collection en sous-collections ayant des caractéristiques communes. Ainsi, l'étude de l'influence de la taille sur les propriétés est affranchie du biais des caractéristiques (parfois très différentes entre sous-collections). Ces travaux ont montré que le rôle que joue la collection de documents dans le processus d'attribution de score à un document donné a un impact sur les performances du modèle quand on passe à l'échelle. Ces travaux étendent également les résultats de Hawking et al [186] sur l'amélioration de la haute précision avec la croissance en taille de collections à plusieurs modèles de RI [180].

Pour étudier l'influence du volume de données sur le processus de la recherche d'information, il faut pouvoir observer les conséquences sur les différentes étapes du processus de RI, de l'étape de construction des collections, évaluation de requêtes, présentation des résultats, et d'en dégager les principaux enjeux. Ces phases constituent les dimensions selon lesquelles nous décrivons la problématique, des solutions (partielles) existantes ainsi que les perspectives envisagées.

3.3.1 Influences sur la construction des collections

La construction des collections est la première phase du processus de base de la recherche d'information consternant l'établissement d'un *modèle de documents* (voir la section 2.2.1.1): qui correspond à la modélisation du contenu sémantique des documents, dans un formalisme de représentation des informations. La de collections volumineuses pose un défi majeur lié fondamentalement à la grande dimension de l'espace de représentation des SRI sur le Web.

Pour les données textuelles, il a été constaté que l'accroissement du volume des collections engendre les défis suivants :

- l'accroissement des index conceptuels (vocabulaire des *corpus*) qui accentue les difficultés liées à la synonymie et à la polysémie [180];
- Une difficulté à maîtriser l'évolution des vocabulaires en fonction de la taille des collections [181] ;
- Une multiplication des zones textuelles accessibles telles que titre, résumé, paragraphe et par conséquent une gestion plus complexe des index [182].

Du point de vue physique, les problèmes qui en découlent sont principalement :

- L'augmentation considérable de l'espace de stockage des fichiers inverses ;
- Le débordement du cache de l'index de la mémoire vive vers le disque ce qui nécessite un nombre supplémentaire d'accès disque pour l'évaluation des requêtes et par conséquent une augmentation des délais de réponse ;
- La difficulté des mises à jour fréquentes des index, ces derniers étant généralement compressés.

3.3.1.1 Représentation conceptuelle de l'information

La représentation conceptuelle de l'information traduit la réduction de l'espace de représentation d'une collection en s'orientant davantage vers la représentation des documents à l'aide de concepts agrégés que sur des unités d'information plus fines telles que les mots ou *n*-grammes. En effet, dans le cas de collections croissantes, la matrice de base termes/documents est de dimension très élevée et de surcroît creuse [2]. Une étude réalisée sur

la distribution des termes dans un échantillon de collections volumineuses révèle que ces matrices ont un taux de densité moyen évalué à seulement 1 % [187]. L'un des premiers modèles proposés dans une perspective de réduction de l'espace de représentation est le modèle LSI (*latent semantic indexing*) [188]. La base mathématique du modèle LSI est la décomposition en valeurs singulières de la matrice termes / documents. Cette décomposition identifie une base vectorielle de dimension plus réduite qui couvre le même espace vectoriel associé que celui de la représentation initiale des documents, obtenue par application d'opérations algébriques (rotation, factorisation) sur les vecteurs d'origine. On trouve des extensions au modèle de base ([189]; [190]), notamment des techniques appropriées à la gestion de matrices creuses non-régulières, traduisant une disparité dans l'occurrence des termes d'indexation dans les collections; elles sont essentiellement basées sur la révision du processus de décomposition [191] et de minimisation de la trace [192]. D'autres méthodes pour la réduction de l'espace de représentation basées sur la factorisation F de la matrice termes / documents ont été développées [193].

En ce qui concerne les données multimédias, la « malédiction de la dimensionnalité » est particulièrement présente. Elle est liée, d'une part, au très grand nombre de descripteurs de bas niveau que l'on peut extraire des images, bandes audio et vidéo et, d'autre part, aux dimensions importantes de la plupart de ces descripteurs (histogrammes, transformées de Fourier, attributs cepstraux, etc.). Leur taille pose des problèmes difficiles de réduction de la dimensionnalité : conservation des métriques, perte d'information et apparition d'artefacts (ce qui peut entraîner un mauvais regroupement des objets sur de pseudo caractéristiques communes importantes). Ces problèmes ont été largement explorés (ACP – analyse en composantes principales –, SOM – *selforganising maps* –, SVM – *support vector machine* –, etc.) Mais sont réactualisés par la dimension croissante des objets mesurés. Ces problèmes de coût de calcul et de précision (densité des données, espace creux) sont partagés par les techniques de classification et d'indexation. Les données multimédias posent également des problèmes difficiles de conceptualisation de l'information de bas niveau en des termes intelligibles par un être humain – autrement que par la visualisation et l'interactivité. Par exemple, si les notions liées à la couleur sont assez facilement transposables dans le langage naturel (« rouge », « bleu », « clair », « vif », etc.), en revanche les nombreux descripteurs de texture, visuelle ou sonore, ne sont pas aisés à conceptualiser. Enfin, il est utile de définir précisément ce que l'on entend par recherche d'information multimédia ; les descripteurs et les métriques varient beaucoup en fonction des applications visées (notamment dans le domaine de la reconnaissance des formes) [2].

3.3.1.2 Problématique d'explosion du contenu multimédia

Dans un système typique de recherche par le contenu dans les bases multimédia (ou CBR – *Content Based Retrieval*) [194] les caractéristiques des données multimédia (images, vidéo ou audio) sont extraites de la base et sont représentées par des vecteurs multidimensionnels appelés *signatures*. Ces signatures peuvent être de différente nature selon l'application considérée : elles peuvent être globales, i.e. calculées sur la totalité de la donnée multimédia (ex. couleur, texture ou forme pour les images), locales, i.e. résumant le contenu à certaines parties jugées pertinentes de la donnée multimédia (ex. les points d'intérêt pour les images ou la vidéo), ou spécifiques à une catégorie d'objets (ex. les descripteurs de visage pour la reconnaissance de visage). Lorsque plusieurs objets sont présents et décrits, il est aussi très pertinent de décrire les relations qui existent entre eux, comme par exemple les relations spatiales existants entre les objets d'une image [195] ou les relations temporelles entre les événements d'une donnée audio.

Lorsque le volume de données multimédia est grand, beaucoup d'approches de description souffrent d'un problème bien connu en bases de données, *la malédiction de la dimension*, rendant la recherche d'information difficile dans un grand volume de signatures multidimensionnelles [196]. Par exemple pour les images, des approches évoluées telles que les méthodes de description locales ou des relations spatiales entre objets ont une complexité connue qui requiert la mise en place de structures d'index multidimensionnelles [197] pour rendre acceptables les temps de recherche dans les espaces de description associés.

3.3.1.3 Problématique de l'accélération de la recherche dans les espaces de description du contenu visuel.

Deux types de méthodes permettant la structuration des contenus visuel : les approches de structuration hiérarchiques, par exemple les arbres quaternaires (ou *quadtree*) [198] [199], et les méthodes de description des relations spatiales entre objets [200][201][202]. Plus précisément, cette association devrait permettre :

- 1) la mise en place d'une nouvelle approche de description des relations spatiales plus efficace mais aussi,
- 2) l'exploitation de la structuration hiérarchique des relations spatiales pour l'appariement d'images dans le cadre du paradigme de la recherche progressive [203] pour accélérer la recherche dans les grands volumes de données, de la façon suivante :

3.3.1.4 Compression physique de l'information

La compression physique de l'information comprend l'ensemble des techniques permettant d'améliorer l'efficience en recherche d'information en se basant sur la réduction de l'espace de stockage des structures physiques représentatives des documents, plus particulièrement les fichiers inverses. Les approches standard pour la construction de l'index conviennent davantage pour de petites collections où le vocabulaire et les fichiers inverses résident en mémoire vive, permettant ainsi une construction rapide de l'index. Le passage à l'échelle engendre l'accroissement du nombre de termes et donc de l'index, l'allongement des listes inverses et l'augmentation de l'espace de stockage. Partant, l'index n'étant plus en mémoire vive, le temps d'évaluation des requêtes devient bien plus important en raison des accès disque induits. L'une des solutions apportées est la compression des fichiers inverses qui a pour principaux avantages [204] :

- la réduction de l'espace de stockage ;
- une meilleure rentabilité des outils de communication (pour des schémas de compression efficaces, le temps de compression ajouté aux temps de transfert et de décompression est inférieur au temps de transfert des données non compressées) ;
- la réduction du nombre moyen d'accès disque puisque la probabilité d'avoir une partie de l'index en mémoire vive est plus importante.

La plupart des techniques de compression se basent sur la représentation des différences, appelées *gaps*, entre séquences ascendantes d'entiers représentant les adresses de documents où apparaissent chaque terme de l'index. En effet, les différences sont plus petites que les adresses absolues et donc représentables sur de plus petits formats. On distingue principalement trois classes de méthodes [205] :

- le principe le plus simple est celui des méthodes de compression à longueur fixe qui associent à chaque plage de valeurs fixes le nombre de bits nécessaires pour sa représentation ;
- les méthodes de compression à longueurs variables consistent à déterminer la longueur du code comme une fraction de sa valeur. La méthode la plus représentative est le code de Elias [206] qui représente un entier x avec $2\log_2 x + 1$ bits ;

$$\sum_{j=1}^{k-1} v_j < x \leq \sum_{j=1}^{k} v_j$$

Puis $$d = x - \sum_{j=1}^{k-1} v_j - 1 \qquad (3.2)$$

Le vecteur v peut être modifié pour avoir différents schémas de compression ajustables en fonction des collections. Une solution élégante [207] consiste

-à faire varier le vecteur V pour chaque liste inverse en posant : $v = (b, 2 \times b, 4 \times b, \ldots, 64 \times b)$ où b est la valeur médiane dans la liste inverse.

En pratique, on montre que des taux de compression appréciables sont atteints en adoptant le code de Golomb pour la représentation des *d-gaps* et des fréquences des termes dans les documents, et le code d'Elias pour les positions des termes dans les documents [206]. Une méthode de compression basée non plus sur l'ordre des identifiants de documents mais sur leur similarité a été proposée récemment [208]. L'algorithme génère un ordre de réaffectation des identifiants de documents en fonction de leurs similitudes puis calcule les *d-gaps* qui sont ainsi plus faibles puisque associés à des séquences de documents de similitudes graduelles.

Les résultats obtenus sur de larges collections montrent que la valeur moyenne du *gap* peut être réduite de 30 % et le taux de compression amélioré de 15 %.

3.3.1.4 Perspectives

Les solutions proposées au problème de la représentation de grands volumes d'informations sont, à notre connaissance, parcellaires : peu de travaux couvrent le niveau conceptuel tandis que la majorité des travaux se sont focalisés sur des aspects d'ordre physique (espace de stockage, nombre d'E/S, etc.). Ces travaux ont pu améliorer les performances de systèmes de recherche d'information mais ils sont insuffisants pour faire face à la croissance des collections.

3.3.1.4.1 Échantillonnage de collections

Il serait intéressant d'aller vers des solutions de représentation des documents qui regroupent les différents niveaux d'abstraction depuis le niveau conceptuel jusqu'aux détails physiques en passant par un contexte d'architecture fonctionnelle. L'objectif visé est d'intégrer dans un même modèle les contraintes des différents niveaux afin d'en déduire des règles de conception d'unités d'information et d'échantillonnage des collections documentaires :

- Construire une collection sur laquelle une caractéristique donnée est la même quelle que soit la portion choisie.
- Découper la collection en sous-collections de taille croissante qui sont des « échantillons» de la collection initiale

- Construire des collections de taille raisonnable qui soient des échantillons *représentatifs* de collections plus volumineuses (Web, ensemble de documents sur un domaine donné,...).
- Étudier le comportement des modèles de RI sur chaque sous-collection et analyser l'influence de la taille
- Reproduire sur une *petite* collection les propriétés (à étudier) d'une collection plus volumineuse.
- Expérimenter sur la *petite* collection et reporter les résultats sur la grande.

Dans le but de remédier à ces difficultés, des améliorations de l'échantillonnage ont été proposées [221]. On cite particulièrement la méthode ISJ (*XXX*) qui utilise un système de recherche d'information interactif pour la sélection des documents à juger et la méthode *Move To Front* basée sur la sélection, pour chaque système participant, d'un nombre de documents dépendant de ses performances.

3.3.1.4.2 Granules documentaires

Une première réflexion à envisager dans les travaux futurs porterait sur la granularité de l'information. En effet, la granularité d'un document joue certainement un rôle, mais la recherche d'information textuelle manipule maintenant depuis longtemps les documents en texte intégral. Un document n'est donc qu'une très importante séquence (ou seulement ensemble) de termes.

Mais chercher à « granulariser » des documents multimédias (images, audio et vidéo) présente de nombreux problèmes [2]. :

- 1) Le premier d'entre eux est qu'il est difficile, voire impossible, de cerner le granule. Dans le cas idéal, encore lointain, les éléments isolés feront sens pour l'être humain. Pour des images, ce sera l'extraction de chacun des objets du monde réel (voiture, humain, animal, végétal, etc.). Ce problème n'est pas inexistant dans le document textuel mais la liaison entre le signifiant et le signifié est considérablement réduite puisque le langage, écrit ou parlé, a été conçu dans ce but.
- 2) Le second problème, lié au premier, est celui de l'emboîtement des descriptions. Ce problème est analogue aux clauses dans les données textuelles et plus généralement aux *thesaurus*. Un élément identifié et nommé peut être associé, le plus souvent de manière hiérarchique mais pas seulement, à d'autres éléments. Par exemple, sur une image on pourra apercevoir un être humain, vu de face, et plus précisément son buste, c'est-à-dire tête, poitrine et partie supérieure des bras. Dans une vidéo, cette information dépend du temps, les éléments apparaissant, se modifiant et finissant par disparaître, brutalement ou progressivement.

- 3) Le premier problème entraîne aussi celui de l'agrégation de descriptions de bas niveau. Etant actuellement incapables, dans toutes les situations, de trouver du sens au signal, on tente de le cerner par une combinaison de descripteurs plus ou moins fiables, plus ou moins corrélés.

- 4) Enfin, les descripteurs proposés et retenus sont bien plus coûteux que les termes d'un document textuel, aussi bien en termes de temps d'extraction, de stockage, que de temps de recherche.

Pour toutes ces raisons, il semble préférable d'examiner des pistes de recherche autour de la classification, le « granule » documentaire étant alors la classe.

3.3.2 Influences sur l'évaluation des requêtes

L'accroissement des volumes des collections a des conséquences évidentes sur l'étape d'évaluation des requêtes tant pour l'efficience que pour l'efficacité.

3.3.2.1 Impact de volume d'information sur l'efficience

L'impact volume d'information sur l'efficience est lié à la complexité en temps de la fonction d'appariement. Cette complexité est cruciale pour le bon fonctionnement d'un système de recherche d'information. En effet, le temps de réponse à une requête doit toujours rester très court (quelques secondes au maximum), *quelle que soit* la taille du *corpus*. Or, évaluer une requête sur l'ensemble des documents engendre un traitement qui pourrait devenir quadratique (c'est-à-dire impossible à réaliser en pratique), et cela indépendamment du modèle d'appariement deux à deux qui peut lui aussi prendre un temps considérable (comme la projection du modèle des graphes conceptuels). De même, l'étape finale de l'appariement, qui est le classement suivant le RSV (*rank score value*) décroissant, peut être problématique. Il est indispensable de limiter très rapidement le sous-ensemble de documents sur lequel va porter un tri. Par exemple, le modèle vectoriel évite tout tri explicite : il exploite une heuristique de classement décroissant des documents vis-à-vis de chaque terme pour en dériver un classement décroissant des documents vis-à-vis de la conjonction des termes, réputé suffisamment fiable. De manière générale, il convient de transférer aussi largement que possible la complexité en temps dans la phase d'indexation de la base.

3.3.2.2 Impact de volume d'information sur l'efficacité

L'impact volume d'information sur l'efficacité est lié à la quantité de documents qu'il faut classifier selon leur pertinence vis-à-vis de la requête. Intuitivement, il est plus facile d'identifier les documents pertinents dans une collection comportant quelques centaines de documents plutôt que dans une collection comprenant des milliards de documents, car le risque d'erreur de classification est statistiquement plus faible dans le premier cas. Les résultats obtenus dans les campagnes TREC confirment ce point [171]. L'hétérogénéité est plus importante dans les collections volumineuses et les descriptions statistiques ne sont plus aussi discriminantes. Par exemple, la conjecture de Luhn suppose que le pouvoir de résolution d'un terme, c'est-à-dire sa capacité de discrimination entre un document pertinent et un document non pertinent, est relatif à une fréquence documentaire moyenne. Cette conjecture est à revoir dans le cas de *corpus* de très grandes dimensions. De même, il semble que l'hypothèse qui prévaut à l'usage de la fréquence documentaire inverse, ne soit pas valide sur de grandes collections car l'influence du facteur *idf* (*inverse-document frequency*), correspondant au nombre de documents où un terme apparaît, va en diminuant avec la taille des collections [209]. On peut également noter l'absence du modèle vectoriel dans les moteurs de recherche sur la Toile, et son omniprésence dans les expérimentations de recherche sur des *corpus* de taille plus faible. Néanmoins, il faut aussi noter que la valeur exacte de la fréquence des termes semble peu influencer les résultats. L'objectif de *haute* précision que nous avançons, qui revient à privilégier la précision des premiers documents sélectionnés, implique des traitements plus sophistiqués pour mieux discriminer les (parties de) documents. De nouveaux algorithmes d'appariement, et de pré-appariement dans la phase d'indexation, restent encore à concevoir.

Pour ce qui est des données multimédias, aux problèmes déjà évoqués s'ajoutent deux autres difficultés. Tout d'abord :

- 1) la diversité des méta-données extraites,
- 2) l'apparition de nouveaux descripteurs ainsi que
- 3) les différences de descriptions dues à des sources variées de données multimédias ou à la non-application de certains traitements coûteux sur certaines données qui rendent plus délicat le processus d'appariement et peuvent donc détériorer la qualité des réponses. Deuxièmement, les techniques d'appariements qui ont été utilisées jusqu'à maintenant (en l'absence de collections volumineuses) s'appuient essentiellement sur des calculs de distances qui se traduisent en substance par des parcours complets des collections de méta-données.

Même en ayant procédé à une classification préalable des méta-données, des sélections heuristiques semblent d'ores et déjà nécessaires ou, de manière plus ambitieuse, la définition de nouvelles métriques plus économiques en temps de calcul au moment de l'interrogation [2].

3.3.2.2 Propositions existantes de réduction de l'échelle

Les propositions de réduction de l'échelle ont porté essentiellement sur la parallélisation des algorithmes de recherche d'information, l'optimisation des accès aux fichiers inverses [176] et la réduction de la complexité des algorithmes proposés dans les modèles classiques de recherche d'information [210].

Le traitement parallèle de grands volumes de données en recherche d'information possède de nombreux avantages comme l'amélioration du temps de réponse et la capacité à effectuer des recherches dans des collections volumineuses. L'index inversé est l'approche la plus populaire pour les moteurs de recherche textuels. Bayley propose une approche basée sur les fichiers inverses et les dictionnaires. Il a testé son approche sur le *corpus* TREC (3 giga-octets). Cependant, la parallélisation des index inversés n'est pas une tâche aisée. Ainsi, le traitement parallèle de Jain ne s'appuie pas sur ces fichiers inverses mais sur l'approche CSR (*compressed sparse row format*).

Les fichiers inverses constituent le coeur de la majorité des algorithmes de recherche d'information. Dans un fichier inverse, un terme est associé à un ensemble de documents qui contiennent le terme. Lorsque le nombre de documents augmente, il en est de même pour la liste des termes uniques et les fichiers inverses. Ainsi, le temps de traitement d'une requête croît approximativement linéairement avec la taille de l'ensemble de documents (et croît approximativement linéairement avec la taille de la collection de données indexées). Dans la plupart des méthodes de recherche d'information, la première étape consiste à générer l'ensemble de tous les documents qui contiennent les termes recherchés. Malheureusement, de nombreuses approches en recherche d'information croissent plutôt de manière exponentielle. Une solution pour réduire l'échelle consiste donc à optimiser les accès à ces fichiers inverses. En effet, la taille des fichiers inverses peut être aussi grande que les textes qu'ils indexent. La compression des fichiers inverses permet de réduire la taille de 80 %.

La réduction de la complexité des algorithmes est principalement basée sur l'approche à base de fichiers de signatures. Il s'agit d'un mécanisme de filtrage qui élimine les blocs de textes qui ne peuvent pas répondent à une requête.

3.3.2.4 Perspectives

En conclusion, les éléments de solution liés au traitement des requêtes sont de plusieurs ordres :

- déporter autant que possible les traitements coûteux dans la phase d'indexation ;
- remplacer les traitements coûteux qui persisteraient dans la phase d'appariement par des heuristiques ;
- élaguer les traitements en tenant compte d'autres facteurs comme l'utilisateur, l'usage et le niveau d'abstraction des informations ;
- revoir les modèles de RI et notamment établir de nouvelles mesures, les descripteurs statistiques « standard » devenant trop peu discriminants dans des collections volumineuses hétérogènes, incomplètes et multimédias [2].

3.4 Visualisation des résultats

Les résultats issus d'un moteur ou encore d'un méta-moteur de recherche d'information sont communément présentés sous la forme d'une liste (liste d'URLs).

Cette liste présente les différents résultats au travers :

- d'un numéro de classement ou une appréciation de la pertinence système,
- d'un nom ou d'une URL associé à un lien hypertexte pour permettre à l'utilisateur d'accéder au document,
- d'un court descriptif présentant les premières lignes du document dans lesquelles apparaissent les termes de la requête.

Cet affichage est facile à mettre en œuvre et à utiliser mais n'est efficace que pour un nombre réduit de résultats (< 20) [211]. Pour un nombre important, les listes de résultats souffrent essentiellement des limites suivantes [212], [213] :

- la position d'un document dans la liste ne permet pas explicitement de déduire sa similarité avec les autres documents de la liste,
- l'utilisateur peut avoir du mal à comprendre pourquoi le document a été inséré à cet endroit dans la liste et quelle est la relation avec la requête sans avoir visualisé son contenu.

Á cela s'ajoute le fait que les résultats sont affichés page par page, ce qui ne facilite pas la vision globale du résultat. Pour obtenir une vision globale des résultats, l'utilisateur doit visiter chacun des documents, un à un, pour en apprécier la réelle pertinence et identifier les liens potentiels entre ceux-ci. De ce fait, il est compréhensible que l'utilisateur se contente en moyenne des 30 premiers résultats en occultant le reste des résultats (éventuellement pertinents) si le nombre de résultats est très important. Cependant, il occulte également un ensemble de documents potentiellement pertinents pour ses besoins.

C'est pour remédier à cet état de fait que des travaux ont été réalisés dans le domaine des interfaces de visualisation. Leur but est de proposer à l'utilisateur un moyen efficace pour apprécier et manipuler les résultats de recherche d'information dans leur globalité.

Diverses visualisations, que ce soit en mode texte, sous forme de représentation graphique dans un espace à deux ou trois dimensions, tentent de pallier aux limites des listes de résultats. Diverses classifications des interfaces de visualisation ont été proposées comme dans [213]

qui traite des visualisations pour la RI ou encore [214] et [215] qui effectuent une taxonomie des différentes techniques de visualisation générale d'informations.

Il est bien clair que la problématique des grands volumes d'informations influence directement la manière de présenter les résultats retrouvés sur l'écran de l'utilisateur. En effet, la taille des *corpus* dans un système de recherche d'information rende plus difficile l'évaluation et donc la perception des bonnes réponses dans un espace devenu plus vaste. De manière générale, les travaux sur la visualisation de grandes masses de données ont été réalisés principalement dans le domaine des interfaces homme-machine ([216]**;** [217]) même s'il y a des travaux spécifiques aux systèmes de recherche d'information ([218]**;** [219]). Les études liées à la présentation de grandes masses de données doivent donc s'adapter à la recherche d'information.

Nous présentons dans ce qui suit une revue basée sur la taxonomie classique en visualisation : une dimension, deux dimensions et trois dimensions, de la visualisation des résultats d'une recherche. De manière générale, il s'agit en premier lieu de présenter les documents proposés par le système en même temps qu'un ordre de pertinence. D'autres éléments annexes peuvent être proposés en réponse comme une partie du contenu de la représentation interne des documents.

Cette analyse est issue d'une étude spécifique sur les interfaces pour la recherche d'informations [220].

3.3.4.2.1. Visualisation en une dimension

La forme la plus simple et aussi la plus courante de présentation des résultats est la liste, en une dimension. Cette dimension correspond généralement à l'ordre de pertinence calculé par le système. Cette présentation, souvent textuelle, est celle employée par tous les moteurs de recherche avec des variantes consistant à regrouper sous une arborescence, donc à l'aide d'une deuxième dimension, les documents appartenant à un même site.

Cette présentation est celle adoptée par la quasi-totalité des moteurs de recherche sur la Toile. Néanmoins, elle limite fortement la quantité d'informations présentables en même temps. Lorsque l'on veut afficher plus d'informations simultanément, il faut utiliser deux dimensions.

3.3.4.2.2. Visualisation en deux dimensions

La représentation en deux dimensions permet de représenter beaucoup plus d'informations que l'ordre de pertinence entre les documents. La représentation peut être l'occasion de laisser percevoir le *corpus*. On peut, par exemple, donner l'impression de la quantité de réponses, donc la taille du *corpus* pour un terme (*starfield*). Cette perception s'apparente à la notion de rappel. Contrairement au simple chiffre exprimant l'ordre de grandeur du nombre de réponse, une interface comme TIAPRI, permet de percevoir l'importance d'un terme dans l'ensemble des documents en réponse. Il y a également un couplage direct entre la requête et les réponses car l'importance d'un terme dans la requête est visible dans l'interface et a une influence sur la visualisation des résultats.

Dans ce genre de représentation, il y a perte de l'information textuelle (titre, auteur, aperçu du contenu, etc.), au profit d'une perception plus globale. Il existe néanmoins des techniques

visuelles permettant d'accéder momentanément au contenu du texte associé à la réponse comme les techniques des lentilles magiques. Le type de média influence également les possibilités de représentations efficaces de grandes quantités d'informations. Dans le cas des images, l'affichage des résultats en deux dimensions dans une matrice d'images est plus efficace que le texte, car il est plus facile de percevoir le contenu d'une image, même réduite, qu'un texte en petit format.

Pour augmenter le nombre d'images perçues, il existe des techniques de compressions visuelles comme la vue « en œil de poisson », qui couplée avec la navigation, permet de percevoir et de manipuler un très grand nombre de résultats simultanément. Par exemple dans le système « ostensif » de Campbell, le système présente en grand format l'image en cours de sélection, et présente dans un graphe compressé d'images, le parcours de l'utilisateur, et donc toutes les images vues, ainsi qu'une nouvelle sélection d'images proposées à l'utilisateur pour poursuivre son exploration. Le système est dit ostensif car aucune requête ne lui est explicitement posée ; elle est déduite de son comportement.

3.3.4.2.3. Visualisation en trois dimensions

L'utilisation de trois dimensions nécessite une représentation en deux dimensions sur la surface de l'écran. Pour permettre à l'utilisateur de se construire une représentation 2D de cette image 3D, les interfaces se basent sur une représentation *métaphorique* du monde réel. Par exemple, le système LyberWorld [216] met en place la métaphore du cône. Un cône représente un ensemble de documents groupés autour d'un thème. Ces cônes sont ensuite organisés en arbres selon la structure hiérarchique des thèmes des réponses. L'interaction se fait par manipulation directe sur les cônes, par translation et rotation. Ce système intègre également la métaphore du « paysage ».

3.5.4 Techniques de visualisation adaptées au volume

Les techniques de visualisation des résultats issus des grands volumes d'informations influence la satisfaction de la recherche des informations. Cette activité de recherche est également dépendante de l'évolution des usages et des techniques. Dans ce contexte, les travaux doivent étudier [2] :

- l'adaptation des techniques de visualisation de grandes quantités d'informations ;
- l'influence du passage à l'échelle sur l'interface de sortie en relation avec une prise en compte de l'utilisateur et de son besoin dans un modèle de recherche d'information donné, comme le modèle ostensif ;
- les apports dans l'évolution des technologies de visualisation, la variété des possibilités d'affichage des résultats demandant la conception d'interfaces plus « plastiques» [222] et capables de mieux gérer la multi-modalité, en entrée (une requête à partir d'une photo) comme en sortie ;
- le développement de l'informatique ubiquitaire (ordinateurs ultra-portables, *palm-top*, *wi-fi*, etc.) qui engendre de nouvelles réflexions et de nouvelles pistes de recherche, comme la recherche d'information *située*, c'est-à-dire fonction du lieu et du moment où s'effectue une recherche.

3.3.4.3. Perspectives

Les perspectives dans la visualisation des résultats issues d'une étude dans [2] sont à examiner selon les trois axes suivants :

- Technologie de visualisation : l'accès à l'information est dépendant de l'artefact technique qui met en relation l'utilisateur avec l'information. L'écran, le clavier et la souris sont toujours les moyens les plus utilisés pour accéder à l'information. L'évolution technologique va dans le sens de davantage de mobilité, vers une informatique omniprésente et multimédia. Les techniques de visualisation seront dépendantes de cette évolution.

- Technique de visualisation : l'accès à de grandes quantités d'information pose le problème de leur perception par l'utilisateur. En premier lieu, le passage à l'échelle influence directement le nombre de réponses potentielles à une requête et pose donc le problème de la navigation parmi ces réponses. Par ailleurs, l'utilisateur doit être capable de s'assurer que l'information proposée par le système est bien pertinente, non pas dans l'absolu mais par rapport au *corpus*. Dans ce cas également, une perception du contenu potentiel peut l'aider à établir son propre jugement de pertinence.

- Couplage entre interaction et modèle de recherche d'information : le modèle ostensif est un exemple d'influence mutuelle entre un modèle de recherche d'information et la technique de navigation dans les documents. Si de nouveaux modèles deviennent nécessaires pour permettre un passage à l'échelle, alors une étude de leur influence sur la présentation des résultats ainsi que sur la manière dont l'utilisateur pourra communiquer son besoin d'information est également nécessaire. Ce dernier point évoque le domaine connexe à la visualisation qui est celui de l'« expressibilité » de la machine et du besoin de l'utilisateur en une communication réellement multimodale : voix, image, gestuelle.

3.6 Conclusion

La quantité d'information numérique produite et consultée a considérablement augmenté et sa diversité s'est accrue. Or, la problématique des grands volumes d'informations joue un rôle important dans la quantité et la qualité des informations perçues par l'utilisateur et issu des SRI. Dans ce chapitre, nous avons présenté un panorama des problèmes qui découlent de ces évolutions ainsi que quelques pistes de réflexions afin de répondre à ce qui semble bien être un défi de « *la recherche d'information volumineuse*».

Nous avons visé à identifier les verrous technologiques liés à la croissance et à la diversification des contenus des SRI et aux processus de recherche d'information associés, et à dresser un bilan prospectif permettant de s'affranchir, en partie, des verrous posés par la problématique des grands volumes d'informations en RI dans le Web. Afin de répondre à cette double problématique, nous avons présenté les facteurs à l'origine de la croissance de WWW, une caractérisation sous l'angle des dimensions *espace/temps* ainsi que les verrous technologiques et scientifiques engendrés. Accompagné par l'influence sur le processus classique de recherche d'information. Plus précisément, nous avons abordé les contraintes imposées par la problématique des grands volumes d'informations sur les phases de construction des collections, d'évaluation des requêtes et de visualisation des résultats d'une recherche.

Le développement de *l'informatique ubiquitaire* (ordinateurs, PDA, *wi-fi*, etc.) qui engendre de nouvelles réflexions et de nouvelles pistes de recherche, comme la recherche d'information adaptée et située, c'est-à-dire fonction du lieu et du moment où s'effectue une recherche. La problématique des grands volumes d'informations en recherche d'information pointe avec précision l'objectif de la contextualisation de la RI dans le Web et en particulier, dans le Web ubiquitaire. La RI sensible au contexte quant à elle pose des problématiques nouvelles allant de la modélisation du contexte jusqu'à la modélisation de la pertinence cognitive en passant par la modélisation de l'interaction entre un utilisateur et un SRI sensible au contexte.

En effet, la recherche d'information pertinente parmi un volume important d'information, adaptée aux besoins spécifiques d'un utilisateur donné, à tous moment et avec n'importe quel dispositif mobile, devient à la fois difficile et nécessaire. Difficile d'abord en matière de surcharge du réseau, et aussi en matière de changement dynamique de contexte d'utilisateurs. En clair, le problème n'est pas tant la disponibilité de l'information mais sa pertinence relativement à un contexte d'utilisation particulier.

Chapitre 4

Une architecture de recherche basée agent mobile et rôle

4.1 Introduction

La recherche d'information distribuée est une tache très complexe dans le *Web ubiquitaire*, où le système est chargé de fournir l'information à n'importe quel moment, n'importe où et avec n'importe quel dispositif mobile utilisé par l'utilisateur. Ce domaine est l'un des domaines d'applications potentiels des agents mobiles, où un gros volume de données peut être accessible à travers différents réseaux hétérogènes et l'information est dispersée sur plusieurs serveurs géographiquement séparés. Surtout, avec la récente explosion des applications *Web ubiquitaire*, nous assistons à l'émergence de nouveaux modèles pour la structuration d'applications réparties et le calcul mobile.

Pour combattre la complexité de la structuration des systèmes de recherche d'information distribuée dans le Web ubiquitaire, Les agents mobiles sont particulièrement attrayants, par la migration aux serveurs d'informations, un agent peut effectuer une opération de recherche localement, en éliminant le transfert de données intermédiaires sur le réseau, mais aussi, l'agent mobile, peut déplacer et utiliser des rôles de recherche spécifique au sein des serveurs.

Le plus important, est qu'un agent mobile peut choisir différentes stratégies de déplacement de comportement et de recherche selon sa tâche et les conditions de réseau courant, et puis change ses stratégies pendant que l'état du réseau change. Ainsi, des comportements complexes, efficaces et robustes peuvent être réalisés avec étonnamment un minimum de code [223].

En effet, les caractéristiques technologiques de ce type d'agent permettant de le considérer comme une solution adaptable pour affronter la dynamicité massive du contexte de l'environnement de son exécution et particulièrement, pour la recherche d'information sensible au contexte sur le Web ubiquitaire.

Nous utilisons l'agent mobile comme outil de génie logiciel et plus précisément, Nous présentons un modèle d'agent mobile à base de rôle adaptable, sensible au contexte dynamiquement pour s'adapter aux variations de son environnement de recherche ubiquitaire. En fait, ce modèle d'agent nourri des travaux qui ont été réalisés dans ce domaine et qui ont des succès notable dans le domaine de la recherche d'information. Le fait que l'agent se voie comme étant un composant, le comportement de l'agent mobile proposé est construit principalement à partir d'un assemblage de rôles de granularité faible, il s'agit, en fait de micro-composants remplaçables et spécialisables. Ce style d'architecture d'agent flexible doit faire face à la forte dynamicité de son environnement ubiquitaire de recherche, dans lequel, différentes stratégies d'assemblages de rôles permettent d'engendrer un comportement sensible aux contextes de la recherche.

Dans la section 2, Nous allons présenter les différentes technologies à la base de la construction d'applications *Web ubiquitaire sensible au contexte*: intergiciel, composant logiciel, agent mobile. Nous allons mettre en lumière leurs apports à la problématique ainsi que leurs limites, au travers de quelques exemples concrets. Nous préconisons que ces différentes technologies ne sont pas suffisantes si elles sont employées individuellement pour répondre aux différents besoins exprimés dans les chapitres précédents, notamment, dans la recherche contextuelle et la recherche d'information volumineuse. En effet, une adaptation de ces technologies peut apporter une avance pour concevoir une architecture de recherche sensible au contexte fortement dynamique dans le *Web ubiquitaire*.
Ce panorama technologique sert de pré-requis pour la compréhension de notre architecture proposée.

Dans la section 3, nous présentons un modèle d'agent mobile de recherche comme une solution qui s'appuie sur une combinaison de technologies : agents mobile, composants, intergiciels sensible aux contextes, inspirations du travail des fourmis, notamment, la déposition de la phéromone sur leur passage aux différentes sources de nourriture.
La section 4 est consacrée à la présentation détaillée de l'architecture proposée avec la spécification des différentes taches d'agents. La présentation sera complétée par une description du processus de la recherche ubiquitaire locale sous forme d'un diagramme de flux. Et en fin, on conclut dans la section 5.

4.2 Technologies logicielles

Ce panorama technologique est un état de l'art partiel qui sert de pré-requis pour la compréhension de notre architecture proposée.

4.2.1 Émergence de la technologie agent

L'agent n'a qu'une vue partielle du problème, en ce qu'il est incapable à lui tout seul de lui résoudre, soit par manque de moyen, soit par manque d'information et soit par besoin de coordination et coopération avec d'autres agents. Ainsi, la solution apportée par les SMA émerge de l'ensemble du groupe d'agents.

L'interaction entre les agents stimule cette émergence. Par exemple, dans le cas où un agent modifie son environnement à des fins d'interactions, un autre agent peut se retrouver stimulé par l'observation des modifications et adapter son comportement de telle sorte que l'effet global amenant à la résolution du problème s'en trouve amélioré. Comme dans le cas d'utilisation de la phéromone par les colonies de fourmis. Qui consiste en fait, en la stimulation d'un agent par le résultat de l'action d'un autre agent. Un bon autre exemple qui nous intéresse pour élaborer notre solution à base de rôle de recherche qui se voie de base comme étant un objet ou une collection d'objet, recevant des données en entrée pour en faire des traitements. Cet exemple d'émergence peut se retrouver dans [224], où le problème est de ramener à un endroit donné un ensemble d'objets se retrouvant dans un environnement quelconque. A.Drogoul [224] considère deux types d'agent : une première population dite *petits poucets e*t une seconde population de *Dockers*. La population *petits poucets* est composées d'agents qui explorent l'environnement de manière aléatoire en déposant des marquages sur le sol, de sorte que les autres agents, attirés par les marquages puissent trouver plus facilement les éléments recherchés. Ce comportement, fortement inspirés par les fourmis à la recherche de nourriture (*voir la section 2.2.3 du chapitre 2*). En effet, l'action des agents modifiant l'environnement mène à une stimulation des autres agents améliorant le résultat final.

La seconde population est composée d'agents qui sont capables de détecter si un autre agent transporte un élément recherché et de le prendre. Le résultat obtenu est une chaîne d'agent allant de la source d'objets jusqu'à la destination dans Laquelle, les objets sont transportés d'agent en agent. Là encore, le comportement collectif est bien efficace que le comportement individuel.

Lorsqu'un agent se déplace, il doit poursuivre son exécution sur le site destination, la technologie agent mobile offre une solution pour la mobilité et le continuité de l'exécution.

4.2.1.1 Agent mobile

Un agent mobile est un agent logiciel qui peut se déplacer d'un site à un autre en cours d'exécution avec son code, ses données et son état d'exécution. La mobilité est contrôlée par l'agent lui-même et non par le système d'exécution. Afin de satisfaire aux contraintes des réseaux de grande taille ou sans fil (latence, non permanence des liens de communication), les agents communiquent principalement par messages asynchrones [225].
L'agent mobile est composé de son code correspondant à un algorithme, ainsi que d'un contexte incluant ces données. Ce contexte peut évoluer en cours d'exécution, par exemple en collectant des données. Lorsqu'un agent réalise une recherche d'information sur un ensemble de serveurs, le code et le contexte de l'agent sont déplacés avec l'agent lorsque celui-ci visite différents serveurs.

Lorsqu'un agent se déplace vers un serveur, il doit poursuivre son exécution sur le site destination. La plupart des systèmes à agents mobiles implantent une migration faible, c'est à dire une fonction de migration où l'agent redémarre son exécution depuis le début. En conséquence, le programmeur doit inclure dans le contexte de l'agent des informations sur l'état de l'exécution, et lorsque l'agent redémarre sur un site, le code de l'agent doit vérifier l'état de l'exécution et se brancher sur la partie de l'algorithme devant être exécutée sur ce site.

4.2.1.1.1 Code mobile

Lors d'un déplacement, le code d'un agent est téléchargé sur la machine cible et s'exécute au sein de la plateforme pour effectuer une tache requise (recherche d'information locale, négociation, etc.), avant de se télécharger de nouveau vers une autre machine ou vers la machine de l'utilisateur afin de rapatrier les résultats.

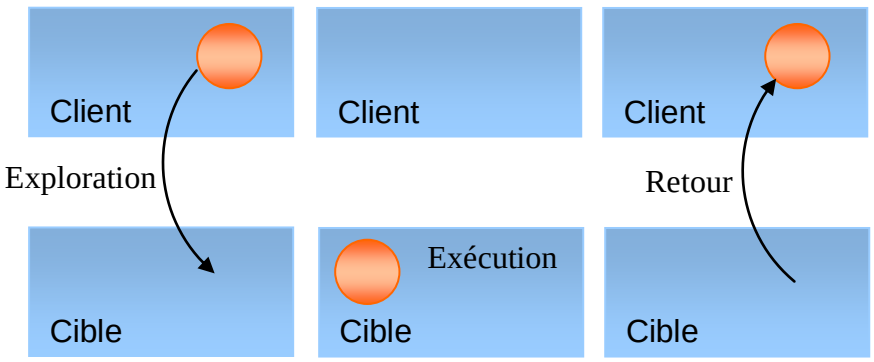

Fig.4.1 Code mobile

4.2.1.1.2 Agent mobile de recherche

Une utilisation des agents mobiles notée comme prometteuse est la recherche d'information distribuée. Les agents mobiles montrent de très bonnes performances du point de vue du réseau et réduction de la bande passante à fin de faciliter le transfert des flux de données.

Divers travaux significatifs ont été menés dans le domaine de la recherche d'information et s'appuient sur la technologie d'agents mobiles. Parmi ceux-ci on distingue:

- Dans [226], les auteurs présentent une application de recherche de documents textuels à base d'agents mobiles. Il s'agit cependant, d'une application de recherche simple dans un réseau local (avec centralisation des informations sur les serveurs via un mécanisme de pages jaunes, sans contrainte de sécurité, ni découverte dynamique de nouveaux serveurs). L'agent de recherche s'appuie sur des agents d'observation de l'état du réseau et sur un agent stationnaire qui sert d'interface avec le serveur d'information local.

- D'autres travaux qui s'intéressent à la recherche d'information textuelle à base d'agents mobiles dans [230]. Ces agents sont particuliers puisqu'ils ont un comportement mimant celui des fourmis. L'idée générale est de marquer le réseau à l'aide de phéromones virtuelles de telle sorte que les agents peuvent repérer les sites contenant de l'information pertinente grâce aux explorations précédentes du réseau. L'émergence d'une sorte de carte routière globale déposée sur le réseau indiquant les chemins pertinents provient de la somme des comportements de chaque agent.

- Le système *DBMS-Aglet* [227] implante une solution à base d'agents mobiles en Java pour l'interrogation de bases de données hétérogènes via le Web. Un agent mobile transporte la requête sur le site serveur où il acquiert dynamiquement le pilote JDBC (*Java Data Base Connectivty*) qui convient, il pose ensuite sa requête et retourne au site client avec les résultats.

- M3 "*MultiMedia Database Mobile agents*" [228] est un système de recherche de données multimédia par le contenu qui repose sur les agents mobiles, Java et CORBA (*Common Object Request Broker Architecture*). L'agent mobile peut mémoriser les informations recueillies sur un site, les utiliser sur les sites visités ensuite, les faire évoluer pendant le parcours. Les problèmes de sécurité sont pris en compte via des mécanismes de sessions indépendantes, les mécanismes de sécurité de CORBA, et des restrictions de droits.

- Une approche dans [229] s'intéresse à la recherche de données multimédia, en effet, des problèmes de propriétés des données se posent. Une vidéo par exemple, peut être protégée par les droits d'exploitation, rendant sa diffusion sur le réseau interdite. Dans ce cas, l'indexation et la recherche doivent se négocier directement avec le propriétaire. L'approche agent mobile est particulièrement intéressante pour ce cas, en ce qu'elle permet aux données de rester à l'endroit de leur stockage et à leur propriétaire de restreindre l'analyse qui en est faite par les agents venus indexer à des fins de recherche.

- d'autres travaux comme AGATHE [100], ARCADIA [231], JAVANE [232], NETSA [233], ISAME [234] proposent une autre alternative pour la recherche d'information : des modèles d'agents mobiles et multi-agents.
- Plusieurs d'autres travaux ont été énoncés dans **[235].**

4.2.1.1.3 Apports à la problématique

L'approche agent mobile est développée aux frontières du génie logiciel et des systèmes répartis, conduit également à la décentralisation de la connaissance et du contrôle. La question la plus importante ici est : qui est le meilleur agent placé pour faire un traitement ? La réponse est claire : l'agent qui détient les informations nécessaires pour accomplir la tache.

En générale, le concept d'agent mobile est a priori destiné à la mise en œuvre d'applications répartis dont les performances varient en fonction de la disponibilité et de la

qualité des services et des ressources, ainsi que du volume des données déplacées. Le principe de mobilité permet à l'agent d'opérer en étant déconnecté du client. La mobilité des agents et leur autonomie améliorent la sûreté (tolérance aux pannes par redéploiement des agents sur des nœuds en service) et permettent le suivi de matériel ou d'utilisateurs mobiles (réseaux hétérogènes, informatique ubiquitaire). Les agents mobiles sont donc un outil à part entière pour l'adaptation des systèmes ubiquitaire.

La recherche d'information a été présentée comme un champ d'application potentiel et important des agents mobiles : sources d'information multiples réparties, volumes importants, prise en compte des spécificités du client, interactions avec le client et la source. Á priori, les avantages des agents mobiles sont nombreux :

- Les agents mobiles ont plusieurs avantages dans *les systèmes de recherche d'information distribuée dans le Web ubiquitaire*, par la migration aux serveurs d'informations, un agent peut effectuer une opération de recherche localement, en éliminant le transfert de données intermédiaires. et donc de réduire le trafic sur le réseau qui ne transporte que les données utiles (éventuellement pré-traitées). En outre, cela permet des transactions plus robustes que les transactions distantes.

- L'agent mobile peut choisir dynamiquement une stratégie de déplacement de son comportement selon sa tâche et les conditions du réseau courant, et puis change ses stratégies pendant que l'état de réseau change. Alors que si on place une partie de comportement de recherche au niveau des serveurs, des comportements complexes, efficaces et robustes peuvent être réalisés avec étonnamment un minimum de code, éventuellement, une réduction importante du comportement de l'agent sera réalisée.
- Ces avantages rendent les agents mobiles particulièrement attrayants dans la conception des *applications Web ubiquitaire*, avec souvent la nécessité de traiter la dégradation de la bande passante, l'augmentation de la latence et les liens incertains des réseaux.
- L'exécution d'agents spécialisés offre d'avantage de souplesse que l'exécution d'une procédure standard sur les sites serveurs. L'asynchronisme, l'autonomie et la réactivité de l'agent mobile lui permettent de réaliser une tâche tout en étant déconnecté du client, ce qui est particulièrement utile dans le cas d'une session de recherche (client ou serveurs d'information).
- Le principal apport des agents mobiles se situe sur un plan du génie logiciel : les agents mobiles sont des unités de structuration des applications ; ils unifient en un modèle unique différents paradigmes d'interaction entre entités réparties (client-serveur, communication par messages, code mobile).

4.2.1.1.4 Limites

Plusieurs expériences significatives ont été menées concernant l'apport des agents mobiles, en particulier dans le domaine de la recherche d'information [236]. Cependant, on constate qu'en pratique les technologies à base d'agents mobiles n'ont encore été que peu utilisées. On peut avancer ici quelques éléments d'explication.

Les problèmes de sécurité posés par l'utilisation d'agents mobiles constituent un frein à l'expansion de cette technologie. Les risques concernent la confidentialité, l'intégrité et la disponibilité des machines hôtes et des agents, ainsi que des échanges sur le réseau [237].

Des éléments de solutions existent à base de chiffrement symétrique ou asymétrique (clé publique/privée). Le problème reste cependant ouvert et dépasse largement la problématique des agents mobiles, comme par exemple le problème des attaques par déni de service (denial of service) dans lesquelles un service est saturé de requêtes afin de le rendre indisponible.

Par rapport aux technologies plus classiques, le concept d'agent mobile offre un cadre fédérateur pour traiter des problèmes différents et se substituer à diverses solutions; c'est l'argument génie logiciel. Mais pour l'instant, cet argument a eu peu de poids face au saut technologique demandé aux développeurs.

Enfin, les expériences primitivement menées avec des applications relativement figées dans des environnements stables et des réseaux locaux mettaient peu en évidence l'intérêt de l'agent mobile en termes de flexibilité. On peut penser qu'aujourd'hui le contexte applicatif a suffisamment changé et qu'il est nécessaire d'explorer d'avantage cette technologie au regard des problèmes posés par le passage à l'échelle et l'ouverture des systèmes.

4.2.2 Composants logiciels

4.2.2.1 Présentation

Sur le plan du génie logiciel, développer une application aujourd'hui revient souvent à trouver le bon compromis entre la production de code sur mesure (spécifique mais coûteux et long à développer et validé) et la réutilisation de logiciel existant (qu'il suffit parfois de configurer, mais qui peut être moins adapté aux besoins). L'approche composant facilite ce compromis et constitue une technique de conception particulièrement avantageuse dans le cadre des systèmes complexes tels que les SRI dans le *Web ubiquitaire*.

Il n'existe pas de définition canonique d'un composant logiciel, une vision consensuelle dans [238] présente un composant comme un morceau de logiciel « *assez petit pour que l'on puisse le créer et le maintenir, et assez grand pour qu'on puisse l'installer et le réutiliser* ». Le composant est une unité de structuration et de composition qui spécifie précisément, via des interfaces, les services synchrones et évènements asynchrones rendus et requis. Le composant logiciel est une évolution du concept d'objet qui reprend ses objectifs d'origine telle que l'encapsulation et la séparation entre interface et réalisation, la réduction de la complexité et la réutilisation dans une perspective de structuration plus importante que dans le modèle objet. Un composant peut être déployé indépendamment de toute application et notamment de son langage de conception, ce qui simplifie son déploiement.

Le composant peut être vu comme une boîte noire (l'implémentation est cachée), dont seule l'interface permet de connaître les services offerts et requis (dépendances entre composants). Avec un modèle de composant donné, programmer revient à construire des architectures dans lesquelles on assemble différents composants.

4.2.2.2 Caractéristiques principales

Pour chaque modèle, un environnement d'exécution (ou plate-forme à composants ou serveur d'applications) fournit l'ensemble des services permettant l'instanciation, la configuration et l'exécution des composants, en leur offrant des services non fonctionnels comme par exemple des mécanismes de communication, de transaction, de sécurité, de persistance. . .

Les composants accèdent à ces services au travers d'une entité appelée conteneur, sorte d'environnement qui gère le cycle de vie des composants. Le paradigme composant / conteneur accentue la séparation entre les aspects non fonctionnels et le code métier du composant : c'est un mécanisme de séparation des préoccupations et des niveaux.

4.2.2.3 Exemple d'un modèle de composant : EJB

Les EJB ou Enterprise JavaBeans™ sont une spécification proposée par Sun[1] Microsystems pour décrire un modèle de composants distribués du monde Java, exécutés du côté serveur. L'interopérabilité entre serveurs d'application est basée sur un middleware réseau qui peut être RMI (*Remote method innvocation*) ou CORBA/IIOP (*Common Object Request Broker Architecture*).

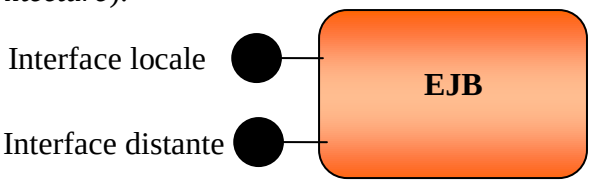

Fig.4.2 Modèle abstrait des composants EJB

Sun est engagé pour la spécification d'un standard qui doit être respecté par les fournisseurs de serveurs d'applications. Dans le but, de fournir une architecture standard de composants logiciels répartis et de simplifier le travail du programmeur en automatisant et en cachant les problèmes complexes. Les annotations (ex : @EntityManager, @Persist,..) sont au cœur de la simplification des EJB, en fait, ce sont des méta-données placées dans les scripts pour définir les comportements du conteneur JEE (*Java Entreprise Edition*) et des objets managés, en effet, Les descripteurs XML sont toujours simple à valider et lisibles grâce à la technique d'annotation.

Dans cette norme, un composant correspond à un Bean (haricot), c'est-à-dire une classe Java pour laquelle, seuls les services fournis sont déclarés (via des interfaces). Dans la version actuelle 3.0 de la norme. La spécification définit trois familles de composants :

- 1) **Session Bean** : sont des composants métier de l'application, comparable à une session interactive, ce type admet deux types :
  - *Stateless session Bean (SLSB)* : est un Bean qui encapsule un traitement fonctionnel à usage unique et ne conserve aucun état de la part du client entre deux appels de service.
  - *Stateful session Bean (SFSB)* : est une collection de services dont chacun est représenté par une méthode avec une conservation de l'état pendant la durée d'une session client.

- 2) **Entity Bean** : pour la représentation de données et la conservation d'un état persistant. Cette famille est constituée de :
  - Container Managed Entity Bean (CMP)
  - Bean Managed Entity Bean (BMP)
- 3) **Message Driven** Bean : dédié au pilotage des messages asynchrones.

1- Sun Microsystems (http://java.sun.com/products/ejb)

La norme EJB définit différents types d'acteurs (schématisés dans la figure 4.3) dans une architecture EJB :

- Le serveur (ou serveur d'applications),
- Le conteneur,
- Le composant.

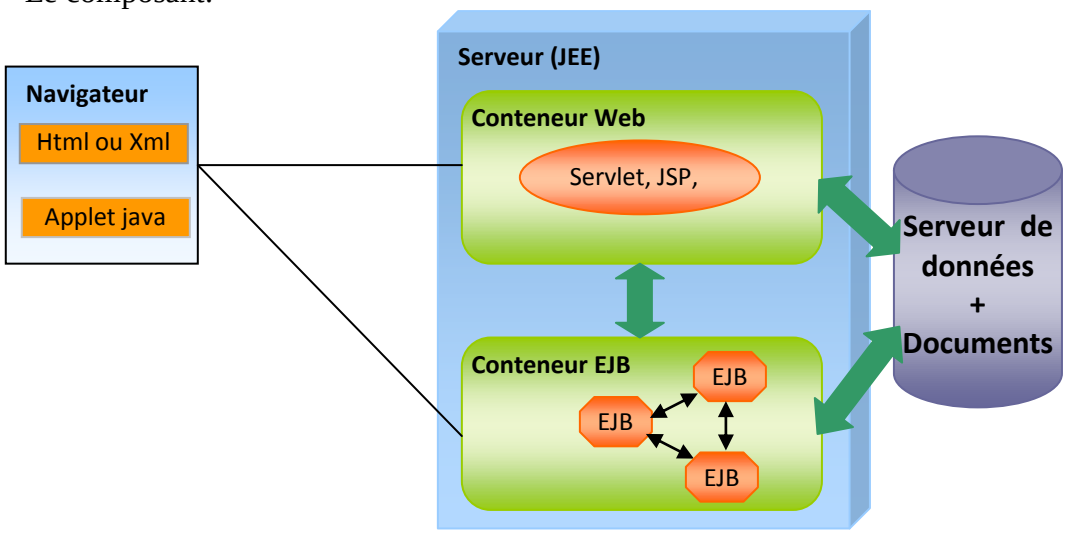

Fig.4.3 Conteneur EJB incorporé dans un serveur JEE

Puissance d'EJB

- Fournit l'infrastructure de déploiement d'applications distribuées.
- Fournit les services de transaction, de sécurité, de nommage.
- Présente une interface publique afin d'offrir la possibilité d'implémenter des conteneurs EJB spécifiques à un domaine ou à une technologie particulière.

Le Conteneur EJB assure le contrat entre le composant et le serveur d'applications.

- Il intercepte la communication entre le client et le composant pour ajouter le code d'infrastructure.
- Il a la charge du cycle de vie des composants et leur offre un contexte d'exécution.
- Il fournit une interface d'accès distant pour les composants.

4.2.2.4 Mécanismes de sûreté d'assemblages

Lors de l'assemblage de composants, on peut essayer de vérifier si celui-ci est valide en confrontant les spécifications des services offerts et requis par les composants. Une première approche consiste à se baser sur des mécanismes de typage, en exploitant les signatures des services. Mais cela peut se révéler insuffisant. Ainsi, lorsqu'un concepteur développe une application par composants, il peut utiliser des composants écrits par des tiers, dont il ne connaît pas le comportement interne, et qui peuvent ne pas répondre à ses attentes.

Pour éviter cela, il est possible d'étendre les spécifications par des contrats qui explicitent des propriétés sémantiques (par exemple au moyen d'expressions OCL, Object Contraint Language, langage de contraintes d'UML), de qualité de service, ou encore de propriétés environnementales du composant.

4.2.2.5 Architectures logicielles

Il est possible de monter d'un niveau en abstraction et de considérer l'organisation, les connexions entre composants ou encore les flots de contrôles et de données échangés. C'est ce qui constitue la base d'une architecture logicielle (ou application framework).

Cette branche récente du génie logiciel cherche à faciliter et à organiser l'implémentation, l'exécution, l'exploitation et la maintenance d'un logiciel sans se focaliser sur des problèmes d'implémentation (algorithmique ou structures de données). Le niveau d'abstraction fourni par une architecture permet de réaliser certaines vérifications (propriétés de sûreté, de sécurité,..) indépendamment de l'implémentation sous-jacente et avant même l'implémentation, ce qui permet de gagner en fiabilité et en temps de conception. La description des assemblages et réassemblages dynamiques d'une architecture peut être réalisée avec des langages plus ou moins formels appelés ADL (*Architecture Description Language*). Certains ADL permettent par la suite de réaliser certaines validations, en particulier au niveau des assemblages obtenus (cohérence de l'architecture).

4.2.2.6 Apports et limites

L'approche composant permet la séparation des différentes préoccupations (aspects métiers entre eux et aspects métiers vs. aspects non-fonctionnels) ainsi que la réutilisation de codes existants. Cela contribue à la simplification du développement et de la maintenance *d'applications Web ubiquitaire*.

Par contre, dans la plupart des implémentations de serveurs d'applications, les services non fonctionnels sont limités en nombre, non extensibles, et donc peuvent ne pas couvrir tous les besoins.

De plus, l'installation d'un composant dans un environnement d'exécution ubiquitaire n'est pas toujours simple. En général, il faut exploiter un descripteur de déploiement (fichier XML), qui contient toutes les caractéristiques nécessaires au bon déploiement d'un composant (catégorie, interfaces, services. . .). Le déploiement consiste à partir de ce descripteur à connecter le composant dans l'architecture existante et l'activer (en initiant son cycle de vie), ce qui peut requérir des tâches additionnelles de configuration, à la charge de l'administrateur du serveur d'application. Ce qui rentre en contradiction avec les besoins d'autonomie des *applications Web ubiquitaire*.

Enfin, l'utilisation des modèles de composants académiques ou industriels cités ci-dessus implique l'installation de l'environnement d'exécution adéquat. La lourdeur des serveurs d'application existants ne semble pas adaptée à des petits périphériques tels que les PDA, les téléphones mobiles..).

Pour ces raisons, nous nous intéressons d'avantage au concept de composant qu'aux modèles de composants existants. Aussi, nous ne préconiserons pas l'utilisation spécifique de l'un d'entre eux dans le cadre de notre solution. En fait, les rôles de recherche qui font les briques de base pour la construction de la solution qui sera présentée pourraient être implémentés par un modèle proche d'EJB, c'est-à-dire qu'un composant est caractérisé par son interface, une ou plusieurs réalisations et un moyen d'exécution (son conteneur).

4.2.3 Intergiciels

4.2.3.1 Présentation

Un intergiciel (ou middleware, ou logiciel médiateur) est une couche intermédiaire entre un système d'exploitation et une application, qui offre à cette dernière des services de haut niveau permettant de faire des abstractions sur le système et le matériel sous-jacent (figure 4.4).

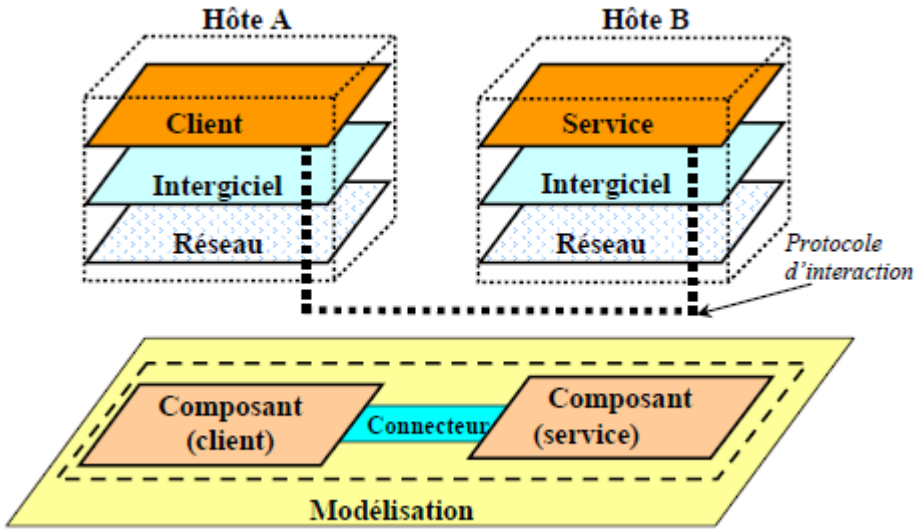

Fig. 4.4 Modélisation des intergiciels, des clients et des services

À l'origine, les premiers intergiciels avaient pour objectif de permettre l'interopérabilité dans le modèle client/serveur, pour cacher la répartition et l'hétérogénéité des composants matériels, systèmes et réseaux. On trouve actuellement dans la littérature un large spectre de middlewares. Des middlewares réseau comme CORBA ou RMI permettent de simplifier l'accès à des objets distribués sur des machines distantes.

Pour déterminer et raisonner sur les différentes solutions possibles pour résoudre l'hétérogénéité des intergiciels il est nécessaire de considérer un *environnement ubiquitaire* comme un système distribué de systèmes. Un tel système peut être modélisé au moyen des concepts issus des architectures logicielles [239] : les composants abstrayant les services (resp. les clients) et les connecteurs abstrayant les intergiciels (voir Figure 4.4).

L'*environnement du Web ubiquitaire* est alors modélisé sous la forme d'un graphe. Les nœuds du graphe reliés par des arcs sont des composants interconnectés *via* des connecteurs. Etant donné l'aspect dynamique d'un *environnement ubiquitaire*, le graphe évolue au cours du temps. L'apparition et/ou la disparition de nouveaux services (resp. de clients) correspond à une modification dynamique du graphe *via* une suppression ou une adjonction de composants et/ou de connecteurs. Cependant, l'évolution du graphe est soumise à une contrainte forte qu'il est nécessaire de surmonter : les composants ne peuvent être associés qu'à des connecteurs pour lesquels ils ont été conçus, ou plus précisément à des connecteurs avec lesquels ils sont compatibles [240]. Ainsi, résoudre l'hétérogénéité des intergiciels revient à résoudre, à un niveau plus abstrait, les incompatibilités d'interconnexions entre les composants et les connecteurs.

4.2.3.2 Catégories d'intergiciels

D'après Bernstein [241], les intergiciels se déclinent en quatre catégories :

- 1. *Moniteur de transaction* (TP, Transaction Processing monitors) qui fournit des outils pour le développement et le déploiement d'applications transactionnelles distribuées ;
- 2. *Appel de procédure à distance* (RPC, Remote Procedure Call) qui permet l'exécution d'une routine sur une application distance comme si elle se trouvait sur la machine locale ;
- 3. *Intergiciel orienté message* (MOM, Message-oriented Middleware) qui fournit des mécanismes asynchrones d'échange de données entre programmes, ce qui permet la création de véritables applications distribuées ;
- 4. *Bus logiciel* (ORB, Object Request Brokers), qui permet aux objets qui constituent une application d'être répartis et partagés au travers d'un environnement réseau hétérogène.

Dans notre système de RI à base d'agents mobiles, nous nous intéressons essentiellement aux catégories 3 et 4.

Les intergiciels adaptables permettent alors d'envisager la conception d'applications efficaces et robustes dans un contexte d'environnement variable. Les évolutions de l'environnement peuvent provenir de la mobilité physique du système, de la mobilité de l'application (éventuellement partielle) autant que des changements dans le contexte d'exécution.

4.2.3.3 Intergiciels ouverts

L'objectif principal d'un intergiciel est d'aider un concepteur d'application à résoudre ou simplifier des problèmes d'interopérabilité et de distribution. Pourtant, la plupart des intergiciels sont basés sur des implémentations ou des protocoles propriétaires, rendant les applications dépendantes d'un unique distributeur. Cette dépendance peut avoir des impacts négatifs sur la flexibilité et la maintenabilité de l'application. Pour cela, il nous semble judicieux d'utiliser des intergiciels ouverts (au sens de l'ouverture logicielle du terme Open Source), c'est-à-dire dont les spécifications sont publiques ou bien reposant sur des standards ouverts. Par exemple, la communication dans le cadre de web services repose sur des protocoles et formats ouverts (HTTP, SOAP, XML. . .), ce qui offre une interopérabilité maximale.

4.2.3.4 Intergiciels à agents

Le modèle de programmation par agent offrant des abstractions sur l'exploitation du réseau (localisation, communication. . .), on peut considérer que le niveau agent est une couche intergicielle.

Par exemple, le système Anthill[2] est un ensemble de machines distribuées sur les quelles sont déployés des systèmes multi-agents. Leur interaction permet de résoudre des problèmes complexes (grâce à des comportements émergents et des algorithmes génétiques (présenté dans la section 2.2.4 chapitre 2) comme par exemple le routage des messages. Anthill s'inspire des colonies de fourmis (présenté dans la section 2.2.3 chapitre 2), en proposant des agents aux comportements simples, autonomes et pourvus d'un environnement sur lequel ils basent leurs actions [242]. Chaque machine possède une interface appelée nid, qui fournit à

2- Anthill (http://www.cs.unibo.it/projects/anthill)

l'application des services spécifiques aux systèmes répartis : gestion de ressources, communication, gestion de la topologie, planification des actions. Dans leurs exemples applicatifs, les concepteurs montrent que le développement d'une application répartie est simplifié par l'usage de leur système.

Il existe quelques intergiciels permettant de concevoir des applications à bases d'agents mobiles, toutefois ceux-ci sont souvent limités par leur manque de flexibilité, que ce soit au niveau de la gestion de l'hétérogénéité des supports d'exécution ou des capacités d'adaptation leur permettant une exécution dans un environnement réellement ubiquitaire.

De nombreuses plates-formes agents peuvent être trouvées à partir de :
http://www.cetus-links.org/oo_mobile_agents.html#oo_mobile_agents_software.

4.2.4 Techniques pour la flexibilité

Dans la section 2.4 du chapitre 2, nous avons montré les différents verrous scientifiques à lever et les besoins d'adaptation des systèmes de recherche d'information contextuelle, nous avons également défini la terminologie employée à propos de l'adaptation au contexte de l'environnement de recherche. Dans cette section, nous présentons des techniques permettant d'implémenter des mécanismes d'adaptation, en insistant sur les principes d'adaptation dynamique. Les technologies classiques d'adaptation statique (principes de modularité, héritage, délégation, code ouvert) sont assez connues pour ne pas être détaillées.

L'adaptation du logiciel est une préoccupation orthogonale aux éléments fonctionnels, au même titre que la sécurité par exemple. Dans le cas général, il n'est pas nécessaire de la prendre en compte ni de l'implémenter pour qu'un système fonctionne, mais cela peut néanmoins lui permettre d'améliorer son exécution. Dans la littérature, on trouve des exemples d'adaptation dynamique à tous les niveaux : à l'échelle de l'instruction (certains virus dits polymorphes peuvent modifier dynamiquement leur code -écrit en assembleur-, afin de se cacher des anti-virus), de la méthode (programmation orientée aspect), de l'objet (réflexivité et protocole à méta-objet), du composant (composant adaptable), de l'intergiciel (intergiciel adaptable) et jusqu'à l'application elle-même (IHM adaptée à l'utilisateur, plugins. . .). Qui feront l'objectif majeur des recherches qui s'intéresse à la RI dans l'informatique ubiquitaire, et en particulier, en RI dans le Web Ubiquitaire.

4.2.4.1 Composants et adaptation

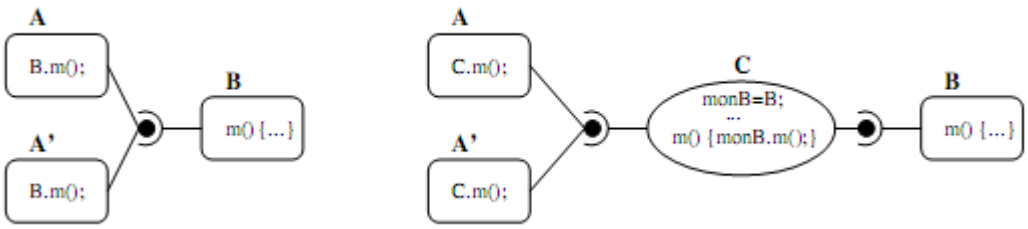

Fig. 4.5 Intérêt du connecteur dans une architecture à composants

Si l'on compose des objets par un mécanisme de délégation ou des composants directement entre eux, l'adaptation dynamique est difficile. En effet, dans l'exemple de gauche de la figure 4.5, l'appel de méthode de A vers B est généralement compilé. Pour changer B (par un autre composant, donc avec une autre référence) il faudrait mettre à jour

donc modifier le code de tous les composants connectés à B, donc de A et A0 dans notre exemple. Le modèle composant/connecteur permet de s'affranchir de cette limitation, en intercalant un composant délégataire appelé connecteur. Dans l'exemple de droite, les composants A et A0 appellent la méthode m() du connecteur C. Celui-ci possède un attribut qui référence l'adresse du composant rendant ce service, ici monB, et l'appel de méthode est délégué à cet objet. Ainsi, lors d'une reconfiguration qui change le composant B en B0, l'évolution est complètement transparente pour les composants A et A0.

Pour réaliser cette adaptation dynamique, il suffit de modifier l'attribut monB du connecteur C.

4.2.4.2 Programmation par aspects

La programmation par aspects (AOP - Aspect-Oriented Programming ou separation of concerns, [243]) est issue du constat des limites de la programmation par objet, lorsqu'il est nécessaire de prendre en compte des propriétés non-fonctionnelles telles la trace, la distribution, l'authentification ou la persistance. La dissémination des portions de code non-fonctionnel dans l'ensemble des applications limite leur réutilisation, et pose desproblèmes de maintenance. L'AOP propose de séparer clairement le code fonctionnel (les services métier de l'application) et les services non-fonctionnels.

Le logiciel est construit par assemblage ou tissage (*weaving*) d'aspects non-fonctionnels dans le programme métier, à des endroits appelés points de jonction. Le tissage est un mécanisme statique, qui fonctionne comme un compilateur (*aspect weaver*). Les concepteurs peuvent ainsi travailler sur trois parties distinctes : aspects métiers, aspects non-fonctionnels et configuration. De plus, chaque élément peut être réutilisé dans un projet futur.

Par exemple, AspectJ[3] est une extension du langage Java développée au Xerox-PARC[4] et actuellement hébergée par le projet Open Source Eclipse[5]. Les aspects sont codés dans un langage Java-étendu et la configuration repose sur l'utilisation de mots-clés spécifiques.
Après tissage, le compilateur dédié retourne du code java pur ou directement du bytecode.
AspectJ ne permet pas l'adaptation dynamique, mais d'autres systèmes (JAC ou EAOP[6]) le permettent, en reposant toutefois sur des outils imposants. Certains auteurs considèrent que les services non-fonctionnels d'un serveur d'application ou plus généralement d'un middleware forment autant d'aspects.

4.2.4.3 Réflexivité

Un système est dit réflexif s'il est capable d'appliquer sur lui-même ses propres capacités d'action. Le terme réflexif est employé dans de nombreux domaines (linguistique, philosophie, mathématiques. . .). En informatique, un système logiciel est dit réflexif [244] lorsqu'il est capable d'inspecter (propriété d'introspection) et de modifier sa structure et sa sémantique (propriété d'intercession).

La réflexivité est un moyen puissant de réaliser un découplage entre les aspects applicatifs et les aspects non-fonctionnels d'un système. On parle de méta-calcul pour désigner l'activité du système lorsque celui-ci interprète les parties réflexives (non fonctionnelles) du programme.

3- http://www.eclipse.org/aspectj
4- Palo Alto Research Center
5- http://www.eclipse.org
6- JAC (Java Aspect Components). EAOP (Event-based Aspect Oriented Programing) est un modèle qui associe l'exécution des aspects à des évènements mis lors de l'exécution du programme et capturés par un moniteur d'exécution.

L'approche réflexive est une technique générale pour implanter des mécanismes d'adaptation dynamique [245]. En effet, pour pouvoir s'adapter et faire les bons choix de configuration, il faut connaître d'abord l'état de l'environnement, mais aussi son propre état (y compris d'architecture logicielle). Par exemple le langage Java(http://java.sun.com) possède certaines capacités réflexives. L'API Java (paquetages java.lang et java.lang.reflect) met à disposition du programmeur des classes spécialisées telles que Class (les instances de Class, des objets donc, représentant les classes connues par la machine virtuelle Java), Method ou Constructor et un ensemble de méthodes pour les exploiter. Toutefois, ces capacités ne fournissent que des moyens d'observation de niveau relativement haut du processus d'interprétation du bytecode Java.

D'un point de vue extérieur, un système réflexif n'est pas plus puissant qu'un système qui ne l'est pas [246]. Tous les problèmes que peut traiter un système réflexif peuvent aussi l'être par un système non réflexif équivalent. La réflexivité, comme les langages structurés, comme l'orienté objet, est une manière d'organisation interne d'un système pour faciliter son développement, son adaptation et sa réutilisation.

Exemple d'adaptation dynamique : ACEEL

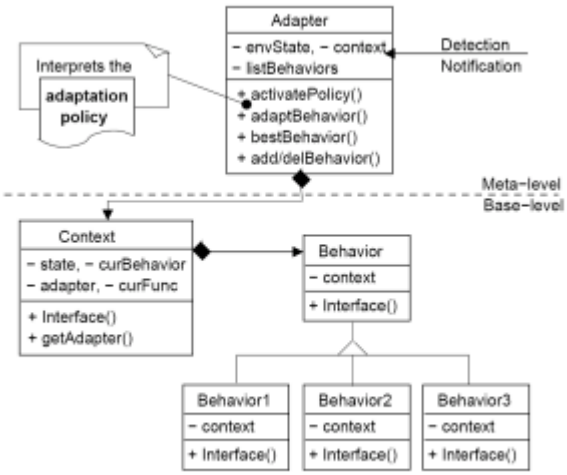

Fig. 4.6 ACEEL: self-Adaptive ComponEnts modEL

Le modèle de composant ACEEL [247] a pour objectif de permettre le changement dynamique du comportement d'un composant (le code métier) réalisant un service particulier. ACEEL utilise le Strategy design pattern, un patron de conception [248] qui définit une même interface pour toutes les variantes d'un algorithme et un accès depuis une classe qui contient le lien de délégation vers l'implantation choisie. Au méta-niveau, un objet Adapter prend en charge la politique d'adaptation du composant. Les décisions d'adaptation sont prises à partir des informations fournies par sous-système de détection des changements de l'environnement d'exécution du composant.

ACEEL a été implémenté en Python[7], puis testé avec une application de vidéo à la demande, qui possède des comportements alternatifs correspondant à des algorithmes d'encodages distincts.

Nous avons présenté un ensemble de technologies logicielles qui offrent chacune des apports intéressants à la construction d'applications réparties, et également, un système de RID sensible aux contextes et ouvert à grande échelle, avec toutefois un ensemble de limitations présentées dans ce tableau.

7- Langage de script interprété, orienté objet, et réflexif : http://www.python.org

Technologie	Apports	Limites
Agent mobile	Décentralisation du contrôle et de la connaissance, adaptation via la mobilité, autonomie et réactivité.	la complexité de la structuration de modèles de comportements, et l'augmentation sans cesse de la charge sur les réseaux.
Composants	Modularité, structuration et réutilisation, séparation des préoccupations.	Déploiement des composants complexe, environnement d'exécution souvent volumineux et peu adapté à des petits périphériques.
Intergiciels	Abstractions du système et du matériel donc, simplification de la programmation et maîtrise de l'hétérogénéité.	Manque de souplesse dans les solutions existantes, support limité d'une montée en charge importante.
Systèmes flexibles	Nombreuses techniques qui permettent en particulier l'adaptation dynamique du système au contexte d'exécution.	Augmentation de la complexité du Développement.

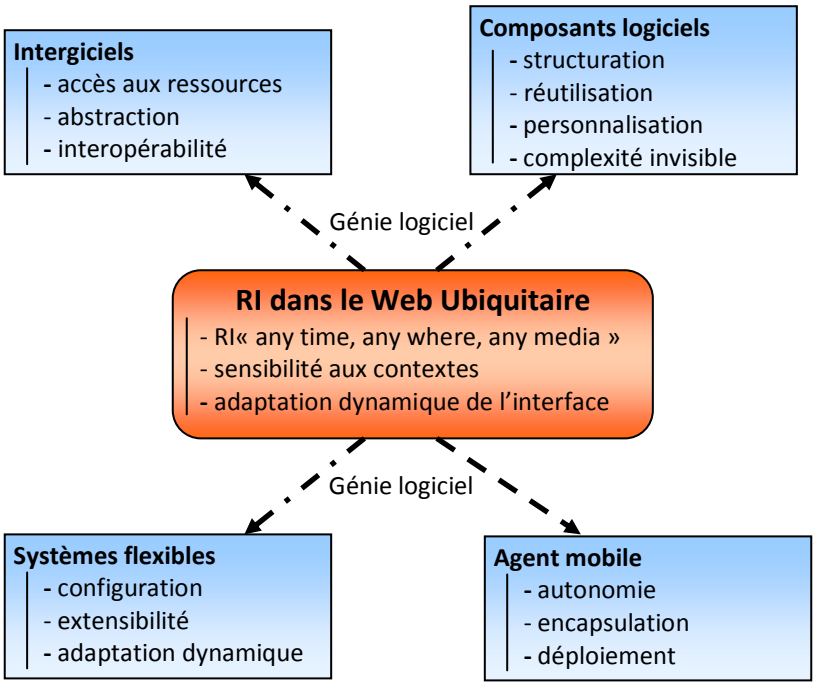

Fig 4.7 Positionnement des technologies

4.3 Modèle d'agent mobile de recherche

Nous présentons un modèle d'agent mobile de recherche à base de rôle adaptable, sensible au contexte dynamiquement, pour s'adapter aux variations de son environnement de recherche ubiquitaire. En fait, ce modèle d'agent nourri des travaux qui ont été réalisés dans ce domaine et qui ont des succès notable dans le domaine de la recherche d'information. Vu que l'agent se voie comme étant un composant, le comportement de l'agent mobile proposé est construit principalement à partir d'un assemblage de rôles de recherche de granularité faible, il s'agit, en fait, de micro-composants remplaçables et spécialisables en fonction de la variabilité du contexte de la recherche. Ce style d'architecture d'agent flexible doit faire face

à la forte dynamicité de son environnement ubiquitaire de recherche, dans lequel, différentes stratégies d'assemblages de rôles permettent d'engendrer un comportement sensible au contexte de la recherche.

Nous allons utiliser le rôle comme étant un composant spécialisé à la RI pour construire par assemblage le comportement des agents mobiles de recherche, nous allons déployer une base de rôle de recherche au sein des serveurs d'informations. Où chaque agent de recherche véhicule le minimum de comportement étant donné qu'à tout serveur d'informations, il existe un comportement spécifique que nous organisons sous forme de rôles. De cette façon, l'agent visitant un serveur, peut déposer ou absorber des rôles selon sa tache.

4.3.1 Rôle de recherche

On restreint donc la définition du comportement des agents mobiles de recherche à la gestion de leurs rôles, passage d'information et changement : la cognicité des agents se limite à la possibilité de choisir dans une situation donnée une collection parmi ceux dont ils disposent. Les rôles pris en charge deviennent entièrement responsables des éléments de comportements de recherche.

4.3.2 Modélisation du rôle de recherche

Les langages de description d'architecture (ou ADL pour Architecture Description Langage), employés dans le domaine des architectures logicielles, reposent sur un certain nombre de concepts fondamentaux pour modéliser les systèmes logiciels, dont principalement les *composants*, les *connecteurs* et les *configurations* [249], [250], que nous réutilisons dans le contexte de notre architecture pour modéliser un rôle de recherche ubiquitaire.

Les rôles se présentent comme des briques logicielles dotées d'une ou de plusieurs interfaces de communication (décrit dans la figure 4.8). Chacune d'entre elles correspond à un port de communication. Un rôle dispose alors d'un ensemble de ports qui sont des points d'interaction avec le monde extérieur. Les ports peuvent également être perçus comme des points d'accès au rôle.

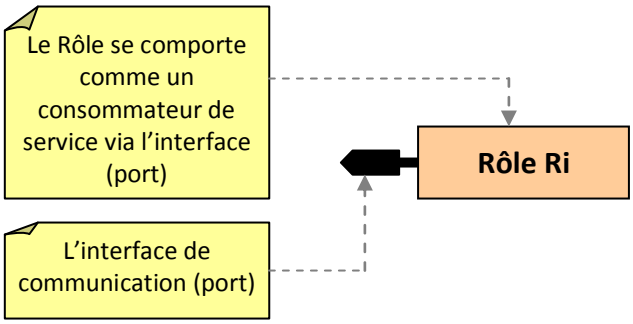

Fig.4.8 rôle de recherche

4.3.3 Comportement de recherche

La configuration décrit l'architecture du système de RI sous la forme d'un assemblage de rôles de recherche et de connecteurs formant ainsi un graphe. Plus particulièrement, la configuration peut être statique ou évolutive. Selon la spécificité de l'ADL considéré pour modéliser un système. La configuration est par exemple statique si l'on considère l'ADL

Wright [251], [252] ou évolutive si l'on considère sa version dynamique [253]. L'aspect statique d'une configuration ne correspond pas à notre vision de l'environnement ubiquitaire où l'assemblage des rôles et des connecteurs est dynamique. Notre perception de la configuration est plus proche des ADL où la configuration est évolutive. On distingue plus précisément deux types d'évolution possible : planifiée, et non planifiée **[250],** [103].

- *Une évolution planifiée* signifie que le système est uniquement en mesure de s'accommoder à des changements dynamiques prévus à l'avance (par exemple, l'approche de l'ADL Darwin [254] ou de la version dynamique de l'ADL Wright [253])
- *Une évolution non-planifiée* caractérise l'aptitude générale du système à s'adapter à des changements dynamiques impromptus (par exemple, l'approche de l'ADL C2 [255]). Cette dernière définition caractérise parfaitement la conception que l'on a d'un environnement ubiquitaire.

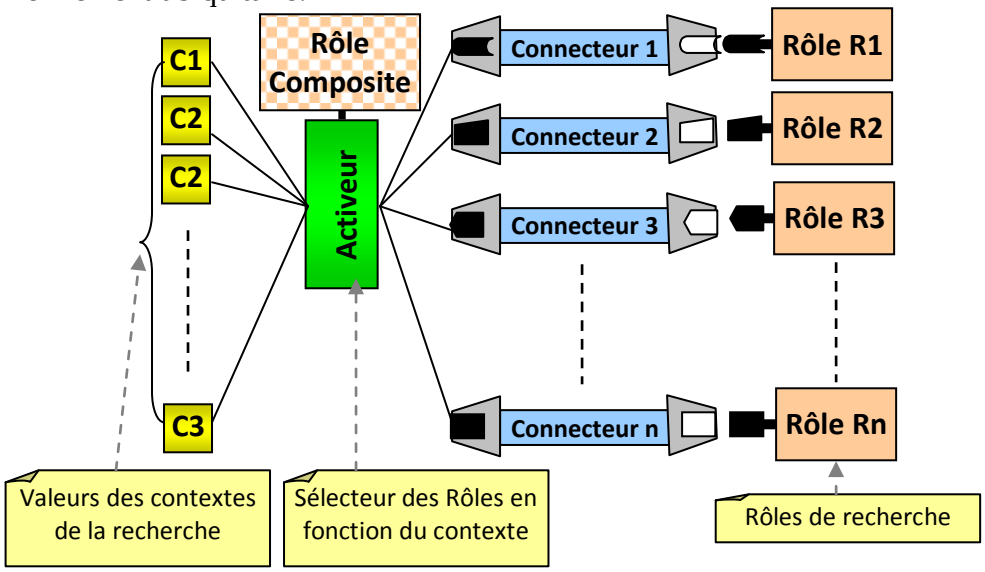

Fig.4.9 Configuration dynamique de rôles de recherche en fonction de du contexte

Celui-ci est donc un graphe pourrait être construit à partir du patron de conception de composition (*composite design pattern*). Par conséquent, le comportement de recherche embarqué au sein des agents mobiles est un graphe de rôles et de connecteurs qui évolue et/ou réagit au cours du temps via des assemblages/désassemblages, des apparitions/disparitions de rôles et de connecteurs.

L'évolution du système ainsi modélisé n'étant pas planifiée, il y a potentiellement des incohérences de couplages, qui se traduisent par l'incapacité d'interconnecter des composants, étant donné l'incompatibilité de leurs ports respectifs avec les connecteurs qui les interconnectent susceptibles de survenir.

4.3.4 Organisation du système d'agents et rôles

Dans notre système multi-agents, plusieurs entités évoluent dans un même environnement. Ces entités qui sont en interaction et utilisent souvent les mêmes ressources ne peuvent être traitées isolement.
Dans les approches *centrées organisation* des systèmes multi-agents, les organisations s'intéressent généralement soit au fonctionnement, soit à la structure interne des agents dans le système. Cependant, la combinaison de ces deux dimensions peut s'avérer bénéfique pour

spécifier les organisations à condition de le faire de manière cohérente et flexible. De ce fait, nous utilisons le formalisme Agent-Groupe-Rôle (AGR) [256]-[257] pour définir l'organisation sociale de notre système de recherche d'information distribuée dans le Web ubiquitaire.

AGR permet de mieux représenter l'organisation sociale d'un système multi-agents, et de gérer l'hétérogénéité et l'ouverture des systèmes informatiques tels qu'Internet d'une manière souple et modulaire.
Dans le formalisme AGR (décrit par la figure 4.10), un agent est une entité active et communicante. Il peut prendre simultanément plusieurs rôles dans un seul groupe de recherche. Un groupe de recherche est défini par un ensemble d'agents de recherche interagissant à travers leurs rôles. Un rôle définit la représentation abstraite de la fonction d'un agent dans un groupe.

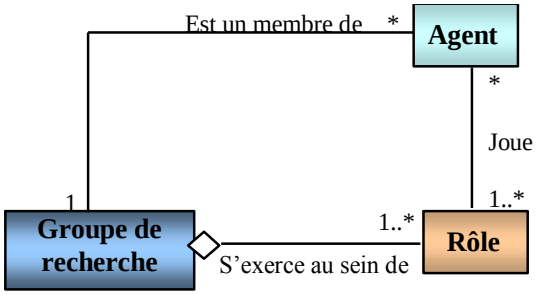

Fig.4.10 : Le modèle d'organisation AGR adapté

L'utilisation du formalisme permettra d'augmenter la souplesse de notre système. Par exemple, un agent peut changer dynamiquement la stratégie d'exécution des rôles de recherche. En termes de programmation et de conception, il permet d'augmenter l'expressivité et la modularité du système. La maintenance du modèle est également plus facile et permet une évolution à long terme pour l'adapter à d'autres situations différentes. De même, l'agent ne communicant qu'à travers les rôles, le traitement des messages est plus facile.

La satisfaction des buts de recherche globaux communs est due à une sorte de mémoire organisationnelle dans laquelle sont stockés les meilleurs plans de recherche (Politiques pour allouer les rôles aux agents, coordination pour exécuter un plan, et qualité d'un plan, temps consommé, ressources utilisées, . . .). Ceci est à la charge d'un *agent Médiateur* qui sera présenté par la suite.

Groupe de recherche

Un groupe de recherche sera formé pour chaque requête utilisateur, par l'envoie d'une copie de comportement configurée par *l'agent Médiateur* aux serveurs. Les agents mobiles créés seront repartis entre les différents serveurs spécifiés et dont leur nombre est défini en fonction des paramètres de la requête et le contexte de la recherche. Chaque agent mobile véhicule alors, un minimum de comportement étant donné qu'à tout serveur existe un comportement spécifique que nous organisons sous forme de rôles. Lui permettant d'enrichir son comportement par absorbation des rôles adéquats.

4.4 Architecture proposée

Dans notre *système de recherche d'information dans le Web ubiquitaire*, les agents mobiles sont des entités informatiques capables d'agir et de communiquer avec les serveurs d'informations, qui peuvent jouer un ou plusieurs rôles dans un seul groupe (Gi) pour une requête donnée (RQi). Quand un agent de recherche (ARij) émigre, non seulement son comportement de recherche, mais également, son état peut être transféré à la destination. En outre, la structuration de modèles de comportements de l'agent peut se baser sur une division des rôles : une partie des rôles sera embarquée au sein de l'agent et une autre sera déployée sur les serveurs. En conséquence, le système peut être ouvert pour changer et évoluer dynamiquement les rôles de recherche et pour s'adapter à l'environnement ubiquitaire d'exécution et aux contextes dynamiques des utilisateurs. La recherche est établie principalement par trois phases :

- 1. *Déplacement au serveur* : par application d'une stratégie de déplacement du comportement, l'agent mobile peut véhiculer le minimum de rôles, Ces rôles peuvent subir dynamiquement à une optimisation selon les conditions de l'environnement ubiquitaire, à fin de restreindre le comportement de l'agent et ainsi diminuer le trafic de déplacement sur le réseau.
  L'amélioration de l'exécution du système nécessite qu'un agent de recherche (ARij) ne se déplace pas tous le temps avec tout son comportement. En particulier, un agent de recherche efficace est dans la mesure de choisir dynamiquement un comportement de recherche minimal, c.-à-d., combien de rôles de recherche peuvent être suffisants pour effectuer une recherche sur un tel serveur.

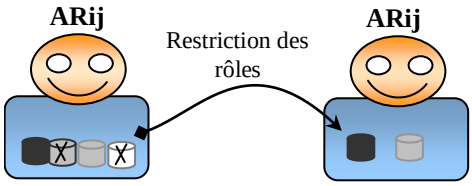

Fig.4.11 : Restriction du comportement de l'agent mobile de recherche

  Si cet agent devrait émigrer au serveur, il devrait sélectionner sur la base de ses paramètres de recherches et les conditions de l'environnement, les rôles de recherche nécessaires à déplacer (Comme le décrit la Figure 4.11). Et ainsi, l'agent doit avoir accès à une richesse de connaissances sur le réseau, les autres agents, les serveurs d'informations, les différentes ressources et les outils correspondants à des algorithmes de planification, de sorte qu'il puisse choisir la stratégie de migration des rôles de recherche la plus efficace pour sa tâche et la plus adéquate aux conditions de l'environnement ubiquitaire courant. Donc, notre système mobile d'agent doit fournir une infrastructure de sensation et de planification étendue [258].

- 2. *Consultation d'une base de rôles* : L'agent peut exploiter un comportement spécifique que nous organisons dans une base sous forme de rôles au sein de chaque serveur. Cette base joue un double rôle, d'une part, elle sert à diminuer le nombre de rôles de recherche qui peut être véhiculé par les agents, et d'autre part, elle sert de station d'évolution de ses rôles de recherche par les différents passages des agents.
- 3. *Recherche locale* : pour effectuer une recherche au sein d'un serveur avec la prise en compte de la sensibilité aux contextes, l'agent de recherche s'engage à extraire les informations recommandées avec l'adoption des caractéristiques en multi niveau du contexte.

4.4.1 Base de rôles de recherche (BR)

Les agents mobiles sont utilisés pour adapter aux besoins des clients un service fourni par un serveur. Nous supposons que l'objectif du serveur est de fournir un service générique pouvant répondre aux besoins variés de tous ses clients potentiels [259].
On propose d'installer une base de rôles qui sert de référence initialement à ce service générique, notant que cette base s'enrichit par les rôles de recherche sous forme de méta connaissances glanées par les agents de recherches au cours de leur parcours des serveurs permettant ainsi de référencier la disponibilité d'informations et l'accès au source d'information de qualité d'une manière évolutive , par exemple : si un agent porteur d'une requête provenant d'un médecin arrive à un serveur d'informations hystérogènes, il trouve un rôle de recherche désignant la disponibilité d'informations sur un autre serveur spécialisé en un domaine précis :médecine, un grain d'efficacité significatif obtenu lors de l'organisation des résultats de recherche à l'utilisateur par ordre de qualité.

4.4.2 Framework

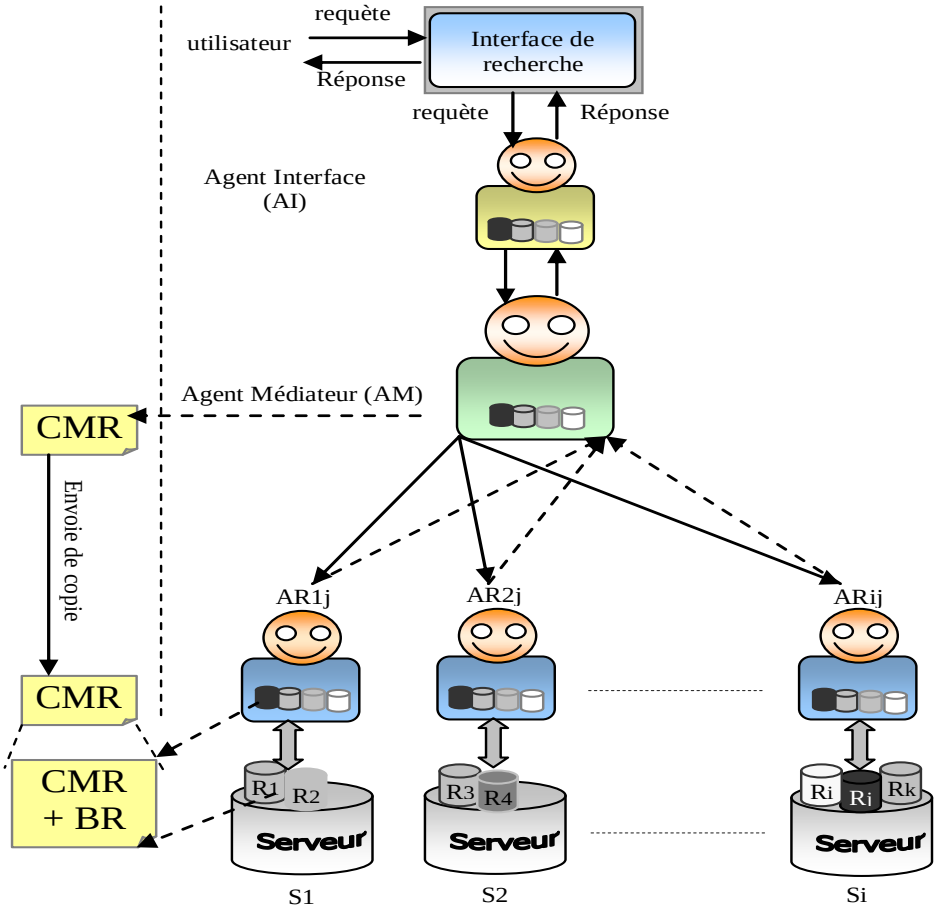

Fig.4.12 : Architecture du système proposé

4.4.3 Spécification des Agents

Notre architecture comporte trois types d'agents, comme indiqué dans la figure 4.12.

4.4.3.1 Agent Interface

Cet agent peut être vu comme un simplificateur permettant aux utilisateurs d'interagir avec le système (voir Fig. 4.13). C'est un agent *stationnaire* doit fournir une infrastructure de

sensation et de capture dynamique de contexte de l'utilisateur et du terminal utilisé, cet agent doit remplir un fonctionnement sensible aux contextes de la recherche via une interface qui sépare l'utilisateur du système. Le contexte n'est pas restreint aux profile du l'utilisateur et les caractéristiques du son terminal mais aussi la localisation géographique et l'ensemble des interactions avec le système. Cet *agent d'interface* est responsable principalement des taches suivantes:

- 1. Acquisition des requêtes des utilisateurs :
- 2. Acquisition du contexte de la recherche :
- 3. Envoie des requêtes et contextes à *l'agent Médiateur.*
- 4. Affichage des résultats adaptés à l'interface utilisateur : Cet agent a la responsabilité de présenter les résultats, pour ce faire, collecte les résultats rendus par *l'agent Médiateur* (AM) puis les range et les présente sur l'interface utilisateur, organisés par une relation de classification.

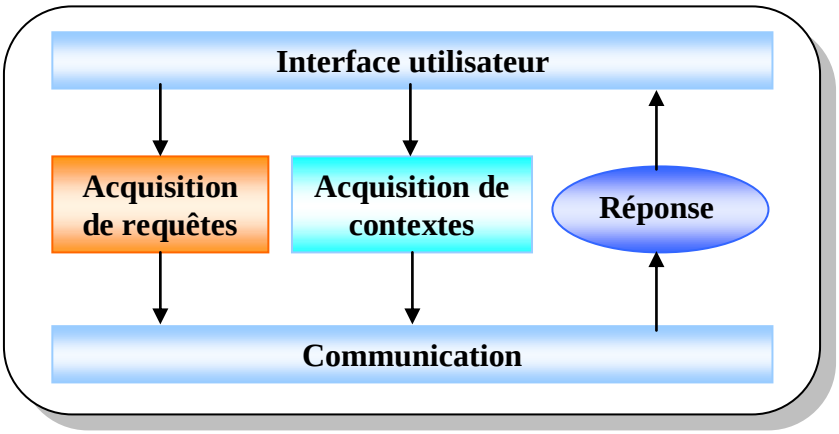

Fig.4.13 Taches interne de l'Agent Interface

4.4.3.2 Agent Médiateur (AM)

L'agent médiateur est un agent *intelligent* qui joue un rôle central, son comportement est due à une sorte de mémoire organisationnelle dans laquelle sont stockés les meilleurs plans de recherche (Politiques pour allouer les rôles aux agents, coordination pour exécuter un plan, et qualité d'un plan, temps consommé, ressources utilisées, . . .). Cet agent est responsable de la configuration du comportement de recherche, par assemblage des rôles de recherche adéquats aux contextes capturés par l'agent Interface. Cet agent a la responsabilité de filtrer les résultats, pour ce faire, il collecte les résultats rendus par les *agents de recherche* dont ils ont été crées par lui même, puis les renvoie à l'agent Interface pour les ranger et les écrire sur l'écran (voir la figure 4.14).

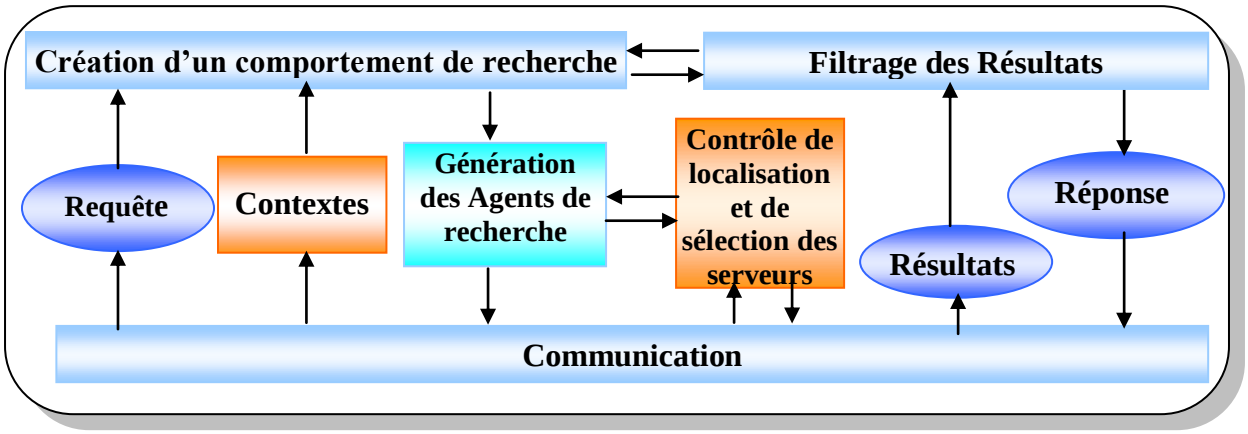

Fig.4.14 les tâches internes de l'agent médiateur AM

Les taches de l'agent médiateur AM sont:

- 1. Création *d'un comportement minimal de recherche (CMR)* : configuration dynamique de rôles de recherche en fonction de contextes doit fournir une infrastructure de sensation et de planification étendue.

- 2. *Génération des Agents de recherche* : l'agent Médiateur a la responsabilité de créer un groupe de recherche (Gi) formé des agents mobiles de recherche (ARij). Chacun d'eux et doté d'une copie du comportement minimal de recherche (CMR). Leur nombre est défini après avoir localisé et sélectionner les serveurs sources d'informations.

- 3. *Sélection des serveurs et Contrôle de localisation*: l'agent Médiateur désigne les serveurs (Si) de la recherche : serveurs d'informations sélectionnés pertinents pour la requête de l'utilisateur liée avec l'ensemble de contextes de la recherche. L'agent médiateur a aussi la faculté de contrôler les localisations des agents mobiles de recherche sur le réseau, ce qui permet de les informer des changements des contextes au cours d'exécution.

- 4. *Récupération des Résultats* : le médiateur rassemble les résultats trouvés par le groupe d'agents chercheurs sous forme d'une réponse globale qui représente l'information demandée par l'utilisateur.
- 5. *Filtrage des Résultats* : La sélection et la fusion des résultats sont à la charge de l'agent Médiateur aussi, il détermine également la pertinence globale des listes de résultats retournées par le groupe de recherche, en fait, cet agent applique un deuxième filtre globale sur les résultats en espérant limiter le bruit dans la réponse.

- 6. *Renvoie des Résultats à l'agent Interface*.

4.4.3.3 Agent de recherche AR

A l'arrivée aux serveurs d'information (Si), chaque agent de recherche (ARij) du groupe (Gi) effectue les traitements suivants (décrit dans la figure 4.15):

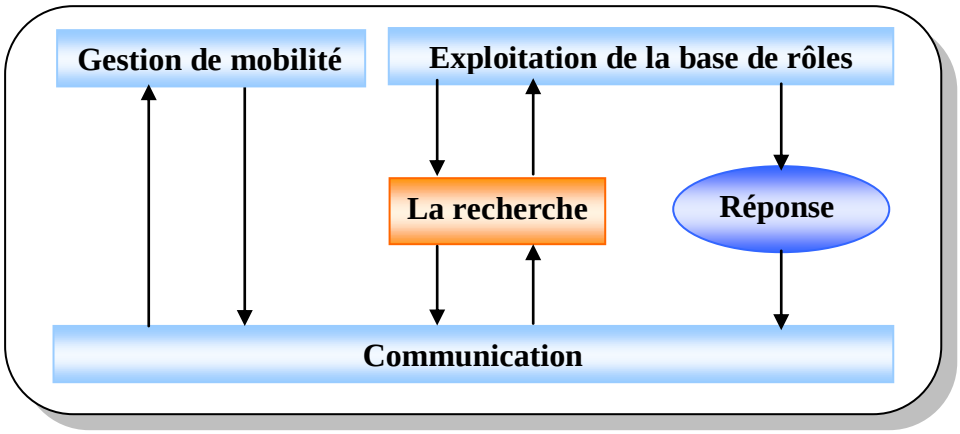

Fig.4.15 Taches interne de l'Agent de Recherche

- 1. *Sélection (BR, roles_exec)* : Recherche d'un nouveau rôle à exécuter dans la base de rôle. Comme le décrit la Figure 4.16.

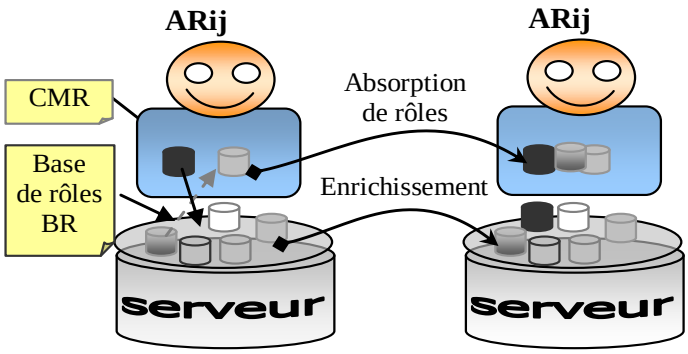

Fig.4.16 Absorbation de rôles et enrichissement de la base de rôles par les agents de recherche

Si oui, *Enrichir (CMR, roles_exec)* : il s'enrichit (enrichit son Comportement Minimal de Rechercher) par les nouveaux rôles à exécuter ;

Sinon, il utilise le CMR.

- 2. *Recherche_locale (Si, CMR)* : il effectue une recherche locale au serveur (Si) en utilisant son Comportement de Recherche jusqu'à obtenir des résultats (Res).

- 3. *Adapte (BR, CMR)* : Une fois la recherche locale est terminée, il adapte ces rôles (les optimise) et les colle sur le serveur (Si) laissant ainsi une trace de son opération de recherche sous forme de rôles. Notant que durant l'opération de la recherche locale, chaque agent (ARij) est mis à l'écoute des différents *notifications (msg, ARij) :* notification provenant des membres de son groupe (Gi) donnant la possibilité d'utiliser des rôles de recherche provenant des autres agents lui permettant ainsi d'adapter son processus de recherche.

- 4. *notification (msg, Gi)* : envoie un message aux autres agents du groupe contenant des informations sur son opération sur le serveur (Si).

- 5. *Résultat (Res, AM)* : Rendu des résultats obtenus par chaque agent de recherche (ARij) à l'agent Médiateur (AM), Eventuellement rendu des rôles de recherches trouvés ou bien optimisés.

4.4.4 Processus de la RI ubiquitaire locale

À l'arrivée aux serveurs d'information (Si), l'agent de recherche (ARij) trouve à sa disposition une base de rôles de recherche tandis que, les informations décrivant le contexte de la recherche quelques soient concernant l'utilisateur, sa localisation ou les caractéristiques physiques et logiques de son terminal utilisé, le contexte peut être divisé en contexte externe (par exemple la présence et position des utilisateurs, leurs préférences, l'heure de la journée, etc.) et contexte des ressources (le niveau de la batterie des dispositifs, la quantité de mémoire libre, le débit des communications, etc.). Une fois que le contexte est capturé, le système doit répondre aux changements détectés. Ceci est appelé adaptation sensible au contexte. Par exemple, si un nouvel utilisateur arrive dans un hôtel, l'architecture logicielle qui soutient la recherche d'information doit changer; par exemple des rôles de recherche devront être automatiquement pris en compte par l'agent de recherche pour que cet utilisateur puisse recevoir des informations pertinentes concernant son endroit et le système doit être prêt pour ses futurs déplacements dans la région ou il se trouve. On suppose que l'ensemble de valeurs de contexte est capturé automatiquement par des mécanismes externes à notre système et enregistré dans une base de contextes du système. Nous préconisons l'utilisation des caractéristiques à multi-niveau [260] du contexte pour l'adaptation du processus de RI aux changements du contexte.

Le processus de la recherche locale est effectué conceptuellement par trois tâches principales, comme affiché dans le schéma suivant :

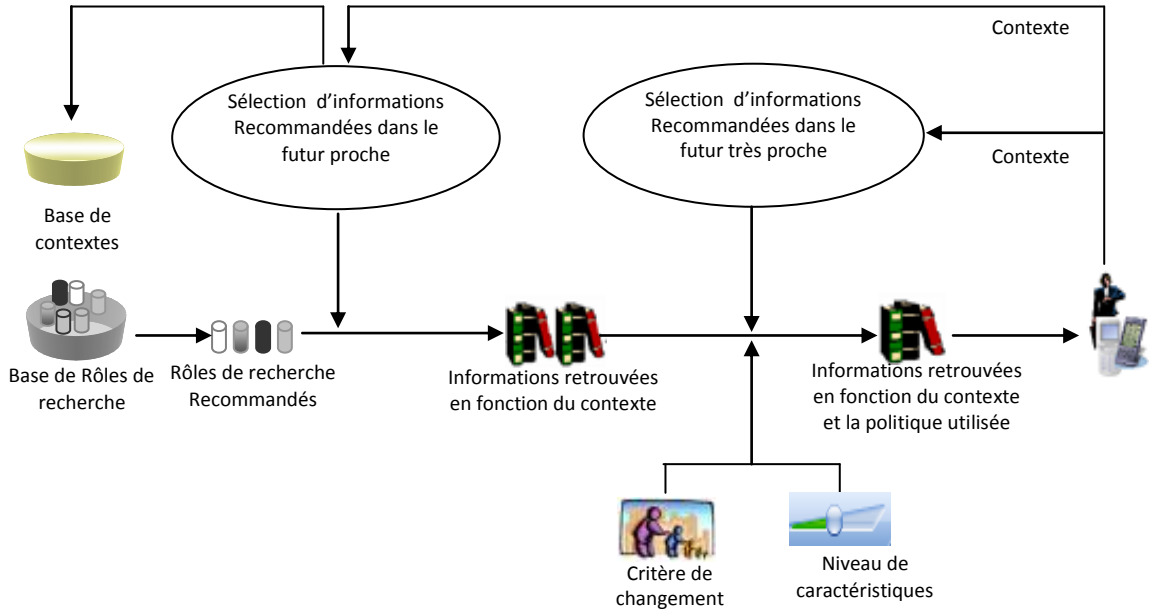

Fig.4.17 Processus de la recherche ubiquitaire locale.

- 1) La première tache s'intéresse à l'extraction des rôles de recherche recommandés de la base de rôles en fonction des valeurs du contexte. Ceux-ci sont extraits de la base de contextes.

- 2) La deuxième étape, les informations retrouvées dans le futur proche sont extraites en utilisant les valeurs du contexte courant et les rôles recommandés. En effet, cette étape établi une RI à un niveau de caractéristiques du contexte qui inclus les valeurs les plus possibles intervenantes dans le future proche.

- 3) Finalement, sélection des informations qui devront être apparues dans le futur très proche en fonction des valeurs du contexte, la politique d'application des règles et la priorité du niveau de caractéristiques du contexte utilisé. Les informations retrouvées dans la deuxième étape peuvent engendrer une surcharge mémoire et augmenter le taux de transfert, pour résoudre ce problème, seules les informations qui peuvent être utilisées dans le très proche futur, seront stockées dans la mémoire et après le processus de recherche peut se répéter pour améliorer la sensibilité de la recherche au contexte.

La politique utilisée pour gérer la valeur du niveau de caractéristiques du contexte courante est déterminée en fonction du critère de changement et le niveau de caractéristiques. Le niveau de caractéristiques signifie en fait, le niveau de détail de l'information et détermine les propriétés de l'information à extraire. Les informations dont on a besoin dans un niveau « *bas* » de détail ou de granularité faible, elles sont de priorité « *élevée* » et vise vers ça.

Dans un niveau de caractéristiques, le SRI retourne les documents en fonction de leur niveau de pertinence. L'évaluation de la requête doit donc pouvoir retourner les documents ayant le plus haut niveau de pertinence en tête des autres documents. Pour ce faire, il est nécessaire de prendre en compte les différents niveaux de pertinence d'un document par rapport à un besoin d'information [**183**] et dans le niveau de caractéristiques courant. Pour déterminer la valeur du niveau de pertinence. On suggère d'utiliser une collection avec une échelle à *n* niveaux de pertinence : (hautement pertinent, suffisamment pertinent, marginalement pertinent, non pertinent, …). Chacun de ces niveaux est quantifié par une valeur numérique qui dépend de la politique utilisée dans un niveau de caractéristiques donné, la politique utilisée détermine les valeurs de pertinence par exemple : le niveau de pertinence "*hautement pertinent*" a la valeur 5 et le niveau " *suffisamment pertinent*" a la valeur 3 ; les autres ont la valeur 1, dans un niveau de caractéristiques « *haut* » et le critère de changement du contexte « *lent* ». Si le contexte de l'utilisateur subit à un changement rapide que ce soit de sa localisation, des caractéristiques de son dispositif mobile, etc. le critère de changement devient « *rapide* », le niveau de pertinence "*hautement pertinent*" diminue sa valeur à 4 et la valeur du niveau " *suffisamment pertinent*" reste 3 ; cependant, la valeur du niveau " *marginalement pertinent*" devient 2. Par conséquent, la recherche d'information dans un niveau de caractéristique « *moyen* » devient à la fois, sensible au changement du contexte et retourne les informations pertinentes de la façon la plus rapide.

La valeur du niveau de caractéristiques ou de détail est influencée par le critère de changement, par exemple le critère de changement dans un contexte de localisation peut avoir la valeur « *lent* » désignant la vitesse de déplacement de l'utilisateur, par conséquent, le niveau de détail prioritaire qui devrait être pris en compte est le niveau « *élevé* » et vise vers ça.

La politique utilisé pour l'extraction des informations selon le niveau de granularité appropriée est absolument importante et vise également,

- la prise en compte du volume d'information lors de l'évaluation des requêtes
- augmentation de la sensibilité du processus au contexte,
- l'extraction de l'information rapide et pertinente.

4.5 Conclusion

Dans ce chapitre, nous avons mis l'accent principalement sur le défi de la sensibilité au contexte, et la surcharge du flux. Pour cela, nous avons pris en compte le contexte de la recherche comme étant une multitude de niveau de caractéristiques qui influence directement la qualité et la vitesse d'extraction des informations. De plus, nous avons proposé une architecture basée sur le concept d'agent mobile et rôle. La recherche d'information distribuée sur Web ubiquitaire à base d'agents mobiles doit faire face au problème de la charge totale des réseaux due à la charge véhiculé par ces agents, ainsi que l'hétérogénéité des sources d'information. Ceci a freiné le succès de diminuer le temps de réponse aux requêtes des utilisateurs de la majeure partie des approches de recherche distribuées dans Web.

Notre approche améliore cette situation par la structuration de modèles de comportements d'agents mobiles, nous avons adopté le concept de rôle pour organiser le comportement interne de l'agent, lui permettant de se déplacer à travers les réseaux avec un ensemble restreint de rôles de recherche étant donné qu'à tout nœud il existe une base spécifique de rôles de recherche. De cette façon, l'agent visitant un nœud sélectionne sur la base de ses paramètres de recherche les rôles adéquats, à fin d'enrichir son comportement et de même, il peut contribuer à l'évolution de cette base de rôles de recherche par déposition de nouveaux rôles, ou bien par adaptation des rôles existants. Ceci réduit la charge au niveau de l'agent mobile et donc le trafic de son déplacement, et par conséquent, réduction de la charge totale des réseaux.

Notre système offre aussi une possibilité d'améliorer les rôles de recherche par échange de messages entre les agents du même groupe. Dans ce dernier point, nous suggérons l'utilisation des rôles de granularité minimale, pour qu'ils soient échangeables par message.
Á la fin du processus de recherche, chaque agent peut laisser une trace de sa visite sur le serveur sous forme de rôles de recherche, parmi ces propres rôles ou bien parmi ceux qu'ils ont subi une évolution lors du processus de recherche.

Cette technique d'amélioration des rôles de recherche conduit à améliorer, d'une part, l'adaptation du processus de recherche à l'hétérogénéité des serveurs et des réseaux et d'autre part d'augmenter le niveau de flexibilité intra-agent pour avoir une sensibilité au contexte de son environnement ubiquitaire d'exécution.

Nous avons utilisé le formalisme Agent-Groupe-Rôle (AGR) pour définir l'organisation sociale de notre système SMA de recherche d'information ubiquitaire. AGR permet de mieux gérer l'hétérogénéité et l'ouverture de l'Internet d'une manière souple et modulaire.

Conclusion générale

Le *Web ubiquitaire* est une nouvelle vision qui offre beaucoup de nouvelles possibilités et qui présente en même temps des défis pour la recherche. En effet, les environnements ubiquitaires combinent des aspects complexes tels que la mobilité, les données contextuelles, les interactions multimodales, l'hétérogénéité des dispositifs, les communications sans fil, etc. Le potentiel de ce type d'environnements n'a pas encore été totalement exploité.

Donc, Le Web *ubiquitaire* se différencie en termes de contenu et de structure. En effet, il se caractérise par une forte hétérogénéité des sources d'information : hétérogénéité des langages de communication, hétérogénéité des médias (texte, image, vidéo), hétérogénéité des structures, etc. La contextualisation de la recherche permet d'adapter le contenu et la structure du document au contexte de l'utilisateur, et de diminuer ainsi l'hétérogénéité des sources d'information. L'émergence du domaine de la RI contextuelle provient également de l'évolution de l'environnement physique de l'utilisateur : PDA, téléphonie mobile, etc. Plusieurs études ont considéré les implications de ce nouvel environnement en RI [**140**] .Les conclusions de ces travaux ont montré la nécessité de prendre en compte le contexte physique dans les modèles de RI.

Un des défis majeurs pour la recherche d'information dans le Web ubiquitaire est le problème des grands volumes d'informations. En plus, un surcroît engendré par les informations du contexte qu'un SRI dans le Web ubiquitaire doit les traiter. En effet, la recherche d'information pertinente dans le *Web ubiquitaire,* parmi un volume important d'information, adaptée aux besoins spécifiques d'un utilisateur donné, à tous moment et avec n'importe quel dispositif mobile, devient à la fois difficile et nécessaire. Difficile d'abord en matière de surcharge du réseau, et aussi en matière de changement dynamique de contexte d'utilisateurs. En clair, le problème n'est pas tant la disponibilité de l'information mais sa pertinence relativement à un contexte d'utilisation particulier. Et nécessaire, pour satisfaire les besoins des utilisateurs quelque soit leur contexte de recherche. La RI sensible au contexte quant à elle pose des problématiques nouvelles allant de la modélisation du contexte jusqu'à la modélisation de la pertinence cognitive en passant par la modélisation de l'interaction entre un utilisateur et un SRI sensible au contexte.

Un SRI dans le *Web ubiquitaire* doit répondre aux requêtes des utilisateurs en utilisant un système distribué, par lequel ils expédient ces requêtes à de nombreux processus de recherche, chacun inspecte à son tour. En effet, Le SRID envoyant la requête originale, reçoit de nombreuses listes des résultats trouvés. À son tour, Le SRID doit alors fusionner ces différentes listes et les présenter à l'utilisateur avec une seule liste rangée.

L'une de nos réflexions est de combattre la complexité de la structuration des systèmes de recherche d'information distribuée dans le *Web ubiquitaire,* pour cela, Nous avons préconisé l'utilisation de l'agent mobile comme outil de génie logiciel et plus précisément, Nous avons présenté un modèle d'agent mobile à base de rôle adaptable, sensible au contexte dynamiquement, pour s'adapter aux variations de son environnement de recherche ubiquitaire. Une des problématiques majeures de la recherche dans le *Web ubiquitaire* est de faire communiquer de façon dynamique, spontanée et transparente ces différents dispositifs entre eux indépendamment de leurs hétérogénéités matérielle et logicielle. Les intergiciels ont été introduits dans cet objectif, En fait, notre modèle d'agent nourri des travaux qui ont été réalisés dans ce domaine et qui ont des succès notable dans le domaine de la recherche

d'information. L'utilisation d'un intergiciel sensible au contexte permet de développer de manière plus souple les applications sensibles au contexte [37].

Vu que l'agent se voie comme étant un composant, le comportement de l'agent mobile proposé est construit principalement à partir d'un assemblage de rôles de recherche de granularité faible, il s'agit, en fait, de micro-composants remplaçables et spécialisables en fonction de la variabilité du contexte de la recherche. Ce style d'architecture d'agent flexible doit faire face à la forte dynamicité de son environnement ubiquitaire de recherche, dans lequel, différentes stratégies d'assemblages de rôles permettent d'engendrer un comportement sensible au contexte de la recherche.

Notre architecture de recherche basée agent mobile et rôle met l'accent principalement sur le défi de la sensibilité au contexte, et la surcharge du réseau. Pour cela, nous avons pris en compte le contexte de la recherche comme étant une multitude de niveau de caractéristiques qui influence directement la qualité et la vitesse d'extraction des informations. De plus, nous avons proposé une architecture basée sur le concept d'agent mobile et rôle. La recherche d'information distribuée sur *Web ubiquitaire* à base d'agents mobiles doit faire face au problème de la charge totale des réseaux due à la charge véhiculé par ces agents, ainsi que l'hétérogénéité des sources d'information. Ceci a freiné le succès de diminuer le temps de réponse aux requêtes des utilisateurs de la majeure partie des approches de recherche distribuées dans Web.

Notre approche améliore cette situation par la structuration de modèles de comportements d'agents mobiles, nous avons adopté le concept de rôle pour organiser le comportement interne de l'agent, lui permettant de se déplacer à travers les réseaux avec un ensemble restreint de rôles de recherche étant donné qu'à tout nœud il existe une base spécifique de rôles de recherche. De cette façon, l'agent visitant un nœud sélectionne sur la base de ses paramètres de recherches les rôles adéquats, à fin d'enrichir son comportement et de même, il peut contribuer à l'évolution de cette base de rôles de recherche par déposition de nouveaux rôles, ou bien par adaptation des rôles existant. Ceci réduit la charge au niveau de l'agent mobile et donc le trafic de son déplacement, et par conséquent, réduction de la charge totale des réseaux.

Nous avons proposé l'utilisation d'une base de rôle de recherche. Cette base joue, également, le rôle d'un conteneur en gérant le cycle de vie des rôles de recherche, le déploiement de ces rôles sur les serveurs d'information peut être effectué par un agent mobile responsable du déploiement, l'implémentation de ces rôles peut être réalisée par les EJBs, nous suggérons, la création d'un nouveau type d'EJB spécialisé à la recherche d'information, vient d'ajouter une quatrième famille des EJBs (actuellement la version 3.0 est constituée des EJB *session*, EJB *entities* et EJB *message driven*).

Nous avons proposé l'utilisation d'une base de contexte pour stocker les informations de sensibilité au contexte. Une perspective à ce niveau porte sur modélisation des informations de sensibilité au contexte par une ontologie afin de faciliter l'automatisation du processus de génération du comportement de recherche par le biais d'une configuration automatique des rôles de recherche en fonction des informations du contexte.

Alors, Un point à approfondir porte sur la modélisation du contexte. Une définition standardisée des éléments du contexte pourrait être proposée. Une autre perspective concerne les modèles de RI contextuelle. La plupart des propositions ne s'intéressent qu'à une

dimension du contexte et il s'agit le plus souvent des centres d'intérêts de l'utilisateur. Les nouveaux modèles doivent prendre en compte toutes les dimensions du contexte (préférence utilisateur, temps, espace, localisation, caractéristiques du terminal, etc.). Les extensions des modèles classiques permettent déjà la prise en compte de plusieurs éléments contextuels.

Les futurs réseaux ubiquitaire qui se caractérisent par des entités mobiles (terminaux, routeurs, PDA, téléphones cellulaires, etc.) communicantes et parfois de très petite taille, posent des problèmes de mobilité, de sécurité et de sûreté (confidentialité des données, fiabilité des applications), de continuité de services (tolérance aux pannes, etc.) et de qualité de service. Ces réseaux serviront de support à des applications communicantes variées (multimedia, grid computing, peer-to-peer, Web, etc.).

Faire fonctionner correctement ces réseaux hétérogènes complexes (hétérogénéité des infrastructures, protocoles, applications, etc.) est un défi scientifique majeur pour les prochaines années. Il nécessite le développement de recherches fondamentales et appliquées en conception des architectures et des protocoles, ainsi qu'en dimensionnement, optimisation et planification des réseaux. Celles-ci doivent s'appuyer sur l'algorithmique, l'évaluation des performances, la simulation et les plates-formes d'expérimentations.

Il est alors possible de partager l'ensemble des informations récoltées entre un groupe d'utilisateurs soulignant les éléments les plus importants pour la communauté toute entière (filtrage collaboratif). Il est également possible de renforcer l'aspect collaboratif en capitalisant les connaissances extraites de résultats issus de plusieurs requêtes. À ce niveau, notre base de rôle de recherche déployée au sein du serveur d'information peut également faire évoluer les rôles de recherche par le biais des différents passages des agents de recherche, en laissant des historiques sur les différents requêtes, ce mécanisme est similaire à la stratégie de recherche de colonie de fourmi à base de la phéromone.

Enfin, l'aspect de visualisation des résultats devrait être renforcé en rajoutant de l'adaptation de l'interface d'affichage à n'importe quel terminal capable de communiquer avec les sources ubiquitaires de l'information selon ses critères physiques (vitesse, mémoire, palette de couleur utilisée …). Par ailleurs, les concepteurs de *système de recherche d'information dans le Web ubiquitaire* doivent disposer de mécanismes et d'architectures afin de stocker, récupérer et délivrer efficacement l'information la plus pertinente, en proposant un accès à travers des *DM*, quels que soient le lieu et le moment. Les concepteurs doivent également tenir compte des capacités réduites de ce type de dispositifs (par exemple, la taille de l'écran, la mémoire, le disque dur) pour un affichage approprié sur le *DM* de l'utilisateur.

Nous croyons que les meilleures manières de recherche l'information dans l'Internet en générale et dans le Web ubiquitaire en particulier sont possibles. Le temps démontre que s'il est relativement facile de réaliser des accords et des normes au sujet de rôles de recherche d'information pour faire face à l'hétérogénéité potentielle des techniques. Mais ceci s'est avérée être une tâche dure, et sa complexité commerciale est inhérente.

Bibliographie

- [1]Wahiba Nesrine Zemirli, Modèle d'accès personnalisé à l'information basé sur les Diagrammes d'Influence intégrant un profil utilisateur évolutif. Thèse de doctorat délivrée par l'Université Toulouse III - Paul Sabatier, P. 1-24, 2008.
- [2] Mohand Boughanem, Lynda Tamine-Lechani, JoséMartinez, Sylvie Calabretto, Jean-Pierre Chevallet, Un nouveau passage à l'échelle en recherche d'information, publié dans "Ingénierie des Systèmes d'Information (ISI) 11, 4 (2006) 9-35", p. 1-27, version 1 - 8 Feb 2009.
- [3] M. Weiser. The computer for the 21st century. Scientific American, vol. 265, no. 3, pp. 94-104. 1991.
- [4] J. Allan, J. Aslam, Challenges in information retrieval and language modeling: report of a workshop held at the center for intelligent information retrieval, university of Massachusetts Amherst, September 2002. *SIGIR Forum*, 37(1) :31.47, 2003.
- [5] M. Addlesee, R. Curwen, S. Hodges, J. Newman, P. Steggles, A. Ward, A. Hopper. Implementing a sentient computing system. IEEE Computer 34:50, 2001
- [6] Berhe, G. Accès et adaptation de contenus multimédia pour les systèmes pervasifs.Thèse de doctorat, Institut National des Sciences Appliquées de Lyon, Lyon, 25 Septembre 2006.
- [7] Angela Cristina CARRILLO RAMOS, Agents ubiquitaires pour un accès adapté aux systèmes d'information : Le Framework PUMAS, thèse de doctorat, UNIVERSITE JOSEPH FOURIER, p, 1-25, 2007.
- [8] Nieto-Carvajal, I., Botia, J.A., Ruiz, P.M., Gomez-Skarmeta, A.F. Implementation and Evaluation of a Location-Aware Wireless Multi-Agent System. In: Yang, L., Guo, M., Gao, G.R., Jha, N.K. (eds.): Procedings of the Embedded and Ubiquitous Computing (EUC 2004) (Aizu-Wakamatsu City, Japan, August 25-27, 2004), Lecture Notes in Computer Science, vol. 3207, Springer-Verlag, Berlin Heidelberg, pp. 528-537. 2004.
- [9] D. Saha, A. Mukherjee. Pervasive Computing: A Paradigm for the 21st Century. IEEE Computer Society, 36(2):25-31. 2003.
- [10] Koch, F., Rahwan, I. Classification of Agent-based Mobile Assistant. In: Proceedings of the Workshop on Agents for Ubiquitous Computing (UbiAgents04) (Columbia University, New York City, USA July 20, in conjunction with AAMAS, 2004.
- [11] Rahwan, T., Rahwan, T., Rahwan, I., Ashri, R. Agent-Based Support for Mobile Users Using AgentSpeak (L). In: Giorgini P., Henderson-Sellers B., Winikoff, M. (eds.): Proceedings of the Workshop on Agent-Oriented Information Systems (AOIS 2003) (Melbourne, Australia, July 14, 2003 - Chicago, USA, October 13, 2003), Lecture Notes in Artificial Intelligence, vol. 3030 Springer-Verlag, Berlin Heidelberg, pp. 45-60. 2004.
- [12] Hristova, N., O'Hare, G. Ad-me: wireless advertising adapted to the user location, device and emotions. In: Proceedings of 37th Annual Hawaii International Conference on System Sciences (HICSS37), Minitrack on Mobile Distributed Information Systems (MDIS) (Hawaii, Janvier 5–8, 2004), part of the Software Technology Track, IEEE Computer Society Press, pp. 1-10, 2004.
- [13] Pirker, M., Berger M., Watzke, M. An approach for FIPA Agent Service Discovery in Mobile Ad Hoc Environnements. In: Proceedings of the Workshop on Agents for Ubiquitous Computing (UbiAgents04) (July 20, 2004, Columbia University, New York City) , 2006.
- [14] Calisti, M., Lozza, T., Greenwood, D. An Agent-Based Middleware for Adaptive Roaming in Wireless Network. In: Proceedings of Workshop on Agents for Ubiquitous Computing (UbiAgents04) (Columbia University, New York City, USA July 20, 2004) in conjunction with AAMAS. 2004.
- [15] Collier, R.W., O'Hare G.M.P., Lowen, T., Rooney, C.F.B. Beyond Prototyping in the Factory of the Agents. In: Marík, V. Müller, J.P., Pechoucek, M. (eds.): Proceedings of the 3rd International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2003) (Prague, Czech Republic, June 16-18, 2003), Lecture Notes in Computer Science, vol. 2691, Springer-Verlag, Berlin Heidelberg (2003), pp. 383-393.
- [16] O'Hare, G., O'Grady, M. Addressing Mobile HCI Need through Agents. In: Paterno, F. (ed.): Proceedings of the 4th International Symposium on Human Computer Interaction with Mobile Devices and Services (MobileHCI'02) (Pisa, Italy, September 18-20, 2002), Lecture Notes in Computer Science, Springer-Verlag, Berlin Heidelberg, vol. 2411, pp. 311-314, 2002.

- [17] R. Want, T. Pering. System Challenges for Ubiquitous & Pervasing Computing. Proceedings of the 27th international conference on Software engineering, (ICSE'05), pp 9-14, NEW York, NY, USA. ACM Press. 2005.
- [18] Yérom-David BROMBERG, Résolution de l'hétérogénéité des intergiciels d'un environnement ubiquitaire, these de doctorat à L'UNIVERSITÉ DE VERSAILLES SAINT-QUENTIN-EN-YVELYNES, p. 5-47, 2007.
- [19] M. P. Papazoglou, D. Georgakopoulos. Service-Oriented Computing. Communications of the ACM, 46(10). 2003.
- [20] R. E. Schantz and D. C. Schmidt. Middleware for Distributed Systems: Evolving the Common Structure for Network-centric Applications. Encyclopedia of Software Engineering (J. Marciniak and G. Telecki, eds.), New York: Wiley & Sons. 2001.
- [21] E. Guttman, C. Perkins, J. Veizades, and M. Day. Service Location Protocol, Version 2. IETF RFC 2608, Network Working Group. 1999.
- [22] Bouchet, J. et Nigay, L. Icare: a component-based approach for the design and development of multimodal interfaces. In CHI '04 : CHI '04 extended abstracts on Human factors in computing systems, pages 1325–1328, New York, NY, USA. ACM Press. 2004.
- [23] Lebert, M. (2001). Le livre 010101. http://www.etudes-francaises.net/ entretiens/00livre.htm.
- [24] Rec. VXML (2004). Voice Extensible Markup Language (VoiceXML) Version 2.0. http://www.w3.org/TR/voicexml20/.
- [25] Millàn, J., Ferrez, P. et Buttfield, A. The idiap braincomputer interface: An asynchronous multi-class approach. In Dornhege, G., d. R. Mill´an, J., Hinterberger, T., McFarland, D. et Muller", K.-R., éditeurs: Towards Brain-Computer Interfacing. The MIT Press. 2007.
- [26] Ochs, M., Sadek, D. et Pelachaud, C. La représentation des émotions d'un agent rationnel. In Workshop Francophone sur les Agents Conversationnels Animes, pages 43–52, 2005.
- [27] Bazin, C., Chuffart, F. et Madelaine, J. Construction d'une application vocale pour la sélection d'objets à l'aide d'un modèle basé sur les hypergraphes. In 9éme Conférence Internationnale sur le Document Numérique. 2006.
- [28] Frédérique LAFOREST, De l'adaptation à la prise en compte du contexte :Une contribution aux systèmes d'information pervasifs, thèse de doctorat, Laboratoire d'Informatique en Images et Systèmes d'information LIRIS UMR CNRS 5205, l'INSA de Lyon et l'Université Claude Bernard Lyon I, p, 1-14, 2007.
- [29] A.K. Dey, G.D. Abowd, and D. Salber. A conceptual framework and toolkit for supporting the rapid prototyping of context-aware applications. *Human-computer Interaction*, 16(2-4 (special issue on context-aware computing)) p. 97–166, December 2001.
- [30] N. Schilit, B.and Adams and R. Want. Context-aware computing applications. In *Proceedings of the 1st IEEE International Workshop on Mobile Computing Systemsand Applications*, Los Alamitos, CA, 1994.
- [31] G. K. Mostéfaoui, G. Pasquier-Rocha, and P. Brézillon. Context-aware computing: A guide for the pervasive computing community. In *ICPS*, pages 39–48, 2004.
- [**32**] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. Van Laerhoven, and W. Van de Velde. Advanced interaction in context. In *Proceedings of the First International Symposium on Handheld and Ubiquitous Computin, HUC 99*, pages 89–101, Karlsruhe, Germany, September 1999.
- [33] A. Harter, A. Hopper, P. Steggles, and A. Ward. The anatomy of a context-aware application. In *Mobile Computing and Networking*, pages 59–68, 1999.
- [34] Carabelea, C., Boissier, O., Ramparany, F. Benefits and Requirements of Using Multi-agent Systems on Smart Devices.In: Kosch, H, Bôszôrményi, L, Hellwagner,H (eds.): Proceedings of the 9th International Euro-Par Conference on Parallel Processing (Euro-Par) (Klagenfurt, Austria, August 26-29, Lecture Notes in Computer Science, vol. 2003. 2790, Springer-Verlag, Berlin Heidelberg, pages1091-1098. 2003.
- [35] Thilliez M, Delot T. Evaluating Location Dependent Queries Using ISLANDS. In Ramos, F, Unger, H., Larios, V. (eds.): Proceedings of the Symposium on Advanced Distributed Systems (ISSADS) (Guadalajara, Mexico, January), Lecture Notes in Computer Science, vol pages 25-30,2004. 3061, Springer-Verlag, Berlin Heidelberg, pages. 126-136,2004.
- [36] Indulska, J., McFadden, T, Kind, M., Henricksen, K. Scalable Location Management for Context-Aware Systems. In Stefani, J.B, Demeure, I.M., Hagimont, D.(eds.) Proceedings of the 4th IFIP WG6.1

International Conference on Distributed Applications and Interoperable Systems (DAIS) Paris, France, November 17-21, 2003. Lecture Notes in Computer Science, vol. 2893, Springer-Verlag, Berlin Heidelberg, pp. 224-235, 2003.

- [37] Dhouha AYED, Déploiement sensible au contexte d'applications à base de composants, Thèse de doctorat de l'Institut National des Télécommunications dans le cadre de l'école doctorale SITEVRY en co-accréditation avec l'Université d'Évry Val d'Essonne, pages 9-20,2005.
- [38] Xu, L., Zhou, S., Zhao, K., Qian, W., Zhou, A. PairBus: A Middleware Framework towards Interoperability among P2P Data Sharing Systems. In: Li, M, Sun, X-H., Deng, Q., Ni, J. (eds.): Proceeding of the 2nd International Workshop in Grid and Cooperative Computing (GCC)), Lecture Notes in Computer Science, vol3032. Springer-Verlag, Berlin Heidelberg pp, 277-284, Shanghai, China , 2003.
- [39] L. Capra, W. Emmerich, and C. Mascolo, CARISMA : Context-Aware Reflective mIddleware System for Mobile Applications. *IEEE Transactions on Software Engineering*, 29(10), 929–945, October 2003.
- [40] J. Dowling and V. Cahill. The k-component architecture meta-model for selfadaptive software. In *Reflection*, 2001.
- **[**41] P. Grace, G. S. Blair, and S. Samuel. Remmoc. A reflective middleware to support mobile client interoperability. In *International Symposium on Distributed Objects and Applications (DOA)*, Catania, Sicily, Italy, November 2003.
- [42] G. S. Blair, G. Coulson, P. Robin, and M. Papathomas.An architecture for next generation middleware.In *Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing*, London, 1998.
- **[**43] J. Keeney and V.Cahill.Chisel.A policy-driven, context-aware dynamic adaptation framework. pages 3–14,In *POLICY* , Italy, 2003.
- [44] A. Fitzpatrick, G. Biegel, S. Clarke, and V. Cahill. Towards a sentient object model. In *Workshop on Engineering Context-Aware Object Oriented Systems and Environments (ECOOSE)*, Seattle, WA, USA, November 2002.
- [45] Oracle technology network.oracle9i application server wireless. http ://technet. oracle.com/products/iaswe/content.html, 2000.
- [46] C. Mascolo, L. Capra, S. Zachariadis, and W. Emmerich.XMIDDLE.A Data-Sharing Middleware for Mobile Computing. *Int. Journal on Personal and Wireless Communications*, 21(1):77–103, April 2002.
- **[47]** Lynda TAMINE-LECHANI et Sylvie CALABRETTO, "Recherche d'information contextuelle et web", publié dans "Recherche d'information, état des lieux et perspectives" pages. 201-224, 2008.
- [48] Gerard Salton, Edward A. Fox, et Harry Wu. Extended booleaninformation retrieval. Commun. ACM, 26(11), pages 1022–1036, 1983.
- [49] Fatiha BOUBEKEUR-AMIROUCHE. Contribution à la définition de modèles de recherche d'information flexibles basés sur les CP-Nets, thèse doctorat, 2008.
- [50] H. P. Luhn. The automatic creation of literature abstracts. IBM Journalof Research and Development, pages 159–165, 1958.
- [51] Gerard Salton, et Michael J. McGill. Introduction to ModernInformation Retrieval. McGraw-Hill, Inc., New York, NY, 1983.
- [52] L.A. Zadeh. Fuzzy sets. Information Control, pages 338–353, 1965.
- [53] J. Lukasiewicz. Elements of Mathematical Logic. Pergamon Press, 1963.
- [54] Zacharis Z. Nick et Panayiotopoulos Themis. Web search using a genetic algorithm. IEEE Internet Computing, 5(2), pages 18–26, 2001.
- [55] George Kingsley Zipf. Human behaviour and the principle of least effort. Addison-Wesley, Cambridge, Massachusetts, 1949.
- [56] C. J. Van Rijsbergen. Information Retrieval, 2nd edition.dept. of Computer Science, University of Glasgow, 1979.
- [57] Stephen E. Robertson, Steve Walker, Micheline Hancock Beaulieu, Aarron Gull,et Marianna Lau. Okapi at TREC. In Text REtrieval Conference, pages 21–30, 1992.
- [58] Patrice Bellot. Méthodes de classification et de segmentation locales on supervisées pour la recherche documentaire. PhD thesis, Université d'Avignon et des Pays de Vaucluse, 2000.
- [59] Fuhr, N, Buckley, CA, probabilistic learning Approach for Document indexing. In ACM Transaction on Information System. Vol. 9, N°3. pp.223-248,1991.

- [60] Turtle H. et Croft BW Evaluation of an inference Network-Based Retrieval Model. InACM Transaction on Information System, Vol. 9, N°3. pp.187-222,1991
- [61] Rijsbergen, C.JA new theorical framework for information retrieval. In proceedings of the 9th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.194-200,1986
- [62] Bruza, P.D. et Lalmas MLogic-Based Inforatiom retrieval .Is it really worth it ? Working Notes of workshop on the treatment of Uncertainty in Logic-Based Models of Information Retrieval Systems, 1995.
- [63] Lalmas, M From a Qualitative towards a Quantitative Representation of Uncertainty in a Situation Theory based model of an Information Retrieval System. Working Notes of the Workshop on the treatment of Uncertainty in Logic-Based Models of Inforatiom retrieval systems,1995.
- [64] Lalmas, Mlogical Models in Information Retrieval. Intreduction and Overview. In Information Processing and Management.vol. 34, N°1.pp. 19-33,1998.
- [65]A.S. Chakravarthy and k.B. Haase. Netserf Using semantic know-Ledge to find internet information archives. In The 18th Annual In-Ternational ACM SIGIR Conference on Research and Deveopment In information Retrieval page 4-11 ,Seattle, WA, July 1995.
- [66] Ellen M.voorhees. Siemens trec-4 report: further experiments With database merging.In the fourth text REtrieval conference (TREC), Gainthersburg, Maryland, 1995.
- [67] Norbert fuhr. A decision-theoretic approach to database selection In networked ir. ACM Transactions on information systems, 17(03):229-249, july1999.
- [68] E.D Katz, M.buther, and R. McGrath. A scalable http server .the ncsa prototype. In *first International* world wide web conference, Geneva, Switzerland, may 1994.
- [69] J. Guyton and M.Schwarz.locating nearby copies of replicated internet servers. Technical Report CU-CS-762-95, University of Colorado at Boulder, 1995.
- [70] R.L. Carter and M.E Crovella. Dynamic server selection using bandwidth probing in wide-area networks. Technical Report BU-CS-96-007, Boston University, march 1996.
- [71].S.Bhattacharjee, M.H. Ammar, E.W.Zegura, V, Shah, and Z.Fei. Application-layer anycasting. In INFOCOM 97, 1997.
- [72] National information Standards Organisation. Z39.50-1992 common command language for online interative information retrieval. NISO Press, Bethesda, 1993.
- [73]A.Negus.development of the euronet-diane common command language. In Third Internationl Online Information Meeting, pages 95-98, 1979.
- [74]ISO. Iso 9777:1993 information and documentation- commands for interctive text searching. In International Organization for stan-dardization, Geneva, Switzerland, 1993.
- [75] Sergey Brin et Lawrence Page. The anatomy of a large-scale hyper-textualWeb search engine. Computer Networks and ISDN Systems, 30(1–7) pages 107–117,1998.
- [76] Kenneth L. Calvert, Matthew B. Doar, et EllenW. Zegura. Modeling internet topology. IEEE Communications Magazine, 35(6),160–163, June 1997.
- [77] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. J. ACM, 46(5) :604–632, 1999.
- [78] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, S.R. Kumar, P. Raghavan, S. Rajagopalan, et A. Tomkins. Hypersearching the web. Scientific American, 280 :54–60, June 1999.
- [79] Allan Borodin, Gareth O. Roberts, Jeffrey S. Rosenthal, et Panayiotis Tsaparas. Finding authorities and hubs from link structures on the world wide web. In Proceedings of the tenth international conference on World Wide Web, pages 415–429. ACM Press, 2001.
- [80] Lawrence Page, Sergey Brin, Rajeev Motwani, et Terry Winograd. The pagerank citation ranking .Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.
- [81] Krishna Bharat et Monika R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In Proceedings of IGIR 98, 21st ACM International Conference on Research and Development in Information Retrieval, pages 104–111, Melbourne, AU, 1998.
- [82] Paul Alexandru Chirita, Daniel Olmedilla, etWolfgang Nejdl. Finding related hubs and authorities. In Proceedings of the 1st Latin-American Web Congress, Santiago, Chile, 2003.
- [82] Andrew Y. Ng, Alice X. Zheng, et Michael I. Jordan. Stable algorithms for link analysis. In Proceedings of the 24th Annual Intl. ACM SIGIR Conference. ACM, 2001.

- [83] R. Lempel et S. Moran. The stochastic approach for link structure analysis (SALSA) and the TKC effect. j-COMP-NET- AMSTERDAM, 33(1–6) .387–401, June 2000.
- [84] Alberto O. Mendelzon et Davood Rafiei. What do the neighbours think? computing web page reputations. IEEE Data Engineering Bulletin, 23(3) .pages 9–16, 2000.
- [85] David Cohn et Huan Chang. Learning to probabilistically identify authoritative documents. In Proc. 17th International Conf. on Machine Learning.pages 167–174. Morgan Kaufmann, San Francisco, CA, 2000.
- [86] Ricardo A. Baeza-Yates et Berthier A. RibeiroNeto. Modern Information Retrieval. ACM Press / Addison-Wesley, 1999.
- [87] Vladimir Meñkov, David J.Neu, et QinShi.Antworld. A collaborative web search tool. DCW proceedings, 2000.
- [88] Koch, F., Rahwan, I. Classification of Agent-based Mobile Assistant. In: Proceedings of the Workshop on Agents for Ubiquitous Computing (UbiAgents04) (Columbia University, New York City, USA) in conjunction with AAMAS, 2004.
- [89] Rahwan, T, Rahwan, T, Rahwan, I., Ashri, R. Agent-Based Support for Mobile Users Using AgentSpeak (L). InGiorgini P., Henderson-Sellers B., Winikoff, M.(eds.).Proceedings of the Workshop on Agent-Oriented Information Systems (AOIS2003) (Melbourne, Australia, July 14, 2003 - Chicago, USA, October 13, 2003). Lecture Notes in Artificial Intelligence, vol. 3030 Springer-Verlag, Berlin Heidelberg (2004), pages 45-60.
- [90] Wooldridge, M., Jennings, N.R. Intelligent Agents: Theory and Practice. In: The Knowledge Engineering Review, vol. 10, pages 115-152,1995,
- [91] Lopez y Lopez, F, Lucj, M., d'Inverno, M. Constraining Autonomy through Norms.In: Proceedings of the 1st International Joint Conferences on Autonomous Agent and Multi-Agent System (AAMAS (Bologna, Italy), ACM Press, New York, NY ,pp. 674-681. 2002.
- [92] Weib, G., Rovatsos, M., Nickles, M. Capturing Agent Autonomy in Roles and XML. In: Proceedings of the Conference on Autonomous Agents and Multi-Agent System (AAMAS) (Melbourne, Australia), ACM Press, New York, NY, USA pp. 105-112. 2003.
- [93] Wooldridge, M., Jennings, N.R. Intelligent Agents: Theory and Practice. In: The Knowledge Engineering Review, vol. 10, no. 2 (1995), pp. 115-152.
- [94] J. Ferber : Les systèmes multi-agents - Vers une intelligence collective. InterEditions, 1995.
- [95] CAGLAYAN A., HARRISON C.,« Les agents », InterEditions ed., ISBN 2225831467, 1998.
- [96] LIEBERMAN H.,« Letizia: an agent that assists web browsing », proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95), Montreal, August, 1995.
- [97] ARMSTRONG R., FREITAG D., JOACHIMS T., « Webwatcher: machine learning and hypertext », Proceedings of the 1995 AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments, Stanford University, California, USA, March 27-29, 1995.
- [98] JACZYNSKI M., TROUSSE B., « Broadway: a world wide web browsing advisor reusing past navigations from a group of users », Proceedings of the 3rd UK Case-Based Reasoning Workshop (UKCBR'97), Manchester, UK, September 9th, 1997.
- [99] PAZZANI M., MURAMATSU J., BILLSUS D., « Syskill & Webert: Identifying interesting web sites », National Conference on Artificial Intelligence, Portland, pp 54-61, 1996.
- [100] Frietas,B .Espinasse, S.Fournier : AGATHE : une architecture générique à base d'agents et d'ontologies pour la collecte d'information sur domaines restreints du Web (2007).
- [101] BALABANOVIC M., SHOHAM Y., « Fab : content-based, coopérativerecommandation », Communications of the ACM, 40(3) ,66-72, March, 1997.
- [102] BUDZIK J., HAMMOND K., « Watson : anticipating and contextualizing information needs », Annual Meeting of the American Society for Information Science (ASIS), Washington, Oct. 31- Nov. 4, 1999.
- [103] MAGLIO P.P., BARRETT R., CAMPBELL C.S, SELKER T, « SUITOR ,an attentive information system », International ACM Conference on Intelligent User Interfaces (IUI), New Orleans, January 9-12, pp 169-176, 2000.
- [104] RÜCKER J, POLANCO M.J, « Siteseer: personalized navigation for the web », Communications of the ACM, 40(3), pp.73-75, March, 1997.

- [105] Cloutier CLOUTIER L., ESPINASSE B., LEFRANÇOIS P., « CAT: a general coordination framework for multi-agent systems », Document de travail 1998-016, Centre de Service, d'Orientation et de Recherche sur la Compétitivité Internationale et l'Ingénierire de l'Entreprise Réseau (SORCIIER), ISBN 2-89524-047-7, 1998.
- [106] CARRE J., MACHONIN A., GLIZE P., « Un système multi-agent auto-organisateur pour l'apprentissage d'un profil utilisateur », Ingénierie des systèmes multi-agents, actes des 7ème journées francophones d'Intelligence Artificielle et Systèmes Multi-Agents (JFIADSMA'99), ISBN 2-7462-0077-5, pp 207-221, 1999.
- [107] Fabien Picarougne, Recherche d'information sur Internet par algorithmes évolutionnaires, thèse 2004.
- [108] Dorigo, M. and Stützle, t. *Handbook of Metaheuristics*, chapter The ant colony optimization metaheuristique: algorithms, applications, and advances. Kluwer, 2003.67, 1995.
- [109] M. Dorigo. Optimization, Learning and Natural Algorithms. PhD thesis, Politecnico di Milano, Italy, 1992.
- [110] E. Bonabeau, M. Dorigo, et G. Theraulaz. Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, New York, 1999.
- [111] L.M. Gambardella et M. Dorigo. Ant-Q, A reinforcement learning approach to the Travelling Salesman Problem. In A. Prieditis et S. Russell, editors, Proceedings of the Twelfth International Conference on MachineLearning.pages 252–260. Morgan Kaufmann, San Mateo, California, 1995.
- [112] T. Stützle et H. Hoos. MAX −MIN Ant System and local search for the Traveling Salesman Problem. In Proceedings of the fourth International Conference on Evolutionary Computation, pages 308–313. IEEE Press, 1997.
- [113] M. Dorigo et L.M. Gambardella. Ant colonies for the Traveling Salesman Problem. BioSystems, 43 :73–81, 1997.
- [114] N. Monmarché, G. Venturini, et M. Slimane. On howPachy-condyla apicalis ants suggest a new search algorithm. Future Generation Computer Systems, 16(8) :937–946, 2000.
- [115] Kenneth A. De Jong and illiam M. Spears. On the state of evolutionary computation. In Proceedings of the 5th International Conference on Genetic Algorithms, pages 618–625, San Francisco, CA, USA, 1993.
- [116] Kenneth A. De Jong. Are genetic algorithms function optimizers? In Reinhard M¨anner and Bernard Manderick, editors, PPSN, pages 3–14. Elsevier, 1992.
- [117] cité Charles Darwin. The Origin of Species by Means of Natural Selection. Mentor Reprints, page 86, 1958.
- [118] D.E. Goldberg. Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley, Reading, MA, USA cité dans, page 87, 124, 217, 1989.
- [119] Darrell Whitley et Thomas Hanson. Optimizing neural networks using faster, more accurate genetic search. In Proceedings of the third international conference on Genetic algorithms, pages Morgan Kaufmann Publishers Inc, 391–396.., 1989.
- [120] Hans-Paul Schwefel. Numerical Optimization of Computer Models. John Wiley & Sons, Inc., New York, NY, USA, cité page 87, 1981.
- [121] L. J. Fogel, A. J. Owens, and M. J.Walsh. Artificial Intelligence through Simulated Evolution. John Wiley, New York, USA, cité page 87 1966.
- [122] J.H. Holland. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI, USA, cité page 86, 87, 87, 881975.
- [123] Dirk Schlierkamp-Voosen et Heinz Mühlenbein. Strategy adaptation by competing subpopulations. In Yuval Davidor, Hans-Paul Schwefel, et Reinhard Männer, editors, Parallel Problem Solving from Nature – PPSN III, pages 199–208, Berlin, 1994. Springer.
- [124] Reiko Tanese. Parallel genetic algorithm for a hypercube. In John J.Grefenstette, editor, Genetic algorithms and their applications,Proc. of the second Int. Conf. on Genetic Algorithms, Hillsdale, Lawrence Erlbaum Assoc,pages 177–183, NJ, 1987..
- [125] Theodore C. Belding. The distributed genetic algorithm revisited. In Proceedings of the 6th International Conference on Genetic Algorithms, Morgan Kaufmann Publishers Inc., pages 114–121,1995.

- [126] J. P. Cohoon, S. U. Hedge, W. N. Martin, et D. Richards. Punctuated equilibria, A parallel genetic algorithm. In John J. Grefenstette, editor, Genetic algorithms and their applications .Proc. of the second Int. Conf. on Genetic Algorithms, Hillsdale, NJ, Lawrence Erlbaum Assoc. pages 148–154, 1987.
- [127] Heinz Mühlenbein. Evolution in time and space - the parallel genetic algorithm. In Gregory J. Rawlins, editor, Foundations of genetic algorithms, Morgan Kaufmann, San Mateo, CA, pages 316–337,1991.
- [128] D.Whitley et T. Starkweather. Genitor II .A distributed genetic algorithm. Journal of Experimental and Theoretical Artificial Intelligence, pages 189–214, 1990.
- [129] Piet Spiessens et Bernard Manderick. A massively parallel genetic algorithm Implementation and first analysis. In Rick Belew et Lashon Booker, editors, Proceedings of the Fourth International Conference on Genetic Algorithms, , San Mateo, CA Morgan Kaufman, pages 279–2861991..
- [130] L. Darrell Whitley. Cellular genetic algorithms. In Proceedings of the 5th International Conference on Genetic Algorithms. Morgan Kaufmann Publishers Inc, page 658 1993.
- [131] Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. A Field Guide to Genetic Programming. Lulu Enterprises, UK Ltd, London, UK, cité page 89, 2008.
- [132]Fernando G. Lobo, Cl´audio F. Lima, and Zbigniew Michalewicz, editors. Parameter Setting in Evolutionary Algorithms, volume 54 of Studies in Computational Intelligence. Springer, cité pages 89-125 2007.
- [133]J.J.Morgan et A.C. Kilgour. Personalising information retrieval using evolutionary modelling. In Proceedings of PolyModel 16.Applications of Artificial Intelligence, pages 142–149, 1996.
- [134] N. Monmarché, G. Nocent, M. Slimane, et G. Venturini. Imagine: a tool for generating HTML style sheets with an interactive genetic algorithm based on genes frequencies. In IEEE International Conference on Systems, Man, and Cybernetics (SMC'99), volume 3, Tokyo, Japan, pages 640–645,October 12-15 1999.
- [135] A. Vakali et Y. Manolopoulos. Caching objects from heterogeneous information sources. In Proceedings International Database Conference (IDC'99), Hong-Kong, July 1999.
- [136] Zacharis Z. Nick et Panayiotopoulos Themis. Web search using a genetic algorithm. IEEE Internet Computing, 5(2) , pages 18–26, 2001.
- [137] Filippo Menczer. Complementing search engines with online web mining agents. Decision Support Syst., 35(2) pages 195–212, 2003.
- [138] C. J. Watkins. Learning from delayed rewards. PhD thesis, Cambridge university, 1989.
- [139] Max Vallin et Juan Manuel Adan Coello. An agent for web in formation dissemination based on a genetic algorithm. In IEEE International Conference on Systems, Man and Cybernetics - IEEE SMC'03, Hyatt Regency, Washington, D.C., USA, IEEE Press, pages 3834–3839, 5 - 8 October 2003.
- [140] LYMAN P, « How much informations 2003 », octobre 2003.
- [141] RHODES B.J., MAES P., «Margin Notes - Building a Contextually Aware Associative Memory », *Proceedings of the International Conference on Intelligent User Interfaces*, New Orleans, LA, 2000
- [142] SARACEVIC T., « The stratefied model of information retrieval interaction: extension and applications », *Proceedings of the* 60th *annual meeting of the American Society for Information Science*, Medford, Etats-Unis, p. 313-327, 1997.
- [143] INGWERSEN P, « Cognitive perspectives of information retrieval interactions .Elements of a cognitive IR Theory », *Annual review of information science and technology*, vol. 52, n° 1, pages 3-50, 1996.
- [144] ALLEN B., « Information seeking in context », *Proceedings of an International Conference on Research in needs, seeking and use in different context*, p. 111-122, 1997.
- [145] INGWERSEN P., JARVELIN K., « Information retrieval in context », *In Proceedings of the* 27th *ACM SIGIR Worshop on information retrieval in context*, p. 6-8, juillet 2004.
- [146] FUHR N., *Information Retrieval, introduction and survey*, Post-Graduate course on Information retrieval, University of Duisburg-Essen, Allemagne, 2000.
- [147] LIU F., YU C., « Personalized Web search for improving retrieval effectiveness », *IEEE Transactions on knowledge Data Engineering*, vol. 16, p. 28-40, 2004.
- [148] SU J., LEE M., « An exploration in personalized and context-sensitive search », *Proceedings of the* 7th *annual UK special interest group for computatonal linguists research colloquium*, 2003.
- [149] BORLUND P., « The IIR evaluation model: A framework for evaluation of interactive information retrieval systems », *Journal of Information Research*, vol. 8, n° 3, p. 152-179, 2003.

- [150] FAN W., GORDON M., PATHAK P., « Discovery of context specific ranking functions for effective information retrieval using genetic programming », *IEEE Transactions on knowledge and data engineering*, vol. 16, n°4, p. 523-527, 2004.
- [151] JEH G., WIDOM J., « Scaling personalized Web search », *Proceedings of the* 12th *International World Wide Web Conference*, 2003.
- [152] CHALLAM V. K. R., Contextual information retrieval using ontology based user profiles, Master's Thesis, 2004.
- [153] MC GOWAN J, The Turn: Integration of Information Seeking and Retrieval in Context, Thesis of Master in Computer Science, Faculty of Science, University College Dublin, 2003.
- [154] WEN J.R., LAO N., MA. W., « Probabilistic Model for Contextual Retrieval », *Proceedings of the* 27th *annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, p. 57-63, 2004.
- [155] MELUCCI M., « Context modeling and discovery using vector space bases », *Proceedings of CIKM 2005*, p. 808-815, 2005.
- [156] RODE H., HIEMSTRA D., « Conceptual Language Models for Context-Aware Text Retrieval », *Proceedings of TREC-13, NIST Special Publication*, 2004.
- [157] FINKELSTEIN L., « Placing Search in Context: The Concept Revisited », *ACM Transactions on Information System*, vol. 20, n° 1, p. 116-131, 2002.
- [158] PRETSCHNER A., GAUCH S., « Ontology Based Personalized Search », *Proceedings of the* 11th *IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, novembre 1999.
- [159] YU S., CAI D., WEN D., MA W., « Improving pseudo-relevance feedback in web information retrieval using web page segmentation », *Proceedings of the World Wide Web conferennce*, 2003.
- [160] HULL R., KUMAR B., SAHUGUET A., MING X., « Have It Your Way .Personalization of Network-Hosted Services », *BNCOD*, p. 1-10, 2002.
- [161]CHURCH K.W, HANKS P, «Word Association Norms, Mutual Information and Lexicography », *Computational Linguistics*, p. 22-29, 1990.
- [162] GLOVER E.J., LAWRENCE S., «Web search - your way », *CACM*, 2000.
- [163] BUDZIK J., HAMMOND K., « User interactions with everyday applications as context for just-in-time information access », *Intelligent User Interface*, p. 41-51, 2000.
- [164] RHODES B., Just-in-time Information Retrieval, PhD Thesis, Massuchessetts Intitute of Technology, 2000.
- [165] ADAR E., KARGER D., ANDREA STEIN L., « Haystack: Per-user information environments », *Proceedings of the 1999 Conference on Information and Knowledge Management (CIKM)*, 1999.
- [166] LIEBERMAN H., « Letizia .An Agent That Assists Web Browsing », *Proceedings of the International Joint Conference IJCAI*, p. 924-929, août 1995.
- [167] ARMSTRONG R., FREITAG D., JOACHIMES T., MITCHELL L., «Web Watcher.A learning apprentice for the World Wide Web », *AAAI Spring Symposium on Information Gathering from Heterogeneous, distributed environments*, 1995.
- [168] PAZZANI M.J., MURAMATSU J., BILLSUS D., « Placing Search in Context .The Concept Revisited », *AAAI/IAAI*, vol. 1, p. 54-61, 1996.
- [169] DUMAIS S., CUTTRELL E., DUMAIS J.J., JANCKE G., SARIN R., ROBBINS D.C., « Stuff I've seen : a system for a personal information retrieval and re-use », *Proceedings of the* 26th *ACM SIGIR*, Toronto, p. 72-79, juillet 2003.
- [170] Huang L., A survey on web information retrieval technologies, Technical report, ECSL, 2000.
- [171] Hawking D., Voorhees E., Craswell N., Bailey P,« Overview of the TREC-8Web Track », *Proceedings of the of the 8th Text REtrieval Conference (TREC-8)*, NIST Special Publication, p. 131-150, 1999.
- [172] Spink A., Wolfarm D., Jansen M. B., Saracevic T., « Searching the Web : the public and their queries », *Journal of American Science on Information and Technology (JASIST)*, vol. 52, n° 3, p. 226-234, 2001.
- [173] Berrani S.-A., Amsaleg L., Gros P., « Recherche par similarités dans les bases de données multidimensionnelles : panorama des techniques d'indexation », *Ingénierie des systèmes d'information*, vol. 7, n° 5-6, p. 9-44, 2002.
- [174] Frieder O., Grossman D. A., Chowdhury A., Frieder G., « Efficiency considerations for scalable information retrieval servers », *Journal of digital information*, 2000.

- [175] Voorhees E., Harman D., « Overview of the Eighth Text REtrieval Conference », *Proceedings of the 8th Text REtrieval Conference (TREC-8)*, NIST Special Publication, p 1-24, 1999.
- [176] Moffat A., Zobel J., « Self-Indexing Inverted Files for Fast Text Retrieval », *ACM Transactions on Information Systems*, vol. 14, n° 4, p. 349-379, 1996.
- [177] Scholer F.,Williams H. E., Yiannis J., Zobel J., « Compression of inverted indexes for fast query evaluation », *Proceedings of the 25th ACM SIGIR Conference on research and development in information Retrieval*, p. 11-15, August, 2002.
- [178] Heinz S., Zobel J., « Efficient single pass index construction for text databases », *Journal of American Science on Information and Technology (JASIST)*, vol. 54, n° 8, p. 713-729, 2003.
- [179] Zobel J., « How reliable are the results of large scale information retrieval experiments », *Proceedings of the 21th ACM SIGIR Conference on research and development in information Retrieval*, August, p. 307-314, 1998.
- [180] IMAFOUO A., BEIGBEDER M., « Evaluer le passage à l'échelle dans des environnements à pertinence multivaluée ». 27th European Conference on Information Retrieval, 2005.
- [181] SARACEVIC T, « Relevance ,A review of and a framework for the thinking on the notion in information science », Journal of the American Society for information Science, vol. 26, 1975, p. 321-343.
- [182] WILSON P « Situational relevance », Information Storage and Retrieval, vol. 9, no 8, p. 457-471. 1973
- [183] KEKÄLÄINEN J., JÄRVELIN K., « Using graded relevance assessments in IR evaluation », Journal of the American Society for Information Science and Technology, vol. 53, no 13, , p. 1120 - 1129. 2002.
- [184] KEKÄLÄINEN J., JÄRVELIN K., « IR evaluation methods for retrieving highly relevant documents », Proceedings of the 23th annual international ACM SIGIR Conference, p. 41-48, 2000.
- [185] KAZAI G., LALMAS M., « Notes on what to measure in INEX », INEX Workshop on Element Retrieval Methodology, 2005.
- [186] HAWKING D., S.ROBERTSON, « On collection size and retrieval effectiveness », Information retrieval, vol. 6, no 1, p. 99-105,2003.
- [187] Berry M. W., « Large-Scale Sparse Singular Value Computations »,*The International Journal of Supercomputer Applications*, Spring, vol. 6, n° 1, p. 13-49, 1992.
- [188] Dumais S., « LSI meets TREC : A status report », *Proceedings of the 1st Text REtrieval Conference (TREC-1)*, NIST Special Publication, p. 137-152, March, 1993.
- [189] Kokiapoulou E., Saad Y., « Polynomial filtering in latent semantic indexing for information retrieval», *Proceedings of the 27th annual international ACM SIGIR Conference on research and development in information retrieval (SIGIR)*, Sheffield, U.K., p. 104-111, July 25-29, 2004.
- [190] Tang C., Dwarkadas S., Xu Z., « On scaling latent semantic indexing for large peer to peer systems», *Proceedings of the 27th annual international ACM SIGIR Conference on researchand development in information retrieval (SIGIR)*, Sheffield, U.K., p. 112-121, July 25-29,2004.
- [191]Lehoucq R., Sorensen D,« Deflation techniques for an implicitly restarted Arnoldi iteration », *SIAM Review*, vol. 17, p. 789-821, 1996.
- [192] Sameh A. H, Wisniewski J. A., « A trace minimisation algorithm for the generalized Eigenvalue problem », *SIAM Review*, vol. 19, n° 3, p. 1243-1259, 1982.
- [193] Zobel J., Moffat A., Sacks-Davis R., « An efficient indexing technique for full-text database systems », *Proceedings of the 8th International conference on very large databases (VLDB)*, p. 352-362, 1992.
- [194] C.-H. Wei and C.-T. Li, "A next generation search: Content-based multimedia retrieval," In: M. Pagani (Ed), Encyclopedia of Multimedia Technology and Networking, 2nd Edition, Idea Group Publishing: Hershey, PA, USA, 2008.
- [195] V. Gouet-Brunet, M. Manouvrier et M. Rukoz. Synthèse sur les modèles de représentation des relations spatiales dans les images symboliques. Rapport scientifique CEDRIC, (ref. CEDRIC 1325) pages51, 2007.
- [196]S-A Berrani, L. Amsaleg and P. Gros. Recherche par similarité dans les bases de données multidimensionnelles:
- panorama des techniques d'indexation. RSTI - Ingénierie des systèmes d'information. Bases de données et multimédia, 7(5-6) pages 9-44, 2002.
- [197] H. Samet, Foundations of Multidimensional And Metric Data Structures, Morgan Kaufmann Pub. 2006.

- [198] I. Ahmad and W. I. Grosky. Indexing and retrieval of images by spatial constraints. Journal of Visual Communication and Image Representation, 14(3), pages 291–320, 2003
- [199] H.-K Kim and J.-D. Region-based shape descriptor invariant to rotation, scale and translation SignalProcessing: Image Communication, 16(1–2), pages 1–293, 2000
- [200]Y.Wang, F. Makedon. R-Histogram: quantitative representation of spatial relations for similarity-based image retrieval. ACM Multimedia, pages 323-326,2003
- [201]Punitha, P. et D. Guru. An Effective and Efficient Exact Match Retrieval Scheme for Symbolic ImageDatabase Systems Based on Spatial Reasoning .A Logarithmic Search Time Approach. IEEE Trans. On Knowledge and Data Eng. 18(10), 1368–1381, 2006.
- [202]Jaume Amores, Nicu Sebe, Petia Radeva, Fast Spatial Pattern Discovery Integrating Boosting with Constellations of Contextual Descriptors, CVPR'05, pp 769-774, 2005
- [203] N. Bouteldja and V. Gouet-Brunet. Exact and progressive image retrieval with the HiPeR framework. In IEEE International Conference on Multimedia & Expo, Hannover, Germany, 2008
- [204] Scholer F, Williams H. E., Yiannis J., Zobel J., « Compression of inverted indexes for fast query evaluation », *Proceedings of the 25th ACM SIGIR Conference on research and development in information Retrieval* August, p. 11-15, 2002.
- [205] Frieder O., Grossman D. A., Chowdhury A., Frieder G., « Efficiency considerations for scalable information retrieval servers », *Journal of digital information*, 2000.
- [206] Elias P., « Universal codeword sets and representations of the integers », *IEEE transactions on information theory*, vol. 21, n° 2, p. 194-203, 1977.
- [207] Zobel J., Moffat A., Sacks-Davis R., « An efficient indexing technique for full-text database systems », *Proceedings of the 8th International conference on very large databases (VLDB)*, p. 352-362, 1992.
- [208] Shieh W.-Y., Chen T.-F., Shann J. J.-J., Chung C.-P., « Inverted file compression through document identifier reassignment », *Inf. Process. Manage.*, vol. 39, n° 1, p. 117-131, 2003.
- [209] Beigbeder M., Mercier A., « Étude des distributions de *tf* et de *idf* sur une collection de 5 millions de pages html », *Atelier de recherche d'information sur le passage à l'échelle,congrès INFORSID*, Nancy, France, June, 2003.
- [210] Bayley P., Hawking D., A Parallel architecture for query processing over a Terabyte of text, Technical report, The Australian National University, 1996.
- [211] CUGINI J.V., LASKOWSKI S., SEBRECHTS M., « Design of 3-D visualisation of search results: evolution and evaluation », 12th International Symposium on Electronic Imaging: Visual Data Exploration & Analysis (SPIE 2000), San Jose, CA, pp 198-210, January 23-28, 2000.
- [212] DUBIN D., « Document analysis for visualization », 18th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp 199-204, 1995.
- [213] ZAMIR, O., « Visualisation of search results in document retrieval systems », General Examination, University of Washington, 1998.
- [214] CHI Ed. H., « A taxonomy of visualization techniques using the data state reference model », INFOVIS, Salt Lake City, pp 69-75, October, 2000.
- [215] HASCOËT M., BEAUDOUIN-LAFON M., « Visualisation interactive d'information », Revue Information-Interaction-Intelligence (I3), « A Journal in Information Engineering Sciences », 1(1), 2001.
- [216] Hendley R. J., Drew N. S., Wood A. M., Beale R., « Narcissus : Visualizing Information »,*in* N. Gershon, S.G.Eick (eds), *Proceedings of Information Visualization Symposium*, IEEE CS Press, Los Alamitos, California, p. 90-96, 1995.
- [217] Nigay L., Vernier F., « Design method of interaction techniques for large information spaces »,*proceedings of the international conference on Advanced Visual Interfaces (AVI'98)*,L'Aquila, Italy, p. 37-46, May, 1998.
- [218] Chalmers M., Chitson P., « Bead : explorations in information visualization », *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM Press, Copenhagen, Denmark, p. 330-337, 1992.
- [219] Hearst M. A., Karadi C., « Cat-a-Cone : an interactive interface for specifying searches and viewing retrieval results using a large category hierarchy », *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, Philadelphia, Pennsylvania, United States, p. 246-255, 1997.

- [220] Chevallet J.-P., Nigay L., « Les interfaces pour la Recherche d'Information », *in* C. Paganelli (ed.), *Interaction homme-machine et recherche d'information*, Hermès Science, Lavoisier, chapter 2, p. 65-102, 2002.
- [221] Zobel J., « How reliable are the results of large scale information retrieval experiments », *Proceedings of the 21th ACM SIGIR Conference on research and development in information Retrieval*, August, p. 307-314, 1998.
- [222]Calvary G, Coutaz J., Thevenin D, « A Unifying Reference Framework for the Development of Plastic User Interfaces », *Proceedings of EHCI'01,IFIP WG2.7 (13.2) Conference*, Springer-Verlag, Toronto, Canada, May, 2001.

- [223] Brian Brewington, Robert Gray, Katsuhiro Moizumi, David Kotz, George Cybenko and Daniela Rus "Mobile agents in distributed information retrieval".
- [224] drogoul, A. and ferber, j From tom thumb to the dockers, some experiments with foraging robots. In proceedings of the second International conference on Simulation of Adaptative Behaviour, p 451-459, 1992.
- [225] Fuggetta A., Picco G., Vigna G., « Understanding Code Mobility », IEEE Transactions on Software Engineering, vol. 24, n 5, p. 342-361. 1998.
- [226]B. Brewington, R. Gray, K. Moizumi, D. Kotz, G. Cybenko, D. Rus: Mobile agents in distributed information retrieval. In Intelligent Information Agents. Springer-Verlag (1999).
- [227]S. Papastavrou, G. Samaras, E. Pitoura: Mobile Agents for World Wide Web Distributed Database Access, IEEE Transactions on Knowledge and Data Engineering, (2000).
- [228] H. kosch, M Doller, L. Boszormenyi ,Content-based indexing and retrival supported by mobile agent technology 2001 .
- [229] Groot, D.d.,boonk,M.,brazier,F.,and Oskamp,A.(2005).Issues in a mobile agent-based multimedia retrieval scenario.In Proceedings of the 4th workshop on the law and Electronic agents ,LEA,pages 33_34,2005.
- [230] Reval, A. (2003). Web-agents inspired by ethology ,a population of « ant »-like agents to help finding user-oriented information. In IEEE WIC'2003 : international Conferance on Web Intelligence .pages 482-285,Haifax, Cannada.IEEE ,IEEE computer Society.
- [231]V. Camps, M. P. Glelzes : Une technique multi-agent pour rechercher des informations réparties. Institut de Recherche en Informatique de Toulouse. Actes des cinquièmes Journées Francophones IAD et SMA, Editions Hermès, Avril (1997).
- [232]J. P. Arcangeli, V. Hennebert, S. Leriche, F. Migeon, M. Pantel : JavAct 0.5.0 : principes, installation, utilisation et développement d'applications. Rapport de recherche, IRIT/2004-5-R, Février 2004.
- [233] M. Côté, N. Troudi : NetSA : Une architecture multiagent pour la recherche sur Internet. Université Laval. Département d'informatique. Pavillon Pouliot. Ste-Foy, Canada 1998.

- [234]S. J. Pelletier, S. Pierre, H. H. Hoang: Modeling a Multi-Agent System for Retrieving Information from Distributed Sources. CIT "Computing and Information Technology", 2003.
- [235] Roth, V, Pinsdorf. U., and peters, J. A dstributed content-based Search engine based on mobile code. In SAC'05 Proceedings of the 2005 ACM Symposium on Applied computing, New York, NY, USA.ACM Press, pages 66-73, 2005.
- [236] H. Kosch, M. Doller et L. Boszormenyi, Content-based indexing andretrieval supported by mobile agent technology. In Second InternationalWorkshop on Multimedia Databases and Image Communication, pages 152–166, 2001.
- [237] C.Tschudin .Mobile agent security. In Intelligent Information Agents. Springer-Verlag, 1999.
- [238] M. Oussalah : Ingenierie des Composants : Concepts, techniques et outils. Vuibert Informatique, 2005. Ouvrage collectif.

- [239] V. Issarny, F. Tartanoglu, J. Liu, F. Sailhan. *Software Architecture for Mobile Distributed Computing*. Proceedings of the 4th IEEE/IFIP Conference on Software Architecture (WICSA) 2004.

- [240] R. Allen, D. Garlan*A Formal Basis for Architectural Connection*. ACM Transactions on Software Engineering and Methodology, 1997.
- [241] P. A. Bernstein: Middleware: A model for distributed services. In Communications of the ACM, pages 86–97, 1996.
- [242] O. Babaoglu, H. Meling et A. Montresor Anthill A framework for the development of agent-based peer-to-peer systems. Rapport technique UBLCS-2001-09, Dept. of Computer Science, Univ. of Bologna, Italy, 2002.
- [243] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. V. Lopes, J. Loingtier et J. Irwin Aspect-oriented programming. In ECOOP 1997 - LNCS 1241, 1997.
- [244] B. Smith Reflection and Semantics in a Procedural Programming Language. Thèse de doctorat, MIT, Boston, MA, US, January 1982.
- [245] F. Boyer et O. Charra : Utilisation de la réflexivité dans les plate-formes adaptables pour applications réparties. In Revue électronique sur lesRéseaux et l'Informatique Répartie, 2001. ISSN 1262-3261.
- [246] P. Maes Concepts and experiments in computational reflection. In OOPSLA '87 Conference proceedings on Object-oriented programming systems, languages and applications, ACM Press, NY, USA pages 147–155., 1987.
- [247] D. Chefrour et F. André : Aceel : modèle de composants auto-adaptatifs application aux environnements mobiles. In Journées Systèmes à composants adaptables et extensibles, 2002.
- [248] E. Gamma, R. Helm, R. Johnson et J. Vlissides Design patterns Elements of Reusable Object-Oriented Software. Addison-Wesley Professional Computing Series, 1995.
- [249]V.Issarny. Configuration-Based Programming Systems. Proceedings of SOFSEM'97: Theory and Practice of informatics, .Springer-Verlag, pages 183-200, 1997.
- [250] N. Medvidovic, R. N. Taylor. A Classification and Comparison Framework for Software Architecture Description Languages. IEEE Transactions on Software Engineering, vol. 26, no. 1, pp. 70-93,2000.
- [251] R. Allen. A Formal Approach to Software Architecture. PhD thesis, Carnegie Mellon, School of Computer Science. Issued as CMU Technical Report CMU-CS-97-144, 1997.
- [252] R. Allen, D. GarlanA Formal Basis for Architectural Connection. ACM Transactions on Software Engineering and Methodology, 1997.
- [253] R. J. Allen, R. Douence, D. GarlanSpecifying and Analyzing Dynamic Software Architectures. Proceedings of the 1998 Conference on Fundamental Approaches to Software Engineering (FASE '98), 1998.
- [254] P. Oreizy, N. Medvidovic, R.N. TaylorArchitecture-Based Runtime Software Evolution. Proceedings of 21st International Conference Software Engineering (ICSE '98), pp. 177-186, 1998.
- [255] J. Magee, J. KramerDynamic Structure in Software Architectures. Proceedings of ACM SIGSOFT '96: Fourth Symposium Foundations of Software Eng. (FSE4), pp. 3-14, 1996.
- [256] G. Abrami, « Niveaux d'organisation dans la modélisation multi-agent pour la gestion de ressources renouvelables, Application à la mise en œuvre de règles collectives de gestion de l'eau agricole dans la basse-vallée de la Drôme », p. 104-123. 2004.
- [257] Gutknecht, O., Ferber, J., et Michel, F. « From agents to organizations: an organizational view of multi-agent systems», Agent-Oriented Software Engineering IV, 2935 p. 214–230. 2004.
- [257] J. Callan, "Distributed information retrieval". In Advances in Information Retrieval, W. B. Croft, Ed. Kluwer Academic Publishers, pages,127–150. 2000.
- [258] D. Hagimont, L. Ismail, « Agents mobiles et client/serveur: évaluation de performance et comparaison. », Technique et science informatiques, vol. 19, n° 9, 2000.
- [260] Joonhee Kwon and Sungrim Kim, Ubiquitous Information Retrieval Using Multi-level Characteristics, D. Lowe and M. Gaedke (Eds.): ICWE 2005, LNCS 3579, in Springer-Verlag Berlin Heidelberg. pp. 167-172, 2005.

Résumé

L'informatique ubiquitaire devient aujourd'hui une réalité, grâce à la mise en réseau d'un nombre croissant de dispositifs informatiques par le biais de technologies réseaux et de l'Internet. Le Web ubiquitaire est vu alors, comme une synthèse de l'Internet et de l'informatique ubiquitaire qui étend le Web existant, la coordination de dispositifs mobiles, la sensibilité au contexte, l'adaptation à l'hétérogénéité des réseaux et des sources d'information et également, la mise en disponibilité des nouvelles technologies de l'information et de la communication au service des utilisateurs, à n'importe quel moment, à partir de n'importe quel endroit et avec n'importe quel dispositif mobile. Cependant, rechercher de l'information pertinente, dans un volume important du Web ubiquitaire est une tache à la fois cruciale et très complexe.

Ce mémoire s'intéresse principalement à deux aspects : d'une part, celui de la sensibilité au contexte, quoi doit fournir un système de recherche d'information dans le Web ubiquitaire. En fait, cet aspect inclut, l'adaptation de l'information retournée à l'utilisateur en fonction de la mobilité et de changement très important du contexte de la recherche dans l'environnement ubiquitaire d'exécution. Et d'autre part, celui de l'augmentation de la charge du réseau. Cet aspect englobe, la croissance massive des informations véhiculées à travers les réseaux, à cela, vient s'ajouter un surcroît par les informations contextuelles qu'un système de recherche d'information dans le Web ubiquitaire doit les traiter.

Pour cela, nous avons pris en compte le contexte de la recherche comme étant une multitude de niveau de caractéristiques qui influence directement la qualité et la vitesse d'extraction de l'information pertinente. De plus, nous avons proposé une architecture basée rôle et agent mobile. Nous avons présenté, également, un modèle d'agent mobile à base de rôle adaptable, sensible au contexte dynamiquement, pour s'adapter aux variations de son environnement de recherche ubiquitaire. Le comportement de l'agent mobile de recherche proposé est construit principalement à partir d'un assemblage de rôles de recherche de granularité faible, il s'agit, de micro-composants remplaçables dont leur configuration se réalise en fonction de la variabilité du contexte de la recherche. Ainsi, l'agent peut se déplacer à travers les réseaux avec un ensemble restreint de rôles de recherche étant donné qu'à tout serveur d'information, il existe une base spécifique de rôles de recherche. De cette façon, l'agent visitant un serveur sélectionne sur la base de ses paramètres de recherche les rôles adéquats, à fin d'enrichir son comportement et de même, il peut contribuer à l'évolution de cette base de rôle par déposition de nouveaux rôles, ou bien par adaptation des rôles existants. Ceci réduit la charge au niveau de l'agent mobile et donc le trafic de son déplacement, et par conséquent, réduction de la charge totale des réseaux.