Méthodologie de développement des systèmes d'information personnalisés : Application à un système d'information au service des usagers des transports terrestres de personnes

Abdouroihamane Anli

LAMIH
LABORATOIRE D'AUTOMATIQUE DE MECANIQUE ET D'INFORMATIQUE INDUSTRIELLES ET HUMAINES

Numéro d'ordre : 06/29

THÈSE

pour l'obtention du

Doctorat de l'Université de Valenciennes et du Hainaut-Cambrésis

Spécialité Automatique et Informatique des Systèmes Industriels et Humains

Mention Informatique

présentée par

Abdouroihamane ANLI

(Maître ès Sciences)

Méthodologie de développement des systèmes d'information personnalisés

Application à un système d'information au service des usagers des transports terrestres de personnes

Soutenue publiquement le 20 octobre 2006 devant le jury composé de :

Bernard Espinasse	Professeur à l'Université d'Aix-Marseille	Rapporteur
Guy Gouardères	Professeur à l'Université de Pau et des Pays de l'Adour	Rapporteur
Arnaud Fréville	Professeur, Conseil Régional Nord-Pas de Calais	Examinateur
Christophe Kolski	Professeur à l'UVHC	Directeur
Emmanuelle Grislin-Le Strugeon	Maître de Conférences à l'UVHC	Co-directeur
Mourad Abed	Professeur à l'UVHC	Co-directeur
Mongi Zidi	Président Directeur Général, Archimed S.A	Invité
Guillaume Uster	Chargé de Recherche, INRETS, Villeneuve d'Ascq	Invité

UVHC

Université de Valenciennes et du Hainaut-Cambrésis
Le Mont Houy, F-59313 Valenciennes cedex 9

CNRS

LAMIH
LABORATOIRE D'AUTOMATIQUE DE MECANIQUE ET D'INFORMATIQUE INDUSTRIELLES ET HUMAINES

Numéro d'ordre : 06/29

THÈSE

pour l'obtention du

Doctorat de l'Université de Valenciennes et du Hainaut-Cambrésis

Spécialité Automatique et Informatique des Systèmes Industriels et Humains

Mention Informatique

présentée par

Abdouroihamane ANLI

(Maître ès Sciences)

Méthodologie de développement des systèmes d'information personnalisés

Application à un système d'information au service des usagers des transports terrestres de personnes

Soutenue publiquement le 20 octobre 2006 devant le jury composé de :

Bernard Espinasse	Professeur à l'Université d'Aix-Marseille	Rapporteur
Guy Gouardères	Professeur à l'Université de Pau et des Pays de l'Adour	Rapporteur
Arnaud Fréville	Professeur, Conseil Régional Nord-Pas de Calais	Examinateur
Christophe Kolski	Professeur à l'UVHC	Directeur
Emmanuelle Grislin-Le Strugeon	Maître de Conférences à l'UVHC	Co-directeur
Mourad Abed	Professeur à l'UVHC	Co-directeur
Mongi Zidi	Président Directeur Général, Archimed S.A	Invité
Guillaume Uster	Chargé de Recherche, INRETS, Villeneuve d'Ascq	Invité

UVHC

Université de Valenciennes et du Hainaut-Cambrésis
Le Mont Houy, F-59313 Valenciennes cedex 9

CNRS

AVANT PROPOS

Le travail présenté dans ce mémoire a été réalisé au Laboratoire d'Automatique, de Mécanique et d'Informatique industrielles et Humaines (LAMIH) de l'Université de Valenciennes et du Hainaut-Cambrésis (UVHC), au sein de l'équipe Raisonnement Automatique et Interaction Homme-Machine (RAIHM), dirigé par le Professeur Christophe Kolski, en collaboration avec la société Archimed S.A dans le cadre d'une bourse co-financée par la région Nord-Pas de Calais et Archimed.

Je suis très reconnaissant envers Messieurs Bernard Espinasse, Professeur à l'Université d'Aix-Marseille et Guy Gouardères, Professeur à l'Université de Pau et des Pays de l'Adour, pour m'avoir fait l'honneur d'être rapporteurs de ce mémoire. Je suis également très reconnaissant envers Monsieur Arnaud Fréville, Professeur, Directeur de la Recherche, de l'Enseignement Supérieur, de la Santé et des Technologies de l'Information et de la Communication, pour avoir accepté d'examiner ce travail.

Cette thèse n'aurait pas pu avoir lieu sans Christophe Kolski qui m'a accueilli au sein de son équipe et Mongi Zidi, Président Directeur Général de la société Archimed, co-financeur de la thèse. Je les en remercie très profondément.

Je tiens également à remercier Emmanuelle Grislin-Le Strugeon, maître de conférences à l'UVHC, qui m'a guidé avec patience et pertinence le long de ces travaux. J'associe à ces remerciements Mourad Abed, professeur à l'UVHC, pour ses encouragements et ses conseils avisés. Mes remerciements vont également à Guillaume Uster, chargé de recherche à l'Institut National de REcherche sur les Transports et leur Sécurité, pour ses avis en termes d'information transport.

Je tiens à remercier tous les collaborateurs de la société Archimed pour leur sympathie et leur bonne humeur. Je remercie particulièrement Olivier Walbecq, Directeur Général, pour son accueil chaleureux et Christian Serrure, Directeur de la division Recherche & Développement, pour m'avoir accepté au sein de la division R&D. Je tiens à remercier Emmanuel Bleuse (Mr MASC), Cédric Lazzerini (Mr XSL), Olivier Chèvre (Mr Charte graphique), Frédéric Ryckman (Mr SIM), Guillaume Liekens (Mr C#), Batiste Deschamps (Mr SARA) et Jérome Penez (Mr GED) pour avoir répondu à mes nombreuses questions. Je n'oublie pas les collaborateurs qui ont rendu très agréable mon travail à Archimed : Christophe Hallard, Paul-Marie Druesne, Ingrid Luera, Céline Dauchy, Nicolas Faillon, Khalifa Belgacem, Thomas Charle, Fabrice Gerometta, Loïc Delambre, Stéphane Lalande, Denis Tep.

Je remercie également les chercheurs, ingénieurs et administratifs du LAMIH de leurs encouragements et de leur amitié. Je remercie Salah Bousbia de notre collaboration pour *l'agentification* d'un atelier de production afin de le rendre auto-adaptatif, Philippe Dos Santos et Gérald Conreur de notre collaboration dans le projet MOUVER.PERSO et Mohamed-Amine Maalej pour son amitié et pour nos discussions sur les réseaux bayésiens. Merci à Emmanuel Adam de sa sympathie et de ses remarques pertinentes sur mes travaux. J'associe également à ces remerciements les personnes successives avec qui j'ai partagé le bureau 127 : Christelle Petit-Rozé, Jingxuan Ma et Sophie Lepreux. Nos discussions ont toujours été très enrichissantes. Merci à David Hanon pour nos déjeuners sympathiques. Mes remerciements vont également à Abdelwaheb Trabelsi et Karim Ammous pour leur amitié et leurs encouragements.

Je remercie les différents étudiants et stagiaires pour l'intérêt qu'ils ont porté à mes sujets d'études ou de recherche. Ils ont apporté une contribution précieuse à ces travaux. Je pense notamment à Dolama Dikpina (IUP1 et IUP2, UVHC), Ibrahim ben Lazreg (DEA, LAMIH), Hacène Beche (DESS, UVHC), Julien Esposito (3ème année ENSIAME, UVHC), Emmanuel Boidin (DEA LAMIH), Talel Ben Ayed (Stagiaire, Archimed), Simona Leo (Stagiaire, Archimed) et Yousra Gassara (Stagiaire, LAMIH).

Enfin, je remercie toute ma famille et tous mes amis pour leurs encouragements et leur soutien. Je remercie mes parents de m'avoir tout sacrifié et comblé d'amour. Je leur exprime ma profonde reconnaissance pour m'avoir soutenu dans tous mes projets. Je remercie bien sûr Dania, mon épouse, pour sa grande patience, sa compréhension et son soutien.

A ma femme,

A mes parents,

A ma famille,

A mes amis.

TABLE DES MATIERES

LISTE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION GENERALE

Les technologies de l'information et de la communication offrent la possibilité d'accéder à des masses d'information de plus en plus grandes sur des supports de plus en plus variés et supportant des modes d'interaction de plus en plus différents. Un utilisateur peut rechercher et/ou recevoir de l'information sur son ordinateur de bureau, son PDA, son téléphone portable, etc. La plupart de ces plates-formes d'accueil permettent une interaction multi-modale combinant le son, l'image et le texte.

Pour répondre au critère d'utilisabilité [Nielsen 93] et permettre à l'utilisateur d'être mis en présence facilement de l'information pertinente qui l'intéresse et le concerne personnellement, la personnalisation se présente comme une solution appropriée. Cette notion de personnalisation s'inscrit dans le prolongement d'un courant de recherche très actif depuis le début des années 80 en interaction homme-machine, visant de nouvelles interactions de plus en plus adaptatives, intelligentes [Schneider-Hufschmidt et al., 93][Kolski et Le Strugeon 98][Höök 00]. Au départ, par son approche centrée sur l'adaptation à l'utilisateur, la personnalisation en Interaction Homme-Machine peut être vue aussi comme très complémentaire aux recherches actuelles sur la plasticité mettant pour l'instant le plus souvent l'accent sur l'adaptation à la plateforme et à l'environnement d'interaction [Thévenin et Coutaz 99][Calvary et al., 03][Kolski et al., 04]. En fait, outre la personnalisation des informations délivrées, d'autres aspects de l'interaction peuvent faire l'objet de personnalisation dans ce cadre, tels que la prise en compte des différents modes d'interaction (vocal, textuel, braille, …), la prise en compte des différentes plates-formes d'interaction (PC, PDA, téléphone portable, …), et l'assistance à l'utilisateur.

La prise en compte des modes d'interaction et des plates-formes utilisées signifie, pour nous, que le système de personnalisation doit les intégrer dans ses raisonnements par rapport aux préférences de l'utilisateur. Le système de personnalisation doit être capable de dire, par exemple, que pour telle information, l'utilisateur préférera l'écouter à partir de son téléphone portable, plutôt que de la lire sur son PDA.

L'objectif principal de nos travaux est de proposer une méthode adaptée au développement des systèmes d'information personnalisés. Cette méthode devrait permettre et faciliter aussi bien le développement d'un nouveau système d'information que la personnalisation d'un système d'information existant. Elle devrait aussi prendre en compte ces différents aspects de personnalisation décrits ci-dessus en favorisant la construction de système de personnalisation évolutif et distribué.

Ces travaux s'inscrivent dans le cadre d'une collaboration entre l'équipe « Raisonnement Automatique et Interaction Homme-Machine » (RAIHM) dirigée par le Professeur C. Kolski, l'Institut National de REcherche sur les Transports et leur Sécurité (INRETS) de Villeneuve d'Ascq (G. Uster dans l'équipe ESTAS dirigée par G. Couvreur) et la société Archimed de Lille (Président Directeur Général M. Zidi). L'objectif applicatif de ces travaux est de concevoir un système d'information transport pour faciliter l'accès à l'information voyageur, intégrant plusieurs modes de transport, d'un usager désirant préparer un déplacement. Ces travaux se situent dans la lignée des recherches communes INRETS-ESTAS/LAMIH-RAIHM, amorcées lors du DEA puis de la thèse de C. Petit-Rozé [Petit-Rozé 03], se basant sur les Systèmes Multi-Agents (SMA) pour la personnalisation de l'information.

Dans le domaine des transports, et comme le soulignait la LOTI (Loi d'Orientation des Transports Intérieurs) dès 1982, l'avenir devra être au « *développement harmonieux et complémentaire des divers modes de transport* ». L'information transport intégrant plusieurs modes de transport est vraisemblablement un des outils pour atteindre ce but idéal. Si «éclairer le choix modal» et «faciliter l'usage des réseaux» sont les deux objectifs assignés à un système d'information transport intégrant plusieurs modes de transport, il n'en demeure pas moins que l'idée principale reste la mise à disposition auprès des usagers, et de manière simple, de « signaux chauds » (selon une étude de Marc Gilles Consultants [Gilles 97]).

Au travers de ce terme, on entrevoit la réalité d'une aide au déplacement par la pertinence de l'information délivrée. L'usager des transports ne souhaite en effet avoir à disposition que peu d'informations, juste celles qui l'intéressent directement. Un système d'information, comme son nom l'indique, est destiné à fournir de l'information à un utilisateur. En fait, il devrait, idéalement, permettre à l'utilisateur de récupérer de l'information à partir des données auxquelles a accès le système. Or, cette transformation des données en information, à savoir cette plus-value apportée aux données qui sont triées, classées, validées et personnalisées, est bien souvent négligée : le système laisse à l'utilisateur la charge de retrouver l'information qui l'intéresse dans la masse de données qui lui est fournie.

Nous essayons de remédier à cela dans le contexte de l'information aux usagers. En effet, l'usager se trouve confronté à un ensemble de sources d'informations disparates (horaires et tarifs délivrés par les différents exploitants, cartes, etc., sur papier, borne interactive ou par Internet) qui s'avèrent parfois difficiles à intégrer en un plan de déplacement précis et unique, adapté à ses besoins et préférences. Dans le contexte de l'information aux usagers intégrant plusieurs modes de transports et leurs connexions, notre objectif est, d'une part, d'aider l'usager dans sa démarche de recherche d'informations et, d'autre part, de lui fournir un résultat personnalisé, c'est-à-dire toute l'information nécessaire et uniquement l'information nécessaire en fonction de son destinataire.

Le premier chapitre présente l'état de l'art des travaux relatifs à la personnalisation. Il explicite les principaux termes rencontrés dans le domaine et expose les principes, approches et méthodes utilisées pour la personnalisation de l'interaction homme-machine. Différents systèmes de personnalisation sont étudiés suivant des critères définis par rapport à un objectif de prise en compte de différents aspects de personnalisation, de réutilisation de l'existant, de support de la distribution par le système et de son évolutivité. Les notions d'agents et de Système Multi-Agents y sont abordées puisque la majorité des systèmes de personnalisation répondant aux critères définis sont à base d'agents logiciels. Ce chapitre présente aussi les systèmes d'information transport existants étudiés sous l'angle de la personnalisation.

Le deuxième chapitre a pour but la recherche d'une méthode pour l'analyse, la conception et la modélisation des systèmes d'information personnalisés. Bien que de nombreux travaux soient effectués pour la personnalisation, il n'existe pas, à notre connaissance, de méthode de génie logiciel adaptée pour la mise en place d'un système d'information personnalisé répondant à nos contraintes de réutilisation de l'existant, de prise en compte de différents types de personnalisation tout en favorisant la construction de système de personnalisation évolutif et distribué. Pour cela, ce chapitre commence par étudier les architectures des systèmes existants qui se rapprochent de nos objectifs. Ensuite, une étude des méthodes représentatives des domaines d'application susceptibles d'intervenir dans le développement de système d'information personnalisé est effectuée dans une perspective d'adaptation et d'intégration de ces méthodes pour la construction d'une nouvelle méthode répondant à notre problématique.

Dans ce cadre, la méthode PerMet (PERsonalization METhodology) est exposée dans le troisième chapitre. PerMet est une méthode pour l'analyse, la conception et la modélisation des systèmes d'information personnalisés. Cette méthode permet aussi bien la mise en place d'un nouveau système d'information personnalisé que la personnalisation d'un système d'information déjà existant. Elle sépare le système d'information vu comme un ensemble de services du système de personnalisation pour prendre en compte la multi-modalité, le multi-canal et la multi-plateforme. PerMet propose un modèle de développement itératif, incrémental et permet une réalisation parallèle des phases spécifiques liées au développement des services et des phases spécifiques liées à la personnalisation. PerMet utilise des formalismes basés sur celles d'UML et de ses extensions.

Pour faciliter l'utilisation de PerMet, le système de personnalisation PerSyst (PERsonalization SYSTem) est proposé dans le quatrième chapitre. PerSyst est un système de personnalisation évolutif et distribué pouvant s'utiliser conjointement avec PerMet pour le développement de système d'information personnalisé. PerSyst est construit à partir d'une architecture multi-agents ce qui le prédispose à des caractéristiques comme l'adaptabilité, l'autonomie et l'assistance lui permettant de prendre en compte différents types de personnalisation. Ce chapitre présente aussi des modèles et des méthodes utiles pour la personnalisation de l'interaction homme-machine.

Le cinquième chapitre décrit une application concrète de la méthode PerMet utilisant le système de personnalisation PerSyst. Cette application, appelée « Mon service transport », porte sur le développement d'un système d'information personnalisé dans le domaine des transports terrestres de personnes. Trois prototypes de services différents sont décrits : un service de recherche d'itinéraire, un service d'agenda transport et un service d'information des perturbations. Des validations et des évaluations de certains principes et caractéristiques de la méthode PerMet et du système de personnalisation PerSyt ont été effectuées au travers des développements de ces trois services.

Ce mémoire se termine par une conclusion générale relative aux travaux menés dans le cadre de cette thèse, proposant également des perspectives de recherche sous la forme d'évolutions et d'extensions de ces travaux ; celles-ci concernent aussi bien la méthode PerMet que le système de personnalisation PerSyst et l'application « Mon service transport ».

Chapitre 1
Personnalisation dans les systèmes d'information

Sommaire

Introduction

La personnalisation dans les systèmes d'information fait l'objet actuellement de recherches très actives au niveau international. Ces travaux visent à fournir une interaction homme-machine adaptative et intelligente dans le but d'améliorer l'efficacité de l'interaction et l'utilisabilité des systèmes. La personnalisation est souvent abordée dans les communautés des Interfaces Homme-Machine (IHM), des Recherches d'Information (RI) et des Bases de Données (BD). Il s'avère que la personnalisation est une problématique complexe qui nécessite plusieurs disciplines dans les domaines des sciences de l'information et des sciences humaines.

Ce chapitre présente l'état de l'art de la personnalisation dans les systèmes d'information. Ce chapitre comprend trois parties. La première partie présente les notions générales, les approches et les méthodes utilisées pour la personnalisation de l'interaction homme-machine. La deuxième partie vise à étudier les systèmes de personnalisation existants. Et enfin, la troisième partie est axée sur la problématique applicative de la thèse, c'est-à-dire la personnalisation dans les systèmes d'information transport.

1.1. Personnalisation : définitions, principes et approches

Cette section définit ce qu'est la personnalisation et les différents angles selon lesquels elle peut être abordée. Les principes et approches pour la collecte d'information, pour la modélisation de l'utilisateur sont ensuite décrits. Enfin, différentes méthodes utilisées pour la personnalisation de l'interaction homme machine sont exposées.

1.1.1. Définitions

Il n'existe pas de définition standard portant sur la personnalisation. La personnalisation est donc généralement définie suivant les objectifs des auteurs. Cependant toutes ces définitions portent sur la capacité de modification de l'interaction homme-machine. A ce propos, différents termes sont utilisés et peuvent être classés en trois catégories. La première catégorie autorise l'utilisateur à changer l'apparence ou le contenu du système interactif. Les termes généralement utilisés sont *personnalisable*, *customisation* et *adaptabilité*. La deuxième catégorie porte sur la modification automatique de l'interaction homme-machine par rapport au profil de l'utilisateur. Les termes principalement employés sont *personnalisé*, *personnalisation* et *adaptativité*. Et la troisième catégorie porte sur la capacité du système interactif à s'ajuster par rapport à son contexte d'utilisation. On parle généralement de *plasticité* et de *context awareness*.

Personnalisable, customisation, adaptabilité : ces termes sont utilisés pour décrire un système interactif pouvant être adapté par l'utilisateur suivant ses préférences en termes de contenu et de contenant. Cela permet à l'utilisateur de se sentir maître du système avec lequel il interagit et augmente sa confiance au système. « *C'est une manière de développer une confiance, une familiarité et un investissement personnel* » [Kimball et Merz 00]. [Cinquin et al., 02] parlent de *customisation* (adaptation de l'anglais customization), en opposition du terme *personnalisation* que nous verrons par la suite, pour souligner l'intervention nécessaire de l'utilisateur. « *la customisation restitue des données à l'utilisateur sur la base d'informations explicites* » [Cinquin et al., 02]. Même si ces derniers temps, les termes les plus utilisés sont *personnalisable* et *customisation*[1], le terme *adaptabilité* demeure celui le plus connu en interaction homme-machine. Les définitions de l'adaptabilité restent consensuelles sur le caractère d'intervention explicite de l'utilisateur pour déclarer ses préférences avant la génération de l'interface [Oppermann et Simm 94][Stephanidis et al., 01].

Personnalisation, personnalisé, adaptativité : il s'agit de générer automatiquement une interface homme-machine suivant les préférences et les attentes de l'utilisateur. Ces termes sont utilisés pour

[1] Les termes personnalisation et customisation sont arrivés avec le développement de l'Internet où ils sont très utilisés pour qualifier qu'un site est adaptable.

insister sur le caractère automatique de la personnalisation sans que l'utilisateur fournisse explicitement ses préférences. « *Un site personnalisé tente de deviner quelles pourraient être les préférences de l'utilisateur* » [Kimball et Merz 00]. [Bazsalicza et Naïm 01] définissent la *personnalisation* comme « *la capacité d'un site dynamique à produire des ressources en fonction de l'identité du demandeur* ». Ces définitions se restreignent aux applications web et à la personnalisation du contenu. Le terme le plus général est celui d'*adaptativité*, régulièrement employé dans la communauté, qui caractérise une interface homme-machine capable de capturer les informations (préférences, besoins, …) sur l'utilisateur en cours d'exécution et s'adapte par rapport à ces informations [Browne et al., 90][Stephanidis et al., 01].

Plasticité, context awareness : la *plasticité* et le *context awareness* complètent *l'adaptativité* en considérant dans la personnalisation non seulement l'utilisateur mais le contexte d'interaction. Pour nous, il n'existe pas de différence entre la *plasticité* et le *context awareness*[2]. Cependant, on pourra noter que, pour l'instant, la majorité des travaux effectués dans le cadre de la plasticité porte sur l'adaptation du contenant et ceux effectués dans le cadre du context awareness sont consacrés à la personnalisation du contenu. La plasticité se définit comme *« la capacité des interfaces à s'adapter à leur contexte d'usage dans le respect de leur utilisabilité . Le contexte d'usage se définit comme le triplet utilisateur, plate-forme et environnement* » [Thévenin et Coutaz 99]. [Abowd et al., 99] définissent : context-awareness (context-aware systems) comme « *l'utilisation du contexte pour fournir des informations appropriées et/ou des services à l'utilisateur ; le contexte étant n'importe quelle information qui peut être utilisée pour caractériser la situation d'une entité qui peut être un utilisateur, un environnement, ou un objet physique ou informatique* ».

Dans ce mémoire, nous entendons par personnalisation, la capacité d'adaptation d'un système interactif par rapport au contexte. Cette adaptation peut aussi bien porter sur le contenu (données, informations, document, …) que sur le contenant (présentation, plate-forme d'interaction, mode de communication, canal de communication, …). Nos travaux se situent donc dans la lignée des travaux portant sur la plasticité et le context awareness.

1.1.2. La collecte d'information pour la personnalisation

Pour pouvoir effectuer de la personnalisation, il est nécessaire de connaître les préférences et les besoins de l'utilisateur. Deux moyens peuvent être utilisés pour collecter les informations à propos de l'utilisateur : la collecte explicite d'information et la collecte implicite d'information.

1.1.2.1. La collecte explicite

La collecte explicite d'information, appelée aussi déclaration, correspond à toute donnée qui a été saisie ou fournie par l'utilisateur final. C'est la méthode la plus directe pour acquérir de l'information sur l'utilisateur. Au niveau le plus basique, l'utilisateur est invité à remplir un questionnaire, généralement présenté sous forme de formulaire, pour fournir des informations à propos de ses préférences ou à propos de son niveau d'expertise. Ceci pourrait être une source particulièrement importante pour les interfaces adaptatives utilisées par une population très diverse. Considérons par exemple, des utilisateurs à vision réduite. Ils ont assez souvent une bonne connaissance de la nature exacte de leurs difficultés, incluant les aspects subtils, comme leur capacité à lire des couleurs particulières. Si les utilisateurs donnent directement quelques informations, elles peuvent être utilisées pour personnaliser les éléments de l'interface. Vraisemblablement, la demande explicite peut être le meilleur moyen pour recueillir des informations de haute qualité, qui peuvent refléter les aspects importants, les besoins subtils et les préférences des utilisateurs. Cependant, la valeur de ce type d'information n'est pas toujours fiable car rien ne garantit la véracité des réponses fournies par l'utilisateur aux formulaires d'inscription. La plupart des utilisateurs du web veulent préserver leur anonymat. [Kimball et Merz 00] souligne que 50% des hommes et 80% des femmes ne fournissent pas leur vrai nom. Et quand on demande aux utilisateurs leur identité ou leur adresse

[2] Les auteurs francophones utilisent le plus souvent le terme plasticité et les auteurs anglo-saxons emploient le terme context-awareness.

électronique sans but précis, au moins la moitié des utilisateurs du web fournissent de fausses informations.

Les approches actuelles (qui relèvent surtout de la définition des IHM évoluées ou de l'ergonomie) consistent à décomposer les longs questionnaires en des petits formulaires qui apparaissent lorsque le système a besoin d'une information particulière sur l'utilisateur pour des besoins de personnalisation.

1.1.2.2. La collecte implicite

La collecte implicite d'information sur l'utilisateur, appelé aussi *tracking* ou *profiling*, permet généralement de recueillir une grande quantité d'information sur l'utilisateur sans lui faire aucune demande explicite. Cette méthode permet de ne pas obliger l'utilisateur à spécifier ses préférences ou ses centres d'intérêts. Etant donné que l'utilisateur laisse à la charge du système le soin d'apprendre des informations sur lui, il se pourrait que certaines de ses activités soient oubliées ou mal interprétées. C'est le principal inconvénient de cette méthode. Par ailleurs, certaines données sont plus facilement évaluées par le système que par l'utilisateur ; ce dernier n'ayant pas forcément les compétences requises ou le temps nécessaire pour cette évaluation. Par exemple, une grande partie des utilisateurs ne peuvent pas évaluer la vitesse de frappe sur le clavier. Il est meilleur de l'évaluer par l'observation.

Cette méthode permet aussi de prendre en compte le changement des préférences utilisateur.

Différentes techniques sont utilisées pour la collecte implicite d'information dans le web :

- **Analyse des fichiers de logs des serveurs web** : les fichiers de logs (ou journaux) servent à conserver une trace de l'activité du serveur web. Ils étaient à l'origine utilisés pour des besoins de maintenance. Chaque requête http est enregistrée sous un format standard CLF (Common Log Format) ou ECLF (Extended CLF). Les données les plus utilisées sont l'adresse de la machine d'où provient la requête (adresse IP du client), la date de la requête, la page demandée (la requête elle-même), le type de réponse du serveur (réussi ou non) et le type de client utilisé. Les travaux issus du *web usage mining* s'intéressent à l'analyse de ce genre de fichier pour identifier le comportement de l'utilisateur (voir par exemple [Murgue 05]).

  Cette technique de profiling présente les avantages suivants :

  - les données sont toujours disponibles puisque tous les serveurs web fournissent ces fichiers de logs.
  - les données enregistrées sont assez fiables puisqu'elles sont issues d'un comportement réel des utilisateurs.
  - l'information récoltée est assez précise. Les pages consultées, leur ordre séquentiel, leur espacement dans le temps sont autant d'informations analysables grâce à ces journaux.

  Les inconvénients de cette technique sont les suivants :

  - cette technique est difficilement adaptable aux sites web dynamiques. En effet, chaque page étant construite dynamiquement, cela rend difficile l'analyse et l'exploitation des fichiers de logs.
  - certaines actions s'exécutant au niveau du poste client ne se retrouvent pas dans ces fichiers de logs. Par exemple, les données des ActiveX, des applets Java ou des transactions marchandes ne figurent pas sur ces fichiers. L'historisation de ce genre d'information doit être réalisée par une autre méthode.
  - l'analyse des fichiers de logs nécessitent le plus souvent des traitements lourds. L'information n'est donc généralement pas disponible immédiatement.

- **Utilisation des moniteurs de réseau** : les moniteurs de réseau capturent les flux sortants au niveau http (couche application), ainsi qu'au niveau des paquets IP (couche réseau). Cette technique permet de capturer certaines actions spécifiques qui ne peuvent pas toujours être

déduites des fichiers de logs de serveur web ; par exemple, la détection des requêtes interrompues par le client (appui du bouton STOP du navigateur). Par contre, puisqu'ils doivent être placés entre le client et le serveur, ces moniteurs de réseau ne peuvent pas capturer les flux http encryptés. Ils sont donc généralement utilisés en complément des fichiers de logs serveur.

- **Utilisation des auditeurs de page web** : cette technique consiste à centraliser la collecte d'information par une application externe. Il s'agit de notifier à un fournisseur de service (comme d-stat[3] ou webhit[4], par exemple) la visualisation d'une page par un client. Le principe consiste à renvoyer à l'utilisateur une page enrichie d'une portion de code (écrite en JavaScript, en général). Cette portion de code qui s'exécute sur le poste client renvoie un certain nombre d'informations sur un site central qui se charge d'enregistrer et d'analyser les informations. La nature des informations collectées est en principe de bonne qualité. En effet, il est très facile de renvoyer les informations telles que le titre de la page, le contenu des méta-balises dans la page HTML, l'URL appelante et même les champs saisis d'un formulaire (nom, adresse, email…). Cette technique présente aussi l'avantage d'alléger le serveur web en production.
- **Utilisation des logs applicatifs** : les fichiers de logs applicatifs servent à conserver une trace des événements intrinsèques liés au fonctionnement même de l'application. Il est difficile de collecter certaines informations (comme le contenu d'un panier d'achat sur un site de commerce électronique, par exemple) suivant une des trois techniques présentées précédemment. Les logs applicatifs pallient ces limitations. Ils se généralisent pour la collection d'informations relativement précises.
- **Utilisation des paramètres de configuration** : les paramètres de configuration utilisés pour la *customisation* de l'interface, le contenu informationnel ou la navigation (comme *My Yahoo*!, [Manber et al., 00] ou *Google*[5], par exemple) peuvent servir pour déduire les préférences de l'utilisateur. Cette technique est généralement utilisée pour alléger la collecte de données par formulaire. Ainsi des informations comme la langue, le lieu de l'utilisateur, les préférences en terme de contenu, etc., seront déduites à partir des paramètres choisis par l'utilisateur et ne figureront pas dans les questionnaires. Son principal inconvénient réside dans la qualité de la déduction effectuée. Il n'est pas sûr qu'un utilisateur habite à Lille parce qu'il a choisi la météo de Lille pour sa page personnalisée ; mais peut-être s'y rend-t-il régulièrement ?

1.1.3. Modélisation de l'utilisateur avec une visée de personnalisation

L'ensemble des informations récupérées par l'une ou l'autre des solutions présentées précédemment sont ensuite mémorisées par le système pour constituer un modèle de l'utilisateur. Le terme modèle utilisateur est utilisé dans différents domaines[6]. Il peut aussi vouloir dire modèle d'étude (*student model*) ou modèle d'apprenant (*learner model*). Parfois, il inclut les modèles cognitifs, mentaux, systèmes, tâches, profil utilisateur et autres. L'utilisation de ces termes a été introduit par D. A. Norman en 1983 [Norman 83]. [Allen 97] définit le modèle utilisateur comme *"l'image que le système informatique a de l'utilisateur"*.

[Kay 01] distingue trois axes principaux selon lesquels un modèle utilisateur peut contribuer à la personnalisation. Ils sont illustrés dans la Figure 1.1, où les double-lignes verticales délimitent l'interaction entre l'utilisateur et un système :

- La première flèche horizontale indique une action de l'utilisateur sur l'interface. Cette action peut être réalisée par l'intermédiaire des périphériques disponibles : une action de la souris, une frappe sur le clavier, le discours de l'utilisateur via un système d'entrée audio, etc. Le modèle utilisateur peut apporter une assistance dans l'interprétation des informations. Par

[3] http://www.d-stat.com
[4] http://www.webhits.fr
[5] http://www.google.fr
[6] Cf. la revue « User Modeling and User-Adapted Interaction » accessible en ligne sur http://www.umuai.org

exemple, si l'entrée de l'utilisateur est ambiguë, le modèle utilisateur peut rendre le système capable de clarifier cette entrée.

- La deuxième flèche horizontale indique les actions du système sur l'interface. Ces actions peuvent être contrôlées par le modèle utilisateur pour améliorer la qualité de l'interaction. Par exemple, certains systèmes adaptent leurs présentations en fonction de l'utilisateur. Pour un utilisateur malvoyant, l'information sera représentée dans une police de grande taille, etc.
- Enfin, le modèle utilisateur peut diriger les actions internes du système. C'est le principal but des systèmes qui filtrent l'information pour l'utilisateur.

Chacune de ces fonctions fait l'objet d'une modélisation plus ou moins approfondie. Par exemple, le filtrage d'information peut se baser simplement sur les mots composant le titre d'un document ou analyser de façon plus détaillée le contenu de ce document.

Un même modèle peut bien évidemment jouer ces trois fonctions.

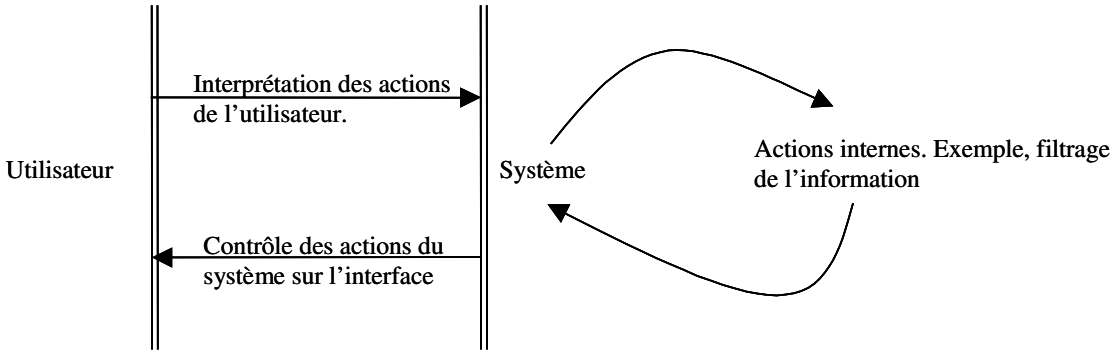


Figure 1.1. Rôle du modèle utilisateur pour la personnalisation [Kay 01]

L'orientation actuelle pour la modélisation de l'utilisateur dans une perspective de personnalisation vise à classifier les informations utilisateur selon des catégories. On pourra citer, par exemple, la proposition de [Bouzeghoub et Kostadinov 04] qui revoit et étend le standard P3P (Platform for Privacy Preferences)[7] pour prendre en compte d'autres caractéristiques comme l'ontologie du domaine ou la qualité attendue des résultats délivrés tout en explicitant les liens sémantiques existants entre les différentes catégories. Cette approche propose une catégorisation du modèle utilisateur selon huit dimensions (données personnelles, centre d'intérêt, ontologie du domaine, qualité attendue des résultats délivrés, customisation, sécurité, retour de préférences et informations diverses).

1.1.4. Méthodes de personnalisation

Les méthodes de personnalisation sont utilisées pour l'exploitation des modèles utilisateur. Elles permettent de fournir à un utilisateur un contenu et/ou un contenant personnalisé. Nous distinguerons deux types de méthodes pour la personnalisation : les méthodes cognitives et les méthodes sociales. Ces deux types de méthodes peuvent être combinés (Cf. chapitre 5).

1.1.4.1. Méthodes cognitives

La notion de méthode cognitive proposée par [Salton 83] en 1983 tire son origine de la recherche d'information traditionnelle. Cette méthode se base sur le profil de l'utilisateur ou sur l'analyse des données contextuelles manipulées par l'utilisateur. Généralement, dans le cadre de la recherche d'information personnalisée, le profil de l'utilisateur et les documents sont représentés par

[7] http://www.w3.org/P3P/

des vecteurs de mots-clés et des poids associés à ces mots-clés. Le principe consiste à calculer la distance entre le vecteur représentatif de l'utilisateur et ceux des documents. Pour une requête utilisateur, le système recommande le document ayant la distance la plus faible [Aas 97][Korfhage 97].

Suivant l'objectif et le type de personnalisation souhaités, d'autres méthodes cognitives sont utilisées. Nous citons l'approche de [Lieberman et al., 01] qui utilise le *TF-IDF* (Terme Frequency - Inverse Document Frequency) [Salton et Buckley 88]. Son système effectue une analyse de la fréquence d'apparition d'un mot dans le document que l'utilisateur est en train de consulter et effectue une recommandation contextuelle. [Ambrosini et al., 97] utilise les *réseaux sémantiques pondérés* pour structurer des documents afin d'effectuer un filtrage lors d'une recherche d'information. [Soltysiak et Crabtree 98] utilise les *réseaux bayésiens* et la *logique floue* pour l'analyse des pages web visitées par l'utilisateur afin de prévoir ses centres d'intérêts. [Linden et al., 97] utilise les *réseaux de contraintes* pour décrire les préférences de l'utilisateur par rapport à un ensemble de solutions possibles. Le système proposé dans [Moghrabi et Eid 98] représente les pages web visitées par l'utilisateur sous forme de *réseaux de neurones*. Le système de [Coyle et Cunninghan 02] se base sur un *raisonnement à partir de cas* pour proposer les informations classées par ordre de pertinence suivant les préférences de l'usager des transports aériens.

La plupart des méthodes utilisées pour la personnalisation du contenu correspondent à des méthodes statistiques ; contrairement à celles utilisées pour la personnalisation du contenant qui utilisent, généralement, des méthodes d'apprentissage symbolique [Pohl 99]. On pourra citer, par exemple, le système de personnalisation BGP-MS (Belief, Goal and Plan Maintenance System) proposé par [Kobsa et Pohl 95] qui utilise *la logique modale* avec un moteur d'inférence d'ordre 1. Les systèmes effectuant de la personnalisation suivant les plates-formes d'interaction utilisent généralement des règles de production (voir par exemple, [Florins et Vanderdonckt 04]).

On notera que plusieurs de ces méthodes peuvent être combinées (voir par exemple, [Hirsh et al., 00] ou [Schiaffino et Amandi 00]).

1.1.4.2. Méthodes sociales

Le principe de méthode sociale a été proposé en 1992 par [Goldberg et al., 92]. Les systèmes basés sur une méthode sociale, appelés aussi des systèmes de filtrage collaboratif ne se basent pas sur l'analyse du profil unique de l'utilisateur courant[8] pour la personnalisation mais se base sur l'expérience des autres utilisateurs. C'est la méthode utilisée par le célèbre site amazon[9] pour la recommandation de ses produits aux internautes. Le filtrage collaboratif peut être abordé selon deux points de vue : celui des utilisateurs ou celui des ressources. Selon le premier point de vue le système regroupe les utilisateurs similaires en analysant leurs préférences par rapport aux ressources (données, informations, etc.) [Shardanand et Maes 95]. Selon le deuxième point de vue, il s'agit de regrouper les ressources en se basant sur les annotations des utilisateurs. On parle alors de « recommandation basée sur le contenu » [Balabanovic et Sholham 97]. Des approches dites de « recommandation basée sur le contenu et la collaboration » combinent ces deux méthodes [Basu et al., 98][Melville et al., 02][Cosley et al., 02][ChoiceStream 04].

Deux types de filtrage collaboratif se distinguent : (i) le filtrage collaboratif basé sur la « mémoire » qui présente l'avantage d'être relativement simple à mettre en œuvre et prend en compte l'évolutivité du profil de l'utilisateur. Son principal inconvénient réside dans la complexité combinatoire importante générée et limite son utilisation à une grande échelle. (ii) le filtrage collaboratif basé sur un « modèle » qui réduit nettement la complexité combinatoire mais nécessite une phase d'apprentissage et prend peu en compte l'évolutivité du profil.

Le filtrage collaboratif basé sur la mémoire fournit une recommandation en analysant les comportements des autres utilisateurs. La solution la plus simple consiste à effectuer un *vote*

[8] Nous appellerons utilisateur courant, l'utilisateur qui est en train d'utiliser le système d'information.
[9] http://www.amazon.fr

majoritaire (recommander à l'utilisateur courant la solution la plus choisie par les autres utilisateurs). Pour un filtrage plus fin, la recommandation est effectuée en calculant la corrélation existante entre l'utilisateur courant et les utilisateurs présentant des goûts et des intérêts communs. Cette corrélation peut être obtenue suivant différentes techniques (*similarité des vecteurs,* l'approche de *Pearson*, *les moindres carrés*, l'approche de *Bayes*, etc.). Pour un aperçu général sur ces méthodes, voir [Breese et al., 98][Berrut et Denos 03].

Le filtrage collaboratif basé sur un modèle définit un ou plusieurs modèles qui seront instanciés pour un utilisateur donné. Pour obtenir un modèle, plusieurs méthodes sont proposées dans la littérature. Nous citerons les méthodes centrées probabilité comme l'*approche probabiliste* qui calcule la prédiction comme une espérance mathématique sur le profil de l'utilisateur, l'approche par *Classifieur « naïf » de Bayes* qui détermine la classe C la plus probable à laquelle l'utilisateur courant va appartenir et l'approche par *arbre de décision* qui consiste en la recherche de dépendances entre les ressources. Citons aussi les approches de *clustering* qui ont pour but de limiter le nombre d'utilisateurs à considérer dans le calcul de la prédiction. Le principe de la recommandation par *clustering* consiste à regrouper les utilisateurs ayant les mêmes goûts en clusters. La prédiction est calculée en fonction des utilisateurs appartenant au même cluster. Les méthodes des *k plus proches voisins* [Herlocker et al., 99], le *clustering répété* [Ungar et Foster 98] et le *clustering hiérarchique* [Fisher 96] sont les plus utilisées.

Depuis quelques années, un nouveau courant de filtrage collaboratif est apparu : le filtrage collaboratif « hybride ». Le principe consiste à combiner un filtrage basé sur la mémoire et un filtrage basé sur un modèle afin de limiter les inconvénients des deux méthodes et de bénéficier de leurs avantages. Parmi les approches que nous pouvons trouver dans la littérature, nous pouvons citer *l'approche de Horting* [Aggarwal et al., 99], l'approche basée sur le *diagnostic de personnalité* [Pennock et al., 00] et l'approche basée sur *la clusterisation et la réduction de dimension* [Goldberg et al., 01].

1.1.4.3. Synthèse

Diverses méthodes sont utilisées pour la personnalisation. Le Tableau 1.1 résume les méthodes de personnalisation adaptées, d'une part pour répondre à la demande d'un utilisateur, d'autre part dans le cadre d'un suivi des centres d'intérêts de l'utilisateur. Dans les deux cas, le système peut baser ses réponses sur les utilisateurs, ou sur le contenu des données présentées.

		Types de filtrage	
		Méthodes sociales	*Méthodes cognitives*
Information de contexte	*Utilisateur*	Filtrage collaboratif	Filtrage basé sur le profil
	Données	Recommandation de communauté	Recommandation contextuelle

Tableau 1.1. Classification des méthodes de personnalisation selon le contexte utilisé et le type de filtrage adaptée de [Cinquin et al., 02]

Chaque famille de méthodes est à la fois diversifiée et en constante évolution. Dans ce contexte, les caractères à la fois modulaire et évolutif des systèmes s'avèrent cruciaux. Concernant la personnalisation, la création de services indépendants des systèmes d'information et aisément modifiables à tout moment afin de profiter des évolutions au fur et à mesure de leur mise à disposition, est un avantage certain.

Cependant, comme nous le verrons par la suite (cf. §1.2.3), très peu de systèmes permettent cette modularité. Les systèmes sont généralement conçus avec une ou plusieurs méthodes de personnalisation dédiées et figées pour répondre à un type de personnalisation bien défini.

1.2. Etudes des systèmes de personnalisation

Nous entendons par « Système de personnalisation », un système logiciel qui permet une interaction homme-machine personnalisée (dans ce cas le système de personnalisation assure les fonctions de collecte d'information utilisateur, de modélisation de l'utilisateur et des méthodes

d'apprentissage automatique pour la personnalisation) ou qui interagit avec une application tierce pour la personnalisation de l'interaction (la collecte d'information utilisateur est généralement laissée aux applications externes). Cette section se décompose en quatre parties. La première partie introduit la notion d'agent et de système multi-agents dans un cadre de personnalisation puisque de nombreux systèmes de personnalisation sont construits à base d'agents logiciels. La deuxième partie définit les critères utilisés pour l'étude comparative des systèmes de personnalisation. La troisième partie présente une étude de quelques systèmes de personnalisation. Et la dernière partie concerne une synthèse et une discussion sur les systèmes de personnalisation existants.

1.2.1. Notions d'agent et de système multi-agents dans un cadre de personnalisation

Depuis le début des années 80, de nombreux travaux visant de nouvelles interactions de plus en plus adaptatives, intelligentes et personnalisées sont effectués [Schneider-Hufschmidt et al., 93][Kolski et Le Strugeon 98][Höök 00]. Pour réaliser de tels systèmes adaptatifs, personnalisés, la plupart des chercheurs se sont orientés vers une approche orientée agent [Chin 91][Laurel 97][Grislin-Le Strugeon et al., 01].

Il n'existe pas une définition unique du concept agent. Celle qui couvre la plupart des thèmes pourrait être la définition de [Ferber 95]. On appelle agent « *une entité physique ou virtuelle (a) qui est capable d'agir dans un environnement, (b) qui peut communiquer directement avec d'autres agents,(c) qui est mue par un ensemble de tendances (sous la forme d'objectifs individuels ou d'une fonction de satisfaction, voire de survie, qu'elle cherche à optimiser), (d) qui possède des ressources propres, (e) qui est capable de percevoir (mais de manière limitée) son environnement, (f) qui possède des compétences et offre des services, (g) qui peut éventuellement se reproduire, (h) dont le comportement tend à satisfaire ses objectifs, en tenant compte des ressources et des compétences dont elle dispose, et en fonction de sa perception, de ses représentations et des communications qu'elle reçoit* ».

Celle de [Maes 94] qui définit un agent logiciel comme « *un système informatique qui existe dans un environnement complexe et dynamique, perçoit et agit de façon autonome dans cet environnement, et de ce fait réalise un jeu d'objectifs et de tâches pour lequel il est conçu* » ou celle de [Jennings et al., 98] qui considère un agent comme « *un système logiciel capable d'agir de manière autonome et flexible dans un environnement* » correspondent aux définitions des agents utilisés pour les systèmes de personnalisation à base d'agents décrits par la suite.

Même s'il n'existe pas de définition unique pour ce qu'est un agent logiciel, les définitions sur les systèmes multi-agents se rejoignent plus ou moins. On pourra citer celle de [Ferber 95] : *«un système multi-agents est composé d'un ensemble d'agents capables de communiquer et de négocier afin de résoudre une tâche commune»,* ou celle de [Bond et Gasser 88] reprise dans [Mandiau et Grislin-Le Strugeon 01] *« un système multi-agents est un ensemble d'agents qui coordonnent leurs connaissances, buts, expériences et plans pour agir ou résoudre des problèmes, incluant le problème de la coordination inter-agent lui-même».*

Les agents logiciels intervenant pour la personnalisation de l'interaction homme-machine possèdent un ou plusieurs des caractéristiques suivantes :

Intentionnalité : un agent intentionnel est guidé par ses buts. L'agent a donc la volonté d'atteindre un but ou d'effectuer une action et cette volonté (intention) est représentée explicitement (voir à ce propos [Cohen et Levesque 90][Erceau et Ferber 91]). L'agent intentionnel construit donc des plans pour atteindre des buts explicites. Par exemple : Un agent possède le but B suivant : Connaître T, la tranche d'âge de l'utilisateur U.

La réalisation d'un agent intentionnel supposera donc de représenter ses buts, classés généralement en buts, sous-buts... Par exemple : B1 = Connaître l'âge A de l'utilisateur U.

Rationalité : un agent rationnel suit le principe suivant : « Si un agent sait qu'une de ses actions lui permet d'atteindre un de ses buts, il la sélectionne. » [Newell 82] .

L'agent a la volonté d'atteindre un but ; son comportement est cohérent avec cette intention : il y a une raison à chacune de ses actions. Il peut donc justifier ses résultats en fonction de ses méthodes [Labidi et Lejouad 93]. Par exemple : L'agent X accomplit l'action A parce qu'il veut obtenir l'effet E.

La mise en oeuvre de ce principe au sein d'un agent est généralement réalisée par la planification : l'agent possède un but prioritaire qu'il décompose en sous-buts jusqu'à un ensemble de buts "primitifs" pour lesquels il dispose des moyens d'actions nécessaires à leurs réalisations.

Par exemple : l'agent veut classer U dans une tranche d'âge T.

Il sait que "classer U" peut être remplacé par « Connaître l'âge A de U » puis « Trouver Ti tel que A Ti » et « Connaître l'âge A de U » = « Connaître la date de naissance de U » puis « Calculer la différence avec la date actuelle ». D'où l'action : « Demander la date de naissance de U ».

Adaptabilité : un agent adaptatif est capable de contrôler son comportement, ses communications selon l'agent avec lequel il interagit. Cet agent contrôle ses activités de perception, de communication, d'action et de raisonnement et peut modifier ses connaissances [Ferber 95].

L'adaptation peut intervenir à différents degrés (faible ou forte adaptation) et ne concerne généralement que certains aspects du comportement.

Autonomie : l'autonomie permet aux agents intelligents de travailler seuls, sans l'aide de l'utilisateur. Il existe différents degrés dans l'autonomie [Castelfranchi 95]. Un agent autonome peut opérer, non seulement en parallèle avec l'utilisateur, mais surtout de sa propre initiative sans demande explicite de l'utilisateur (dans ce cas, on parle de pro-activité de l'agent).

Par exemple : l'agent estime que l'utilisateur a besoin d'une certaine information et effectue les actions nécessaires (recherche de l'information sur le web, par exemple) à l'obtention de cette information sans que l'utilisateur n'intervienne.

Communicabilité : la communication donne aux agents de nombreuses possibilités. La communication permet de synchroniser les actions des agents, de résoudre les conflits de ressources et de buts par la négociation, de coopérer [Mathieu et Verrons 03]. Les agents échangent des connaissances et des savoir-faire. L'envoi d'information peut se faire d'un agent vers un autre agent précis ou par diffusion à des groupes d'agents, en réponse à une requête ou non.

Les protocoles de communication les plus utilisés actuellement sont KQML (Knowledge Query and Manipulation Language) [Finin et al., 97] développé en 1993 à l'université de Maryland (USA) et FIPA-ACL (Agent Communication Language)[FIPA 02] développé en 1999 par la FIPA (Fondation for Intelligent Physical Agents).

Mobilité : la mobilité est une autre caractéristique possible pour un agent. Les agents mobiles font références à des programmes capables de migrer sur un réseau informatique et d'agir au nom de l'utilisateur ou d'une autre entité [Fuggetta et al., 98][Gray et al., 00]. Les tâches pour lesquelles ces agents sont utiles sont essentiellement des tâches de veille sur le réseau (l'agent déclenche des actions en fonction des événements et prévient l'utilisateur) et de réalisation d'interactions avec des serveurs (par exemple, de la collecte de données).

Reproductibilité : la reproduction permet à un agent de créer d'autres agents ayant les mêmes compétences. Par exemple, le système proposé par [Petit-Rozé et Grislin-Le Strugeon 02] utilise ce principe de la reproduction des agents pour instancier automatiquement un agent assistant destiné à s'occuper de l'utilisateur durant sa connexion au système.

Une application importante des agents pour la personnalisation de l'interaction homme-machine concerne les agents d'**assistance.** [Beale et Wood 94] présentent les principales fonctionnalités offertes par les agents en terme d'assistance : filtrage de données, adaptation de l'interface, support à l'exécution de tâches, etc. Ces agents portent parfois des dénominations différentes (agents assistants, agents utilisateur, agents guides, par exemple) mais leur fonction principale s'avère identique : assister l'utilisateur dans la réalisation de sa tâche en offrant des services de manière autonome, c'est-à-dire sans que l'utilisateur doive en formuler la demande.

[Banks et al., 97] pensent que les agents intelligents permettront de réduire la complexité des interfaces et de rendre plus efficace le travail de l'utilisateur.

Les agents assistants sont, d'après [Rich 96], des agents qui interagissent avec des programmes d'applications partagées à travers une même interface de manière à observer l'utilisateur. Ce sont en quelques sortes des « robots » dont les senseurs et les effecteurs sont les capacités d'entrée et de sortie de l'interface. L'agent peut observer les actions de l'utilisateur sur l'interface et faire des modifications sur les objets de l'interface affichés à l'écran. Il surveille les actions de l'utilisateur sur de longues périodes, repère des schémas et propose de les automatiser. Ils sont donc d'après [Lashkari 97] des systèmes semi-intelligents qui assistent l'utilisateur sur des tâches quotidiennes. Avant de pouvoir les utiliser, ces agents ont besoin de suffisamment de temps et sont limités aux situations qu'ils connaissent.

Ces agents sont parfois représentés anthropomorphiquement par un visage sur l'écran. Des études ont été menées sur les effets d'une représentation de l'agent par un humain. La classification de [Lieberman 97] détermine une classe à part entière pour les agents anthropomorphiques. Ce sont des agents qui imitent la forme humaine essentiellement par un visage et des expressions.

En règle générale, on souligne qu'un agent d'interface ne serait pas d'une grande aide s'il fallait toujours que l'utilisateur lui donne des instructions très explicites. Il serait préférable d'avoir un assistant qui pourrait agir de manière indépendante et surtout simultanément par rapport à l'utilisateur.

1.2.2. Critères pour l'étude comparative des systèmes de personnalisation

Dans cette partie sont présentés les critères utilisés pour l'étude comparative des systèmes de personnalisation. Nous distinguons sept critères : *la multi-application*, la *distribution*, *la recherche d'information*, *le type de personnalisation*, *la collecte d'information*, *la méthode de personnalisation, l'évolutivité*. Ces critères ont été définis par rapport à notre objectif de personnalisation aussi bien du contenant que du contenu tout en favorisant la réutilisation de l'existant, le support de la distribution par le système et son évolutivité.

- **La multi-application :** le système peut être accessible par plusieurs applications. Cela permet de centraliser la personnalisation afin d'utiliser les mêmes préférences utilisateur pour différentes applications. L'utilisateur s'authentifie une seule fois et son profil est utilisé par plusieurs applications. C'est le principe de « *single sign-on* » [Pfitzmann 04]. Pour nous, cela permet aussi de pouvoir prendre en compte les différentes plates-formes d'interaction (multi-plateforme), les différents modes d'interaction (interaction multimodale) et les différents canaux (multi-canal) possibles pour l'accès ou la diffusion de l'information. Le système doit aussi bien pouvoir interagir avec un serveur web qu'avec un serveur vocal, par exemple.
- **La distribution :** c'est la capacité du système à distribuer ses différents modules. Les différents composants du système de personnalisation peuvent ne pas tous se situer sur une même machine. Par exemple, le composant de gestion de profil peut se situer sur un serveur dédié à la gestion des profils utilisateur alors que le composant de recherche d'information peut se situer sur un serveur d'information. Ce critère est important afin que le système de personnalisation puisse supporter des méthodes d'apprentissage automatique nécessitant parfois beaucoup de ressources [Lumineau 03]. Le système proposé par [Hagimont et Layaïda 02] utilise un agent mobile pour l'adaptation des données d'un serveur multimédia en fonction des besoins spécifiques de l'utilisateur ou des contraintes liées au médium de communication. Cela a permis de réduire la charge de calcul sur la machine de l'utilisateur, de réduire la charge sur le réseau et de personnaliser la prise en compte de panne.
- **La recherche d'information :** le système recherche lui-même les données dont l'utilisateur a besoin. La recherche peut s'effectuer sur une base de données ou sur le web. Concernant la recherche sur le web, des méthodes issues de la découverte automatique de données (*information retrieval*) ou de la fouille de texte (*text mining*) sont généralement utilisées. La plupart des systèmes de personnalisation capables d'effectuer de la recherche d'information personnalisée sur

le web ou sur des bases de données hétérogènes sont à base d'agents logiciels. Ceci s'explique par le fait que la recherche d'information personnalisée nécessite une certaine autonomie du système ; le développement de ce genre de systèmes peut ainsi être facilité par une approche orientée agent.

- **La collecte d'information :** la collecte d'information correspond à la méthode utilisée par le système pour recueillir de l'information à propos de l'utilisateur. Différentes techniques sont utilisées (voir §1.1.2, pour plus de détail sur ces techniques).
- **Le type de filtrage :** il s'agit de la méthode de filtrage utilisée pour la gestion du profil utilisateur ; voir §1.1.4 pour plus de détails sur les méthodes utilisées pour la personnalisation. Le Tableau 1.1 présente les quatre types de filtrage possibles.
- **Le type de personnalisation :** des systèmes effectuent de la personnalisation par rapport au contenant et d'autres s'intéressent surtout à la personnalisation du contenu. Différents types de personnalisation sont possibles par rapport à ces deux axes. Pour la personnalisation du contenu, on citera à titre d'exemple, le filtrage d'information [Waern et al., 98], la recommandation de données [Lieberman 95], le tri des informations par rapport aux préférences de l'utilisateur [Asnicar et Tasso 97] et l'aide à la formulation de requête [Deschaine et al., 00]. Et pour la personnalisation du contenant, on pourra citer par exemple, la plasticité [Thévenin et Coutaz 99], l'assistance de l'utilisateur [Virvou et Kabassi 02] et la composition de l'interface[10] par l'utilisateur.
- **L'évolutivité :** le dernier critère concerne la capacité du système de personnalisation à prendre en compte d'autres besoins de personnalisation que ceux définis au départ. Comme nous l'avons vu précédemment, différentes techniques sont disponibles dans un processus de personnalisation de l'interaction homme-machine (collecte d'information, méthode de personnalisation et type de personnalisation). Par exemple, un type de personnalisation peut nécessiter une méthode de personnalisation particulière (parce que cette méthode est la plus adéquate pour répondre aux objectifs fixés) et peut aussi nécessiter une technique de collecte d'information bien définie. Le système doit donc permettre l'utilisation des techniques adéquates pour prendre en compte ce nouveau besoin. Ce critère est important par le fait que la mise en place d'un système de personnalisation nécessite un investissement assez lourd (en terme de coût de développement). Un système de personnalisation évolutive permettrait de réduire ces coûts car il pourrait bénéficier des modules déjà existants pour la mise en place d'un nouveau service personnalisé. En effet, une technique de collecte d'information, par exemple, peut servir pour différents types de personnalisation.

1.2.3. Etude des systèmes de personnalisation

Plusieurs systèmes contribuent à des avancées dans le domaine de la personnalisation en interaction homme-machine. La plupart de ces systèmes sont destinés à l'aide à la navigation sur le web. Ils assurent les fonctions d'observation du comportement utilisateur, de recherche, de filtrage et de présentation de ces informations.

1.2.3.1. Systèmes de personnalisation dédiés

Une grande majorité des systèmes de personnalisation existants sont fournis sous forme d'application dédiée à une fonction particulière pour un objectif fixe.

Letizia [Lieberman 95] : est un agent intelligent pour le parcours du web. Il enregistre les URLs choisies par l'utilisateur, lit les pages et dessine un profil de l'utilisateur au fur et à mesure que celui-ci visite des pages. A partir de là, il recherche d'autres pages susceptibles d'intéresser l'utilisateur et présente ses résultats sur une fenêtre indépendante. A chaque fois que l'utilisateur change de pages, Letizia recentre ses recherches en fonction de la nouvelle page.

[10] l'application proposée par http://www.netvibes.com/, par exemple, permet à l'utilisateur de choisir les éléments de son portail web et de le personnaliser par rapport à ses préférenes. Ce portail permet une personnalisation *manuelle* du contenant et du contenu.

IFWeb [Asnicar et Tasso 97] : est un agent qui effectue la recherche et le filtrage des documents en prenant en compte les besoins spécifiques de l'utilisateur. Lorsqu'un document est pointé par l'utilisateur, le système recherche dans le web les documents similaires et les montre à l'utilisateur en les classant par ordre d'intérêt.

ConCall [Waern et al., 98] : est un service d'information adaptatif à base d'agents logiciels. Le système sert à rassembler, filtrer et diffuser des appels pour des conférences, des workshops, etc., sur internet. Les utilisateurs sont classés selon leur profil (chercheur, éditeur, …). Le modèle utilisateur est basé sur un ensemble de mots clés. Le profil utilisateur est utilisé à la fois comme filtre lors de la recherche d'informations et comme moyen d'ordonner les résultats de la recherche (évaluation de la pertinence du résultat).

RESCUER (Reasoning System about Commands Using Evidence Reasonably) [Virvou et Du Boulay 99] : est un système d'aide aux utilisateurs du système d'exploitation Unix. Il génère des hypothèses concernant les croyances de l'utilisateur à partir de l'observation de ses actions. Le modèle de l'utilisateur consiste en un ensemble de valeurs associées aux objets, états du système Unix : cet ensemble représente les croyances de l'utilisateur par rapport au système. Le modèle est basé sur l'historique des commandes et donc sur l'état précédent des croyances de l'utilisateur.

IFM (Intelligent File Manipulator) [Virvou et Kabassi 02] : est un système d'aide pour la manipulation graphique des systèmes de fichiers. Il intervient automatiquement et propose des conseils lorsqu'un utilisateur fait des démarches inconsistantes ou commet des erreurs de manipulation. IFM se base sur les stéréotypes et incorpore un mécanisme de reconnaissance des buts de l'utilisateur.

InfoSleuth [Deschaine et al., 00] : est un système multi-agents qui met en relation les requêtes des utilisateurs et les fournisseurs de services correspondants. Le travail des agents d'InfoSleuth repose sur l'utilisation d'ontologies, qui leur permettent de préciser les requêtes, de les décomposer, puis de fusionner les informations collectées à partir de sources hétérogènes et distribuées.

Gulliver's Genie [O'Hare et O'Grady 03] : est un guide touristique à base d'agents logiciels qui fournit aux utilisateurs des informations personnalisées sur un PDA. Le système se base sur la localisation, la direction et les préférences de l'utilisateur pour chercher les endroits culturels ou touristiques qui pourraient intéresser l'utilisateur. Le système se base sur le modèle BDI (Beliefs, Desires and Intentions) [Rao et Georgeff 91] pour déduire l'état mental de l'utilisateur.

WebMate [Keeble et Macredie 00] : est un agent d'aide à la recherche d'informations sur le web. Il apprend le profil de l'utilisateur, lui prépare un journal d'actualités (de news) personnel et l'aide à améliorer sa recherche d'information. Le profil de l'utilisateur est un ensemble de mots clés trouvés dans les pages que celui-ci a sélectionnées. Ces pages sont des exemples positifs pour l'algorithme d'apprentissage. Cet algorithme utilise la fréquence d'apparition des mots dans les pages. Il existe un mécanisme d' « expansion » des mots clés : l'adjonction de mots de connotation voisine à un mot clé donné afin d'en préciser la signification.

Smart Radio [Hayes et Cunningham 01 ; Hayes et Cunningham 04] : permet aux utilisateurs d'écouter leur musique préférée ou un programme musical recommandé par le système. Afin de connaître leurs goûts musicaux, Smart Radio demande à ses utilisateurs de noter de 1 à 5 (5 étant la note la plus élevée) un ensemble de morceaux musicaux, ou des programmes musicaux, afin de constituer des profils utilisateurs. L'utilisateur a ainsi la possibilité de se constituer une playlist personnelle pour une écoute ultérieure de ces morceaux préférés. Mais, grâce à ce profil, le système peut trouver un ou plusieurs utilisateurs ayant des goûts communs en matière de musique et ainsi recommander à chacun la playlist des autres.

MAPIS (MultiAgent Personalized Information System) [Petit-Rozé 03] : développé au LAMIH, MAPIS est un système de personnalisation à base d'agents logiciels qui aident les usagers des transports publics de personnes dans leur choix d'itinéraire. Le profil de l'utilisateur correspond aux poids associés par rapport aux différents modes (bus, train, metro, marche, …), la durée du voyage, le nombre de changement et le coût (prix) du voyage. MAPIS utilise un mécanisme d'apprentissage par renforcement pour la gestion du profil utilisateur. Nos recherches se situent dans le prolongement de

ce travail ; l'un des objectifs de nos travaux étant de rendre plus générique MAPIS afin qu'il puisse prendre en compte d'autres types de personnalisation (prendre en compte la diversité des plates-formes d'interaction, personnaliser les informations de perturbations, etc.).

1.2.3.2. Systèmes de personnalisation pour l'aide au développement de système d'information personnalisé

D'autres systèmes ont pour objectif de faciliter la conception des systèmes d'information personnalisée. Ils se présentent généralement sous forme d'application configurable assurant la représentation du profil utilisateur et les mécanismes d'inférence pour le choix de la solution à proposer. La collecte d'information utilisateur est effectuée au niveau de l'application interagissant directement avec l'utilisateur final qui transmet ces données au système de personnalisation.

BGP-MS (Belief, Goal and Plan Maintenance System) [Kobsa et Pohl 95][Pohl et Höhle 97] : est un système de modélisation de l'utilisateur qui permet de prendre en compte les buts, croyances et connaissances de l'utilisateur. Il fonctionne selon différents types d'inférence (il comprend un moteur d'ordre 1) à partir d'hypothèses basées sur un questionnaire initial, les actions observées et des connaissances sur un ensemble de sous-groupes prédéfinis. Ce système peut être utilisé dans un serveur avec multi-utilisateurs et multi-applications.

BroadWay [Trousse et al., 99] : est un système qui s'utilise comme proxy pour la personnalisation du parcours du web. Il se base sur la navigation des utilisateurs pour recommander des liens à un utilisateur particulier, en exploitant pour cela un moteur de raisonnement à base de cas. Typiquement, il permet, par exemple, d'anticiper les besoins informationnels de l'utilisateur en lui proposant un lien sur la page sur laquelle l'utilisateur aimerait accéder. BroadWay est indépendant du navigateur.

Eperson [Dickinson et al., 03] : l'objectif du projet est de fournir une plate-forme commune ouverte permettant à des agents logiciels de pouvoir assister l'utilisateur tout en préservant la confidentialité des informations sur l'utilisateur. Le système se présente sous forme d'un serveur fournissant des services web pour la gestion des profils utilisateur. Les données utilisateurs sont organisées sous forme d'ontologie [Chandrasekaran et al., 99] en utilisant le langage DAML+OIL [Van Harmelen et al., 01].

1.2.3.3. Systèmes de personnalisation commerciaux

Des systèmes de personnalisation commerciaux sont aussi disponibles. Ils permettent la mise en œuvre de la personnalisation dans des systèmes d'information existants ou en construction. Ces systèmes sont généralement utilisés pour des objectifs de marketing dans le commerce électronique. On parle souvent de systèmes de Gestion de la Relation Client (ou CRM, *Customer Relationship Management*).

NetP 7[11] : est un système qui permet de prédire les centres d'intérêts des utilisateurs pour le commerce électronique. La prédiction s'effectue à partir d'inférences sur des données fournies explicitement par l'utilisateur et des données recueillies implicitement par les requêtes et les commandes (achats de produits) de l'utilisateur. Ce système peut être utilisé pour les applications web, les centres d'appel téléphonique, le courriel, les catalogues publicitaires, etc. Ce système est, par exemple, utilisé sur le site de vente Musician's Friend[12] pour la recommandation des produits de musique.

WebSphere[13] **:** IBM fournit dans sa suite WebSphere une application logicielle permettant de détecter les tendances et les préférences des utilisateurs. Cette application gère le contenu et la structure du site commercial en les adaptant par rapport au client. Par exemple, la personnalisation d'information sur le site de vente de matériels de randonnée REI[14] est effectuée avec WebSphere.

[11] http://www.tornago.com
[12] http://www.musiciansfriend.com
[13] http://www.ibm.com
[14] http://www.rei.com/

PassPort.Net [Oppliger 04] : est un serveur de profil utilisateur. L'utilisateur s'inscrit au service en donnant ses données personnelles. Ces données sont exploitées par les applications externes (intégrant leurs propres méthodes de personnalisation) pour fournir à l'utilisateur des services personnalisés correspondant à son profil. L'objectif global de PassPort.Net est de permettre à l'utilisateur de bénéficier de services personnalisés dans des applications différentes sans pour autant lui demander à chaque applications de préciser ses données personnelles ou ses préférences. Par exemple, MSN[15] utilise le serveur PassPort.net pour retrouver les informations de l'utilisateur sur les sites et les applications qu'il propose (MSN hotmail, MSN Messenger, MSN Music, etc.).

1.2.4. Synthèse et discussions

De nombreux travaux sont effectués pour la construction de systèmes de personnalisation (voir Tableau 1.2, pour une synthèse sur les différents systèmes de personnalisation). On distingue deux approches pour la construction de systèmes de personnalisation.

La première approche consiste à fournir un système interactif qui incorpore lui-même (ad-hoc) la personnalisation comme IFM, RESCUER et SmartRadio. Ces systèmes se présentent généralement sous forme d'agents logiciels (voir Letizia, IFWeb, Concall, InfoSleuth, Guilliver's Genie, WebMate et MAPIS) qui assurent les fonctions de recherche, de filtrage et de présentation des informations. Ces systèmes présentent l'avantage d'être directement en contact avec l'utilisateur ce qui facilite la collecte de données sur le comportement de l'utilisateur pour une personnalisation plus fine. Leur inconvénient majeur réside dans le fait que la réutilisation de ces systèmes est très limitée.

La deuxième approche consiste à fournir un système dédié à la personnalisation et qui interagit avec une application tierce pour la personnalisation (voir ePerson, BGP-MS, Broadway, NetP 7, Passport.Net et WebSphere). Les fonctions majeures pour ces systèmes consistent à gérer le profil des utilisateurs et la sélection des données pertinentes dont les applications tierces se chargent de présenter à l'utilisateur. Ces systèmes sont beaucoup plus flexibles que les précédents. Ils sont généralement utilisés pour personnaliser plusieurs applications susceptibles d'être utilisées par les mêmes utilisateurs. Outre les avantages dont bénéficie l'utilisateur final (même profil pour les différentes applications, utilisation par le système des expériences utilisateur d'une application à une autre, …), les coûts de développement d'applications personnalisées sont sensiblement réduits. On développe une fois le système de personnalisation et on intègre les applications voulant fournir une interaction personnalisée au système de personnalisation déjà opérationnelle. Ceci permet aussi de supporter la personnalisation multi-canal. Bien évidemment, un protocole de communication est nécessaire pour permettre la communication entre le système de personnalisation et les applications tierces.

Les systèmes existants sont généralement destinés à un type particulier de personnalisation et incorporent des méthodes de personnalisation bien définies. Il est très rare de retrouver un système qui assure aussi bien la personnalisation du contenant que du contenu, par exemple. Ceci nécessiterait l'intégration, dans un seul système, de différentes méthodes (souvent lourdes) de collecte et de gestion de profil utilisateur pour répondre aux différents types de personnalisation. L'idéal serait de disposer d'un système de personnalisation assez générique (qui peut supporter les différents types de personnalisation) et favorisant une intégration incrémentale des différentes méthodes pour la personnalisation. Les systèmes existants ne le permettant pas ou très peu, nous nous proposons de concevoir notre propre système de personnalisation (Cf. chapitre 4).

[15] http://www.msn.com

		Systèmes	Multi-application	Distribution	Evolutivité	Recherche d'information	Collecte d'information	Type de filtrage	Type de personnalisation
Systèmes universitaires ou libres	à base d'agents	Letizia	-	-	-	√	implicite	[C] : Terme Frequency-Inverse Document Frequency (TF-IDF)	Recommandation d'une page web
		IFWeb	-	-	-	√	implicite	réseaux sémantiques [P]	filtrage et tri d'information
		ConCall	-	-	-	√	implicite / explicite	[P] [RC] : stéréotypes et TF-IDF	filtrage, tri et rappel d'information
		InfoSleuth	-	√	√	√	implicite / explicite	[P] : ontologie	reformulation de requête
		Gulliver's Genie	-	√	-	√	implicite / explicite	[P] [C] : Belief, Desire and Intention (BDI)	Recommandation de lieux touristiques, guidage
		WebMate	-	-	-	√	implicite / explicite	[P] : TF-IDF	Filtrage d'information
		MAPIS	-	-	-	√	implicite / explicite	[P] : apprentissage par renforcement	Filtrage d'information
		Eperson	√	√	√	√	implicite / explicite	[P] [FC] : Ontologie	filtrage d'information
	autres	RESCUER	-	-	-	-	implicite	[C] : human plausible reasoning	Aide et conseil
		IFM	-	-	-	-	implicite	[[C] : stereotype, human plausible reasoning	Aide et conseil
		Smart Radio	-	-	-	√	implicite / explicite	[FC] : Pearson	recommandation de musique
		BGP-MS	√	-	-	-	implicite / explicite	[P] [RC] : logique modal	adaptation de l'interface
		BroadWay	√	-	-	-	implicite / explicite	[FC] : raisonnement à partir de cas	recommandation de liens
Systèmes commerciaux		NetP 7	√	?	?	?	implicite / explicite	Différentes techniques de datamining	recommandation de produits, personnalisation de contenu
		PassPort.Net	√	-	-	-	laissé aux applications tierces	laissé aux applications tierces	laissé aux applications tierces
		WebSphere	√	?	?	?	implicite / explicite	différentes techniques de datamining	recommandation de produits, personnalisation de contenu

Tableau 1.2. Etude comparative des systèmes de personnalisation

[FC] : filtrage collaboratif, [P] : filtrage basé sur le profil,
[C] : recommandation contextuel et [RC] : recommandation de communauté

1.3. Cadre applicatif : Personnalisation dans les systèmes d'information au service des usagers des transports terrestres de personnes

Ce travail s'inscrit dans le contexte applicatif de la personnalisation de l'information dans les transports terrestres de personnes. L'objectif repose sur la mise à disposition auprès de l'usager des transports, des informations pertinentes lui permettant de préparer son déplacement en lui présentant toute l'offre disponible sur l'itinéraire demandé en terme de choix de mode de transport, de temps de parcours, de prix, etc., de l'accompagner tout au long de son déplacement ou voyage, en offrant l'ensemble des éléments susceptibles de réduire son incertitude tout en prenant en compte la diversité des plates-formes d'interaction dont il dispose. Il s'agit donc de fournir à l'usager *« toute l'information nécessaire et uniquement l'information nécessaire »* [Uster 98]. Il s'agit aussi « *de compléter ce service d'intérêt collectif, qui reste vécu individuellement comme une contrainte, par une offre plus*

valorisante à titre personnel » [Uster 04]. Ce travail a été réalisé en collaboration avec la société Archimed et l'INRETS.

1.3.1. Besoins de personnalisation dans un système d'information au service des usagers des transports terrestres

Suite au large développement des sources d'informations distribuées et en réseau, plus spécialement sur Internet, est apparu l'intérêt de bénéficier de points de références, tels que des sites « portails » proposant un accès facilité, voire personnalisé, vers l'ensemble des ressources disponibles. L'information relative au transport est elle aussi concernée par la distribution des données, leur facilité d'accès et leur fiabilité. Une étude menée dans le cadre du projet européen Infopolis2 a ainsi montré quelles sont les attentes des voyageurs [Lecomte et Patesson 00] et conclut sur les services à développer en terme d'information :

- fournir de l'information en « temps réel » à l'usager ;
- proposer des solutions utilisant plusieurs modes de transport ;
- rendre l'information accessible par différents supports ;
- offrir des informations « personnalisées » au travers de systèmes interactifs.

En particulier, et dans le cadre du développement durable, il apparaît que la proposition d'informations aux voyageurs via un portail unique et personnalisé serait susceptible de favoriser l'utilisation des transports en commun, et ainsi tenter de réduire la part modale de l'automobile dans les déplacements. Ceci correspond également aux résultats d'une étude sur les besoins d'information des voyageurs [JPF Consultant 96]. Cette étude met en avant l'importance de la prise en compte de la catégorie du voyageur dans la définition des informations à présenter :

- informer les utilisateurs des possibilités qui leur sont offertes ;
- inciter les utilisateurs à utiliser les transports collectifs ;
- aider les utilisateurs à choisir le meilleur itinéraire ;
- aider les utilisateurs pendant leurs déplacements.

D'autres études ont décrit les besoins des usagers en terme d'information ainsi que l'intérêt des systèmes d'information au service des usagers d'un point de vue économique [Perreau 02]. Ces travaux mettent en évidence leur apport aussi bien dans le cas de déplacements contraints (travail, école) pour lesquels les usagers sont dépendants de l'heure d'arrivée que dans le cas de déplacements occasionnels. Dans le premier cas, ces systèmes d'information voyageur pourraient apporter une information précise et fiable sur les temps de parcours et les perturbations éventuelles sur les réseaux. Lorsque ces déplacements sont de plus conséquents tant en terme de distance que de temps, il est important que le trajet se passe dans les meilleures conditions de confort. Dans ce contexte, ils faciliteraient le bon choix en terme de mode de transport, d'itinéraire, etc., en fonction des capacités et préférences de chaque usager. Dans le second cas, ils pourraient, d'une part, éclairer les usagers sur les offres de transport pour se rendre à la nouvelle destination, et d'autre part, les aider à organiser leurs déplacements.

1.3.2. Approches existantes pour la personnalisation de l'information transport

Cette section présente différents projets et systèmes visant à améliorer la qualité de l'information délivrée aux voyageurs. Nous citerons :

PIEPSER (Personalised information on disruptions to public transport exclusive to users of public transport) [Hoyer et Czogolla 02] : le but du projet est de prévenir les usagers des retards et des interruptions éventuelles dans les transports en Allemagne. Des alternatives combinant plusieurs

modes de transport sont proposées par SMS pour répondre aux besoins spatial, temporel et modal. Le système est personnalisé par rapport aux perturbations des transports :

- une alternative spatiale est proposée seulement dans le cas d'une interruption d'un moyen de transport ;
- le passager peut se voir aussi proposer de quitter tôt ou de quitter tard sa résidence si le train est trop en retard ou annulé ;
- d'autres moyens de transport (vélo, auto) sont proposés dans le cas où le passager pourrait rater sa correspondance, s'il y a assez de temps ou s'il y a assez de place de parking.

Le passager (régulier) s'abonne au service en donnant l'heure et le lieu de départ, et choisit les lignes de correspondance. Il donne aussi son numéro de portable.

Le recueil des informations s'effectue par l'intermédiaire :

- des horaires des différents modes de transport ;
- de l'observation vidéo de l'état du trafic routier pour identifier, par exemple, les embouteillages, l'état des parcs de stationnement, etc. *;*
- de la collaboration avec la municipalité pour connaître, par exemple, les travaux éventuels, etc. ;
- de la carte du réseau routier pour calculer le parcours.

TISONI (Traveller Information System ON Internet) [Lam et Xie 02] : ce système est développé pour fournir une information personnalisée et combinant plusieurs modes de transport dans l'Ouest de Singapour. La personnalisation couvre deux aspects :

- reconnaissance des utilisateurs et leurs préférences ;
- personnalisation de l'information fournie à l'utilisateur par apprentissage.

Le noyau de TISONI est le modèle de choix du chemin qui est étalonné à partir du comportement de l'utilisateur (en mode de transport) et le choix des routes. Le système supporte une architecture trois-tiers : la base de données, le moteur de calcul et l'interface utilisateur.

TISONI distingue trois sortes de données :

- base de données statiques (les stations, les distances, les lignes, …) ;
- base de données temps réel reliée avec la base de donnée statique ;
- base de données sur l'information utilisateur (nom, âge, sexe, tolérance de la marche, …).

L'utilisateur s'inscrit en fournissant ses informations personnelles comme l'âge, son revenu, sexe, vitesse de marche, etc. Le système propose le chemin préféré par l'utilisateur et affiche sur une carte les différents itinéraires en donnant les arrêts, les temps d'arrêt, le temps de parcours, …

Les données temps réel sont collectées par GPS (Global Positioning System), capteur, surveillance vidéo, interrogation des véhicules et autres. Les données statiques sont collectées dans les agences de transport.

La rapidité des réponses (temps de recherche, de transmission et de calcul de données), l'intelligence (capacité à reconnaître les utilisateurs, d'extraire automatiquement les données de la base, d'afficher les itinéraires) et l'efficacité (efficacité des algorithmes à trouver les chemins optimaux) sont les critères optés pour l'évaluation de la performance du prototype.

MyBus [Maclean et Dailey 02] : ce système délivre des informations transport (bus) pour la région de Seattle aux Etats Unis. Il est accessible par le web, wap et palm. L'utilisateur fournit le lieu de départ et le numéro de la route et, le système donne la destination, l'heure et l'état de départ. Pour le web un lien existe pour visualiser le plan.

Sur le wap et le palm, l'utilisateur peut fournir les données sous format numérique pour répondre à des besoins pratiques d'utilisabilité. Sinon, le système intègre une procédure de choix par route ou par région à travers une liste déroulante. Une notation a été adoptée pour l'affichage des données :

- « d » : le bus vient de partir ;
- « * » : il n'y a pas eu d'estimation pour cet événement, le temps fourni provient du programme des horaires ;
- « : » : le temps fourni est estimé par un moteur de calcul d'itinéraire.

Les données sur la localisation des bus se collectent au moyen d'AVL (Automatic Vehicle Location). La Figure 1.2 illustre un exemple d'affichage sur le wap des départs des bus pour l'arrêt UNIVNE45.

Figure 1.2. Visualisation des horaires sur un téléphone portable [Maclean et Dailey 02]

AGENPERSO (AGENts logiciels PERSOnnels) [Petit-Rozé et al., 03a ; Petit-Rozé et al., 03b] : l'objectif du projet réalisé au sein du LAMIH dans le cadre du PREDIT, est d'une part, d'aider l'usager des transports dans sa démarche de recherche d'information et, d'autre part, de lui fournir un résultat personnalisé. AGENPERSO est une application[16] qui fournit une aide personnalisée dans la préparation d'itinéraires combinant un ensemble de modes de transport (bus, tramway, métro, marche, etc.). L'application intègre le système de personnalisation MAPIS (voir §1.2.3). L'usager s'identifie auprès d'une interface web et peut alors lancer une requête de recherche d'itinéraire. Le système se base sur les préférences de l'utilisateur pour lui fournir l'itinéraire correspondant à son profil. Le profil de l'utilisateur correspond aux poids associés à quatre critères (le plus rapide, le plus court chemin, le moins cher et le moins de correspondance) et aux poids associés aux modes de transport (bus, metro, taxi, TER, tramway, marche et métro).

Le système permet aussi une recherche thématique d'itinéraire. Par exemple, l'utilisateur peut simplement spécifier qu'il veut aller faire des courses et le système lui fournit la liste des magasins et le conseille sur l'itinéraire à prendre.

De nombreux autres systèmes d'information transport personnalisée existent. La plupart des ces systèmes avaient pour objectifs initiaux la mise à disposition de l'usager d'informations relatives au transport. Devant les masses d'informations et l'exigence croissante de l'usager en terme de pertinence d'information, des services transport personnalisés ont commencé à apparaître. Au niveau national, on citera la SNCF[17] (transport interurbain), RATP[18] (transport en île de France) et Le Pilote[19] (transport de la région marseillaise et du département des bouches du Rhône) ; et au niveau

[16] L'application est à l'état de maquette.
[17] http://www.sncf.fr
[18] http://www.ratp.fr
[19] http://www.lepilote.fr

européen nous citerons YTV[20] (transport dans la ville de Helsinki en Finlande) et TFL[21] (transport à Londres et ses environs) qui proposent de mémoriser les trajets les plus courants de l'utilisateur pour lui éviter de ressaisir les lieux de départs et d'arrivées. D'autres services personnalisés, comme l'envoi de SMS pour signaler des perturbations, sont parfois fournis.

1.3.3. Synthèse et discussions

Accompagner le citoyen dans ses déplacements quotidiens comme pour la découverte de territoires urbains inconnus est sans aucun doute un des leviers potentiels d'action pour aller vers une mobilité durable, garant d'un meilleur équilibre entre l'automobile et les transports collectifs. Aujourd'hui, les systèmes d'information liée aux transports, quand ils existent, ne présentent qu'une vue parcellaire, et parfois trop complexe, des offres de déplacements sur un territoire donné. La profusion, comme l'absence d'informations, ne permettent pas de proposer un système efficace d'aide à la décision. Une approche plus contextualisée, plus personnalisée pourrait relever ce défi [Petit-Rozé et al., 04].

Plusieurs travaux sont en cours pour la personnalisation de l'information transport. Ces travaux visent à personnaliser l'information « avant », « pendant » et parfois même « après » le déplacement tout en essayant de couvrir le plus de canaux et de plates-formes d'interaction possibles. Le Tableau 1.3 résume les systèmes d'information transport présenté ci-dessus selon cinq caractéristiques :

- les *modes de transport* correspondent aux types de transport (TER, bus, métro, tramway, etc.) considérés par le système.
- les *canaux* et les *plates-formes d'interaction* décrivent les moyens d'accès (web, wap, SMS, etc.) du système d'information et les supports d'interaction (PC, téléphone mobile, PDA, etc.) au travers desquels il est destiné.
- les *principaux services* résument les informations et les applications auxquelles le système met à la disposition de l'utilisateur.
- la *personnalisation* présente les modes de collecte d'information utilisateur et les types de personnalisation fournis par le système.

[20] http://www.ytv.fi/eng/
[21] http://www.tfl.gov.uk/tfl/

Systèmes	Modes de transports	Canaux / Plate-formes d'interaction	Principaux services	Personnalisation
SNCF	TGV, Train, TER	Web, Mail, SMS / PC, Téléphone	- Recherche d'itinéraire inter-urbains - Réservation et achat de billets - Organisation du voyage de bout en bout (réservation voiture, hôtel, etc.) - Prévision du trafic TGV - Informations des perturbations du trafic par région	- Remplissage automatique des préférences (trajets habituels, confort, compagnons, réductions, etc.) à partir d'un questionnaire rempli lors de l'inscription - Envoi de SMS dans le cas d'une perturbation sur un trajet planifié par l'usager
RATP	Train (SNCF), RER, Métro Tramway, Bus, Marche	Web, SMS, Wap / PC, Téléphone	- Recherche d'itinéraire en île de France - Plans de proximité - Affichage des horaires - Comparaison entre temps de déplacements en TC et VP - Plans des lignes et des réseaux - Informations des perturbations	- Préférences (requêtes, choix, critères, etc.) sauvegardées pour les utiliser dans une autre connexion
LePilote	Bus, Tramway TER (SNCF), Métro, Marche	Web / PC	- Recherche d'itinéraire dans Marseille et ses environs - Recherche des horaires - Evénements culturels, sportifs, etc. - Informations sur les travaux prévus	- Précision des critères (plus rapide, sans métro, etc.) transports par l'utilisateur lors de la recherche d'itinéraire
TFL	Bus, Métro, Train, Tramway, Marche	WAP, SMS, Web, Mail / PC, Téléphone, PDA	- Recherche d'itinéraire à Londres - Tableau des départs - Informations connexes aux déplacements (Toilettes, accessibilité, etc.) - Tarification, achat et réservation de billets	- Préférences (requêtes, choix, critères, etc.) sauvegardées pour les utiliser dans une autre connexion
AGENPERSO	Bus, Métro, Tramway, Train, Marche	Web / PC	- Recherche d'itinéraire dans la région Nord Pas-de-Calais - Recherche d'itinéraire thématique	- Apprentissage automatique des préférences (modes, critères, etc.) de l'utilisateur
PIEPSER	Train, Bus, Tramway, Vélo, Voiture, Marche	SMS, Web, WAP / PC, Téléphone	- Recherche d'itinéraire en Allemagne - Horaires des lignes - Perturbations sur le trajet	- Inscription au service d'information personnalisée - Proposition des alternatives en cas de perturbation
YTV	Bus, Métro, Tramway, Train, Marche	Web / PC	- Recherche d'itinéraire à Helsinki - Tarification, cartes des lignes, liste des horaires	- Préférences (requêtes, choix, critères, etc.) sauvegardées pour les utiliser dans une autre connexion
TISONI	Modes transport urbains.	Web / PC	- Recherche d'itinéraire au Singapoure	- Apprentissage automatique des modes de transport préférés par l'utilisateur - Estimation des temps de parcours en fonction du trafic
MyBus	Bus	Web, WAP / PC, Palm, Téléphone	- Recherche d'itinéraire à Seattle	- Interface adaptée par rapport à la plate-forme d'interaction

Tableau 1.3. Synthèse des systèmes d'information transport étudiés

Conclusion

L'objectif de nos travaux est de fournir une méthode adaptée pour le développement de systèmes d'information personnalisés. Cette méthode devrait, en particulier, supporter différents types de personnalisation et garantir l'évolutivité du système. Ce chapitre a présenté un état de l'art sur la personnalisation de l'interaction homme-machine. De nombreux travaux sont effectués à ce sujet. La plupart de ces travaux visent à fournir des approches, des modèles et des méthodes pour la collecte d'information utilisateur, la gestion des profils utilisateurs et l'adaptation de l'interaction par rapport à ses besoins et ses préférences. Ces modèles et ces méthodes sont généralement adaptés pour répondre à une problématique précise relative à un type de personnalisation particulier.

L'étude des systèmes de personnalisation a montré que les systèmes existants effectuent généralement un seul type de personnalisation et incorporent des méthodes figées de collecte d'informations et d'exploitation de ces informations. Les rares systèmes qui soient relativement flexibles et évolutifs sont construits à base d'agents logiciels. Il semble donc que les agents logiciels se préteraient bien au développement de systèmes d'information personnalisés flexibles et évolutifs.

Cependant, très peu de travaux fournissent une méthode globale qui permettrait de guider l'analyse, la conception et la modélisation dans le processus de développement de système d'information personnalisé. Le chapitre suivant a donc pour objectif de rechercher une méthode adéquate pour l'analyse, la conception et la modélisation de tels systèmes.

Bibliographie du chapitre 1

[Aas 97] Aas K. *A Survey on Personalised Information Filtering Systems for the World Wide Web* ; Research Report n° 922, Oslo, Norway, Norwegian Computing Center, December 1997.

[Abowd et al., 99] Abowd G.D., Dey A.K., Brown P.J., Smith M. and Steggles P. Towards a Better Understanding of Context and Context-Awareness. *Lecture Note In Computer Science*, Vol. 1707. *Proceedings of th 1st international symposium on Handheld and Ubiquitous Computing*, Karlsruhe, Germany, pp. 304-307, September 1999.

[Aggarwal et al., 99] Aggarwal C.C., Wolf J.L., Wu K-L. and Yu P.S. Horting Hatches an Egg : A New Graph-Theoretic Approach to Collaborative Filtering. In *Proceedings of KDD'99*, *5th International Conference on Knowledge Discovery and Data Mining*, pp. 201-212, San Diego, USA, August 1999.

[Allen 97] Allen R. B. Mental Models and User Models. In Helander, M., Landauer, T.K. and Prabhu, P. (Eds.), *Handbook of Human-Computer Interaction*. Elsevier Science, pp. 49-63, 1997.

[Ambrosini et al., 97] Ambrosini L., Cirillo V. and Micarelli A. A Hybrid Architecture for User-Adapted Information Filtering on the World Wide Web. In Jameson, A., Paris, C. and Tasso, C. (Eds.), *User Modeling: Proceedings of the Sixth International Conference*, UM97, New York : Springer Wien, pp. 59-61, 1997.

[Asnicar et Tasso 97] Asnicar F.A. and Tasso C. IfWeb: a Prototype of User Model-Based Intelligent Agent for Document Filtering and Navigation in the World Wide Web. In *Proceedings of the workshop « Adaptive Systems and User Modeling on the World Wide Web »,* Chia Laguna, Sardina, pp. 3-12, 2-5 June 1997.

[Balabanovic et Sholham 97] Balabanovic M. and Sholham Y. Fab : Content-based, collaborative recommendation. *Communications of the ACM*, vol. 40, pp. 66-72, March 1997.

[Banks et al., 97] Banks S.B., Harrington R.A., Santos E., Brown M. Jr and Brown S.M. Usability testing of an intelligent interface agent. In *Interfaces 97*, Montpellier, France, May 1997.

[Basu et al., 98] Basu C., Hirsh H. and Cohen W. : Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In C. Rich and J. Mostow (Eds.), *Proceedings of the 15th National Conference on Artificial Intelligence*, pp. 714-720. Madison, USA, AAAI Press/MIT Press, 1998.

[Bazsalicza et Naïm 01] Bazsalicza M. et Naïm P. *Data mining pour le web : Profiling – Filtrage collaboratif – Personnalisation client*. Paris : Editions Eyrolles, 2001.

[Beale et Wood 94] Beale R. and Wood A. Agent-Based Interaction. In *People and Computers IX: Proceedings of the HCI'94*, pp. 239-245, Glasgow, UK, Cambridge University Press, August 1994.

[Berrut et Denos 03] Berrut C. et Denos N.. Filtrage collaboratif. In *Assistance intelligente à la recherche d'informations,* E. Gaussier, M.H. Stéfanini (dir.), Hermès-Lavoisier, pp. 241-269, 2003.

[Bond et Gasser 88] Bond A. and Gasser L. *Readings in distributed artificial intelligence*. Morgan Kaufman, San Mateo, CA, 1988.

[Bouzeghoub et Kostadinov 04] Bouzeghoub M et Kostadinov D. *Une approche multidimensionnelle pour la personnalisation de l'information*. Rapport PRiSM, Versailles, France, 2004.

[Breese et al., 98] Breese J.S., Heckerman D. and Kadie C. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *Proceedings of the Fourtheenth Conference on Uncertainty in Artificial Intelligence*. Madison, Morgan Kaufmann Publisher. July 1998.

[Browne et al., 90] Browne D., Totterdell P. and Norman M (Eds.). *Adaptive User Interfaces*. Academic Press, Computer And People Series, 1990.

[Castelfranchi 95] Castelfranchi C. Guarantees for autonomy in cognitive agent architecture. In M. Wooldridge and N.R. Jennings (Eds.)*, Intelligent Agents-Agent Theories, Architectures, and Languages*, volume 890 of Lecture Notes in Computer Science (subseries LNAI), pages 56-70, Springer-Verlag, 1995.

[Chandrasekaran et al., 99] Chandrasekaran B., Josephson J. R. and Benjamins V. R. What Are Ontologies, and Why Do We Need Them? *IEEE Intelligent Systems*, Vol. 14(1), pp. 20-26, 1999.

[Chin 91] Chin D. Intelligent interfaces as agents. In J. W. Sullivan and S. W. Tyler (Eds.), *Intelligent User Interfaces*, Addison-Wesley, New York, pp. 177-206, 1991.

[ChoiceStream 04] *ChoiceStream. Review of Personalization Technologies : Collaborative Filtering vs. ChoiceStream's Attributized Bayesian Choice Modeling*. Technology Brief. Disponible à l'adresse : www.choicestream.com/pdf/ChoiceStream_TechBrief.pdf

[Cinquin et al., 02] Cinquin L., Lalande P. A et Moreau N. *Le projet ECRM : Relation client et Internet*. Editions Eyrolle, Paris, 2002.

[Cohen et Levesque 90] Cohen P.R. and Levesque H.J. Intention is choice with commitment. *Artificial intelligence*, 42, pp. 213-261, 1990.

[Cosley et al., 02] Cosley D., Lawrence S. abd Pennock D. M. REFEREE : An open framework for practical testing of recommender systems using ResearchIndex. In *Proceeding of the 28th VLDB Conference*. Hong Kong, China, August 2002.

[Coyle et Cunninghan 02] Coyle L. and Cunninghan P. A Case-Based Personal Travel Assistant for Elaborating User Requirements and Assessing Offers. *Proceedings of the 6th European Conference, ECCBR 2002*, S. Craw, A. Preece (Eds.). LNAI Vol. 2416, pp. 505-518, Springer-Verlag, 2002.

[Deschaine et al., 00] Deschaine L., Brice R. and Nodine M. Use of InfoSleuth to Coordinate Information Acquisition, Tracking and Analysis in Complex Applications. In *Proceedings of Advanced Simulation Technologies Conference*, Washington, D.C., USA, April 2000.

[Dickinson et al., 03] Dickinson I., Reynolds D., Banks D., Cayzer S. and Vora P. User Profiling with privacy : A framework for Adaptive Information Agents, M. Klush et al. (Eds.) : *Intelligent Information Agents*, LNAI 2586, pp. 123-151, Berlin : Springer-verlag, 2003.

[Erceau et Ferber 91] Erceau J. and Ferber J. L'Intelligence Artificielle Distribuée. *La Recherche*, n° 233, pp. 750, Juin 1991.

[Ferber 95] Ferber J. *Les systèmes multi-agents : Vers une intelligence collective*. Paris : InterEditions, 1995.

[Finin et al., 97] Finin T., Labrou Y. and Mayfield J. KQML as an Agent Communication Language. In J. Bradshaw (Eds.), *Software Agents*, pp. 291-316. Cambridge, USA, MIT Press, 1997.

[FIPA 02] FIPA. *FIPA ACL Message Structure Specification*. Rapport technique N°SC00061G, Geneva, Switzerland, December 2002.

[Fisher 96] Fisher D. Iterative Optimization and Simplification of Hierarchical Clusterings. *Journal of Artificial Intelligence Research*, n° 4, 1996, pp. 147-179.

[Florins et Vanderdonckt 04] Florins M. and Vanderdonckt J. Graceful Degradation of User Interfaces as a Design Method for Multiplatform Systems. In *Proceedings of the 2004 International Conference on Intelligent User Interfaces IUI 2004*, Funchal, Madeira Island, Portugal, January 13-16, pp. 140-147, 2004.

[Fuggetta et al., 98] Fuggetta A., Picco G.P., and Vigna G. Understanding code mobility. *IEEE Transactions on software engineering*, 24(5), 342-361, May 1998.

[Goldberg et al., 01] Goldberg K., Roeder T., Gupta D. and Perkins C.. Eigentaste : A Constant Time Collaborative Filtering Algorithm. *Information Retrieval Journal*, vol. 4(2), pp. 133-151, 2001.

[Goldberg et al., 92] Goldberg D., Nichols D., Oki B. and Terry D. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, vol. 35, pp. 61-70, 1992.

[Gray et al., 00] Gray R.S., Cybenko G., Kotz D. and Rus D. *Mobile agents: Motivations and State of the Art*. Technical Report TR2000-365, Dept. of Computer Science, Dartmouth College, 2000.

[Grislin-Le Strugeon et al., 01] Grislin-Le Strugeon E., Adam E. et Kolski C. Agents intelligents en interaction Homme-Machine dans les Systèmes d'information. In C. Kolski (Eds.), *Environnements évolué et évaluation de l'I.H.M.*, *Interaction Homme-Machine pour les S.I. 2*, Paris : Editions Hermes, pp. 209-248, 2001.

[Hagimont et Layaïda 02] Hagimont D. et Layaïda N. Adaptation d'une application multimédia par un code mobile. *Technique et Science Informatiques* (TSI), numéro spécial Agents et code mobile, Vol. 21, N° 6, pp. 877-897, 2002.

[Hayes et Cunningham 01] Hayes C. and Cunningham P. Smart Radio – community based music radio. *Knowledge-Based Systems*, n° 14, 2001, pp. 197-201.

[Hayes et Cunningham 04] Hayes C. and Cunningham P. Context boosting collaborative recommendations. *Knowledge-Based Systems*, n° 17, pp. 131-138, 2004.

[Herlocker et al., 99] Herlocker J. L, Konstant J. A., Brochers A. and Riedl J. A algorithmic framework for perfomrming collaborative filtering. In *proc. 1999 Conf. Research and Developpement in Information retrieval*, pp 230-237, Berkeley, CA, Août 1999.

[Hirsh et al., 00] Hirsh H., Basu C. and Davison B. Learning to personalize. *Communications of the ACM*, 43(8):102-106, 2000.

[Höök 00] Höök K. Steps to take before intelligent user interfaces become real. *Interacting with computers*, pp. 409-426, 12, 2000.

[Hoyer et Czogolla 02] Hoyer R. and Czogolla O. Approach to personalised information services to public transport. *In 9th world congress on Intelligent transportation systems*, Chicago Illinois, october 14-18, 2002.

[Jennings et al., 98] Jennings N. R., Sycara,K. and Wooldridge M. A Roadmap of Agent Research and Development. *Autonomous Agents and Multi-Agent Systems*, Vol. 1, pp. 7-38, 1998.

[JPF Consultant 96] JPF Consultant. *Etude de faisabilité sur la mise en place d'un système d'information sur l'offre de transport collectif de voyageur* - Phase 4 : Préconisation ; Rapport technique, octobre 1996.

[Kay 01] Kay J. User Modeling for Adaptation. In Stephanidis, C. (Ed.) *User Interface For All.*, London : LEA publishers, 2001, pp. 271-294.

[Keeble et Macredie 00] Keeble R.J. and Macredie R.D. Assistant agents for the world wide web intelligent interface design challenges. *Interacting with computers,* Vol. 12 No. 4, pp. 357-381, 2000.

[Kimball et Merz 00] Kimball R. et Merz R.. *Le data web house : Analyser les comportements clients.* Paris : Editions Eyrolles, 2000.

[Kobsa et Pohl 95] Kobsa A. and Pohl W. The User Modeling Shell System BGP-MS. *User Modeling and User-Adapted Interaction,* Vol. 4, pp. 59-106, 1995.

[Kolski et Le Strugeon 98] Kolski C. and Le Strugeon E. A review of intelligent human-machine interfaces in the light of the ARCH model. *International Journal of Human-Computer Interaction*, 10(3), pp. 193-231, 1998.

[Korfhage 97] Korfhage R.R. *Information storage and retrieval*. Wiley Computer Publishing. ISBN 0-471-14-338-3, 1997.

[Labidi et Lejouad 93] Labidi S. et Lejouad W. *De l'intelligence artificielle distribuée aux systèmes multi-agents*. Rapport de recherche 2004, INRIA, Sophia-Antipolis, France, août 1993.

[Lam et Xie 02] Lam S. H. and Xie F. Provision of Personalised Transit Travel Information System and Architecture. *In 9th world congress on Intelligent transportation systems*, Chicago Illinois, october 14-18, 2002.

[Lashkari 97] Lashkari Y., Metral M., and Maes P. Collaborative interface agents. In *M.N. Huhns and M.P. Singh (eds), Readings in Agents*, Morgan Kaufmann, 1997.

[Laurel 97] Laurel B. Interface agents: metaphors with character. In J.M Bradshaw (Ed.), *Software agents,* pp.67-77, Mento Park, CA : AAAI Press, 1997.

[Lecomte et Patesson 00] Lecomte N. et Patesson R. Le panel des voyageurs : une étude des activités et des besoins d'information des utilisateurs des transports publics. In *Actes de la conférence ERGO-IHM* (ERGO-IHM'00 - Biarritz, France, 3-6 octobre), D. Scapin and E. Vergison (Eds.), pp. 129-135, 2000.

[Lieberman 95] Lieberman H. Letizia : An Agent That Assists Web browsing. In *International Joint Conference on Artificial Intelligence (IJCAI'95 – Montreal, Canada, August 20-25).* San Francisco, USA, Morgan Kaufman Publishers, 1995.

[Lieberman 97] Lieberman H. Autonomous interface agents. In *Proceedings of CHI'97 (Human factors in computing systems)*, pp. 67-74, Atlanta, GA USA, March 22-27, ACM Press, 1997.

[Lieberman et al., 01] Lieberman H., Fry C. and Weitzman L. Exploring the Web with Reconnaissance Agents. In *ACM Conference on Human-Computer Interface*. ACM Press, pp. 69-75, August 2001.

[Linden et al., 97] Linden G., Hanks S. and Lesh N. Interactive Assessment of User Preference Models : The Automated Travel Assistant. In Jameson, A., Paris, C. and Tasso, C. (eds), *User Modeling: Proceedings of the Sixth International Conference*, *UM97*. New York : Springer Wien, pp. 67-78, 1997.

[Lumineau 03] Lumineau N. *Un tour d'horizon du filtrage collaboratif*. Travail réalisé dans le cadre de l'AS Personnalisation de l'Information, janvier – décembre 2003, disponible à l'adresse : www.prism.uvsq.fr/recherche/themes/sial/cnrs/Fichiers/Rapport\TourDHorizonDuFiltrageCollaboratif-LIP6.pdf.

[Maclean et Dailey 02] Maclean S. D. And Dailey D. J. The use of wireless Internet Service to access Real-Time Transit Information. *In 9th world congress on Intelligent transportation systems*, Chicago Illinois, october 14-18, 2002.

[Maes 94] Maes P. Agents that reduce work and information overload. *Communications of the ACM* 37(7), pp. 30-40, 1994.

[Manber et al., 00] Manber U., Patel A. and Robison J. Exeprience with personalization on Yahoo! *Communictions of the ACM*, 43(8) : 35-39, August 2000.

[Mandiau et Grislin-Le Strugeon 01] Mandiau R. et Grislin-Le Strugeon E. Systèmes Multi-Agents. *Techniques de l'ingénieur*, Paris, 2001.

[Mathieu et Verrons 03] Mathieu P. et Verrons M.H. A generic negotiation model for MAS using XML. *International Workshop series Agents for Business Automation : Research and Development* (ABA03) (IEEE), pp. 4262-4267, 2003.

[Melville et al., 02] Melville P., Mooney R.J. and Nagarajan R. Content-Boosted Collaborative Filtering for Improved Recommendations. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI-2002)*. Edmonton, Canada, July, AAAI Press, pp. 187-192, 2002.

[Moghrabi et Eid 98] Moghrabi C. and Eid M.S. Modeling users through an expert system and a neural network. *Computers & Industrial Engineering*, Vol. 35, No 3-4, pp. 583-586, 1998.

[Murgue 05] Murgue T. De l'importance du pré-traitement des données pour l'utilisation de l'inférence grammaticale en Web Usage Mining. In *Workshop Modélisation Utilisateurs et Personnalisation de l'Interaction Homme-Machine*, Extraction et Gestion de Connaissances (EGC 2005), Paris, 2005.

[Newell 82] Newell A. The knowledge level. *Artificial intelligence*, 18, pp. 87-127, January 1982.

[Nielsen 93] Nielsen J. *Usability Engineering*. Academic Press : London, 1993.

[Norman 83] Norman D. A. Some observations on mental models. In *Mental models*, D. Gentner and A. L. Stevens (Eds.), pp. 7-14, New Jersey : Lawrence Erlbaum, 1983.

[O'Hare et O'Grady 03] O'Hare G.M.P. and O'Grady M.J. Gulliver's Genie: a multi-agent system for ubiquitous and intelligent content delivery. *Computer Communications*, Vol. 26, pp.1177-1187, 2003.

[Oppermann et Simm 94] Oppermann R. et Simm H. Adaptability : User-initiated individualization. In R. Oppermann (Ed.), *Adaptative User Support: ergonomic design of manually and automatically adaptable software*, pp. 14-66, Lawrence Earlbaum, 1994.

[Oppliger 04] Oppliger R. Microsoft .NET Passport and identity management. *Information Security Technical Report*, Vol 9, Issue 1, pp. 26-34, January-March 2004.

[Pennock et al., 00] Pennock D., Horvitz E., Lawrence S. and Giles C. Collaborative Filtering by Personality Diagnosis : A Hybrid Memory- and Model-Based Approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, UAI-2000, Morgan Kaufman, San Francisco, pp. 473-480, 2000.

[Perreau 02] Perreau C. *Les systèmes d'information multimodale : apports et potentialités dans l'optimisation des déplacements urbains*. Thèse de doctorat, Institut d'Etudes Politiques, Paris, juillet 2002.

[Petit-Rozé 03] Petit-Rozé C. *Organisation Multi-Agent au service de la personnalisation de l'information : Application à un système d'information multimodale pour le transport terrestre de personnes*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Décembre 2003.

*[Petit-Rozé et al., 03a] Petit-Rozé C., Anli A., Grislin-Le Strugeon E., Abed M. et Kolski C. *AGENPERSO : Interface homme-machine à base d'AGENts Logiciels PERSOnnels d'information aux usagers des TC*. Rapport final de projet PREDIT, LAMIH, Valenciennes, janvier 2003.

*[Petit-Rozé et al., 03b] Petit-Rozé C., Kolski C., Grislin-Le Strugeon E., Anli A., Abed M. et Uster G. AgenPerso : Interface homme-machine à base d'agents logiciels personnels d'information au service des usagers des transports collectifs. *15ème Conférence Francophone sur l'Interaction Homme-Machine*. Caen, ACM Press, Novembre 2003.

*[Petit-Rozé et al., 04] Petit-Rozé C., Anli A., Grislin-Le Strugeon E., Abed M., Uster G. et Kolski C. Système d'Information Transport Personnalisée à base d'agents logiciels. *Génie Logiciel*, 70, pp. 29-38, 2004.

[Petit-Rozé et Grislin-Le Strugeon 02] Petit-Rozé C. et Grislin-Le Strugeon E. Systèmes d'information à base d'agents. In R. Mandiau, E. Grislin-Le Strugeon, A Péninou (Eds.), *Organisation et application des SMA*, Paris : Hermès, pp. 307-319, 2002.

[Pfitzmann 04] Pfitzmann B. Privacy in enterprise identity federation – policies for Liberty 2 single sign on. *Information Security Technical Report*, Vol 9, Issue 1, pp. 45-58, January-March 2004.

[Pohl 99] Pohl W. Logic-Based Representation and Reasoning for User Modeling Shell Systems. *User Modelling and User-Adapted Interaction*, Vol. 9, Number 3, pp. 217-282, 1999.

[Pohl et Höhle 97] Pohl W. and Höhle J. Mechanisms for flexible representation and use of knowledge in user modeling shell systems. In Jameson, A., Paris, C. and Tasso, C. (eds), *User Modeling: Proceedings of the Sixth International Conference*, UM97. New York : Springer Wien, pp. 403-414, 1997.

[Rao et Georgeff 91] Rao A.S. and Georgeff M.P. Modeling Rational Agents within a BDI Architecture. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, pp. 473-484, 1991.

[Rich 96] Rich C. Window sharing with collaborative interface agents. *SIGCHI Bulletin*, 28(1), pp. 70-78, January 1996.

[Salton 83] Salton G. and McGill M. *Introduction to modern information retrieval*. New York : McGraw-Hill, 1983

[Salton et Buckley 88] Salton T. and Buckley C. Term-Weighting Approaches in Automated Text Retrieval. *Information Processing and Management*, Vol 24(5), pp. 513-523, 1988.

[Schiaffino et Amandi 00] Schiaffino S. N and Amandi A. User profiling with Case-Based Reasoning and Bayesian Networks. In *Open Discussion Track Proceedings – International Joint Conference, IBERAMIA-SBIA 2000,* Atibaia, Brazil, pp. 12-21, 2000.

[Schneider-Hufschmidt et al., 93] Schneider-Hufschmidt M., Kühme T. and Malinkowski U. (Eds.). *Adaptive User Interfaces*. North Holland, 1993.

[Shardanand et Maes 95] Shardanand U. and Maes P. Social Information Filtering : Algorithms for Automating "World of Mouth". In *Proceeding of the CHI-95 Conference*. Denver, USA, ACM Press, May 1995.

[Soltysiak et Crabtree 98] Soltysiak S. J. and Crabtree I. B. Automatic learning of user profiles – towards the personalization of agent services. *BT Technologie*, Vol. 16 N° 3, pp. 110-117, July 1998.

[Stephanidis et al., 01] Stephanidis C., Paramythis A, Sfyrakis M. and Savidis A. A case Study in Unified User Interface Development: The AVANTI Web Browser. In C. Stephanidis (Ed.), *User Interfaces for All – concepts, methods and tools*. pp. 525-568, NJ Mahwah : Lawrence Erlbaun Associates, 2001.

[Thévenin et Coutaz 99] Thévenin D. et Coutaz J. Plasticity of user interfaces: framework and research agenda. In *Proceedings of Interact'99 seventh IFIP Conference on Human-Computer Interaction*, Edinburgh, Scotland, August 1999.

[Trousse et al., 99] Trousse B., Jaczynski M. and Kanawati R. Using user behavior similarity for recommandation computation : The broadway approach, In *Proceedings of the 8th international conference on Human Computer Interaction (HCI'99),* Munich, 1999.

[Ungar et Foster 98] Ungar L.H. and Foster D.P. Clustering Methods for Collaborative Filtering. *AAAI Workshop on Recommendation Systems,* M. Kaufman (Ed.) « Empirical analysis of predictive algorithms for collaborative filtering. Uncertainty in Artificial Intelligence », *Proceedings of the 14th Conference*, pp. 43-52, 1998.

[Uster 04] Uster G. Pour une mobilité raisonnée. In *Mobilités.net : Villes, transports, technologies face aux nouvelles mobilités*, D. Kaplan et H. Lafont (Eds.), pp. 324-320, Collection « Questions numériques », 2004.

[Uster 98] Uster G. *Information multimodale*. Document interne. INRETS. 1998.

[Van Harmelen et al., 01] Van Harmelen F., Patel-Schneider P. and Horrocks I. *A Model-Theoretic Semantics for DAML+OIL* (March 2001) Revision 4.1. 2001. Available from : http://www.daml.org/2001/03/model-theoretic-semantics.html

[Virvou et Du Boulay 99] Virvou M. and Du Boulay B. Human plausible reasoning for intelligent help. *User Modeling and User-Adapted Interaction*, Vol. 9 No. 4, pp. 321-375, 1999.

[Virvou et Kabassi 02] Virvou M. and Kabassi K. IFM: An Intelligent Graphical User Interface Offering Advice. In *2nd Hellenic Conf. on AI, SETN-2002*, Thessaloniki, Greece, pp. 155-164, April 2002.

[Waern et al., 98] Waern A., Averman C., Tierney M., Rudström A. and Laaksolahti J. *ConCall: an information service for researchers based on EdInfo*. Research report T98:04, Swedish Institute of Computer Science, October 1998.

Chapitre 2
Architectures et Méthodes pour le développement de système d'information personnalisé

Sommaire

Introduction

Bien que de nombreux travaux aient été effectués dans le domaine de la personnalisation de l'interaction homme-machine (Cf. chapitre 1), il n'existe pas ou très peu de méthodes dédiées à l'analyse et à la conception de système d'information personnalisé. Notre objectif est d'obtenir une méthode adaptée à ce sujet. La méthode devrait aussi bien convenir pour la mise en place d'un nouveau système d'information personnalisé que pour la personnalisation d'un système d'information existante. Elle doit aussi permettre la prise en compte de différents types de personnalisation en favorisant la construction de système de personnalisation évolutif et distribué.

Ce chapitre a pour objectif de nous orienter vers le choix d'une telle méthode ou, le cas échéant, de donner les principes de base pour la construction d'une méthode, pour le développement de système d'information personnalisé, basée sur des méthodes existantes.

Les méthodes existantes de génie logiciel, sont généralement destinées à un domaine particulier ou à une technologie particulière. Par exemple, on pourra citer des méthodes spécialisées issues de notre laboratoire telles MAMOSACO (Méthode Adaptable de MOdélisation de Systèmes Administratifs COmplexes) [Adam 00] qui vise à l'analyse et la modélisation des systèmes administratifs complexes, TOOD (Task Oriented Object Design) [Mahfoudi et al., 95][Tabary 01] destinées à l'analyse et la conception des interfaces homme-machine centrée sur une modélisation des tâches humaines et ADESIAD (Approche de DÉveloppement d'un SIAD) [Lepreux 05] pour le développement des systèmes interactifs d'aide à la décision. Pour les autres méthodes, on citera, OSSAD (Office Support System Analysis and Design) [Dumas et Charbonnel 90] pour l'analyse et la spécification de systèmes d'information centrée sur l'organisation du travail et CISAD (Cooperative Information System Analysis and Design) [Nurcan 96] pour l'analyse et la conception d'application coopératives de type Workflow.

Il est donc important de disposer de critères afin de sélectionner la méthode adéquate qui puisse faciliter l'analyse, la conception et la modélisation de système d'information personnalisé.

La première partie de ce chapitre présente les architectures de quatre systèmes de personnalisation qui sont à base d'agents logiciels. Ces systèmes sont InfoSleuth, Gulliver's Genie, ePerson et MAPIS. Ce sont les systèmes qui ont certaines caractéristiques chères à nos travaux comme la distribution, l'évolution ou la multi-application.

La deuxième partie de ce chapitre présente un cadre de comparaison composé de 30 critères destinés à faciliter le choix ou la construction d'une méthode pour l'analyse, la conception et la modélisation de système d'information personnalisé.

La troisième partie est consacrée à l'étude de quatre méthodes représentatives des méthodes susceptibles d'intervenir dans le développement de système d'information personnalisé. Ces méthodes sont : MERISE, 2TUP, WAE et AODPU. Une synthèse de ces méthodes est proposée en regard de la méthode adéquate qui devrait faciliter le développement de système d'information personnalisé.

2.1. Architectures à base d'agents pour le développement de système d'information personnalisé

Cette partie présente les architectures des systèmes de personnalisation *InfoSleuth*, *Gulliver's Genie, ePerson* et *MAPIS*. Les systèmes InfoSleuth et Gulliver's Genie et ePerson ont été choisis car ils sont à base d'agents logiciels et remplissent la majorité des critères définis lors de l'étude des systèmes de personnalisation (voir chapitre 1). L'architecture de MAPIS est naturellement rappelée ici puisque nos travaux de recherche se situent dans la lignée des travaux de MAPIS [Anli et al., 04].

2.1.1. L'architecture InfoSleuth

Le projet InfoSleuth [Bayardo et al., 97 ; Nodine et al., 03] est lancé par les laboratoires de recherche MCC (Microelectronics and Computer technology Corporation) au Texas. InfoSleuth est un système dédié à la recherche d'information dans les environnements dynamiques (en constante évolution), hétérogènes et distribués. L'architecture d'InfoSleuth (voir Figure 2.1) comprend différents modèles d'agents (*User*, *Ontology*, *Broker*, *Ressource*, *Data Analysis*, *Task Execution*, *Query* et *Monitor*) qui communiquent et coopèrent entre eux pour la personnalisation.

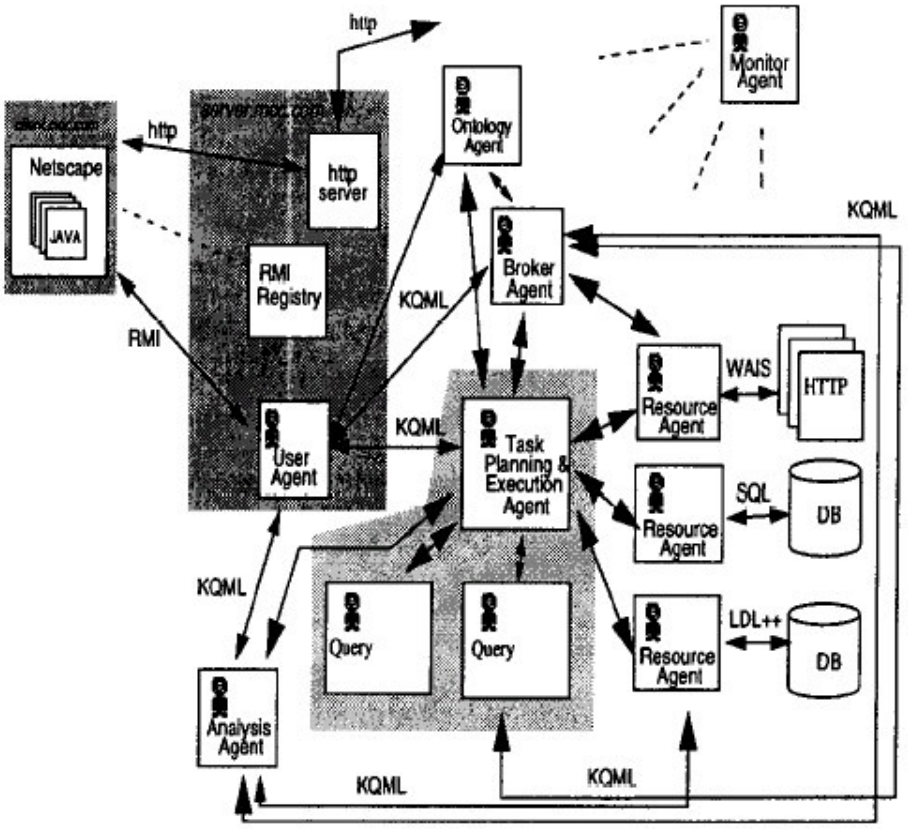


Figure 2.1. L'architecture InfoSleuth [Bayardo et al., 97]

User Agent : constitue l'interface de communication entre l'utilisateur et le système. Cet agent utilise les connaissances de l'utilisateur en se basant sur une ontologie pour assister l'utilisateur dans la formulation de ses requêtes et pour la présentation des résultats.

Ontology Agent : fournit une connaissance globale des ontologies et se charge de répondre aux requêtes relatives aux ontologies.

Broker Agent : reçoit et stocke les informations sur les capacités (compétences) des agents. Cet agent sert comme une sorte d'annuaire permettant aux différents agents de retrouver les agents adéquats pouvant répondre à leurs requêtes.

Ressource Agent : a pour objectif de rendre disponible une information contenue dans une source d'information (une base de données, par exemple) pour qu'elle soit retrouvée ou pour qu'elle soit mise à jour. Les agents *Ressource* effectuent une traduction des messages (exprimés en KQML [Finin et al., 97]) des autres agents en des requêtes (SQL ou LDL++ [Zaniolo 91], par exemple) compréhensibles au niveau des sources d'information.

Data Analysis Agent : ce sont des agents *Ressource* qui sont spécialisés dans la fouille automatique de données.

Task Execution Agent : coordonne l'exécution des différentes tâches pour la recherche d'information. Cela inclut la décomposition des tâches en sous tâches pour les distribuer aux agents adéquats pour la réalisation de l'objectif global. Cet agent est conçu d'une manière à être flexible dans un environnement dynamique et évolutif. Chaque fois qu'une requête provient de l'agent *User* l'agent *Task Execution* crée un agent *Query* pour s'occuper de la requête.

Monitor Agent : capte les messages échangés entre les différents agents et l'état d'exécution des agents pour les afficher au travers d'une interface graphique.

2.1.2. L'architecture Gulliver's Genie

Gulliver's Genie [O'Hare et O'Grady 03 ; O'Grady et al., 05] est un système développé à l'UCD (University College Dublin) qui permet d'assister les utilisateurs dans la découverte des endroits touristiques ou culturels. Gulliver's Genie est construit à partir de différents modèles d'agents et adopte un mode de traitement client/serveur. Le client est un PDA étendu d'un GPS[22] et d'un GPRS[23]. Le serveur est une station de travail (PC) accessible sur internet (voir Figure 2.2).

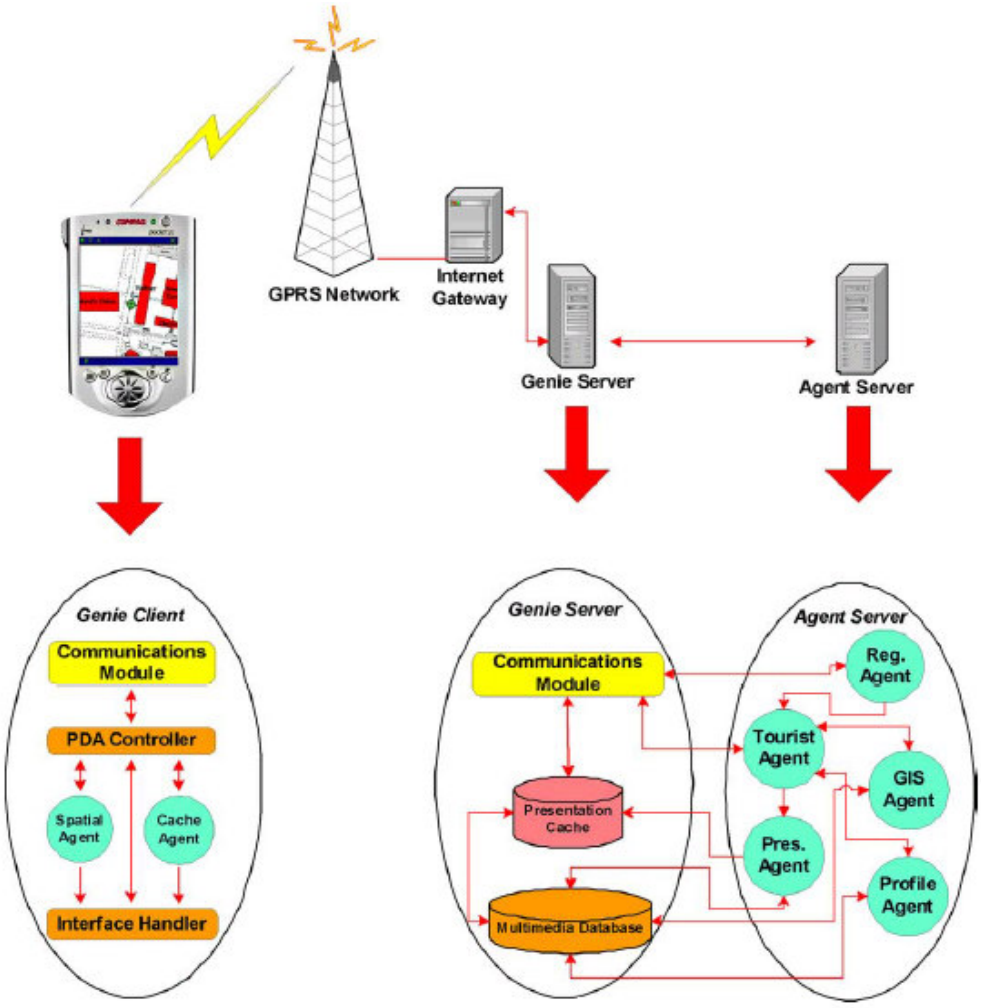


Figure 2.2. L'architecture Gulliver's Genie [O'Grady et al., 05]

[22] Global Positioning System
[23] General Packet Radio Service

Le client est composé de deux agents (*Spatial Agent* et *Cache Agent*) et de trois composants (*PDA Controller, Interface Handler* et *Communication Module*)

Spatial Agent : surveille continuellement les sorties du GPS pour extraire la position et la direction de l'utilisateur. Après avoir vérifié la consistance de ces données, il les envoie aux composants adéquats.

Cache Agent : collabore avec les autres agents (incluant les agents se trouvant sur le serveur) pour s'assurer que les informations mises en cache au niveau du PDA sont pertinentes par rapport à la localisation de l'utilisateur et reflètent bien les préférences de ce dernier. C'est cet agent qui décide s'il doit informer l'utilisateur de l'information de proximité.

PDA Controller : ce composant se charge de l'initialisation et de la terminaison des sessions *Guilliver's Genie*. Ce composant est généralement utilisé par les agents *Spatial Agent* et *Cache Agent*.

Interface Handler : s'occupe de l'interface homme-machine avec laquelle l'utilisateur peut interagir au travers du stylet ou des boutons de navigation du PDA.

Communications Module : est responsable de l'envoi des messages au serveur et de la récupération des réponses y provenant.

La partie serveur est décomposée en deux parties : *Génie server* (comprenant *Communications Module*, *Presentation Cache* et *Database*) et *Agent server* (comprenant *Registration Agent*, *Profile Agent*, *GIS Agent*, *Tourist Agent* et *Presentation Agent*).

Communication Module : transmet les requêtes provenant des clients *Guilliver's Genie* à l'agent ou au composant approprié et retourne les réponses au client correspondant.

Presentation Cache : la base de présentations est contrôlée par l'agent *Presentation Agent*. Cette base contient des présentations pré construites de toutes les attractions (lieux touristiques, culturels, etc.) qui sont dans le voisinage immédiat où se trouve l'utilisateur. Elle est mise à jour en fonction des déplacements de l'utilisateur et de son profil.

Database : il s'agit d'une base de données contenant différentes sortes de données comme des données géospatiales permettant de relier une information à une localité ou des données graphiques permettant d'avoir les plans et les cartes.

Registration Agent : après avoir authentifié l'utilisateur, l'agent *Registration Agent* crée un agent (*Tourist Agent*) destiné à s'occuper de l'utilisateur. A la déconnection de l'utilisateur, toutes les ressources qui lui étaient allouées sont reprises.

Profile Agent : est responsable de la déduction des interactions de l'utilisateur avec le système et de la mise à jour du profil de l'utilisateur.

GIS Agent : l'agent *GIS* (Geographic Information System) s'occupe des demandes de mise à jour de la cache du client. Il interroge la base de données et, en se basant sur la localisation de l'utilisateur, construit un modèle de l'environnement de l'utilisateur qui inclut le détail des attractivités se situant aux alentours. Il renvoie ces données à l'agent *Cache Agent* pour un affinage de ces données. La coordination de l'agent *GIS Agent* et *Cache Agent* est effectuée par l'agent *Tourist Agent*.

Tourist Agent : un agent *Tourist Agent* est affecté à chaque utilisateur. Cet agent se place comme une interface entre le client et le serveur. En se basant sur les informations provenant des agents se trouvant sur le client, il a une connaissance en temps réel des activités de l'utilisateur. Il coopère avec l'agent *GIS Agent* et l'agent *Profile Agent* pour s'assurer que les contenues des caches au niveau du client sont toujours appropriées par rapport à la position de l'utilisateur.

Presentation Agent : construit les interfaces pour les présentations de toutes les attractivités se trouvant sur le voisinage de la position de l'utilisateur et les stocke dans la cache du serveur, en se basant sur le profil de l'utilisateur et sur le modèle de son environnement. Les requêtes de l'utilisateur sont anticipées ainsi pour accélérer la construction de l'interface à envoyer au client. Cette cache est continuellement mise à jour pour prendre en compte le changement du modèle de l'environnement de l'utilisateur et de son profil.

2.1.3. L'architecture ePerson

Le projet ePerson [Dickinson et al., 03] a été lancé par les laboratoires de recherche de Hewlett Packard afin de faciliter la conception de systèmes d'information personnalisés. L'objectif est de fournir une plate-forme qui soit ouverte, distribuée et extensible ; l'utilisateur doit pouvoir avoir accès à son profil et le manipuler, ce profil devant être accessible par des clients à connectivités limitées ; il s'agit enfin de permettre le stockage de parties des profils des utilisateurs dans des bases de données externes tout en préservant la confidentialité des données des utilisateurs. La plate-forme ePerson propose une architecture dont des agents logiciels (*User agents*, *External agents*, *Update agents*) accèdent aux profils des utilisateurs au travers de service web. Elle fournit une interface (*User interface*) permettant la manipulation du profil par l'utilisateur, comprend un moteur de gestion des accès (*Policy engine*) et contient une base de données de profils (*Pofile data*) et un composant (*Gateway*) pour la gestion des références aux sources de données externes.

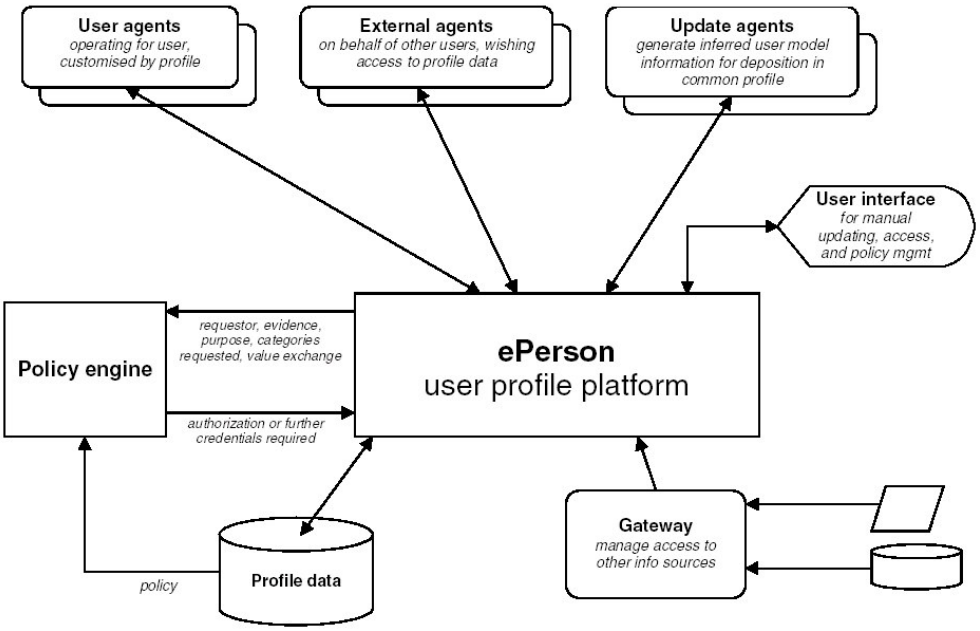


Figure 2.3. L'architecture ePerson [Dickinson et al., 03]

User agents : ce sont des agents assistants qui aident l'utilisateur dans la réalisation de sa tâche. Ils adaptent l'interaction homme-machine en se basant sur le profil de l'utilisateur. Un agent *User agent* représente un utilisateur unique et a accès à l'intégralité du profil de l'utilisateur qu'il représente.

External agents : ce sont des agents *User agents*. ePerson les appelle *External agents* pour préciser que ces agents veulent accéder aussi au profil d'un autre utilisateur que celui qu'ils représentent. Cela pourrait être le cas, par exemple, pour effectuer un filtrage collaboratif en analysant les comportements des autres utilisateurs. Ces agents ont un accès limité aux profils des autres utilisateurs conformément à une politique d'accès définie par l'utilisateur lui-même ou par un administrateur.

Update agents : ce sont des agents spécialisés pour la mise à jour des profils utilisateur. Ils génèrent un modèle de l'utilisateur en se basant sur son comportement et le déposent dans la base des profils utilisateur. Ces agents ont un accès au profil limité par rapport aux autorisations définies.

User interface : permet la visualisation et la modification manuelle d'un profil d'un utilisateur. Chaque utilisateur dispose d'un droit de manipulation de son profil. Il peut, par exemple, changer les informations le concernant ou mettre des droits d'accès pour que des utilisateurs (ou des agents *external agents*) puissent accéder ou non à certaines parties de son profil.

Profile data : est la base de données contenant les profils utilisateur.

Gateway : c'est un module qui gère le référencement des données utilisateur qui ne sont pas stockées au niveau de la base de données *Profile data*. Les profils utilisateurs peuvent être stockés dans des bases de données externes et leurs accès via ePerson est transparent grâce à ce module.

Policy engine : intègre la politique des contrôles d'accès aux données contenues dans le profil utilisateur. Le profil est organisé d'une manière hiérarchique et l'utilisateur peut définir les droits d'accès sur les différents points de la hiérarchie.

2.1.4. L'architecture MAPIS

Développé au sein de notre laboratoire durant la thèse de [Petit-Rozé 03], MAPIS (Multi-Agents Personalized Information System) est un système qui se base sur une organisation multi-agents pour la personnalisation de l'information. MAPIS est composé de quatre modèles d'agents (assistance, coordination, recherche de données et gestion de profils) dont chacun joue un rôle spécifique dans le système (voir Figure 2.4).

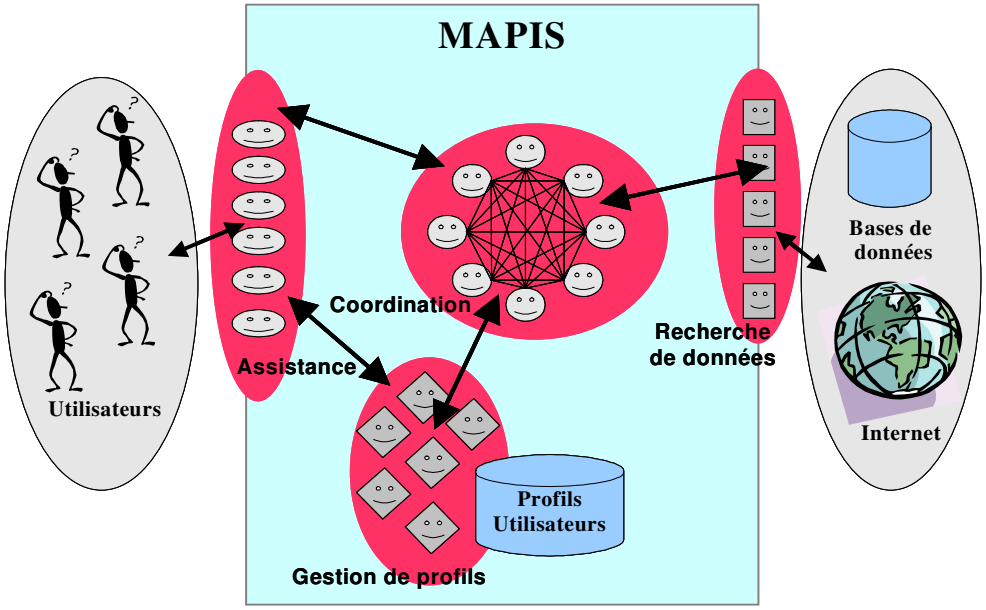


Figure 2.4. L'architecture MAPIS [Petit-Rozé 03]

L'assistance : correspond aux agents en interaction avec l'utilisateur dans le but d'identifier les requêtes de l'utilisateur et/ou de l'aider à exprimer ses demandes. Les agents du modèle *assistance* agissent suite à une demande d'un utilisateur ou lorsqu'ils reçoivent un message d'un autre agent. Dans le cas d'une demande d'un utilisateur les agents du modèle *assistance* peuvent interagir avec les agents du modèle de *gestion de profils* (demande d'inscription d'un utilisateur, par exemple) ou avec les agents du *modèle de coordination* (demande de recherche d'itinéraire, par exemple). Dans le cas d'une réception de message d'un autre agent, les agents du modèle *assistance* ont pour objectif la présentation du message à l'utilisateur.

La coordination : correspond aux agents qui ont pour rôle la supervision des traitements des requêtes. Les agents du modèle *coordination* coordonnent et planifient les activités nécessaires pour fournir le plus rapidement possible une réponse personnalisée à l'utilisateur. Ces agents collaborent entre eux pour essayer de réduire les temps de réponse à l'utilisateur. Ils interagissent avec les autres modèles d'agents pour la mise à jour de leur base de connaissance concernant les informations auxquelles a accès le système (interaction avec les agents du modèle de *recherche de données*) et concernant les préférences des utilisateurs (interaction avec les agents du modèle de *gestion de profils*).

La recherche de données : ce sont les agents qui s'occupent de la récupération des informations qui peuvent être centralisées ou distribuées sur plusieurs bases données hétérogènes. Ces agents ont pour

rôles de répondre aux requêtes de recherche d'information des agents du modèle de *coordination* et de prévenir ces derniers de toute modification de données pertinentes. Les agents du modèle de *recherche de données* se servent de leur mobilité pour la recherche des informations.

La gestion des profils : est assurée par des agents dont le rôle principal est de maintenir à jour une base de données contenant les requêtes des utilisateurs et leurs préférences. La mise à jour des préférences des utilisateurs est effectuée au travers de plusieurs méthodes de personnalisation (stéréotypes, apprentissage par renforcement, etc.). Ces agents s'activent soit pour répondre à un agent du modèle de *coordination* (demande de profil, par exemple), soit lors d'une requête d'un agent du modèle *d'assistance* (demande de création d'un compte utilisateur, par exemple), soit de leur propre initiative (rechercher de nouvelles connaissances sur l'utilisateur, par exemple).

2.1.5. Synthèse

Cette section a étudié les architectures à base d'agents logiciels des systèmes de personnalisation InfoSleuth, Gulliver's Genie, ePerson et MAPIS. Même si ces architectures semblent différentes les unes par rapport aux autres, elles répondent bien aux objectifs définis pour chacun de ces systèmes de personnalisation. L'architecture Gulliver's Genie est destinée à l'assistance des utilisateurs dans leur découverte des lieux touristiques et culturels en prenant en compte l'environnement où se trouve l'utilisateur et ses préférences. Les architectures de MAPIS et d'InfoSleuth sont surtout adaptées pour le filtrage par rapport aux préférences de l'utilisateur lors de la recherche d'information sur internet. L'architecture ePerson a pour objectif principal la mise à disposition aux applications externes d'un serveur standardisé permettant l'accès et la mise à jour des profils tout en préservant la confidentialité des données des utilisateurs.

Différents agents sont utilisés dans ces systèmes de personnalisation. Ces agents jouent des rôles bien définis pour répondre aux objectifs des systèmes de personnalisation. Cependant nous remarquons que même si ces agents portent des noms différents pour chaque système de personnalisation, certains de ces agents peuvent appartenir à un même modèle d'agent. Par exemple, les agents *assistance* de MAPIS, les agents *user* de ePerson et d'InfoSleuth et les agents *cache* et *spatial* de Gulliver's Genie peuvent appartenir aux mêmes modèles d'agents *assistants* qui interagissent directement avec l'utilisateur. La Figure 2.5 présente une classification des différents agents des systèmes de personnalisation présentés ci-dessus par rapport à cinq modèles d'agents. Il s'agit du modèle d'*assistance* qui se présente comme une interface entre l'utilisateur et le système, du modèle de *coordination* qui se charge de la coordination des différents agents, du modèle de *profil* qui gère le profil de l'utilisateur, du modèle de *recherche* pour la recherche d'information et du modèle d'*administration* qui permet la gestion et l'observation des exécutions des agents se trouvant dans le système. Les agents qui ne correspondent pas à ces cinq modèles sont classés dans *Autre*. Notons que certains agents jouent plusieurs rôles pouvant les classer dans différents catégories d'agent. Par exemple, *Update Agent* de ePerson est aussi bien un agent *d'assistance* qu'un agent de *gestion de profil*. Dans ce cas, seule la fonction principale a été considérée à savoir la gestion de profil.

Systèmes / Modèles	InfoSleuth	Guilliver's Genie	ePerson	MAPIS
Assistance	User Agent	Spatial Agent Cache Agent	User Agent External Agent	Assistance
Coordination	Task planning execution Agent Broker Agent Query Agent	Registration Agent Tourist Agent		Coordination
Profil		Profile Agent	Update Agent	Profil
Recherche	Ressource Agent Analysis	GIS Agent		Recherche
Administration	Monitor Agent		(assurée par une interface graphique)	
Autre	Ontology Agent	Presentation Agent		

Figure 2.5. Classification des agents des systèmes de personnalisation étudiés

2.2. Cadre de référence pour l'étude des méthodes de génie logiciel pour le développement de système d'information personnalisé

Cette partie résume les critères composant les dimensions utilisés pour l'analyse des méthodes dans un but de développement de systèmes d'information personnalisés. Les développements qui vont suivre sont fortement inspirés de [Pascot et Bernadas 93] et de [Adam et Kolski 99 ; Adam 00]. [Pascot et Bernadas 93] ont proposé un cadre de référence pour la comparaison des méthodes d'analyse et de conception de système d'information. [Adam 00] a adapté ce cadre de référence pour répondre à ses objectifs d'analyse et de conception d'un système interactif dédié aux systèmes administratifs complexes. Les dimensions développées dans cette partie sont une adaptation de ce cadre de référence selon notre objectif de personnalisation de système d'information.

Le cadre de référence proposé est composé de trois dimensions : la *dimension méthodologie*, la *dimension représentation* et la *dimension technologie*.

2.2.1. La dimension méthodologie

Une méthodologie de génie logiciel décrit les différentes phases et leurs enchaînements dans le processus de développement d'une application informatique. Les phases les plus récurrentes dans les méthodologies de génie logiciel sont *l'analyse de l'existant*, *la spécification*, la *conception*, *l'implémentation* et l'*évaluation.* Une phase d'*intégration* est parfois explicitée lorsqu'il s'agit de combiner deux processus de développement relatifs à deux domaines différents. C'est, par exemple, le cas de l'approche DIAMOND (Decentralized Iterative Approach for Multiagent Open Networks Design) [Jamont 05] qui propose une méthode pour le développement de systèmes embarqués combinant le logiciel et le matériel. Ces phases sont généralement adaptées au type d'application à mettre en œuvre : des phases seront plus ou moins développées suivant la nature de l'application. Par exemple [Nanci et Espinasse 01] proposent un tableau précisant le degré d'implication des différentes phases de la méthode MERISE pour la spécification et la conception d'un système d'information. Ainsi suivant le projet, certaines phases peuvent être *ignorées*, *réduites*, *développées* ou effectuées *normalement*.

Plusieurs modèles existent pour l'enchaînement des différentes phases (voir [Kolski 01], par exemple, pour plus de détails sur ces différents modèles particulièrement dans le cas où le logiciel visé est un système interactif). Chacun de ces modèles convient plus ou moins à un domaine et répond à des besoins ou à des principes particuliers. Nous citerons, par exemple, le modèle en X [Hodgson 91][Coulange 96] qui repose sur le principe de réutilisation et qui est surtout utilisé dans un cadre de développement orienté objet ou le modèle en U [Millot et Roussillon 91, Abed 01, Lepreux et al., 03] qui est adapté pour la prise en compte des facteurs humains dans le développement des interfaces homme-machine. Les modèles de génie logiciel les plus utilisés sont le modèle en cascade (Waterfall), le modèle incrémental et le processus unifié.

Le modèle en cascade : le modèle en cascade a été introduit en 1970 par [Royce 70] puis étendu par [Boehm 81]. Le modèle en cascade définit une réalisation séquentielle des phases dans le processus de développement logiciel avec des retours possibles à la phase précédente pour prendre en compte des lacunes identifiées. Ce modèle est très utilisé par les entreprises et est à la base de plusieurs autres modèles comme le modèle en V [McDermid et Ripkin 84] ou le modèle en spirales [Boehm et al., 84].

Le modèle incrémental : le modèle incrémental appelé aussi modèle évolutif[24] consiste à développer un logiciel en procédant par incrément (ou lot). Ce modèle applique itérativement le modèle en cascade mais sans trop détailler les phases pour répondre à des besoins de certains logiciels tellement mouvants qu'il est illusoire de fonder le développement sur une spécification figée. Il fournit pour

[24] Des auteurs comme [Toffolon 99] différencient le modèle incrémental du modèle évolutif en précisant que dans le modèle incrémental toutes les versions du logiciel doivent être planifiées avant le démarrage des cycles.

chaque lot une version du logiciel qui sera immédiatement mise en exploitation dans un environnement opérationnel [ESA 91].

Le processus unifié : un processus unifié appelé aussi processus UP (Unified Process) est un processus de développement logiciel construit sur UML (Unified Modelling Language)[Booch et al., 00]. Il est itératif, centré sur l'architecture, conduit par les cas d'utilisation et piloté par les risques [Jacobson et al., 00]. UP est un processus générique de développement logiciel c'est-à-dire qu'il est nécessaire de l'adapter au contexte du projet, de l'équipe, du domaine et/ou de l'organisation. Il existe donc un certain nombre de méthodes issues de UP comme RUP (Rational Unified Process) [Kruchten 00], AUP (Agile Unified Process)[Alhir 05], EUP (Enterprise Unified Process) [Ambler et al., 05] ou ICONIX Unified Process [Rosenberg et Scott 99].

Dans le cas de développement de système d'information personnalisé, il est nécessaire d'inclure les utilisateurs finaux (ou panel d'utilisateurs représentant l'ensemble des utilisateurs susceptibles d'utiliser le système) dans certaines de ces phases (notamment dans les phases d'analyse et dans les phases d'évaluation).

La dimension méthodologie est donc associée à quatre critères dont chacun est composé de sous critères :

Phases concernées : analyse, spécification, conception, implémentation, intégration et/ou évaluation

Modèles de développement : en cascade, incrémental ou selon le processus unifié

Degrés d'implication de l'utilisateur : essentiel, beaucoup, moyen, peu ou pas d'implication

Moment d'implication de l'utilisateur : début, milieu et/ou fin de cycle de développement

2.2.2. La dimension représentation

Les formalismes de représentation d'une méthode permettent de fournir une vue abstraite d'un système. Ces formalismes dépendent fortement de l'objectif visé par la méthode et différent d'une méthode à une autre. A titre d'exemple, nous citerons SADT (Structured Analysis and Design Technique)[IGL 89] qui utilise des datagrammes et des actigrammes pour la modélisation des *fonctionnalités* des systèmes et Tropos [Giorgini et al., 01, Giorgini et al., 02] qui propose un formalisme pour une analyse et une modélisation *orientée but* pour le développement des systèmes à base d'agents logiciels.

Depuis le milieu des années 90, le langage UML (Unified Modelling Language), qui sera décrit par la suite, a connu un fort engouement et est devenu une norme OMG[25] (Object Management Group) en 1997. Dans le cadre d'un développement d'un système d'information personnalisé, nous privilégierons les méthodes utilisant le langage UML. Ceci afin de faciliter la prise en main de la méthode par l'utilisation de notations standardisées et connues par un grand nombre d'utilisateurs (développeurs).

La dimension représentation est donc associée à un critère composé de trois sous critères :

Les formalismes : UML, Extension d'UML, autre

2.2.3. La dimension technologie

Nous recherchons une méthode de développement d'un système d'information personnalisé. L'analyse de l'état de l'art (cf. chapitre 1) nous a conduit à identifier trois technologies principales intervenant pour le développement de tels systèmes d'information personnalisés :

- Approche orientée *agent* pour la gestion de la personnalisation. Cette approche permettrait aussi de répondre aux critères d'évolutivité et de distributivité du système de

[25] http://www.omg.org/

personnalisation. Les approches de programmation structurée, orientée objet, à base de composants et qui reposent sur des bases de données sont aussi à prendre en compte. Par exemple, il serait nécessaire de concevoir la base de données des utilisateurs dans le cas d'un premier développement d'un service personnalisé.

- Mode de traitement *client-serveur* pour répondre au critère multi-application (cf. chapitre 1, §1.5.2) du système de personnalisation. Les autres modes de traitement (batch, synchrone et asynchrone) sont aussi à considérer. Par exemple, il pourrait être nécessaire de disposer d'un traitement batch permettant de regrouper (en *offline,* pour répondre à des besoins de performances du système) les utilisateurs suivant leur degré de similarité.
- Utilisation d'*XML* (eXtensible Markup Language) [Harold et Means 01] pour le rendu (en terme d'interface homme-machine). XML se présente comme un langage prédisposé pour la réalisation d'interface adaptative [Habieb-Mammar et al., 02]. Il facilite aussi le développement d'interface multi-cible [Puerta et Eisentein 06][Ali et Abrams 01] et multi-modale [Rouillard 02, Rouillard 04]. La personnalisation pouvant intégrer ces aspects d'interaction homme-machine, il nous paraît important que la méthode de développement des systèmes d'information personnalisés supporte XML.

  XML conviendrait aussi pour l'échange de messages entre les applications externes et le système de personnalisation. D'ailleurs, XML s'impose actuellement comme un standard pour l'échange de données entre applications [Paquel et Bezaut 02]. Notre priorité étant d'abord de privilégier les méthodes basées sur UML, cela ne limite pas la prise en compte de la technologie XML puisque UML permet aussi la modélisation des applications basées sur XML (Voir [Carlson 01], par exemple).

La dimension technologie est donc associée à trois critères :

Programmation : structurée, orientée base de donnée, à base de composants, orientée objet et/ou orientée agent

Mode de traitement : batch, synchrone, asynchrone et/ou client-serveur

XML : (considéré ou non)

2.2.4. Conclusion

Cette section a présenté un cadre de référence adapté pour l'étude des méthodes de génie logiciel selon un objectif d'analyse, de conception et de modélisation de systèmes d'information personnalisés. Le cadre de référence proposé est composé de trois dimensions : la *dimension méthodologie*, la *dimension représentation* et la *dimension technologie.* Chaque dimension comprend des critères définis suivant des caractéristiques généralement retrouvées dans le développement de systèmes d'information personnalisés.

2.3. Etude de méthodes pour le développement de système d'information personnalisé

Dans le but de développement de système d'information personnalisé, quatre méthodes ont été étudiées. Ces quatre méthodes sont représentatives des domaines d'application susceptibles d'intervenir dans l'analyse et la conception de système d'information personnalisé. Ces domaines concernent particulièrement l'analyse et la conception de système d'information, le développement orienté objet, le développement des applications web et, l'analyse et la conception des systèmes à base d'agents logiciels. Ces méthodes sont :

- MERISE (Méthode d'Etude et de Réalisation Informatique des Systèmes d'Entreprise) [Tardieu et al., 83, Tardieu et al., 00][Nanci et Espinasse 01] : la plus représentative des méthodes systémiques d'analyse et de conception de système d'information.

- 2TUP (Two Track Unified Process) [Roques et Vallée 00] représentative des méthodes d'analyse et de conception orientées objet. Cette méthode a été choisie parmi d'autres bien connues comme RUP (Rational Unified Process) [Kruchten 00] ou AUP (Agile Unified Process)[Alhir 05] par son processus de développement qui distingue les aspects fonctionnels des aspects techniques. En effet, ce principe rejoint celui que nous proposerons d'utiliser dans le développement de système d'information personnalisé séparant le système d'information du système de personnalisation.
- WAE (Web Application Extension) [Conallen 00] : représentative des méthodes dédiées à l'analyse et la conception d'applications web. D'autres méthodes comme WSDM (Web Site Design Method) [De Troyer et Leune 98], WebML (Web Modeling Language) [Ceri et al., 00] ou UWE (UML-based Web Engineering) [Koch et al., 01] auraient pu convenir, le but étant d'étudier une méthode de développement d'application web puisque la plupart de systèmes d'information personnalisés sont des applications web.
- AODPU (Agent-Oriented Design Process with UML) [Conssentino et al., 00] : représentative des méthodes d'analyse et de conception des systèmes logiciels à base d'agents logiciels. Cette méthode a été considérée parmi d'autres comme MaSE (Multiagent System Engineering) [DeLoach et al., 01], PASSI (Process for Agent Societies Specification and Implementation) [Burrafato and Cossentino 02] ou GAIA [Wooldridge et al., 00] car il est basé sur AUML (Agent Unified Modeling Language) [Odell et al., 00], une extension d'UML largement adoptée par la communauté travaillant sur les systèmes à base d'agents logiciels. Le modèle de développement de AODPU favorisant un développement incrémental a aussi motivé ce choix.

2.3.1. La méthode MERISE

MERISE (Méthode d'Etude et de Réalisation Informatique des Systèmes d'Entreprise) [Tardieu et al., 83][Tardieu et al., 00][Nanci et Espinasse 01] est une des méthodes systémiques d'analyse et de conception de système d'information la plus connue. Proposée en 1978 par un groupement de sept SSII et l'administration française, des extensions de MERISE ont été proposées, au fur et à mesure, pour prendre en compte des technologies ou des principes nouveaux [Morejon 94][Gabay 01].

MERISE préconise un développement logiciel basé sur le modèle en cascade [Boehm 81]. L'utilisateur intervient pendant le développement du système mais sa participation se situe uniquement au début lors de la phase d'analyse du système d'information.

Au niveau technologique, MERISE est surtout adaptée pour l'analyse et la conception des bases de données qui peuvent supporter un mode de traitement batch, synchrone et client-serveur. Elle peut aussi aboutir à une programmation structurée. A notre connaissance, il n'existe pas d'adaptation ou d'extension de MERISE permettant de prendre en compte le développement d'applications basées sur XML.

MERISE fournit des formalismes pour la modélisation conceptuelle (le modèle conceptuel de données et le modèle conceptuel des traitements), la modélisation logique (le modèle logique des données et le modèle logique des traitements) et la modélisation physique (le modèle physique de données et le modèle opérationnel des traitements).

Méthodologie		
Phases Concernées	Analyse	√
	Spécification	√
	Conception	√
	Implémentation	√
	Intégration	
	Evaluation	√
Modèles de développement	Cascade	√
	Incrémental	
	Processus unifié	
Degré d'implication de l'utilisateur	Essentiel	
	Beaucoup	
	Moyen	√
	Peu	
	Pas	
Moment d'implication de l'utilisateur	Début	√
	Milieu	
	Fin	

Technologie		
Programmation	Structurée	√
	Base de données	√
	Objet	
	Composant	
	Agent	
Mode de traitement	Batch	√
	Synchrone	√
	Asynchrone	
	Client-Serveur	√
XML		

Représentation		
Formalisme	UML	
	Extension d'UML	
	Autre	√

Tableau 2.1. Analyse de la méthode MERISE selon les trois dimensions

2.3.2. Le processus 2TUP

2TUP (2 Track Unified Process) [Roques et Vallée 00], appelé aussi modèle de développement en Y, est un processus de développement basé sur le processus UP. Comme tout processus UP, 2TUP se base sur les formalismes de représentation UML. UML (Unified Modeling Language) [Booch et al., 00] se définit comme un langage de modélisation destiné à décrire des besoins, spécifier et documenter des systèmes. Ce langage ayant pour visée le développement logiciel unifie les principales notations et concepts orientés objet. Il fournit neuf diagrammes pour permettre la modélisation d'un système depuis la spécification jusqu'à l'implémentation. UML représente un système logiciel selon deux aspects complémentaires : la modélisation statique ou structurelle et la modélisation dynamique ou comportementale. La modélisation statique s'appuie sur le diagramme de cas d'utilisation, le diagramme de classes, le diagramme d'objets, le diagramme de composants, et le diagramme de déploiement. La modélisation dynamique s'appuie sur le diagramme d'états, le diagramme d'activité, le diagramme de collaboration et le diagramme de séquence.

2TUP apporte une réponse aux contraintes d'évolution continuelle des systèmes d'information en décomposant l'analyse du système suivant un axe fonctionnel et suivant un axe technique. Le processus de développement 2TUP comprend trois branches (voir Figure 2.6).

La branche de gauche (branche fonctionnelle) contient deux phases :

Capture des besoins fonctionnels : 2TUP commence par une description exhaustive des besoins fonctionnels et opérationnels en modélisant les comportements attendus du système. Il s'agit d'une formalisation, par des cas d'utilisation, des exigences fonctionnelles du système sans imposer le mode de réalisation de ces fonctionnalités. La description des cas d'utilisation peut être complétée par des diagrammes dynamiques.

Analyse : cette phase consiste à étudier la spécification fonctionnelle au travers d'une analyse orientée objet pour trouver les classes fondamentales du système et leurs relations. Ces classes sont organisées suivant un regroupement logique sous forme de catégories (packages). Les classes sont complétées et détaillées itérativement au travers d'une analyse de la dynamique du système modélisée par des diagrammes de séquences, de collaboration et d'états.

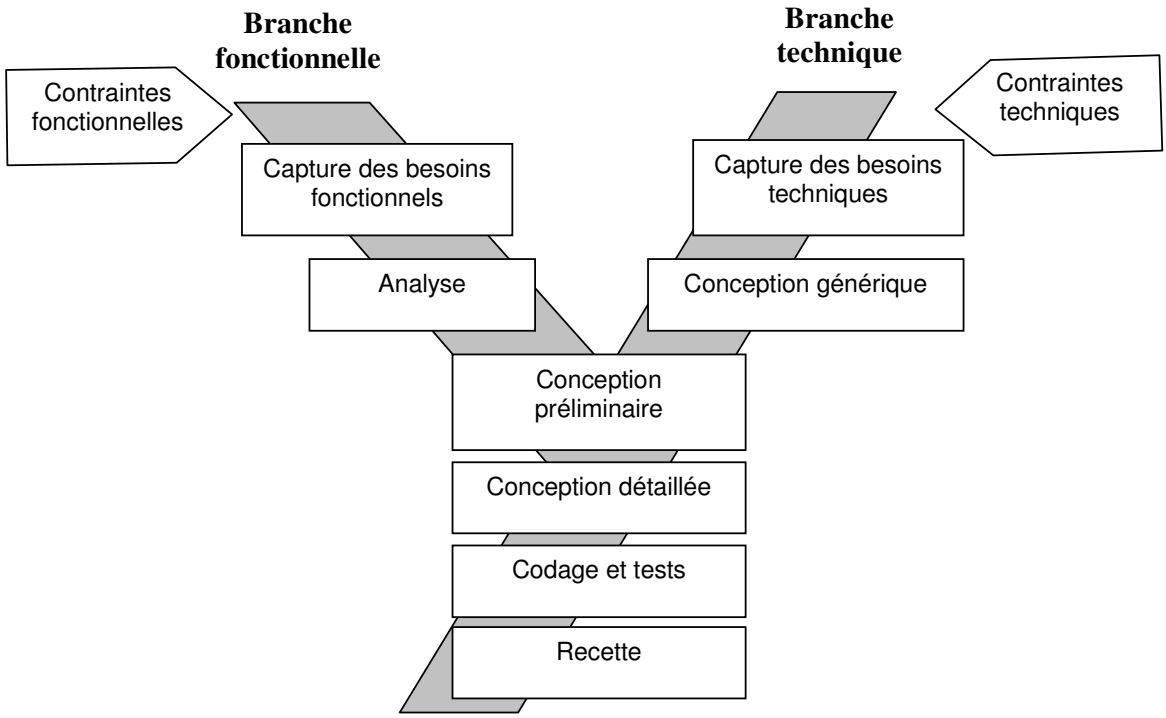


Figure 2.6. Le processus de développement en Y [Roques et Vallée 00]

La branche de droite (branche technique) intègre deux phases :

Capture des besoins techniques : cette phase recense toutes les contraintes qui ne traitent ni de la description du métier des utilisateurs, ni de la description de l'application. Il s'agit de la spécification des outils (logiciels), de la structure des matériels à exploiter et des contraintes d'intégration avec l'existant. La spécification logicielle et la structure du matériel sont décrites au travers de diagrammes de cas d'utilisation, de packages, de composants et de déploiements.

Conception générique : cette phase définit les composants nécessaires à la construction de l'architecture technique indépendamment des aspects fonctionnels spécifiés dans la branche gauche. Cette conception générique peut aboutir à un développement d'un prototype de manière à valider les principes par le codage et les tests.

La branche du milieu se compose de quatre phases :

Conception préliminaire : la conception préliminaire consiste à appliquer les concepts liés aux fonctionnalités du système et à intégrer les composants techniques au système. Il s'agit d'intégrer les fonctions métiers et applicatives dans l'architecture technique définie dans la phase de conception générique.

Conception détaillée : il s'agit d'étudier la manière de réaliser les composants. L'objectif d'une conception détaillé consiste à produire un modèle « prêt-à-coder ». Beaucoup de représentations UML sont utilisées dans cette phase : le diagramme de classes représente l'organisation des classes, les diagrammes d'interactions (séquence ou collaboration) montrent la dynamique d'échanges entre les classes en mettant en valeur l'utilité des différentes opérations, les diagrammes d'activités servent à détailler des méthodes, les diagrammes d'états représente les mécanismes d'une classe à état et le diagramme de composants modélisent de la configuration logicielle des sous-systèmes.

Codage et tests : cette phase consiste à la production des composants logiciels et à leurs tests au fur et à mesure de leur implémentation.

Recette : c'est la phase de validation des fonctionnalités du système développé.

L'utilisateur est assez bien pris en compte dans le processus de développement du système. Il intervient principalement dans les phases de capture des besoins fonctionnels, d'analyse et de recette.

Le Tableau 2.2 résume l'analyse du processus 2TUP selon les trois dimensions.

Méthodologie		
Phases concernées	Analyse	√
	Spécification	√
	Conception	√
	Implémentation	√
	Intégration	
	Evaluation	√
Modèles de développement	Cascade	
	Incrémental	
	Processus unifié	√
Degré d'implication de l'utilisateur	Essentiel	√
	Beaucoup	
	Moyen	
	Peu	
	Pas	
Moment d'implication de l'utilisateur	Début	√
	Milieu	
	Fin	√

Technologie		
Programmation	Structurée	√
	Base de données	√
	Objet	√
	Composant	√
	Agent	
Mode de traitement	Batch	√
	Synchrone	√
	Asynchrone	√
	Client-Serveur	√
XML		√

Représentation		
Formalisme	UML	√
	Extension d'UML	
	Autre	

Tableau 2.2. Analyse de 2TUP selon les trois dimensions

2.3.3. La méthode WAE

WAE (Web Application Extension) [Conallen 00] est une méthode dédiée pour le développement d'applications web[26]. WAE fournit une extension d'UML pour la modélisation d'application web. Etendre UML consiste à définir de nouveaux *stéréotypes*, *étiquettes* et *contraintes* pour prendre en compte les spécificités d'un domaine d'application.

- Un stéréotype permet d'associer une nouvelle signification à un élément d'un modèle UML. Les stéréotypes sont généralement représentés par une chaîne de caractères entre guillemets (« ») ou par une nouvelle icône.
- Une étiquette permet la définition d'une nouvelle propriété d'un élément du modèle. Elle est représentée par une chaîne de caractères entre chevrons (<>).
- Une contrainte spécifie une règle que le modèle doit vérifier pour qu'il soit sémantiquement valide. Les contraintes sont représentées par des chaînes de caractères entre accolades ({}).

La Figure 2.7 donne un exemple de diagramme de classes WAE pour le parcours d'une page web.

[26] Une application web est un système logiciel client-serveur qui possède, au moins, un navigateur HTML/XML qui communique avec un serveur web via HTTP et un serveur d'application qui gère la logique métier [Conallen 00].

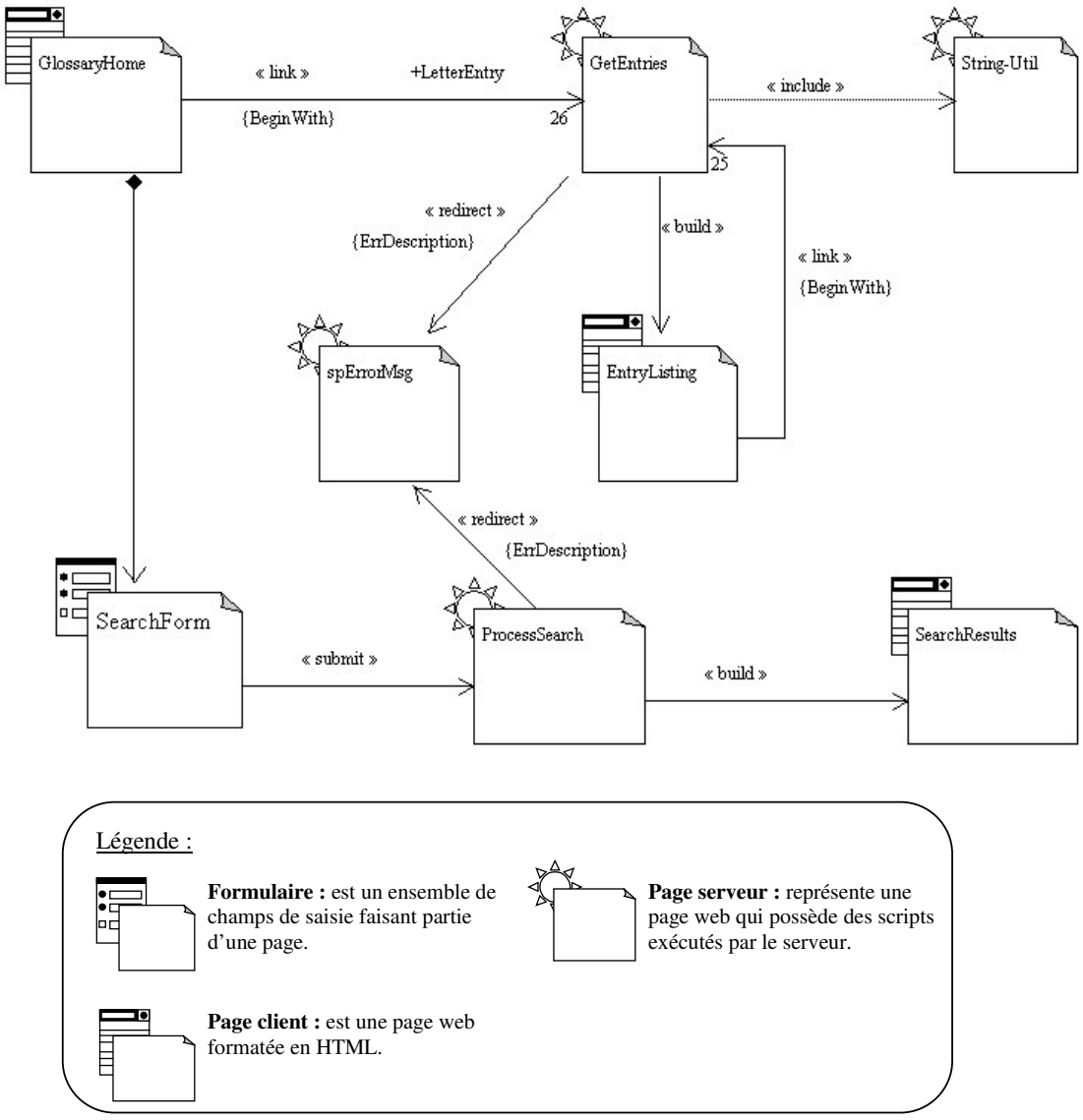


Figure 2.7. Exemple de modélisation avec WAE [Conallen 00]

Le modèle de développement préconisé par WAE est basé sur le processus unifié de Rational (RUP, Rational Unified Process) [Kruchten 00] et le ICONIX Unified Process [Rosenberg et Scott 99]. Le processus WAE est donc piloté par les cas d'utilisation, centré sur l'architecture et itératif. WAE effectue itérativement les phases classiques de développement logiciel suivant un objectif de qualité (voir Figure 2.8). La planification de toutes les itérations est nécessaire avant le commencement des phases de développement.

Besoins : la spécification des besoins décrit le système logiciel à développer. Il s'agit d'une spécification des besoins fonctionnels (par exemple, le système doit pouvoir calculer le total de toutes les commandes effectuées) et des besoins non fonctionnels qui peuvent être classés suivant des catégories (besoins d'utilisabilité, de performance, de disponibilité/fiabilité, de sécurité, de déploiement, etc.). Chaque besoin doit être vérifiable. Par exemple, « l'interface utilisateur doit être intuitive » ne peut pas être testé objectivement et donc n'est pas un énoncé de besoin vérifiable.

Analyse : cette phase consiste en la modélisation de la structure du système en partant des cas d'utilisation et des besoins fonctionnels. Ce sont les mêmes activités que celles définies dans la phase d'analyse de la méthode 2TUP (cf. 2.2.2). Il s'agit de structurer et de représenter les objets du domaine

métier par une série de diagrammes de classes (package et classe) et d'interaction (séquence, collaboration et activités).

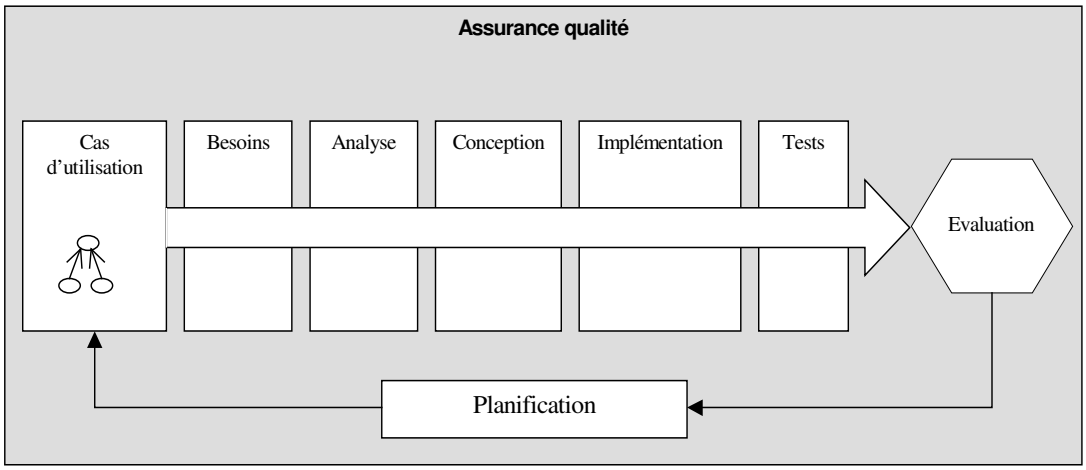


Figure 2.8. Modèle de développement de WAE [Conallen 00]

Conception : la conception vise à préciser le modèle d'analyse de telle sorte qu'il puisse être implémenté avec les composants de l'architecture. Les activités de conception se focalisent sur les diagrammes de classes et d'interaction. De nouvelles classes (pour le soutien à l'implémentation) sont souvent ajoutées. La modélisation des modules et des exécutables est exprimée au travers de diagrammes de composants. WAE fournit une extension d'UML permettant de traduire le modèle de conception en code source pour les applications web.

Implémentation : c'est la phase de traduction du modèle de conception en code source et en composants. L'utilisation des environnements de développement intégrés pour générer le code à partir du modèle de conception est recommandée. Cependant, la génération du code avec les environnements intégrés se limite généralement à des éléments structuraux. C'est à l'utilisateur (développeur) d'implémenter la dynamique de l'application. Il s'agit aussi de répercuter les modifications apportées pendant l'implémentation au modèle de conception pour que celui-ci reflète exactement l'implémentation (rétro-ingéniérie).

Tests : WAE préconise des tests d'évaluation des propriétés du système :

- Des *tests de performance* consistant à évaluer la capacité du système à fonctionner rapidement sous la pression d'une forte charge.
- Des *tests de charge* visant à établir le point de rupture ou la courbe de performance relative du système.
- Des *tests fonctionnels*, qui découlent le plus souvent des cas d'utilisation, permettant de vérifier l'adéquation des fonctions entre la spécification des besoins et l'implémentation.

D'autres tests permettent la validation de certaines phases du processus :

- Des *tests unitaires* portant généralement sur chaque composant développé ou une collaboration restreinte de composants développés.
- Des *tests d'intégration* pour vérifier et valider la compatibilité des différentes interfaces des composants. Ils sont effectués après l'interconnexion de certaines parties du système mais avant l'assemblage de l'ensemble du système.
- Des *tests système*, effectués à l'issue de l'assemblage total du système ; ces tests permettent de vérifier si l'ensemble des besoins sont satisfaits.

- Des *tests d'acceptation* par lesquels les utilisateurs participent pour valider le système. Si le système est accepté, il pourra être déployé.

Parallèlement à ces tests de validation, des *tests de non-regression* sont effectués pour contrôler à nouveau un système qui a potentiellement changé.

WAE permet de prendre en compte l'utilisateur dans le processus de développement du système. Il intervient principalement dans les phases de capture des besoins et des tests d'acceptation.

Les extensions WAE favorise un passage direct du modèle de conception au code des applications web caractéristiques des applications client serveur.

Le Tableau 2.3 résume l'analyse de WAE selon les trois dimensions.

Méthodologie		
Phases concernées	Analyse	√
	Spécification	√
	Conception	√
	Implémentation	√
	Intégration	
	Evaluation	√
Modèles de développement	Cascade	
	Incrémental	
	Processus unifié	√
Degré d'implication de l'utilisateur	Essentiel	√
	Beaucoup	
	Moyen	
	Peu	
	pas	
Moment d'implication de l'utilisateur	Début	√
	Milieu	
	Fin	√

Technologie		
Programmation	Structurée	√
	Base de données	√
	Objet	√
	Composant	√
	Agent	
Mode de traitement	Batch	√
	Synchrone	√
	Asynchrone	√
	Client-Serveur	√
XML		√

Représentation		
Formalisme	UML	
	Extension d'UML	√
	Autre	

Tableau 2.3. Analyse de WAE selon les trois dimensions

2.3.4. La méthode AODPU

La méthode AODPU (Agent-Oriented Design Process with UML) [Conssentino et al., 00] fournit un cadre pour le développement des systèmes à base d'agents logiciels. Elle utilise le formalisme de représentation UML pour la modélisation du système dans ses différentes phases. AODPU commence par un modèle général et l'affine au fur et à mesure pour aboutir à une description précise donnant à une implémentation du système. AODPU utilise le modèle visible en Figure 2.9 pour le développement du système.

Identification des agents : AODPU commence par une description fonctionnelle du système par une série hiérarchique de diagrammes de cas d'utilisation. Les cas d'utilisation correspondent aux différents agents, les relations entre cas d'utilisation correspondent aux communications entre les agents et les acteurs correspondent aux entités externes (environnement, applications externes, utilisateurs, …).

Définition de la structure des agents : dans cette phase, la définition de la structure de chaque agent est réalisée au travers de diagrammes de classes. Les méthodes correspondent aux sous tâches que l'agent est capable d'effectuer. C'est aussi dans cette phase que les types d'interaction (communication directe, diffusion des messages, …) entre les agents sont définis. Ces interactions sont modélisées par les associations entre les classes. Chaque agent joue un rôle pendant ses interactions avec les autres agents.

Description des comportements : pour compléter la structure de chaque agent (représentée sous forme de diagramme de classes), les scénarios relatifs aux cas d'utilisation sont décrits au travers

de diagrammes de séquence et, la coopération entre agents, par les diagrammes de collaboration. Les diagrammes d'activité décrivent les actions des différents agents qui coopèrent pour atteindre un but. Pour la modélisation des messages concurrents dans les diagrammes de séquences, AODPU propose l'utilisation des extensions d'UML notamment l'utilisation de AUML [Odell et al., 99 ; Odell et al., 00].

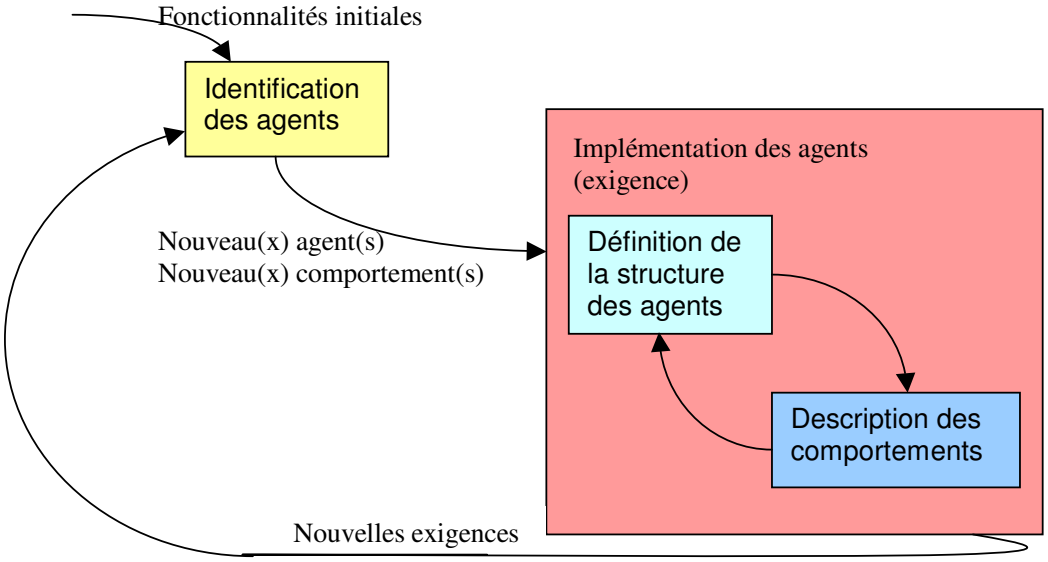


Figure 2.9. Le modèle de développement d'AODPU [Conssentino et al., 00]

AODPU prend moyennement en compte l'utilisateur qui intervient principalement dans la phase d'identification des agents dans le processus de développement du système.

Le Tableau 2.4 résume l'analyse de AODPU selon les trois dimensions.

Méthodologie		
Phases concernées	Analyse	√
	Spécification	√
	Conception	√
	Implémentation	√
	Intégration	
	Evaluation	√
Modèles de développement	Cascade	
	Incrémental	√
	Processus unifié	
Degré d'implication de l'utilisateur	Essentiel	
	Beaucoup	
	Moyen	√
	Peu	
	pas	
Moment d'implication de l'utilisateur	Début	√
	Milieu	
	Fin	

Technologie		
Programmation	Structurée	
	Base de données	
	Objet	
	Composant	
	Agent	√
Mode de traitement	Batch	√
	Synchrone	√
	Asynchrone	√
	Client-Serveur	√
XML		

Représentation		
Formalisme	UML	
	Extension d'UML	√
	Autre	

Tableau 2.4. Analyse de AODPU selon les trois dimensions

2.3.5. Synthèse et discussion

Le Tableau 2.5 présente une synthèse des méthodes MERISE, 2TUP, WAE et AODPU par rapport aux trois dimensions. L'analyse de ce tableau montre que chaque méthode accorde plus ou moins d'importance à des critères par rapport à ses objectifs initiaux (domaine d'application).

Bien que MERISE soit adaptée pour l'analyse et la conception des systèmes d'information reposant sur une base de données, cette méthode convient peu pour l'analyse et la conception de systèmes d'information personnalisés. Ses lacunes par rapport aux critères définis se situent surtout au niveau des dimensions technologie et représentation.

2TUP répond à la majorité des critères. La non prise en compte du développement à base d'agents logiciels constitue sa principale insuffisance.

WAE comme 2TUP répond à la majorité des critères relatifs au développement de système d'information personnalisé. Elle ne prend pas en compte non plus le développement à base d'agents logiciels. Sa principale différence avec 2TUP porte sur la dimension représentation : WAE fournit des formalismes beaucoup plus précis pour la modélisation des applications web en particulier qui peuvent se généraliser aux applications client serveur.

AODPU répond essentiellement au critère de conception à base d'agent logiciel dans la dimension technologie.

Ainsi aucune méthode ne répond entièrement aux différents critères définis pour l'analyse et la conception de système d'information personnalisé. Par contre, des critères comme le moment d'implication de l'utilisateur et le degré d'implication de l'utilisateur dans la dimension méthodologie ou la prise en compte de la technologie client-serveur sont plus ou moins pris en compte par les quatre méthodes. On remarque aussi que la phase d'intégration dans la dimension méthodologie n'est prévue (explicitement) par aucune des quatre méthodes. Cependant, il paraît envisageable de combiner et d'adapter ces méthodes pour définir une méthode spécifique plus adéquate pouvant répondre à notre problématique d'analyse, de conception et de modélisation de système d'information personnalisé.

Le Tableau 2.6 montre à quel niveau (par rapport aux critères) des éléments de ces méthodes pourraient intervenir pour la construction d'une nouvelle méthode d'analyse et de conception de système d'information personnalisé.

Concernant la dimension méthodologie, 2TUP répond aux principales phases de l'analyse des besoins à l'évaluation d'un système d'information. AODPU décrit ces phases suivant une approche orientée agent. Pour le modèle de développement, MERISE se base sur un modèle de développement classique, AODPU favorise un développement incrémental des agents logiciels et 2TUP préconise un modèle de développement unifié. 2TUP assure un degré d'implication essentielle de l'utilisateur qui intervient au début et à la fin des cycles.

Concernant la dimension technologie, 2TUP permet la programmation structurée, orientée objet et à base de composants. 2TUP permet la conception des bases de données objets et relationnelles (voir par exemple, [Soutou 02] pour la conception des bases de données avec UML). AODPU intervient pour la programmation à base d'agents logiciels. Les modes de traitement batch, synchrone et asynchrone sont assurés par 2TUP et WAE intervient pour le mode client-serveur. 2TUP prend en compte la technologie XML.

Bien évidemment, la dimension représentation est essentiellement assurée par 2TUP. WAE intervient pour la modélisation des applications client-serveur. AODPU utilise la modélisation UML et renvoie vers des extensions UML comme AUML pour la prise en compte de certaines particularités liées aux agents logiciels (par exemple, la modélisation de l'envoi de messages à un groupe d'agents ou la modélisation du processus de sélection d'un agent parmi un ensemble d'agents dans les diagrammes de séquence).

Méthodologie					
		MERISE	**2TUP**	**WAE**	**AODPU**
Phases concernées	Analyse	√	√	√	√
	Spécification	√	√	√	√
	Conception	√	√	√	√
	Implémentation	√	√	√	√
	Intégration				
	Evaluation	√	√	√	√
Modèles de développement	Cascade	√			
	Incrémental				√
	Processus unifié		√	√	
Degré d'implication de l'utilisateur	Essentiel		√	√	
	Beaucoup				
	Moyen	√			√
	Peu				
	pas				
Moment d'implication de l'utilisateur	Début	√	√	√	√
	Milieu				
	Fin		√	√	

Technologie					
		MERISE	**2TUP**	**WAE**	**AODPU**
Programmation	Structurée	√	√	√	
	Base de données	√	√	√	
	Objet		√	√	
	Composant		√	√	
	Agent				√
Mode de traitement	Batch	√	√	√	√
	Synchrone	√	√	√	√
	Asynchrone		√	√	√
	Client-Serveur	√	√	√	√
XML			√	√	

Représentation					
		MERISE	**2TUP**	**WAE**	**AODPU**
Formalisme	UML		√		
	Extension d'UML			√	√
	Autre	√			

Tableau 2.5. Synthèse des quatre méthodes par rapport aux trois dimensions

Méthodologie		
Phases concernées	Analyse	2TUP AODPU
	Spécification	
	Conception	
	Implémentation	
	Intégration	
	Evaluation	
Modèles de développement	Cascade	MERISE
	Incrémental	AODPU
	Processus unifié	2TUP
Degré d'implication de l'utilisateur	Essentiel	2TUP
	Beaucoup	
	Moyen	
	Peu	
	pas	
Moment d'implication de l'utilisateur	Début	2TUP AODPU
	Milieu	
	Fin	2TUP WAE

Technologie		
Programmation	Structurée	2TUP
	Base de données	
	Objet	
	Composant	
	Agent	AODPU
Mode de traitement	Batch	2TUP
	Synchrone	
	Asynchrone	
	Client-Serveur	WAE
XML		2TUP

Représentation		
Formalisme	UML	2TUP
	Extension d'UML	WAE AODPU
	Autre	

Tableau 2.6. Contribution des quatre méthodes pour la construction d'une méthode pour le développement de SI personnalisé

Conclusion

Une méthode adéquate pour le développement de système d'information personnalisé devrait répondre à trois dimensions : la dimension méthodologie pour décrire les différentes phases et leurs enchaînements pour le développement de l'application, la dimension technologie pour prendre en compte les techniques utilisées pour la personnalisation comme l'approche à base d'agents logiciels, le mode client-serveur ou l'utilisation de XML et la dimension représentation pour la spécification et la modélisation de l'application.

L'étude des méthodes représentatives des domaines d'application susceptibles d'intervenir dans le développement de systèmes d'information personnalisés montre que bien qu'elles répondent bien aux objectifs (système client-serveur, système à base d'agents, système fondé sur les bases de données, …) auxquels elles sont définies, ces méthodes ne permettent pas de répondre entièrement à notre problématique ; à savoir obtenir une méthode générale permettant aussi bien la mise en place d'un nouveau système d'information que la personnalisation d'un système d'information existante et permettant différents types de personnalisation par la construction de système de personnalisation évolutif et distribué. Cependant, ces méthodes paraissent pertinentes par rapport à certains critères. Par exemple, 2TUP, AODPU et MERISE pour les aspects méthodologies et, 2TUP, AODPU et WAE pour les aspects technologies et représentations.

L'adaptation et l'intégration de ces méthodes paraît donc possible pour la construction d'une nouvelle méthode plus adéquate répondant à notre problématique.

Sur cette base, le chapitre 3 a pour but de proposer une méthode adaptée pour l'analyse, la conception et la modélisation de systèmes d'information personnalisés permettant aussi bien le développement d'un nouveau système d'information que la personnalisation d'un système d'information existant. La méthode doit aussi permettre la prise en compte de différents types de personnalisation au travers du système de personnalisation qui doit être évolutif et distribué.

Bibliographie du chapitre 2

[Abed 01] Abed M. *Méthodes et Modèles formels et semi-formels de conception et évaluation des systèmes homme-machine*. Mémoire de HDR, université de Valenciennes et du Hainaut-Cambrésis, 2001.

[Adam 00] Adam E. *Modèle d'organisation multi-agent pour l'aide au travail coopératif dans les processus d'entreprise : application aux systèmes administratifs complexes*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, septembre 2000.

[Adam et Kolski 99] Adam E. et Kolski C. Etude comparative de méthodes de génie logiciel utiles au développement de systèmes interactifs dans les processus administratifs complexes. *Génie Logiciel*, 49, pp. 40-54, 1999.

[Alhir 05] Alhir S.S. The Agile Unified Process (AUP). *UML & Design World 2005 Conference,* Austin. TX, USA. June 13-16, 2005.

[Ali et Abrams 01] Ali M.F. and Abrams M. Simplifying construction of multi-platform User Interfaces using UIML. Presented at *UIML Europe 2001 Conference*, March 2001.

[Ambler et al., 05] Ambler S.W., Nalbone J. and Vizdos M. *The Enterprise Unified Process : Extending the Rational Unified Process*. Prentice Hall PTR, 2005.

*[Anli et al., 04] Anli A., Petit-Rozé C. and Grislin-Le Strugeon E. User-based Decision Support System. *In International Conference on Advances in Intelligent Systems - Theory and Applications in cooperation with IEEE Computer Society*, AISTA'2004, Luxembourg, novembre 2004.

[Bayardo et al., 97] Bayardo Jr. R. J., Bohrer W., Brice R., Cichocki A., Fowler J., Helal A., Kashyap V., Ksiezyk T., Martin G., Nodine M., Rashid M., Rusinkiewicz M., Shea R., Unnikrishnan C., Unruh A. and Woelk D. InfoSleuth: agent-based semantic integration of information in open and dynamic environments. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of data,* pp.195-206, May 11-15, Tucson, Arizona, United States, 1997.

[Boehm 81] Boehm B.W. *Software Engineering Economics*. Englewood Cliffs : Prentice Hall,1981.

[Boehm et al., 84] Boehm B.W., Gray T.E. and Seewaldt T. Prototyping versus specifying : a multiproject experiment. *IEEE transactions on Software Engineering*, 10(3), pp. 290-302, 1984.

[Booch et al., 00] Booch G., Rumbaugh J. and Jacobson I. *Le guide de l'utilisateur UML*. Paris : Eyrolles, 2000.

[Burrafato and Cossentino 02] Burrafato P. and Cossentino M. Designing a multi-agent solution for a bookstore with the PASSI methodology. *Proceedings of the Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002 at CaiSE'02).* Toronto (Ontario, Canada), May 27-28, 2002.

[Carlson 01] Carlson D. *Modélisation d'applications XML avec UML*. Paris : Eyrolles, 2001.

[Ceri et al., 00] Ceri S., Fraternali P. and Bongio A. Web Modeling Language (WebML) : a modeling language for designing web sites. In *Ninth International World Wide Web Conference WWW9*, Amsterdam, 15-19 May 2000.

[Conallen 00] Conallen J. *Concevoir des applications web avec UML*. Paris : Eyrolles, 2000.

[Conssentino et al., 00] Conssentino M., Chella A. and Faso L. U. Designing agent-based systems with UML. In *International symposium on Robotics and Automation*, ISRA'2000, Monterrey, Mexico, November 2000.

[Coulange 96] Coulange B. *Réutilisation du logiciel*. Paris : MASSON, 1996.

[De Troyer et Leune 98] De Troyer O.M.F. and Leune C.J. WSDM : A User centered Design Method for Web Sites. *Proceedings of the 7th International Conference on World Wide Web*. Brisbane, Australia, pp. 85-94, April 1998.

[DeLoach et al., 01] DeLoach S. A., Wood M. and Sparkman C. Multiagent system engineering. *International Journal of Software Engineering and Knowledge Engineering*, 11(3), 231-258, 2001.

[Dickinson et al., 03] Dickinson I., Reynolds D., Banks D., Cayzer S. and Vora P. User Profiling with privacy : A framework for Adaptive Information Agents, M. Klush et al. (Eds.) : *Intelligent Information Agents*, LNAI 2586, pp 123-151, Berlin : Springer-verlag, 2003.

[Dumas et Charbonnel 90] Dumas P. and Charbonnel G. *La méthode OSSAD, pour maîtriser les technologies de l'information. Tome 1 : principes*. Paris : Les éditions d'organisation, 1990.

[ESA 91] ESA. *Software Engineering Standards*. European Space Agency, Rapport Issue 2, 1991.

[Finin et al., 97] Finin T., Labrou Y. and Mayfield J. KQML as an Agent Communication Language. In J. Bradshaw (Eds.), *Software Agents*, pp. 291-316. Cambridge, USA, MIT Press, 1997.

[Gabay 01] Gabay J. *Merise et UML pour la modélisation des systèmes d'information*. Dunod : Paris, 2001.

[Giorgini et al., 01] Giorgini P., Perini A. Mylopoulos J., Giunchiglia F. and Bresciani P. Agent-Oriented Software Development : A case study. In *proceeding of the 13th International conference on Software Engineering & Knowledge Engineering* (SEKE 01), Buenos-Aires, Argentina, June 13-15, 2001.

[Giorgini et al., 02] Giorgini P., Kolp M. and Mylopoulos J. Multi-Agent and Software Architectures : A comparative case study. *Proceeding of the international conference on Autonomous Agents and MAS* (AAMAS'02), Bologna, Italy, July 2002.

[Habieb-Mammar et al., 02] Habieb-Mammar H., Tarpin-Bernard et Prevot P. Modélisation XML des interfaces adaptatives intégrant le profil cognitif de l'utilisateur. In *Documents Virtuels Personnalisables* (DVP 2002). Brest, ENST Bretagne, 10 & 11 juillet 2002.

[Harold et Means 01] Harold E.R. and Means W.S. *XML in a nutshell*. Paris : O'Reilly, 2001.

[Hodgson 91] Hodgson R. The X-model: a process model for object-oriented software development. *Acte du congrès Le Génie Logiciel et ses applications*, Toulouse, 1991.

[IGL 89] I.G.L. Technology. *SADT, un langage pour communiquer*. Paris : Eyrolles, 1989.

[Jacobson et al., 00] Jacobson I., Booch G. and Rumbaugh J. *Le processus unifié de développement logiciel*. Paris : Eyrolles, 2000.

[Jamont 05] Jamont J.P. *DIAMOND : une approche pour la conception de système multi-agents embarqués*. Thèse de doctorat, Institut National Polytechnique de Grenoble, septembre 2005.

[Koch et al., 01] Koch N., Kraus A. and Hennicker R. The Authoring Process of the UML-based Web Engineering Approach. In *1st International Workshop on Web-Oriented Software Technology IWWOST'2001*, Valencia, Spain, 18-20 June, 2001.

[Kolski 01] Kolski C. (dir) *Analyse et conception de l'IHM*. Paris : Hermes, 2001.

[Kruchten 00] Kruchten P. *Introduction au Rational Unified Process*. Paris : Eyrolles, 2000.

[Lepreux 05] Lepreux S. *Approche de Développement centré décideur et à l'aide de patrons de Systèmes Interactifs d'Aide à la Décision, Application à l'investissement dans le domaine ferroviaire*. Mémoire de Doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, juin 2005.

[Lepreux et al., 03] Lepreux S., Kolski C. and Abed M. Design method for component-based DSS. *International Workshop on Component-Based Business Information Systems Engineering* (CBBISE), Geneve, Suisse, 2003.

[Mahfoudi et al., 95] Mahfoudi A., Abed M. and Angué J-C. An Object Oriented Methodology for Man-Machine systems analysis and design. In Anzai Y., Ogawa K., Mori H. (Eds.), *Symbiosis of Human and Artefact, HCI International'95: 6th Internationa*l, Tokyo, Japan, Elsevier, Amsterdam, pp. 965-970, janvier 1995.

[McDermid et Ripkin 84] McDermid J. and Ripkin K. *Life cycle support in the ADA environment*. Cambridge University Press, 1984.

[Millot et Roussillon 91] Millot P. and Roussillon E. Man-Machine cooperation in Telerobotics : Problematics and Methodologies. *Second Symposium on Robotics*, Gif-sur-Yvette, 1991.

[Morejon 94] Morejon J. *MERISE, vers une modélisation orientée objet*. Paris : Les Editions d'Organisation, 1994.

[Nanci et Espinasse 01] Nanci D. et Espinasse B. *Ingénierie des systèmes d'information : MERISE.* 4ème édition. Paris : Vuibert, 2001.

[Nodine et al., 03] Nodine M., Ngu A.H.H., Cassandra A. and Bohrer W.G. Scalable Semantic Brokering over Dynamic Heterogeneous Data Sources in InfoSleuth™. *IEEE Transactions on Knowledge and Data Engineering*, vol. 15(5), pp. 1082-1098, 2003.

[Nurcan 96] Nurcan S. Analyse et conception de systèmes d'information coopératifs. *Technique et Science Informatiques*, 15(9), pp. 1287-1315, 1996.

[O'Grady et al., 05] O'Grady M.J., O'Hare G.M.P. and Sas C. Mobile agents for mobile tourists: a user evaluation of Gulliver's Genie. *Interacting with Computers*, 17, pp. 343-366, 2005.

[O'Hare et O'Grady 03] O'Hare G.M.P. and O'Grady M.J. Gulliver's Genie: a multi-agent system for ubiquitous and intelligent content delivery. *Computer Communications*, 26, pp.1177-1187, 2003.

[Odell et al., 00] Odell J., Van Dyke Parunak H. and Bauer B. Extending UML for Agents. *Proceeding of the Agent-Oriented Information Systems Workshop at the 17th National conference on Articial Intelligence*. Gerd Wagner, Yves Lesperance and Eric Yu eds., Austin, TX, pp. 3-17, accepted paper, *AOIS Workshop at AAAI*, 2000.

[Odell et al., 99] Odell J., Van Dyke Parunak H and Bock C. Representing agent interaction protocols in UML. In *OMG Document/ad99-12-01*. Intellicorp Inc. December 1999.

[Paquel et Bezaut 02] Paquel N. et Bezaut O. *XML et développement des EDI*. Paris : Hermès, 2002.

[Pascot et Bernadas 93] Pascot D. et Bernadas C. L'essence des Méthodes : Etude Comparative de Six Méthodes de Conception de Systèmes d'Information Informatisés. *Actes du congrès INFORSID'93 « Systèmes d'information, systèmes à base de connaissances »,* Lille, 11-14 Mai 1993.

[Petit-Rozé 03] Petit-Rozé C. *Organisation Multi-Agent au service de la personnalisation de l'information : Application à un système d'information multimodale pour le transport terrestre de personnes*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Décembre 2003.

[Puerta et Eisentein 06] Puerta A. and Eisentein J. *XIML : A universal language for user interfaces*. Accessible sur le web en janvier 2006 sur : http://www.ximl.org.

[Roques et Vallée 00] Roques P. et Vallée F. *UML en action : de l'analyse des besoins à la conception en java*. Paris : Eyrolles, 2000.

[Rosenberg et Scott 99] Rosenberg D. and Scott K. *Use Case Driven Object Modeling with UML : a Pratical Approach*. Addison Wesley : Longman, 1999.

[Rouillard 02] Rouillard J. A multimodal E-commerce application coupling HTML and VoiceXML. In *the eleventh international world wide web conference* (www 2002). Honolulu, Hawaii, USA, 7-11 may 2002.

[Rouillard 04] Rouillard J. *VoiceXML, le langage d'accès à Internet par téléphone*. Paris : Vuibert, 2004.

[Royce 70] Royce W.W. Managing the Development of Large Software Systems. *Proceedings of IEEE WESCON*. August 1970.

[Soutou 02] Soutou C. *De UML à SQL : Conception de bases de données*. Editions Eyrolles : Paris, 2002.

[Tabary 01] Tabary D. *Contribution à TOOD, une méthode à base de modèles pour la spécification et la conception des systèmes interactifs*. Mémoire de Doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, janvier 2001.

[Tardieu et al., 00] Tardieu H., Rochfield A. et Colletti R. *La méthode MERISE, tome 1 : principes et outils*. Editions d'organisation : Paris, 2000.

[Tardieu et al., 83] Tardieu H., Rochfield A. et Colletti R. *La méthode MERISE, tome 1 : principes et outils*. Editions d'organisation : Paris, 1983.

[Toffolon 99] Toffolon C. *Cours de genie logiciel ou comment amener de petits génies à écrire des grands logiciels*. Cours de DESS-ICC, Université du Littoral, 1999.

[Wooldridge et al., 00] Wooldridge M., Jennings N.R. and Kinny D. The Gaia Methodology for Agent-Oriented Analysis and Desing. *Journal of Autonomous Agents and Multi-Agent Systems*, 3(3), pp. 285-312, 2000.

[Zaniolo 91] Zaniolo C. *The Logical Data Language (LDL): An Integrated Approach to Logic and Databases*. MCC Technical Report STP-LD-328-91, 1991.

Chapitre 3
PerMet, Méthodologie de développement de SI Personnalisé

Sommaire

Introduction

L'état de l'art du chapitre 1 a montré que des travaux de recherche très actifs visent à fournir une interaction homme-machine adaptative, intelligente et personnalisée dans le but d'améliorer l'efficacité de l'interaction et l'utilisabilité des systèmes. Ainsi, différentes méthodes et techniques de collecte d'information sur l'utilisateur, de modélisation de l'utilisateur et de gestion de profil utilisateur sont fournies dans la littérature. Cependant, à notre connaissance, il n'existe pas de méthode de génie logiciel qui soit dédiée au développement de systèmes d'information personnalisés.

Un des objectifs de nos travaux est de proposer une méthode adaptée à l'analyse, à la conception et à la modélisation de systèmes d'information personnalisés. Cette méthode devrait permettre et faciliter aussi bien le développement d'un nouveau système d'information que la personnalisation d'un système d'information existant. Elle devrait aussi prendre en compte différents types de personnalisation en favorisant la construction de système de personnalisation évolutif et distribué.

Pour répondre à cet objectif, la méthode doit satisfaire les critères méthodologiques, technologiques et représentationnels définis dans le chapitre 2. Comme l'a montré le chapitre précédent, aucune des méthodes de génie logiciel étudiées ne répond entièrement à ces critères. Il convient donc de construire une nouvelle méthode en adaptant et en intégrant les parties les plus pertinentes, vis-à-vis de nos objectifs, des méthodes existantes.

Cette méthode devra couvrir l'ensemble des phases nécessaires depuis l'analyse des besoins jusqu'à l'évaluation. Elle devra aussi proposer les formalismes de représentation utiles pour chaque phase.

Dans ce chapitre, nous allons décrire la méthode que nous proposons, intitulée PerMet (PERsonalization METhodology), pour l'analyse, la conception et la modélisation de systèmes d'information personnalisés.

La première partie de ce chapitre fournit une présentation générale de notre approche de développement de système d'information personnalisé vu comme un ensemble de services. Nous expliquons aussi pourquoi nous préconisons une construction de système de personnalisation basé sur des agents logiciels.

La deuxième partie décrit les différentes phases de notre méthode.

Enfin, la troisième partie concerne des propositions de formalismes de modélisation utiles pour l'analyse et la conception de systèmes d'information personnalisés.

3.1. Présentation globale de la méthode

Notre approche se situe dans la lignée des approches qui séparent le Système d'Information (SI) du Système de Personnalisation (SP) [Kobsa et Pohl 95][Trousse et al., 99][Dickinson et al., 03]. Ce choix de séparation entre SI et SP est motivé par notre objectif de personnalisation d'un SI déjà existant [Grislin-Le Strugeon et al., 06]. En effet, avec un SP indépendant du SI, il serait plus facile d'intégrer le SP avec un SI existant pour la personnalisation et obtenir un Système d'Information Personnalisé (SIP) (voir Figure 3.1). Cette séparation est aussi nécessaire pour répondre au critère multi-application qui permet le *Single Sign On* (SSO, voir chapitre 1, §1.2.2) et favorise une personnalisation de l'interaction multi-modale, multi-canal et multi-plateforme [Anli et al., 05a].

Pour mieux distinguer les parties du SI qui sont personnalisées (ou qui sont à personnaliser), nous considérons un SI comme un ensemble de services (voir Figure 3.2). Un service est « *une action effectuée par une entité (personne physique ou morale, entreprise, machine, programme) pour le bien d'une autre, avec ou sans contrepartie* »[27]. En considérant, par exemple, le portail web Yahoo[28] comme

[27] http://fr.wikipedia.org/wiki/Service
[28] http://www.yahoo.fr

un SI, un service pourrait être Yahoo!Actualités[29], Yahoo!Music[30] ou Yahoo!Auto[31]. Un service peut être composé de plusieurs services (ou sous services). Par exemple, le service Yahoo!Auto contient des services comme la recherche d'itinéraire, l'achat et la vente de véhicules d'occasion, l'information de trafic routier, etc. Un service personnalisé correspond alors à une unité fonctionnelle permettant une interaction homme-machine personnalisée. Par rapport à l'exemple Yahoo!, la personnalisation pourrait porter sur le service de recherche d'itinéraire, l'information de trafic, etc. Un SIP peut donc être vu comme un SI fournissant au moins un service personnalisé.

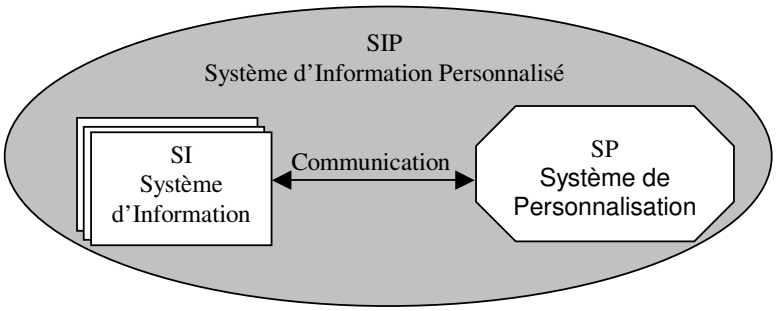


Figure 3.1. Vue générale d'un système d'information personnalisé

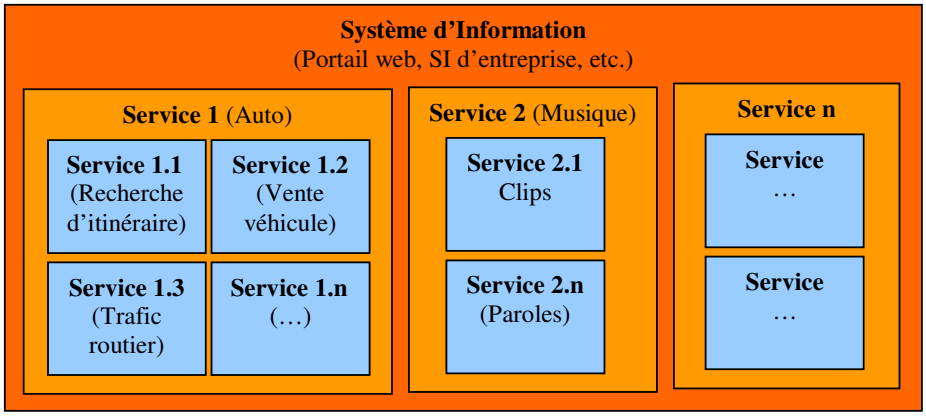


Figure 3.2. Système d'information vu comme un ensemble de services

Pour répondre à l'objectif de distributivité et d'évolutivité, nous préconisons, comme [Deschaine et al., 00] ou [Dickinson et al., 03], l'utilisation de SP à base d'agents logiciels [Anli et al., 05b]. En effet, la distributivité et l'évolutivité pourraient être facilitées grâce aux caractéristiques intrinsèques des agents logiciels (cf. chapitre 1, §1.5.1). La distributivité est assurée grâce aux caractéristiques d'autonomie, de communicabilité et de mobilité des agents logiciels. L'évolutivité est favorisée grâce aux caractéristiques d'adaptabilité et de reproductibilité.

La méthode (PerMet) que nous proposons suit un modèle de développement suivant trois parties (voir Figure 3.3). La partie SI concerne le développement d'un service de SI. La partie SP concerne l'adaptation et la configuration[32] d'un SP à base d'agents logiciels pour répondre aux objectifs de personnalisation du service. Ces deux parties SI et SP suivent les phases de développement classique (analyse, conception et implémentation) pouvant se dérouler en parallèle et se rejoignent pour former la partie commune. Le modèle développement PerMet est itératif et incrémental. Les phases doivent être parcourues plusieurs fois pour aboutir à un service personnalisé

[29] http://fr.news.yahoo.com
[30] http://fr.music.yahoo.com
[31] http://fr.cars.yahoo.com
[32] Nous expliquerons par la suite, dans le chapitre 4, ce que nous entendons par adaptation et configuration dans ce cas spécifique d'un SP à base d'agents logiciels.

opérationnel. Chaque itération donne lieu à un incrément qui vise en principe à améliorer l'utilisabilité [Nielsen 93] du service. Par exemple, pour réaliser un service de recherche d'itinéraire personnalisé, une première itération pourrait être la réalisation d'un prototype d'une application de recherche d'itinéraire qui permet de recueillir les préférences de l'utilisateur ; une deuxième itération pourrait concerner la personnalisation des résultats suivant les préférences de l'utilisateur ; une troisième itération pourrait amener à produire des explications sur le résultat fourni ; etc. Les itérations pour le développement d'un service personnalisé sont planifiées à l'avance. Cependant, il n'est pas nécessaire de spécifier tous les services à personnaliser dans un SI. Les autres services peuvent être spécifiés et développés au fur et à mesure des besoins.

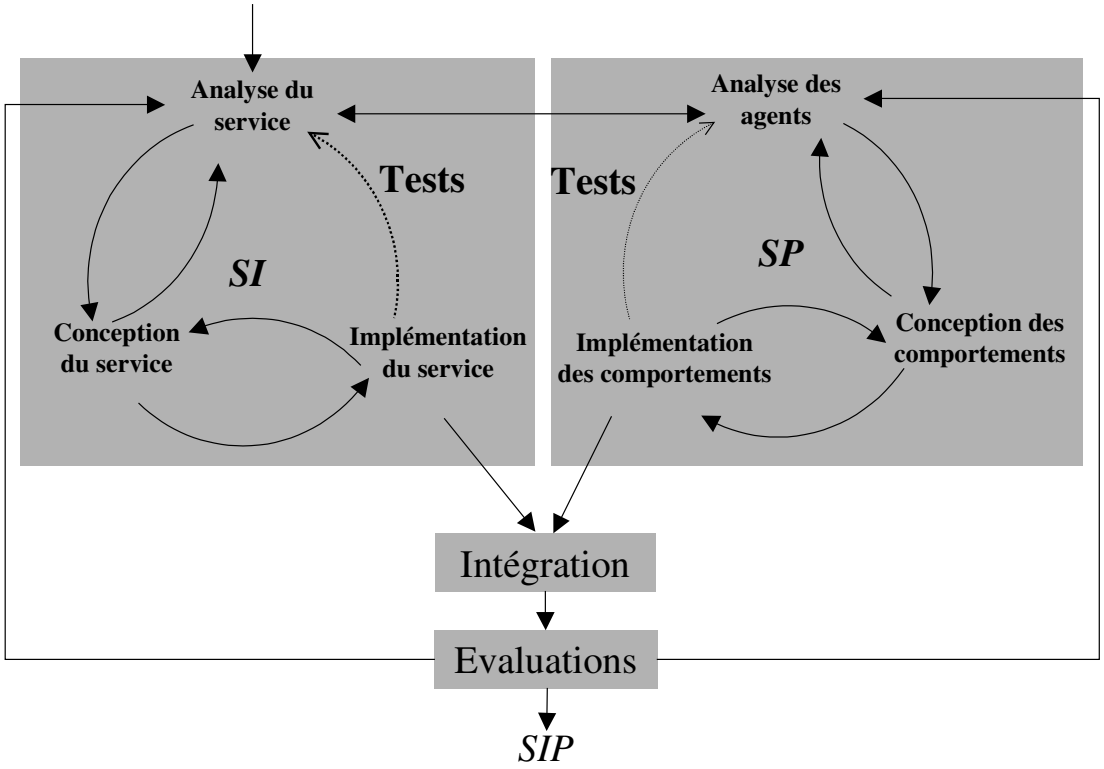


Figure 3.3. Le modèle de développement de la méthode PerMet

3.2. Les différentes phases de la méthode PerMet

Cette section détaille les différentes phases de la méthode PerMet. La planification des itérations et les études préliminaires pour l'identification des services ne sont pas traitées dans ce mémoire (voir par exemple, [Jacobson et al., 00] pour plus de détails).

La réalisation effective des différentes phases peut légèrement varier par rapport à ce que nous allons décrire ci-dessous. En effet, selon les technologies et l'expérience propre du développeur d'autres concepts peuvent intervenir dans la méthode. Par exemple, il est évident que le développeur ne suivra pas exactement les recommandations que nous proposons dans les différentes phases de la partie SP suivant que le SP soit à base d'agents MADKit (Multi-Agent Development Kit) [Gutknecht et al., 00] ou à base d'agents Jade (Java Agent DEvelopment Framework) [Bellifemine et al., 00], par exemple. Le développeur sera donc amené à ajouter ou à retirer certaines activités par rapport aux contraintes spécifiques liées à la plate-forme développement d'agents (voir par exemple [Gutknecht et Ferber 98] et [Nikraz et al., 06] pour les méthodes préconisées liées aux plates-formes respectives MADKit et Jade).

Il n'est pas toujours nécessaire de suivre toutes les phases du modèle PerMet. Certaines phases de la méthode doivent être adaptées par rapport à l'existant ou par rapport aux développements déjà

effectués. PerMet utilise le Tableau 3.1, inspiré de [Nanci et Espinasse 01], pour l'adaptation des différentes phases.

	Ignorer	Réduire	Normal	Développer
Partie SI				
Analyse du service				√
Conception du service			√	
Implémentation du service			√	
Partie SP				
Analyse des agents		√		
Conception des comportements		√		
Implémentation des comportements		√		
Partie commune				
Intégration		√		
Evaluation			√	

Tableau 3.1. Adaptation des phases du modèle PerMet

La Figure 3.4 présente quatre modèles de développement adaptés de PerMet pour la construction d'un service personnalisé. Le modèle (a) convient dans le cas où le service n'existe pas et que le type de personnalisation à réaliser n'a jamais été mis en place. Le modèle (b) correspond au cas où le service existe déjà mais que le type de personnalisation à réaliser n'a jamais été mis en place. Il s'agira donc de développer le type de personnalisation et de revoir le service pour le prendre en compte. Un faible effort consacré à la partie SI et à la partie du milieu est fourni dans le cas (c) où le service et le type de personnalisation souhaité existent déjà. Dans le cas (d) où le type de personnalisation souhaité avait déjà été mis en place mais que le service n'existe pas encore, le développeur ignore la partie SP pour se consacrer à la partie SI. L'intégration et l'évaluation (branche du milieu) sont réduites car certaines activités de ces phases avaient déjà eu lieu lors de la première intégration de la partie SP.

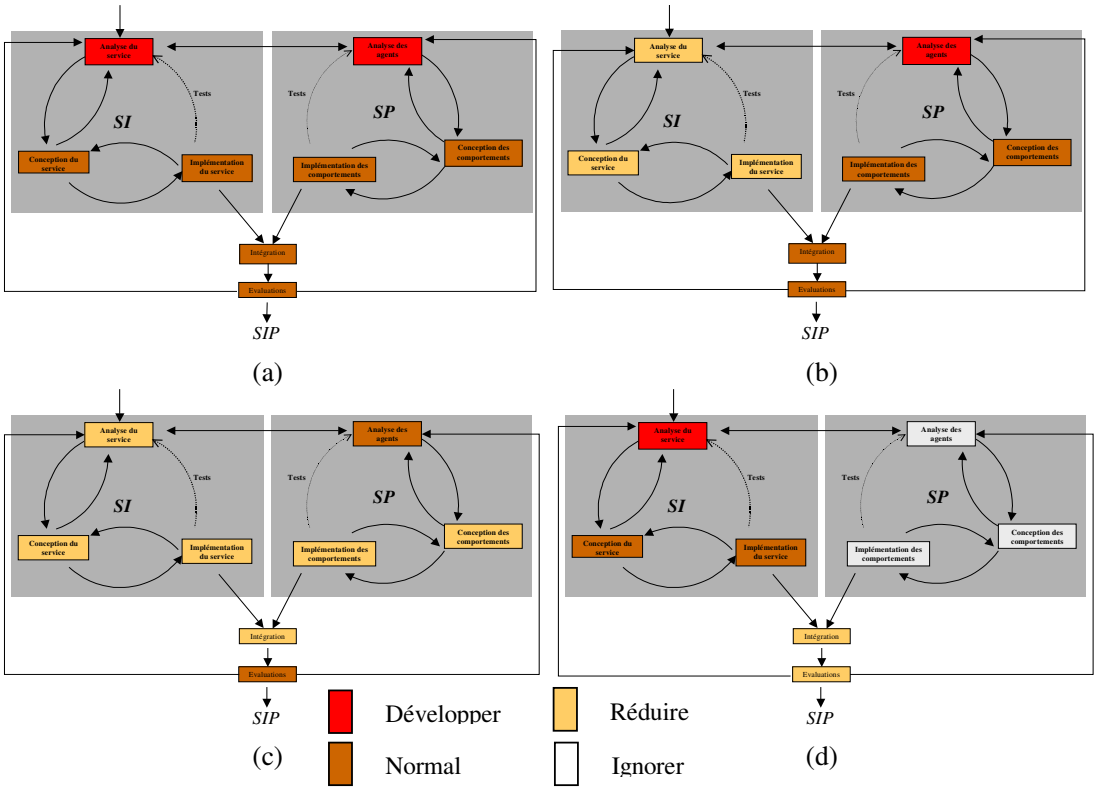


Figure 3.4. Quatre modèles adaptés de PerMet pour le développement d'un service personnalisé

3.2.1. Analyse du service

Cette phase présuppose qu'une étude préliminaire a été effectuée et que le service personnalisé à mettre en œuvre est identifié. Cette phase suit le modèle d'analyse de 2TUP (2 Track Unified Process) [Roques et Vallée 00] pour répondre aux contraintes d'évolution du service. L'analyse du service est donc effectuée suivant un aspect fonctionnel et suivant un aspect technique pouvant se dérouler en parallèle (voir Figure 3.5). Evidemment, les activités à réaliser pour chaque aspect dépendent fortement du projet. Par exemple, pour la personnalisation d'un service déjà existant, les étapes liées à l'aspect technique seront allégées par réutilisation des modèles existants issus du développement initial du service. Dans ce cas, l'analyste se consacrera surtout aux activités d'analyse fonctionnelle liées aux besoins de personnalisation.

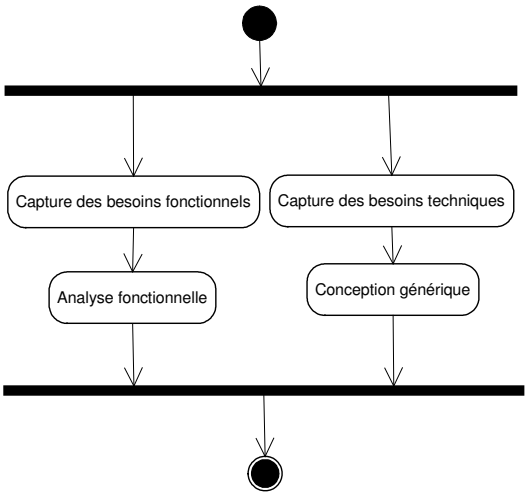


Figure 3.5. Les étapes de la phase d'analyse du service

La phase d'analyse comprend donc quatre étapes :

Capture des besoins fonctionnels : la capture des besoins décrit le service à développer. Il s'agit d'une description exhaustive des besoins fonctionnels et opérationnels par la modélisation des comportements attendus du service. Les scénarios d'interaction entre l'utilisateur final et le service sont modélisés. Les exigences fonctionnelles du service sont modélisées sans imposer le mode de réalisation de ces fonctionnalités. Arrivés à ce point du cycle, il est très tôt pour décrire la manière dont l'utilisateur va interagir avec le service. Par exemple, on dira « le système recueillera le choix de l'utilisateur » au lieu de dire « l'utilisateur doit cocher un itinéraire puis cliquera sur le bouton valider ».

Analyse fonctionnelle : l'analyse fonctionnelle décrit la structure du service en partant des besoins fonctionnels. Il s'agit de structurer et de représenter les objets du domaine métier du service par une série de diagrammes statiques et dynamiques. C'est aussi dans cette étape que les modèles de données échangées entre le service et le système de personnalisation sont définis. Ces modèles de données serviront de support commun entre la partie SI (notamment dans les phases de conception et d'implémentation du service) et la partie SP (notamment dans les phases de conception et d'implémentation des comportements des agents).

Capture des besoins techniques : la capture des besoins techniques recense toutes les contraintes qui ne traitent ni de la description du métier des utilisateurs, ni de la description du service. Il s'agit de la spécification des outils (logiciels), de la structure des matériels à exploiter et des contraintes d'intégration avec l'existant.

Conception générique : la conception générique définit les éléments nécessaires à la construction de l'architecture technique indépendamment des aspects fonctionnels issus de la capture des besoins fonctionnels et de l'analyse fonctionnelle. L'architecture doit être construite de manière à

favoriser la réutilisation. Cette conception générique peut aboutir à un développement d'un ou plusieurs prototypes de manière à vérifier et à valider les principes définis.

3.2.2. Conception du service

La conception vise à préciser le modèle d'analyse de telle sorte qu'il puisse être implémenté avec les éléments de l'architecture. Il s'agit d'exprimer les modèles statiques et dynamiques qui seront traduits directement sous forme de codes exécutables dans la phase d'implémentation du service. Pour plus de détail voir [Roques et Vallée 00], par exemple.

3.2.3. Implémentation du service

C'est l'étape de la réalisation effective du service. Le service est développé conformément aux modèles conceptuels définis lors de la phase de conception du service. Pendant l'implémentation, le développeur veillera à ce que le modèle de conception reflète exactement le code informatique produit. Des tests du service peuvent s'effectuer par simulation des données (conformément aux modèles de données définis dans la phase d'analyse) à fournir au service (ces données seront issues du SP lorsque la phase d'intégration sera achevée). Ces tests peuvent aboutir à une révision du modèle d'analyse du service ce qui nécessitera une autre réalisation des différentes phases de la partie SI et aussi de la partie SP.

3.2.4. Analyse des agents

Il s'agit ici de l'analyse des différents agents utiles pour les besoins de personnalisation. Cette phase peut démarrer à l'issue de l'étape d'analyse fonctionnelle de la phase d'analyse du service. Les modèles des agents, les comportements, les accointances et les déploiements des agents sont identifiés à ce niveau. La Figure 3.6 décrit les activités à réaliser dans cette phase.

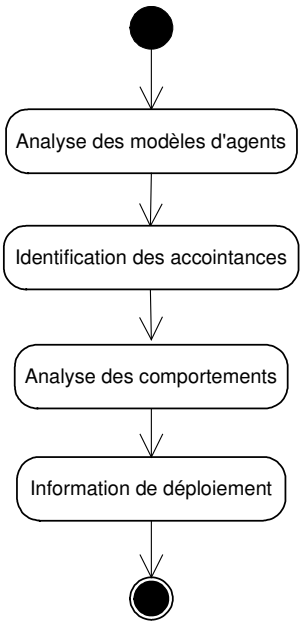


Figure 3.6. Les étapes de l'analyse des agents

Analyse des modèles d'agents : cette étape consiste à identifier les types d'agents nécessaires pour la réalisation des besoins fonctionnels définis dans la phase d'analyse du service. Les modèles d'agents peuvent être identifiés en suivant les règles ci-dessous :

Règle 1. un modèle d'agent par fonctionnalité attendue du service. Par exemple, la fonctionnalité « rechercher un itinéraire » sera traduite par un modèle d'agent de « recherche d'itinéraire ».

Règle 2. un modèle d'agent par type de plate-forme d'interaction. Par exemple, un modèle d'agent pour PDA, un autre modèle pour PC, etc.

Règle 3. un modèle d'agent par ressource externe à laquelle le système doit accéder. Par exemple, un modèle d'agent pour la recherche d'itinéraire sur les serveurs des exploitants transport, un autre pour la recherche des restaurants sur les serveurs des pages jaunes, etc.

Identification des accointances : cette étape décrit les relations entre les différents agents et leurs croyances sur les capacités des uns et des autres. Les relations d'interaction de chaque agent avec les autres y sont modélisées.

Analyse des comportements : il s'agit, ici, d'analyser pour chaque agent les comportements adéquats pour la réalisation de ses rôles. Cette analyse est assez délicate car il n'y a pas de méthode objective pour déterminer la granularité nécessaire à la décomposition des comportements. Cela dépendra donc de l'expérience et du savoir-faire du développeur. Une règle principale à appliquer consiste à décomposer un comportement tant qu'une tâche associée à ce comportement peut être associée à un autre comportement. Par exemple, prenons le comportement *« enregistrer le choix de l'utilisateur»*. Ce comportement nécessite l'exécution de la tâche « charger le profil de l'utilisateur » puis l'exécution de la tâche « mettre à jour le profil ». En analysant d'autres comportements comme par exemple *« envoyer un mail »*, la tâche « charger le profil de l'utilisateur » (pour récupérer l'adresse électronique de l'utilisateur) est également nécessaire. Il revient donc que « charger le profil de l'utilisateur » soit considéré comme un comportement.

Information de déploiement : il s'agit d'une description des différentes localisations physiques où les agents logiciels vont s'exécuter. Cette description peut aussi bien concerner les informations de localisation statique (l'agent est localisé à un même endroit et ne change jamais de place) que les informations de localisation dynamique (l'agent peut changer d'endroit dynamiquement suivant les tâches qu'il veut accomplir).

3.2.5. Conception des comportements des agents

C'est la phase de conception des comportements des agents identifiés dans la phase ci-dessus. Il s'agit d'affiner les modèles issus de la phase d'analyse des agents pour aboutir à des modèles qui puissent être directement traduits en code.

3.2.6. Implémentation des comportements des agents

Il s'agit de la réalisation effective des comportements des agents. Pour chaque comportement, des tests unitaires sont effectués. La création des agents, leur déploiement et l'intégration de leur comportement se feront grâce à des outils d'administration des agents. Ces outils sont généralement fournis par les plates-formes de développement d'agents. Par exemple, MADKit et Jade fournissent des outils graphiques permettant de créer, de déployer et d'interconnecter (relations d'accointance ou de simples relations de communications) les agents logiciels. Les outils actuels ne permettent pas de changer dynamiquement les comportements des agents en cours d'exécution pour répondre à notre objectif d'évolutivité du SP mais les caractéristiques des agents prédisposent ces outils à intégrer cette fonctionnalité. D'ailleurs, l'outil fourni par Jade prévoit cette fonctionnalité (en juin 2006, la fonctionnalité n'était pas encore opérationnelle mais les interfaces graphiques étaient déjà développées). Cette fonctionnalité devrait donc être disponible dans l'outil de Jade très bientôt.

Après avoir déployé les agents, des tests du SP doivent être effectués pour vérifier si le système multi-agents formant le SP répond bien aux objectifs définis dans la phase d'analyse des agents. Ces tests s'effectueront par simulation des données envoyées par le service (au niveau du SI)

au SP. Ces tests peuvent aboutir à une révision du modèle d'analyse des agents ce qui nécessitera une autre réalisation des différentes phases de la partie SP.

3.2.7. Intégration

C'est la phase d'intégration de SI avec le SP pour former le SIP. Il s'agit de faire communiquer le SI et le SP pour l'adaptation du service développé dans la partie SI. Cette phase peut être vue comme un problème classique de EDI (Echange de Données Informatisées) [Marchand et al., 99] ou de EAI (Enterprise Application Integration) [Manouvrier 01]. Plusieurs méthodes sont proposées pour la communication entre applications hétérogènes [Paquel et Bezaut 02][Abou-Harb et Rivard 03]. L'approche actuelle la plus utilisée consiste à utiliser une communication par service web [Monfort et Goudeau 04] qui utilise le protocole de communication SOAP (Simple Object Access Protocol) [Kadima et Monfort 03]. Cet engouement pour l'utilisation des services web se justifie, non seulement parce qu'ils permettent aux applications de communiquer au travers du réseau, indépendamment de leur plate-forme d'exécution et de leur langage d'implémentation en se basant sur des technologies standards (XML, HTTP, SMTP, …) mais aussi par leur relative facilité de mise en oeuvre. Cependant, cet engouement autour de cette technologie ne doit pas faire oublier que c'est une technologie jeune et que parfois il n'est pas aisé d'utiliser les services web pour certains besoins. Par exemple, la propagation d'un contexte transactionnel entre plusieurs services web, la gestion de l'état conversationnel des objets distribués lors de forte montée en charge, etc. Dans ces cas, l'utilisation des middlewares classiques (.Net Remoting[33], CORBA[34], RMI[35], …) en remplacement ou en complément peuvent s'avérer plus efficaces.

3.2.8. Evaluations

A l'issue de la phase d'intégration, il s'agit de tester et d'évaluer le service personnalisé obtenu. Différents types d'évaluations doivent être effectués. Nous préconisons des évaluations fonctionnelles ainsi que des évaluation de la personnalisation, de performances, de charges, techniques et ergonomiques.

- les *évaluations fonctionnelles* vérifient l'adéquation des fonctionnalités du service définies dans la phase d'analyse et le service obtenu.
- les *évaluations de la personnalisation* consistent à évaluer la qualité de personnalisation réalisée. Il s'agit de vérifier si la personnalisation effectuée au niveau du service répond bien aux préférences de l'utilisateur. Par exemple, pour un service de recherche d'information personnalisée sur Internet, une évaluation de la personnalisation consisterait à vérifier si les informations présentées à l'utilisateur correspondent bien à celles que l'utilisateur voulait recevoir. Il s'agira donc de mesurer le degré de satisfaction de l'utilisateur par rapport à la personnalisation fournie.
- les *évaluations de performances* testent la capacité du système à fonctionner rapidement sous la pression d'une forte charge.
- les *évaluations de charges* établissent le point de rupture ou la courbe de performance relative du système.
- les *évaluations techniques* évaluent la conformité du système par rapport aux normes et aux standards existants. Par exemple, pour une application accessible à partir d'un navigateur, il pourrait s'agir de la vérification de la bonne implémentation du code HTML préconisé par le W3C[36] pour assurer une meilleure portabilité de l'application sur tous les navigateurs répondant à ces normes.

[33] http://msdn.microsoft.com/library/en-us/dndotnet/html/hawkremoting.asp
[34] Common Object Request Broker Architecture [http://www.omg.org/corba]
[35] Remote Method Invocation [http://java.sun.com/products/jdk/rmi/]
[36] World Wide Web Consortium [http://www.w3c.org].

- les *évaluations ergonomiques* se consacrent à la vérification de l'utilité et de l'utilisabilité [Nielsen 93][Grislin et Kolski 96][Bastien et Scapin 01] du service proposé. De nombreuses méthodes dédiées aux évaluations ergonomiques sont proposées dans la littérature. Le Tableau 3.2 suivant fournit des exemples de méthodes parmi les plus utilisées pour les évaluations ergonomiques. Bien évidemment, il ne s'agira pas d'utiliser systématiquement toutes ces méthodes. Suivant le service à évaluer et suivant l'état d'avancement du projet (numéro d'itération), une ou plusieurs de ces méthodes devront être combinées et/ou adaptées.

Méthodes	**Principe**
Les grilles d'évaluation [Senach 90]	Les utilisateurs répondent à certaines questions relatives à des critères ergonomiques après un test d'utilisation du système. Ces questions sont notées dans une grille.
Les questionnaires d'utilisations [Moser et Karlton 71]	Les utilisateurs exécutent certaines tâches, et répondent à des questions liées à l'utilité et à l'utilisabilité du système pour la réalisation de ces tâches.
Les mouchards électroniques [Trabelsi et al., 06]	Les évènements systèmes et les actions utilisateurs sont recueillis par un logiciel. Ces données sont analysées par l'ergonome pour vérifier l'utilisabilité et l'utilité du système.
L'analyse des mouvements oculaires [Abed et Angue 90]	L'activité oculaire des utilisateurs sur l'écran est enregistrée pendant l'utilisation du système. Les données recueillies sont analysées pour évaluer les caractéristiques de l'interface homme-machine.
Les méthodes d'inspection [Nielsen et Mack 94]	L'utilisateur est assisté par un évaluateur pendant l'utilisation du système. L'ergonome analyse les données recueillies par l'évaluateur après le test d'utilisation du système.
Le recourt à l'expert [Hammond et al., 84]	Le système est évalué par rapport aux recommandations et aux normes ergonomiques existantes. Ce travail est généralement effectué par un ergonome.

Tableau 3.2. Description de méthodes d'évaluation ergonomique représentatives de celles existantes

Parallèlement à ces évaluations, des *tests de non régression* doivent être effectués pour contrôler le bon fonctionnement du SIP dans sa globalité. Il s'agira notamment de vérifier si la mise en place du nouveau service personnalisé n'a pas affecté les propriétés d'autres services du SIP.

A l'issue de ces évaluations, trois cas se présentent :

- l'itération courante est reprise si les évaluations liées à cette itération n'ont pas été concluantes. Cette reprise peut nécessiter la réalisation de la partie SI et/ou de la partie SP suivant les défauts identifiés lors de la phase des évaluations. Par exemple, si le défaut identifié concerne seulement la qualité de la personnalisation (étape d'évaluations de la personnalisation), il est très probable que le problème provient du système de personnalisation donc de la partie SP (méthodes de personnalisation pas efficace, recherche d'information très limitée, …). Cependant, il se pourrait aussi que l'observation du comportement utilisateur soit mal effectuée au niveau du service (donc partie SI) et que le SI transmet au SP des informations erronées ou ne présentant pas d'intérêt pour un apprentissage efficace des préférences de l'utilisateur.
- passage à l'itération suivante si toutes les évaluations liées à l'itération courante sont satisfaites.
- mise à disposition des utilisateurs (déploiement du SIP, livraison au client, mise en production du service, …) si toutes les itérations planifiées ont été effectuées.

3.3. Eléments de modélisation dans la méthode PerMet

Cette section a pour but de présenter les formalismes utilisés par la méthode PerMet dans les phases d'analyse et de conception. Les formalismes utilisés dans PerMet sont basés sur celles d'UML et de ses extensions[37]. Nous avons ajouté d'autres extensions à UML pour la modélisation de certaines propriétés[38] liées à PerMet et dont nous n'avons pas trouvé d'extensions existantes pouvant les prendre en compte.

Suivant le même principe que UML, PerMet représente le système selon les deux aspects complémentaires : la modélisation statique ou structurelle et la modélisation dynamique ou comportementale.

La modélisation statique s'appuie sur :

- le diagramme de cas d'utilisation pour la représentation des fonctionnalités nécessaires aux utilisateurs du système,
- les diagrammes de classes pour la représentation des entités connues des utilisateurs, de la structure du code orienté objet, de la structure des agents et de la structure des comportements des agents,
- les diagrammes d'objets qui servent à illustrer les structures de classes afin de vérifier l'adéquation d'un diagramme de classes à différents cas possibles,
- le diagramme de composants pour la représentation de la structure des composants d'exploitation comme les librairies dynamiques, les instances des bases de données, les objets distribués, les progiciels, les exécutables, etc.,
- et le diagramme de déploiement qui permet la représentation de la disposition physique des matériels qui composent le système, la répartition des composants et des plates-formes permettant l'exécution des agents sur ces matériels et la localisation physique des agents sur les dispositifs matériels.

La modélisation dynamique s'appuie sur :

- le diagramme d'états pour la représentation du cycle de vie des objets et des agents,
- le diagramme d'activité pour la représentation du déroulement d'un cas d'utilisation, de l'exécution d'une méthode et du comportement d'un agent lorsqu'il effectue une tâche (service, action interne ou action proactive[39]),
- le diagramme de collaboration pour la représentation du contexte d'interaction entre les objets et entre les agents du système,
- le diagramme de séquence pour la représentation des scénarios d'utilisation du système, des scénarios d'interaction entre les objets et des scénarios d'interaction entre les agents.

3.3.1. Modèles proposés pour l'analyse et la spécification du service

Comme nous l'avons vu au §3.2.1, la phase d'analyse du service comprend quatre étapes : capture des besoins fonctionnels, analyse fonctionnelle, capture des besoins techniques et conception générique.

[37] Nous préciserons à chaque fois qu'il s'agira d'une extension UML issue de la littérature.
[38] Ces propriétés sont surtout liées à des concepts relatifs aux agents logiciels.
[39] Nous appelons *service d'un agent*, le savoir-faire d'un agent que celui-ci met à disposition des autres agents. Une *action* interne correspond à une activité supplémentaire que l'agent doit effectuer pour répondre à un service demandé ou pour réaliser une action proactive. Une action interne peut nécessiter l'exécution d'une autre action interne. Une *action proactive* correspond à une activité réalisée par l'agent afin de satisfaire ses propres buts. Une action proactive est réalisée à l'initiative propre de l'agent.

Capture des besoins fonctionnels : la modélisation des besoins fonctionnels s'effectue au travers des *diagrammes de cas d'utilisation*. Ces cas d'utilisation vont piloter la réalisation de toutes les autres phases du cycle. Ainsi les besoins fonctionnels et particulièrement les besoins des utilisateurs resteront une préoccupation permanente durant tout le processus de développement. Chaque cas d'utilisation doit être suivi d'une fiche de description textuelle. Cette fiche de description n'est pas normalisée par UML. [Roques 05] propose une structuration de cette fiche en quatre parties comme décrit sur le Tableau 3.3. La Figure 3.7 donne un exemple d'une fiche de description textuelle pour un cas d'utilisation (il s'agit du cas d'utilisation « rechercher un itinéraire »). Il peut être utile de détailler certains cas d'utilisation par des scénarios au travers de *diagrammes de séquences*, *d'activités* et/ou *de collaboration*.

Sommaire d'identification (obligatoire)	Titre, but, résumé, version, auteur, dates, responsable, acteurs,…
Description des enchaînements (obligatoire)	Préconditions, scénarios nominaux, scénarios alternatifs, scénarios d'erreur et postconditions.
Besoins d'IHM (optionnel)	Contraintes d'IHM : séquences d'apparition des interfaces, éléments déclenchables par l'utilisateur, …
Contraintes non-fonctionnelles (optionnel)	Fréquence, temps de réponse, concurrence, disponibilité, intégrité, confidentialité,…

Tableau 3.3. Les quatre parties composant une fiche de description textuelle d'un cas d'utilisation

Sommaire d'identification :

Titre : Rechercher un itinéraire

But : fournir un itinéraire à un utilisateur en se basant sur ses préférences.

Résumé : reconnaître l'utilisateur, envoyer la requête à PerSyst et afficher l'itinéraire personnalisé à l'utilisateur.

Acteurs : utilisateur (principal), PerSyst (secondaire)

Date de création : 10/05/2004 **Date de mise à jour** : 03/02/2005

Version : 1.0 **Responsable** : A. Anli

Description des enchaînements :

Préconditions :

- L'utilisateur est authentifié

Scénarios nominaux :

N1 : Lancer une requête

Ce scénario commence lorsqu'un utilisateur veut rechercher un itinéraire.

1. l'utilisateur formule et valide sa requête.
2. le système envoie la requête de l'utilisateur à PerSyst.
3. PerSyst recherche les itinéraires répondant à la requête et les envoie au système en précisant l'itinéraire susceptible d'intéresser l'utilisateur.
4. le système affiche l'itinéraire susceptible d'intéresser l'utilisateur.

Scénarios alternatifs :

A1 : Lieu de départ non fourni

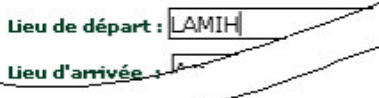
Ce scénario commence après l'étape 1 du scénario N1.

2. informer l'utilisateur que le lieu de départ est obli

Besoins d'Ihm

- Pour formuler la requête

L'utilisateur doit pouvoir formuler sa requête pour la recherche d'itinéraire en précisant le départ, le lieu d'arrivée et l'heure d'arrivée.



Contraintes non fonctionnelles :

Contraintes	Descriptifs
Temps de réponse	Les itinéraires doivent être affichés en moins de deux secondes.
Disponibilité	Le service doit être accessible aux utilisateurs 24 heures sur 24 et 7 jours sur 7
Confidentialité	Les noms des utilisateurs participant à la recommandation d'un itinéraire ne doivent pas être mentionnés dans la réponse fournie.

Figure 3.7. Exemple d'une fiche de description textuelle d'un cas d'utilisation

Analyse fonctionnelle : la structure du service est modélisée au travers de *diagrammes de classes*. Les concepts connus des utilisateurs (objets métiers) et les concepts applicatifs liés à l'informatisation permettent d'identifier les classes candidates. L'utilisation des fiches CRC (Class-Responsbility-Collaborator) [Beck and Cunningham 89] s'avère une méthode simple et efficace pour l'identification des classes et de leurs responsabilités[40]. En effet, les classes sont d'abord définies par rapport à leurs responsabilités plutôt que par leurs attributs et leurs opérations. Les responsabilités peuvent être modélisées sous forme de *note graphique* UML [Roques et Vallée 00] associée à la classe (la Figure 3.8 donne un exemple de modélisation des responsabilités d'une classe). Les responsabilités permettent ensuite d'identifier les attributs, les opérations et les associations entre les classes pour répondre aux besoins modélisés par les cas d'utilisation dans la phase d'analyse des besoins. Dans l'exemple de la Figure 3.8, la première responsabilité pourrait se traduire par des attributs et les suivantes par des associations. Le comportement du service est ensuite décrit au travers des diagrammes dynamiques (*diagramme de séquences*, d'*activités*, d'*éta*t et de *collaboration*).

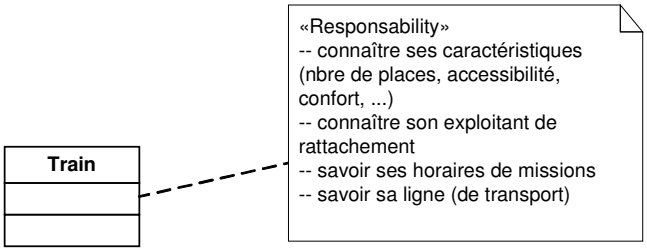


Figure 3.8. Exemple de modélisation d'une responsabilité de classe

Les modèles de données échangées entre le service et le système de personnalisation sont modélisés au travers de *diagrammes* de classes (Figure 3.9). Ces modèles de données peuvent être complétés par des *diagrammes d'objets*.

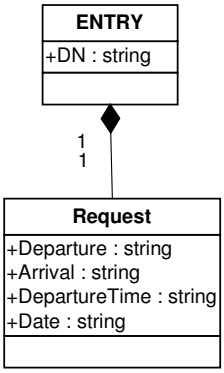


Figure 3.9. Exemple de modèle de données

Capture des besoins techniques : la configuration matérielle est modélisée au travers de *diagrammes de déploiement* et la spécification logicielle au travers de *diagrammes de composants.* La spécification logicielle peut être complétée par des *diagrammes de cas d'utilisation technique*. Chaque cas d'utilisation doit être associé à une fiche de description textuelle et peut être complété par des diagrammes dynamiques.

Conception générique : les modèles utilisés dans la conception générique concernent essentiellement *les diagrammes de classes* et de *composants.* L'analyse doit se faire de manière à

[40] Une responsabilité est un contrat ou une obligation pour une classe.

favoriser la réutilisation de ces modèles par la construction de *design pattern*[41] et de *frameworks*[42] [Pree 98].

3.3.2. Modèles proposés pour la conception du service

Les *diagrammes de classes* de la phase d'analyse du service (analyse fonctionnelle et conception générique) sont développés pour permettre un passage direct du modèle de conception à l'implémentation. Parfois il est nécessaire de réorganiser les diagrammes de classes issus de la phase d'analyse pour prendre en compte certains aspects techniques liés au langage de programmation choisi. De nouvelles classes peuvent être introduites pour prendre en compte de nouvelles responsabilités purement techniques ou pour décharger des classes issues de la phase d'analyse de certains de leurs aspects techniques. Les types des attributs de classes, les types et les paramètres d'entrées/sorties des méthodes doivent être précisés.

Les diagrammes dynamiques sont utilisés pour préciser les états des classes (*diagramme d'états*), l'algorithme des méthodes (*diagramme d'activités*) et les interactions entre les différentes classes (*diagramme de séquence* et/ou *de collaboration*).

Les *diagrammes de composants* issus de la phase d'analyse seront complétés et précisés à l'issue de la conception de toutes les classes utiles au développement du service.

La Figure 3.10 illustre les différents diagrammes et leur ordre d'utilisation dans la phase de conception du service. Dans le cas d'un service accessible sur Internet, les extensions UML proposées par [Conallen 00] sont utilisées.

[41] Un pattern est une solution générique pour un problème qui est souvent rencontré dans divers développements logiciels.
[42] Un framework est un ensemble de classes qui participent à la réalisation d'une responsabilité. Les applications sont construites par une adaptation de ces classes et réutilisent le modèle de conception fourni par ce framework.

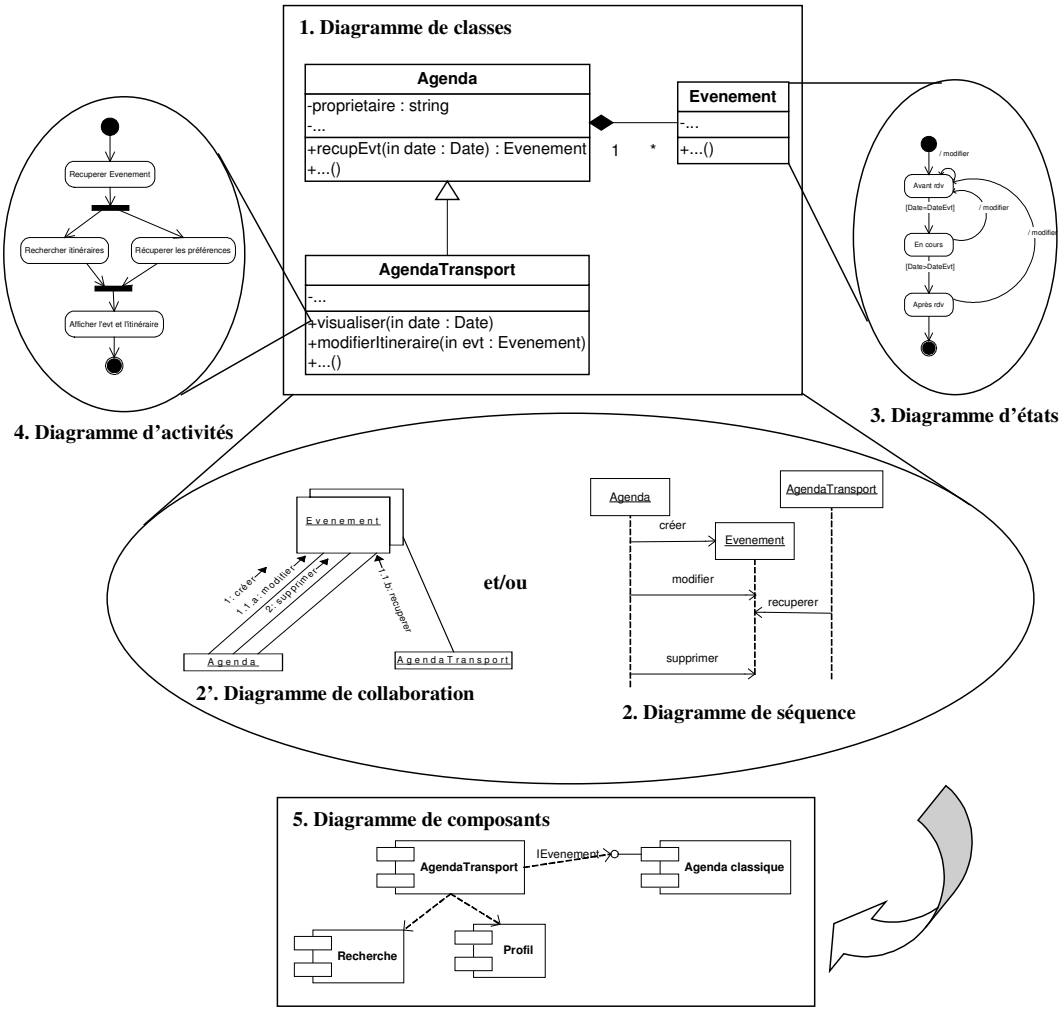


Figure 3.10. Diagrammes utilisés et leurs enchaînements dans la phase de conception du service

3.3.3. Modèles proposés pour l'analyse des agents

Comme nous l'avons vu au §3.2.4, l'analyse des agents comprend quatre étapes : analyse des modèles d'agents, analyse des comportements, identification des accointances et information de déploiement.

Analyse des modèles d'agents : l'identification des modèles d'agents nécessaires pour la mise en place du service personnalisé se base sur les diagrammes des cas d'utilisation et sur les diagrammes de déploiement de la phase d'analyse du service. En effet, les diagrammes de cas d'utilisation modélisent les fonctionnalités du service et les ressources externes (modélisées par les acteurs secondaires) avec lesquelles le service va être en interaction (règle 1 et 3 pour l'identification des modèles d'agents, cf. §3.2.4. Et les diagrammes de déploiements modélisent la configuration matérielle et donc les plates-formes d'interaction permettant d'accéder au service (règles 2, cf. §3.2.4). Dans cette étape d'analyse des modèles d'agents, les diagrammes de cas d'utilisation et de déploiement de la phase d'analyse du service sont repris sous l'angle de la personnalisation. Seuls les aspects jugés comme faisant partie de la personnalisation sont considérés. Pour cela, seuls les cas d'utilisation pouvant nécessiter une méthode d'apprentissage automatique et les matériels que l'utilisateur va utiliser pour accéder au service sont modélisés. Par exemple, le cas d'utilisation « recherche d'information » peut être considéré dans le diagramme de cas d'utilisation pour l'analyse des modèles d'agents puisque pour effectuer de la recherche d'information des méthodes

d'apprentissage automatique issues de la découverte automatique de données, par exemple, peuvent être utilisées. Par contre, le cas d'utilisation « modifier le planning » ne sera pas considéré sauf s'il y a une réorganisation automatique du planning de l'utilisateur. Cette réorganisation automatique pouvant nécessiter l'utilisation de méthodes de satisfaction de contraintes [Ma et al., 04], par exemple.

Comme dans PASSI (Process for Agent Societies Specification and Implementation) [Burrafato et Cossentino 02] la structure d'un agent est modélisée par une classe portant le stéréotype « agent ». Mais contrairement à PASSI, cette classe ne comporte pas d'attributs ni de méthodes. Une note similaire à celle présentée dans la Figure 3.8 décrit le rôle de l'agent (voir Figure 3.11). Ces rôles serviront par la suite de support pour déterminer les différents comportements dont doit disposer l'agent pour répondre à ses objectifs.

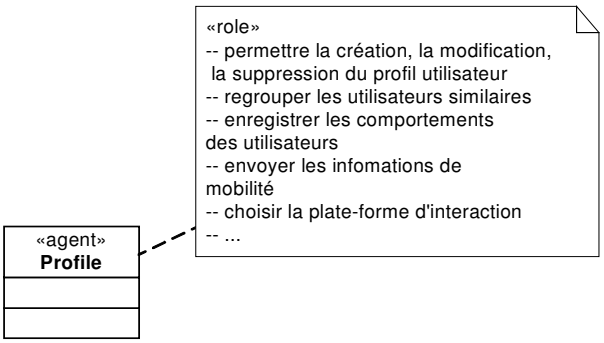


Figure 3.11. Modélisation d'un agent

Identification des accointances : les *diagrammes de séquence* et de *collaboration* permettent de modéliser les interactions entre les différents agents. PerMet utilise les extensions proposées par AUML [Odell et al., 99] pour représenter certains concepts (envoi de messages concurrents, choix d'un agent parmi d'autres, ..) spécifiques aux agents logiciels. Par exemple, la Figure 3.12 décrit un message envoyé en parallèle par l'agent *coordinator* aux agents *SearchTranspole* (c'est un agent qui a pour rôle la recherche d'information transport de Lille et de ses environs), *SearchSemurval* (cet agent se charge de la recherche d'information transport dans le valencennois) et *SearchSNCF* (s'occupe de l'information transport inter-regional). Ces diagrammes d'interaction permettent d'identifier les relations d'accointances entre les différents agents qui sont modélisées par des associations entre les classes représentant les agents. Ces associations portent le stéréotype « acquaintance ». La Figure 3.13 (a) montre que l'agent *Coordinator* connaît toutes les capacités de l'agent *Profile*. Et la Figure 3.13 (b) montre que les deux agents se connaissent mutuellement. Par contre, la Figure 3.13 (c) montre que les deux agents peuvent communiquer mais aucun des deux ne connaît ce que sait faire l'autre (pas d'association d'accointance).

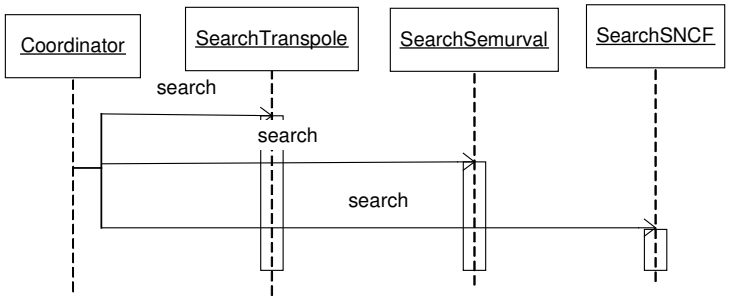


Figure 3.12. Exemple de diagramme de séquence en AUML

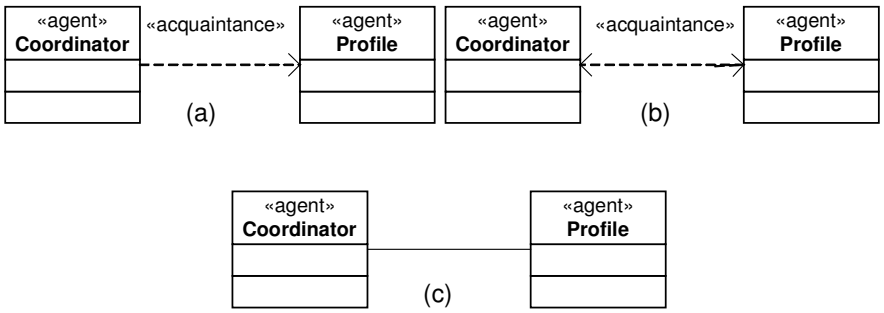


Figure 3.13. Modélisation des accointances

Analyse des comportements : les comportements nécessaires à un agent sont modélisés par des *classes* portant le stéréotype « behaviour » (voir Figure 3.14). Les attributs (toujours privés) correspondent aux états de l'agent. Les méthodes publiques correspondent aux *services* que l'agent met à disposition des autres agents. Les méthodes privées correspondent aux *actions internes* que l'agent peut effectuer. Et les méthodes protégées correspondent aux *actions proactives*. Ces méthodes peuvent être complétées par des *diagrammes d'activités*. Lorsqu'il s'agit d'une action proactive et avec différents états, la méthode représentant cette action peut être modélisée par un *diagramme d'état*.

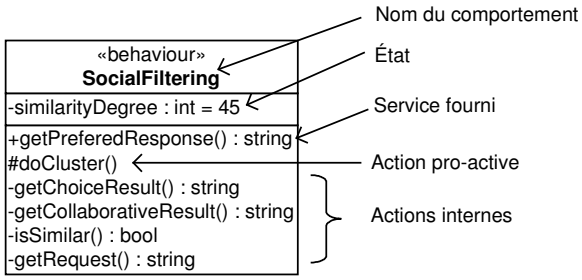


Figure 3.14. Modélisation d'un comportement

Une association entre un agent et un comportement signifie que l'agent dispose de ce comportement. Un agent peut avoir plusieurs comportements pour répondre aux rôles définis dans l'étape d'analyse des modèles d'agents. Un comportement peut être associé à plusieurs agents.

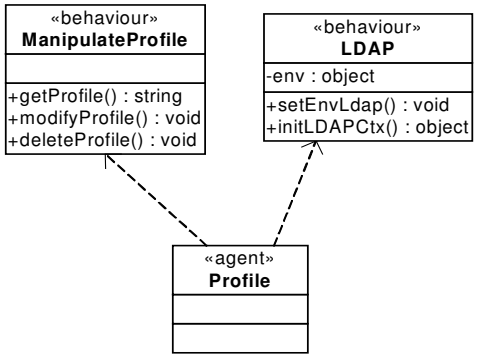


Figure 3.15. Association des comportements à un agent

Un comportement associé à un autre comportement signifie qu'au moins un service ou une action (interne ou proactive) du comportement source a besoin du comportement destination pour

terminer son exécution. Par exemple, la Figure 3.16 montre que le comportement *ManipulateProfile* a besoin du comportement *LDAP* (pour accèder à la base de données des profils utiliseur). Ce n'est pas nécessaire que les deux comportements soient au sein du même agent. Ils peuvent être au niveau de deux agents différents qui peuvent même se situer sur des machines différentes mais à condition que les deux agents aient une relation d'accointance permettant à l'agent disposant du comportement source de solliciter les services de l'agent disposant du comportement destination. Pour marquer cette différence fondamentale avec les liens d'association entre classes (au sens approche orientée objet), ces liens d'association comportent le stéréotype « *require* ».

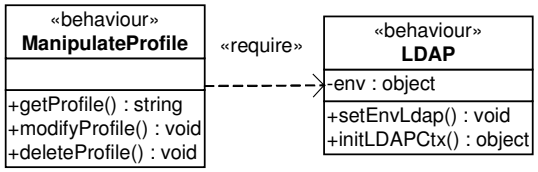


Figure 3.16. Association entre deux comportements

Information de déploiement : les informations pour la localisation physique des agents sont représentées au travers *d'un diagramme de déploiement.* Un nœud correspond à un matériel physique où un ou plusieurs agents peuvent s'exécuter. Un agent est représenté par un *objet* portant le nom de l'agent. Le fait qu'un agent doit se déplacer d'un endroit à un autre est modélisé par un lien discontinu avec le stéréotype «move» (voir Figure 3.17).

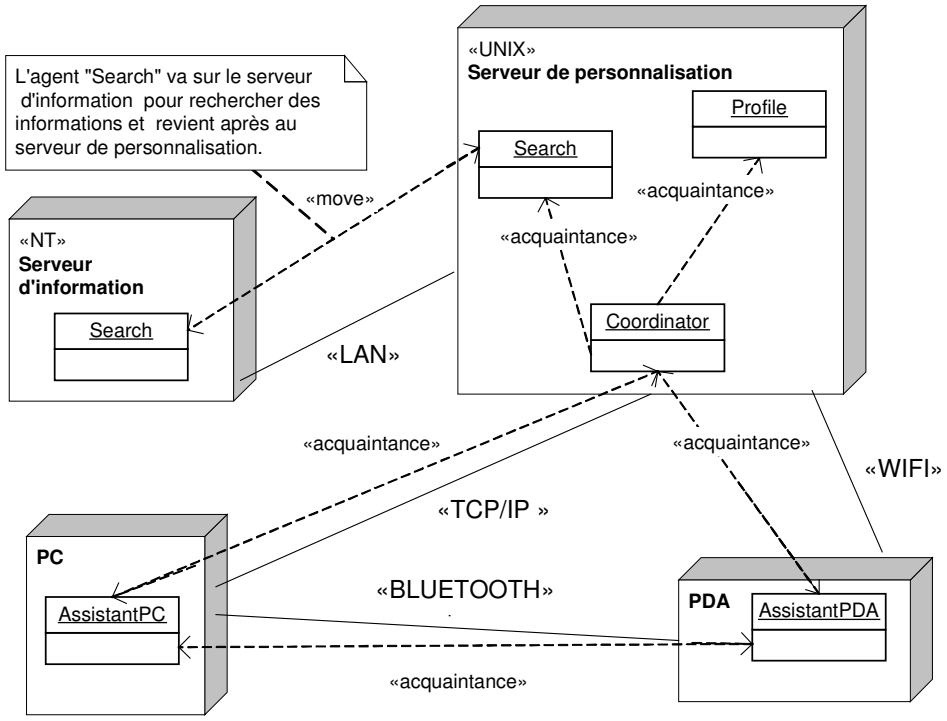


Figure 3.17. Exemple de modélisation de l'information de déploiement des agents

3.3.4. Modèles proposés pour la conception des comportements des agents

Les *diagrammes de classes* modélisant les comportements des agents sont détaillés pour permettre une implémentation directe des comportements des agents. Dans cette phase, il s'agira essentiellement de préciser les types des attributs et les paramètres d'entrées/sorties des différentes méthodes des classes modélisant les comportements des agents. De nouvelles classes de

comportements et de nouvelles actions internes peuvent apparaître grâce à une modélisation des interactions entre comportements par des *diagrammes de séquence* ou *de collaboration*.

Les activités des méthodes des classes de comportements sont modélisées au travers de *diagrammes d'activités.* Les méthodes représentant des actions proactives à état peuvent être représentées par des *diagrammes d'états*.

La Figure 3.18 illustre les différents diagrammes et leur ordre d'utilisation dans la phase de conception des comportements des agents.

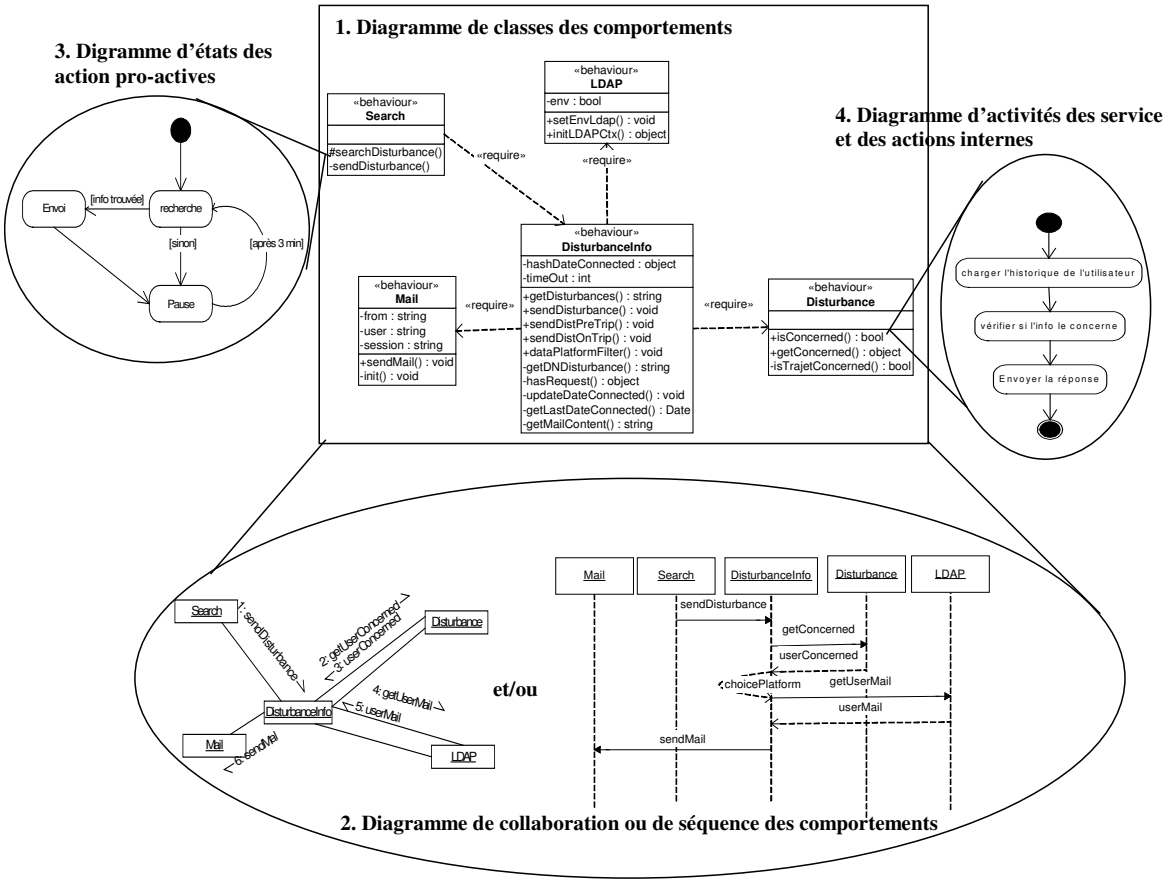


Figure 3.18. Diagrammes utilisés et leurs enchaînements dans la phase de conception des comportements

Conclusion

Ce chapitre a décrit une partie de notre contribution pour la personnalisation de l'interaction homme-machine. Nous avons proposé une méthode appelée PerMet (PERsonalization METhodology) pour le développement des systèmes d'information personnalisés. Cette méthode permet aussi bien la mise en place d'un nouveau système d'information personnalisé que la personnalisation d'un système d'information déjà existant.

Notre approche sépare le système d'information vu comme un ensemble de services du système de personnalisation. Cette séparation permet la personnalisation de l'interaction homme-machine en prenant en compte la multi-modalité, le multi-canal et la multi-plateforme. Le système de personnalisation est à base d'agents logiciels ce qui lui confère une flexibilité accrue grâce à leurs caractéristiques, entre autres, d'adaptabilité, d'autonomie, de reproductibilité et de mobilité.

PerMet propose un modèle de développement itératif, incrémental et permet une réalisation parallèle des phases spécifiques liées au développement des services et des phases spécifiques liées à la personnalisation. Les activités à effectuer dans chaque phase aussi bien au niveau des services à personnaliser qu'au niveau du système de personnalisation ont été décrites. Les formalismes utiles à la modélisation structurelle et comportementale pour les phases d'analyse et de conception ont été précisés. Ces formalismes sont basés sur ceux d'UML et de ses extensions ce qui permettrait une prise en main relativement facile (puisque une grande majorité de développeurs ont adopté UML) et dont il existe de nombreux ateliers de génie logiciel dédiés.

Nous avons vu que PerMet préconise l'utilisation d'un système de personnalisation évolutif et distribué pouvant prendre en compte différent types de personnalisation (cf. chapitre 1). L'étude des systèmes de personnalisation (cf. chapitre 1) a montré qu'il n'existe pas ou très peu de systèmes de personnalisation répondant à ces caractéristiques. Pour faciliter l'utilisation de notre méthode, nous nous proposons de fournir un système de personnalisation supportant la méthode PerMet. Le chapitre suivant décrit, à ce propos, un système de personnalisation générique pouvant être utilisé conjointement avec PerMet pour le développement de systèmes d'information personnalisés.

Bibliographie du chapitre 3

[Abed et Angue 90] Abed M. and Angue J.C. Using the measure of eye mouvements to modelise an operator's activity. *Ninth European Annual conference on "Human decision making and manual control",* Varese, Italy, September 1990.

[Abou-Harb et Rivard 03] Abou-Harb G. et Rivard F. *L'EAI au service de l'entreprise évolutive*. Paris : Maxima, 2003.

*[Anli et al., 05a] Anli A., Grislin-Le Strugeon E. et Abed M. A Generic Personalization Tool based on a Multi-agent Architecture. In C. Stephanidis (Ed.), *Proceedings HCI International (Las Vegas, Nevada, July 22-27, 2005), Volume 7 - Universal Access in HCI: Exploring New Interaction Environments*, Lawrence Erlbaum Associates, Mahwah, New Jersey, pp. 1-8, juillet 2005.

*[Anli et al., 05b] Anli A., Kolski C. et Abed M. Principes et architecture pour la personnalisation d'information en interaction homme-machine : Application à l'information transport. In *Proceedings of IHM 2005, International Conference Proceedings Series*, ACM Press, Toulouse, pp. 123-130, septembre 2005.

[Bastien et Scapin 01] Bastien J. et Scapin D. Evaluation des systèmes d'information et critères ergonomiques. In *C. Kolski (Ed.), Environnements évolués et évaluation de l'IHM. Interaction Homme-Machine pour les SI*, pp. 53-79, Paris : Editions Hermès, 2001.

[Beck and Cunningham 89] Beck K. and Cunningham W. A Laboratory For Teaching Object-Oriented Thinking. In *the OPSLA'89 Conference Proceedings* October 1-6, 1989, New Orleans, Louisiana. Special issue of SIGPLAN Notices 24(10), 1989. Disponible à http://c2.com/doc/oopsla89/paper.html.

[Bellifemine et al., 00] Bellifemine F., Poggi A. and Rimassa G. Developing multi-agent systems with JADE. In se*venth International Workshop on Agent Theories, Architectures, and Languages (ATAL-2000)*, Boston, MA, pp. 85-99, 2000.

[Booch et al., 00] Booch G., Rumbaugh J. and Jacobson I. *Le guide de l'utilisateur UML*. Paris : Eyrolles, 2000.

[Burrafato et Cossentino 02] Burrafato P. and Cossentino M. Designing a multi-agent solution for a bookstore with the PASSI methodology. *Proceedings of the Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems* (AOIS-2002 at CAiSE*02), Toronto (Ontario, Canada), May 27-28, 2002.

[Conallen 00] Conallen J. *Concevoir des applications web avec UML*. Paris : Eyrolles, 2000.

[Deschaine et al., 00] Deschaine L., Brice R. and Nodine M. Use of InfoSleuth to Coordinate Information Acquisition, Tracking and Analysis in Complex Applications. In *Proceedings of Advanced Simulation Technologies Conference*, April 2000.

[Dickinson et al., 03] Dickinson I., Reynolds D., Banks D., Cayzer S. and Vora P. User Profiling with privacy : A framework for Adaptive Information Agents, M. Klush et al. (Eds.) : Intelligent Information Agents, , LNAI 2586, pp 123-151, Springer-verlag, Berlin, 2003.

[Grislin et Kolski 96] Grislin M. et Kolski C. Evaluation des interfaces homme-machine lors du développement de système interactif. *Technique et Science Informatiques (TSI)*, 2, pp. 265-296, 1996.

*[Grislin-Le Strugeon et al., 06] Grislin-Le Strugeon E., Anli A. and Adam E. A methodology to bring MAS to information systems. In T. Latour, M. Petit (Eds.), *CAISE'06 The 18th International Conference on Advances Information Systems Engineering (Luxembourg, June 5-9, 2006), Proceedings of Workshops and Doctoral Consortium*, Presses Universitaires de Namur, pp. 64-75, juin 2006.

[Gutknecht et al., 00] Gutknecht O., Ferber J. Michel F. MADKit : une expérience d'architecture de plateforme multi-agent générique. In *Proceedings of the JFIADSMA'00*, Hermès, pp. 223-236, 2000.

[Gutknecht et Ferber 98] Gutknecht O. et Ferber J. Un modèle d'analyse, de construction et d'exécution de systèmes multi-agents. In *Jean-Pierre Barthès (eds.), Actes des Journées Francophones en Intelligence Artificielle Distribuée et Système Multi-Agents 1998*, Hermès, novembre 1998.

[Hammond et al., 84] Hammond N., Hinton G., Barnard P., Maclean A., Long J. and Whiteield A. Evaluating the interface of a document processor: a comparison of expert judgement and user observation. *Proceeding of the First IFIP Conference on Human-Computer Interaction: Interact'84*, London, pp. 725-729, 1984.

[Jacobson et al., 00] Jacobson I., Booch G. and Rumbaugh J. *Le processus unifié de développement logiciel*. Paris : Eyrolles, 1999.

[Kadima et Monfort 03] Kadima H. et Monfort V. *Les web services : Techniques et outils XML, WSDL, SOAP, UDDI, Rosetta, UML*. Paris : Dunod, 2003.

[Kobsa et Pohl 95] Kobsa A. and Pohl W. The User Modeling Shell System BGP-MS. *User Modeling and User-Adapted Interaction* Vol. 4, pp. 59-106, 1995.

[Ma et al., 04] Ma J., Piechowiak S. and Mandiau R. A Multi-Factor Concept on Guiding Constraint Relaxation in Distributed Constraint Satisfaction. *WSEAS Transactions on Computers*, 3, pp. 442-448, 2004.

[Manouvrier 01] Manouvrier B. *EAI : intégration des applications d'entreprise*. Paris : Hermès science publications, 2001.

[Marchand et al., 99] Marchand R., Agnoux H. et Chiaramonti C. *Application EDI sur l'Internet : commerce électronique B to B*. Paris : Eyrolles, 1999.

[Monfort et Goudeau 04] Monfort V. et Goudeau S. *Web services et Interopérabilité des SI : WS-I, WSAD/J2EE, Visual Studio .Net et BizTalk*. Paris : Dunod, 2004.

[Moser et Karlton 71] Moser C. and Karlton G. *Survey methods in social investigation*. 2nd Edition. London : Hermann, 1971.

[Nanci et Espinasse 01] Nanci D. et Espinasse B. *Ingénierie des systèmes d'information : MERISE*. 4ème édition. Paris : Vuibert, 2001.

[Nielsen 93] Nielsen J. *Usability Engineering*. Academic Press : London, 1993.

[Nielsen et Mack 94] Nielsen J. and Mack R. L. *Usability inspection methods*. New York : john Wiley & Sons, 1994.

[Nikraz et al., 06] Nikraz M., Caire G. and Bahri P.A. A Methodology for the Analysis and Design of Multi-Agent Systems using JADE. *International Journal of Computer Systems Science & Engineering*, special issue on A. Garcia and C. Lucena (Eds.), "*Software Engineering for Multi-Agent Systems*". Vol. 21 (2), 2006.

[Odell et al., 99] Odell J., Van Dyke Parunak H and Bock C. Representing agent interaction protocols in UML. In *OMG Document/ad99-12-01*, Intellicorp Inc., December 1999.

[Paquel et Bezaut 02] Paquel N. et Bezaut O. *XML et développement des EDI*. Paris : Hermès, 2002.

[Pree 98] Pree W. *Design patterns et architectures logicielles*. Paris : Vuibert 1998.

[Roques 05] Roques P. *UML 2 par la pratique – Etudes de cas et exercices corrigés*. Paris : Eyrolles, 2005.

[Roques et Vallée 00] Roques P. et Vallée F. *UML en action : de l'analyse des besoins à la conception en java*. Paris : Eyrolles, 2000.

[Senach 90] Senach B. *Evaluation ergonomique des interfaces Homme-machine : une revue de la littérature*. Rapport de recherche, INRIA, Sophia Antipolis, n° 1180, 1990.

[Trabelsi et al., 06] Trabelsi A., Ezzedine H. et Kolski C. Un mouchard électronique orienté agent pour l'évaluation de systèmes interactifs de supervision. In *Conférence Internationale Francophone d'Automatique CIFA'2006,* Bordeaux, France, mai 2006.

[Trousse et al., 99] Trousse B., Jaczynski M. and Kanawati R. Using user behavior similarity for recommandation computation : The broadway approach, In *Proceedings of the 8th international conference on Human Computer Interaction (HCI'99),* Munich, August 1999.

Chapitre 4
PerSyst, un système de personnalisation supportant la méthode PerMet

Sommaire

Introduction

Le chapitre précédent a décrit une méthode nommée PerMet pour le développement de système d'information personnalisé. PerMet sépare le système d'information du système de personnalisation pour prendre en compte dans le processus de personnalisation différentes modalités d'entrées-sorties (son, image, braille, etc.), différents canaux de communication (Internet, SMS, courriel, etc.) et différentes plate-forme d'interaction (PC, PDA, télévision, PSP[43], etc.).

La méthode PerMet insiste sur la nécessité d'un système de personnalisation évolutif et distribué pouvant prendre en compte différents types de personnalisation. Pour cela, PerMet préconise l'utilisation d'un système de personnalisation à base d'agents logiciels.

Cependant, il n'existe pas ou très peu de système de personnalisation répondant à ces caractéristiques. Pour faciliter l'utilisation de PerMet, nous proposons PerSyst (PERsonalization SYSTem), un système de personnalisation générique pouvant s'utiliser conjointement avec PerMet pour le développement de système d'information personnalisé.

Après une présentation générale de la plate-forme utilisée pour le développement des agents logiciels composant PerSyst, ce chapitre décrit l'architecture générale et la conception de notre système de personnalisation ; les différentes fonctionnalités permettant l'évolutivité et la distributivité de PerSyst sont exposées. Enfin, la dernière section de ce chapitre présente des modèles généraux utiles au développement d'un système d'information personnalisé.

4.1. Présentation globale de la plate-forme d'agents choisie

De nombreuses plates-formes existent pour le développement d'applications à base d'agents logiciels. Le choix d'une plate-forme est généralement motivé par sa facilité de prise en main et par sa facilité de mise en œuvre des différents modèles d'agent. En effet, certaines caractéristiques relatives aux agents sont plus ou moins faciles à mettre en oeuvre suivant la plate-forme de développement utilisée. Par exemple, il est beaucoup plus aisé de développer des agents mobiles avec la plate-forme VOYAGER de Recursion Software[44] qu'avec la plate-forme JACK d'Agent Oriented Software[45] [Howden et al., 01] qui, elle, est bien adaptée pour le développement d'agents de type BDI (Belief-Desire-Intention).

Pour développer notre système de personnalisation, nous avons opté pour la plate-forme Magique (Multi-AGent hiérarchIQUE) [Routier et al., 01] de l'équipe Système Multi-Agents et Comportements (SMAC) du Laboratoire d'Informatique Fondamentale de Lille (LIFL)[46]. Ce choix a été établi non seulement grâce à la proximité de nos laboratoires respectifs ou parce que nous travaillons ensemble notamment dans le cadre des projets NIPO (Nouvelles Interactions Personnes-Organisations) et MIAOU (Modèles d'Interaction et Architectures Orientées Usages) mais aussi et surtout parce que certains principes de Magique, que nous détaillerons par la suite, nous sont apparus pertinents, pouvant faciliter la conception de notre système de personnalisation.

4.1.1. Définitions et concepts

La plate-forme Magique se présente sous forme d'une API[47] Java qui facilite la conception de système multi-agents. Magique se base sur un ensemble de concepts pour la construction d'un système multi-agents. Les définitions de ces concepts qui sont présentées dans cette partie sont issues de [Mathieu et Routier 01], [Routier et al., 01], [Secq 03] et de la rubrique consacrée à Magique de la page web de l'équipe SMAC. Magique se repose essentiellement sur le concept de *Compétence*.

[43] PlayStation Portable (http://www.yourpsp.fr)
[44] http://www.recursionsw.com/
[45] http://www.agent-software.com
[46] http ://www.lifl.fr/SMAC
[47] Application Program Interface

Définition 1 : *"une **compétence** désigne un ensemble cohérent de capacités"* [Routier et al., 01].

Une compétence précise donc les tâches que l'agent peut effectuer. Elle décrit une série d'activités que l'agent doit réaliser pour effectuer une opération précise. La compétence définit donc le comportement d'un agent. Les compétences peuvent être développées indépendamment de tout agent. Elles sont ensuite ajoutées aux agents pour définir leurs comportements. Par conséquent, les capacités de communication, de reproduction, de mobilité, d'apprentissage, etc., peuvent être associées à des compétences dont l'agent peut disposer ou non.

Au niveau de l'implémentation, une compétence correspond à une classe Java héritant de la classe *MagiqueDefaultSkill* fournie par l'API Magique dont les méthodes publiques correspondent aux tâches que l'agent peut effectuer et mettre au service des autres agents composant le système multi-agents.

Définition 2 : *"un **agent** est une entité douée de compétences"* [Routier et al., 01].

Un agent est donc une entité possédant un certain nombre de compétences. Ces compétences permettent aux agents de jouer un rôle au sein du SMA. Les compétences d'un agent peuvent évoluer dynamiquement (par apprentissage/acquisition) au cours de son existence.

Pour préciser cette définition les auteurs de Magique proposent la notion *d'agent atomique*.

Définition 3 : *"un **agent atomique** est une entité douée de deux compétences : une pour interagir et une pour apprendre de nouvelles compétences. Un agent est un agent atomique qui a appris des compétences au travers de communications"* [Routier et al., 01].

Les auteurs de Magique soulignent la nécessité d'une capacité d'interaction et d'apprentissage (de compétences) pour un agent logiciel. *« Sans la "compétence d'acquisition", un tel agent ne serait qu'une coquille vide incapable de faire quoi que ce soit. Sans la "compétence de communication", un agent est isolé et perd de ce fait tout intérêt »* [Secq 03].

Les agents Magique possèdent donc les deux compétences de communication *ConnectionSkill* et d'acquisition de compétence *AddSkillSkill.* Ils peuvent communiquer en asynchrone (*perform* / *ask*) et en synchrone (*askNow*).

L'une des particularités de Magique se situe au niveau des principes d'oubli et d'échange de compétences. Un agent peut oublier une compétence s'il juge, par exemple, que cette compétence est obsolète. Il peut aussi enseigner à un autre agent une compétence dont il dispose ou demander auprès d'un agent de lui enseigner une compétence pour pouvoir, par exemple, effectuer une tâche. Ces principes sont implémentés dans Magique sous forme de compétences et les agents Magique disposent de ces compétences, par défaut, à leur création.

Organisation : Magique utilise une structure organisationnelle hiérarchique. Les agents feuilles sont appelés « *spécialistes* » et les autres appelés « *superviseurs* ». Cette structure décrit les liens d'accointances et définit le support de communications entre les agents. Magique se base sur les relations d'accointances pour le routage par défaut des messages. La communication est donc essentiellement verticale. Cependant, il est toujours possible d'établir une relation directe d'accointances (donc de communication) entre les agents (voir Figure 4.1).

Magique se base sur la structure hiérarchique pour réaliser un mécanisme de délégation de tâche. Lorsqu'un agent doit exécuter une tâche et qu'il n'a pas la compétence nécessaire, il regarde s'il a une accointance particulière pour cette compétence et lui demande de réaliser la tâche pour lui. Sinon, il la fait exécuter par un membre de sa hiérarchie qui possède cette compétence. Sinon, il demande à son superviseur de lui chercher un agent compétent à ce sujet. Son superviseur re-applique ce mécanisme récursivement. Lorsque aucun agent du SMA ne dispose de la compétence adéquate, la requête est stockée au niveau de l'agent racine jusqu'à ce qu'un des agents formant le SMA acquière cette compétence pour exécuter la tâche.

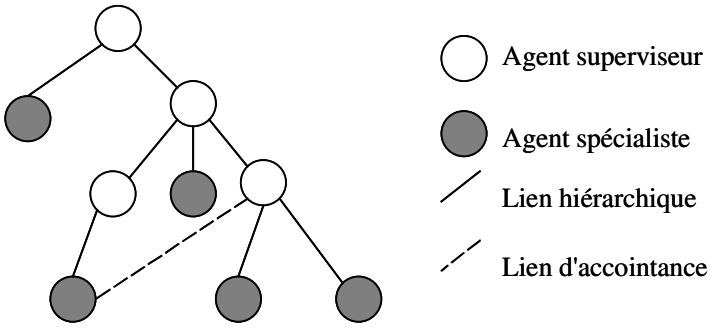


Figure 4.1. Structure organisationnelle hiérarchique

4.1.2. Conception et Interaction avec les agents

Un environnement graphique permettant la conception et le déploiement des agents Magique est disponible. Cet environnement permet la construction des agents par la définition des compétences dont doit disposer chaque agent, l'organisation hiérarchique de ces agents, la précision de la plate-forme d'accueil où ils doivent s'exécuter et leur déploiement. Toutes ces actions sont effectuées avant l'exécution effective des agents.

Cet environnement intègre un interpréteur de commande Java permettant l'interaction avec les agents en exécution (voir Figure 4.2). C'est une interaction textuelle et l'utilisateur voulant interagir avec les agents doit utiliser des commandes spécifiques. Par exemple, dans la figure ci-dessous, la commande [*print(agent.askNow("ps.magic-Coordinator","getName"));*] permet de demander à l'agent *ps.magic-Coordinator* son nom complet. L'agent a affiché que son nom complet est : *ps.magic-Coordinator:192.168.6.225:444.*

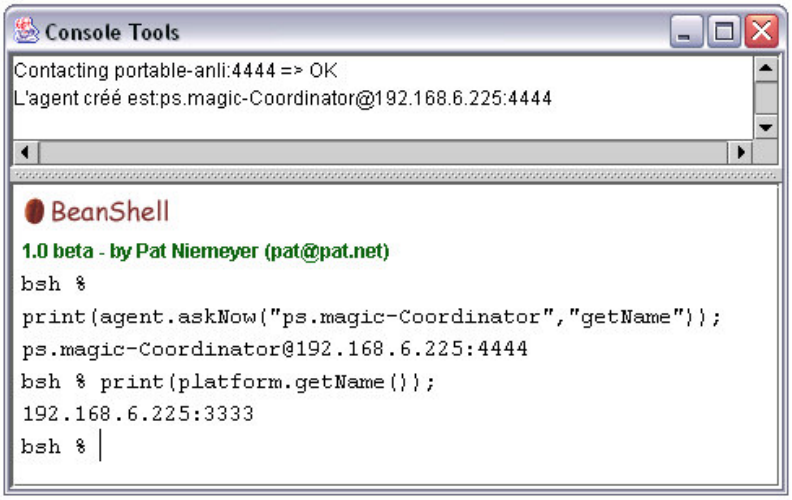


Figure 4.2. Outil d'interaction textuelle avec les agents

Aussi, il n'est pas possible d'utiliser cet outil (tel qu'il est fourni actuellement) pour interagir avec des agents qui ne soient pas lancés au travers de l'environnement graphique de Magique. Et la fermeture de cet environnement graphique interdit toute interaction ultérieure avec les agents créés auparavant.

4.2. Architecture générale et conception de PerSyst

Comme nous l'avons souligné dans les chapitres précédents, les deux caractéristiques essentielles que doit avoir le Système de Personnalisation (SP) sont : la possibilité de communication avec des applications externes (non nécessairement à base d'agents logiciels) et l'évolutivité. Il est

donc naturel que l'architecture de PerSyst comporte un agent permettant cette communication (agent de communication) et un agent permettant de gérer l'évolutivité du SMA formant le SP (agent d'administration). D'autres agents pourraient ensuite apparaître dans le SP suivant les besoins d'un projet (ces agents sont établis lors du déroulement de la partie SP de la méthode PerMet). Pour faire le lien entre les différents agents du SP, un autre agent (agent de coordination) a été défini. En effet, puisque les agents sont complètement autonomes et peuvent être localisés en différents point du réseau, il est nécessaire de disposer d'un référentiel qui permettra au développeur de localiser les agents et éventuellement d'interagir avec eux (faire évoluer leur compétence, changer leur localisation, etc.). Cet agent de coordination intervient aussi pour la transition des différents messages que les agents peuvent s'échanger pour répondre à un objectif global du SP. L'architecture générale de PerSyst est donc composée de différents agents [Anli et al., 05a]. La Figure 4.3 présente l'architecture générale de PerSyst et ses interactions avec les systèmes d'information existants.

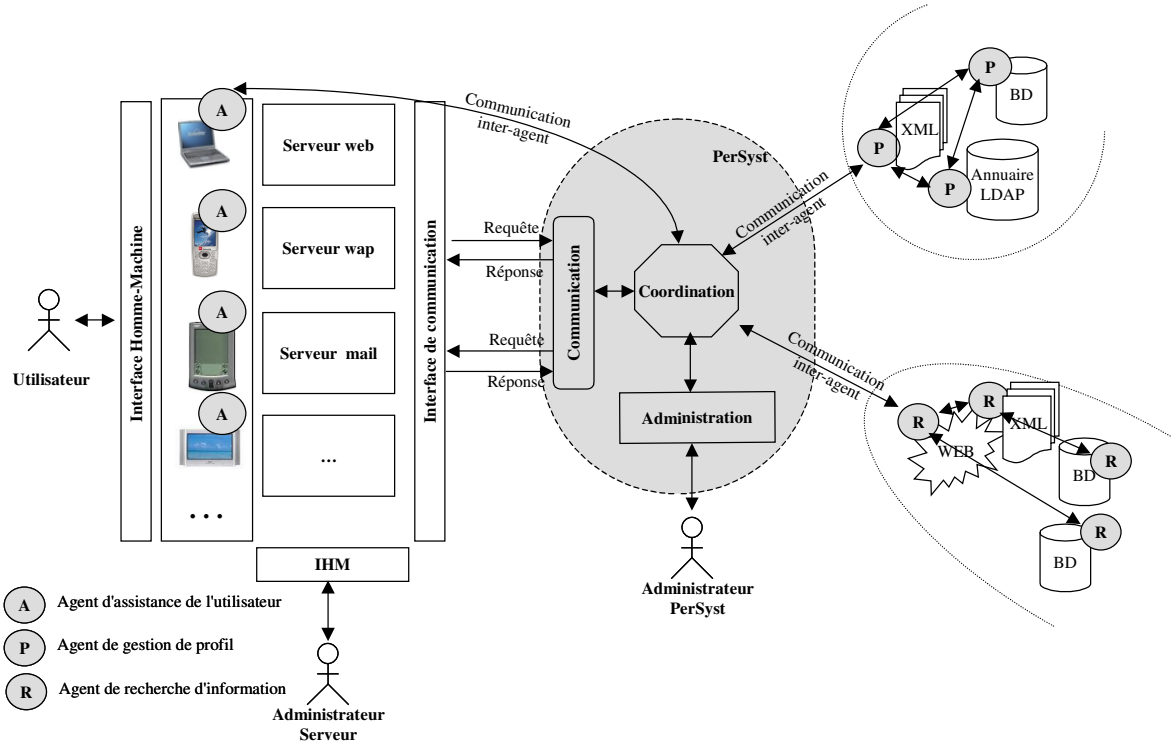


Figure 4.3. Architecture générale de PerSyst

Les trois agents de *communication*, de *coordination* et *d'administration* (ce sont les agents contenus dans l'ellipse à fond grisé) forment le noyau de PerSyst. PerSyst se base sur ces trois agents pour l'intégration de services personnalisés [Anli et al., 04a]. Les autres agents, que nous appelons *agents applicatifs* (A comme Assistant, P comme Profil, R comme Recherche), sont des exemples d'agents (ce sont les modèles d'agents les plus utilisés pour la construction de systèmes d'information personnalisés) qui pourraient être définis pour répondre à des objectifs précis suivant un projet particulier. Les comportements et les mécanismes de coordination de ces agents sont donc établis dans les phases d'analyse des modèles d'agent et de conception des comportements de la méthode PerMet (le lecteur pourra trouver des exemples d'applications utilisant ces modèles d'agents dans le chapitre 5). La seule contrainte que doivent respecter ces agents consiste à avoir un lien d'accointance avec l'agent de coordination pour qu'ils puissent être localisés pour des besoins d'évolutivité. Il n'est pas nécessaire que cela soit un lien d'accointance direct. Il suffit qu'il existe un « chemin d'accointances » reliant un agent avec l'agent de coordination. Par exemple, si un agent A1 a une relation d'accointance avec l'agent de coordination, il suffit qu'un agent A2 ait une relation d'accointance avec A1 pour que PerSyst puisse le localiser dans le système. Par défaut, les agents de PerSyst exploitent la structure organisationnelle hiérarchique pour retrouver ses différents agents (voir Figure 4.4).

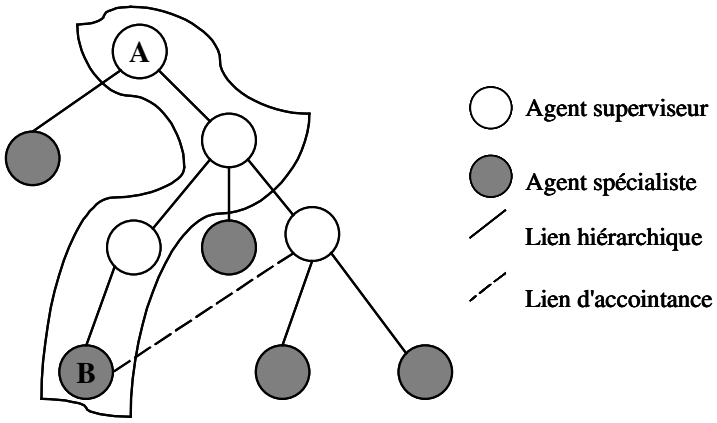


Figure 4.4. Exemple d'un chemin d'accointance reliant un agent A à un agent B

4.2.1. L'agent de coordination

L'agent de coordination permet de faire le lien de communication entre les différents agents composant PerSyst. L'agent de coordination assure trois rôles : (1) la coordination des tâches des agents applicatifs, (2) la coordination avec les applications externes et (3) la coordination avec l'utilisateur.

Coordination des tâches des agents applicatifs

L'agent de coordination coordonne les messages échangés entre les différents agents applicatifs. PerSyst préconise de ne gérer au niveau de cet agent que la coordination des messages échangés entre agents de domaine d'activités différentes. La coordination entre agents du même domaine d'activités est déléguée à un autre agent applicatif. Cela évite de surcharger l'agent de coordination et permet d'avoir une meilleure structuration du système (pour des besoins de maintenance). L'analyse des modèles d'agents permet de distinguer les différents domaines d'activités des agents. Un modèle d'agent est une abstraction d'un type d'agent effectuant les mêmes activités. Les agents issus d'un même modèle d'agent disposent donc des mêmes compétences. Ces modèles d'agent peuvent être organisés suivant une structure hiérarchique.

La Figure 4.5 décrit une organisation hiérarchique des modèles d'agent de recherche d'information transport. Un ou plusieurs agents peuvent être créés par modèle d'agent. Cette organisation hiérarchique des modèles d'agents ne reflète d'aucune manière l'organisation physique des agents issus de ces modèles. Tout dépend du modèle d'organisation effectif intégrés aux agents. Cependant cette organisation hiérarchique renseigne sur les domaines d'activités dont les agents issus de ces modèles appartiennent. Il suffit seulement de considérer la racine de l'arbre obtenu puisque tous les nœuds fils correspondent nécessairement à des modèles d'agent spécialisés. Par conséquent, la coordination des messages échangés entre les agents issus des modèles de la Figure 4.5 sera assurée par un de ces agents (si l'organisation physique des agents est une organisation hiérarchique, généralement, c'est le supérieur hiérarchique qui coordonne les échanges entre ses agents fils). Mais si des agents issus des modèles d'agent de recherche d'information transport (voir Figure 4.5) doivent coopérer avec des agents issus des modèles d'agent de gestion de profil (voir

Figure 4.6), la coordination entre ces agents se fera au niveau de l'agent de coordination. Par exemple, si un agent de recherche de perturbation (RP) Transpole (recherche d'information de perturbation dans l'agglomération lilloise) doit coopérer avec un agent de filtrage des perturbations (FP) pour informer les usagers des transports des perturbations éventuelles, la coordination entre ces deux agents se fait au niveau de l'agent de coordination. Les agent RP et FP n'ont pas besoin de se connaître mutuellement. Le but de RP est de rechercher les perturbations. Le but de FP est de filtrer les perturbations pour informer les usagers des perturbations qui les concernent. C'est l'agent de coordination qui coordonnera les messages de ces deux agents pour que le but global soit atteint (informer les usagers concernés des perturbations qui surviennent). La Figure 4.7 illustre la coordination entre les agents de recherche d'information et ceux de gestion de profil de l'utilisateur.

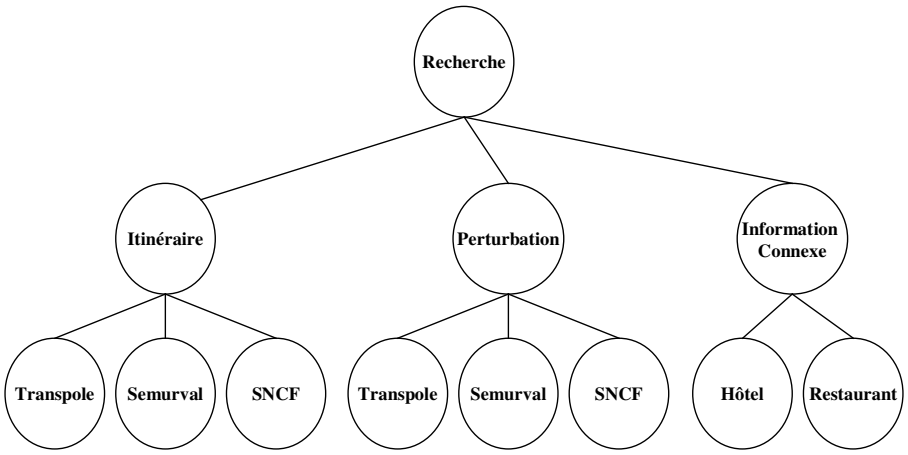


Figure 4.5. Organisation hiérarchique des modèles d'agent de recherche

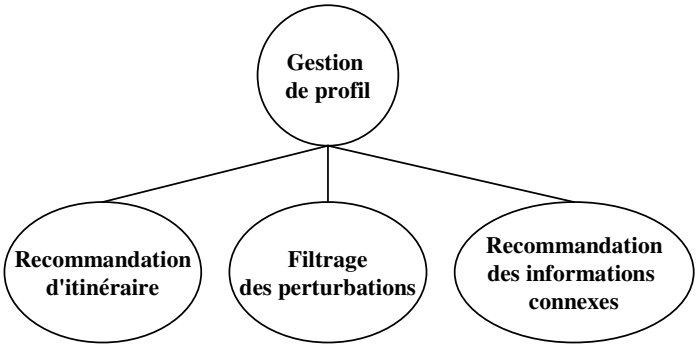


Figure 4.6. Organisation hiérarchique des modèles d'agent de gestion de profil

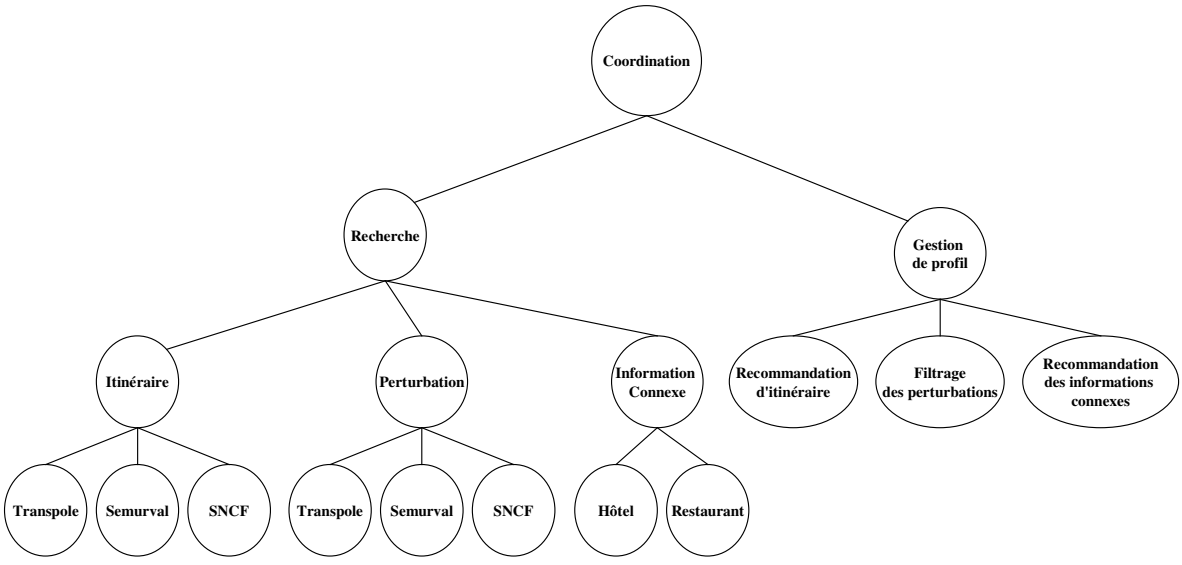


Figure 4.7. Coordination des agents de domaines d'activités différentes

Coordination avec les applications externes

Les messages échangés entre les applications externes et les agents applicatifs transitent par l'agent de coordination. Bien évidemment, une transformation de ces messages est effectuée (par l'agent de communication, qui sera décrit par la suite) pour que les messages envoyés par les applications externes soient compréhensibles par les agents logiciels et vice versa. C'est l'agent de coordination qui distribue les messages aux agents applicatifs concernés et renvoie les réponses aux applications externes.

Le diagramme d'activité de la Figure 4.8 donne un exemple d'échange de messages entre une application externe (un serveur web) et les agents de PerSyst. Dans cet exemple, PerSyst est composé de deux agents applicatifs (Recherche et Profil) et deux agents systèmes (Communication et Coordination). Lorsqu'un utilisateur lance une requête au niveau du serveur web, ce dernier demande à PerSyst l'information qui pourrait intéresser l'utilisateur. La demande du serveur web est traduite par l'agent de communication et arrive à l'agent de coordination. L'agent de coordination transmet le message à l'agent de recherche d'information et demande à l'agent de gestion de profil ce que pourrait préférer l'utilisateur. Ensuite, l'agent de coordination transmet la réponse personnalisée au serveur web qui va la fournir à l'utilisateur. Cette première partie de l'exemple correspond à une décomposition par l'agent de coordination de la requête initiale lancée par le serveur web. La requête a été traduite au niveau de l'agent de coordination en une requête destinée à l'agent de recherche et à une autre à l'agent de profil. Supposons que l'utilisateur choisisse une information. Le serveur web transmet cette réponse à PerSyst. Lorsque la réponse arrive à l'agent de coordination, celui-ci la transmet à l'agent de profil (puisque c'est l'agent de profil qui gère les préférences de l'utilisateur).

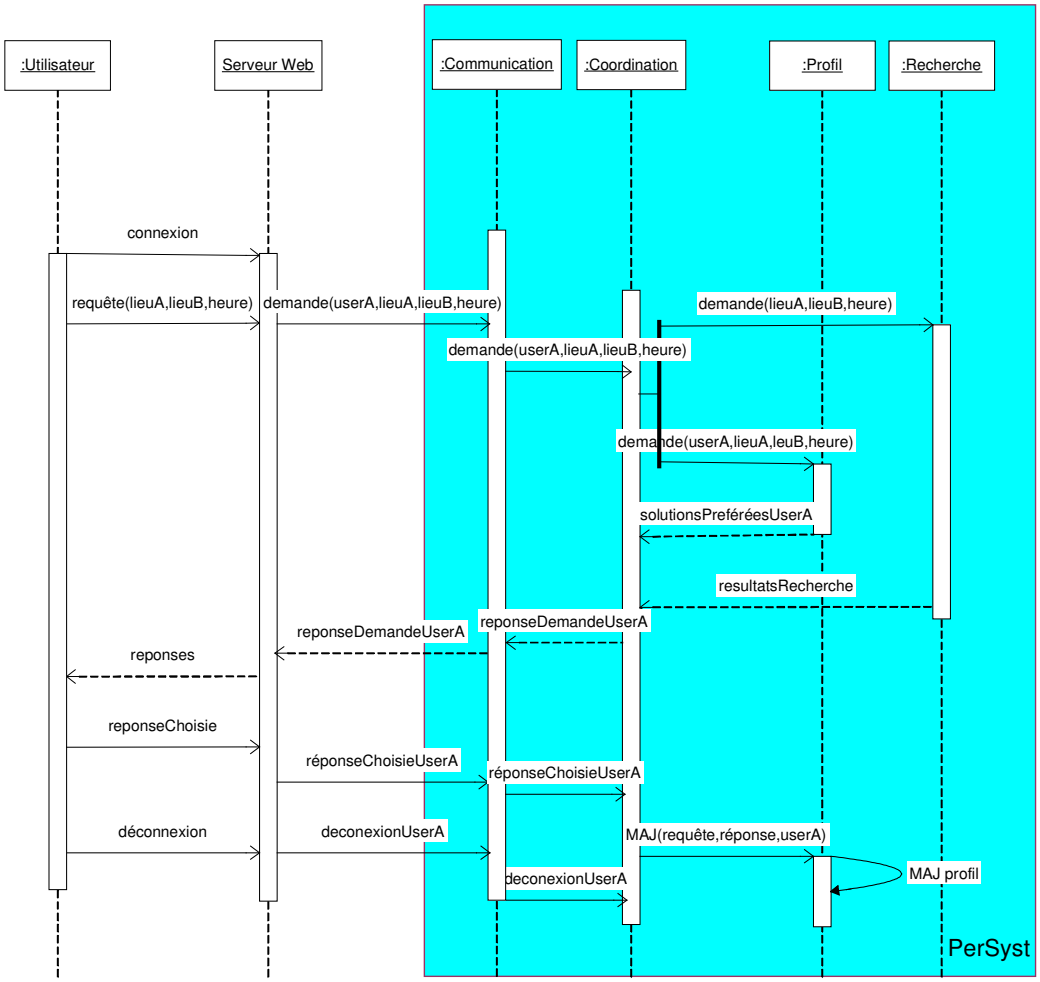


Figure 4.8. Coordination avec les applications externes

Coordination avec l'utilisateur

Les messages échangés entre l'utilisateur[48] et les agents applicatifs transitent par l'agent de coordination. Les requêtes de l'utilisateur sont interprétées par l'agent d'administration (que nous verrons par la suite) puis transmises à l'agent de coordination qui se charge d'envoyer la requête à l'agent concerné. La transmission des messages de l'utilisateur est assurée grâce la compétence *AdministratorSkill* dont dispose l'agent de coordination (voir Figure 4.9). Cette compétence fournit des services permettant de transmettre à un agent particulier des requêtes d'administration. Par exemple, le service *public Team getChilds(String agentName)* permet d'obtenir la liste des agents fils (dans une organisation hiérarchique) d'un agent donné ou le service *public void addSkill(String agentName, String skill, String teachAgent)* permet de dire à un agent (*agentName*) d'apprendre une compétence (*skill*) dont dispose un autre agent (*teachAgent*).

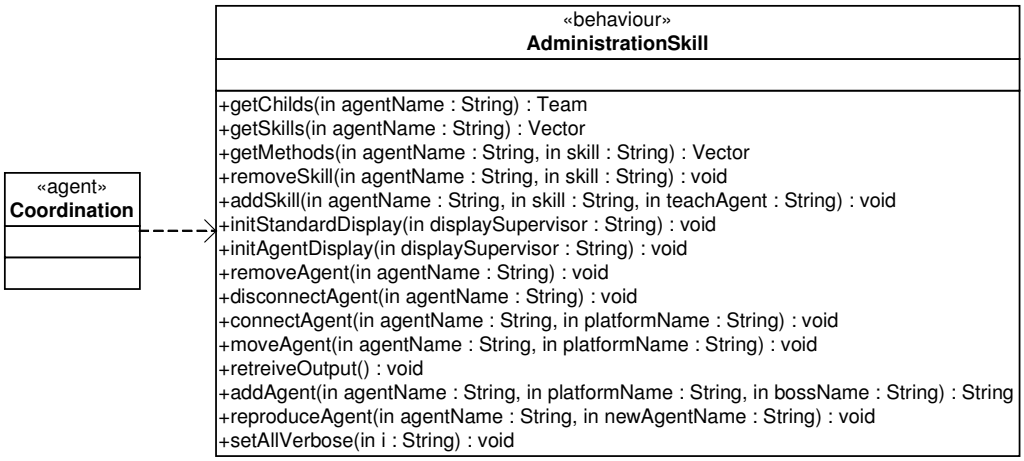


Figure 4.9. La compétence AdministrationSkill

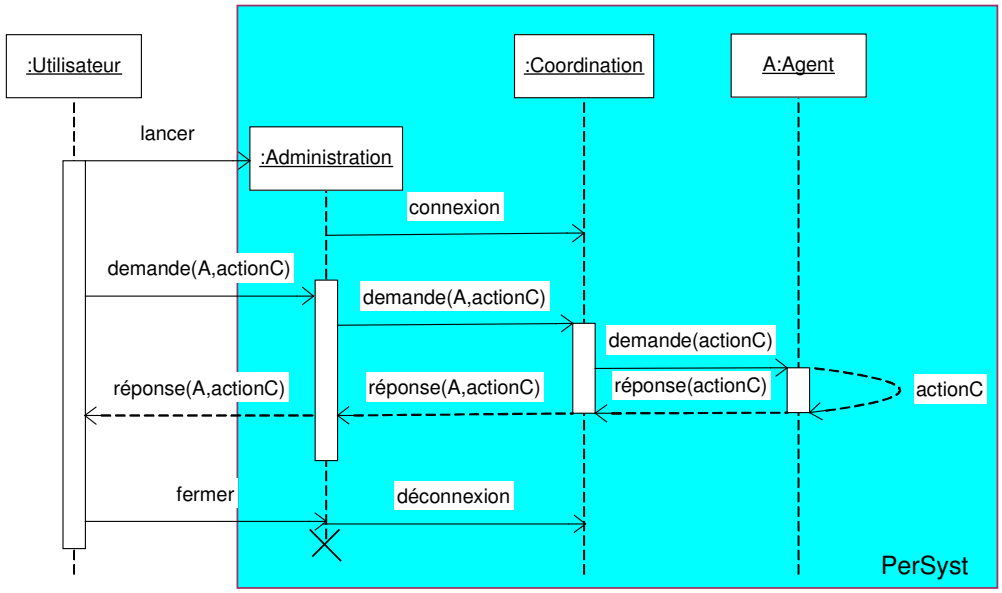


Figure 4.10. Coordination avec l'utilisateur

[48] Ici, l'utilisateur est un développeur.

La Figure 4.10 illustre la manière dont les requêtes de l'utilisateur sont transmises à un agent de PerSyst[49]. Dans cet exemple, l'utilisateur lance l'agent d'administration pour demander à un agent A de faire une action C. A son lancement, l'agent d'administration se connecte à l'agent de coordination pour pouvoir communiquer. Lorsque l'utilisateur demande à l'agent d'administration la réalisation de l'action C par l'agent A, cette requête est transmise à l'agent de coordination qui la renvoie à l'agent A. Les réponses sont transmises à l'utilisateur en suivant le chemin inverse.

4.2.2. L'agent de communication

L'agent de communication permet la traduction des requêtes envoyées par les applications externes en des messages compréhensibles par l'agent de coordination et vice versa. Cet agent diffère des autres agents de PerSyst. En effet, même si cet agent intègre des aspects de Magique (il utilise l'API Magique pour envoyer les messages à l'agent de coordination), son fonctionnement et sa structure interne sont différents de ceux d'un agent Magique. La Figure 4.11 présente l'architecture de l'agent de communication et ses interactions avec les applications externes et avec l'agent de coordination. L'agent de coordination se décompose en deux parties : la première partie appelée *ApplicationCommunicator* se charge de la communication avec les applications externes et la deuxième partie appelée *MagiqueCommunicator* s'occupe de l'interaction avec l'agent de coordination.

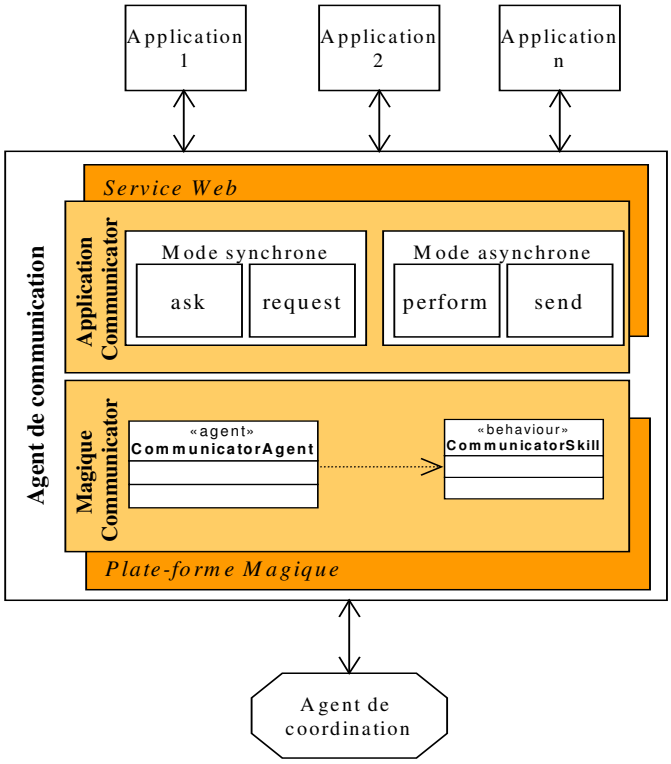


Figure 4.11. Architecture de l'agent de communication

La partie ApplicationCommunicator

La partie ApplicationCommunicator utilise le protocole SOAP (Simple Object Access Protocol) [Kadima et Monfort 03] pour l'interaction avec les applications externes qui peuvent être écrites avec n'importe quel langage de programmation. Elle fournit quatre primitives de communication (*ask*, *request*, *perform* et *send*) exposées sous forme de services web [Monfort et Goudeau 04]. Les paramètres d'entrée/sortie de ces services sont des chaînes de caractères. En effet,

[49] Nous utilisons «agent de PerSyst» pour dire que l'agent fourni ses compétences au service de PerSyst. L'agent de coordination a donc un chemin d'accointance avec cet agent permettant leur communication.

actuellement, il n'existe pas d'équivalence systématique entre les types d'objet des différents langages de programmation. Un objet XML Java, par exemple, ne sera pas reconnu lors de la communication SOAP comme un objet XML C#. Seuls les types de bases (entier, réel, chaîne de caractère, etc.) sont relativement bien reconnus par la majorité des langages de programmation. Pour cela, nous avons opté pour les chaînes de caractères sachant que le but, ici, est seulement de transférer des messages (donc pouvant être encodés sous forme de chaînes de caractères). Ces messages peuvent être structurés sous le format XML. La Figure 4.12 donne un exemple de message pouvant être envoyé par une application externe. Dans cet exemple, l'application externe fournit à PerSyst le « *distinguished name* » pour référencer un objet (dans cet exemple, c'est pour identifier un utilisateur) enregistré dans un annuaire LDAP (Lightweight Directory Access Protocol) [Rizcallah 00] et une requête de recherche d'itinéraire comportant les lieux de départ et d'arrivée ainsi que la date et l'heure de départ.

```
- <REQUEST>
    <DN>LDAP://SRV-ETCL-
      DC01:389/cn=john,OU=Sitp,DC=DOMTRANSPORT,DC=local</DN>
  - <Request>
      <Departure>LAMIH</Departure>
      <Arrival>Archimed</Arrival>
      <DepartureTime>07:00</DepartureTime>
      <Date>12/07/2006</Date>
    </Request>
  </REQUEST>
```

Figure 4.12. Exemple de message envoyé par une application externe à l'agent de communication

Deux des quatre primitives permettent une communication synchrone et les deux autres permettent une communication asynchrone.

- **Communication synchrone :** l'application appelante attend une réponse de la part de PerSyst pour poursuivre son exécution. Les interfaces fournies sont :

  - public String ask(String questionName); cette primitive peut être utilisée par l'application externe lorsqu'elle veut poser une question. La question est spécifiée au travers du paramètre d'entrée « questionName ». Par exemple, supposons qu'un agent de PerSyst fournit un service « getUsers » permettant de renvoyer la liste des utilisateurs inscrits au système. Les applications externes peuvent demander la liste des utilisateurs inscrits en utilisant la primitive de la manière suivante *result=object.ask("getUsers")*[50].

  - public String request(String serviceName, String param); cette primitive assure les mêmes fonctionnalités que la précédente sauf qu'elle autorise l'appel de services des agents comportant un paramètre d'entrée. Par exemple, l'instruction *result=object.request ("searchItinerary",param);* avec *param* une variable contenant la chaîne de caractères de la Figure 4.12 par exemple, permet à une application de solliciter le service "searchItinerary" d'un agent en lui passant le paramètre *param*. Bien évidemment, l'agent de communication transmet d'abord le message à l'agent de coordination qui va se charger d'envoyer la requête à l'agent adéquat.

- **Communication asynchrone :** l'application appelante n'attend pas de réponse de la part de PerSyst. Elle envoie l'information et poursuit son exécution. Les interfaces fournies sont :

  - public void perform(String eventName); cette primitive peut être utilisée par l'application externe pour déclencher un événement à un agent de PerSyst sans attendre de réponse en retour. Par exemple, activer des agents, signaler le lancement d'une application, signaler l'arrêt d'une application, etc.

[50] La syntaxe varie selon le langage de programmation utilisé. L'exemple fourni est écrit en Java.

- public void send(String msgName, String msg); cette primitive assure les mêmes fonctionnalités que la précédente sauf qu'elle est utilisée pour déclencher des événements nécessitant un paramètre d'entrée. Comme pour la primitive *request*, lorsqu'il y a plusieurs paramètres, ils peuvent être structurés selon le format XML. Envoyer le choix de l'utilisateur, informer la nature de la plate-forme d'interaction de l'utilisateur, envoyer la localisation de l'utilisateur, etc., sont des exemples de cas pouvant nécessiter l'utilisation de cette primitive.

Ces primitives permettent de récupérer les messages des applications externes pour les envoyer à la partie MagiqueCommunicator.

La partie MagiqueCommunicator

La partie MagiqueCommunicator récupère les messages issus de la partie ApplicationCommunicator pour les traduire en des requêtes compréhensibles par des agents Magique. Concrètement, elle instancie une plateforme Magique contenant un agent Magique nommé *CommunicatorAgent*. Cet agent a la capacité de se connecter à l'agent de coordination de PerSyst grâce à la compétence *CommunicatorSkill* (voir Figure 4.13) pour pouvoir communiquer avec lui.

Cette partie est fournie sous forme d'une API Java pouvant être importée dans une application Java indépendamment de PerSyst. Ainsi lorsqu'une application Java a besoin de communiquer avec un agent Magique, il suffit de créer un objet de la classe *MagiqueCommunicator* (voir Figure 4.14). L'application pourra ensuite invoquer des services d'agents Magique par invocation des méthodes de l'objet créé. La Figure 4.15 donne un exemple d'utilisation de *MagiqueCommunicator* dans une application Java. Cet exemple illustre le cas d'une application qui pose la question «howAreYou» à l'agent Magique nommé «anliAgent» se trouvant sur le pc «univ-valenciennes.fr» en utilisant le port de communication RMI 1355[51].

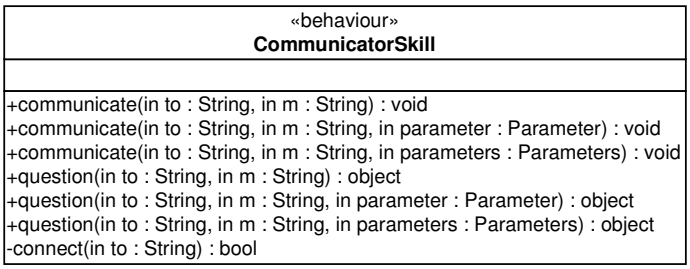


Figure 4.13. La compétence CommunicatorSkill

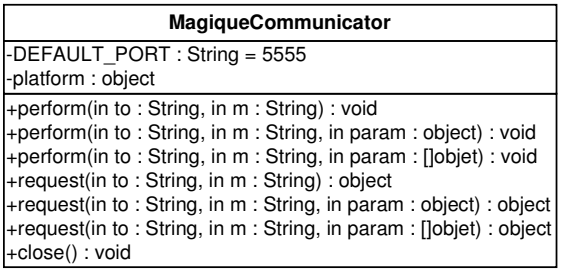


Figure 4.14. La classe MagiqueCommunicator

```
MagiqueCommunicator mc = new MagiqueCommunicator();
String response=(String)mc.request("anliAgent@univ-valenciennes.fr:1355",
                                   "howAreYou");
```

Figure 4.15. Un exemple d'envoi de message à un agent Magique utilisant la classe MagiqueCommunicator

[51] Magique utilise le protocole Remote Method Invocation (RMI) pour l'envoi des messages entre les agents.

4.2.3. L'agent d'administration

L'agent d'administration permet la gestion et l'administration de PerSyst. Son rôle principal est de permettre à un utilisateur (dans le cas de PerSyst, un développeur) d'interagir avec les agents logiciels pour l'adaptation et la configuration de PerSyst. Son rôle semble donc être le même que celui de l'outil fourni par la plate-forme Magique. Mais comme évoqué plus haut (cf. §4.1.2), l'outil fourni par Magique permet seulement une interaction textuelle avec les agents. De plus, cet outil doit rester toujours actif pour continuer à assurer le lien de communication entre l'utilisateur et les agents. Si l'outil est arrêté (fermeture de l'application, redémarrage de la machine, etc.), il n'est plus possible d'interagir avec les agents même en relançant l'outil. Ce dernier point constitue une contrainte forte rendant incompatible l'utilisation de cet outil par PerSyst. En effet, il suffirait que la machine sur laquelle est lancé l'outil d'administration tombe en panne ou soit arrêtée par inadvertance pour qu'il n'y ait plus aucune possibilité d'évolution de PerSyst. En plus, l'utilisateur serait liée à une même machine (là où est lancé l'outil d'administration) sans possibilité d'en changer.

Au vu de toutes ces contraintes imposées par l'outil existant, PerSyst fournit son propre outil d'administration d'agents. Cet outil se présente sous forme d'un agent Magique disposant des compétences nécessaires permettant l'administration et la gestion des agents. La Figure 4.16 présente le diagramme de classes de l'agent d'administration de PerSyst nommé *PSManager*.

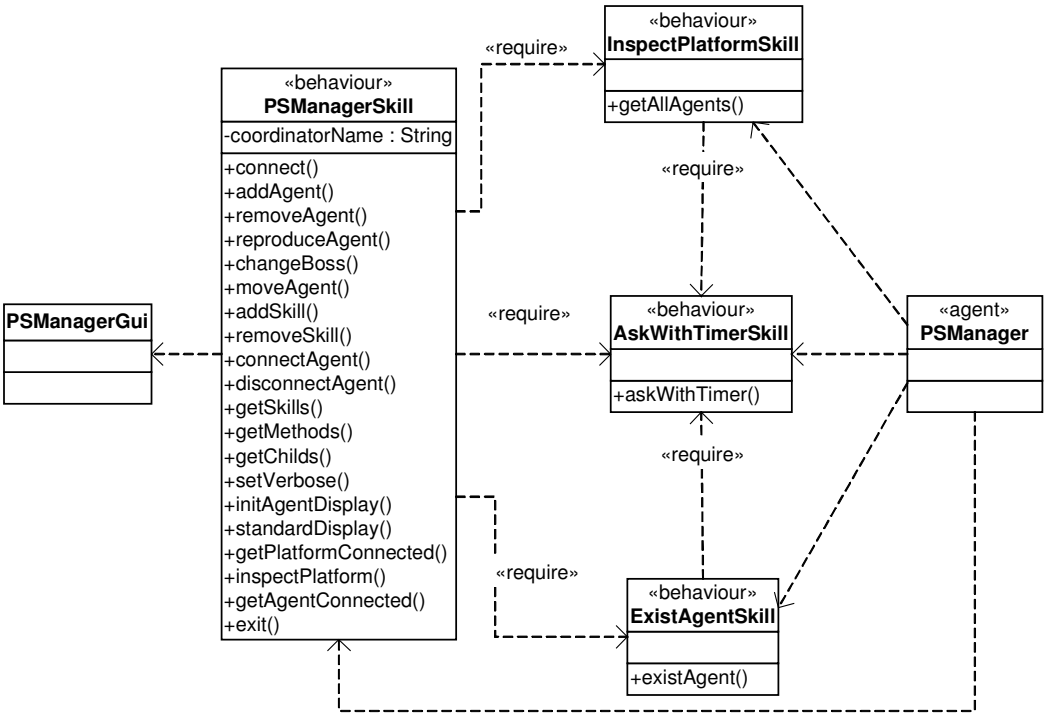


Figure 4.16. Le diagramme de classes de l'agent d'administration

La compétence *InspectPlatformSkill* permet à l'agent d'administration de connaître les agents présents dans une plate-forme donnée. La compétence *AskWithTimerSkill* lui permet d'envoyer des requêtes avec attente de réponse sans être bloqué indéfiniment au cas où il ne reçoit pas de réponse. La compétence *ExistAgentSkill* lui permet de savoir si tel agent existe. Et la compétence *PSManagerSkill* fournit des services pour la gestion des agents (création, suppression, reproduction d'un agent), pour la manipulation de leur topologie (organisation, distribution), pour l'administration de leurs compétences (ajout, suppression de compétences) et pour la (dé)connexion des agents (connexion, déconnexion). *PSManagerSkill* implémente d'autres services comme l'affichage des messages de trace des agents ou la définition du niveau de trace des agents. *PSManagerSkill* est associée avec une interface graphique (implémentée par la classe *PSManagerGui*) permettant à l'utilisateur d'interagir avec les agents (voir la Figure 4.17).

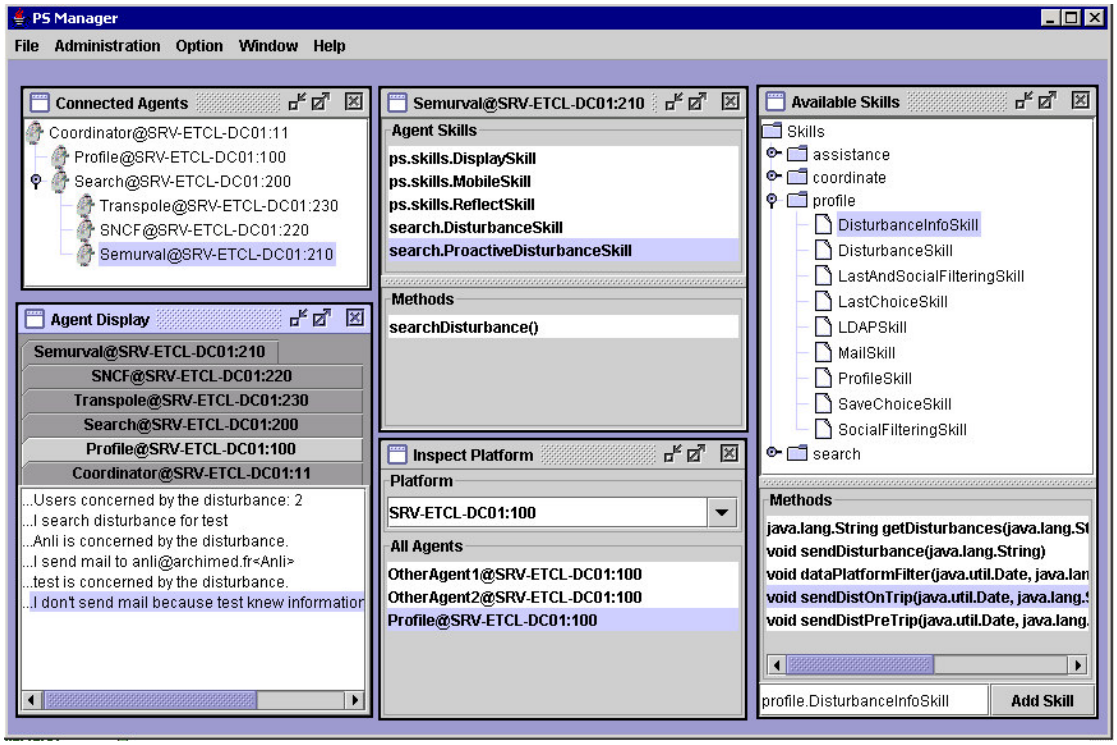


Figure 4.17. L'agent d'administration de PerSyst

L'administrateur dispose d'une vue sur les agents actifs, sur leur organisation et sur leur localisation (fenêtre *Connected Agents*), sur les compétences (*Agent Skills*) et sur le contenu de leurs échanges (*Agent Display*). Il peut créer et ôter des agents, ainsi qu'agir sur chacun d'eux pour lui supprimer ou lui ajouter une des compétences disponibles (*Available Skills*). Il a aussi la possibilité de connecter un agent existant (*Inspect Platform*) ou de l'enlever de PerSyst.

Les sous-sections suivantes décrivent les fonctionnalités fournies par l'agent d'administration. Notons que cet agent pourrait s'utiliser indépendamment de PerSyst et pourrait contribuer à l'amélioration de l'environnement de conception de Magique par la proposition d'un outil d'interaction graphique avec les agents Magique.

4.2.3.1. Création, suppression, reproduction d'un agent

Création d'un agent : la création d'un agent s'effectue au travers de l'interface présentée par la Figure 4.18. L'utilisateur spécifie le nom de l'agent, la plate-forme Magique[52] sur laquelle il va s'exécuter et le supérieur hiérarchique de l'agent.

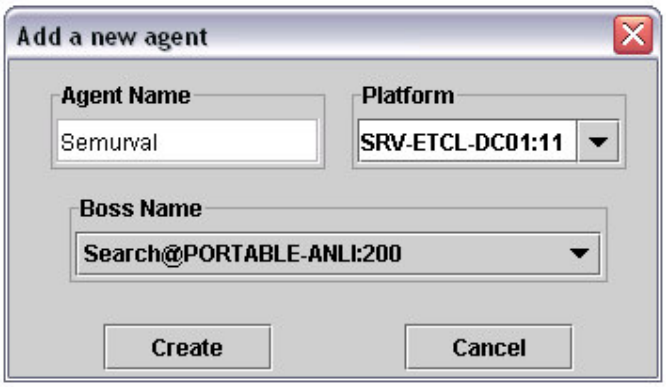


Figure 4.18. Ajout d'un agent

Suppression d'un agent : la suppression d'un agent consiste à l'enlever physiquement de sa plate-forme d'exécution. Pour supprimer un agent, il suffit de sélectionner l'agent dans la fenêtre *Agent Connected* (Figure 4.17) et de le supprimer par le menu contextuel (ou par menu général). Lorsqu'un agent superviseur est supprimé, ses agents spécialistes sont automatiquement déconnectés de PerSyst (la déconnexion est décrite par la suite).

Reproduction d'un agent : la reproduction d'un agent permet de créer un agent ayant les mêmes compétences qu'un agent déjà existant. C'est une *reproduction de redondance* [Cardon 00] ; une reproduction de redondance est généralement utilisée pour garantir la robustesse d'un système. Dans le cas de PerSyst, cela permet aussi de créer rapidement un agent ayant des compétences identiques d'un autre déjà existant. Des compétences sont ensuite ajoutées ou retirées pour spécialiser le nouvel agent. La reproduction crée l'agent dans la même plate-forme et l'agent a le même superviseur que son géniteur. L'agent doit avoir la compétence de reproduction (*ReproduceSkill*, voir Figure 4.19) pour qu'il puisse se reproduire. C'est l'utilisateur qui lance le processus de reproduction au travers d'une boîte de dialogue. D'autres types de reproduction[53] pourraient être envisagés en ajoutant aux agents les compétences adéquates. Par exemple, les agents pourraient se reproduire dynamiquement en utilisant des algorithmes génétiques [Goldberg 94] pour améliorer leur adaptabilité et leur performance [Bousbia et al., 05].

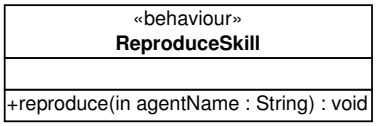


Figure 4.19. La compétence de reproduction

[52] Une plate-forme Magique est un couple (nom de la machine, numéro de port de communication). Il est donc possible d'avoir plusieurs plate-forme d'agent dans une même machine en variant le port de communication.
[53] Voir [Cardon 00] pour une revue des types de reproduction d'agents.

4.2.3.2. Ajout, suppression des compétences

Ajout de compétences à un agent : pour ajouter une compétence à un agent, il suffit de sélectionner l'agent dans la fenêtre *Agent Connected* et de lui enseigner une compétence disponible dans la fenêtre *Available Skills* (voir Figure 4.17).

Suppression de compétences d'un agent : pour supprimer une compétence d'un agent, il suffit de sélectionner l'agent dans la fenêtre *Agent Connected* et de supprimer, au travers du menu contextuel (ou du menu général), la compétence sélectionnée dans la fenêtre *Agent Skills* (Figure 4.17).

4.2.3.3. Organisation, distribution des agents

Organisation des agents : il s'agit d'une organisation hiérarchique. Le principe consiste à lier hiérarchiquement un agent (qui doit être selectionné de la fenêtre *ConnectedAgents*) à un autre spécifié dans la boîte de dialogue *Change Boss Agent* (voir Figure 4.20). Pour prendre en compte d'autres modèles d'organisations [Mandiau et al., 02], il est nécessaire de les spécifier au travers de compétences à ajouter aux agents.

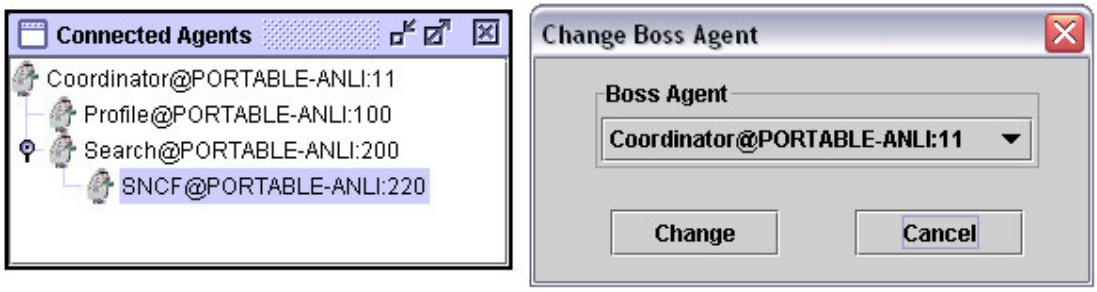


Figure 4.20. Changement de l'organisation du système multi-agents

Distribution des agents : la distribution consiste à envoyer les agents dans des plates-formes localisées sur le réseau. Le lieu où l'agent doit aller est spécifié au travers d'une boîte de dialogue. Pour qu'un agent puisse être déplacé, il doit disposer de la compétence *MobileSkill* (voir Figure 4.21). La Figure 4.22 décrit les activités effectuées par un agent lors de l'exécution de la tâche *move*. La mobilité de l'agent est *faible*[54] car seul le code et l'état de l'agent est migré : l'agent recommence son cycle d'exécution. Avant de se détruire, l'agent s'assure donc d'avoir terminé toutes les tâches initiées dans l'ancienne plate-forme.

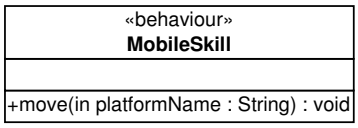


Figure 4.21. La compétence pour la mobilité des agents

[54] par opposition à une mobilité *forte* dont le code, l'état et le contrôle de l'état est migré à la plate-forme de destination. L'agent reprend son exécution au sein de la plate-forme de destination exactement au niveau où il était avant sa migration.

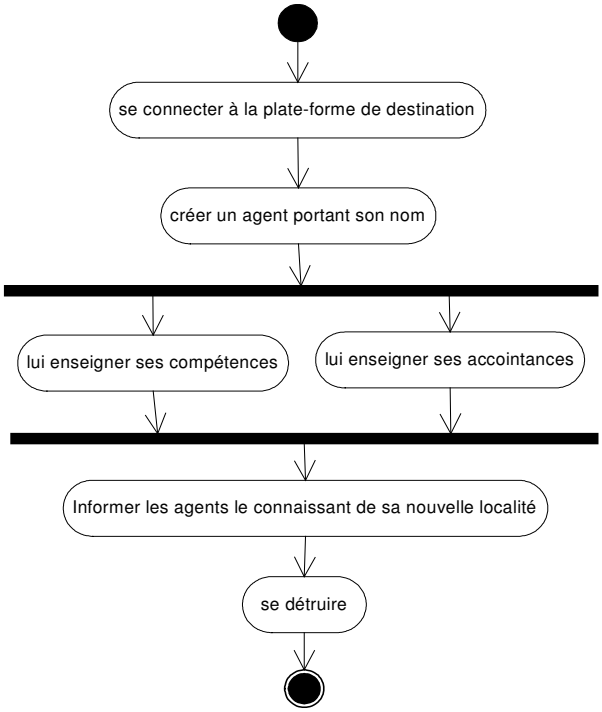


Figure 4.22. Activités pour le déplacement d'un agent dans le réseau

4.2.3.4. Connexion, déconnexion des agents

Connexion des agents : la connexion d'un agent consiste à inviter un agent existant à se connecter à PerSyst pour participer à la réalisation d'une tâche. Ainsi, n'importe quel agent Magique est potentiellement connectable à PerSyst. Pour connecter un agent, l'utilisateur choisit le nom de l'agent et le superviseur auquel il doit être connecté (voir Figure 4.23).

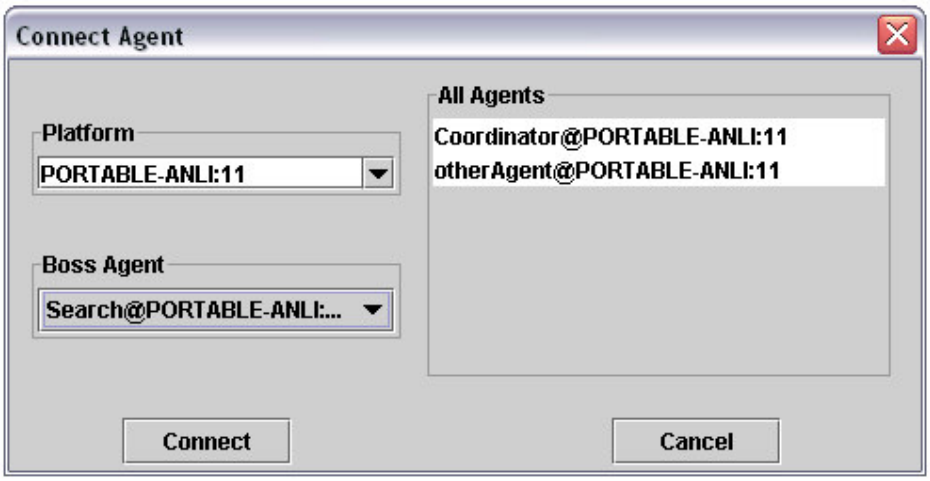


Figure 4.23. Connexion d'un agent

Déconnexion des agents : la déconnexion d'un agent consiste à l'enlever de PerSyst. Pour déconnecter un agent, il suffit de sélectionner l'agent dans la fenêtre *Agent Connected* (Figure 4.17) et de le déconnecter par le menu contextuel (ou par menu général). L'agent n'agira plus pour PerSyst. Cependant, il sera toujours présent dans le réseau informatique et pourra rejoindre un autre système multi-agents ou revenir plus tard rejoindre le système multi-agents composant PerSyst.

4.3. Modèles pour le développement de systèmes d'information personnalisés

Afin de faciliter le développement de systèmes d'information personnalisés, nous proposons des modèles pour la personnalisation de l'interaction homme-machine. Ces modèles pourraient être adaptés suivant le domaine d'application pour être intégrés à PerSyst. L'objectif, à terme, étant de proposer ces modèles sous forme de compétences génériques pouvant être apprises par les agents logiciels. Ces compétences pourraient ensuite être configurées pour répondre à un domaine spécifique d'application (ceci fait l'objet de travaux ultérieurs, voir les perspectives de recherche). Actuellement, les compétences disponibles dans PerSyst sont spécifiques pour la personnalisation des itinéraires dans les transports terrestres de personnes [Anli et Abed 05].

Nous proposons des modèles pour la représentation du profil de l'utilisateur et pour la collecte d'information sur l'utilisateur. Des modèles et des méthodes sont aussi proposés pour exploiter le modèle de l'utilisateur pour la personnalisation de l'interaction.

4.3.1. Modélisation de l'utilisateur

Nous répartissons le profil de l'utilisateur selon trois catégories d'informations : les données statiques, les données pondérées et l'historique [Anli 02]. Cette décomposition a pour but de faciliter le maintien des profils en séparant les données peu modifiées (les données statiques) des données susceptibles d'être modifiées régulièrement (les données pondérées), l'historique gardant une trace des comportements de l'utilisateur. Elle rejoint l'approche de [Bouzeghoub et Kostadinov 04] qui vise à catégoriser les informations de l'utilisateur.

Formellement, en considérant U l'ensemble des utilisateurs d'un système d'information personnalisé et $u \in U$ un utilisateur; le profil de l'utilisateur u, noté M_u se définit par :

$$M_u = S_u \cup P_u \cup H_u \quad \text{[Petit-Rozé et Grislin-Le Strugeon 05]} \quad (4.1)$$

où

- S_u est l'ensemble des données statiques
- P_u est l'ensemble des données pondérées
- H_u est l'ensemble de l'historique de l'utilisateur

Les données statiques : représentent l'ensemble des connaissances qui décrivent directement l'utilisateur et qui peuvent contribuer à la personnalisation. Ces données peuvent concerner les différents types d'informations tels que l'authentification (identifiant, mot de passe, empreintes, etc), les informations personnelles (nom, courriel, âge, sexe, etc.) et les informations de localisation «statique»[55] (adresse de domicile, lieu de travail, etc.) et des informations générales comme la profession de l'utilisateur, ses hobbies, ses cartes de réduction, etc. Ce sont des connaissances qui sont rarement modifiées.

Les données statiques sont représentées par un ensemble de couples de valeurs :

$$S_u = \{(e_1, v_1), (e_2, v_2), ..., (e_n, v_n)\}_{n \in IN} \quad (4.2)$$

où

- e_i (avec $1 \leq i \leq n$) est le type d'information sur l'utilisateur
- $v_i = \{c_1, c_2, ..., c_n\}_{n \in IN}$ est un ensemble de constantes (numériques ou alphanumériques) associées au type d'information e_i

Par exemple, pour le type d'information e_i = *«cartes de réduction»*, v_i = {réduction trains SNCF, réduction bus Valenciennes}

[55] Par opposition aux informations de géo-localisation temps réel (par le biais du GPS, par exemple).

Les données pondérées : sont les données susceptibles d'évoluer régulièrement. Ce sont les informations représentant les préférences de l'utilisateur.

Les données pondérées sont représentées en un ensemble de couples :

$$P_u = \{(p_1, \alpha_1), (p_2, \alpha_2), ..., (p_n, \alpha_n)\}_{n \in IN} \quad (4.3)$$

où

- p_i est le type de préférence
- α_i est un ensemble de données associées au type de préférence p_i. L'ensemble α_i dépend des types de préférences à modéliser. Cela peut aller d'une simple valeur numérique à un ensemble de valeurs alphanumériques.

Par exemple, pour le type de préférence p_i = « préfère la marche ». Les préférences associées à p_i pourraient être, par exemple, une valeur numérique α_i fonction d'un contexte [Poulon et Vaudry 03][Zimmermann et al., 05] dépendant du quadruple (température, humidité, état physique, localité) :

$$\alpha_i : T \times H \times E \times L \longrightarrow IR$$

$$(t,h,e,l) \longmapsto \alpha_i(t,h,e,l)$$

où

- *T, H, E, L,* représentent respectivement l'ensemble des températures, d'humidité, d'états physique et des localités
- *R* est l'ensemble des réels

Des exemples de valeurs de préférences pourraient être :

$\alpha_i(20°,20\%,bon,valenciennes)=9$; $\alpha_i(10°,100\%,bon,valenciennes)=3$;
$\alpha_i(20°,20\%,malade,valenciennes)=2$; etc.

L'historique : l'historique garde une trace des interactions de l'utilisateur avec le système d'information personnalisé. Cela peut servir de base de connaissance pour la mise à jour des données statiques et des données pondérées. L'analyse de l'historique pourrait, par exemple, renseigner le système que l'utilisateur habite à Valenciennes et travaille à Lille (car l'utilisateur part le matin à Lille et revient le soir à Valenciennes sauf les jours fériés où il reste à Valenciennes). Et en analysant les itinéraires choisis par l'utilisateur, le système pourrait déduire les préférences de l'utilisateur par rapport aux modes de transports.

L'historique est représenté selon un ensemble de couples de valeurs :

$$H_u = \{(h_1, \beta_1), (h_2, \beta_2), ..., (h_n, \beta_n)\}_{n \in IN} \quad (4.4)$$

où

- h_i est le type d'action (but, objectif, requête, etc.) souhaitée. Bien évidemment, cela peut être des types d'actions relatives aussi bien au contenant (agencement des composants de l'interface, plate-forme d'interaction, mode d'interaction, etc.) qu'au contenu (information demandée).
- β_i est un ensemble de données associée au type d'action h_i. L'ensemble β_i dépend des types d'action à modéliser. Cela peut aller d'une simple valeur numérique à un ensemble de valeurs alphanumériques.

L'historique enregistre seulement les comportements de l'utilisateur dont ce dernier dispose de plusieurs alternatives. Par exemple, le comportement «l'utilisateur a utilisé la fonctionnalité de recherche d'information simplifiée» sera enregistré si l'utilisateur dispose d'une autre manière pour «rechercher une information». Sinon ce comportement ne sera pas enregistré car ce serait une

information inutile (puisqu'il n'existe aucune base de comparaison pour déduire une préférence) ne pouvant pas servir pour la personnalisation.

4.3.2. Collecte d'information sur l'utilisateur

Dans cette partie nous proposons une méthode d'apprentissage automatique permettant de collecter les préférences d'un utilisateur. Notre approche constitue une contribution aux méthodes de collecte implicite d'information sur l'utilisateur. Notre objectif est d'analyser l'historique de l'utilisateur pour mettre à jour les données pondérées (donc ses préférences). La méthode d'acquisition de l'historique de l'utilisateur n'est pas précisée ici (cela pourrait se faire par exemple, par des logs applicatifs, voir chapitre 1).

Dans ce qui suit, nous supposons que le profil de l'utilisateur est décrit suivant notre approche de modélisation de l'utilisateur avec les hypothèses suivantes :

Hypothèse 1 : l'ensemble des choix est totalement ordonné suivant des contextes.

Pour chaque type d'action h_i, l'ensemble des choix possibles est totalement ordonné et il est possible de déterminer le rang d'un choix (comportement) de l'utilisateur par rapport à un type de contexte $c_j^i \in C^i = \{c_1^i, c_2^i, ..., c_n^i\}$. β_i est une matrice dont chaque élément représente une valeur de rang par rapport à un type de préférence et un type de contexte.

Hypothèse 2 : les préférences sont modélisées suivant des contextes.

Pour chaque type de préférence p_i, sa valeur α_i est un vecteur de IR^n dont chaque élément représente une valeur de préférence pour un type de contexte c_j^i.

Nous introduisons la fonction $note_i$ définie par :

$$\begin{aligned} note_i : \beta^i &\longrightarrow IR^n \\ b &\longmapsto note_i(b) = \frac{\delta}{b} \end{aligned} \qquad (4.5)$$

où β^i est l'ensemble des choix de l'utilisateur pour le type de préférence p_i et δ est une constante de normalisation.

La fonction $note_i$ associe une note aux choix de l'utilisateur suivant les contextes pour un type de préférence p_i.

La préférence α_i d'un utilisateur par rapport à un type de préférence p_i se calcule par :

$$\alpha_i = Moyenne(note_i) \qquad (4.6)$$

L'exploitation de ces informations collectées sur l'utilisateur est effectuée au niveau des méthodes de personnalisation.

4.3.3. Méthodes de personnalisation

Dans cette partie nous présentons deux méthodes complémentaires de raisonnement automatique pour la personnalisation de l'interaction. La première méthode effectue un filtrage collaboratif en analysant les préférences (données pondérées) et les comportements (historique) des utilisateurs. Le principe consiste à prédire le comportement d'un utilisateur par comparaison des comportements des autres utilisateurs ayant les mêmes préférences pour un contexte donné. La deuxième méthode utilise le même type de raisonnement sauf qu'elle se base uniquement sur les comportements (historique) des utilisateurs.

Méthode 1 : filtrage collaboratif basé sur les préférences et sur les comportements des utilisateurs

Dans un contexte fixé, nous supposons qu'à chaque type de préférence p_i est associée une valeur numérique V_{ui} correspondant à la préférence de l'utilisateur. Les préférences de l'utilisateur u peuvent dont être modélisées sous forme de vecteur $\vec{V}_u = (V_{u1},\ldots,V_{un})$.

Par rapport à notre méthode de collecte d'information exposée au §4.3.2 les valeurs V_{ui} peuvent être obtenues en définissant une fonction f_i pour chaque type de préférence p_i.

$$\begin{aligned} f_i : \alpha_i \times C_i &\longrightarrow R \\ (a,\ c) &\longmapsto V_{ui} = f_i(a,c) \end{aligned} \qquad (4.7)$$

où C_i est l'ensemble des valeurs pour les types des contextes C^i.

f_i permet d'obtenir la préférence de l'utilisateur (en attribuant une valeur numérique V_{ui}) par rapport à un type de préférences p_i suivant le contexte d'interaction c. Pour un type de préférence p_i, si les valeurs des vecteurs $a=(a_1,\ldots,a_n)$ et $c=(c_1,\ldots,c_n)$ sont normalisées, une fonction simple de f_i pourrait être :

$$f_i(a,c) = \frac{\sum_{1\le i\le n} c_i a_i}{\sum_{1\le i\le n} c_i} \qquad (4.8)$$

Pour le choix des historiques à considérer par rapport au contexte, on définit une distance[56] d permettant de mesurer le degré de similarité entre deux contextes. Par rapport à un contexte et pour un type d'action h_i, c'est le choix dont le contexte est le plus proche du contexte actuel qui sera considéré.

L'objectif de notre méthode est de déduire pour un type d'action h_i d'un utilisateur u, son choix $s_u \in S = \{s_1,\ldots,s_k\}$ où $k \in IN$ le nombre de choix possibles pour l'action h_i.

La méthode décrite ici est une méthode statistique utilisant une approche probabiliste de Bayes[57]. Le principe consiste à estimer, par rapport à un type d'action h_i, les probabilités $P(s_u = s_d \mid \vec{V}_u)$ qu'un utilisateur u ayant les préférences $\vec{V}_u$ de faire le choix s_d (avec $1 \le d \le k$) et connaissant les préférences des autres utilisateurs et leurs choix respectifs (voir Figure 4.24). Le choix de l'utilisateur serait le s_d qui maximise $P(s_u = s_d \mid \vec{V}_u)$.

[56] On pourra utiliser la distance de Manhattan $d(x,y)=\sum|x_i-y_i|$ (voir annexe F).

[57] Les réseaux bayésiens *«sont des graphes acycliques orientés pour lesquels les nœuds représentent des variables et les arcs représentent l'indépendance conditionnelle entres les différents nœuds»* [Pearl 88].

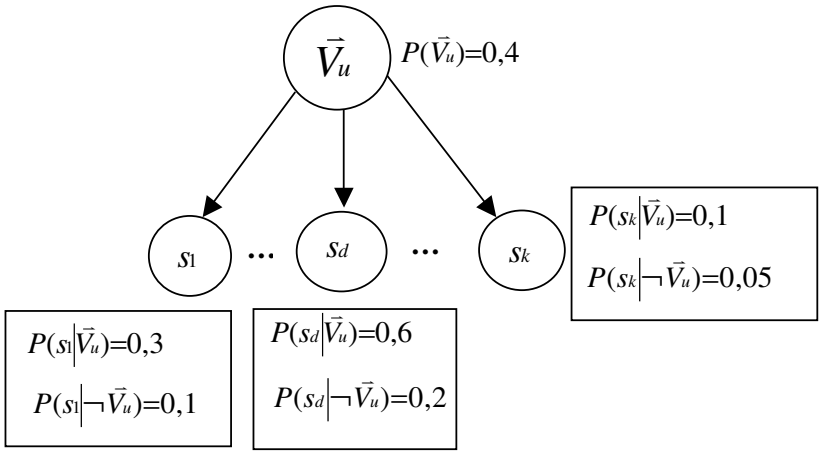


Figure 4.24. Réseau bayésien modélisant les relations entre les préférences de l'utilisateur et ses choix par rapport à un type d'action

L'idée est d'utiliser la formule de Bayes suivante :

$$P(s_u=s_d|\vec{V}_u)=\frac{P(s_u=s_d)\times P(\vec{V}_u|s_u=s_d)}{P(\vec{V}_u)} \quad (4.9)$$

où

- $P(s_u=s_d) = P(s=s_d)$, la probabilité de faire le choix s_d.
- $P(\vec{V}_u|s_u=s_d) = P(\vec{V}_u|s=s_d)$, la probabilité d'avoir les préférences $\vec{V}_u$ de l'utilisateur *u* sachant qu'on a fait le choix s_d pour le type d'action h_i.
- $P(\vec{V}_u)$, la probabilité d'avoir les préférences $\vec{V}_u$ de l'utilisateur *u*.

Ainsi, estimer la probabilité $P(s_u=s_d|\vec{V})$ revient à calculer le second membre de la formule (4.9) ($P(s=s_d) * P(\vec{V}_u|s=s_d)$)/ $P(\vec{V}_u)$. Ce calcul peut se faire en analysant les profils des utilisateurs (plus précisément, en analysant les préférences et les historiques).

Soient les notations suivantes :

NbUtilisateurs(s_d , h_i) : le nombre d'utilisateurs ayant fait le choix s_d pour le type d'action h_i.

TotalUtilisateurs(h_i) : le nombre total d'utilisateurs ayant effectué le type d'action h_i.

On obtient alors :

$$P(s=s_d)=\frac{NbUtilisateurs(s_d,h_i)}{TotalUtilisateurs(h_i)} \quad (4.10)$$

En faisant l'hypothèse d'indépendance entre les types de préférences, on obtient :

$$P(\vec{V}_u|s=s_d)=\prod_{p=1}^{n} P(V_p=V_{up}|s=s_d) \quad (4.11)$$

où $P(V_p=V_{up}|s=s_d)$ est la probabilité d'avoir la valeur V_{up} pour le type de préférence *p* sachant qu'on a effectué le choix s_d pour le type d'action h_i.

Soient les notations suivantes :

$NbUtilisateurs(V_p=V_{up}|s=s_d)$: le nombre d'utilisateurs ayant la valeur V_{up} pour le type de préférence p, pour les utilisateurs ayant fait le choix s_d .

$TotalUtilisateurs(s_d)$: le nombre total d'utilisateurs ayant choisi s_d pour le type d'action h_i .

Pour ne pas annuler $P(\vec{V}_u|s=s_d)$, il faut éviter d'avoir $P(V_p=V_{up}|s=s_d)=0$. Pour cela on ne pose pas :

$$P(V_p=V_{up}|s=s_d)=\frac{NbUtilisateurs(V_p=V_{up}|s=s_d)}{TotalUtilisateurs(s_d)} \tag{4.12}$$

Mais on pose :

$$P(V_p=V_{up}|s=s_d)=\frac{1+NbUtilisateurs(V_p=V_{up}|s=s_d)}{n+TotalUtilisateurs(s_d)} \tag{4.13}$$

On a : $0< P(V_p=V_{up}|s=s_d) \leq 1$ et $\sum_{p=1}^{n} P(V_p=V_{up}|s=s_d)=1$

En effet,

$$\sum_{p=1}^{n} P(V_p=V_{up}|s=s_d)=\sum_{p=1}^{n}\frac{1+NbUtilisateurs(V_p=V_{up}|s=s_d)}{n+TotalUtilisateur(s_d)}$$

$$=\frac{\sum_{p=1}^{n}1+\sum_{p=1}^{n}NbUtilisateurs(V_p=V_{up}|s=s_d)}{n+TotalUtilisateur(s_d)}$$

$$=\frac{n+TotalUtilisateur(s_d)}{n+TotalUtilisateur(s_d)}$$

$$=1$$

Enfin, par la méthode de Bayes, pour un type d'action h_i , pour déterminer le choix d'un utilisateur u, on cherche s_d qui maximise :

$$F_{S_d}=\frac{NbUtilisateurs(s_d,h_i)}{TotalUtilisateurs(h_i)}\prod_{p=1}^{n}\frac{1+NbUtilisateurs(V_p=V_{up}|s=s_d)}{n+TotalUtilisateurs(s_d)} \tag{4.14}$$

Les estimations $P(s_u=s_d|\vec{V}_u)$ peuvent s'effectuer par le calcul de $P(\vec{V}_u)$.

Soit la notation suivante :

$NbUtilisateurs(\vec{V}_u, h_i)$: le nombre d'utilisateurs ayant les mêmes préférences[58] que l'utilisateur u et ayant effectué le type d'action h_i .

On a :

$$P(\vec{V}_u)=\frac{NbUtilisateurs(\vec{V}_u,h_i)}{TotalUtilisateurs(h_i)} \tag{4.15}$$

Finalement, l'estimation de $P(s_u=s_d|\vec{V}_u)$ est donnée par,

[58] On définit une distance pour le calcul de la similarité entre les préférences de deux utilisateurs. On pourra, par exemple, utiliser la distance de Manhattan $d(x,y)=\sum|x_i-y_i|$ (voir annexe F). Deux utilisateurs sont similaires si et seulement si $d(x,y)<\delta$ où δ est une constante.

$$P(s_u=s_d|\vec{V})=\frac{NbUtilisateurs(s_d,h_i)}{NbUtilisateurs(\vec{V}_u,h_i)}\prod_{p=1}^{n}\frac{1+NbUtilisateurs(V_p=V_{up}|s=s_d)}{n+TotalUtilisateurs(s_d)} \quad (4.16)$$

La méthode présentée ci-dessus permet de prédire le comportement d'un utilisateur en se basant sur ses préférences. Cette prédiction est calculée par l'analyse des comportements des autres utilisateurs ayant les mêmes préférences que lui.

Un élément important de cette méthode, c'est qu'elle pourrait être appliquée aussi bien pour la personnalisation du contenu que pour la personnalisation du contenant. En effet, pour la personnalisation du contenu, prenons l'exemple d'une application de recherche d'itinéraire. Les préférences de l'utilisateur pourraient être les critères transport comme *le moins de marche*, *le plus rapide*, *le moins coûteux*, etc. L'historique pourrait être les couples (requête, itinéraire choisi). Avec notre méthode, il est possible de prédire l'itinéraire susceptible de préférer l'utilisateur pour une requête donnée. Pour la personnalisation du contenant, prenons l'exemple d'une application de traitement de textes. Les préférences de l'utilisateur pourraient être *l'utilisation des raccourcis clavier*, *l'utilisation de l'aide*, *les types de fonctionnalité utilisés*, etc. L'historique pourrait être les couples (action précédente, action suivante). Un type de personnalisation pourrait être, par exemple, de cacher, de mettre en apparence ou d'ordonner les éléments de l'interface par rapport au comportement de l'utilisateur. Prenons l'exemple d'un utilisateur qui clique sur le menu *Edition* qui donne le choix entre un *collage simple* ou un *collage spécial*. Le système pourrait déduire les probabilités qu'un utilisateur fasse ces deux types de collage et, par exemple, mettre en valeur le type collage le plus probable (par exemple, un utilisateur qui utilise souvent les raccourcis fournirait une forte probabilité pour le collage spécial puisque le collage simple fournit un raccourci alors que le collage spécial n'en fournit pas).

Un autre élément important de cette méthode, c'est qu'elle permet de prédire le comportement d'un utilisateur même si ce dernier ne dispose pas d'un historique se situant dans le même domaine d'activités que les autres utilisateurs. Pour illustrer ce propos, prenons deux usagers des transports habitant dans deux régions différentes. Les historiques de ces deux usagers sont complètement différents puisque les couples (requête, itinéraire) de chacun de ces usagers portent sur une région différente de l'autre. Cependant, si un de ces usagers part dans la région de l'autre, et s'ils ont les mêmes préférences, la méthode exposée ci-dessus permet à l'autre usager de conseiller un itinéraire à celui qui arrive dans sa région.

Une des limites de cette méthode porte sur la contrainte qu'elle impose sur la nécessité d'avoir des préférences utilisateur clairement définies et pouvant être quantifiées. Or, il n'est pas aisé de définir des types de préférences pertinents pouvant guider un choix utilisateur par rapport à un type d'action. Pour éviter d'avoir à spécifier des types de préférences et à chercher des méthodes pour leurs mises à jour, nous proposons une méthode de personnalisation basée sur les comportements uniques des utilisateurs.

Méthode 2 : filtrage collaboratif basé sur les comportements uniques des utilisateurs

Nous proposons, dans cette partie, une autre méthode qui ne nécessite pas une connaissance explicite des préférences des utilisateurs pour prédire un choix utilisateur. Cette méthode utilise le même principe de Bayes décrite ci-dessus mais en posant la formule suivante :

$$P(s_u=s_d|\vec{H}_u)=\frac{P(s_u=s_d)\times P(\vec{H}_u|s_u=s_d)}{P(\vec{H}_u)} \quad (4.17)$$

où[59]

- $\vec{H}_u=(B_{u1},...,B_{un})$ est le vecteur des choix effectués par l'utilisateur *u* pour les types d'actions $(h_1,\ldots,h_n)$.
- $P(\vec{H}_u)$, la probabilité d'avoir l'historique $\vec{H}_u$ de l'utilisateur *u*.

[59] Dans ce qui suit, nous utilisons les mêmes notations avec les mêmes significations que ce qui précède. Nous expliquons seulement les nouvelles notations qui seront introduites.

- $P(\vec{H}_u | s_u=s_d) = P(\vec{H}_u | s=s_d)$, la probabilité d'avoir l'historique $\vec{H}_u$ de l'utilisateur u sachant qu'on a fait le choix s_d pour le type d'action h_i.

Soient les notations suivantes :

NbUtilisateurs($B_p=B_{up}|s=s_d$) : le nombre d'utilisateurs ayant fait le choix B_{up} pour le type d'action h_p, pour les utilisateurs ayant fait le choix s_d.

TotalUtilisateurs(p, s_d) : le nombre total d'utilisateurs ayant choisi s_d pour le type d'action h_i et ayant effectué le type d'action h_p.

NbUtilisateurs($\vec{H}_u, h_i$) : le nombre d'utilisateurs ayant la même historique[60] que l'utilisateur u et ayant effectué le type d'action h_i.

Par un raisonnement analogue que précédemment[61], pour prédire le choix s de l'utilisateur pour une action h_i, il suffit de prendre le s_d qui maximise :

$$F_{S_d} = \frac{NbUtilisateurs(s_d,h_i)}{TotalUtilisateurs(h_i)} \prod_{p=1}^{n} \frac{1+NbUtilisateurs(B_p=B_{up}|s=s_d)}{n+TotalUtilisateurs(p,s_d)} \qquad (4.18)$$

et

$$P(s_u=s_d|\vec{H}) = \frac{NbUtilisateurs(s_d,h_i)}{NbUtilisateurs(\vec{H}_u,h_i)} \prod_{p=1}^{n} \frac{1+NbUtilisateurs(B_p=B_{up}|s=s_d)}{n+TotalUtilisateurs(p,s_d)} \qquad (4.19)$$

Cette méthode permet la prédiction d'un comportement d'un utilisateur par l'analyse de ses comportements passés et de ceux des autres utilisateurs. Cette méthode remédie à l'inconvénient majeur (nécessité de spécifier les types de préférences qui guident le comportement des utilisateurs) de la première méthode que nous avons proposée. Cependant, elle ne permet pas à des utilisateurs n'appartenant pas aux mêmes domaines d'activités de participer à la recommandation d'un choix. Concrètement, par rapport au transport par exemple, cela veut dire que cette méthode ne permet pas qu'un utilisateur se déplaçant seulement à Valenciennes recommande un itinéraire à celui qui se déplace seulement à Lille. Cette limite est d'autant plus gênante qu'un utilisateur a généralement besoin des conseils d'un autre qui soit spécialiste du domaine qu'il ne maîtrise pas : si l'utilisateur qui se déplace seulement à Lille vient à Valenciennes, il aurait besoin des conseils de celui qui se déplace à Valenciennes pour choisir un itinéraire. C'est pour cela que l'utilisation de cette méthode en complément de la première semble pertinente ; la combinaison de ces deux méthodes dans un même système de personnalisation nous paraît un bon compromis pour bénéficier des avantages des deux méthodes.

Les deux méthodes de personnalisation exposées ci-dessus font parties des méthodes de filtrage social basées sur la mémoire. Comme vu au chapitre 1, ces types de méthodes sont relativement simples à mettre en œuvre et prennent en compte l'évolutivité du profil de l'utilisateur. Par contre, elles génèrent une forte complexité combinatoire qui peut limiter leur utilisation dans une application de grande échelle. Pour ce genre d'application, il sera donc nécessaire de combiner ces méthodes avec des méthodes à base de modèles.

[60] On définit une distance pour le calcul de la similarité entre deux historiques en comptabilisant, par exemple, le nombre de couples (action,choix) qui diffèrent.

[61] L'analogie avec l'équation (4.11) suppose que les choix possibles, pour un type d'action, sont indépendants les uns des autres.

Conclusion

Ce chapitre a décrit une autre partie de notre contribution pour la personnalisation de l'interaction homme-machine. Nous avons proposé un système de personnalisation générique appelé PerSyst (PERsonalization SYSTem) qui peut être utilisé conjointement avec notre méthode PerMet pour le développement d'un système d'information personnalisé. PerSyst est construit à partir d'une architecture multi-agents ce qui lui confère des caractéristiques comme l'adaptabilité, l'autonomie et l'assistance lui permettant de prendre en compte différents types de personnalisation.

PerSyst distingue les *agents applicatifs* qui sont définis lors d'une intégration de PerSyst dans un système d'information (comme les agents de gestion de profil utilisateur, de recherche d'information, d'assistance, etc.) *des agents systèmes* qui sont l'agent de communication, l'agent de coordination et l'agent d'administration. Ces trois agents systèmes forment le noyau de PerSyst et chacun joue un rôle précis dont, tous les trois combinés, permettent à PerSyst de répondre à ses objectifs de multi-applications, de distributivité et d'évolutivité.

L'agent de coordination sert de pont de communication entre les différents agents présents dans PerSyst. Il assure la coordination des messages échangés entre les différents agents applicatifs, la coordination des messages échangés entre les applications externes et les agents applicatifs et la coordination des messages échangés entre le développeur et les agents applicatifs.

L'agent de communication permet la traduction des requêtes envoyées par les applications externes en des messages compréhensibles par l'agent de coordination et vice versa. L'architecture de cet agent comprend deux parties : la première partie se charge de la communication avec les applications externes et la deuxième partie s'occupe de l'interaction avec l'agent de coordination. Cette deuxième partie est fournie sous forme d'une API Java et peut être utilisée indépendamment de PerSyst pour faire communiquer une application java avec les agents construits avec la plate-forme Magique.

L'agent d'administration permet la gestion et l'administration de PerSyst. Il permet à un utilisateur d'interagir graphiquement avec les agents logiciels en exécution composant PerSyst. L'utilisateur dispose d'une vue sur les agents actifs, sur leur organisation et sur leur localisation, sur les compétences et sur le contenu de leurs échanges. Il peut créer et ôter des agents, ainsi qu'agir sur chacun d'eux pour lui supprimer ou lui ajouter une ou des compétences. Il a aussi la possibilité de connecter un agent existant ou de l'enlever de PerSyst. Cet agent peut aussi être utilisé, indépendamment de PerSyst, comme un outil d'interaction graphique avec les agents construits avec la plate-forme Magique.

PerSyst propose des modèles et des méthodes pour la personnalisation de l'interaction homme-machine. Des modèles pour la représentation du profil de l'utilisateur et pour la collecte d'information sur l'utilisateur ont été proposés. De plus, deux méthodes de filtrage collaboratives pour la personnalisation ont été décrites. Ces modèles et ces méthodes sont génériques et doivent être adaptés suivant le domaine d'application.

Le chapitre suivant décrit des applications de la méthode PerMet utilisant le système de personnalisation PerSyst. Ces applications portent sur le développement d'un système d'information personnalisé dans le domaine des transports terrestres de personnes.

Bibliographie du chapitre 4

*[Anli 02] Anli A. *Vers un modèle utilisateur pour la personnalisation de l'information dans les transports terrestres de personnes*. Rapport de DEA, Université de Valenciennes et du Hainaut-Cambrésis. Juillet 2002.

*[Anli et Abed 05] Anli A. et Abed M. PerSyst : Un Système de Personnalisation générique, Application à l'information voyageur. *Workshop Méthodologies et Heuristiques pour l'Optimisation des Systèmes Industriels (MHOSI'05).* Hammamet, Tunisie, avril 2005.

*[Anli et al., 04a] Anli A., Petit-Rozé C. et Grislin-Le Strugeon E. Plate-forme d'intégration de services personnalisés à base d'agents logiciels. *Génie Logiciel*, 71, pp. 34-39, 2004.

*[Anli et al., 05] Anli A., Grislin-Le Strugeon E. et Abed M. Une plate-forme de personnalisation basée sur une architecture multi-agents. *Revue des Nouvelles Technologies de l'Information* (RNTI-E), 5, pp. 95-100, 2005.

*[Bousbia et al., 05] Bousbia S., Anli A., Trentesaux D. and Grislin-Le Strugeon E. Agile Scheduling of flexible manufacturing systems of production. In P. Borne, M. Benrejeb, N. Dangoumeau, L. Lorimier (Eds.), *17th IMACS World Congress "Scientific Computation, Applied Mathematics and Simulation"* (July 11-15, Paris), pp. 316-323, juillet 2005.

[Bouzeghoub et Kostadinov 04] Bouzeghoub M et Kostadinov D. *Une approche multidimensionnelle pour la personnalisation de l'information*. Rapport PRiSM, Versailles, France, 2004.

[Cardon 00] Cardon A. *Conscience artificielle et systèmes adaptatifs*. Paris : Eyrolles, 2000.

[Goldberg 94] Goldberg D. *Algorithmes génétiques : exploration, optimisation et apprentissage automatique*. Paris : Addison Wesley, 1994.

[Howden et al., 01] Howden N., Rönnquist R., Hodgson A. and Lucas A. JACK Intelligent Agents – Summary of an Agent Infrastructure. *5th International Conference on Autonomous Agents*, Montreal, Canada, May 2001.

[Kadima et Monfort 03] Kadima H. et Monfort V. *Les web services : Techniques et outils XML, WSDL, SOAP, UDDI, Rosetta, UML*. Paris : Dunod, 2003.

[Mandiau et al., 02] Mandiau R., Grislin-Le Strugeon E. et Péninou A. (Eds.). *Organisation et applications des SMA*. Paris : Hermès, 2002.

[Mathieu et Routier 01] Mathieu P. et Routier J. C. Une contribution du multi-agent aux applications de travail coopératif. *TSI Hermès Science Publication*, Numéro Spécial : Télé-applications, Vol. 13, n°2-3, pp. 207-232, 2001.

[Monfort et Goudeau 04] Monfort V. et Goudeau S. *Web services et Interopérabilité des SI : WS-I, WSAD/J2EE, Visual Studio .Net et BizTalk*. Paris : Dunod, 2004.

[Pearl 88] Pearl J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo : Morgan Kaufmann Publishers, 1988.

[Petit-Rozé et Grislin-Le Strugeon 05] Petit-Rozé C. et Grislin-Le Strugeon E. Modélisation d'interactions par et avec des agents en personnalisation d'information. In A. Herzig, Y. Lespérance, A.i. Mouaddib (Eds.), *Actes des Troisièmes Journées Francophones Modèles Formels de l'Interaction (MFI'2005)* (Caen, France, 25-27 Mai), Cepadues-Editions, Mai 2005.

[Poulon et Vaudry 03] Poulon A. et Vaudry C. Recherche d'informations : de l'utilisateur au contexte. In *15ème Conférence Francophone sur l'Interaction Homme-Machine, IHM'03*. Caen, France, novembre 2003.

[Rizcallah 00] Rizcallah M. *Construire un annuaire d'entreprise avec LDAP*. Paris : Eyrolle, 2000.

[Routier et al., 01] Routier J.C., Mathieu P. and Secq Y. Dynamic skill learning : A support to agent evolution. In *Proceedings of the AISB'01 Symposium on Adaptive Agents and Multi-Agent Systems*, pp. 25-32, University of York, United Kingdom, March 2001.

[Secq 03] Secq Y. *RIO : Rôles, Interactions et Organisations. Une méthodologie pour les systèmes multi-agents ouverts*. Thèse de doctorat, Université des sciences et technologies de Lille, décembre 2003.

[Zimmermann et al., 05] Zimmermann A., Specht M. and Lorez A. Personalization and Context Management. *User Modeling and User-Adapted Interaction*, 15, pp. 275-302, 2005.

Chapitre 5
Mon service transport, une application de PerMet pour le développement d'un SI personnalisé

Sommaire

Introduction

Ce cinquième chapitre décrit une application de la méthode PerMet utilisant le système de personnalisation PerSyst. Cette application porte sur le développement d'un système d'information personnalisé dans le domaine des transports terrestres de personnes. L'objectif est de faciliter l'accès à l'information multi-modes de transport d'un usager désirant effectuer un déplacement. En effet, l'usager se trouve confronté à un ensemble de sources d'informations disparates (horaires et tarifs délivrés par les différents exploitants, cartes,... sur papier, borne interactive ou par Internet) qui s'avèrent parfois difficiles à intégrer en un plan de déplacement précis et unique, adapté à ses besoins et préférences. Comme souligné dans le chapitre 1, la personnalisation s'avère une approche pertinente pour relever ce défi. Dans le contexte de l'information aux usagers intégrant plusieurs modes de transports et leurs connexions, notre objectif est, d'une part, d'aider l'usager dans sa démarche de recherche d'informations et, d'autre part, de lui fournir un résultat personnalisé, c'est-à-dire toute l'information nécessaire et uniquement l'information nécessaire en fonction de son destinataire [Petit-Rozé et al., 04].

Le système d'information développé est un portail web réalisé à partir de la plate-forme MASC[62] de la société Archimed. Trois services différents sont décrits. Le premier service (recherche d'itinéraire personnalisé) sert à valider PerMet pour le développement d'un nouveau service personnalisé. Le deuxième (agenda transport) illustre la personnalisation d'un service déjà existant. Ce deuxième service utilise le même type de personnalisation que celui du premier service dans le but de valider la réutilisation de la partie droite (SP) de la méthode PerMet. Le troisième service (information personnalisée des perturbations) consiste en la mise à disposition personnalisée des perturbations dans les transports terrestres de personnes. Ce service utilise un type de personnalisation complètement différent. L'objectif étant de démontrer la capacité de PerSyst à intégrer différents types de personnalisation. Ce service illustre aussi la capacité de PerSyst à effectuer de la personnalisation prenant en compte différents canaux de communication et différentes plates-formes d'interaction.

Les exemples d'illustration porteront sur le réseau transport de la Figure 5.1 volontairement simplifié mais se basant sur des données réelles. Ce réseau décrit les points de connexions permettant à un usager d'effectuer un déplacement du laboratoire LAMIH de l'université de Valenciennes vers la société Archimed à Lille. Trois exploitants interviennent dans la réalisation du voyage :

- Semurval assure le déplacement de l'usager dans la ville de Valenciennes et de ses environs. Les modes de transport considérés sont le bus, la voiture et la marche à pied. D'autres arrêts de bus auraient pu être considérés. Par exemple, à part l'arrêt « Aulnoy université », nous n'avons pas considéré les trois autres arrêts de bus qui sont situés à proximité du LAMIH. Le mode de transport « tramway » aurait pu aussi être pris en compte.
- Transpole s'occupe de la région lilloise. Les modes de transport considérés sont le bus et le métro. D'autres modes comme le vélo ou le taxi auraient pu être considérés.
- SNCF relie les deux villes de Valenciennes et de Lille par le train Ter de la région nord Pas-de-Calais. Nous avons aussi intégré la possibilité qu'un usager prenne une voiture pour aller du LAMIH à Archimed sans passer par le réseau des transports publics.

Par rapport à ce réseau, l'usager dispose de neuf itinéraires possibles pour aller du LAMIH à Archimed.

[62] MASC est une plateforme de gestion de portails web capable d'héberger un ensemble des services accessibles en Intranet, Extranet ou via Internet. Il permet de créer des services et de contrôler des profils d'accès, c'est-à-dire l'ensemble des règles qui doivent être respectées pour permettre à une catégorie d'utilisateurs d'accéder aux services du portail. La gestion des droits dans le portail MASC est basé sur la définition de rôles. Chaque utilisateur connecté hérite d'un ou plusieurs rôles qui définissent son profil et conditionnent l'accès aux services et aux informations du portail. MASC intègre un ensemble de services comme la messagerie, les forums, l'agenda, etc.

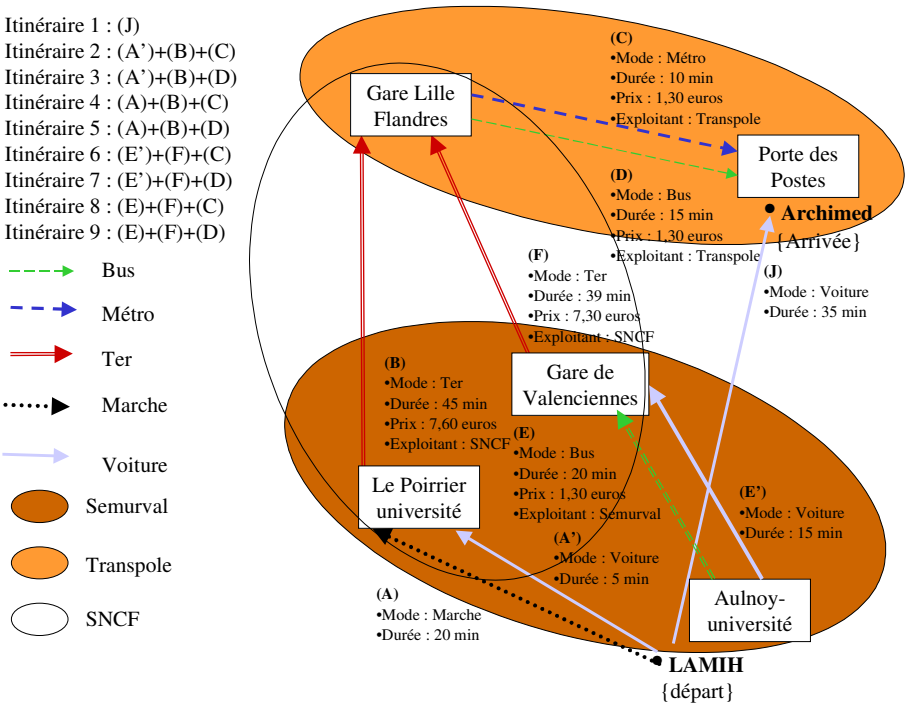


Figure 5.1. Choix d'itinéraires de l'exemple de déplacement

Ces données relatives serviront à illustrer l'utilisation des services développés. Les parties suivantes décrivent tour à tour, et pour chacun de ces services, la mise en pratique de la méthode proposée.

5.1. Recherche d'itinéraire personnalisé

Le premier service concerne la recherche d'itinéraire de manière "classique", telle qu'elle est généralement présente dans les systèmes d'information transport. La particularité de ce service se situe au niveau de la fourniture des résultats et aussi le chaînage des modes de transport qui sont personnalisés par rapport aux préférences de l'utilisateur [Anli et al., 04]. L'utilisateur fournit le lieu de départ, le lieu d'arrivée et l'heure d'arrivée. Le système lui propose un itinéraire suivant ses préférences. L'utilisateur a la possibilité de valider la proposition ou de choisir un autre itinéraire.

Il s'agit d'un nouveau service donc les efforts à fournir dans les différentes phases sont particulièrement importantes.

5.1.1. Analyse et spécification du service

Capture des besoins fonctionnels

Les besoins fonctionnels ont été modélisés au travers de diagrammes de cas d'utilisation. La Figure 5.2 présente les besoins fonctionnels pour le service de recherche d'itinéraire. Les cas d'utilisation peuvent se structurer en trois catégories :

- la première catégorie concerne l'identification de l'utilisateur (cas d'utilisation « S'inscrire » et « S'authentifier »). Ces fonctionnalités sont déjà fournies par la plate-forme MASC et seront directement utilisées par le service de recherche d'itinéraire.
- la deuxième catégorie regroupe les fonctionnalités pour la recherche d'itinéraire personnalisé (cas d'utilisation « Rechercher un itinéraire », « Accéder à la liste des itinéraires possibles », « Afficher le détail d'un itinéraire » et « Choisir un itinéraire »).

- la troisième catégorie concerne les cas d'utilisation pour la manipulation du profil utilisateur (« Gérer le profil », « Visualiser le profil », « Modifier les préférences » et « Supprimer le profil »).

Chaque cas d'utilisation est suivi d'une fiche de description textuelle. La fiche de description textuelle pour le cas d'utilisation « Rechercher un itinéraire » est fournie en Annexe E.

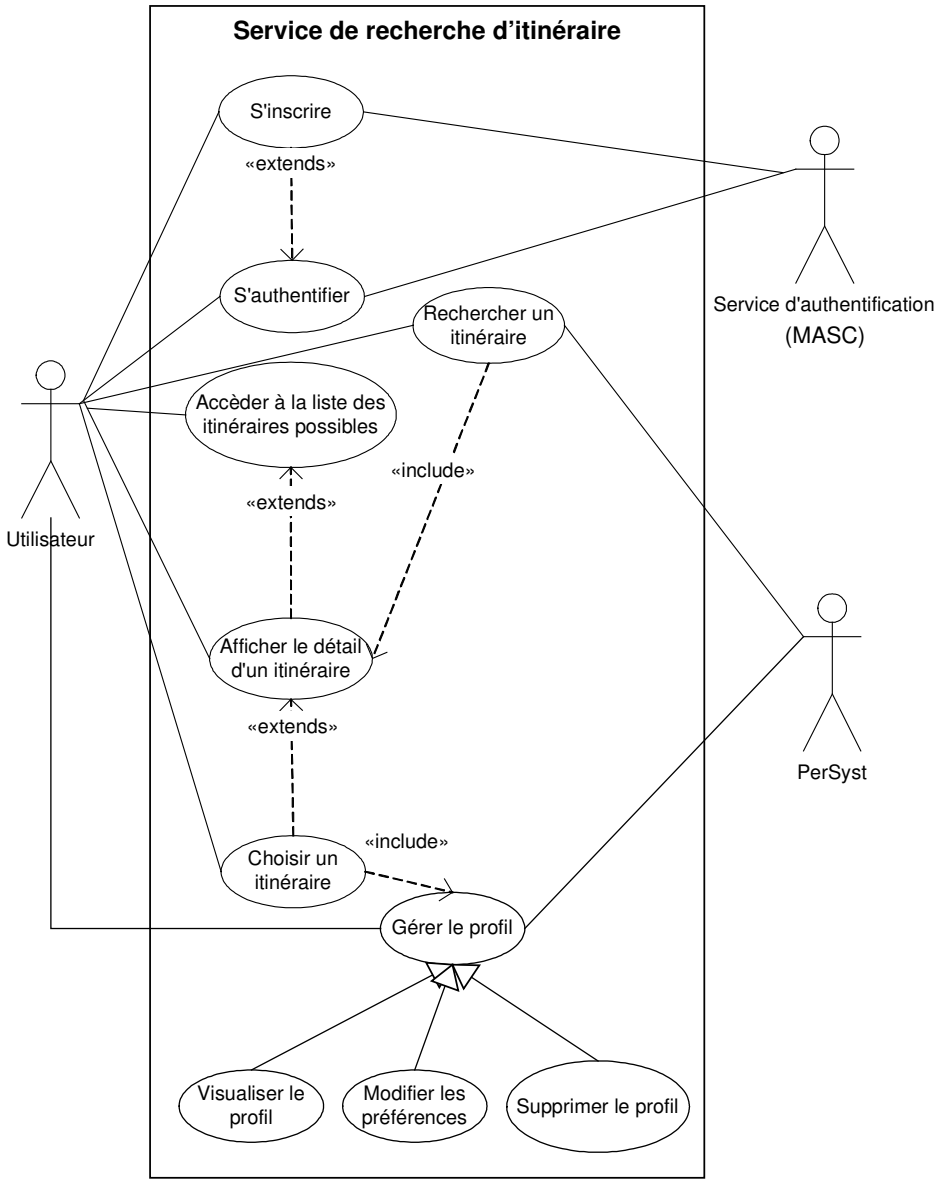


Figure 5.2. Diagramme des cas d'utilisation du service de recherche d'itinéraire personnalisé

Analyse fonctionnelle

En partant du diagramme des cas d'utilisation issus de l'étape des captures des besoins fonctionnels, nous avons structuré et modélisé les objets métiers du service. La Figure 5.3 présente la modélisation statique du service de recherche d'itinéraire. Les classes nécessaires pour assurer les différentes responsabilités ont été définies. Les modules qui ne font pas parties du service (mais qui sont utiles au service) ont été représentés sous forme de packages. Pour affiner le modèle statique, une

série de modèles dynamiques a été établie. Par exemple, la Figure 5.4 présente un scénario pour la recherche d'itinéraire en détaillant les interactions entre les différents objets du service. Après s'être connecté à MASC, l'utilisateur lance sa requête au travers du service (*search*) de recherche d'itinéraire personnalisé. *Search* demande le nom de l'utilisateur connecté à MASC et envoie la requête complétée du nom de l'utilisateur à PerSyst. PerSyst envoie à *Search* tous les itinéraires possibles répondant à la requête et recommande un itinéraire susceptible d'intéresser l'utilisateur. *Search* crée *ResultSearch* qui fournit les résultats personnalisés à l'utilisateur.

Nous avons aussi défini les modèles de données que le service devrait échanger avec le système de personnalisation (voir Figure 5.5). Une requête précise l'identifiant de l'utilisateur (*DN*) et les données (*departure*, *arrival*, *arrivalTime*, …) pour la recherche d'itinéraire. Une réponse (*RESPONSE*) reprend la requête (avec le DN) et ajoute tous les itinéraires possibles (*Result*). Cette réponse est associée à un choix (*Choice*) qui permet de référencer l'itinéraire susceptible d'intéresser l'utilisateur. Un itinéraire (*Result*) est composé de plusieurs trajets (*way*) et comprend ses caractéristiques propres (*Criteria*).

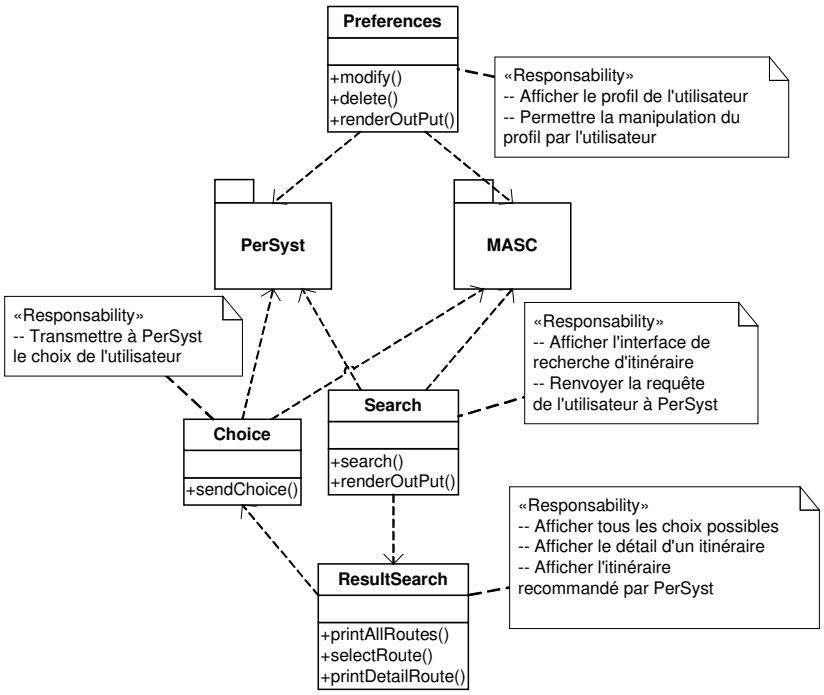


Figure 5.3. Diagramme de classe pour le service de recherche d'itinéraires

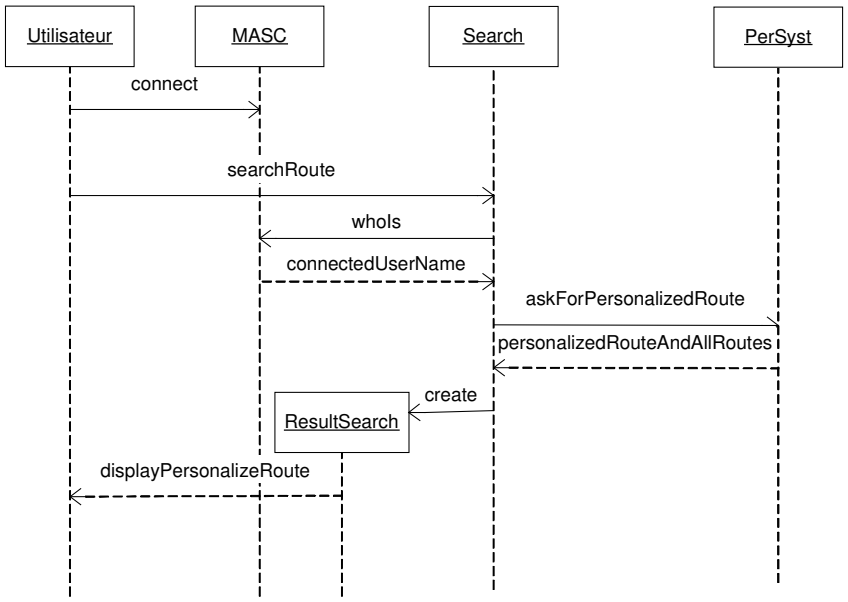


Figure 5.4. Diagramme de séquences pour la recherche d'itinéraire personnalisé

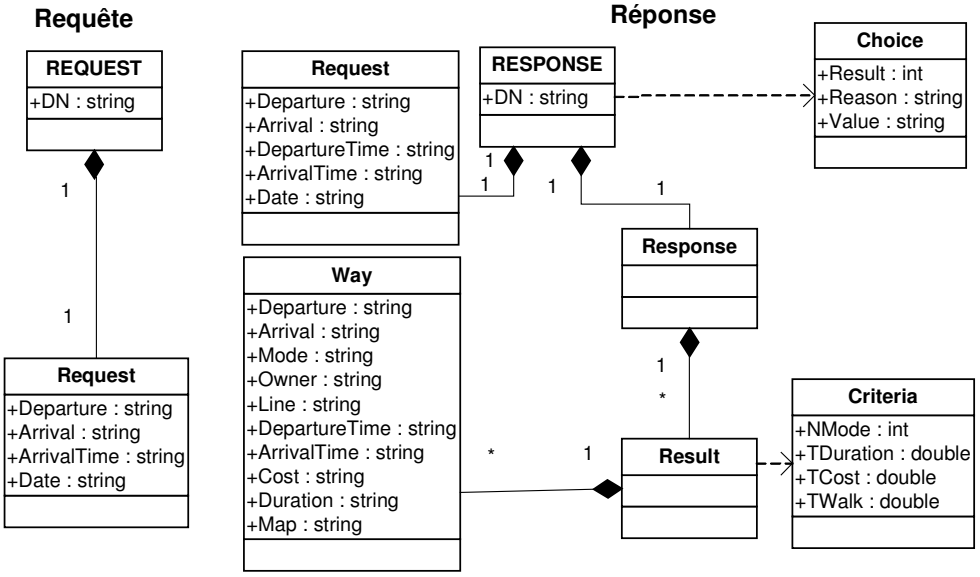


Figure 5.5. Modèles de données échangées entre le service de recherche d'itinéraire et PerSyst

Capture des besoins techniques

Nous disposons d'un serveur sur lequel est installé Windows 2000 Server. Tous les composants logiciels seront installés sur ce serveur (voir Figure 5.6). L'utilisateur accède au service de recherche d'itinéraire à partir de son poste de travail en passant par le réseau internet.

Conception générique

Nous avons décomposé le service en trois sous-systèmes (Manipulation des préférences des utilisateurs, Recherche d'itinéraire, Accès à PerSyst). Chaque sous-système correspond à un regroupement d'éléments de modélisation fournissant une même unité de comportement. Par exemple,

le sous-système « Accès à PerSyst » contient la classe *PersonalizeSystem*. La classe *PersonalizeSystem* utilise le même *design pattern* qu'une classe « singleton » [Gamma et al., 99]. La différence de la classe *PersonalizeSystem* avec une classe singleton se trouve au niveau de l'objet envoyé lors de l'appel à la méthode « *getInstance*() ». Dans une classe singleton, l'objet envoyé correspond à une instance de la classe singleton elle-même alors que dans *PersonalizeSystem*, il s'agit d'un objet d'une autre classe (*PerSyst)*.

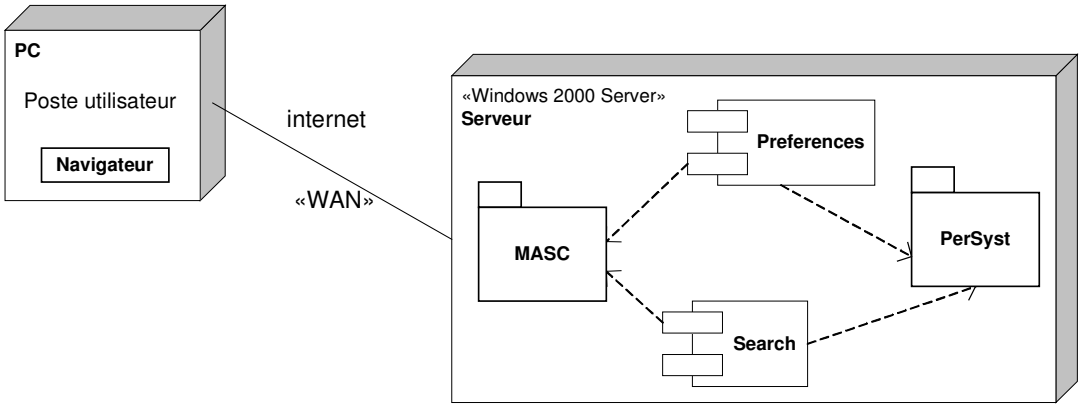


Figure 5.6. Capture des besoins techniques pour le service de recherche d'itinéraire

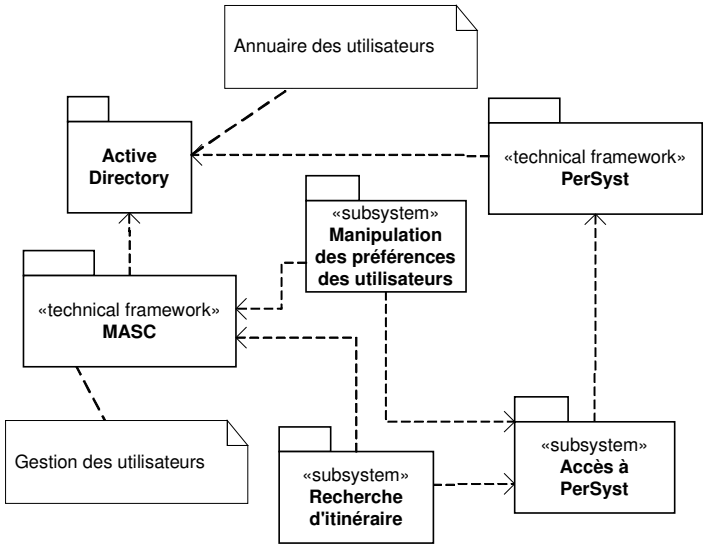


Figure 5.7. Conception générique pour le service de recherche d'itinéraire

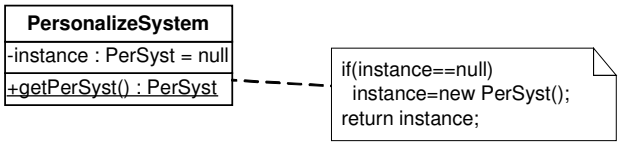


Figure 5.8. Design pattern pour la récupération d'une référence à PerSyst

5.1.2. Conception du service

Le service étant une application web, nous avons utilisé les extensions UML proposées par [Conallen 00] pour la modélisation conceptuelle du service. La Figure 5.9 présente le diagramme de classes conceptuelles pour la recherche d'itinéraire personnalisé (pour ne pas surcharger la figure, les classes pour la manipulation du profil utilisateur et celles pour l'authentification de l'utilisateur ne sont pas représentées dans ce diagramme).

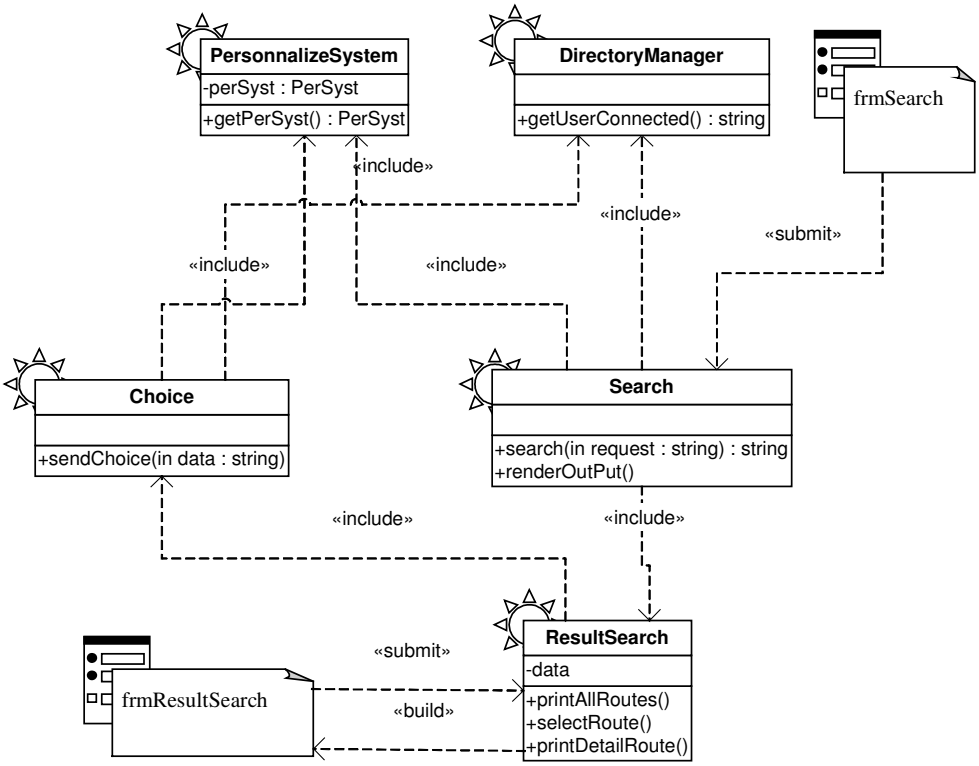


Figure 5.9. Diagramme de classes de conception pour le service de recherche d'itinéraire

5.1.3. Implémentation du service

Les applications (*Search*, *ResultSearch*, *Choice*, etc.) ont été implémentées au travers de pages ASP (Active Server Page). L'implémentation du module *PersonnalizeSystem* n'est pas complètement finalisée dans cette phase. En effet, le module « *PersonnalizeSystem* » fournit une fonction (*getPerSyst*) ayant pour objectif le renvoi d'un objet permettant l'invocation des primitives de PerSyst. Puisque cette partie SI est effectuée indépendamment de la partie SP (donc de PerSyst), il a été nécessaire de simuler l'objet qui devrait être renvoyé par cette fonction. Cet objet simule les données que devraient s'échanger le service et PerSyst en respectant les modèles de données définis dans la phase d'analyse du service. Ces données sont transcodées sous le format XML (la Figure 5.10 donne un exemple de données XML pour une requête de recherche d'itinéraire).

Les présentations (*frmSearch*, *frmResultSearch*, …) sont réalisées au travers d'une transformation XSLT en utilisant le moteur de transformation JSE[63] d'Archimed. La Figure 5.11 présente la page de recherche d'itinéraire permettant aussi la visualisation et la modification des préférences de l'utilisateur.

[63] Le moteur JSE d'Archimed adopte une architecture qui permet l'intégration de plusieurs chartes graphiques sur un même portail web tout en utilisant les mêmes feuilles de style XSLT. Ce moteur utilisé conjointement avec MASC permet la personnalisation des interfaces utilisateur suivant le rôle de l'utilisateur.

```
- <REQUEST>
    <DN>LDAP://SRV-ETCL-
      DC01:389/cn=john,OU=Sitp,DC=DOMTRANSPORT,DC=local</DN>
  - <Request>
      <Departure>LAMIH</Departure>
      <Arrival>Archimed</Arrival>
      <DepartureTime>07:00</DepartureTime>
      <Date>12/07/2006</Date>
    </Request>
  </REQUEST>
```

Figure 5.10. Exemple de données XML pour une requête de recherche d'itinéraire

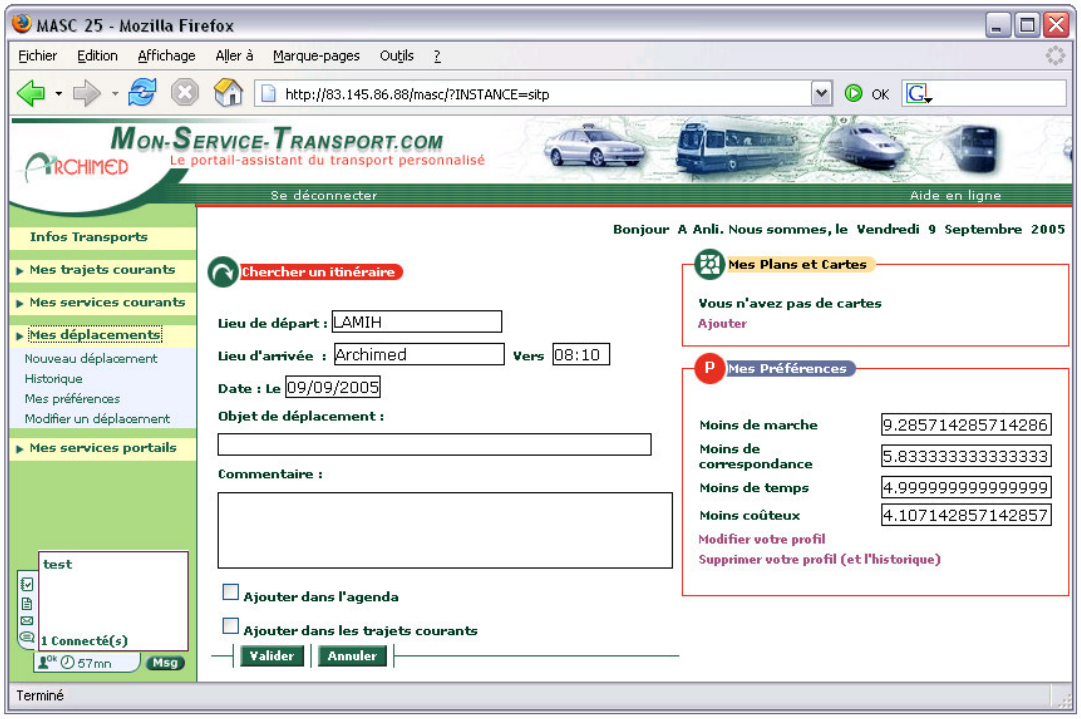


Figure 5.11. Page de recherche d'itinéraire incluant la visualisation et la modification du profil de l'utilisateur

A l'issue de l'implémentation du service, des tests d'ergonomie[64] et de vérification des fonctionnalités du service ont été effectués. Même si ces tests utilisaient des données simulées (au niveau du module *PersonnalizeSystem*), ils ont été utiles puisque, par exemple, ces tests nous ont permis d'ajouter des attributs (*reason*, *value*) dans la classe *Choice* du modèle de données des réponses envoyées par PerSyst. Ces attributs ont été nécessaires pour fournir une explication à l'utilisateur sur la réponse recommandée par le système de personnalisation.

[64] Ces tests ont surtout porté sur l'utilisabilité du système par le recueil des commentaires et remarques de quelques utilisateurs en laboratoire et à la société Archimed. Des tests plus approfondis restent à être effectués.

5.1.4. Analyse des agents

Cette phase débute la partie SP de la méthode PerMet. Elle a été démarrée en parallèle avec la partie SI (à l'issue de la phase d'analyse du service).

Analyse des modèles d'agents

Le diagramme des cas d'utilisation suivant (voir Figure 5.12) reprend celui de la Figure 5.2 sous l'angle de la personnalisation. Dans cette application, nous supposons qu'il existe une autorité fédératrice qui regroupe les bases de données de Transpole, Semurval et SNCF. Nous avons simulé cette autorité en créant une base de donnée XML permettant la recherche des itinéraires pour un déplacement de Valenciennes à Lille. Les informations des utilisateurs sont stockées dans l'annuaire LDAP qui est le même que celui utilisé par MASC pour la gestion des utilisateurs.

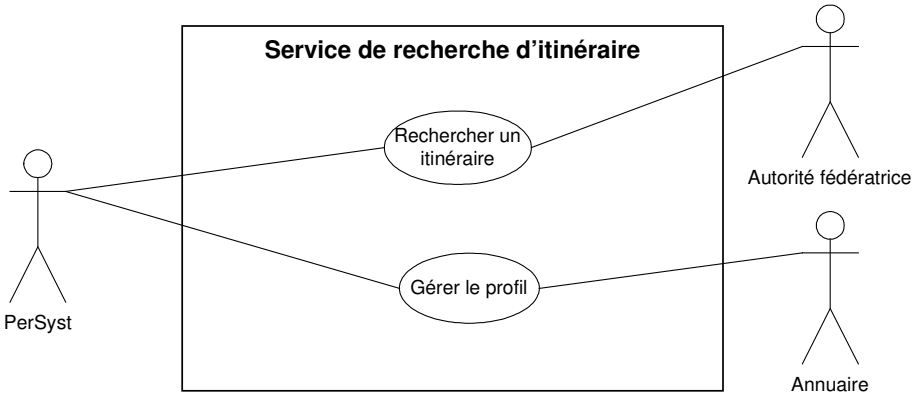


Figure 5.12. Diagramme des cas d'utilisation du service de recherche d'itinéraire vu sous l'angle de la personnalisation

En analysant ce diagramme des cas d'utilisation et en suivant les règles de la méthode PerMet pour l'analyse des modèles d'agents (voir chapitre 3, §3.2.4), nous obtenons un modèle d'agent pour la recherche d'itinéraire et un modèle d'agent pour la gestion du profil utilisateur. En effet, la règle 1 (un modèle d'agent par fonctionnalité attendue du service) et la règle 3 (un modèle d'agent par ressource externe) nous guident à un modèle d'agent pour la recherche d'itinéraire et à un modèle d'agent pour la gestion des informations de l'utilisateur. La règle 2 qui stipule un modèle d'agent par plate-forme d'interaction ne sera pas appliquée ici, car dans cette application, l'objectif est de laisser la gestion de l'interaction entre l'utilisateur et le service à la plate-forme MASC. L'utilisateur accède au service au travers d'un navigateur web.

Les différents rôles que doivent jouer ces agents sont décrits dans la Figure 5.13.

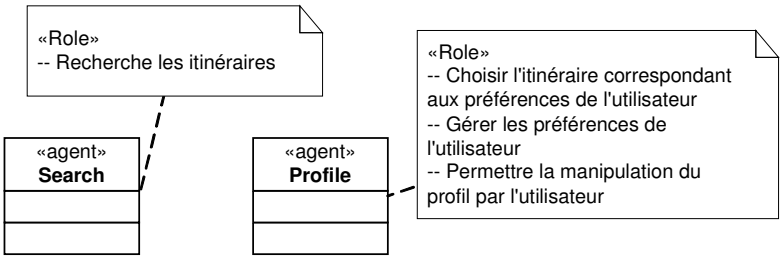


Figure 5.13. Modèles d'agents pour le service de recherche d'itinéraire

Identification des accointances

La Figure 5.14 représente les interactions des agents pour la recherche d'itinéraire. Remarquons l'intervention de deux agents de PerSyst : L'agent de communication qui permet la

transmission des messages issus du service de recherche d'itinéraire intégré dans MASC et l'agent de coordination qui assure la coordination des messages entre l'agent « *Search* » et l'agent « *Profile* ». L'analyse de ce diagramme d'interaction nous a permis de définir les relations d'accointances décrites dans la Figure 5.15.

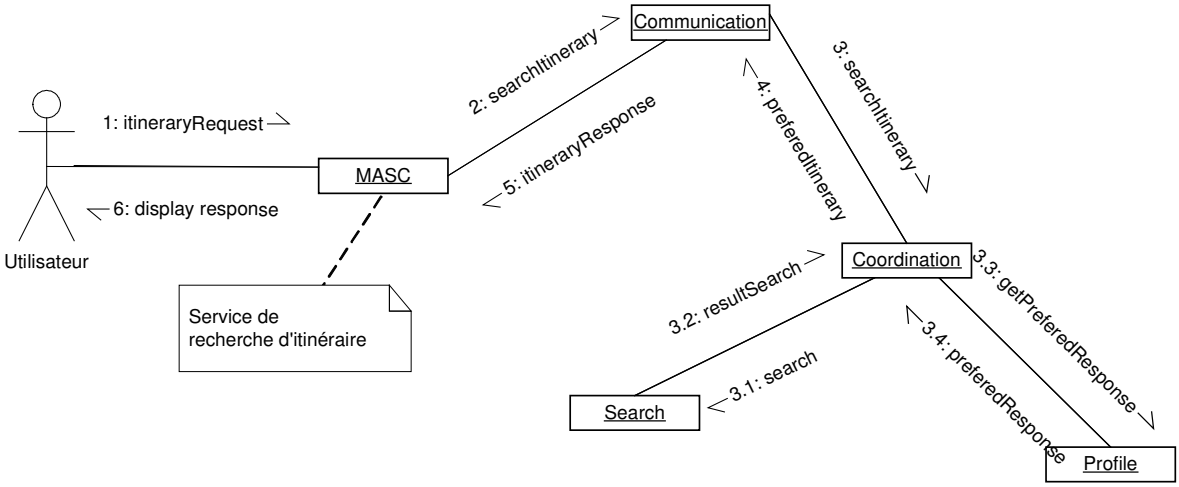


Figure 5.14. Interactions entre les agents pour le service de recherche d'itinéraire

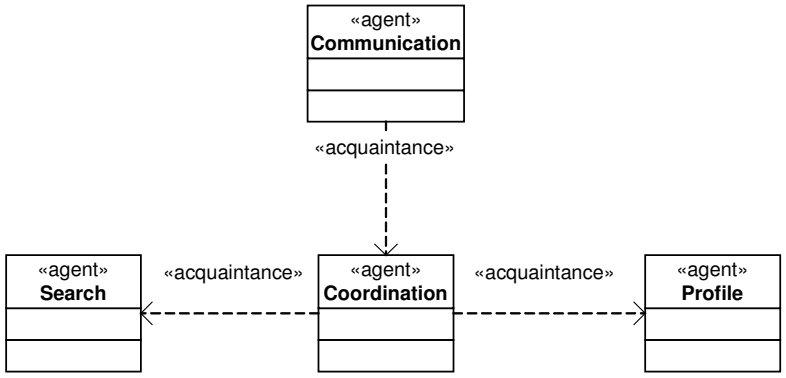


Figure 5.15. Relations d'accointance entre les agents pour le service de recherche d'itinéraire

Analyse des comportements

Nous avons analysé les comportements nécessaires dont doit disposer chaque modèle d'agent pour la réalisation de ses rôles. L'analyse des modèles d'agents nous a permis d'identifier les comportements des agents « *Search* » et « *Profile* ». Par exemple, la Figure 5.17 présente les comportements nécessaires à l'agent « *Profile* » pour le service de recherche d'itinéraire. L'analyse du modèle des interactions entre les agents nous a permis d'identifier d'autres comportements comme celui dont doit disposer l'agent de coordination pour pouvoir assurer la coordination des agents (voir Figure 5.16).

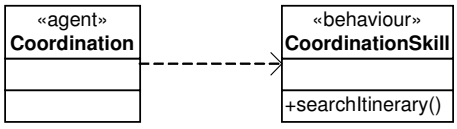


Figure 5.16. Compétence de l'agent de coordination

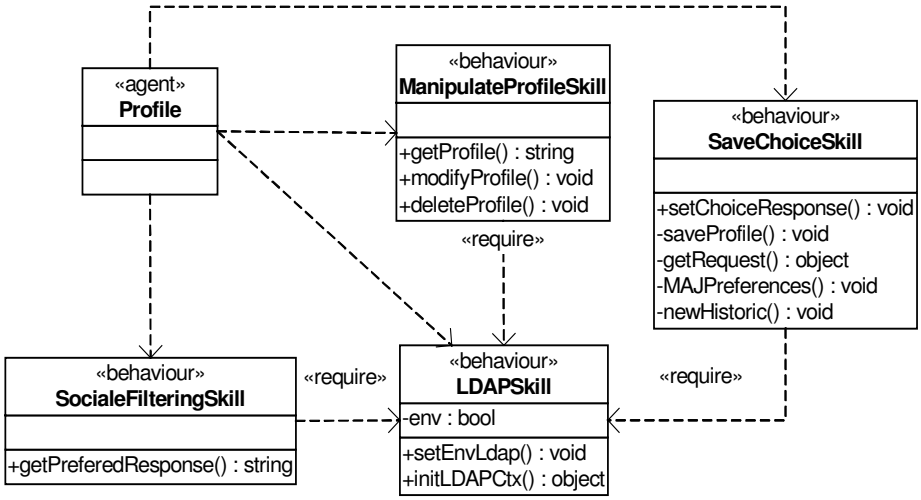


Figure 5.17. Compétences de l'agent de gestion de profil

Information de déploiement

La Figure 5.18 présente le diagramme de déploiement des agents logiciels interagissant avec le service pour la personnalisation des itinéraires.

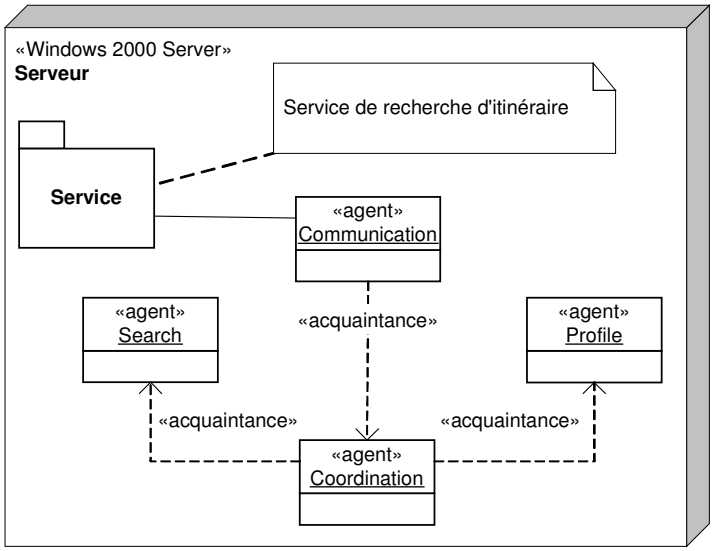


Figure 5.18. Déploiement des agents

5.1.5. Conception des comportements des agents

Dans cette phase, pour chaque modèle d'agent, nous avons affiné et détaillé les comportements des agents. Par exemple, pour l'agent de gestion de profil, nous avons appliqué notre modèle de l'utilisateur proposé dans le chapitre 4. Ce modèle est stocké au niveau des agents de gestion de profil sous forme de document XML (un exemple de profil d'un utilisateur est fourni en Annexe D). La partie statique concerne les informations d'authentification (login, mot de passe) et les données personnelles de l'utilisateur (nom, prénom, lieu de travail, etc.). Ces données proviennent de l'annuaire LDAP où MASC enregistre les informations de l'utilisateur lors de son inscription. La partie pondérée modélise les préférences de l'utilisateur par rapport aux critères transport (moins de correspondance, le plus rapide, le moins de marche à pied, le moins coûteux). Chaque critère est associé à une note, comprise entre 0 et 10, qui représente le degré de préférence de l'utilisateur par

rapport à ce critère. La partie historique enregistre les requêtes et les itinéraires choisis par l'utilisateur. Les contextes d'interaction [Zimmermann et al., 05] ne sont pas pris en compte dans cette application.

La Figure 5.19 présente les activités réalisées lors de l'exécution de l'action interne « *MAJPreferences* » pour la compétence « *SaveChoiceSkill* ». C'est une application de la méthode de collecte automatique des préférences de l'utilisateur présentée dans le chapitre 4 (voir §4.3.2). L'objectif ici est de déduire les préférences de l'utilisateur suivant des critères associés aux itinéraires (moins de correspondance, le plus rapide, le moins de marche à pied, le moins coûteux).

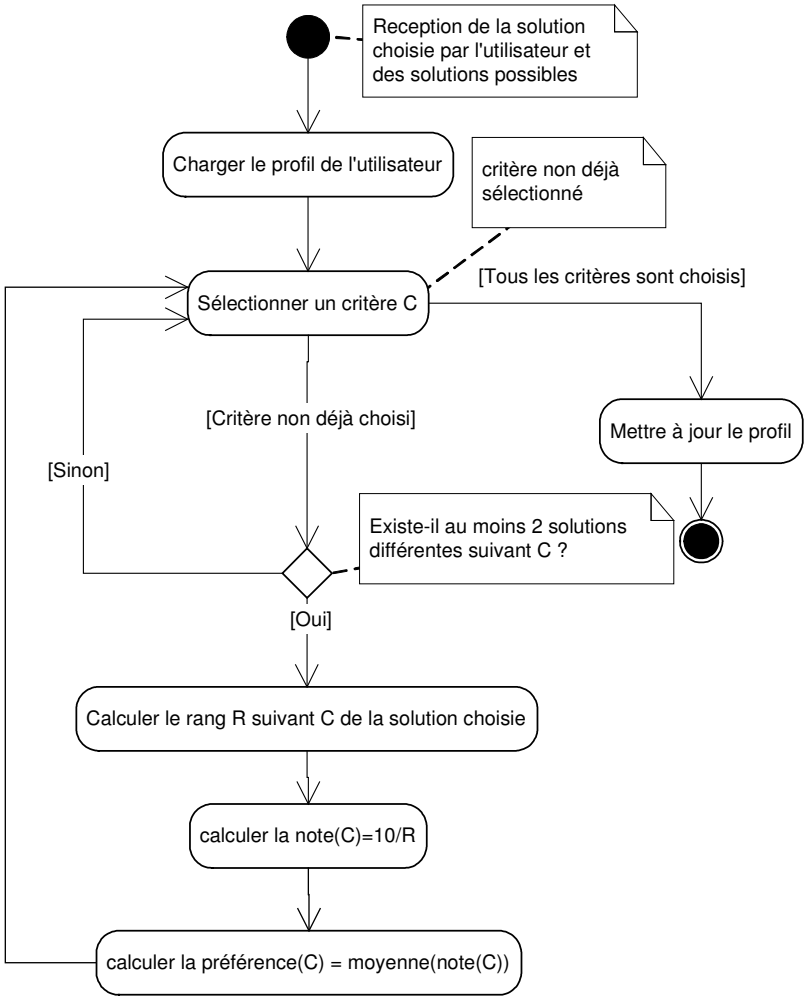


Figure 5.19. Activités pour la mise à jour des préférences de l'utilisateur

La Figure 5.20 présente les activités réalisées lors de l'exécution du service « *getPreferedResponse* » de la compétence « *SocialeFilteringSkill* ». Pour sélectionner l'itinéraire susceptible d'intéresser l'utilisateur, un vote majoritaire (sélectionner l'itinéraire le plus choisi par les utilisateurs) est effectué sur les itinéraires si l'utilisateur courant n'a pas de profil. Sinon, si l'utilisateur possède un profil et a déjà effectué la requête, l'itinéraire qu'il avait choisi lui est recommandé. Sinon, s'il possède un profil mais n'a jamais effectué la requête, un filtrage collaboratif (c'est la méthode de filtrage collaboratif basé sur les préférences et sur les comportements des utilisateurs du chapitre 4 qui a été appliquée) est effectué pour choisir l'itinéraire à proposer à

l'utilisateur. Ce modèle combine donc une méthode cognitive (recommandation par rapport au profil) et des méthodes sociales (vote majoritaire et filtrage collaboratif).

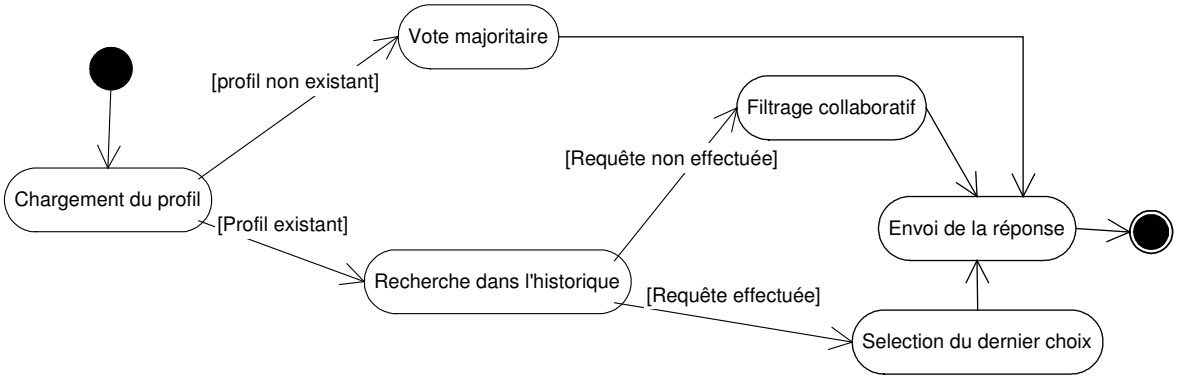


Figure 5.20. Activités pour le choix de l'itinéraire

5.1.6. Implémentation des comportements des agents

Les compétences des agents ont été implémentées et testées au fur et à mesure de leur développement. La création des agents, l'association de leurs compétences et leur déploiement ont été réalisés au travers de l'interface d'administration de PerSyst (voir Figure 5.21). Après le déploiement des agents, des tests de vérification du bon fonctionnement du SMA composant PerSyst ont été effectués par une simulation des données qui devraient provenir des applications externes.

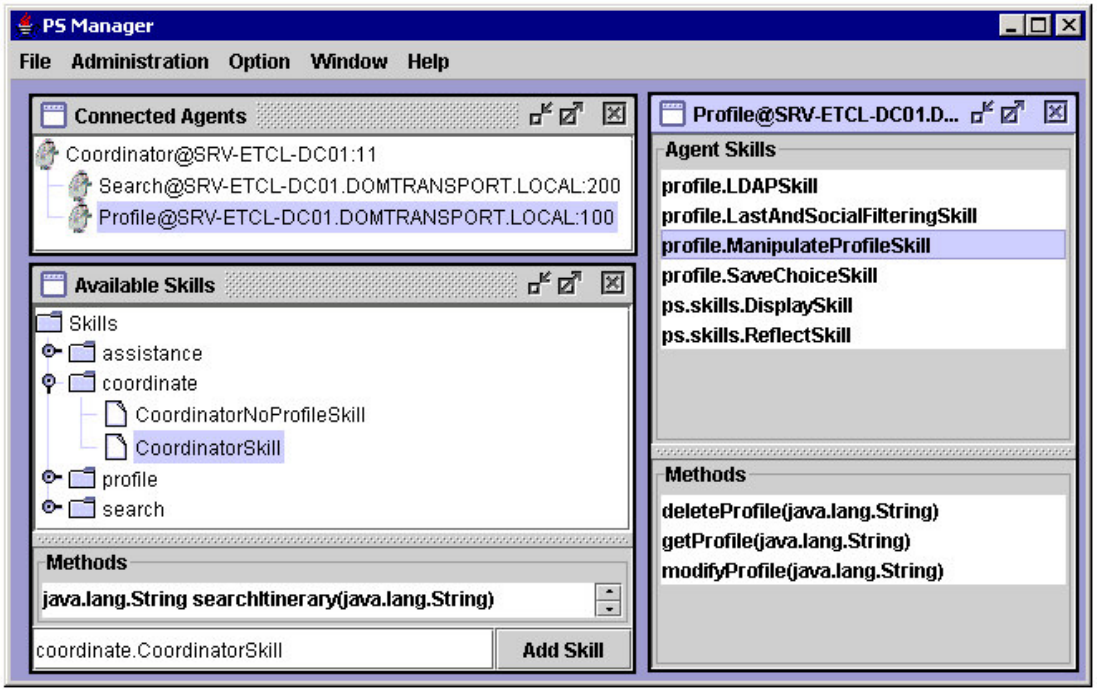


Figure 5.21. Association des compétences et déploiement des agents au travers de l'interface d'administration de PerSyst

5.1.7. Intégration

La communication entre les applications du service de recherche d'itinéraire et PerSyst passe par le module « *PersonnalizeSystem* ». La fonction « *getPerSyst* » a été implémentée pour qu'elle fournisse une référence sur un objet permettant d'invoquer les primitives de PerSyst au travers de message SOAP (Single Objet Access Protocol) [Kadima et Monfort 03]. Les données provenant de PerSyst sont utilisées pour être affichées au niveau des pages Web du service de recherche d'itinéraire (voir Figure 5.22). La primitive « *request* » a été utilisée pour la recherche d'itinéraire et pour permettre à l'utilisateur de visualiser son profil. La primitive « *send* » a été utilisée pour l'envoi à PerSyst du choix de l'utilisateur et pour la modification du profil par l'utilisateur.

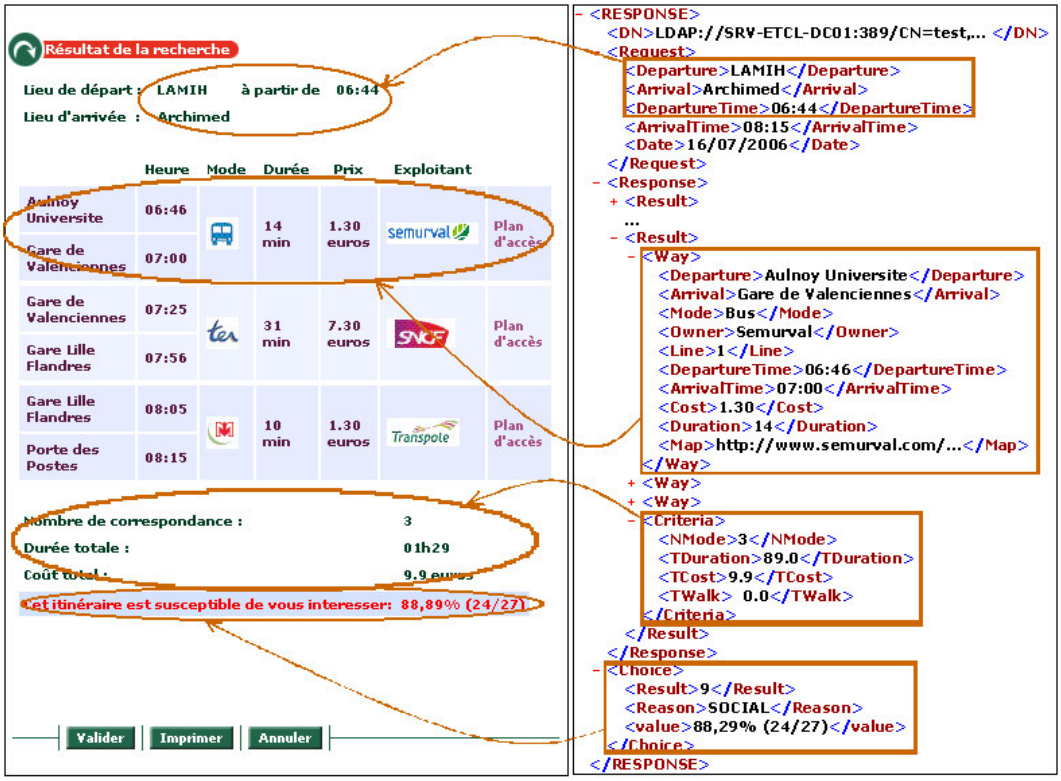


Figure 5.22. Utilisation des données provenant de PerSyst par le service de recherche d'itinéraire

5.1.8. Evaluations

Comme nous l'avons vu au chapitre 3, plusieurs types d'évaluation sont envisageables. A part les tests effectués au fur et à mesure que nous développions le service, dans cette phase, nous avons effectué les évaluations à l'issue de la phase d'intégration. Il s'agit donc d'une évaluation globale combinant la partie SI et la partie SP.

Les *évaluations fonctionnelles* ont été assez rapides puisqu'elles étaient déjà effectuées dans la phase d'implémentation du service. Nous avons surtout vérifié que les fonctionnalités du service à l'issue de la phase d'intégration restaient bien conformes à celles définies dans l'étape de capture des besoins fonctionnels.

Quelques *évaluations techniques* du service ont été effectuées. Nous avons vérifié la conformité du code HTML produit par rapport à Internet Explorer 6.0 et par rapport à Firefox 1.5. D'autres tests sont envisagés, notamment la vérification de la compatibilité par rapport à d'autres versions de ces navigateurs et par rapport à d'autres navigateurs (Netscape, Opera, etc.). En effectuant

des tests au travers de l'outil[65] de validation en ligne des pages HTML, nous avons remarqué que des améliorations du code sont à effectuer pour répondre entièrement aux recommandations du W3C.

Nous avons aussi effectué des tests *d'évaluation de la performance* du service de recherche d'itinéraire personnalisé. Ces tests concernaient le recueil des temps de réponse du service pour une requête de recherche d'itinéraire. Nous avons mesuré le temps de réponse moyen du service par rapport au nombre d'utilisateurs contenus dans la base des utilisateurs. La Figure 5.23 présente les temps de réponse suivant la méthode de filtrage utilisée au niveau des agents de gestion de profil. Une méthode cognitive (filtrage basé sur le profil de l'utilisateur : dernier choix) et deux méthodes sociales (vote majoritaire, bayes) ont été testées. Nous avons expressément choisi ces trois méthodes car elles sont incluses dans le modèle d'activité de l'agent de gestion de profil pour le choix de l'itinéraire. Les résultats obtenus montrent que lorsqu'il s'agit d'un vote majoritaire, les temps de réponse augmentent en fonction du nombre d'utilisateurs enregistrés dans le système mais ces temps restent acceptables (moins d'une seconde pour 500 utilisateurs). Pour le filtrage à base de réseau bayésien (notre méthode de filtrage collaboratif basé sur les préférences et sur les comportements des utilisateurs), le temps de réponse est exponentiel à partir de 100 utilisateurs. Au delà de 100 utilisateurs, le temps de réponse dépasse 3 secondes. Nous pensons améliorer les performances en optimisant l'algorithme implémenté. Pour le filtrage basé sur le profil de l'utilisateur, le nombre d'utilisateurs enregistrés dans le système n'influe pas sur la performance du système. D'autres évaluations de performance sont envisagées, par exemple, l'évaluation de la performance du service par rapport au nombre d'utilisateurs connectés simultanément au service ou l'étude de l'impact de la distribution physique des agents de PerSyst sur la performance globale du système d'information.

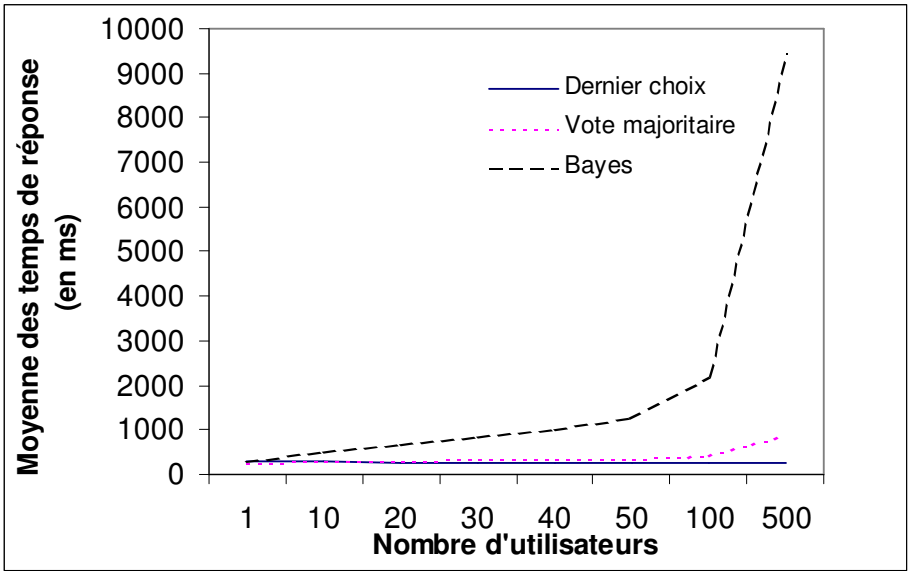


Figure 5.23. Temps de réponse du système en fonction du nombre d'utilisateurs enregistrés au service

Des études pour des *évaluations ergonomiques* sont en cours et de *premières évaluations de la personnalisation* ont été effectuées dans le cadre d'un projet de Mastère Recherche [Boidin 05] que nous avons co-encadré. Les évaluations de personnalisation ont porté sur une étude de la qualité, a priori, de la personnalisation fournie à l'utilisateur. Il s'agit d'une étude qualitative des méthodes pour la personnalisation de l'information transport. Les méthodes étudiées concernaient deux méthodes de filtrage collaboratif (la méthode de similarité des vecteurs et la méthode de Pearson ; voir chapitre 1, §1.1.4.2). La Figure 5.24 donne les erreurs de prédiction moyennes en fonction du nombre d'utilisateurs, pour les deux méthodes, avec une base de données comprenant 10 requêtes. Nous envisageons de compléter ces évaluations de la personnalisation dans nos travaux ultérieurs.

[65] http://validator.w3.org/

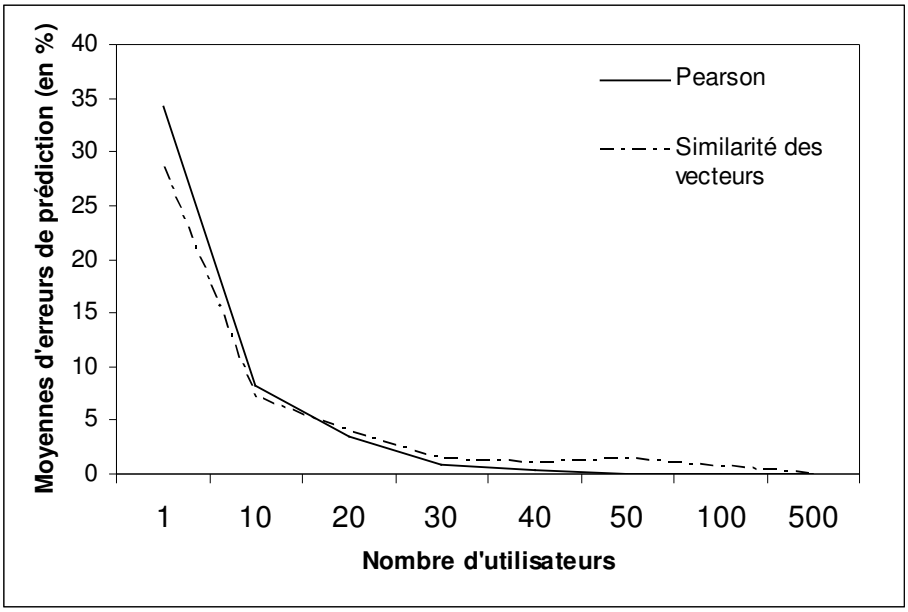


Figure 5.24. Erreurs de prédiction moyenne en fonction du nombre d'utilisateurs pour les méthodes de Pearson et de similarité des vecteurs

5.2. Agenda transport

Dans cette partie, nous décrivons un exemple d'application de PerMet utilisant PerSyst pour la personnalisation d'un service déjà existant. Il s'agit du service d'agenda personnel. L'objectif est d'étendre le service d'agenda personnel fourni par la plate-forme MASC pour proposer aux usagers des transports un nouveau service pouvant leur faciliter la planification et l'organisation de leur déplacement [Ben Lazreg 04]. Le service que nous appelons « agenda transport » consiste à fournir l'itinéraire associé à une planification d'un événement (rendez-vous, sorties, missions, emploi du temps, etc.) par l'utilisateur dans son agenda personnel. L'itinéraire doit correspondre aux préférences et aux attentes de l'utilisateur. Ce service utilise donc le même type de personnalisation que celui utilisé par le service de recherche d'itinéraire décrit précédemment. Il n'a donc pas été nécessaire d'effectuer la partie SP de la méthode PerMet (voir Tableau 5.1).

	Ignorer	Réduire	Normal	Développer
Partie SI				
Analyse du service			√	
Conception du service		√		
Implémentation du service		√		
Partie SP				
Analyse des agents	√			
Conception des comportements	√			
Implémentation des comportements	√			
Partie commune				
Intégration		√		
Evaluation				√

Tableau 5.1. Adaptation des phases de la méthode PerMet pour le développement de l'agenda transport

5.2.1. Analyse et spécification du service

Le service d'agenda transport se basant sur l'agenda de MASC, l'analyse et la spécification du service (capture des besoins fonctionnels, analyse fonctionnelle, capture des besoins techniques, conception générique) ont été largement réduites. En effet, dans l'étape de capture des besoins fonctionnels, seules quelques fonctionnalités supplémentaires ont été nécessaires, en dehors des fonctionnalités de base d'un agenda personnel. La Figure 5.25 fournit les fonctionnalités supplémentaires nécessaires à la réalisation de l'agenda transport. Les autres fonctionnalités (ajout d'un événement, suppression d'un événement, modification d'un événement, visualisation de l'agenda standard, etc.) sont assurées par l'agenda de MASC et ne sont donc pas modélisées ici. Dans l'étape d'analyse fonctionnelle, il n'y a pas eu besoin de définir le modèle de données échangées entre le service d'agenda transport et PerSyst (nous avons réutilisé le même modèle que celui spécifié pour le service de recherche d'itinéraire). Les modèles de capture des besoins techniques et de conception générique restent les mêmes que ceux du service de recherche d'itinéraire.

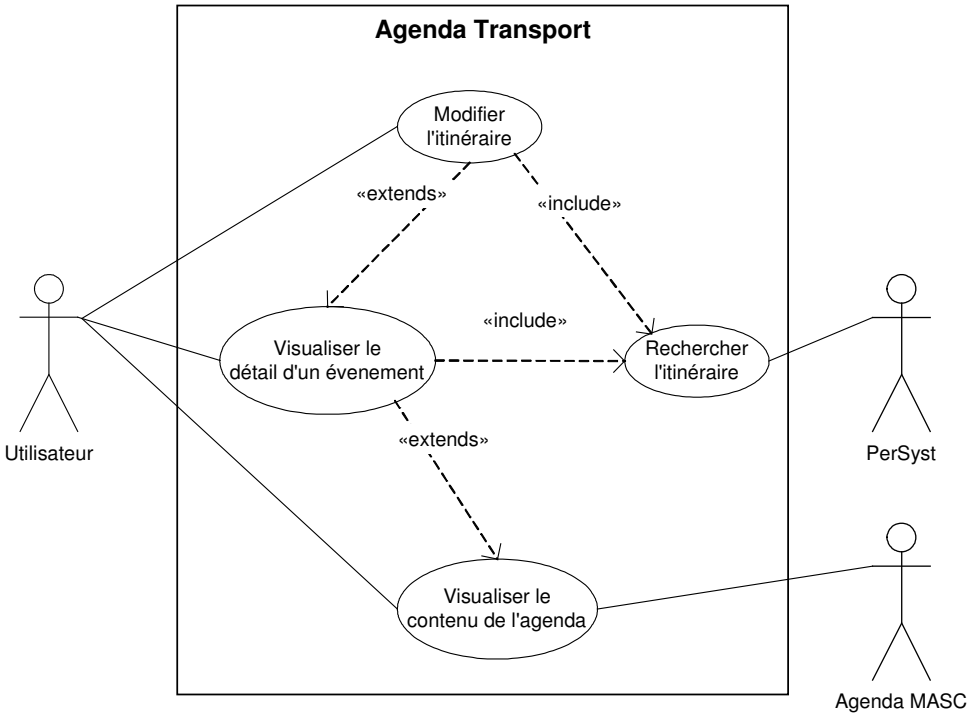


Figure 5.25. Diagramme de cas d'utilisation pour l'agenda transport

5.2.2. Conception du service

Nous avons réutilisé le plus possible l'existant. Seules deux classes (*TransportPlanning* et *frmTransportPlanning*) sont nouvelles parmi les neuf classes nécessaires à la conception du service (voir Figure 5.26). Le taux de réutilisation[66] de l'existant est donc de 77,78%. Les classes *Planning* et *frmPlanning* proviennent du service d'agenda standard fourni par MASC. Les classes *PersonnalizeSystem*, *Choice*, *ResultSearch*, *frmResultSearch*, etc. proviennent du service de recherche d'itinéraire personnalisé.

[66] taux de réutilisation =(1- nombre de classes nouvelles/nombre de classes nécessaires)*100

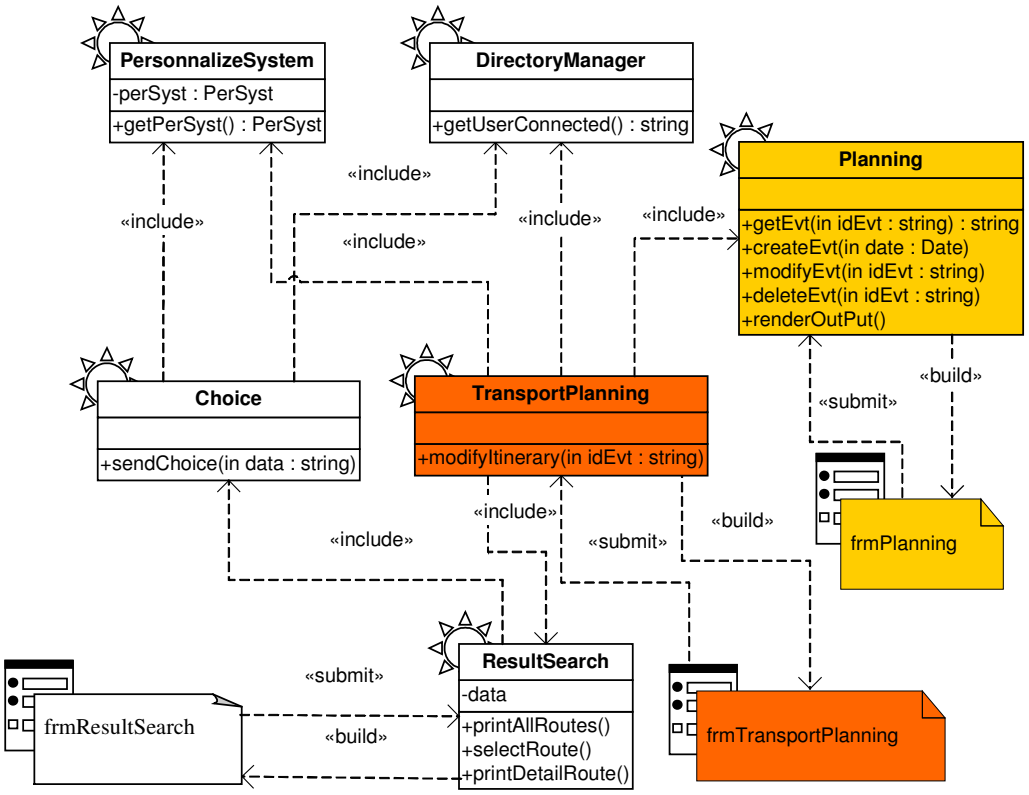


Figure 5.26. Diagramme de classes de conception pour le service d'agenda transport

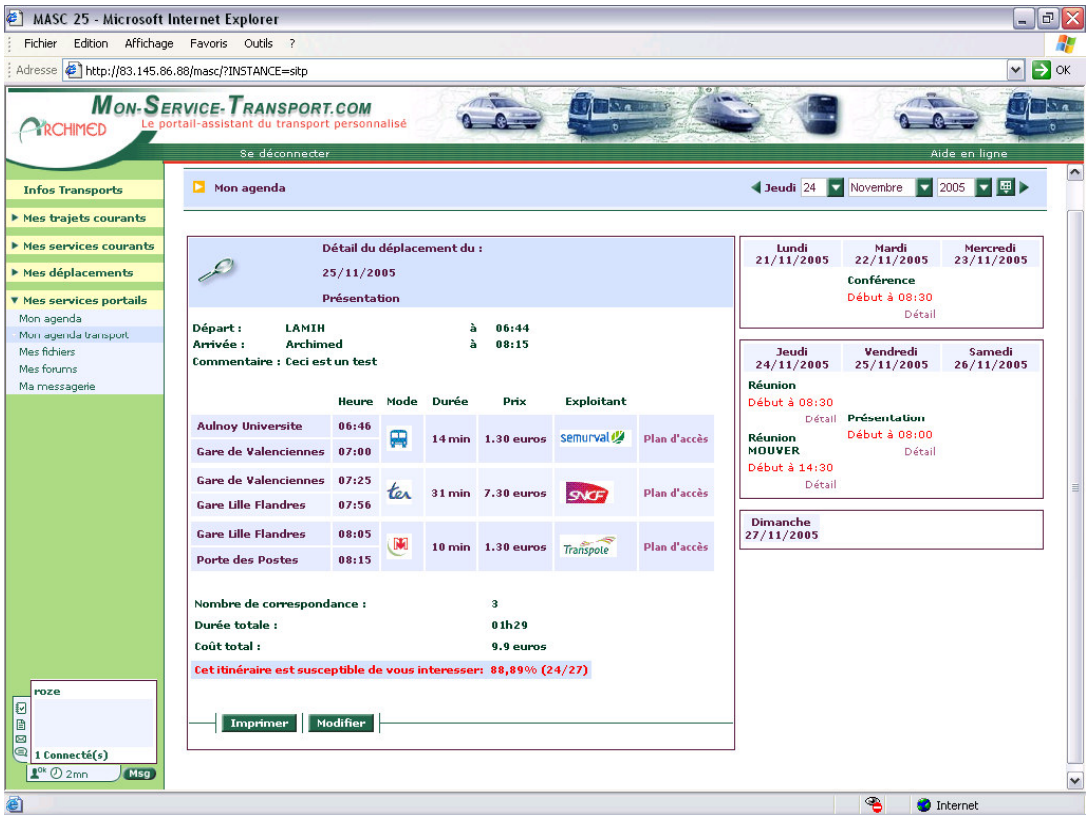


Figure 5.27. Page de visualisation de l'agenda transport

5.2.3. Implémentation du service

Pour rester homogène avec les technologies déjà utilisées avec le service d'agenda de MASC, la page *TransportPlanning* a été implémentée en ASP et l'affichage (*frmTransportPlanning*) a été réalisé grâce à des transformations XSLT. La Figure 5.27 présente la page de visualisation de l'agenda transport.

5.2.4. Analyse des agents, Conception des comportements des agents et Implémentation des comportements des agents

Il n'a pas été nécessaire d'effectuer ces trois phases puisque le service d'agenda transport utilise le même type de personnalisation que le service de recherche d'itinéraire.

5.2.5. Intégration

Comme pour le service de recherche d'itinéraire, la communication entre les applications du service d'agenda transport et PerSyst est effectuée au travers de messages SOAP en passant par le module « *PersonnalizeSystem* ». La primitive « *request* » a été utilisée pour la recherche d'itinéraire personnalisé. Les données nécessaires à la formulation de la requête sont extraites au travers des informations fournies par l'utilisateur pour remplir son agenda personnel. Ainsi le lieu d'arrivée, la date de départ et l'heure d'arrivée sont extraits respectivement des champs « emplacement de l'événement », « date de début » et « heure de début de l'événement » (voir Figure 5.28). Le lieu de départ est déduit par rapport aux planifications précédentes de l'utilisateur : le système suppose que le lieu de départ de l'utilisateur correspond au lieu d'arrivée l'événement précédent.

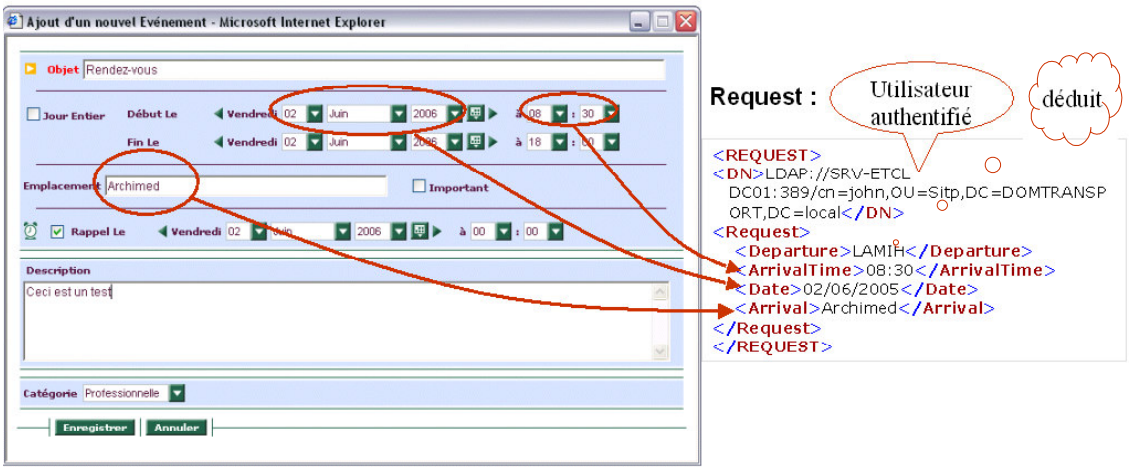


Figure 5.28. Utilisation des données liées à un événement de l'agenda pour l'envoi d'une requête de recherche d'itinéraire personnalisé à PerSyst

5.2.6. Evaluations

Des études pour l'évaluation de ce service sont en cours. Ces évaluations ne portent pas seulement sur l'agenda transport présenté ci-dessus. Elles portent aussi sur un autre agenda transport développé[67] spécifiquement pour être accessible sur PDA dans le cadre du projet PREDIT MOUVER.PERSO (Mobilité et mUltimodalité Voyageurs Etudiants en Région Nord-Pas de Calais) [Grislin-Le Strugeon et al., 06a] (voir Annexe C).

[67] Le développement a été effectué suivant la méthode PerMet en utilisant notre système de personnalisation PerSyst.

5.3. Information personnalisée des perturbations

Dans cette partie, nous décrivons brièvement un exemple d'application de PerMet utilisant PerSyst pour le développement d'un service personnalisé prenant en compte différents canaux de communication (SMS, mail, web) et différentes plates-formes d'interaction (PC, téléphone portable).

Le principe du service consiste à mettre à disposition de l'usager des transports des informations relatives aux perturbations (travaux, retards, grèves…) du réseau correspondant à ses besoins ou à ses attentes. L'objectif est de choisir le meilleur canal pour acheminer l'information en prenant en compte les besoins spécifiques de chaque utilisateur et ses plates-formes d'interaction.

5.3.1. Analyse et spécification du service

Il s'agit d'effectuer un *push* [Bonnet et Macary 97] à un usager des transports pour l' informer des perturbations. Ces informations peuvent être mises à la disposition de l'usager par SMS, par mail ou via un portail Web. Si l'information est fournie au travers d'un portail web l'utilisateur a la possibilité de rechercher un itinéraire alternatif (voir Figure 5.29).

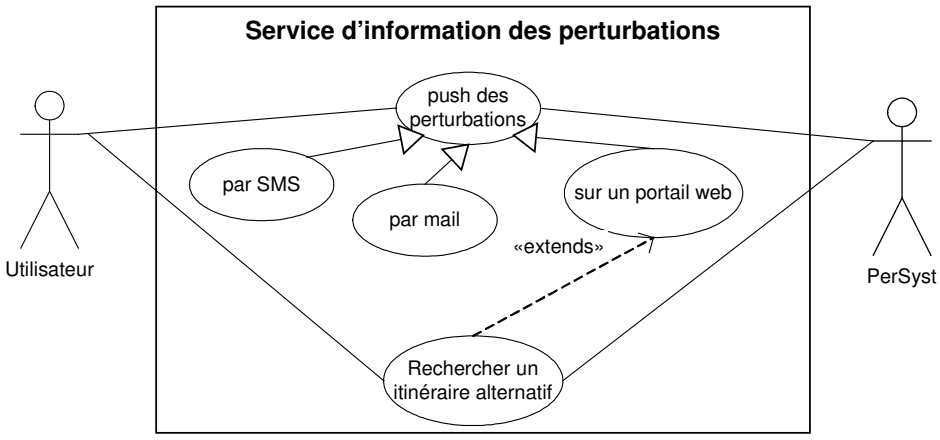


Figure 5.29. Diagramme des cas d'utilisation du service d'information des perturbations

Dans cette phase, le modèle objet pour l'analyse du service a été défini. Il a été aussi nécessaire de définir un autre modèle de données pour la communication des perturbations de PerSyst au portail web « mon service transport ». Pour la recherche d'itinéraire alternatif personnalisé, c'est le service de recherche d'itinéraire personnalisé qui a été utilisé (réutilisation donc du modèle de données défini pour la recherche d'itinéraire).

5.3.2. Conception du service

Dans la phase de conception du service, nous avons réutilisé le plus possible les objets existants. Comme on peut le voir sur le diagramme de classe de conception (Figure 5.30), seules deux nouvelles classes (*Disturbance* et *frmDisturbance*) ont été introduites.

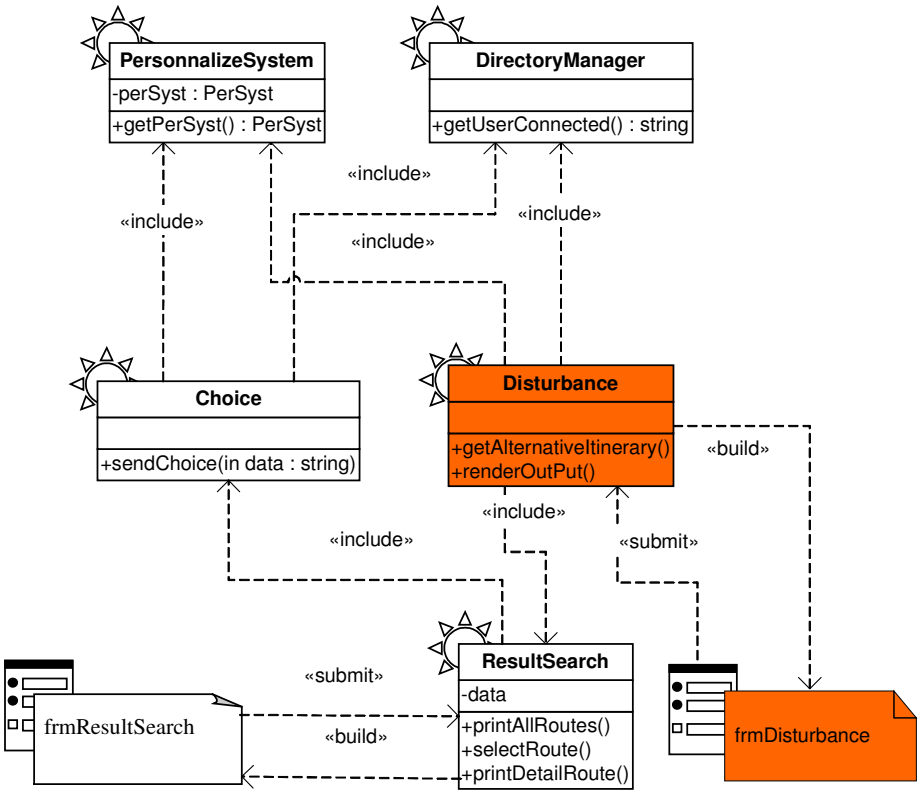


Figure 5.30. Diagramme de classes de conception pour le service d'information des perturbations

5.3.3. Implémentation du service

La fourniture des informations de perturbation sur un portail web a été implémentée sous forme d'un *portlet*[68] (voir Figure 5.31) pouvant être affiché sur la page d'accueil de l'utilisateur. La présentation du mail à envoyer à l'utilisateur est codé en HTML. Afin de pouvoir tester le service, nous avons développé un émulateur pour simuler l'envoi de SMS sur un téléphone portable. Cet émulateur permet de simuler un téléphone portable associé à un utilisateur. Un serveur (émulé) assure l'envoi d'un message à un téléphone donné qui l'affiche sous forme de SMS.

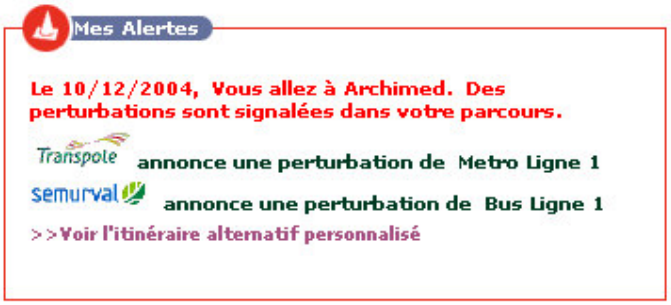


Figure 5.31. Portlet de présentation des informations de perturbation

[68] « *Un portlet est une application informatique qui s'exécute côté serveur que l'on peut placer dans un portail web, qui sert alors de conteneur. Un portlet traite les requêtes d'une tâche ou d'un service donné et génère dynamiquement le contenu web affiché à l'utilisateur.* » [http://fr.wikipedia.org]

5.3.4. Analyse des agents

A partir de l'analyse du service sous l'angle de la personnalisation (voir Figure 5.32) nous avons identifié les différents agents pouvant intervenir dans le développement du service, les accointances des agents et leurs comportements. Pour la recherche des perturbations, nous avons supposé que chaque exploitant (Transpole, Semurval ou SNCF) gère ses perturbations (ceci permet d'illustrer le fonctionnement de PerSyst dans un cas où les informations sont distribuées). En suivant les règles d'identification des agents proposées par la méthode PerMet, nous avons obtenu trois agents pour la recherche des perturbations (Transpole, Semurval, SNCF) et trois agents pour la gestion des plates-formes d'interaction (SMS, mail, portail web). Les agents Transpole, Semurval et SNCF font partie des agents de recherche d'information. Nous avons optés pour le fait de gérer leur coordination au niveau de l'agent de recherche d'information. Et les agents SMS, mail et portail web ont pour rôle la personnalisation de l'interaction par rapport au profil de l'utilisateur. Leur coordination est gérée au niveau de l'agent de gestion de profil.

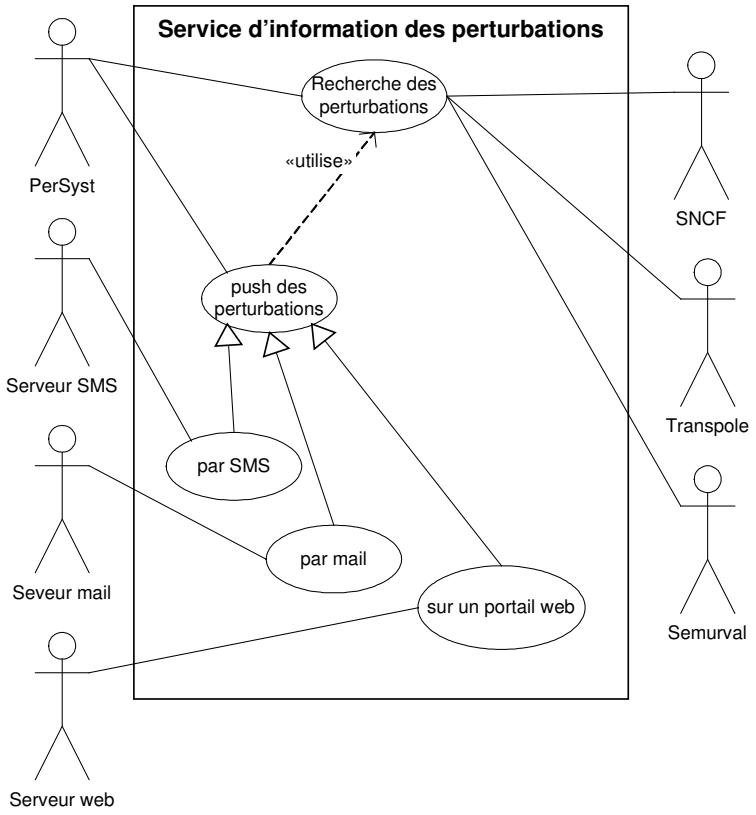


Figure 5.32. Diagramme des cas d'utilisation du service d'information des perturbations vu sous l'angle de la personnalisation

5.3.5. Conception des comportements des agents

Dans cette phase, pour chaque modèle d'agent, nous avons affiné et détaillé les comportements des agents. Par exemple, pour l'agent de gestion de profil (coordinateur des agents Mail, SMS et portail web) nous avons défini ses activités (voir Figure 5.33) lorsqu'il reçoit une information de perturbation. Lorsque l'agent de gestion de profil reçoit une information (provenant de l'agent de recherche), il effectue un premier filtrage (des utilisateurs) pour n'avoir que les utilisateurs concernés par cette information. Ensuite, il isole les plates-formes d'interaction les plus adéquates à l'information, parmi les plates-formes dont dispose l'utilisateur, pour l'envoi de l'information. Si

l'utilisateur dispose de plusieurs plates-formes d'interaction qui peuvent recevoir cette information, l'agent choisit la plate-forme par rapport aux préférences de l'utilisateur[69] [Anli et Abed 06].

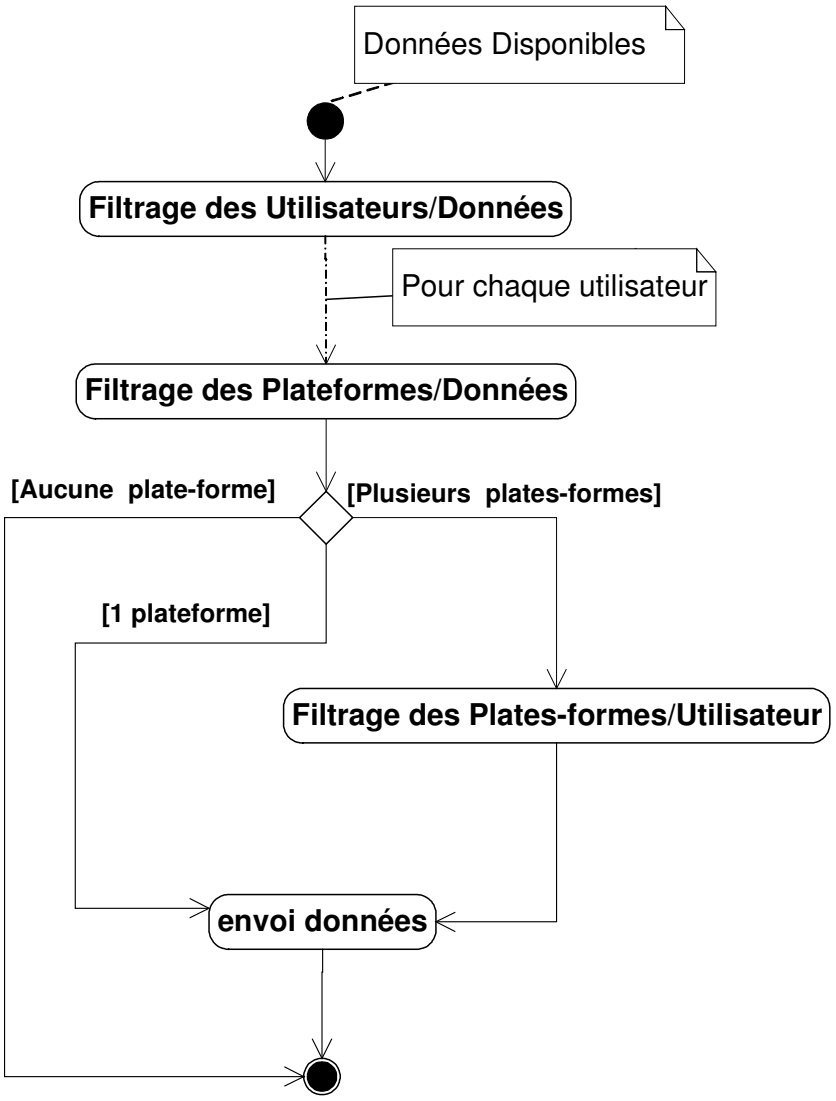


Figure 5.33. Modèle d'activités pour le push d'information prenant en compte les utilisateurs, les données et les plates-formes d'interaction

5.3.6. Implémentation des comportements des agents

Les compétences des agents ont été implémentées et testées au fur et à mesure de leur développement. Puisque les agents de gestion de profil et de recherche d'information étaient déjà créés (pendant le développement du service de recherche d'itinéraire), il a juste fallu leur ajouter leurs nouvelles compétences au travers de l'interface d'administration de PerSyst. La création des nouveaux agents (SNCF, Transpole, Mail, SMS, etc.), l'association de leurs compétences et leur déploiement ont aussi été réalisés au travers de l'interface d'administration de PerSyst. Après le déploiement des agents, des tests de vérification du bon fonctionnement du SMA composant PerSyst ont été effectués par une simulation des données qui devraient provenir des applications externes (notamment du portail web MASC).

[69] Le filtrage des plates-formes par rapport à l'utilisateur n'est pas encore implémenté.

5.3.7. Intégration

La communication entre le *portlet* d'information des perturbations et PerSyst est effectuée au travers de messages SOAP en passant par le module « *PersonnalizeSystem* » défini pendant le développement du service de recherche d'itinéraire. La primitive « *request* » a été utilisée pour la recherche des perturbations et pour la recherche des itinéraires alternatifs.

L'envoi de mail est effectué au niveau de l'agent *Mail* qui se connecte à un serveur de mail en utilisant le protocole SMTP[70].

Comme nous l'avons souligné plus haut, l'envoi de SMS est simulé au travers de notre émulateur de téléphone portable. Notre émulateur est à base d'agents ce qui a facilité son intégration avec PerSyst. En effet, l'agent *SMS* communique avec le serveur de SMS comme s'il communiquait à un agent. L'intégration de PerSyst à un serveur de SMS réel ne devrait pas comporter de difficulté particulière car des services web[71] permettent l'interaction en messages SOAP pour l'envoi de SMS à un téléphone particulier.

5.3.8. Evaluations

Les évaluations complètes de ce service ne sont pas encore effectuées. Seules les évaluations fonctionnelles et quelques évaluations de la personnalisation ont été réalisées. Les évaluations de la personnalisation concernaient surtout des tests de vérification de la bonne mise à disposition de l'information à l'utilisateur concerné. Il s'agissait de faire planifier un voyage à un utilisateur et de simuler une perturbation sur un trajet qu'il a prévu de prendre. Ensuite, nous vérifions si le système lui indiquait bien qu'il y avait une perturbation sur son itinéraire. Nous avons, bien évidemment, vérifié que seul l'utilisateur concerné était prévenu de la perturbation. Nous avons aussi vérifié si la personnalisation prenait bien en compte les différentes plates-formes dont dispose l'utilisateur. Par exemple, nous avons simulé deux utilisateurs qui ont choisi un même itinéraire comportant plusieurs changements. L'un des utilisateurs a planifié son voyage deux jours après la date de l'autre. Pendant que le premier utilisateur est en cours de déplacement, nous avons généré une perturbation sur son prochain changement. Nous avons bien constaté que le système a envoyé un SMS (voir Figure 5.34) à l'utilisateur qui est en cours de déplacement et a envoyé un mail (voir Annexe A) à l'autre utilisateur (en se basant sur l'hypothèse que l'utilisateur consulte ses mails au moins une fois par jour et pourra prendre connaissance de l'information de perturbation à temps).

Figure 5.34. Message SMS sur l'émulateur de téléphone portable

[70] Simple Mail Transfer Protocol
[71] Voir par exemple, http://www.esendex.com/fr

Conclusion

Nous avons présenté dans ce chapitre une application de la méthode PerMet utilisant le système de personnalisation PerSyst. Cette application porte sur le développement d'un système d'information personnalisé dans le domaine des transports terrestres de personnes. Le système d'information a été décomposé en trois services : *recherche d'itinéraire*, *agenda transport* et *information des perturbations*.

Le service de recherche d'itinéraire a servi pour la validation de PerMet et de PerSyst pour le développement d'un nouveau service personnalisé. Il s'agit du service de recherche d'itinéraire classique généralement présent dans les systèmes d'information transport. La particularité de ce service se situe au niveau de la fourniture des résultats qui sont personnalisés par rapport aux préférences de l'utilisateur. Deux *agents applicatifs* ont été définis au niveau de PerSyst : l'agent de recherche qui a pour rôle la recherche des itinéraires et l'agent de gestion de profil qui s'occupe de la personnalisation des itinéraires par rapport à l'utilisateur. Nous avons appliqué la modélisation de l'utilisateur, la collecte implicite des préférences et la méthode de filtrage collaboratif basé sur les préférences et les comportements des utilisateurs proposées au chapitre 4 pour la personnalisation des itinéraires.

Le service d'agenda transport a illustré la personnalisation d'un service existant. Ce service utilise un type de personnalisation déjà existant dans le but de valider la réutilisation de l'existant au sein de la méthode PerMet. Le principe du service consiste à fournir automatiquement l'itinéraire à un utilisateur suivant son planning par rapport à un agenda standard. L'utilisateur remplit normalement son planning sans se soucier du transport. L'application détecte si par rapport à ce planning, l'utilisateur est amené à se déplacer en comparant l'endroit où se trouverait l'utilisateur et le lieu où il a planifié l'événement. Si c'est le cas, le système envoie une requête à PerSyst pour recevoir un itinéraire personnalisé. Le développement de ce service n'a pas nécessité d'effort supplémentaire au niveau de la personnalisation puisqu'il utilise le même type de personnalisation que celui du service de recherche d'itinéraire.

Le service d'information des perturbations consiste en la mise à disposition personnalisée des perturbations dans les transports terrestres de personnes. Ce service utilise un autre type de personnalisation, l'objectif étant de démontrer la capacité de PerSyst à intégrer différents types de personnalisation. Ce service illustre aussi la capacité de PerSyst à effectuer de la personnalisation en prenant en compte les canaux de communication et les plates-formes d'interaction que disposent l'utilisateur. Six nouveaux *agents applicatifs* ont été ajoutés à PerSyst. Trois agents sont dédiés à la recherche des perturbations, chacun s'occupant d'un site (SNCF, Semurval ou Transpole) et trois autres agents sont destinés à la personnalisation de l'interaction, chacun s'occupant d'un canal de communication (Mail, Web et SMS). Les trois agents de recherche des perturbations sont coordonnés par l'agent de recherche d'information. Et les trois agents dédiés aux canaux de communication sont coordonnés par l'agent de gestion de profil qui intègre notre modèle de filtrage prenant en compte les données, les utilisateurs et les plates-formes d'interaction.

L'application « Mon service transport » nous a permis d'illustrer l'utilisation concrète de notre méthode pour le développement d'un système d'information personnalisé. Les trois exemples de services développés dans le cadre de cette application montrent que notre méthode et notre système de personnalisation prennent en compte différents types de personnalisation, permettent la réutilisation de l'existant et favorisent un développement incrémental. Cela a permis de valider notre méthode et d'évaluer certaines caractéristiques (évolutivité, distributivité, etc.) de notre système de personnalisation.

Différents points peuvent faire l'objet d'études ou d'améliorations, tant en ce qui concerne la méthode PerMet qu'en ce qui concerne le système de personnalisation PerSyst et l'application « Mon service transport ». Ces études et ces améliorations font partie de nos perspectives de recherche.

Bibliographie du chapitre 5

*[Anli et Abed 06] Anli A. et Abed M. PerSyst : un Système de Personnalisation de l'information transport multimodale. *CIFA'2006, Conférence Internationale Francophone d'Automatique (30 mai-1 juin)*, Bordeaux, France, mai 2006.

*[Anli et al., 04] Anli A., Petit-Rozé C. and Grislin-Le Strugeon E. User-based Decision Support System. *In International Conference on Advances in Intelligent Systems - Theory and Applications in cooperation with IEEE Computer Society*, AISTA'2004, Luxembourg, novembre 2004.

[Ben Lazreg 04] Ben Lazreg I. *Conception de système d'information au service des usagers dans les transports terrestres de personnes.* Rapport de DEA, Université de Valenciennes et du Hainaut-Cambrésis, juillet 2004.

[Boidin 05] Boidin E. *Apprentissage automatique pour la personnalisation : Application aux transports terrestres de personnes*. Rapport de DEA, Université de Valenciennes et du Hainaut-Cambrésis, juillet 2005.

[Bonnet et Macary 97] Bonnet C. et Macary J. F. *Technologie PUSH*. Paris : Eyrolles, 1997.

[Conallen 00] Conallen J. *Concevoir des applications web avec UML*. Paris : Eyrolles, 2000.

[Gamma et al., 99] Gamma E., Helm R. et Johnson R. *Design patterns catalogue de modèles de conception réutilisables*. Paris : Vuibert, 1999.

*[Grislin-Le Strugeon et al., 06a] Grislin-Le Strugeon E., Petit-Rozé C., Anli A., Abed M. et Kolski C. *Mouver.Perso : Mobilité et mUltimodalité Voyageurs Etudiants en Région Nord Pas de Calais – Système d'information multimodale personnalisée*. Rapport intermédiaire de projet PREDIT, LAMIH, Valenciennes, janvier 2006.

[Kadima et Monfort 03] Kadima H. et Monfort V. *Les web services : Techniques et outils XML, WSDL, SOAP, UDDI, Rosetta, UML*. Paris : Dunod, 2003.

*[Petit-Rozé et al., 04] Petit-Rozé C., Anli A., Grislin-Le Strugeon E., Abed M., Uster G. et Kolski C. Système d'Information Transport Personnalisée à base d'agents logiciels. *Génie Logiciel*, 70, pp. 29-38, 2004.

[Zimmermann et al., 05] Zimmermann A., Specht M. and Lorez A. Personalization and Context Management. *User Modeling and User-Adapted Interaction*, 15, pp. 275-302, 2005.

CONCLUSION GENERALE ET PERSPECTIVES DE RECHERCHE

Conclusion générale

Les travaux de recherche présentés dans ce mémoire se situent dans la thématique de la personnalisation en interaction homme-machine. Ils contribuent au développement des systèmes d'information personnalisés.

Après une introduction des principaux termes rencontrés dans le domaine, nous avons étudié les approches et méthodes existantes pour la personnalisation. Trois processus sont récurrents dans une activité de personnalisation : le processus de collecte d'information sur l'utilisateur pour connaître ses préférences et ses besoins, le processus de modélisation de ces préférences (modèle utilisateur) et le processus d'exploitation du modèle utilisateur pour la personnalisation (méthode de personnalisation). Différentes approches sont fournies dans la littérature pour réaliser chacun de ces processus.

Nous avons aussi étudié différents systèmes de personnalisation. Cette étude a montré que les systèmes existants sont généralement destinés à un type particulier de personnalisation et incorporent des méthodes de personnalisation fixes. Les rares systèmes qui puissent répondre à des critères comme la distribution ou l'évolutivité sont construits à partir d'une architecture multi-agents.

C'est pourquoi une étude des architectures des systèmes de personnalisation à base d'agents logiciels a été effectuée dans le deuxième chapitre. Certains modèles d'agents[72] (recherche d'information, gestion de profil, interaction avec l'utilisateur et coordination) sont récurrents dans la majorité des architectures à base d'agents des systèmes de personnalisation. D'autres modèles d'agents apparaissent suivant les objectifs et l'usage auxquels le système de personnalisation est destiné.

Un autre objectif de ce deuxième chapitre a été de rechercher une méthode de génie logiciel adaptée au développement de système d'information personnalisé. Une méthode adéquate pour le développement de système d'information personnalisé devrait répondre à trois dimensions : la *dimension méthodologie* pour décrire les différentes phases et leurs enchaînements pour le développement de l'application, la *dimension technologie* pour prendre en compte les techniques utilisées pour la personnalisation comme la programmation à base d'agents logiciels, le mode client-serveur ou l'utilisation de XML et la *dimension représentation* pour la spécification et la modélisation de l'application. L'étude des méthodes représentatives des domaines d'application susceptibles d'intervenir dans le développement de système d'information personnalisé montre que ces méthodes, prises séparément, ne répondent pas entièrement à tous les critères associés aux trois dimensions ci-dessus. Cependant leurs adaptations et leur intégration nous ont paru pertinentes pour la construction d'une nouvelle méthode adaptée au développement de système d'information personnalisé.

Ceci fût l'objet du troisième chapitre qui a proposé une méthode pour l'analyse, la conception et la modélisation de système d'information personnalisé. Cette méthode, appelée PerMet (PERsonalization METhodology), permet aussi bien la mise en place d'un nouveau système d'information personnalisé que la personnalisation d'un système d'information déjà existant. Elle sépare le système d'information vu comme un ensemble de services du système de personnalisation. PerMet propose un modèle de développement itératif, incrémental et permet une réalisation parallèle des phases spécifiques liées au développement des services et des phases spécifiques liées à la personnalisation. Les formalismes de modélisation sont basés sur ceux d'UML et de ses extensions. PerMet insiste sur la nécessité d'un système de personnalisation évolutif et distribué pouvant prendre en compte différents types de personnalisation.

Le quatrième chapitre a exposé une architecture et une approche pour la conception d'un système de personnalisation générique. Le système de personnalisation appelé PerSyst

[72] Ces modèles d'agents ne portent pas toujours le même nom mais s'identifient assez facilement au travers des rôles qu'ils jouent au sein du système.

(PERsonalization SYSTem) peut être utilisé conjointement avec la méthode PerMet pour le développement d'un système d'information personnalisé. PerSyst est construit à partir d'une architecture multi-agents ce qui le dote des caractéristiques de mobilité et d'apprentissage lui permettant la distributivité et l'évolutivité de ses entités et des caractéristiques d'adaptabilité, d'autonomie et d'assistance lui permettant de prendre en compte différents types de personnalisation.

PerSyst distingue les *agents applicatifs* qui sont définis lors d'une intégration de PerSyst dans un système d'information (comme les agents de gestion de profil utilisateur, de recherche d'information, d'assistance, etc.) *des agents systèmes* qui sont *l'agent de communication, l'agent de coordination* et *l'agent d'administration*.

L'agent de coordination sert de pont de communication entre les différents agents présents dans PerSyst. Il assure la coordination des messages échangés entre les différents agents applicatifs, la coordination des messages échangés entre les applications externes et les agents applicatifs et la coordination des messages échangés entre le développeur et les agents applicatifs.

L'agent de communication permet la traduction des requêtes envoyées par les applications externes en des messages compréhensibles par l'agent de coordination et vice versa. L'architecture de cet agent comprend deux parties : la première partie se charge de la communication avec les applications externes et la deuxième partie s'occupe de l'interaction avec l'agent de coordination.

L'agent d'administration permet la gestion et l'administration de PerSyst. Il permet à un utilisateur d'interagir graphiquement avec les agents logiciels en exécution composant PerSyst. L'utilisateur dispose d'une vue sur les agents actifs, sur leur organisation et sur leur localisation, sur les compétences et sur le contenu de leurs échanges. Il peut créer et ôter des agents, ainsi qu'agir sur chacun d'eux pour lui supprimer ou lui ajouter des compétences. Il a aussi la possibilité de connecter un agent existant ou de l'enlever de PerSyst.

Le quatrième chapitre propose aussi des modèles et des méthodes génériques, prenant en compte le contexte d'interaction, pour la *représentation du profil utilisateur*, pour la *collecte d'information* sur l'utilisateur et pour *l'exploitation des profils* utilisateur. La méthode de collecte d'information sur l'utilisateur est une méthode d'apprentissage automatique qui se base sur l'historique de l'interaction de l'utilisateur pour calculer ses préférences. Deux méthodes de personnalisation (exploitation des profils utilisateur pour la personnalisation) à base de réseaux bayésiens, sont proposées. La première méthode effectue un filtrage collaboratif en analysant les préférences (données pondérées) et les comportements (historique) des utilisateurs. Le principe consiste à prédire le comportement d'un utilisateur par comparaison des comportements des autres utilisateurs ayant les mêmes préférences pour un contexte donné. La deuxième méthode utilise le même type de raisonnement que précédemment sauf qu'elle se base uniquement sur les comportements (historique) des utilisateurs.

La méthode PerMet et le système de personnalisation PerSyst ont été appliqués pour le développement d'un système d'information personnalisé dans le domaine des transports terrestres de personnes. Il s'agit ici d'aider les usagers des transports collectifs dans leur choix d'itinéraires, de les informer des perturbations éventuelles et de les guider tout au long de leur déplacement. Trois services ont été développés *: recherche d'itinéraire*, *agenda transport* et *information des perturbations*.

Le service de recherche d'itinéraire a servi pour la validation de PerMet et de PerSyst pour le développement d'un nouveau service personnalisé. Il s'agit du service de recherche d'itinéraire classique généralement présent dans les systèmes d'information transport. La particularité de ce service se situe au niveau de la fourniture des résultats qui sont personnalisés par rapport aux préférences de l'utilisateur. Deux agents applicatifs ont été définis au niveau de PerSyst : l'agent de recherche qui a pour rôle la recherche des itinéraires et l'agent de gestion de profil qui s'occupe de la personnalisation des itinéraires par rapport à l'utilisateur. Nous avons appliqué la modélisation de l'utilisateur, la collecte implicite des préférences et la méthode de filtrage collaboratif basé sur les préférences et les comportements des utilisateurs proposées au chapitre 4 pour la personnalisation des itinéraires.

Le service d'agenda transport a illustré la personnalisation d'un service existant. Ce service utilise un type de personnalisation déjà existant dans le but de valider la réutilisation de l'existant au sein de la méthode PerMet. Le principe du service consiste à fournir automatiquement l'itinéraire à un utilisateur suivant son planning par rapport à un agenda standard. L'utilisateur remplit normalement son planning sans se soucier du transport. L'application détecte si par rapport à ce planning, l'utilisateur est amené à se déplacer en comparant l'endroit où se trouverait l'utilisateur et le lieu où il a planifié l'événement. Si c'est le cas, le système envoie une requête à PerSyst pour recevoir un itinéraire personnalisé. Le développement de ce service n'a pas nécessité d'effort supplémentaire au niveau de la personnalisation puisqu'il utilise le même type de personnalisation que celui du service de recherche d'itinéraire (100% de réutilisation au niveau de la partie SP de PerMet). Le service personnalisé étant construit à partir d'un service déjà existant, il y a eu une forte réutilisation de l'existant (environ 80% de réutilisation des classes de conception, au niveau de la partie SI de PerMet).

Le service d'information des perturbations consiste en la mise à disposition personnalisée des perturbations dans les transports terrestres de personnes. Ce service utilise un autre type de personnalisation. L'objectif étant de démontrer les capacités de PerMet et de PerSyst à intégrer différents types de personnalisation. Ce service illustre aussi la capacité de PerSyst à effectuer de la personnalisation en prenant en compte les canaux de communication et les plates-formes d'interaction dont dispose l'utilisateur. Six nouveaux agents applicatifs ont été ajoutés à PerSyst. Trois agents sont dédiés à la recherche des perturbations, chacun s'occupant d'un site (SNCF, Semurval ou Transpole) et trois autres agents sont destinés à la personnalisation de l'interaction, chacun s'occupant d'un canal de communication (Mail, Web et SMS). Les trois agents de recherche des perturbations sont coordonnés par l'agent de recherche d'information. Et les trois agents dédiés aux canaux de communication sont coordonnés par l'agent de gestion de profil qui intègre un modèle de filtrage prenant en compte les données, les utilisateurs et les plates-formes d'interaction.

Perspectives de recherche

Différentes perspectives sont envisagées. Ces perspectives portent aussi bien sur la méthode de développement de système d'information personnalisé PerMet que sur le système de personnalisation PerSyst et le système d'information transport personnalisé « mon service transport ».

Perspectives liées à la méthode

La méthode PerMet a été utilisée pour le développement de plusieurs applications. Même si ces applications portent toutes sur la même thématique transport, elles sont relativement différentes et ont été réalisées par différentes personnes. Ces applications ont porté sur le développement d'un assistant sur PC avertissant de perturbations liées au transport, d'un agenda transport accessible sur PC, d'un agenda transport adapté sur PDA et d'une application sur PDA pour la recherche d'itinéraire personnalisé. Hormis le rédacteur de ce mémoire, cette méthode a été utilisée par un étudiant d'IUP2, deux étudiants de Master 2, deux ingénieurs et un docteur. La méthode a été bien adoptée par ces utilisateurs qui disent que la méthode les a bien aidés dans le développement de leurs applications. Cependant pour mieux étudier les avantages et les lacunes de PerMet, nous envisageons d'établir un questionnaire rigoureux pour recueillir les avis des utilisateurs de PerMet.

L'utilisation de PerMet pour le développement de système d'information personnalisé portant sur d'autres domaines est aussi envisagé. Un des secteur d'activité de la société Archimed, co-financeur de la thèse, porte sur le développement des systèmes d'information EPN (Espaces Publiques Numériques) où l'utilisateur a accès à une masse considérable d'informations. Dans ce genre d'application, la personnalisation est une valeur ajoutée certaine servant de critère important dans le choix d'une solution EPN. Beaucoup d'efforts sont donc effectués à ce niveau et l'utilisation d'une méthode rigoureuse et efficace pouvant réduire les coûts de développement et offrant une évolutivité aisée est très appréciable. Il serait donc très intéressant d'utiliser et d'évaluer PerMet dans le développement de ce genre d'applications.

Afin de faciliter l'utilisation de PerMet, il est envisagé à moyen terme de fournir un atelier de génie logiciel supportant PerMet. Il ne s'agit pas de créer un nouvel atelier de génie logiciel. L'idée consiste à utiliser un atelier de génie logiciel extensible (Microsoft Visio©, par exemple) et d'intégrer les modèles et les stéréotypes spécifiques proposés par PerMet. Il faudra ensuite y spécifier les règles nécessaires pour la génération automatique du code informatique adéquat relatif à ces modèles. Cela permettrait de disposer d'un même atelier pour la génération du code aussi bien pour la partie SI que pour la partie SP de la méthode PerMet. A terme, il s'agira de coupler l'atelier de génie logiciel à un environnement de développement (Eclipse ou NetBeans, par exemple) pour que le code généré à partir de la modélisation puisse être directement accessible au travers de l'environnement de développement et que les modifications du code effectuées au sein de l'environnement de développement puissent se répercuter automatiquement au niveau de la modélisation.

Perspectives liées au système de personnalisation

Le système de personnalisation PerSyst a été intégré dans des applications pour le développement de systèmes d'information transport personnalisés. L'une des particularités de PerSyst, c'est qu'il fournit une interface d'administration permettant de créer et de déployer les agents logiciels nécessaires à une application. Cette interface a été appréciée par les différents utilisateurs (ce sont les mêmes utilisateurs de PerMet, présentés ci-dessus). Ces utilisateurs ont souligné sa facilité d'utilisation et son utilité (cela leur a facilité la création et le déploiement de leurs agents) dans le développement de leurs applications. Cette interface a aussi rassuré les utilisateurs qui n'étaient pas familiers des agents logiciels puisqu'ils ont une représentation graphique des agents et peuvent visionner les échanges de messages pendant leur exécution. Bien évidemment, il serait intéressant de disposer d'une communauté d'utilisateurs de PerSyst plus nombreux et travaillant dans des domaines d'application variés pour mieux évaluer PerSyst.

Cependant, suite à ces quelques applications, certaines améliorations possibles du système sont apparues. Effectivement, l'utilisation de PerSyst a montré que parfois le déplacement d'un agent d'une plate-forme à une autre lui fait perdre des compétences. Ceci s'explique par le fait que les activités réalisées pour la mobilité de l'agent ne correspondent pas toujours au modèle d'activités proposé (voir chapitre 4, §4.2.3.3). En effet, pour l'instant, il n'y a pas de contrôle efficace par les agents dans l'exécution de leurs activités. L'apprentissage de compétences est géré au niveau des agents sous forme de tâche exécutée en parallèle des autres tâches. Il arrive donc que l'agent source se soit détruit avant que l'agent destination ait appris toutes les compétences de l'agent source. Une solution intuitive pour y remédier serait d'ajouter un contrôle pour s'assurer que l'agent destination ait appris toutes les compétences avant que l'agent source ne se détruise.

L'un des points faibles actuels de PerSyst réside dans la difficulté de mise à jour d'une compétence déjà ajoutée à un agent. Ce point faible est d'autant plus gênant que l'objectif principal de PerSyst consiste à aider les développeurs dans leurs mise en œuvre d'un système d'information personnalisé. Il est très rare de développer une compétence d'un agent qui soit immédiatement correcte et opérationnelle. Le développeur a généralement besoin de tester plusieurs versions d'une même compétence pour aboutir à une version relativement stable et robuste. Lorsqu'une compétence est ajoutée à un agent, elle est mise en cache[73] et c'est cette version en cache qui est utilisée pendant toute la durée de vie du système (sauf si le système est redémarré). Ainsi pour tester plusieurs versions d'une compétence ou pour la mettre à jour, le développeur est obligé de changer le nom de la compétence (le faire suivre d'un numéro, par exemple) ; ce qui est fastidieux lorsqu'il est nécessaire d'effectuer plusieurs tests pendant le développement de la compétence. Nous envisageons de remédier à cela en fournissant un mécanisme de réinitialisation de cette cache de compétences.

Actuellement, une compétence est conçue pour répondre à un besoin spécifique et bien délimité. Il n'est pas aisé de concevoir une compétence générique pour l'ajouter à un agent et l'affiner dynamiquement en fonction des besoins. A titre d'exemple, il serait intéressant de concevoir des

[73] Cette cache est gérée par la plateforme Magique se reposant elle-même sur la gestion des chargements des classes Java (ClassLoader) effectuée par la JVM (Java Virtual Machine).

compétences génériques intégrant les méthodes de personnalisation proposées au chapitre 4. Ces compétences seraient ensuite ajoutées aux agents puis paramétrées suivant le domaine d'application (par exemple, les critères d'apprentissage seraient associés aux modes de transport dans le cas d'une application de recherche d'itinéraire ou seraient associés à des mots clés dans le cas d'une application de recherche de documents numériques). L'idée consiste à associer à une compétence une interface graphique. Cette interface serait générée au niveau de l'agent d'administration permettant ainsi le paramétrage d'une compétence d'un agent pour l'adapter au domaine d'application lors du développement d'un système d'information personnalisé.

A moyen terme, il est souhaitable d'intégrer PerSyst à un atelier de modélisation et à un environnement de développement. Les agents pourraient ainsi être créés (en associant leurs compétences et en les déployant) automatiquement à partir de la modélisation dont le code informatique des compétences serait affiné au niveau de l'environnement de développement.

Des recherches pour la production de nouvelles méthodes plus efficaces et plus performantes pour la personnalisation de l'interaction homme-machine sont aussi envisagées. A moyen terme, il s'agira de combiner nos méthodes de filtrage basées sur les réseaux bayésiens avec des méthodes à base de modèles comme la méthode des *k plus proches voisins* pour augmenter la performance d'exécution (en terme de temps de calcul).

Il existe d'autres problèmes, comme la sécurité et la confidentialité, de PerSyst qui n'ont pas été abordés dans le cadre de cette thèse et dont il serait très intéressant d'apporter des solutions adaptées.

Perspectives liées à l'application

Une des difficultés liées à nos travaux résidait sur l'obtention des données relatives au transport. En effet, il existe des contraintes fortes d'ordres économique (contextes de concurrence, informations commerciales…) et juridique (propriété de l'information, responsabilité d'utilisation…), dont l'étude ne rentre pas dans le cadre de cette thèse, pour l'obtention et l'exploitation des données transport. C'est pourquoi nous avons opté pour la construction de notre propre base de données transport (en se basant, tout de même, sur des données réelles récupérées sur des sites commerciaux) pour pouvoir tester nos applications. Ainsi les applications développées n'utilisent pas de données suffisamment complètes, provenant des exploitants ou des autorités organisatrices (AO), pour envisager de les mettre à disposition du grand public afin de les évaluer au travers d'usages réels. Ces contraintes devraient s'alléger avec les projets en cours d'étude dans la région Nord-Pas de Calais, en collaboration avec plusieurs exploitants (SNCF, Transpole et Semurval) visant, entre autres, à constituer un *gisement de données* regroupant les informations transport régionales. Il serait donc très profitable que le système d'information « mon service transport » utilise les données fournies par ce gisement de données.

L'interconnexion de « mon service transport » avec ce gisement de données ne devrait pas poser de difficulté particulière. Il suffirait de concevoir les comportements adéquats des agents de recherche d'information pour qu'ils puissent se connecter à la base de données transport pour effectuer des recherches d'itinéraires ou des recherches sur les perturbations signalées par les exploitants.

De ce fait, des évaluations faisant intervenir des utilisateurs en situation réelle de déplacement pourraient être effectuées. Ces évaluations présupposent des études et des recherches approfondies pour disposer de critères rigoureux liés au caractéristique *personnalisé* du système d'information. Dans ce cadre, une thèse vient d'être amorcée pour fournir des méthodes adaptées aux évaluations (techniques et ergonomiques) des systèmes d'information personnalisés. En particulier ; ces méthodes devraient servir de support pour l'évaluation global de « mon service transport ».

Nous envisageons aussi de concevoir de nouveaux services innovants pour les usagers des transports terrestres de personnes et d'améliorer les services existants en les rendant plus personnalisés, plus contextualisés. Par exemple, il serait judicieux d'ajouter d'autres fonctionnalités au service d'agenda transport pour non seulement proposer des itinéraires mais aussi recommander à l'utilisateur des hôtels et/ou des restaurants par rapport à ses préférences liées au contexte de son déplacement.

Il serait aussi intéressant d'interconnecter le service d'agenda transport fourni par « mon service transport » aux emplois du temps des étudiants[74] et aux agendas personnels et professionnels pour que l'utilisateur retrouve automatiquement ses planifications dans son agenda transport ; il pourrait ainsi bénéficier des recommandations d'informations transport liées à ses planifications.

D'autres services qui nécessiteraient des études ou des recherches beaucoup plus approfondies pourraient porter sur l'intégration des contextes, comme les conditions climatiques, l'état physique ou mental de l'utilisateur, dans la personnalisation de l'information transport. En effet, le choix d'un itinéraire par un utilisateur pourrait être motivé par rapport aux conditions climatiques, par exemple. Un utilisateur pourrait préférer le vélo lorsqu'il fait beau et choisir le bus lorsqu'il pleut ou lorsqu'il fait très froid. Un autre utilisateur pourrait aimer le métro mais favoriserait un taxi s'il est malade… La méthode de personnalisation que nous avons proposée semble pouvoir convenir puisque qu'elle modélise les préférences utilisateur selon des contextes. Toutefois, il n'est pas aisé de définir des contextes appropriés pouvant influencer la décision d'un utilisateur par rapport à un ensemble de choix. Et parfois, il est très difficile de fournir une méthode efficace pour capturer un contexte sans alourdir l'interaction homme-machine. Comment savoir que l'utilisateur est malade sans le lui demander ? Ce sont des problématiques intéressantes sur lesquelles il conviendrait de se pencher.

Nous commencerons par considérer les préférences des utilisateurs par rapport aux contextes relatifs aux conditions climatiques pour la recommandation des itinéraires. La température, l'humidité et la force du vent pourraient s'obtenir au travers de services web. A partir des informations (ville de départ, ville d'arrivée, date de départ, etc.) fournies par l'utilisateur lors de sa recherche d'itinéraire, il serait alors possible de déduire les conditions climatiques du déplacement. En appliquant notre méthode de personnalisation (voir chapitre 4, §4.3), il serait donc envisageable de recommander des itinéraires par rapport aux préférences des utilisateurs associées aux conditions climatiques. Nous pourrions continuer, ensuite, ces travaux pour la prise en compte d'autres contextes (luminosité, état physique, état psychique, etc.) aussi bien pour le filtrage du contenu que pour l'adaptation du contenant.

[74] Ce service est d'ores et déjà en cours d'étude dans le cadre du projet Mouver.Perso.

Bibliographie générale

[Aas 97] Aas K. *A Survey on Personalised Information Filtering Systems for the World Wide Web* ; Research Report n° 922, Oslo, Norway, Norwegian Computing Center, December 1997.

[Abed 01] Abed M. *Méthodes et Modèles formels et semi-formels de conception et évaluation des systèmes homme-machine*. Mémoire de HDR, université de Valenciennes et du Hainaut-Cambrésis, 2001.

[Abed et Angue 90] Abed M. and Angue J.C. Using the measure of eye mouvements to modelise an operator's activity. *Ninth European Annual conference on "Human decision making and manual control",* Varesse, Italy, September 1990.

[Abou-Harb et Rivard 03] Abou-Harb G. et Rivard F. *L'EAI au service de l'entreprise évolutive*. Paris : Maxima, 2003.

[Abowd et al., 99] Abowd G.D., Dey A.K., Brown P.J., Smith M. and Steggles P. Towards a Better Understanding of Context and Context-Awareness. *Lecture Notes In Computer Science*, Vol. 1707. *Proceedings of th 1st international symposium on Handheld and Ubiquitous Computing*, Karlsruhe, Germany, pp. 304-307, September 1999.

[Adam 00] Adam E. *Modèle d'organisation multi-agent pour l'aide au travail coopératif dans les processus d'entreprise : application aux systèmes administratifs complexes*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, septembre 2000.

[Adam et Kolski 99] Adam E. et Kolski C. Etude comparative de méthodes de génie logiciel utiles au développement de systèmes interactifs dans les processus administratifs complexes. *Génie Logiciel*, 49, pp. 40-54, 1999.

[Aggarwal et al., 99] Aggarwal C.C., Wolf J.L., Wu K-L. and Yu P.S. Horting Hatches an Egg : A New Graph-Theoretic Approach to Collaborative Filtering. In *Proceedings of KDD'99*, *5th International Conference on Knowledge Discovery and Data Mining*, pp. 201-212, San Diego, USA, August 1999.

[Alhir 05] Alhir S.S. The Agile Unified Process (AUP). *UML & Design World 2005 Conference,* Austin. TX, USA. June 13-16, 2005.

[Ali et Abrams 01] Ali M.F. and Abrams M. Simplifying construction of multi-platform User Interfaces using UIML. Presented at *UIML Europe 2001 Conference*, March 2001.

[Allen 97] Allen R. B. Mental Models and User Models. In Helander, M., Landauer, T.K. and Prabhu, P. (Eds.), *Handbook of Human-Computer Interaction*. Elsevier Science, pp. 49-63, 1997.

[Ambler et al., 05] Ambler S.W., Nalbone J. and Vizdos M. *The Enterprise Unified Process : Extending the Rational Unified Process*. Prentice Hall PTR, 2005.

[Ambrosini et al., 97] Ambrosini L., Cirillo V. and Micarelli A. A Hybrid Architecture for User-Adapted Information Filtering on the World Wide Web. In Jameson, A., Paris, C. and Tasso, C. (Eds.), *User Modeling: Proceedings of the Sixth International Conference*, UM97, New York : Springer Wien, pp. 59-61, 1997.

*[Anli 02] Anli A. *Vers un modèle utilisateur pour la personnalisation de l'information dans les transports terrestres de personnes*. Rapport de DEA, Université de Valenciennes et du Hainaut-Cambrésis. Juillet 2002.

*[Anli et Abed 05] Anli A. et Abed M. PerSyst : Un Système de Personnalisation générique, Application à l'information voyageur. *Workshop Méthodologies et Heuristiques pour l'Optimisation des Systèmes Industriels (MHOSI'05).* Hammamet, Tunisie, avril 2005.

*[Anli et Abed 06] Anli A. et Abed M. PerSyst : un Système de Personnalisation de l'information transport multimodale. *CIFA'2006, Conférence Internationale Francophone d'Automatique (30 mai-1 juin)*, Bordeaux, France, mai 2006.

*[Anli et al., 04] Anli A., Petit-Rozé C. and Grislin-Le Strugeon E. User-based Decision Support System. *In International Conference on Advances in Intelligent Systems - Theory and Applications in cooperation with IEEE Computer Society*, AISTA'2004, Luxembourg, novembre 2004.

*[Anli et al., 04a] Anli A., Petit-Rozé C. et Grislin-Le Strugeon E. Plate-forme d'intégration de services personnalisés à base d'agents logiciels. *Génie Logiciel*, 71, pp. 34-39, 2004.

*[Anli et al., 05] Anli A., Grislin-Le Strugeon E. et Abed M. Une plate-forme de personnalisation basée sur une architecture multi-agents. *Revue des Nouvelles Technologies de l'Information* (RNTI-E), 5, pp. 95-100, 2005.

*[Anli et al., 05a] Anli A., Grislin-Le Strugeon E. et Abed M. A Generic Personalization Tool based on a Multi-agent Architecture. In C. Stephanidis (Ed.), *Proceedings HCI International (Las Vegas, Nevada, July 22-27, 2005), Volume 7 - Universal Access in HCI: Exploring New Interaction Environments*, Lawrence Erlbaum Associates, Mahwah, New Jersey, pp. 1-8, juillet 2005.

*[Anli et al., 05b] Anli A., Kolski C. et Abed M. Principes et architecture pour la personnalisation d'information en interaction homme-machine : Application à l'information transport. In *Proceedings of IHM 2005, International Conference Proceedings Series*, ACM Press, Toulouse, pp. 123-130, septembre 2005.

[Asnicar et Tasso 97] Asnicar F.A. and Tasso C. IfWeb: a Prototype of User Model-Based Intelligent Agent for Document Filtering and Navigation in the World Wide Web. In *Proceedings of the workshop « Adaptive Systems and User Modeling on the World Wide Web »,* Chia Laguna, Sardina, pp. 3-12, 2-5 June 1997.

[Balabanovic et Sholham 97] Balabanovic M. and Sholham Y. Fab : Content-based, collaborative recommendation. *Communications of the ACM*, vol. 40, pp. 66-72, March 1997.

[Banks et al., 97] Banks S.B., Harrington R.A., Santos E., Brown M. Jr and Brown S.M. Usability testing of an intelligent interface agent. In *Interfaces 97*, Montpellier, France, May 1997.

[Bastien et Scapin 01] Bastien J. et Scapin D. Evaluation des systèmes d'information et critères ergonomiques. In *C. Kolski (Ed.), Environnements évolués et évaluation de l'IHM. Interaction Homme-Machine pour les SI*, pp. 53-79, Paris : Editions Hermès, 2001.

[Basu et al., 98] Basu C., Hirsh H. and Cohen W. : Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In C. Rich and J. Mostow (Eds.), *Proceedings of the 15th National Conference on Artificial Intelligence*, pp. 714-720. Madison, USA, AAAI Press/MIT Press, 1998.

[Bayardo et al., 97] Bayardo Jr. R. J., Bohrer W., Brice R., Cichocki A., Fowler J., Helal A., Kashyap V., Ksiezyk T., Martin G., Nodine M., Rashid M., Rusinkiewicz M., Shea R., Unnikrishnan C., Unruh A. and Woelk D. InfoSleuth: agent-based semantic integration of information in open and dynamic environments. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of data,* pp.195-206, May 11-15, Tucson, Arizona, United States, 1997.

[Bazsalicza et Naïm 01] Bazsalicza M. et Naïm P. *Data mining pour le web : Profiling – Filtrage collaboratif – Personnalisation client*. Paris : Editions Eyrolles, 2001.

[Beale et Wood 94] Beale R. and Wood A. Agent-Based Interaction. In *People and Computers IX: Proceedings of the HCI'94*, pp. 239-245, Glasgow, UK,. Cambridge University Press, August 1994.

[Beck and Cunningham 89] Beck K. and Cunningham W. A Laboratory For Teaching Object-Oriented Thinking. In *the OPSLA'89 Conference Proceedings* October 1-6, 1989, New Orleans, Louisiana. Special issue of SIGPLAN Notices 24(10), 1989. Disponible à http://c2.com/doc/oopsla89/paper.html.

[Bellifemine et al., 00] Bellifemine F., Poggi A. and Rimassa G. Developing multi-agent systems with JADE. In se*venth International Workshop on Agent Theories, Architectures, and Languages (ATAL-2000)*, Boston, MA, pp. 85-99, 2000.

[Ben Lazreg 04] Ben Lazreg I. *Conception de système d'information au service des usagers dans les transports terrestres de personnes.* Rapport de DEA, Université de Valenciennes et du Hainaut-Cambrésis, juillet 2004.

[Berrut et Denos 03] Berrut C. et Denos N.. Filtrage collaboratif. In *Assistance intelligente à la recherche d'informations,* E. Gaussier, M.H. Stéfanini (dir.), Hermès-Lavoisier, pp. 241-269, 2003.

[Boehm 81] Boehm B.W. *Software Engineering Economics*. Englewood Cliffs : Prentice Hall,1981.

[Boehm et al., 84] Boehm B.W., Gray T.E. and Seewaldt T. Prototyping versus specifying : a multiproject experiment. *IEEE transactions on Software Engineering*, 10(3), pp. 290-302, 1984.

[Boidin 05] Boidin E. *Apprentissage automatique pour la personnalisation : Application aux transports terrestres de personnes*. Rapport de DEA, Université de Valenciennes et du Hainaut-Cambrésis, juillet 2005.

[Bond et Gasser 88] Bond A. and Gasser L. *Readings in distributed artificial intelligence*. Morgan Kaufman, San Mateo, CA, 1988.

[Bonnet et Macary 97] Bonnet C. et Macary J. F. *Technologie PUSH*. Paris : Eyrolles, 1997.

[Booch et al., 00] Booch G., Rumbaugh J. and Jacobson I. *Le guide de l'utilisateur UML*. Paris : Eyrolles, 2000.

*[Bousbia et al., 05] Bousbia S., Anli A., Trentesaux D. and Grislin-Le Strugeon E. Agile Scheduling of flexible manufacturing systems of production. In P. Borne, M. Benrejeb, N. Dangoumeau, L. Lorimier (Eds.), *17th IMACS World Congress "Scientific Computation, Applied Mathematics and Simulation"* (July 11-15, Paris), pp. 316-323, juillet 2005.

[Bouzeghoub et Kostadinov 04] Bouzeghoub M et Kostadinov D. *Une approche multidimensionnelle pour la personnalisation de l'information*. Rapport PRiSM, Versailles, France, 2004.

[Breese et al., 98] Breese J.S., Heckerman D. and Kadie C. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *Proceedings of the Fourtheenth Conference on Uncertainty in Artificial Intelligence*. Madison, Morgan Kaufmann Publisher. July, 1998.

[Browne et al., 90] Browne D., Totterdell P. and Norman M (Eds.). *Adaptive User Interfaces*. Academic Press, Computer And People Series, 1990.

[Burrafato and Cossentino 02] Burrafato P. and Cossentino M. Designing a multi-agent solution for a bookstore with the PASSI methodology. *Proceedings of the Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002 at CaiSE'02).* Toronto (Ontario, Canada), May 27-28, 2002.

[Calvary et al., 03] Calvary G., Coutaz J., Thevenin D., Limbourg Q., Bouillon L. and Vanderdonckt, J. A unifying reference framework for multi-target user interfaces. *Interacting with Computers*, 15(3), pp. 289-308, 2003.

[Cardon 00] Cardon A. *Conscience artificielle et systèmes adaptatifs*. Paris : Eyrolles, 2000.

[Carlson 01] Carlson D. *Modélisation d'applications XML avec UML*. Paris : Eyrolles, 2001.

[Castelfranchi 95] Castelfranchi C. Guarantees for autonomy in cognitive agent architecture. In M. Wooldridge and N.R. Jennings (Eds.), *Intelligent Agents-Agent Theories, Architectures, and Languages*, volume 890 of Lecture Notes in Computer Science (subseries LNAI), pages 56-70, Springer-Verlag, 1995.

[Ceri et al., 00] Ceri S., Fraternali P. and Bongio A. Web Modeling Language (WebML) : a modeling language for designing web sites. In *Ninth International World Wide Web Conference WWW9*, Amsterdam, 15-19 May 2000.

[Chandrasekaran et al., 99] Chandrasekaran B., Josephson J. R. and Benjamins V. R. What Are Ontologies, and Why Do We Need Them? *IEEE Intelligent Systems*, Vol. 14(1), pp. 20-26, 1999.

[Chin 91] Chin D. Intelligent interfaces as agents. In J. W. Sullivan and S. W. Tyler (Eds.), *Intelligent User Interfaces*, Addison-Wesley, New York, pp. 177-206, 1991.

[ChoiceStream 04] *ChoiceStream. Review of Personalization Technologies : Collaborative Filtering vs. ChoiceStream's Attributized Bayesian Choice Modeling*. Technology Brief. Disponible à l'adresse : www.choicestream.com/pdf/ChoiceStream_TechBrief.pdf

[Cinquin et al., 02] Cinquin L., Lalande P. A et Moreau N. *Le projet ECRM : Relation client et Internet*. Editions Eyrolle, Paris, 2002.

[Cohen et Levesque 90] Cohen P.R. and Levesque H.J. Intention is choice with commitment. *Artificial Intelligence*, 42, pp. 213-261, 1990.

[Conallen 00] Conallen J. *Concevoir des applications web avec UML*. Paris : Eyrolles, 2000.

[Conssentino et al., 00] Conssentino M., Chella A. and Faso L. U. Designing agent-based systems with UML. In *International symposium on Robotics and Automation*, ISRA'2000, Monterrey, Mexico, November 2000.

[Cosley et al., 02] Cosley D., Lawrence S. abd Pennock D. M. REFEREE : An open framework for practical testing of recommender systems using ResearchIndex. In *Proceeding of the 28th VLDB Conference*. Hong Kong, China, August 2002.

[Coulange 96] Coulange B. *Réutilisation du logiciel*. Paris : MASSON, 1996.

[Coyle et Cunninghan 02] Coyle L. and Cunninghan P. A Case-Based Personal Travel Assistant for Elaborating User Requirements and Assessing Offers. *Proceedings of the 6th European Conference, ECCBR 2002*, S. Craw, A. Preece (Eds.). LNAI Vol. 2416, pp. 505-518, Springer-Verlag, 2002.

[De Troyer et Leune 98] De Troyer O.M.F. and Leune C.J. WSDM : A User centered Design Method for Web Sites. *Proceedings of the 7th International Conference on World Wide Web*. Brisbane, Australia, pp. 85-94, April 1998.

[DeLoach et al., 01] DeLoach S. A., Wood M. and Sparkman C. Multiagent system engineering. *International Journal of Software Engineering and Knowledge Engineering*, 11(3), 231-258, 2001.

[Deschaine et al., 00] Deschaine L., Brice R. and Nodine M. Use of InfoSleuth to Coordinate Information Acquisition, Tracking and Analysis in Complex Applications. In *Proceedings of Advanced Simulation Technologies Conference*, Washington, D.C., USA, April 2000.

[Dickinson et al., 03] Dickinson I., Reynolds D., Banks D., Cayzer S. and Vora P. User Profiling with privacy : A framework for Adaptive Information Agents, M. Klush et al. (Eds.) : *Intelligent Information Agents*, LNAI 2586, pp. 123-151, Berlin : Springer-verlag, 2003.

[Dumas et Charbonnel 90] Dumas P. and Charbonnel G. *La méthode OSSAD, pour maîtriser les technologies de l'information. Tome 1 : principes*. Paris : Les éditions d'organisation, 1990.

[Erceau et Ferber 91] Erceau J. and Ferber J. L'Intelligence Artificielle Distribuée. *La Recherche*, n° 233, pp. 750, Juin 1991.

[ESA 91] ESA. *Software Engineering Standards*. European Space Agency, Rapport Issue 2, 1991.

[Ferber 95] Ferber J. *Les systèmes multi-agents : Vers une intelligence collective*. Paris : InterEditions, 1995.

[Finin et al., 97] Finin T., Labrou Y. and Mayfield J. KQML as an Agent Communication Language. In J. Bradshaw (Eds.), *Software Agents*, pp. 291-316. Cambridge, USA, MIT Press, 1997.

[FIPA 02] FIPA. *FIPA ACL Message Structure Specification*. Rapport technique N°SC00061G, Geneva, Switzerland, December 2002.

[Fisher 96] Fisher D. Iterative Optimization and Simplification of Hierarchical Clusterings. *Journal of Artificial Intelligence Research*, n° 4, pp. 147-179, 1996.

[Florins et Vanderdonckt 04] Florins M. and Vanderdonckt J. Graceful Degradation of User Interfaces as a Design Method for Multiplatform Systems. In *Proceedings of the 2004 International Conference on Intelligent User Interfaces IUI 2004*, Funchal, Madeira Island, Portugal, January 13-16, pp. 140-147, 2004.

[Fuggetta et al., 98] Fuggetta A., Picco G.P., and Vigna G. Understanding code mobility. *IEEE Transactions on software engineering*, 24(5), 342-361, May 1998.

[Gabay 01] Gabay J. *Merise et UML pour la modélisation des systèmes d'information*. Dunod : Paris, 2001.

[Gamma et al., 99] Gamma E., Helm R. et Johnson R. *Design patterns catalogue de modèles de conception réutilisables*. Paris : Vuibert, 1999.

[Gilles 97] Gilles M. *Les attentes des usagers*. Rapport du FIER n° 22, UTP, 1997.

[Giorgini et al., 01] Giorgini P., Perini A. Mylopoulos J., Giunchiglia F. and Bresciani P. Agent-Oriented Software Development : A case study. In *proceeding of the 13th International conference on Software Engineering & Knowledge Engineering* (SEKE 01), Buenos-Aires, Argentina, June 13-15, 2001.

[Giorgini et al., 02] Giorgini P., Kolp M. and Mylopoulos J. Multi-Agent and Software Architectures : A comparative case study. *Proceeding of the international conference on Autonomous Agents and MAS* (AAMAS'02), Bologna, Italy, July 2002.

[Goldberg 94] Goldberg D. *Algorithmes génétiques : exploration, optimisation et apprentissage automatique*. Paris : Addison Wesley, 1994.

[Goldberg et al., 01] Goldberg K., Roeder T., Gupta D. and Perkins C.. Eigentaste : A Constant Time Collaborative Filtering Algorithm. *Information Retrieval Journal*, vol. 4(2), pp. 133-151, 2001.

[Goldberg et al., 92] Goldberg D., Nichols D., Oki B. and Terry D. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, vol. 35, pp. 61-70, 1992.

[Gray et al., 00] Gray R.S., Cybenko G., Kotz D. and Rus D. *Mobile agents: Motivations and State of the Art*. Technical Report TR2000-365, Dept. of Computer Science, Dartmouth College, 2000.

[Grislin et Kolski 96] Grislin M. et Kolski C. Evaluation des interfaces homme-machine lors du développement de système interactif. *Technique et Science Informatiques (TSI)*, 2, pp. 265-296, 1996.

[Grislin-Le Strugeon et al., 01] Grislin-Le Strugeon E., Adam E. et Kolski C. Agents intelligents en interaction Homme-Machine dans les Systèmes d'information. In C. Kolski (Eds.), *Environnements évolué et évaluation de l'I.H.M.*, *Interaction Homme-Machine pour les S.I. 2*, Paris : Editions Hermes, pp. 209-248, 2001.

*[Grislin-Le Strugeon et al., 06] Grislin-Le Strugeon E., Anli A. and Adam E. A methodology to bring MAS to information systems. In T. Latour, M. Petit (Eds.), *CAISE'06 The 18th International Conference on Advances Information Systems Engineering (Luxembourg, June 5-9, 2006),*

Proceedings of Workshops and Doctoral Consortium, Presses Universitaires de Namur, pp. 64-75, juin 2006.

*[Grislin-Le Strugeon et al., 06a] Grislin-Le Strugeon E., Petit-Rozé C., Anli A., Abed M. et Kolski C. *Mouver.Perso : Mobilité et mUltimodalité Voyageurs Etudiants en Région Nord Pas de Calais – Système d'information multimodale personnalisée*. Rapport intermédiaire de projet PREDIT, LAMIH, Valenciennes, janvier 2006.

[Gutknecht et al., 00] Gutknecht O., Ferber J. Michel F. MADKit : une expérience d'architecture de plateforme multi-agent générique. In *Proceedings of the JFIADSMA'00*, Hermès, pp. 223-236, 2000.

[Gutknecht et Ferber 98] Gutknecht O. et Ferber J. Un modèle d'analyse, de construction et d'exécution de systèmes multi-agents. In *Jean-Pierre Barthès (eds.), Actes des Journées Francophones en Intelligence Artificielle Distribuée et Système Multi-Agents 1998*, Hermès, novembre 1998.

[Habieb-Mammar et al., 02] Habieb-Mammar H., Tarpin-Bernard et Prevot P. Modélisation XML des interfaces adaptatives intégrant le profil cognitif de l'utilisateur. In *Documents Virtuels Personnalisables* (DVP 2002). Brest, ENST Bretagne, 10 & 11 juillet 2002.

[Hagimont et Layaïda 02] Hagimont D. et Layaïda N. Adaptation d'une application multimédia par un code mobile. *Technique et Science Informatiques* (TSI), numéro spécial Agents et code mobile, Vol. 21, N° 6, pp. 877-897, 2002.

[Hammond et al., 84] Hammond N., Hinton G., Barnard P., Maclean A., Long J. and Whiteield A. Evaluating the interface of a document processor: a comparison of expert judgement and user observation. *Proceeding of the First IFIP Conference on Human-Computer Interaction: Interact'84*, London, pp. 725-729, 1984.

[Harold et Means 01] Harold E.R. and Means W.S. *XML in a nutshell*. Paris : O'Reilly, 2001.

[Hayes et Cunningham 01] Hayes C. and Cunningham P. Smart Radio – community based music radio. *Knowledge-Based Systems*, n° 14, pp. 197-201, 2001.

[Hayes et Cunningham 04] Hayes C. and Cunningham P. Context boosting collaborative recommendations. *Knowledge-Based Systems*, n° 17, pp. 131-138, 2004.

[Herlocker et al., 99] Herlocker J. L, Konstant J. A., Brochers A. and Riedl J. A algorithmic framework for perfomrming collaborative filtering. In *proc. 1999 Conf. Research and Developpement in Information retrieval*, pp 230-237, Berkeley, CA, Août 1999.

[Hirsh et al., 00] Hirsh H., Basu C. and Davison B. Learning to personalize. *Communications of the ACM*, 43(8):102-106, 2000.

[Hodgson 91] Hodgson R. The X-model: a process model for object-oriented software development. *Acte du congrès Le genie logiciel et ses applications*, Toulouse, 1991.

[Höök 00] Höök K. Steps to take before intelligent user interfaces become real. *Interacting with computers*, pp. 409-426, 12, 2000.

[Howden et al., 01] Howden N., Rönnquist R., Hodgson A. and Lucas A. JACK Intelligent Agents – Summary of an Agent Infrastructure. *5th International Conference on Autonomous Agents*, Montreal, Canada, May 2001.

[Hoyer et Czogolla 02] Hoyer R. and Czogolla O. Approach to personalised information services to public transport. *In 9th world congress on Intelligent transportation systems*, Chicago Illinois, october 14-18, 2002.

[IGL 89] I.G.L. Technology. *SADT, un langage pour communiquer*. Paris : Eyrolles, 1989.

[Jacobson et al., 00] Jacobson I., Booch G. and Rumbaugh J. *Le processus unifié de développement logiciel*. Paris : Eyrolles, 2000.

[Jamont 05] Jamont J.P. *DIAMOND : une approche pour la conception de système multi-agents embarqués*. Thèse de doctorat, Institut National Polytechnique de Grenoble, septembre 2005.

[Jennings et al., 98] Jennings N. R., Sycara,K. and Wooldridge M. A Roadmap of Agent Research and Development. *Autonomous Agents and Multi-Agent Systems*, Vol. 1, pp. 7-38, 1998.

[JPF Consultant 96] JPF Consultant. *Etude de faisabilité sur la mise en place d'un système d'information sur l'offre de transport collectif de voyageur* - Phase 4 : Préconisation ; Rapport technique, octobre 1996.

[Kadima et Monfort 03] Kadima H. et Monfort V. *Les web services : Techniques et outils XML, WSDL, SOAP, UDDI, Rosetta, UML*. Paris : Dunod, 2003.

[Kay 01] Kay J. User Modeling for Adaptation. In Stephanidis, C. (Ed.) *User Interface For All*., London : LEA publishers, pp. 271-294, 2001.

[Keeble et Macredie 00] Keeble R.J. and Macredie R.D. Assistant agents for the world wide web intelligent interface design challenges. *Interacting with computers,* Vol. 12 No. 4, pp. 357-381, 2000.

[Kimball et Merz 00] Kimball R. et Merz R.. *Le data web house : Analyser les comportements clients*. Paris : Editions Eyrolles, 2000.

[Kobsa et Pohl 95] Kobsa A. and Pohl W. The User Modeling Shell System BGP-MS. *User Modeling and User-Adapted Interaction,* Vol. 4, pp. 59-106, 1995.

[Koch et al., 01] Koch N., Kraus A. and Hennicker R. The Authoring Process of the UML-based Web Engineering Approach. In *1st International Workshop on Web-Oriented Software Technology IWWOST'2001*, Valencia, Spain, 18-20 June, 2001.

[Kolski 01] Kolski C. (dir) *Analyse et conception de l'IHM*. Paris : Hermes, 2001.

*[Kolski et al., 04] Kolski C., Petit-Rozé C., Anli A., Abed M., Grislin-Le Strugeon E., Ezzedine H. et Trabelsi A. La plasticité vue sous l'angle de la personnalisation ou selon les besoins vis-à-vis de l'information transport. *Journées thématiques de l'AS Plasticité du RTP 16 IHM*, Namur, Belgique, août, 2004.

[Kolski et Le Strugeon 98] Kolski C. and Le Strugeon E. A review of intelligent human-machine interfaces in the light of the ARCH model. *International Journal of Human-Computer Interaction*, 10(3), pp. 193-231, 1998.

[Korfhage 97] Korfhage R.R. *Information storage and retrieval*. Wiley Computer Publishing. ISBN 0-471-14-338-3, 1997.

[Kruchten 00] Kruchten P. *Introduction au Rational Unified Process*. Paris : Eyrolles, 2000.

[Labidi et Lejouad 93] Labidi S. et Lejouad W. *De l'intelligence artificielle distribuée aux systèmes multi-agents*. Rapport de recherche 2004, INRIA, Sophia-Antipolis, France, août 1993.

[Lam et Xie 02] Lam S. H. and Xie F. Provision of Personalised Transit Travel Information System and Architecture. *In 9th world congress on Intelligent transportation systems*, Chicago Illinois, october 14-18, 2002.

[Lashkari 97] Lashkari Y., Metral M., and Maes P. Collaborative interface agents. In *M.N. Huhns and M.P. Singh (eds), Readings in Agents*. Morgan Kaufmann, 1997.

[Laurel 97] Laurel B. Interface agents: metaphors with character. In J.M Bradshaw (Ed.), *Software agents,* pp.67-77, Mento Park, CA : AAAI Press, 1997.

[Lecomte et Patesson 00] Lecomte N. et Patesson R. Le panel des voyageurs : une étude des activités et des besoins d'information des utilisateurs des transports publics. In *Actes de la conférence ERGO-IHM* (ERGO-IHM'00 - Biarritz, France, 3-6 octobre), D. Scapin and E. Vergison (Eds.), pp. 129-135, 2000.

[Lepreux 05] Lepreux S. *Approche de Développement centré décideur et à l'aide de patrons de Systèmes Interactifs d'Aide à la Décision, Application à l'investissement dans le domaine ferroviaire*. Mémoire de Doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, juin 2005.

[Lepreux et al., 03] Lepreux S., Kolski C. and Abed M. Design method for component-based DSS. *International Workshop on Component-Based Business Information Systems Engineering* (CBBISE), Geneve, Suisse, 2003.

[Lieberman 95] Lieberman H. Letizia : An Agent That Assists Web browsing. In *International Joint Conference on Artificial Intelligence (IJCAI'95 – Montreal, Canada, August 20-25).* San Francisco, USA, Morgan Kaufman Publishers, 1995.

[Lieberman 97] Lieberman H. Autonomous interface agents. In *Proceedings of CHI'97 (Human factors in computing systems)*, pp. 67-74, Atlanta, GA USA, March 22-27, ACM Press, 1997.

[Lieberman et al., 01] Lieberman H., Fry C. and Weitzman L. Exploring the Web with Reconnaissance Agents. In *ACM Conference on Human-Computer Interface*. ACM Press, pp. 69-75, August 2001.

[Linden et al., 97] Linden G., Hanks S. and Lesh N. Interactive Assessment of User Preference Models : The Automated Travel Assistant. In Jameson, A., Paris, C. and Tasso, C. (eds), *User Modeling: Proceedings of the Sixth International Conference*, *UM97*. New York : Springer Wien, pp. 67-78, 1997.

[Lumineau 03] Lumineau N. *Un tour d'horizon du filtrage collaboratif*. Travail réalisé dans le cadre de l'AS Personnalisation de l'Information, janvier – décembre 2003, disponible à l'adresse : www.prism.uvsq.fr/recherche/themes/sial/cnrs/Fichiers/Rapport\TourDHorizonDuFiltrageCollaboratif-LIP6.pdf.

[Ma et al., 04] Ma J., Piechowiak S. and Mandiau R. A Multi-Factor Concept on Guiding Constraint Relaxation in Distributed Constraint Satisfaction. *WSEAS Transactions on Computers*, 3, pp. 442-448, 2004.

[Maclean et Dailey 02] Maclean S. D. And Dailey D. J. The use of wireless Internet Service to access Real-Time Transit Information. *In 9th world congress on Intelligent transportation systems*, Chicago Illinois, october 14-18, 2002.

[Maes 94] Maes P. Agents that reduce work and information overload. *Communications of the ACM* 37(7), pp. 30-40, 1994.

[Mahfoudi et al., 95] Mahfoudi A., Abed M. and Angué J-C. An Object Oriented Methodology for Man-Machine systems analysis and design. In Anzai Y., Ogawa K., Mori H. (Eds.), *Symbiosis of Human and Artefact, HCI International'95: 6th Internationa*l, Tokyo, Japan, Elsevier, Amsterdam, pp. 965-970, janvier 1995.

[Manber et al., 00] Manber U., Patel A. and Robison J. Exeprience with personalization on Yahoo! *Communictions of the ACM*, 43(8) : 35-39, August 2000.

[Mandiau et al., 02] Mandiau R., Grislin-Le Strugeon E. et Péninou A. (Eds.). *Organisation et applications des SMA*. Paris : Hermès, 2002.

[Mandiau et Grislin-Le Strugeon 01] Mandiau R. et Grislin-Le Strugeon E. Systèmes Multi-Agents. *Techniques de l'ingénieur*, 2001.

[Manouvrier 01] Manouvrier B. *EAI : intégration des applications d'entreprise*. Paris : Hermès science publications, 2001.

[Marchand et al., 99] Marchand R., Agnoux H. et Chiaramonti C. *Application EDI sur l'Internet : commerce électronique B to B*. Paris : Eyrolles, 1999.

[Mathieu et Routier 01] Mathieu P. et Routier J. C. Une contribution du multi-agent aux applications de travail coopératif. *TSI Hermès Science Publication*, Numéro Spécial : Télé-applications, Vol. 13, n°2-3, pp. 207-232, 2001.

[Mathieu et Verrons 03] Mathieu P. et Verrons M.H. A generic negotiation model for MAS using XML. *International Workshop series Agents for Business Automation : Research and Development* (ABA03) (IEEE), pp. 4262-4267, 2003.

[McDermid et Ripkin 84] McDermid J. and Ripkin K. *Life cycle support in the ADA environment*. Cambridge University Press, 1984.

[Melville et al., 02] Melville P., Mooney R.J. and Nagarajan R. Content-Boosted Collaborative Filtering for Improved Recommendations. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI-2002)*. Edmonton, Canada, July, AAAI Press, pp. 187-192, 2002.

[Millot et Roussillon 91] Millot P. and Roussillon E. Man-Machine cooperation in Telerobotics : Problematics and Methodologies. *Second Symposium on Robotics*, Gif-sur-Yvette, 1991.

[Moghrabi et Eid 98] Moghrabi C. and Eid M.S. Modeling users through an expert system and a neural network. *Computers & Industrial Engineering*, Vol. 35, No 3-4, pp. 583-586, 1998.

[Monfort et Goudeau 04] Monfort V. et Goudeau S. *Web services et Interopérabilité des SI : WS-I, WSAD/J2EE, Visual Studio .Net et BizTalk.* Paris : Dunod, 2004.

[Morejon 94] Morejon J. *MERISE, vers une modélisation orientée objet*. Paris : Les Editions d'Organisation, 1994.

[Moser et Karlton 71] Moser C. and Karlton G. *Survey methods in social investigation*. 2nd Edition. London : Hermann, 1971.

[Murgue 05] Murgue T. De l'importance du pré-traitement des données pour l'utilisation de l'inférence grammaticale en Web Usage Mining. In *Workshop Modélisation Utilisateurs et Personnalisation de l'Interaction Homme-Machine*, Extraction et Gestion de Connaissances (EGC 2005), Paris, 2005.

[Nanci et Espinasse 01] Nanci D. et Espinasse B. *Ingénierie des systèmes d'information : MERISE.* 4ème édition. Paris : Vuibert, 2001.

[Newell 82] Newell A. The knowledge level. *Artificial intelligence*, 18, pp. 87-127, January 1982.

[Nielsen 93] Nielsen J. *Usability Engineering*. Academic Press : London, 1993.

[Nielsen et Mack 94] Nielsen J. and Mack R. L. *Usability inspection methods*. New York : john Wiley & Sons, 1994.

[Nikraz et al., 06] Nikraz M., Caire G. and Bahri P.A. A Methodology for the Analysis and Design of Multi-Agent Systems using JADE. *International Journal of Computer Systems Science & Engineering*, special issue on A. Garcia and C. Lucena (Eds.), "*Software Engineering for Multi-Agent Systems*". Vol. 21 (2), 2006.

[Nodine et al., 03] Nodine M., Ngu A.H.H., Cassandra A. and Bohrer W.G. Scalable Semantic Brokering over Dynamic Heterogeneous Data Sources in InfoSleuth™. *IEEE Transactions on Knowledge and Data Engineering*, vol. 15(5), pp. 1082-1098, 2003.

[Norman 83] Norman D. A. Some observations on mental models. In *Mental models*, D. Gentner and A. L. Stevens (Eds.), pp. 7-14, New Jersey : Lawrence Erlbaum, 1983.

[Nurcan 96] Nurcan S. Analyse et conception de systèmes d'information coopératifs. *Technique et Science Informatiques*, 15(9), pp. 1287-1315, 1996.

[O'Grady et al., 05] O'Grady M.J., O'Hare G.M.P. and Sas C. Mobile agents for mobile tourists: a user evaluation of Gulliver's Genie. *Interacting with Computers*, 17, pp. 343-366, 2005.

[O'Hare et O'Grady 03] O'Hare G.M.P. and O'Grady M.J. Gulliver's Genie: a multi-agent system for ubiquitous and intelligent content delivery. *Computer Communications*, 26, pp.1177-1187, 2003.

[Odell et al., 00] Odell J., Van Dyke Parunak H. and Bauer B. Extending UML for Agents. *Proceeding of the Agent-Oriented Information Systems Workshop at the 17th National conference on Articial Intelligence*. Gerd Wagner, Yves Lesperance and Eric Yu eds., Austin, TX, pp. 3-17, accepted paper, *AOIS Workshop at AAAI*, 2000.

[Odell et al., 99] Odell J., Van Dyke Parunak H and Bock C. Representing agent interaction protocols in UML. In *OMG Document/ad99-12-01*. Intellicorp Inc. December 1999.

[Oppermann et Simm 94] Oppermann R. et Simm H. Adaptability : User-initiated individualization. In R. Oppermann (Ed.), *Adaptative User Support: ergonomic design of manually and automatically adaptable software*, pp. 14-66, Lawrence Earlbaum, 1994.

[Oppliger 04] Oppliger R. Microsoft .NET Passport and identity management. *Information Security Technical Report*, Vol 9, Issue 1, pp. 26-34, January-March 2004.

[Paquel et Bezaut 02] Paquel N. et Bezaut O. *XML et développement des EDI*. Paris : Hermès, 2002.

[Pascot et Bernadas 93] Pascot D. et Bernadas C. L'essence des Méthodes : Etude Comparative de Six Méthodes de Conception de Systèmes d'Information Informatisés. *Actes du congrès INFORSID'93 « Systèmes d'information, systèmes à base de connaissances »,* Lille, 11-14 Mai 1993.

[Pearl 88] Pearl J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo : Morgan Kaufmann Publishers, 1988.

[Pennock et al., 00] Pennock D., Horvitz E., Lawrence S. and Giles C. Collaborative Filtering by Personality Diagnosis : A Hybrid Memory- and Model-Based Approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, UAI-2000, Morgan Kaufman, San Francisco, 2000, pp. 473-480.

[Perreau 02] Perreau C. *Les systèmes d'information multimodale : apports et potentialités dans l'optimisation des déplacements urbains*. Thèse de doctorat, Institut d'Etudes Politiques, Paris, juillet 2002.

[Petit-Rozé 03] Petit-Rozé C. *Organisation Multi-Agent au service de la personnalisation de l'information : Application à un système d'information multimodale pour le transport terrestre de personnes*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Décembre 2003.

*[Petit-Rozé et al., 03a] Petit-Rozé C., Anli A., Grislin-Le Strugeon E., Abed M. et Kolski C. *AGENPERSO : Interface homme-machine à base d'AGENts Logiciels PERSOnnels d'information aux usagers des TC*. Rapport final de projet PREDIT, LAMIH, Valenciennes, janvier 2003.

*[Petit-Rozé et al., 03b] Petit-Rozé C., Kolski C., Grislin-Le Strugeon E., Anli A., Abed M. et Uster G. AgenPerso : Interface homme-machine à base d'agents logiciels personnels d'information au service des usagers des transports collectifs. *15ème Conférence Francophone sur l'Interaction Homme-Machine*. Caen, ACM Press, Novembre 2003.

*[Petit-Rozé et al., 04] Petit-Rozé C., Anli A., Grislin-Le Strugeon E., Abed M., Uster G. et Kolski C. Système d'Information Transport Personnalisée à base d'agents logiciels. *Génie Logiciel*, 70, pp. 29-38, 2004.

[Petit-Rozé et Grislin-Le Strugeon 02] Petit-Rozé C. et Grislin-Le Strugeon E. Systèmes d'information à base d'agents. In R. Mandiau, E. Grislin-Le Strugeon, A Péninou (Eds.), *Organisation et application des SMA*, Paris : Hermès, pp. 307-319, 2002.

[Petit-Rozé et Grislin-Le Strugeon 05] Petit-Rozé C. et Grislin-Le Strugeon E. Modélisation d'interactions par et avec des agents en personnalisation d'information. In A. Herzig, Y.

Lespérance, A.i. Mouaddib (Eds.), *Actes des Troisièmes Journées Francophones Modèles Formels de l'Interaction (MFI'2005)* (Caen, France, 25-27 Mai), Cepadues-Editions, Mai 2005.

[Pfitzmann 04] Pfitzmann B. Privacy in enterprise identity federation – policies for Liberty 2 single sign on. *Information Security Technical Report*, Vol 9, Issue 1, pp. 45-58, January-March 2004.

[Pohl 99] Pohl W. Logic-Based Representation and Reasoning for User Modeling Shell Systems. *User Modelling and User-Adapted Interaction*, Vol. 9, Number 3, pp. 217-282, 1999.

[Pohl et Höhle 97] Pohl W. and Höhle J. Mechanisms for flexible representation and use of knowledge in user modeling shell systems. In Jameson, A., Paris, C. and Tasso, C. (eds), *User Modeling: Proceedings of the Sixth International Conference*, UM97. New York : Springer Wien, pp. 403-414, 1997.

[Poulon et Vaudry 03] Poulon A. et Vaudry C. Recherche d'informations : de l'utilisateur au contexte. In *15ème Conférence Francophone sur l'Interaction Homme-Machine, IHM'03*. Caen, France, novembre 2003.

[Pree 98] Pree W. *Design patterns et architectures logicielles*. Paris : Vuibert 1998.

[Puerta et Eisentein 06] Puerta A. and Eisentein J. *XIML : A universal language for user interfaces*. Accessible sur le web en janvier 2006 sur : http://www.ximl.org.

[Rao et Georgeff 91] Rao A.S. and Georgeff M.P. Modeling Rational Agents within a BDI Architecture. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, pp. 473-484, 1991.

[Rich 96] Rich C. Window sharing with collaborative interface agents. *SIGCHI Bulletin*, 28(1), pp. 70-78, January 1996.

[Rizcallah 00] Rizcallah M. *Construire un annuaire d'entreprise avec LDAP*. Paris : Eyrolle, 2000.

[Roques 05] Roques P. *UML 2 par la pratique – Etudes de cas et exercices corrigés*. Paris : Eyrolles, 2005.

[Roques et Vallée 00] Roques P. et Vallée F. *UML en action : de l'analyse des besoins à la conception en java*. Paris : Eyrolles, 2000.

[Rosenberg et Scott 99] Rosenberg D. and Scott K. *Use Case Driven Object Modeling with UML : a Pratical Approach*. Addison Wesley : Longman, 1999.

[Rouillard 02] Rouillard J. A multimodal E-commerce application coupling HTML and VoiceXML. In *the eleventh international world wide web conference* (www 2002). Honolulu, Hawaii, USA, 7-11 may 2002.

[Rouillard 04] Rouillard J. *VoiceXML, le langage d'accès à Internet par téléphone*. Paris : Vuibert, 2004.

[Routier et al., 01] Routier J.C., Mathieu P. and Secq Y. Dynamic skill learning : A support to agent evolution. In *Proceedings of the AISB'01 Symposium on Adaptive Agents and Multi-Agent Systems*, pp. 25-32, University of York, United Kingdom, March 2001.

[Royce 70] Royce W.W. Managing the Development of Large Software Systems. *Proceedings of IEEE WESCON*. August 1970.

[Salton 83] Salton G. and McGill M. *Introduction to modern information retrieval*. New York : McGraw-Hill, 1983

[Salton et Buckley 88] Salton T. and Buckley C. Term-Weighting Approaches in Automated Text Retrieval. *Information Processing and Management*, Vol 24(5), pp. 513-523, 1988.

[Schiaffino et Amandi 00] Schiaffino S. N and Amandi A. User profiling with Case-Based Reasoning and Bayesian Networks. In *Open Discussion Track Proceedings – International Joint Conference, IBERAMIA-SBIA 2000,* Atibaia, Brazil, pp. 12-21, 2000.

[Schneider-Hufschmidt et al., 93] Schneider-Hufschmidt M., Kühme T. and Malinkowski U. (Eds.). *Adaptive User Interfaces*. North Holland, 1993.

[Secq 03] Secq Y. *RIO : Rôles, Interactions et Organisations. Une méthodologie pour les systèmes multi-agents ouverts*. Thèse de doctorat, Université des sciences et technologies de Lille, décembre 2003.

[Senach 90] Senach B. *Evaluation ergonomique des interfaces Homme-machine : une revue de la littérature*. Rapport de recherche, INRIA, Sophia Antipolis, n° 1180, 1990.

[Shardanand et Maes 95] Shardanand U. and Maes P. Social Information Filtering : Algorithms for Automating "World of Mouth". In *Proceeding of the CHI-95 Conference*. Denver, USA, ACM Press, May 1995.

[Soltysiak et Crabtree 98] Soltysiak S. J. and Crabtree I. B. Automatic learning of user profiles – towards the personalization of agent services. *BT Technologie*, Vol. 16 N° 3, pp. 110-117, July 1998.

[Soutou 02] Soutou C. *De UML à SQL : Conception de bases de données*. Editions Eyrolles : Paris, 2002.

[Stephanidis et al., 01] Stephanidis C., Paramythis A, Sfyrakis M. and Savidis A. A case Study in Unified User Interface Development: The AVANTI Web Browser. In C. Stephanidis (Ed.), *User Interfaces for All – concepts, methods and tools*. pp. 525-568, NJ Mahwah : Lawrence Erlbaun Associates, 2001.

[Tabary 01] Tabary D. *Contribution à TOOD, une méthode à base de modèles pour la spécification et la conception des systèmes interactifs*. Mémoire de Doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, janvier 2001.

[Tardieu et al., 00] Tardieu H., Rochfield A. et Colletti R. *La méthode MERISE, tome 1 : principes et outils*. Editions d'organisation : Paris, 2000.

[Tardieu et al., 83] Tardieu H., Rochfield A. et Colletti R. *La méthode MERISE, tome 1 : principes et outils*. Editions d'organisation : Paris, 1983.

[Thévenin et Coutaz 99] Thévenin D. et Coutaz, J. Plasticity of user interfaces: framework and research agenda. In *Proceedings of Interact'99 seventh IFIP Conference on Human-Computer Interaction*, Edinburgh, Scotland, 1999.

[Toffolon 99] Toffolon C. *Cours de genie logiciel ou comment amener de petits génies à écrire des grands logiciels*. Cours de DESS-ICC, Université du Littoral, 1999.

[Trabelsi et al., 06] Trabelsi A., Ezzedine H. et Kolski C. Un mouchard électronique orienté agent pour l'évaluation de systèmes interactifs de supervision. In *Conférence Internationale Francophone d'Automatique CIFA'2006,* Bordeaux, France, Mai 2006.

[Trousse et al., 99] Trousse B., Jaczynski M. and Kanawati R. Using user behavior similarity for recommandation computation : The broadway approach, In *Proceedings of the 8th international conference on Human Computer Interaction (HCI'99)*, Munich, August 1999.

[Ungar et Foster 98] Ungar L.H. and Foster D.P. Clustering Methods for Collaborative Filtering. AAAI Workshop on Recommendation Systems, M. Kaufman (Ed.) « *Empirical analysis of predictive algorithms for collaborative filtering. Uncertainty in Artificial Intelligence* », *Proceedings of the 14th Conference*, pp. 43-52, 1998.

[Uster 04] Uster G. Pour une mobilité raisonnée. In D. Kaplan et H. Lafont (Eds.), *Mobilités.net : Villes, transports, technologies face aux nouvelles mobilités*, pp. 324-320, Collection « Questions numériques », 2004.

[Uster 98] Uster G. *Information multimodale*. Document interne, INRETS, 1998.

[Van Harmelen et al., 01] Van Harmelen F., Patel-Schneider P. and Horrocks I. *A Model-Theoretic Semantics for DAML+OIL* (March 2001) Revision 4.1. 2001. Available from : http://www.daml.org/2001/03/model-theoretic-semantics.html

[Virvou et Du Boulay 99] Virvou M. and Du Boulay B. Human plausible reasoning for intelligent help. *User Modeling and User-Adapted Interaction*, Vol. 9, No. 4, pp. 321-375, 1999.

[Virvou et Kabassi 02] Virvou M. and Kabassi K. IFM: An Intelligent Graphical User Interface Offering Advice. In *2nd Hellenic Conf. on AI, SETN-2002*, Thessaloniki, Greece, pp. 155-164, April 2002.

[Waern et al., 98] Waern A., Averman C., Tierney M., Rudström A. and Laaksolahti J. *ConCall: an information service for researchers based on EdInfo*. Research report T98:04, Swedish Institute of Computer Science, October 1998.

[Wooldridge et al., 00] Wooldridge M., Jennings N.R. and Kinny D. The Gaia Methodology for Agent-Oriented Analysis and Desing. *Journal of Autonomous Agents and Multi-Agent Systems*, 3(3), pp. 285-312, 2000.

[Zaniolo 91] Zaniolo C. *The Logical Data Language (LDL): An Integrated Approach to Logic and Databases*. MCC Technical Report STP-LD-328-91, 1991.

[Zimmermann et al., 05] Zimmermann A., Specht M. and Lorez A. Personalization and Context Management. *User Modeling and User-Adapted Interaction*, 15, pp. 275-302, 2005.

ANNEXE A
Exemple d'un mail envoyé par « mon service transport »

(Illustration des propos du §5.3.8, chapitre 5)

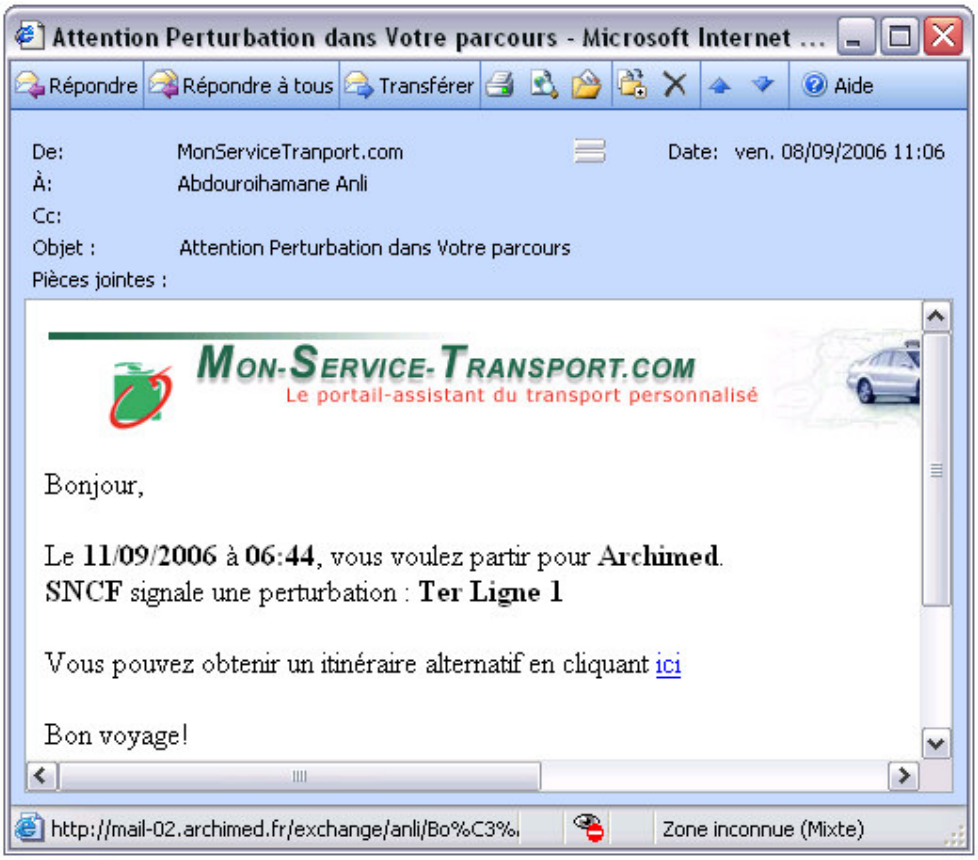


Figure A. 1. Exemple d'un mail envoyé à l'usager

ANNEXE B
Exemple d'une autre application développée avec PerSyst suivant la méthode PerMet

(cf. Perspectives liées à la méthode)

Cette application a été réalisée par Yousra Gassara dans le cadre de son projet de fin d'études pour l'obtention du diplôme d'ingénieur en génie informatique de l'Ecole Nationale d'Ingénieur de Sfax (Tunisie). L'un des objectifs de son projet était de fournir une application pour PDA permettant la recherche d'itinéraire personnalisé.

L'application a été développée en C# et intéragit avec le même système de personnalisation (PerSyst) que celui de "mon service transport" pour la personnalisation des itinéraires. La Figure B.1 présente l'interface de recherche des itinéraires et celle en Figure B.2 donne le resultat personnalisé.

Figure B. 1. Recherche d'itinéraire

Figure B. 2. Détail des trajets

ANNEXE C
Le projet Mouver.Perso

(Cette annexe a pour objectif de complèter les explications du §5.2.6, chapitre 5)

Mouver.Perso (MObilité et mUltimodalité Voyageurs Etudiants en Région Nord Pas de Calais) est un projet PREDIT (Programme national de Recherche et D'Innovation dans les Transports terrestres) financée par la PREDIM (Plate-forme de Recherche et d'Expérimentation pour le Développement de l'Information Multimodale). Ce projet d'une durée de 24 mois (janvier 2005 - décembre 2006) est réalisé au sein du LAMIH en collaboration avec l'INRETS et la société Archimed.

L'objectif de ce projet est d'assurer une passerelle entre le monde de la mobilité et la société de l'information. Cela consiste à mettre en œuvre des services d'accompagnement personnalisés autour de la mobilité quotidienne des étudiants en Nord-Pas de Calais. Les étudiants disposent d'un emploi du temps initialement "statique" (mais qui évolue au cours du temps). Autour de cet emploi du temps, ils occupent leurs temps libres de différentes manières : sports et loisirs, shopping, études, cultures. . . L'originalité de ce projet est d'accompagner les étudiants tout au long de leurs occupations de manière personnalisée

- en terme de déplacements (moyen de transport préféré, coût du déplacement, . . .)
- en terme d'organisation des journées (par rapport aux préférences des étudiants et en fonction de l'offre de transport).

La réalisation de ce projet a été effectuée suivant la méthode PerMet. L'application est un agenda adapté pour PDA accessible au travers du web. Dans PerSyst, trois modèles d'agents ont été définis. Les agents de *Recherche* pour la recherche des itinéraires, les agents de *Gestion de profil* pour la recommandation des itinéraires et les agents *Agenda* qui s'occupent de la gestion du planning de l'étudiant. Les figures C1 et C2 fournissent quelques illustrations de l'application (l'application a été développée en PHP). La Figure C1 illustre l'interface d'édition d'un rendez-vous. L'étudiant peut remplir l'objet du rendez-vous, le lieu du rendez-vous, les heures, etc. La Figure C2 donne un exemple de visualisation d'un itinéraire associé à un déplacement dans l'agenda. L'application est dans la phase d'évaluation.

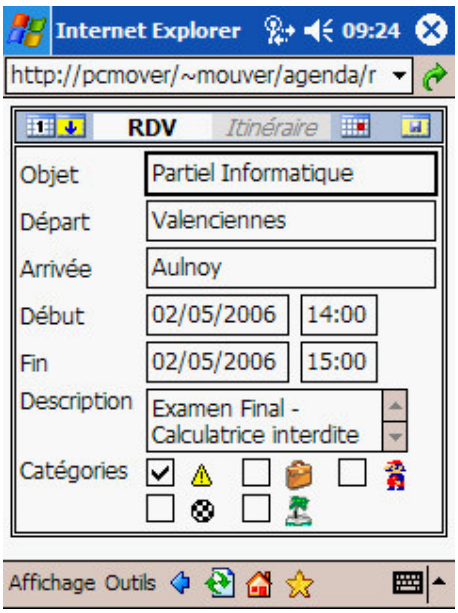


Figure C. 1. Edition d'un rendez-vous

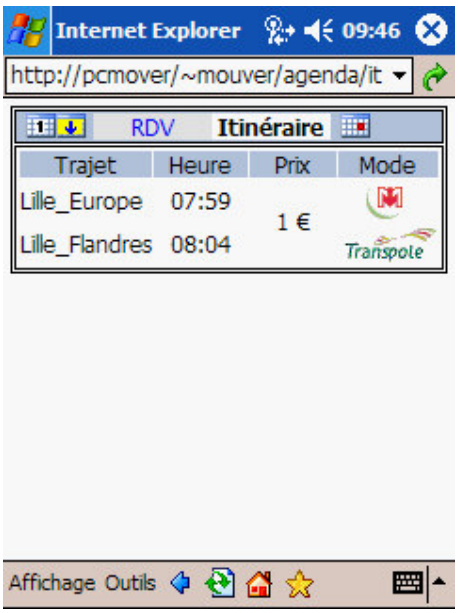


Figure C. 2. Visualisation d'un itinéraire

Annexe D
Exemple de profil d'un utilisateur

(cf. §5.1.5, chapitre 5)

```
<?xml version="1.0" encoding="UTF8" ?>
 <ENTRY>
   <Static>
       <DN>LDAP://SRVETCLDC01:389/CN=test,OU=Sitp,DC=DOMTRAN
       SPORT,DC=local</DN>
    </Static>
   <TransportPreferences>
      <TMarche total="35.0" sum="4">8.75</TMarche>
      <NMode total="15.0" sum="3">5.0</NMode>
      <TDuree total="18.333333333333332"
         sum="4">4.583333333333333</TDuree>
      <TCout total="12.5" sum="4">3.125</TCout>
    </TransportPreferences>
   <Historic>
     <TransportHistoric>
       <Request>
          <LieuDepart>Lille_Flandres</LieuDepart>
          <LieuArrivee>Lille_Europe</LieuArrivee>
          <HeureDepart>08:00</HeureDepart>
          <HeureArrivee>08:05</HeureArrivee>
          <Date>09/09/2005</Date>
        </Request>
       <Result>
         <Trajet>
            <Depart>Lille_Flandres</Depart>
            <Arrivee>Lille_Europe</Arrivee>
            <TC>Metro</TC>
            <Exploitant>Transpole</Exploitant>
            <Ligne>Ligne 2</Ligne>
            <Heure_Depart>08:00</Heure_Depart>
            <Heure_Arrivee>08:05</Heure_Arrivee>
            <Prix>1</Prix>
            <Duree>5</Duree>
          </Trajet>
        </Result>
       <Choice>
          <Result>1</Result>
          <Reason>DEFAULT</Reason>
        </Choice>
      </TransportHistoric>
     <TransportHistoric>
       <Request>
          <LieuDepart>LAMIH</LieuDepart>
          <LieuArrivee>Archimed</LieuArrivee>
          <HeureDepart>06:44</HeureDepart>
          <HeureArrivee>08:10</HeureArrivee>
```

```
            <Date>11/09/2006</Date>
          </Request>
          <Result>
            <Trajet>
              <Depart>Aulnoy Universite</Depart>
              <Arrivee>Gare de Valenciennes</Arrivee>
              <TC>Bus</TC>
              <Exploitant>Semurval</Exploitant>
              <Ligne>Ligne 1</Ligne>
              <Heure_Depart>06:46</Heure_Depart>
              <Heure_Arrivee>07:00</Heure_Arrivee>
              <Prix>1.30</Prix>
              <Duree>14</Duree>
            </Trajet>
            <Trajet>
              <Depart>Gare de Valenciennes</Depart>
              <Arrivee>Gare Lille Flandres</Arrivee>
              <TC>Ter</TC>
              <Exploitant>SNCF</Exploitant>
              <Ligne>Ligne 1</Ligne>
              <Heure_Depart>07:25</Heure_Depart>
              <Heure_Arrivee>07:56</Heure_Arrivee>
              <Prix>7.30</Prix>
              <Duree>31</Duree>
            </Trajet>
            <Trajet>
              <Depart>Gare Lille Flandres</Depart>
              <Arrivee>Porte des Postes</Arrivee>
              <TC>Metro</TC>
              <Exploitant>Transpole</Exploitant>
              <Ligne>Ligne 1</Ligne>
              <Heure_Depart>08:05</Heure_Depart>
              <Heure_Arrivee>08:15</Heure_Arrivee>
              <Prix>1.30</Prix>
              <Duree>10</Duree>
            </Trajet>
          </Result>
          <Choice>
            <Result>9</Result>
            <Reason>LAST</Reason>
          </Choice>
        </TransportHistoric>
      </Historic>
    </ENTRY>
```

Annexe E
Fiche de description textuelle pour le cas d'utilisation "Rechercher un itinéraire"

(cf. §5.1.1, chapitre 5)

Sommaire d'identification :

Titre : Rechercher un itinéraire

But : fournir un itinéraire à un utilisateur en se basant sur ses préférences.

Résumé : reconnaître l'utilisateur, envoyer la requête à PerSyst et afficher l'itinéraire personnalisé à l'utilisateur.

Acteurs : utilisateur (principal), PerSyst (secondaire)

Date de création : 10/05/2004 **Date de mise à jour** : 03/02/2005

Version : 1.0 **Responsable** : A. Anli

Description des enchaînements :

Préconditions :

- L'utilisateur est authentifié

Scénario nominal :

1. l'utilisateur formule et valide sa requête.
2. le système envoie la requête de l'utilisateur à PerSyst.
3. PerSyst recherche les itinéraires répondant à la requête et les envoie au système en précisant l'itinéraire susceptible d'intéresser l'utilisateur.
4. le système affiche l'itinéraire susceptible d'intéresser l'utilisateur.

Scénarios alternatifs :

A1 : Lieu de départ non fourni

Ce scénario commence après l'étape 1 du scénario nominal.

2. informer l'utilisateur que le lieu de départ est obligatoire.

Revenir à l'étape 1 du scénario nominal.

A2 : Lieu d'arrivée non fourni

Ce scénario commence après l'étape 1 du scénario nominal.

2. informer l'utilisateur que le lieu d'arrivée est obligatoire.

Revenir à l'étape 1 du scénario nominal.

A3 : Heure d'arrivée non fournie

Ce scénario commence après l'étape 1 du scénario nominal.

2. informer l'utilisateur que l'heure d'arrivée est obligatoire.

Revenir à l'étape 1 du scénario nominal.

A4 : Pas d'itinéraire trouvé pour la requête

Ce scénario commence après l'étape 2 du scénario nominal.

3. PerSyst envoie une réponse vide

4. le système informe l'utilisateur qu'il n'y a pas d'itinéraire satisfaisant sa requête

Revenir à l'étape 1 du scénario nominal.

Scénarios d'erreur :

E1 : PerSyst ne répond pas ou envoie une réponse non conforme au modèle de donnée défini

Ce scénario commence après l'étape 2 du scénario nominal.

3. le système affiche un message d'erreur.

Revenir à l'étape 1 du scénario nominal.

Postconditions :

- Mise à jour du profil de l'utilisateur

Besoins d'IHM :

- Pour formuler la requête

L'utilisateur doit pouvoir formuler sa requête pour la recherche d'itinéraire en précisant le lieu de départ, le lieu d'arrivée et l'heure d'arrivée. La Figure E. 1 donne un prototype d'interface possible.

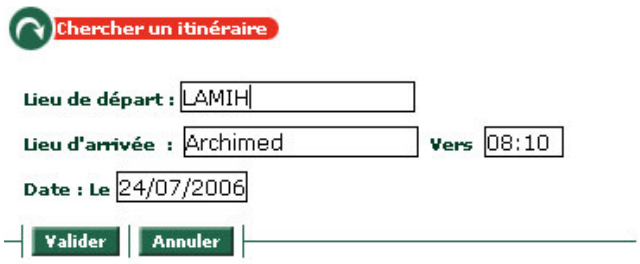


Figure E. 1. Prototype d'interface pour la formulation de la requête

- Pour lister les itinéraires

Le système affiche le détail de l'itinéraire en précisant pour chaque trajet composant l'itinéraire, les gares de départ et d'arrivée, les horaires, la durée, le prix, l'exploitant et le plan d'accès. L'utilisateur dispose d'une vue globale sur tous les itinéraires possibles (voir Figure E. 2).

Figure E. 2. Prototype d'interface pour l'affichage des itinéraires

Contraintes non fonctionnelles :

Contraintes	Descriptifs
Temps de réponse	Les itinéraires doivent être affichés en moins de deux secondes.
Disponibilité	Le service doit être accessible aux utilisateurs 24 heures sur 24 et 7 jours sur 7
Confidentialité	Les noms des utilisateurs participant à la recommandation d'un itinéraire ne doivent pas être mentionnés dans la réponse fournie.

Tableau E. 1. Les contraintes non fonctionnelles

Annexe F
Mesures pour le calcul de similarité entre deux objets

(cf. §4.3.3, chapitre 4)

Cette annexe présente quelques mesures pour le calcul de similarité entre deux objets $\vec{X}_i$ et $\vec{X}_j$. Ces mesures pourraient servir pour le calcul de la similarité entre deux contextes, entre deux profils utilisateur, etc.

- Distance de Manhattan :

$$d_1\left(\vec{X}_i,\vec{X}_j\right)=\left|\vec{X}_i-\vec{X}_j\right|=\sum_{k=1}^{n}\left|X_{ik}-X_{jk}\right|$$

- Distance euclidienne :

$$d_2\left(\vec{X}_i,\vec{X}_j\right)=\left\|\vec{X}_i-\vec{X}_j\right\|_2=\sqrt{\sum_{k=1}^{n}\left(X_{ik}-X_{jk}\right)^2}$$

- Distance de Minkowski :
(généralisation de la distance de Manahattan et de la distance enclidienne)

$$d_p\left(\vec{X}_i,\vec{X}_j\right)=\left\|\vec{X}_i-\vec{X}_j\right\|_p=\sqrt[p]{\sum_{k=1}^{n}\left|X_{ik}-X_{jk}\right|^p}$$

- Cosinus :

$$Cos\left(\vec{X}_i,\vec{X}_j\right)=\frac{\vec{X}_i\bullet\vec{X}_j}{\left\|\vec{X}_i\right\|\bullet\left\|\vec{X}_j\right\|}=\frac{\sum_{k=1}^{n}\left(X_{ik}\bullet X_{jk}\right)}{\sqrt{\sum_{k=1}^{n}X_{ik}^2\bullet\sum_{k=1}^{n}X_{jk}^2}}$$

- Corrélation de Pearson :
(cosinus de l'écart à la moyenne des deux objets)

$$Pearson\left(\vec{X}_i,\vec{X}_j\right)=Cos\left(\vec{X}_i-\overline{X}_i,\vec{X}_j-\overline{X}_j\right)=\frac{\sum_{k=1}^{n}\left(X_{ik}-\overline{X}_i\right)\left(X_{jk}-\overline{X}_j\right)}{\sqrt{\sum_{k=1}^{n}\left(X_{ik}-\overline{X}_i\right)^2\sum_{k=1}^{n}\left(X_{jk}-\overline{X}_j\right)^2}}$$

TITRE

Méthodologie de développement des systèmes d'information personnalisés : Application à un système d'information au service des usagers des transports terrestres de personnes

RESUME

Cette thèse propose une méthode pour l'analyse, la conception et la modélisation des systèmes d'information personnalisés. Cette méthode que nous appelons PerMet (PERsonalization METhodology) permet aussi bien la mise en place d'un nouveau système d'information personnalisé que la personnalisation d'un système d'information déjà existant. Elle sépare le système d'information vu comme un ensemble de services du système de personnalisation pour prendre en compte la multi-modalité, le multi-canal et l'aspect multi-plateforme. PerMet propose un modèle de développement itératif, incrémental et permet une réalisation parallèle des phases spécifiques liées au développement des services et des phases spécifiques liées à la personnalisation. PerMet utilise des formalismes basés sur celles d'UML et de ses extensions.

Pour faciliter l'utilisation de PerMet, nous proposons PerSyst (PERsonalization SYSTem), un système de personnalisation générique pouvant s'utiliser conjointement avec PerMet pour le développement de système d'information personnalisé. PerSyst est construit à partir d'une architecture multi-agents ce qui lui confère une flexibilité accrue grâce aux caractéristiques, entre autres, d'adaptabilité, d'autonomie, de reproductibilité et de mobilité des agents logiciels.

Les deux contributions PerMet et PerSyst ont été appliquées pour le développement d'un système d'information transport personnalisé.

Mot-clés : méthodologie, personnalisation, architecture, agents logiciels, transport

---

TITLE

Development methodology of the personalized information systems: Application to an information system dedicated to traveller information

ABSTRACT

This thesis proposes a methodology for the analysis, the design and the modelling of the personalized information systems. This methodology called PerMet (PERsonalization METhodology) allows as well the development of a new personalized information system as the personalization of an already existing information system. It separates the information system seen like a set of services of the personalization system to take into account the multimodality, the multi-channel and the multi-device. PerMet is iterative, incremental and allows a parallel realization of the specific phases related to the development of the services and the specific phases related to personalization. PerMet uses UML based formalisms and its extensions.

To facilitate the use of PerMet, we propose PerSyst (PERsonalization SYSTem), a generic personalization system which can be used jointly with PerMet for the development of personalized information system. PerSyst is built from a multi-agent architecture that confers an increased flexibility thanks to the software agents characteristics like adaptability, autonomy, reproducibility and mobility.

The two contributions PerMet and PerSyst were applied for the development of a personalized traveller information system.

Keywords: methodology, personalization, architecture, software agent, transport