

Projet CPER MOSAIQUES
MOdèles et infraStructures pour Applications ubIQUITairES
Programme TAC 2004-2007
thème 2.1 Logiciel pour la Communication
USTL/UVHC/EMD/INRETS
Rapport de fin de projet
Début de projet février 2005

Responsable scientifique Laurence Duchien
Laurence.Duchien@lifl.fr

25 juin 2007

Table des matières

1	Bilan Scientifique	7
1.1	Rappel du contexte et des objectifs généraux	7
1.2	Bilan Scientifique	8
1.2.1	Objectifs définis à l'origine et atteints	8
1.2.2	Autres objectifs atteints	8
1.2.3	Principaux résultats obtenus	8
1.2.4	Programme de travaux réalisés	9
1.2.5	Difficultés rencontrées du point de vue scientifique	10
1.3	Bilan concernant la mise en œuvre du programme	10
1.3.1	Stratégies et méthodes mises en œuvre	10
1.3.2	Calendrier de travail respecté	10
1.3.3	Difficultés rencontrées dans la mise en œuvre du programme	11
1.4	Bilan concernant les moyens affectés au programme	11
1.4.1	Moyens humains mobilisés	11
1.4.2	Moyens matériels affectés	12
1.4.3	Collaborations engagées avec d'autres partenaires régionaux, nationaux et internationaux	12
2	Bilan de l'Axe 1 : L'Adaptabilité dans un Support Méthodologique	13
2.1	Introduction	13
2.2	Modèles d'une application MOSAIQUES	13
2.2.1	Le contexte dans MOSAIQUES	13
2.2.2	La structure d'une application	14
2.2.3	Modèle dynamique de l'adaptation	15
2.3	Processus de développement	16
2.3.1	Micro-processus de modélisation	17
2.3.2	Projections des modèles	18
2.4	Nouvelles applications vs applications existantes	18
2.4.1	La transformation de programmes pour environnements ubiquitaires	19
3	Bilan de l'Axe 2 : Infrastructure Logicielle, Support de l'Adaptabilité	21
3.1	Introduction	21
3.2	Architecture d'Intermédiation	22
3.3	Auto-Fractal	22
3.4	DACAR (Distributed Autonomous Component-based Applications Reconfiguration)	23

3.5	FDF : Un framework générique pour automatiser le déploiement logiciel . . .	24
3.6	JITS	24
3.7	OpenCCM-MOSAIQUES : Plate-forme intergielle	25
3.8	STAN	26
3.9	Système multi-agent distribués	27
3.10	UbiquiTalk	29
4	Bilan de l’Axe 4 : Démonstration de Faisabilité	31
4.1	Introduction	31
4.2	Action 4.1 - Applications liées aux transports	31
4.2.1	L’application “ trains au départ ”	31
4.2.2	L’application “ notification d’arrêt ”	34
4.2.3	Conclusion	36
4.2.4	Retombées de l’axe	37
4.3	Action 4.2 - Applications liées aux E-services	37
4.3.1	Environnement augmenté	37
4.3.2	Interactions multimodales	38
4.3.3	Valorisation	38
5	Ecole des Mines de Douai	43
5.1	Liste des participants	43
5.2	Contribution sur l’adaptation structurelle de systèmes à base de composants logiciels dans les environnements ubiquitaires	43
5.2.1	Problématique	43
5.2.2	Résultats	44
5.3	Contribution sur l’auto-adaptation de composants logiciels pour une gestion efficace des ressources limitées dans les environnements ubiquitaires	44
5.3.1	Problématique	44
5.3.2	Résultats	44
5.4	Contribution sur les architectures logicielles ouvertes et dynamiques pour les environnements ubiquitaires	45
5.4.1	Problématique	45
5.4.2	Résultats	45
5.5	Contribution avec l’ensemble des équipes impliquées dans le cadre du projet MOSAIQUES à la définition d’un modèle structurel et dynamique d’une application ubiquitaire	45
5.6	Contribution sur l’adaptation dynamique par ré-assemblage d’applications ubiquitaires à base composants	45
5.6.1	Problématique	45
5.6.2	Résultats	46
5.7	Contribution sur les architectures d’agents logiciels adaptables pour la gestion de ressources limitées et variables	46
5.7.1	Problématique	46
5.7.2	Résultats	46
5.8	Logiciels développés	47

6	INRETS - Laboratoire LEOST	49
6.1	Liste des participants	49
6.2	Objectifs et problèmes développés	49
6.2.1	Cycle de vie d'une application contextuelle	50
6.2.2	Le mécanisme de découverte	50
6.3	Les résultats	51
6.3.1	Définition de scenario applicatifs	51
6.3.2	Exemple d'application pour l'axe 1	52
6.3.3	Proposition d'extension de la plate-forme OpenCCM dans l'axe 2	53
6.3.4	Réalisation d'applications dans l'axe 4	53
6.4	Logiciels développés	53
6.5	Retombées du projet	53
7	USTL - Laboratoire LIFL - Equipe GOAL	55
7.1	Liste des participants	55
7.2	Problèmes abordés et résultats	56
7.2.1	Cadre de développement	56
7.2.2	Les résultats dans l'axe 1	56
7.2.3	Les résultats dans l'axe 2	58
7.3	Logiciels développés	59
7.4	Thèses et HDR soutenues et en cours	60
7.5	Retombées du projet	60
8	USTL- Laboratoire LIFL- Equipe NOCE	63
8.1	Liste des participants	63
8.2	Problèmes abordés et résultats	63
8.2.1	Modélisation des applications ubiquitaires	63
8.2.2	Résultats	64
8.3	Logiciels développés	64
8.4	Retombées du projet	66
9	USTL- Laboratoire LIFL- Equipe RD2P	67
9.1	Liste des participants	67
9.2	Problèmes abordés et résultats	67
9.2.1	Contexte des travaux	67
9.2.2	Logique de déploiement	67
9.2.3	Outils de spécialisation	68
9.2.4	Support d'exécution minimaliste	68
9.3	Logiciels développés	68
9.4	Thèses et HDR soutenues et en cours	69
9.5	Retombées du projet	69
10	USTL-Laboratoire LIFL-Equipe SMAC	71
10.1	Liste des participants	71
10.2	Les objectifs et problèmes abordés	71
10.3	Les propositions effectuées et les résultats du projet	72
10.4	Thèses et HDR soutenues et en cours	75

11 USTL-Laboratoire LIFL- Equipe STC	77
11.1 Liste des participants	77
11.2 Problèmes abordés et résultats	77
11.2.1 Cadre de développement	77
11.2.2 Composition et extensibilité	77
11.2.3 Autonomie et sécurité des systèmes	78
11.2.4 Sécurité des données	78
11.2.5 Coopération des systèmes autonomes	78
11.3 Logiciels développés	79
11.4 Thèses et HDR soutenues et en cours	79
11.5 Retombées du projet	79
12 UVHC- Laboratoire LAMIH- Equipe ROI	81
12.1 Liste des participants	81
12.2 Objectifs et problèmes abordés	81
12.3 Les résultats	81
12.4 Logiciels développés	82
12.5 Thèses et HDR soutenues et en cours	82
12.6 Retombées du projet	83
13 Les retombées du programme de recherche et les perspectives	85
13.1 Les retombées du programme	85
13.1.1 Retombées sur le plan scientifique	85
13.1.2 Retombées sur le plan de la structuration de la recherche	87
13.1.3 Retombées sur le plan social, économique et culturel	87
13.1.4 Retombées sur le plan du rayonnement	88
13.1.5 Préoccupations de développement durable	89
13.2 Les perspectives de développement	90
13.3 Confidentialité	90

Chapitre 1

Bilan Scientifique

1.1 Rappel du contexte et des objectifs généraux

L'objectif de ce projet MOSAIQUES (MODèles et InfraStructures pour Applications ubI-QUitairES) est de définir un cadre de développement (méthodologie et outillage) pour la définition d'applications fonctionnant dans un environnement ubiquitaire. Ce projet étudie la prise en compte de la notion d'adaptabilité à l'exécution dans les applications et les infrastructures logicielles pour les applications ubiquitaires.

Le projet est constitué de 8 équipes (EMD, INRETS-LEOST, LIFL-GOAL, LIFL-RD2P, LIFL-STC, LIFL-SMAC, LIFL-NOCE, LAMIH-ROI) venant de l'Ecole des Mines de Douai, de l'INRETS, de l'Université des Sciences et Technologies de Lille, et de L'Université de Valenciennes et du Haut Cambrasis. Sept de ces équipes avaient déjà travaillé ensemble lors du contrat de plan état-région précédent. L'Ecole des Mines de Douai est venue dans ce projet renforcer le potentiel génie logiciel de la région.

L'objectif de ce projet était donc d'aborder la conception et la mise en oeuvre d'applications ubiquitaires selon les axes suivants :

1. Axe 1 : la définition d'un support méthodologique pour la construction d'applications ubiquitaires. Ce premier axe a pour objectif l'étude des méthodologies pour la conception des applications fonctionnant sur un environnement de type ubiquitaire ;
2. Axe 2 : la réalisation d'un support d'exécution avec des propriétés d'adaptabilité dans un environnement ubiquitaire. Celui-ci fournira les moyens nécessaires pour qu'un assemblage d'éléments logiciels puisse être adapté dynamiquement en fonction de l'environnement d'accueil, et par conséquent l'application qui le contient. Dans cet axe, nous étudions également un ensemble de services système et utilisateur. En effet, un certain nombre de problèmes tels que la sécurité ou encore la sûreté de fonctionnement doivent être revisités dans un tel contexte ;
3. Axe 3 : la validation de l'adaptabilité dans un environnement ubiquitaire. Celle-ci passe par la définition d'un ensemble de méthodes et d'outils permettant de prendre en compte l'adaptabilité que ce soit au niveau statique ou dynamique. Cet axe se focalisera sur la définition et la vérification d'une relation entre services offerts, ressources disponibles et qualité de service requise ;
4. Axe 4 : la démonstration de la faisabilité de l'approche par la réalisation d'applications dans des domaines particuliers comme les transports ou encore les E-services ubiquitaires

pour la E-Santé, la E-formation et le E-Commerce. Cette démonstration se fera en lien étroit avec le volet Nouveaux Usages du projet TAC.

1.2 Bilan Scientifique

1.2.1 Objectifs définis à l'origine et atteints

Les objectifs définis dans la réponse à projet en 2004 ont été atteints pour les différents axes du projet :

- Un support méthodologique permettant la prise en compte des particularités des applications ubiquitaires, et plus particulièrement de l'adaptabilité, a été étudié et validé dans l'axe 1.
- Dans l'axe 2, des études ont été menées afin de pouvoir supporter l'adaptabilité dans les infrastructures logicielles. Ces études nous ont conduits sur plusieurs pistes. Nous avons réduit par exemple au minimum les plates-formes à composants existantes telles que CCM pour pouvoir les embarquer sur des assistants numériques, de travailler sur des supports d'exécution minimalistes, ou encore de proposer des services adaptés à ces environnements ubiquitaires.
- L'axe 3 a rencontré très rapidement des difficultés, essentiellement dues au fait qu'il lui fallait dialoguer fortement avec les personnes travaillant sur le support méthodologique ou sur les plates-formes. En effet, il était nécessaire d'avoir défini pour cet axe ce que nous voulions valider. Pour cette raison, les personnes se sont jointes aux deux axes 1 et 2. Les résultats sont donc communs à ces deux axes. Il s'agit par exemple pour l'axe 1 de l'étude de la composition et de l'extensibilité et pour l'axe 2 de l'étude de l'autonomie et de la sécurité des systèmes et des données.
- L'axe 4 a consisté à la spécification et au développement de deux démonstrateurs, l'un sur le transport et l'autre sur les E-Services.

1.2.2 Autres objectifs atteints

Nous avons travaillé dans l'axe 1 sur une approche dirigée par les modèles, ce qui implique que nous avons pour objectif la réalisation de nouvelles applications pouvant fonctionner sur des environnements ubiquitaires. Cette approche ne supporte pas l'évolution d'applications existantes vers des environnements de type ubiquitaire. Nous avons mené une expérience permettant la transformation de programmes existants fonctionnant dans un environnement non ubiquitaire en programmes fonctionnant dans un environnement ubiquitaire. Ce travail a été possible parce que nous avons travaillé à la conception d'applications nouvelles et donc acquis des connaissances sur les environnements ubiquitaires.

1.2.3 Principaux résultats obtenus

Nous résumons ici les résultats de chaque axe. Le lecteur trouvera dans les chapitres suivants (de 2 à 12) les détails de ces résultats. Les chapitres 2, 3, et 4 décrivent les principaux résultats par axe, les chapitres 5 à 12 décrivent les résultats de chaque équipe. Le dernier chapitre décrit les retombées du programme et les perspectives de développement. Finalement nous donnons une bibliographie des communications faites dans le cadre du projet MOSAIQUES.

Les principaux résultats sont :

- Pour l’axe 1, la définition d’une méthodologie de modélisation pour l’intelligence ambiante a été établie. Nous avons montré une démarche complète permettant la prise en compte de l’adaptation dès les premières étapes du cycle de développement de l’application. Nous avons modélisé le contexte dans un méta-modèle et l’avons pris en compte dans la structure de l’application. L’application est alors constituée d’un ensemble de parties constituées de services fournis partageant un même contexte. Ces différentes parties peuvent ensuite être déployées sur différentes machines de technologies et d’environnements d’exécution différents. Les services sont contraints à des règles évaluées selon les disponibilités décrites dans le contexte. Un modèle dynamique de l’application a été proposé. Celui-ci est sujet aux modifications du contexte. Finalement, nous avons intégré le tout dans un processus de développement dirigé par les modèles en trois étapes : modélisation itérative de l’application, projection du modèle vers une plateforme technologique pour la production automatique du code ainsi que vers le modèle de déploiement, mise en œuvre des fonctionnalités de l’application dans le cadre de la technologie cible et raffinement du modèle de déploiement.
- Dans l’axe 2, la modélisation et le développement d’une infrastructure logicielle permettant l’adaptabilité des applications déployées sur les supports de l’ubiquité numérique ont été étudiés. L’étude avait pour objectifs d’identifier les spécificités des matériels informatiques impliqués, de caractériser les mécanismes fondamentaux capables de supporter les fonctionnalités de base des applications ubiquitaires et finalement de caractériser les fondements théoriques permettant de garantir la fiabilité des applications hébergées. Nous avons proposé plusieurs solutions se regroupant en trois familles : l’adaptation à la cible ubiquitaire, l’adaptation aux besoins applicatifs et finalement la fiabilisation de l’ubiquité. Plusieurs architectures ont été définies telles que l’architecture d’intermédiation, AutoFractal, DACAR, FDF, JITS, OpenMOSAIQUES, STAN un systèmes multi-agents distribués et UbiTalk. Ces différentes plates-formes sont décrites dans le chapitre 3.
- Finalement, l’axe 4 a été un moteur d’idées permettant de prendre conscience des problèmes liés à l’ubiquité numérique. Il a permis aux deux axes d’avoir des cas d’utilisation pour définir les problèmes de l’ubiquitaire puis ensuite de tester les solutions proposées. Nous avons travaillé sur des études de cas en lien avec le transport via l’INRETS et sur des applications E-Services. Pour chacune de ces applications, les travaux réalisés sont allés de la définition des besoins applicatifs, jusqu’à la réalisation des démonstrateurs. Elles ont permis de déterminer des besoins concernant la découverte de l’environnement et le déploiement à la volée des composants logiciels. Ces applications nous ont également permis de pointer du doigt certains aspects de l’ubiquité numérique, comme par exemple les problèmes liés à la vie privée.

1.2.4 Programme de travaux réalisés

Les travaux se sont déroulés sur les deux années de projet de décembre 2004 à juin 2007. Les trois axes ont travaillé en parallèle. L’axe 4 a fourni régulièrement des exemples aux axes 1 et 2, leur permettant dans un premier temps de réfléchir à la problématique, puis ensuite de valider les solutions proposées.

1.2.5 Difficultés rencontrées du point de vue scientifique

Les principales difficultés ont été le manque de maturité du domaine. Il a en effet été difficile de s'accorder sur le vocabulaire et les définitions de concepts manipulés tels que ubiquitaire, contexte ou encore adaptabilité. Les définitions existantes dans la littérature sont souvent encore sujettes à discussion.

Nous avons également rencontré des difficultés de dialogues autour des plates-formes, ceci est essentiellement dû à la diversité des compétences des équipes. Il nous est rapidement apparu qu'il ne serait pas envisageable de travailler sur une plate-forme unique. Nous avons alors opté pour que chaque équipe conçoive et développe sa propre plate-forme selon les environnements de développement qu'elle maîtrisait.

Finalement, comme souligné plus haut, l'axe 3 a dû rapidement travailler en symbiose avec les axes 1 et 2. La difficulté rencontrée ici est due au fait que pour valider les propriétés associées aux modèles ou aux plates-formes, il était nécessaire de participer à leur définition, afin de mieux comprendre les concepts retenus.

1.3 Bilan concernant la mise en œuvre du programme

1.3.1 Stratégies et méthodes mises en œuvre

La stratégie suivie pendant le projet a été de travailler par axe. Les axes 1 et 2 se sont réunis régulièrement (environ une fois par mois pour l'axe 1 et une fois tous les deux mois pour l'axe 2). L'axe 1 a dans un premier temps défini un vocabulaire commun et a travaillé à la définition de la méthodologie de conception. Ces réunions ont abouti à une publication commune entre les équipes ayant participé à cet axe [CCG⁺06]. Dans l'axe 2, les différentes équipes ont, dans un premier temps, exposé chacune leur vision de la plate-forme et leurs objectifs, et, finalement, elles ont réalisé chacune leur plate-forme et/ou les services associés et validés par les démonstrateurs de l'axe 4. Les équipes de l'axe 4 (INRETS-LEOST et LIFL-NOCE) ont travaillé séparément sur les spécifications des démonstrateurs et participé ensuite aux réunions des deux axes 1 et 2 pour apporter des exemples.

Nous avons également partagé des informations via une liste de diffusion et un site web collaboratif. Trois réunions plénières ont eu lieu, une en début de projet, une en milieu de projet et une en fin de projet. Un livrable commun a été écrit sur l'état de l'art du domaine en début de projet [Mos06].

1.3.2 Calendrier de travail respecté

Le projet visait à un développement de la recherche sur les modèles et infrastructures pour les applications ubiquitaires pour une période de trois ans, ramenée finalement à deux ans.

Quatre axes devaient se développer concurremment. Les partenaires devaient s'organiser comme suit :

- L'axe 1 sur l'adaptabilité dans un support méthodologique devait impliquer conjointement les équipes LIFL-GOAL, LAMIH-ROI et EMD et être porté par l'équipe LIFL-GOAL.
- L'axe 2 sur l'infrastructure logicielle avec des propriétés d'adaptabilité dynamique dans un environnement ubiquitaire était porté par l'équipe LIFL-RD2P et impliquait toutes les équipes du projet sauf LIFL-STC.

- L’axe 3 sur la validation de l’adaptabilité dans un environnement ubiquitaire devait être porté par l’équipe LIFL-STC et impliquer les équipes LIFL-GOAL, LIFL-STC et LIFL-LAMIH.
- L’axe 4 qui correspondait à la démonstration de la faisabilité devait être porté par l’équipe LIFL-NOCE et impliquait les équipes LIFL-NOCE, LAMIH-ROI et INRETS-LEOST.

Enfin, les équipes se sont réparties de la façon suivante :

- L’axe 1 sur l’adaptabilité dans un support méthodologique a impliqué conjointement les équipes INRETS-LEOST, LIFL-GOAL, LAMIH-ROI, LIFL-SMAC, LIFL-STC et EMD et a été porté par l’équipe LIFL-GOAL.
- L’axe 2 sur l’infrastructure logicielle avec des propriétés d’adaptabilité dynamique dans un environnement ubiquitaire a été porté par l’équipe RD2P et a impliqué toutes les équipes du projet.
- L’axe 4 qui correspondait à la démonstration de la faisabilité a été porté par l’équipe LIFL-NOCE et a impliqué les équipes LIFL-NOCE et INRETS-LEOST.

Le calendrier a été respecté. Fin juin 2007, les démonstrateurs sont en phase de terminaison.

1.3.3 Difficultés rencontrées dans la mise en œuvre du programme

Sur le plan scientifique, comme mentionné plus haut, nous avons revu rapidement le découpage en axes du projet. Nous avons réparti les chercheurs de l’axe 3 vers les axes 1 et 2.

Sur le plan organisationnel, les dates de départ et de fin de projet, du fait du financement multiple, sont restées assez floues (entre décembre 2004 et juin 2007). Cela a perturbé quelque peu le lancement et l’arrêt du projet, ainsi que les dates d’éligibilité des dépenses.

1.4 Bilan concernant les moyens affectés au programme

1.4.1 Moyens humains mobilisés

Une soixantaine de chercheurs ont été mobilisés sur ce projet à des degrés divers. 35 permanents (Professeur, Maître de conférences ou chercheur) ont travaillé sur l’un des axes. Dix-huit doctorants ont participé à un moment ou à un autre à l’élaboration des connaissances. Des ingénieurs, des post-docs et des stagiaires viennent compléter le potentiel humain mobilisé sur le projet.

Pour chaque équipe, nous avons noté dans les chapitres 5 à 12, le détail du personnel affecté, leur statut et le pourcentage de leur temps consacré au programme.

Dix huit thèses ont été soutenues ou sont en cours. Quatre habilitations à diriger des recherches (HDR) ont été soutenues par des maîtres de conférences à Valenciennes et Lille et trois d’entre eux ont obtenu un poste de Professeur pendant cette période.

Nous regrettons cependant qu’aucun financement de thèse directement dédié à ce projet n’ait été possible. Toutes les thèses ont été financées soit par des bourses régionales ou nationales, ou encore sur d’autres projets sur des sujets qui étaient pour partie dans le projet MOSAIQUES et pour autre partie dans l’intérêt de l’équipe.

Equipes	Pr	MCf	CR	Doct	Ing	Post-doc	Stag.	Tot.
EMD		3		3	1		5	12
INRETS-LEOST			1	1			3	5
LIFL-GOAL	3	5	3	6	3	2	2	24
LIFL-NOCE	1	3		1			4	9
LIFL-RD2P	1	1		1	1		2	6
LIFL-SMAC	1	4				1	4	10
LIFL-STC	1	2		3				6
LAMIH-ROI	1	5		3			2	11
Total	8	23	4	18	5	3	22	83

TAB. 1.1 – Nombre de personnes ayant participé au projet MOSAIQUES

1.4.2 Moyens matériels affectés

Le projet MOSAIQUES a permis d'équiper les personnels et les stagiaires travaillant sur ce thème. Il a également permis l'achat de petits terminaux permettant de tester les solutions. Une imprimante commune a été partagée entre les équipes du LIFL ainsi qu'une plate-forme de démonstration (un serveur, un ordinateur portable et 2 assistants numériques).

Les logiciels utilisés dans le développement des démonstrateurs sont des logiciels libres. Les équipes ont partagé les informations et les connaissances via une liste de diffusion de messagerie et un outil coopératif (wiki) (<http://www.lifl.fr/mosaiques>).

1.4.3 Collaborations engagées avec d'autres partenaires régionaux, nationaux et internationaux

Le projet MOSAIQUES a permis aux équipes de tisser des liens entre elles, mais également au niveau national ou international.

- Dans le cadre des pôles de compétitivité de la région (I-Trans et PICOM), les équipes ont eu l'occasion de monter des projets avec les partenaires régionaux, et plus particulièrement avec les industriels tels que Auchan, Les 3 Suisses ou NorSys.
- Plusieurs projets ANR ont été acceptés (voir en fin de chaque chapitre 5 à 12), d'autres sont en cours d'évaluation.
- Une équipe participe à un projet européen (ITEA S4ALL) et plusieurs équipes ont eu l'occasion de participer à des rencontres internationales autour de l'informatique ubiquitaire.
- Les équipes ont également reçu des chercheurs invités sur ces thématiques.

Chapitre 2

Bilan de l’Axe 1 : L’Adaptabilité dans un Support Méthodologique

2.1 Introduction

Les objectifs de l’axe 1 étaient d’étudier la modélisation et le développement dirigé par les modèles des applications ubiquitaires dans le cadre du projet MOSAIQUES. Notre motivation était de faire des propositions quant à la prise en compte dès les phase amont d’un développement des spécificités de ces applications, en particulier l’adaptation d’une application à son contexte d’exécution logiciel et matériel.

Nos propositions concernent trois aspects de la conception des applications ubiquitaires : des moyens de modélisation (méta-modèles), une démarche de conception (processus) et des moyens d’exploitation (transformation de modèles).

2.2 Modèles d’une application MOSAIQUES

Dans cette section, nous présentons les différents modèles définis par l’axe 1 : méta-modèle de contexte, méta-modèle structurel et modèle d’exécution d’une application ubiquitaire Mosaiques. Ceux-ci sont illustrés sur le scénario bus de l’INRETS-LEOST défini l’axe 4.

2.2.1 Le contexte dans MOSAIQUES

Le contexte, notion clé de l’informatique ubiquitaire [CCDG05], est central pour une application MOSAIQUES. Nous le définissons comme étant l’ensemble des caractéristiques de l’environnement d’exécution, de l’application et de l’utilisateur. Ce contexte est réifié pour être exploitable dans le modèle de la dynamique et donc manipulable dans les plates-formes. La figure 2.1 présente le méta-modèle que nous avons retenu. Le contexte est actuellement structuré en trois parties (*ContextPart*) : *EnvironmentCtxt*, *ApplicationCtxt*, *UserPreferences*. Chaque partie contient un certain nombre d’éléments de contexte (*ContextElement*) organisés hiérarchiquement. Un *ContextElement* représente une information élémentaire du contexte ou un ensemble d’informations hiérarchiques (*ContextCategory*).

L’*EnvironmentCtxt* regroupe les propriétés relatives aux capacités de l’environnement d’exécution (capacité réseau, type d’affichage, mémoire, batterie, services techniques dis-

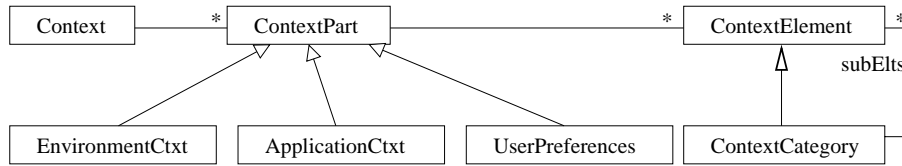


FIG. 2.1 – méta-modèle du *contexte*

ponibles, etc). L'*ApplicationCtxt* regroupe les propriétés relatives aux besoins des services (services applicatifs). Les préférences utilisateurs (*UserPreferences*) regroupent les souhaits de l'utilisateur (sortie vocale, graphique, toujours utiliser le réseau le plus rapide, économiser la batterie, abonnements aux services, etc). Ces préférences utilisateurs agissent comme des filtres sur les capacités de l'environnement (souhait de prise en compte du réseau) ou les besoins de l'application (souhait d'utilisation du service *plan du réseau de bus*).

2.2.2 La structure d'une application

Dans MOSAIQUES, une *application* ubiquitaire est composée de plusieurs *parties d'application* (*ApplicationPart*) qui représentent des unités de déploiement. Elles peuvent être mises en oeuvre de plusieurs manières pour prendre en compte différentes technologies et différents types d'environnement d'exécution. Une partie d'application correspond à un ensemble de *services* fournis partageant un même contexte. Enfin, un service est défini par une *interface* qui spécifie les *opérations* disponibles.

Les services sont sujets à des contraintes évaluées par rapport aux éléments de contexte (voir section 2.2.1). Des *règles* sont associées aux services pour formuler ces contraintes de disponibilité en fonction du contexte. La figure 2.2 présente le méta-modèle structurel des applications MOSAIQUES.

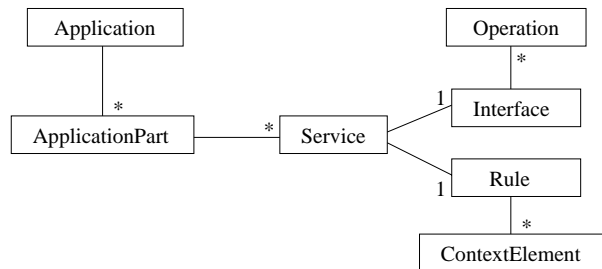


FIG. 2.2 – Structure d'une application MOSAIQUES

Les éléments de contexte servent à la définition et à l'évaluation des règles associées aux services pour exprimer leurs contraintes de disponibilité. Ces *règles* offrent une formalisation des contraintes d'utilisation. C'est une expression logique basée sur un certain nombre d'éléments du contexte pour lesquels elle détermine des intervalles de valeurs acceptables.

Dans le cadre de l'application *bus*, les différents éléments que nous venons de présenter se déclinent comme suit (voir Figure 2.3) :

- l'application est formée d'une partie résidant sur le terminal de l'utilisateur (*Terminal :ApplicationPart*) et d'une partie localisée au niveau du bus (*Bus :ApplicationPart*).

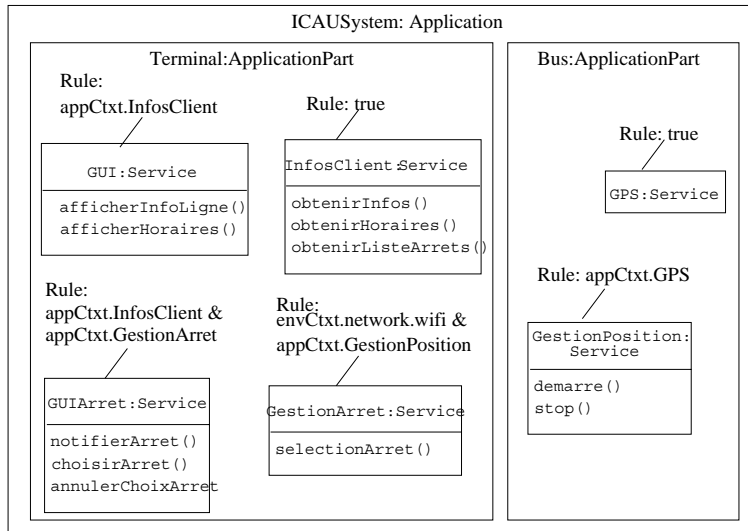


FIG. 2.3 – Structure de l’application bus

- la partie *bus* contient le service qui se charge de propager l’arrêt courant à partir des évènements reçus par le service de géo-localisation.
- la partie *terminal* de l’application comporte deux services : le premier offre la consultation du réseau de bus (les lignes, les arrêts et les horaires), le second traite les informations en provenance du serveur de notification d’arrêts (notification à l’utilisateur uniquement de l’arrêt souhaité). La disponibilité du dernier service est exprimée par une règle sur la présence du réseau dans le bus. Cette partie d’application comprend également deux services gérant l’interface utilisateur (consultation du réseau de bus, sélection et notification de l’arrêt). Cet exemple montre la possibilité de structurer les services en *services IHM* et *services fonctionnels*, mais cette structuration n’est pas imposée par le méta-modèle.

Ce découpage des applications MOSAIQUES a pour bénéfice de permettre la prise en compte et la gestion de l’adaptation non pas au niveau de l’application globale mais au niveau de ses parties. Le fonctionnement global de l’application est subordonné au fonctionnement individuel des parties ainsi qu’à leur collaboration.

2.2.3 Modèle dynamique de l’adaptation

La figure 2.4 présente la dynamique de l’adaptation d’une partie d’application MOSAIQUES, c’est-à-dire les différents états qu’elle peut prendre en fonction de l’évolution de la disponibilité des services la composant. Cette évolution est sujette aux modifications du contexte quelles qu’elles soient.

L’état *Initialisation* correspond au démarrage d’une partie d’application. Cette phase est immédiatement suivie d’une évaluation de la compatibilité des différentes contraintes de cette partie (environnement, application, utilisateur). Cette évaluation correspond à l’état *Reconfiguration*.

S’il y a un ensemble (même minimal) de services qui peut être instancié (configuration valide), alors l’exécution de la partie d’application est possible (phase classique d’initialisation

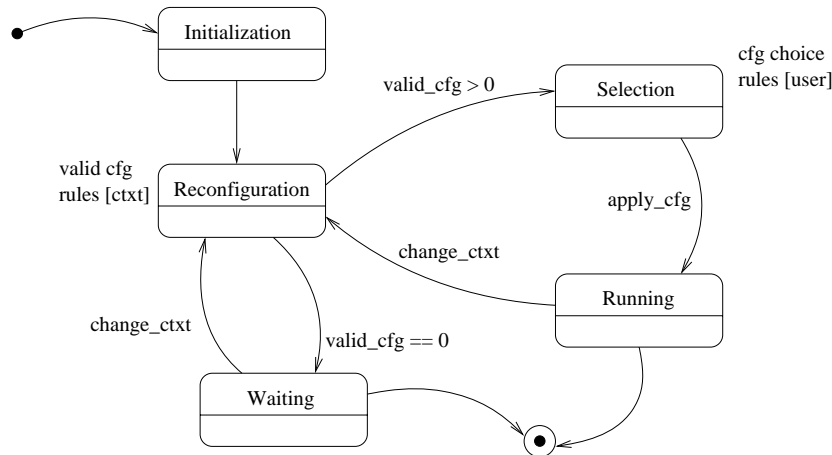


FIG. 2.4 – Dynamique de l’adaptation

d’une application). S’il existe plusieurs ensembles de services instanciables, une phase de sélection (automatique ou manuelle) devra se faire. Dans le cas où aucune configuration valide n’existe, la partie d’application considérée passe en *attente*.

Cette phase (exécution ou attente) persiste jusqu’à ce qu’une modification du contexte déclenche une nouvelle évaluation de la configuration (phase de reconfiguration). Dans le cas d’une partie d’application en attente, on va rechercher une configuration valide. Dans le cas d’une partie d’application en cours d’exécution, on va vérifier que l’exécution peut se poursuivre (éventuellement avec amélioration ou dégradation de l’exécution).

Enfin, l’utilisateur peut arrêter sa partie d’application qu’elle soit en phase d’exécution ou d’attente.

Dans le cas de l’application *bus*, le scénario suivant illustre le modèle de la dynamique de la partie d’application du terminal client :

1. l’usager souhaite utiliser le service (initialisation de la partie d’application),
2. l’usager monte dans le bus (reconfiguration suite à un changement de contexte, apparition du service de notification du bus, puis passage en phase d’exécution après sélection par l’usager de la configuration souhaitée),
3. l’usager descend du bus pour prendre une correspondance (reconfiguration suite à un changement de contexte, perte du service de notification) et il consulte l’horaire de sa correspondance (qui est lui toujours disponible),
4. l’usager monte dans le second bus (reconfiguration comme dans l’étape 2),
5. l’usager descend du second bus et met fin à la partie d’application de son terminal.

2.3 Processus de développement

Pour le développement des applications MOSAIQUES, nous proposons un processus de développement dirigé par les modèles composé de trois étapes (ce qui suit le découpage modèle abstrait / modèle technologique / implémentation de l’approche MDA –*Model Driven Architecture*) :

1. Modélisation itérative de l'application (voir section 2.3.1).
2. Projections du modèle d'une part vers une plate-forme technologique pour la production automatique d'une partie du code de celle-ci et d'autre part vers le modèle de déploiement (voir section 2.3.2).
3. Mise en œuvre des fonctionnalités de l'application dans le cadre de la technologie cible et raffinement du modèle de déploiement (ce dernier point ne sera pas détaillé ici car il sort du périmètre de l'axe 1).

2.3.1 Micro-processus de modélisation

La figure 2.5 présente un micro-processus de modélisation [MN06] pour les applications MOSAIQUES à utiliser conjointement avec le méta-modèle structurel (voir section 2.2.2). Ce micro-processus spécifie que l'activité de modélisation peut être décomposée en quatre étapes : l'identification des parties d'une application, la spécification du mode d'interaction pour avec un service (par exemple synchrone ou asynchrone), la spécification de l'architecture de l'application (regroupement logique des différentes *ApplicationPart*), et la spécification des contraintes d'usage des différentes *ApplicationPart*.

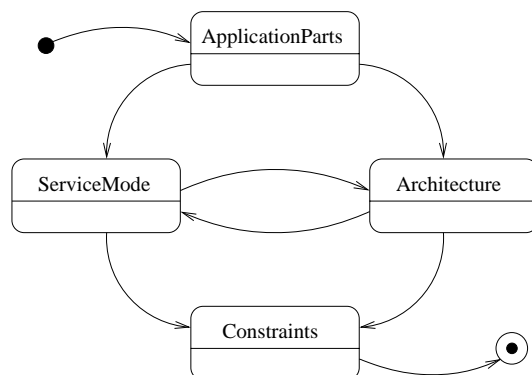


FIG. 2.5 – Processus de modélisation.

Ce processus de conception est itératif par définition. Il est impossible de modéliser intégralement une application en une seule itération, et donc le modèle d'une application sera construit de manière incrémentale d'une part en ajoutant progressivement des détails, d'où la présence de plusieurs étapes dans le micro-processus de modélisation, et d'autre part en modélisant l'application fonction par fonction.

Pour cette dernière séparation, nous nous reposons sur la décomposition d'un système en vues fonctionnelles[MCC⁺07] : chaque vue est modélisé de manière indépendante (ce qui réduit la complexité du modèle aux éléments relatifs à la vue). Le micro-processus de la figure 2.5 est utilisé pour la modélisation de chacune de ces vues. Ensuite, le modèle complet de l'application est produit par fusion des différentes vues (les éléments de modèle sont identifiés sur leur nom pour réaliser cette fusion avec des opérateurs comme le *merge* d'UML2).

Ce micro-processus a été outillé sous la forme d'un profil UML2 pour les applications ubiquitaires MOSAIQUES réalisé d'une part dans l'outil ModX et d'autre part dans l'outil de modélisation Objecteering.

2.3.2 Projections des modèles

Les projections de modèles d'applications représentent le lien entre l'axe 1 et l'axe 2 du projet MOSAIQUES. Ces projections sont exprimées comme des transformations de modèles (au sens de l'OMG) entre deux méta-modèles. La réalisation de ce type de transformations peut se faire aussi bien en utilisant un langage dédié comme MOF 2.0 QVT (*Query / View / Transformation*) basé sur l'utilisation de règles de traduction, qu'en utilisant un langage généraliste de manipulation de modèles comme EMFScript[TV06] voire même Java.

Projection vers un modèle technologique La seconde projection correspond à la projection du modèle de l'application vers une technologie particulière de mise en œuvre. Pour cela nous nous reposons sur l'utilisation du profil UML2 pour la partie modélisation puis, par exemple, sur la projection standard de l'OMG entre UML et le modèle de composants CORBA[MM01] ou encore sur la projection d'un modèle UML 2.0 vers le modèle de composants Fractal[MP06].

A titre d'exemple, un élément de modèle de type *ApplicationPart* dans une application MOSAIQUES est traduit en un assemblage de composants dans le cadre du modèle de composants CORBA et en un composite dans le cas du modèle de composants Fractal. Un élément de type service sera quant à lui traduit en un composant dans les deux cas.

Projection vers un modèle de déploiement Le projet DACAR a pour objectif d'abstraire et de généraliser le déploiement des application en le dirigeant par les modèles. Un des bénéfices de cette approche est de pouvoir plus simplement réaliser le déploiement d'une application répartie dans un contexte d'exécution fortement hétérogène. Le service FDF utilisé dans l'axe 2 fait parti des différentes plates-formes de déploiement supporté par ce projet. Nous avons défini une projection entre les méta-modèles d'application MOSAIQUES et le méta-modèle de déploiement de DACAR.

A titre d'exemple, un élément de modèle de type *ApplicationsPart* est projeté en un élément de modèle *System* dans le méta-modèle de déploiement de DACAR et un élément de type *Service* en un élément de type *Software*. Pour ce qui est de la dynamique d'une application, un élément de type *Rule* est projeté en un élément de type *Condition* dans le méta-modèle *autonomie* de DACAR, et tout changement de contexte à l'exécution sera signalé par un événement au sens de DACAR afin de piloter l'adaptation de l'application par un re-déploiement éventuel de certaines entités logicielles de l'application.

2.4 Nouvelles applications vs applications existantes

L'approche dirigée par les modèles que nous proposons se prête bien à la réalisation de nouvelles applications MOSAIQUES. Toutefois, faute d'avoir étudié des techniques de rétro-ingénierie, cette approche ne se prête pas à l'évolution d'une application existante vers une application MOSAIQUES. D'autre part, l'approche que nous proposons permet de produire une partie seulement des applications MOSAIQUES. Il reste donc au développeur à mettre en œuvre les fonctionnalités de son application, ce qui n'est pas une tâche facile dans le contexte ubiquitaire (même si nous arrivons à générer automatiquement une partie non négligeable du code et que nous utilisons les plates-formes d'exécution de l'axe 2). Pour répondre à ces deux besoins, nous avons aussi étudié l'utilisation de la transformation de programmes dans le contexte des environnements ubiquitaires.

2.4.1 La transformation de programmes pour environnements ubiquitaires

Programmer des applications pour des environnements ubiquitaires est une tâche difficile. Plusieurs limitations inhérentes au domaine, telles que les connexions volatiles, ou encore la faible puissance des batteries, amènent le développeur à revoir ses méthodes de développement. Nous avons dans ce travail voulu partir de programmes existants en environnement connecté et les transformer en programmes fonctionnant sur des environnements ubiquitaires. Nous avons travaillé sur deux particularités de ces programmes que sont les connexions volatiles et la conscience du contexte.

Nous avons basé notre approche sur les techniques de génération et de transformation de code sur lesquelles nous travaillons dans l'équipe GOAL, et plus particulièrement sur notre outil Spoon (<http://spoon.gforge.inria.fr>). Nous avons proposé des annotations et des transformations associées permettant la transformation d'une application fonctionnant en mode connecté en une application fonctionnant dans un environnement ubiquitaire. Un ensemble d'annotations a été proposé et un outil de transformation de code appelé Graffiti a été développé. Ce travail se place dans le cadre de la thèse Carlos Noguera et a donné lieu à deux publications [FN07, PDNP07].

Chapitre 3

Bilan de l’Axe 2 : Infrastructure Logicielle, Support de l’Adaptabilité

3.1 Introduction

Les objectifs de l’axe 2 étaient d’étudier la modélisation et le développement d’une infrastructure logicielle fournissant les services essentiels à l’adaptabilité des applications déployées sur les supports de l’ubiquité numérique. L’enjeu de cette étude était, en premier lieu, (i) d’identifier les spécificités des matériels informatiques impliqués dans l’émergence de l’ubiquité numérique, mais aussi, (ii) de caractériser les mécanismes fondamentaux capables de supporter les fonctionnalités de base propres aux applications ubiquitaires, et enfin, (iii) de caractériser les fondements théoriques permettant aux l’environnements ubiquitaires de garantir la fiabilité des applications hébergées. Cette étude de base devait servir de socle pour la conception d’un environnement d’exécution commun. Cependant, dès les premières réunions de l’axe, les études préliminaires ont montré qu’il serait vain d’espérer regrouper, autour un seul environnement d’exécution, l’ensemble des services pertinents et des outils nécessaires à l’intégration des caractères propre du domaine. Aussi, renonçant à cette vue étreiquée d’une informatique monochromatique les différents partenaires impliqués dans l’axe 2 ont convenu d’une reformulation de l’ambition de l’axe afin que la pluralité des infrastructures finalement proposées reflète au mieux la singularité de chacune des multiples facettes de l’informatique ubiquitaire.

Nous présentons donc, dans ce rapport, les différentes infrastructures logicielles, résultant de notre étude. Ces infrastructures répondent, selon les cas, (i) à des contraintes d’adaptabilité issues de spécificités des matériels visés, (ii) aux besoins de flexibilité des applications ubiquitaires (notamment issues de l’axe 1), ou encore (iii) aux besoin de fiabilisation de ces logiciels fortement altérés par le milieu auquel ils s’appliquent.

Le tableau 1 reprend les différentes familles de solutions proposées, en les positionnant par rapports aux trois aspects fondamentaux des infrastructures logicielles, support de l’adaptabilité au milieu ubiquitaire. Le reste du chapitre décrit chacune des infrastructures en les inscrivant dans le cadre dans lequel elles prévalent.

Infrastructures	Equipes	Adaptation à la cible ubiquitaire	Adaptation aux besoins applicatifs	Ubilisation dans l'ubiquité
Architecture d'intermédiation	LIFL-NOCE			x
AutoFractal	Ecole des Mines-Douai			x
DACAR	LIFL-GOAL			x
FDF	LIFL-GOAL	x		
JITS	LIFL-RD2P			x
OpenCCM-MOSAIQUES	LIFL-GOAL	x		
STAN	LIFL-STC			x
Système multi-agents distribués	LIFL-SMAC			x
UbiTalk	Ecole des Mines-Douai			x

TAB. 3.1 – Synthèse des infrastructures

3.2 Architecture d'Intermédiation

Dans le cadre de l'axe 2, l'équipe NOCE s'est attachée à développer une infrastructure d'intermédiation permettant la gestion de services ubiquitaires. Ce travail s'articule autour de l'orchestration des services en fonction du contexte et trouve son pendant dans le projet MIAOU en terme d'interaction homme-machine pour l'utilisation des services à travers différents canaux de communication. La Figure 1 présente une vision simplifiée de l'architecture développée avec à droite les différents canaux permettant d'accéder aux services dans différentes situations (traditionnelle, en mobilité) et à gauche les services qui fournissent les fonctionnalités applicatives.

L'architecture d'intermédiation est basée sur un système multi-agents JADE. Les différents agents qui composent le système gèrent :

- Les propriétés des canaux de communication ;
- Le contexte d'utilisation qui peut couvrir des aspects tels que la localisation des utilisateurs ou leurs profils ;
- Les règles organisationnelles qui doivent être appliqués à l'utilisateur ;
- Les propriétés et fonctions des services proposés.

En fonction du canal utilisé, du contexte et des règles organisationnelles, le système pourra proposer des services pertinents aussi bien en terme de support à l'activité en cours qu'en terme de facilité d'usage en fonction des propriétés des canaux utilisés.

3.3 Auto-Fractal

Auto-Fractal est une plate-forme d'adaptation dynamique d'applications à base de composants Fractal, réalisée par Guillaume Grondin dans le cadre de sa thèse au sein de l'école des mines de Douai.

Cette plate-forme est une spécialisation pour le modèle de composants Fractal de MaDcAr :

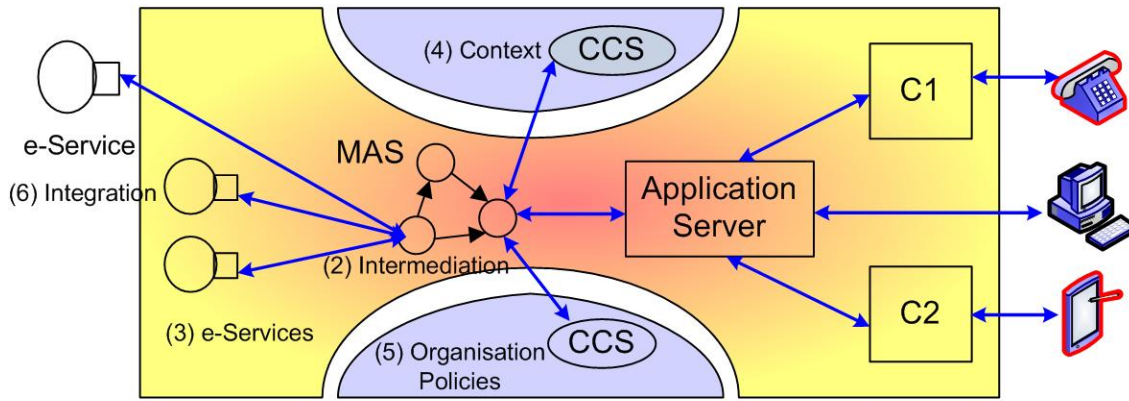


FIG. 3.1 – Architecture d'intermédiation

"Model for automatic and Dynamic component Assembly reconfiguration". Concrètement, AutoFractal permet de concevoir des applications dynamiquement adaptables. L'adaptation "à chaud" consiste à faire passer l'application d'un assemblage de composants à un autre. Ce passage est une reconfiguration qui peut comporter des ajouts/suppression/remplacement de composants, des révisions des connexions entre composants et des changements des valeurs des attributs des composants. Pour ce faire, le concepteur de l'application doit fournir la description de l'application qui est une spécification de l'ensemble des assemblages valides. Cette spécification est abstraite dans la mesure où elle ne fait pas directement référence aux composants à utiliser. Le choix de l'assemblage à réaliser et des composants à assembler se fait selon une politique d'adaptation, également spécifiée par le concepteur de l'application. La politique d'adaptation est un ensemble de règles qui, selon le contexte d'exécution (ex : état du réseau), génère un ensemble de contraintes. La résolution de ce problème de satisfaction de contraintes débouche sur l'assemblage le plus pertinent au contexte courant.

3.4 DACAR (Distributed Autonomous Component-based Applications Reconfiguration)

Dans des environnements ouverts comme l'informatique ubiquitaire, le domaine de déploiement, c-à-d l'ensemble des machines impliquées dans l'application, ne peut à aucun moment être connu statiquement. Ainsi il est impossible de prendre en compte l'apparition ou la disparition dynamique de machines sur le domaine. Or, dans le cadre d'applications ubiquitaires, l'apparition d'une nouvelle machine correspond à l'entrée d'un terminal sur le réseau, et donc à l'entrée d'un nouvel acteur dans l'application, pour lequel il faut déployer des composants logiciels métiers. DACAR est un paradigme de conception ainsi qu'une plate-forme d'exécution d'architectures logicielles autonomes en environnements ouverts s'appuyant sur les concepts de l'informatique autonome définis par IBM (Autonomic Computing). DACAR permet de décrire les composants logiciels impliqués dans l'application ainsi qu'une configuration de départ du plan de déploiement. Ces descriptions sont accompagnées de règles

basées sur le paradigme Evènement-Condition-Action, afin d'exprimer les règles d'autonomie de l'architecture. Les évènements peuvent par exemple notifier l'apparition d'un terminal, l'action correspondante sera alors de déployer les composants métiers adéquats sur ce terminal. Un premier prototype de DACAR permet de déployer des assemblages de composants CCM autonomes, en s'appuyant sur la plate-forme OpenCCM.

3.5 FDF : Un framework générique pour automatiser le déploiement logiciel

Fractal Deployment Framework (FDF) est un outil générique permettant d'automatiser l'exécution des tâches hétérogènes qui composent le processus de déploiement logiciel (<http://gforge.inria.fr/projects/fdf/>). Le déploiement peut être défini comme un ensemble de tâches à réaliser telles l'installation (désinstallation) et l'activation (désactivation) d'instances logicielles sur des noeuds distants. Ces tâches de déploiement doivent être orchestrées dans un ordre bien défini. L'ensemble de ces tâches, comme l'accès aux différents noeuds distants, le téléchargement, l'installation de logiciels ou plates-formes intergicielles, de bibliothèques, de binaires ou autres programmes exécutables sur ces noeuds, la configuration ou le démarrage des processus distants, s'effectue actuellement "manuellement" ou au mieux de manière "ad hoc" via par exemple l'utilisation de scripts difficilement réutilisables dans un autre contexte. Ainsi, le déploiement sur des systèmes complexes et dynamiques, tels les environnements ubiquitaires qui sont composés de terminaux et réseaux potentiellement hétérogènes ou les grilles informatiques pouvant comporter plusieurs milliers de noeuds, est une activité complexe pour les administrateurs. FDF permet d'une part, de décrire dans un langage de haut niveau les configurations à déployer (i.e. les noeuds du système cible et les logiciels à y déployer) au lieu de programmer ou scripter la manière dont le déploiement sera réalisé, d'autre part de s'abstraire de tout problèmes de dépendances ou synchronisation entre les tâches ou logiciels. FDF fournit un ensemble de composants génériques réifiant les mécanismes et outils intervenants dans tout processus de déploiement (comme les protocoles d'accès distants SSH/Telnet..., les protocoles de transfert de fichiers HTTP/FTP/SCP..., shells, ports, etc.) et les personnalités représentant les logiciels à déployer. Il devient de ce fait possible d'automatiser des déploiements complexes. FDF, implanté en Java via le modèle de composants Fractal, est indépendant des technologies à déployer mais également de la granularité des logiciels (intergiciels, services, serveurs d'applications, composants logiciels, objets, bibliothèques, daemons, processus, etc.). Nous avons expérimenté cet outil pour le déploiement automatique de la plate-forme OpenCCM-Mosaïques et des applications ubiquitaires sur les serveurs des fournisseurs de services et un ensemble de PDA utilisateurs.

3.6 JITS

Dans le cadre de l'axe 2, l'équipe RD2P a promu une infrastructure de déploiement de logicielle spécialisée nommée JITS (<http://jits.gforge.inria.fr>). La conception de ce support d'exécution ubiquitaire a été guidée par l'étude des différentes infrastructures offrant des mécanismes de déploiement d'applications ubiquitaires. Parmi les infrastructures les plus pertinentes nous avons identifié, le modèle des Bundles (OSGi), le modèle des Servlets (J2EE), et le modèle des Cardlets (JavaCard). L'ensemble de ces modèles repose sur les mécanismes de

chargement et de liaison dynamique fournis par l'environnement d'exécution de Java. Cependant ce sont ces mêmes mécanismes qui constituent l'essentiel de la complexité des supports d'exécutions Java, et qui rendent impossible le déploiement des applications au sein des infrastructures embarquées minimalistes, souvent mises en jeu au sein des scénarios ubiquitaires.

Nous avons proposé un support d'exécution particulier, JITS (pour Java In The Small) qui nous permet de réaliser un « pré-déploiement » des applications quel que soit la logique de déploiement (Bundle, Servlet, Cardlet, ?) sur laquelle elles reposent. Sur cette base, nous avons montrés qu'il était possible d'adapter les applications aux matériels ubiquitaires les plus contraints tel que les cartes à puces ou les étiquettes électroniques. En effet, Le mécanisme de pré-déploiement que nous avons proposé, permet de réaliser les opérations liées à la logique de déploiement des applications ubiquitaire en amont de l'exécution de ses mêmes applications. Selon le modèle de pré-déploiement que nous préconisons, les applications ne sont transférées sur la cible ou elles seront exécutées qu'une fois que leur déploiement est achevé. Dans ce contexte, il devient possible de spécialiser l'application après sont déploiement et avant sont transfert vers la cible réelle.

La spécialisation « à chaud » des infrastructures logicielles support de l'ubiquitaire a largement était discutée dans le cadre de Mosaïque. Nos résultats reposent sur des outils d'analyses puissants, appliqués à l'image figée du système ubiquitaire, après déploiement. A l'aide de ses outils, il est tout d'abord seulement possible de spécialiser les applications après qu'elles aient été déployées pour un contexte d'utilisation établi. Il est aussi possible d'utiliser le produit de ses analyses pour spécialiser le support d'exécution ubiquitaire lui-même, et ainsi d'appliquer les bénéfices de l'adaptabilité au support qui les mets en oeuvre. Dans le cadre de Mosaïque, nous avons eu l'opportunité de réaliser un outil de spécialisation de la machine virtuelle Java. Sur ces différents points, nous avons obtenus des résultats probants. Ils démontrent que la spécialisation « à chaud » issue du modèle de pré-déploiement que nous avons proposée est particulièrement profitable lorsqu'elle est appliquée à la spécialisation des machines virtuelles.

3.7 OpenCCM-MOSAIQUES : Plate-forme intergicielle

La multiplication des terminaux mobiles (ordinateurs portables, téléphones mobiles, PDA...) ainsi que la généralisation des réseaux sans-fil impliquent des changements dans la conception et l'exécution des applications logicielles. La problématique est maintenant clairement identifiée sous le terme "informatique ubiquitaire". Plusieurs problèmes nécessitent d'être pris en compte comme par exemple l'hétérogénéité matérielle et logicielle des équipements, la limitation des ressources (terminaux mobiles et réseaux sans-fil), la découverte des services ou encore le déploiement des applications sur les terminaux. Afin d'adresser ces différents challenges, nous avons proposé un modèle d'architecture d'une infrastructure distribuée à base de composants logiciels, nommée OpenCCM-MOSAIQUES. La plate-forme OpenCCM-MOSAIQUES permet de concevoir, découvrir, déployer et exécuter des services contextuels ubiquitaires, c'est-à-dire fonction du contexte d'exécution des terminaux mobiles / utilisateurs, comme par exemple leur position géographique ou les caractéristiques matérielles et logicielles des terminaux. Ces services ubiquitaires sont eux-mêmes conçus sous la forme d'assemblage de composants logiciels distribués. L'infrastructure OpenCCM-MOSAIQUES offre un certains nombre de services dédiés aux environnements ubiquitaires tels la possibilité de découvrir dynamiquement les services métiers proposés par les fournisseurs dans une zone de

couverture donnée et de les déployer (ou plus précisément la partie mobile s'exécutant sur les terminaux) automatiquement et à la demande des utilisateurs. Le choix de cette architecture, notamment en ce qui concerne le protocole de communication utilisé ("multicast") pour la découverte, a été guidé par les considérations énergétiques des terminaux (via une analyse de la consommation énergétique des terminaux en fonction du mode de communication, transmission ou réception). Une phase de négociation entre le terminal et le serveur des fournisseurs de services permet de prendre en compte les aspects d'adaptabilité des services aux terminaux (la négociation aboutit à la présentation aux utilisateurs des seuls services adaptés aux caractéristiques de leur terminaux, ou bien, si plusieurs possibilités existent, à un choix final des utilisateurs). Finalement, une fois le déploiement effectué, le support d'exécution gère la désinstallation future du service depuis le terminal et offre la possibilité de mise en cache des composants logiciels du service afin d'anticiper une prochaine réutilisation du même service. La plate-forme OpenCCM-MOSAIQUES a été implantée via le modèle de composants CORBA de l'OMG, au dessus de la plate-forme intergicielle libre OpenCCM, ainsi que l'exemple d'application ubiquitaire "trains au départ". Un prototype est disponible à l'adresse suivante : <http://cvs.forge.objectweb.org/cgi-bin/viewcvs.cgi/openccm/mosaiques/>

3.8 STAN

Dans les systèmes informatiques ubiquitaires modernes, le nombre d'applications malicieuses et non sûres est de plus en plus important. Les travaux de l'équipe STC se situent dans le domaine des petits systèmes embarqués supportant plusieurs applications. Ces applications manipulent des données secrètes et ne sont pas isolées : elles peuvent interagir dans un mode direct, ou bien partager du code commun, pour optimiser l'utilisation de la mémoire. Afin d'assurer la sécurité de ces systèmes, il faut garantir que les données secrètes ne soient pas accessibles aux autres programmes/utilisateurs qui peuvent observer les comportements ou interagir avec ces systèmes.

En pratique, les applications ubiquitaires ne sont systématiquement pas embarquées « à l'avance » : nous considérons un monde ouvert où des applications peuvent être chargées dynamiquement. Lorsque de nouvelles applications doivent être chargées, il faut assurer leur sécurité, mais aussi la sécurité des applications déjà existantes.

Dans nos travaux, nous avons essayé de fournir une solution à ces problèmes : assurer la sécurité des applications dédiées aux systèmes embarqués dans un monde ouvert. Notre solution se base sur l'analyse de flot de contrôle. L'exécution de cette analyse est très coûteuse et n'est donc pas adaptée aux petits objets portables qui ont des ressources limitées. Pour utiliser les ressources du monde extérieur, l'algorithme comporte deux étapes : une étape extérieure, durant laquelle on calcule les informations nécessaires pour vérifier la sécurité, et une seconde étape oncard, durant laquelle on vérifie uniquement les résultats obtenus lors de la première étape.

Nous avons développé l'outil STAN (STatic Alias aNalyser) qui est proposé sous la forme d'un logiciel libre à l'URL <http://www2.lifl.fr/~ghindici/STAN/>. STAN implémente l'algorithme décrit dans la première étape. STAN est un interpréteur abstrait Java qui permet d'analyser statiquement de bytecode et de vérifier le flot d'information. STAN vérifie statiquement de code déjà compile et annote les fichier exécutable de Java (.class) avec des signatures vérifiables a l'embarquement. Les résultats obtenus montrent qu'on peut utiliser les signatures dans un système embarqué.

On a effectuée deux cas d'étude. La première application testée se nomme PACAP. Il s'agit d'un porte-monnaie électronique (largement étudié dans la littérature) auquel peuvent venir s'adjoindre des loyalties. En deuxième temps, nous avons développé une application gérant, sur une carte à puce, le dossier médical d'un patient. Notre but a été d'assurer la confidentialité de dossier médical. C'est une carte à puce d'identification et d'information qui peut contenir une ou plusieurs applications qui peuvent lire et transmettre à l'extérieur les parties du dossier médical aux quelles elles sont autorisées d'accéder. On a montré que la confidentialité est respectée par rapport au flux d'information. De plus, ces applications peuvent être installées/désinstallées.

Pour tester les deux cas d'étude, nous avons annoté API de la plateforme JITS (cf. 3.4) qui se présente comme une machine virtuelle Java dédiée aux petits systèmes portables. Les résultats montrent que les annotations peuvent être embarquées sans surcoût, grâce à leur taille étant réduite.

3.9 Système multi-agent distribués

Dans le cadre du projet MOSAIQUES, l'objectif est d'étudier les impacts produits lors de la conception de logiciels devant s'exécuter dans un contexte ubiquitaire, c'est-à-dire un environnement dans lequel les applications sont physiquement réparties sur des périphériques aux capacités très variables (en terme de calcul, de mémoire, de connexion réseau ou encore d'interaction homme machine). Dans ce contexte, la conception de logiciels est profondément modifiée car il n'est plus possible de supposer qu'un seul organisme développe l'ensemble des applications proposées à l'utilisateur. D'autre part, les différences d'environnement d'exécution impliquent une prise en compte de ces derniers lors de la création et surtout lors de l'exécution de l'application. Il est alors nécessaire de représenter ces contextes d'exécution pour que les applications puissent s'adapter aux capacités disponibles à un instant donné.

L'équipe SMAC étudie depuis sa création la conception de système multi-agents distribués. Plus récemment, l'émergence de nombreuses plateformes nous a mené à étudier plus particulièrement les problèmes d'interopérabilité. Ces problèmes surgissent lorsque des systèmes conçus par différentes organisations doivent interagir. La solution habituelle consiste à définir des composants ad-hoc permettant d'établir un pont entre les deux systèmes. Cette solution n'est pas satisfaisante et totalement impraticable lorsque le nombre de systèmes augmente.

Pour illustrer ces problématiques et les solutions que nous apportons, nous avons effectué deux propositions fortes dans le projet MOSAIQUES. La première proposition concerne un scénario « Foire de Paris » permettant d'illustrer les objectifs poursuivis en terme d'applications ubiquitaires. La deuxième proposition se rapporte à l'approche en terme de méthode de conception et des éléments d'infrastructure permettant de développer des services pouvant être exploités dans un environnement ubiquitaire.

Dans le cas d'étude « Foire de Paris », un utilisateur vient à Paris pour se rendre à la Foire de Paris. Il dispose d'un périphérique (PDA, téléphone, ?) qui héberge son assistant, un agent logiciel servant d'interface avec les services « ubiquitaires ». Voici un exemple d'interaction pouvant se produire entre l'assistant de l'utilisateur et un agent médiateur pour les services de transport :

Assistant : "Quel service de transport puis-je utiliser pour me rendre à la Foire de Paris?"

AgST : Le taxi est un service de transport qui permet de se

rendre a la Foire de Paris.
Assistant : Pourquoi peut-il etre qualifie de service de transport pour la Foire de Paris ?
AgST : Il permet de se rendre a Paris
Assistant : Selon moi, ce service de transport doit me permettre de me rendre place de la porte de Versailles.
AgST : Selon moi, ce service de transport ne doit pas permettre de se rendre place de la porte de Versailles mais avec un taxi vous le pouvez.
Assistant : OK, je vais considerer les services de taxi".

Dans cet exemple, les deux agents ne partagent pas les mêmes représentations concernant les services de transport. Dans un environnement multi-fournisseurs, ce type de situation se produit souvent, il est donc nécessaire de proposer des mécanismes résolvant (dans une certaine mesure) ces incompréhensions.

Pour cela, nous proposons d'utiliser la notion d'ontologie, pour représenter les informations échangées par les agents, conjointement à un système dialogique, pour résoudre les conflits de représentation et effectuer la découverte et la sélection de services.

Une ontologie définie formellement un vocabulaire commun pour partager l'information d'un domaine entre plusieurs agents. Une ontologie repose sur des définitions interprétables par des machines des concepts d'un domaine et de leurs relations. Le formalisme que nous utilisons est une logique de description ALC. Ce langage correspond au niveau intermédiaire utilisé par le W3C dans OWL (OWL-DL). L'utilisation de cette logique apporte des automatisations sur la vérification de la cohérence de la description ainsi que des capacités de classification automatique des ressources.

L'ontologie intervient dans la caractérisation des services et des ressources présentes dans le système. Comme il paraît peut probable qu'une ontologie unique s'impose (le Graal poursuivit par le projet CYC?), il est nécessaire de traiter le problème d'interopérabilité des ontologies.

Dans notre solution nous ne créons pas de ponts entre les ontologies, mais nous utilisons des techniques d'argumentation. A l'aide d'un mécanisme dialogique permet aux agents d'aboutir à un consensus sur leurs représentations et permet la sélection de services répondant aux spécifications de l'utilisateur. Notre approche, DIALRAR (DIALRAR Is an Argumentative Labour to Reach Agreement on Representation) est un cadre formel dans lequel les agents argumentent pour s'accorder sur une représentation.

Dans ce système, un dialogue est une séquence cohérente de coups qui a pour but de faire évoluer une situation initiale pour atteindre les buts des participants. En l'occurrence, le but des dialogues consiste en la résolution d'une requête pour sélectionner un service ou découvrir les capacités d'un service. Dans la situation initiale, les deux agents impliqués ne partagent pas la même définition d'un objet, soit parce que l'un deux est ignorant, soit parce que leurs définitions ne proposent pas le même point de vue. Au terme du dialogue qui les confronte, les agents doivent atteindre un accord qui concerne la définition de cet objet ce qui permet de sélectionner ou non le service comme correspondant à la requête du client.

Bien que les étapes de découvertes et de sélection d'un service soient fondamentales, la composition d'un ensemble de services est d'autant plus importante. L'objectif ultime est d'avoir un enchaînement automatique d'un ensemble de services découverts dynamiquement pour résoudre une requête de l'utilisateur. Pour parvenir à cet objectif, il serait nécessaire de

pouvoir représenter la sémantique d'un service et de son utilisation. Cet objectif étant très ambitieux, nous avons travaillé sur deux étapes nécessaires et préalables : un mécanisme semi automatisé de composition de services et un système de négociation sur des planifications de composition de services.

Notre proposition est une composition automatisée de services par fusion d'information et chaînage de services. La phase de fusion d'information consiste à intégrer les informations publiées par les services, afin de pouvoir raisonner sur plusieurs sources d'information simultanément. La phase de chaînage de services propose un enchaînement de services en se basant sur les informations consommées et produites par les services.

Pour que la phase de fusion soit possible, nous proposons un modèle de conception d'ontologie prescrivant une structuration des ontologies selon quatre niveaux :

- Le niveau I regroupe les connaissances les plus abstraites, partagées par l'ensemble des agents du système. Dans notre exemple nous décrivons à ce niveau les concepts de Graphe, Noeud et Arc qui sont utilisables dans des domaines qui peuvent être très différents du transport.
- Le niveau II contient les connaissances du domaine, ces connaissances spécialisent des concepts de niveau I. Dans notre exemple nous décrivons à ce niveau un Réseau qui est un ensemble de Point (Noeud géographiquement situé par des coordonnées GPS) et des Etapes reliant les Points.
- Le niveau III concerne les connaissances spécifiques au service mais explicitées avec les connaissances communes. Ainsi le médiateur va pouvoir comprendre les informations (Stations, Arrêts) à l'aide du niveau II (Point) qu'il partage avec les autres agents de domaine proche.
- Le niveau IV est dédié aux connaissances propres au service, n'ayant pas à être partagées avec les autres agents. Par exemple les noms des chauffeurs de bus ou de métro.

Une fois les ontologies définies, il est possible de caractériser les services. Chaque service possède deux parties :

- La description des opérations précise pour chaque opération du service le nom de l'opération et à l'aide des concepts du niveau I, II ou III, les informations consommées (ou requises) et produites par l'opération. Pour l'exemple des services de transport, trois opérations sont définies : `simuler` qui fournit un `Trajet` construit par le service de transport à partir de son réseau pour relier les 2 Points fournis. L'opération `evaluerCoût` va construire un `BilletFactice`, c'est-à-dire un devis sur le coût pour effectuer le `Trajet` fourni. L'opération `acheter` sera effectuée par l'assistant de notre utilisateur afin d'obtenir son `Billet` de transport à partir du `BilletFactice` fourni par le médiateur.
- Les informations publiques sont les connaissances que fournit le service afin d'être utilisé par un utilisateur et de permettre des raisonnements. Dans l'exemple de transport pour la Foire de Paris, les services de Bus et de Métro fournissent un réseau d'arrêts ou stations (Points géographiquement situés) reliés par les routes pour le bus ou les voies de métro, ce qui équivalent aux informations reprises par notre utilisateur avec les annotations de localisation.

3.10 UbiquiTalk

UbiquiTalk est une plate-forme P2P orientée service, réalisée par Michaël Piel, ingénieur de recherche à l'Ecole des Mines de Douai.

La finalité première de cette plate-forme est de servir de support aux expérimentations de scénario nécessitant l'adaptation et en particulier mettre en oeuvre AutoFractal. En effet, UbiqTalk est une plate-forme pour applications pair-à-pair (Peer-to-Peer, P2P). Elle fournit un canevas (framework) qui permet de développer des services (au sens Service Oriented Architecture) déployés de manière non-anticipée. En effet, une machine équipée d'UbiqTalk peut découvrir automatiquement les autres machines dotées également de la même plate-forme, et éventuellement utiliser des services fournis par les machines ainsi découvertes. Cette utilisation implique souvent le déploiement de clients sur la machine utilisatrice, opération fournie par la plate-forme. L'hétérogénéité des machines est prise en compte à cette étape, puisque le code client transféré est choisi en fonction des propriétés de la cible. L'architecture, le processeur, la quantité de mémoire disponible sont des exemples de propriétés utilisées pour discriminer les clients. Une autre propriété importante, sur laquelle l'accent a été mis lors du développement d'UbiqTalk, est la taille de l'écran. En effet, UbiqTalk dispose de deux canevas d'IHM : un pour les machines de type PDA avec un petit écran et l'autre pour des machines de type ordinateur de bureau, où l'écran est plus grand et on peut manipuler des fenêtres

Chapitre 4

Bilan de l’Axe 4 : Démonstration de Faisabilité

4.1 Introduction

L’axe 4 du projet MOSAIQUES a en charge la démonstration de la faisabilité de l’approche par la réalisation d’applications dans des domaines particuliers comme les transports (action 4.1) et les E-services ubiquitaires (action 4.2) pour la E-Santé, la E-formation et le E-Commerce.

Bien que chronologiquement arrivant en dernier, les activités de cet axe ont débuté en même temps que les autres activités du projet.

Les travaux initiés dans l’axe 4 sur les applications tels que :

- Les cas d’utilisation des applications (“use cases”) ont permis à l’axe 1 d’appliquer leurs travaux de modélisation sur des exemples concrets.
- Les exigences requises (“user needs”) pour permettre la réalisation des applications ont servis de point de départ dans le développement d’extensions dans des plate-formes réalisées dans l’axe 2.

4.2 Action 4.1 - Applications liées aux transports

Le sous axe 4.1 du projet a en charge les démonstrations de faisabilité pour des applications liées au monde des transports.

Les travaux réalisés vont de la définition des besoins applicatifs jusqu’à la réalisation des applications. Dans le cadre du projet, deux applications ont été réalisées. Celles-ci ont permis de déterminer des besoins concernant la découverte de l’environnement et le déploiement à la volée de composants logiciels. Les applications nous ont également permis de pointer du doigt certains problèmes liés à la vie privée en ne dévoilant pas trop d’informations personnelles dans le système.

4.2.1 L’application “ trains au départ ”

L’application est issue du monde ferroviaire. L’objectif de cette application est de permettre à un voyageur se déplaçant en train de connaître automatiquement le quai de départ

de son train. Cette information lui étant automatiquement fournie sur son terminal mobile personnel (PDA, smartphone, ...) lorsqu'il entre dans la gare de départ.

Cette application est composée de plusieurs éléments logiciels qui vont s'exécuter sur des machines soit fixes (installées à demeure dans la gare) soit des machines mobiles (les terminaux des passagers).

La figure 6.2 présente le cas d'utilisation de cette application.

Les différentes fonctions de l'application peuvent être résumées comme suit :

- *Database train schedule* Ce composant contient la liste des horaires de l'ensemble des trains de l'opérateur.
- *Train Component* Ce composant est installé dans la gare sur une machine fixe. Il va extraire de la base de données de l'opérateur la liste des trains qui concerne la gare à laquelle il est associé. Il doit également répondre aux requêtes qui vont venir des terminaux mobiles (composant suivant).
- *Service mobile part GUI / Text to Speech UI* Ces composants s'exécutent sur les terminaux des utilisateurs. Certains utilisateurs préféreront une intergace graphique (GUI), d'autres utiliseront une interface à base de synthèse vocale qui leur lira la liste des trains au départ ainsi que le quai associé.

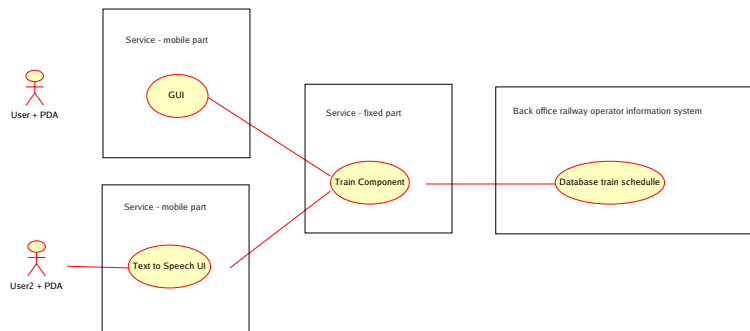


FIG. 4.1 – Diagramme de cas d'utilisation

Cette application a été réalisée à l'aide de composants CORBA. Les éléments présentés dans le cas d'utilisation en Figure ?? se traduisent en composants CORBA CCM (Figure 4.2).

La figure 4.3 présente une photo du fonctionnement du système sur un terminal utilisateur.

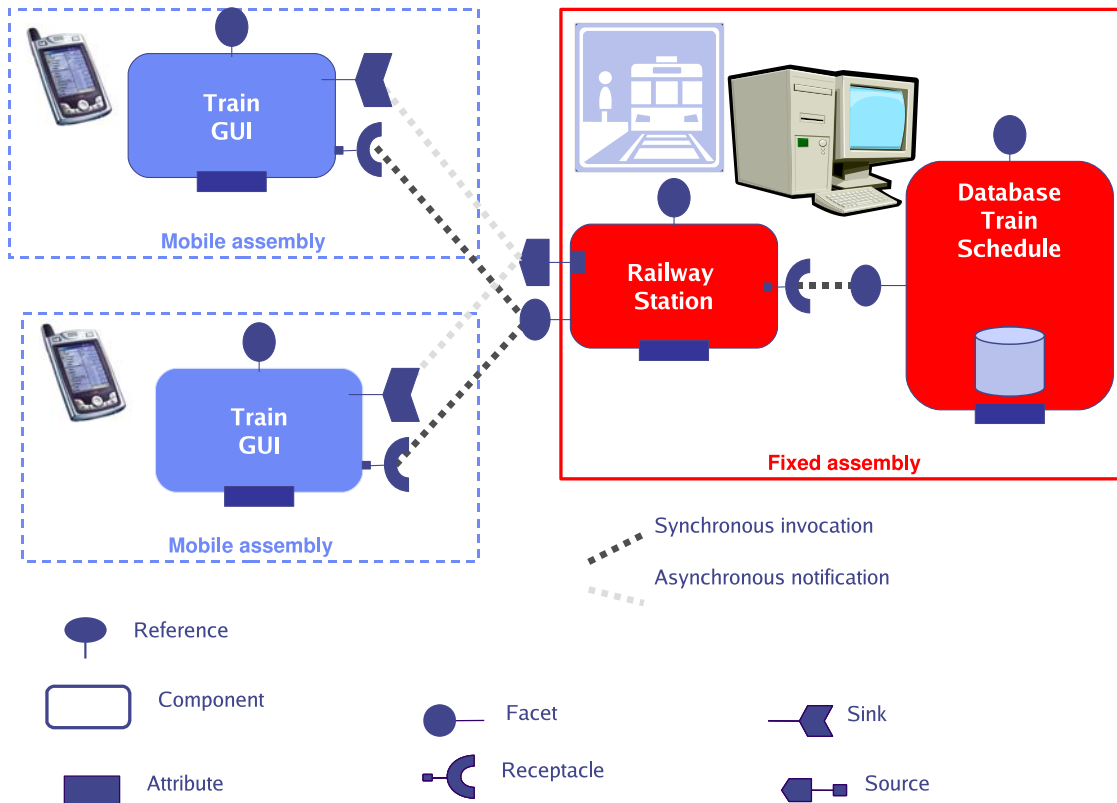


FIG. 4.2 – Architecture sous forme de composants CCM



FIG. 4.3 – Fonctionnement de l’application sur un terminal de type PDA

4.2.2 L’application “ notification d’arrêt ”

Actuellement, les usagers des transports en commun souffrent d’un manque d’informations pour les guider durant leur voyage. Lorsque cette information est disponible (*e.g.* affichage), son accès n’est pas toujours pratique (*e.g.* être assis dos au bandeau d’information), non personnalisé (*e.g.* mal-voyant), et souvent sous-exploitée (*e.g.* absence d’alerte pour usager distrait ou occupé).

L’application que nous proposons permet aux usagers d’être prévenus automatiquement et de manière individuelle dès que le bus approche de l’arrêt où ils souhaitent descendre [RGA04b]. Un prototype matériel de cet exemple a été réalisé et breveté [RGA04a].

Partant de cette expérience, nous pouvons exprimer les caractéristiques de cette application comme suit :

- Un usager est muni d’un terminal de type PDA communicant. Sur ce terminal est déployé un logiciel fournissant les informations relatives au réseau urbain qu’il utilise : plan du réseau de transport et horaires théoriques de passage des bus aux arrêts.
- Le service de notification d’arrêt (également appelé ICAU) lui permet de choisir sur son terminal l’arrêt où il souhaite descendre. Ce service de notification dialogue via Wi-Fi avec le calculateur embarqué dans le bus. Celui-ci, via un système de géo-localisation tel que le GPS, connaît à tout instant sa position sur le réseau de transport. Lorsque le bus approche d’un arrêt sur la ligne, le système embarqué envoie aux usagers du bus les informations relatives à cet arrêt (*e.g.* le nom de l’arrêt). Chaque terminal usager compare cette information avec le souhait exprimé par le possesseur du terminal. Si les informations correspondent, le terminal prévient son possesseur par une alarme de son

choix (sonore, visuelle, vibreur, ...) qu'il doit se préparer à descendre.

Nous pouvons donc voir dans ces besoins que certaines fonctionnalités (plan, horaire) seront toujours disponibles sur le terminal de l'utilisateur alors que le service de notification ne fonctionnera que dans un bus équipé d'un système de localisation et muni de l'application de notification.

La figure 4.4 présente une vue d'artiste de la zone de pertinence dans laquelle le système doit prévenir l'utilisateur que le bus est proche de l'arrêt. Les différentes couleurs représentent les zones de notification par sens de circulation du bus.

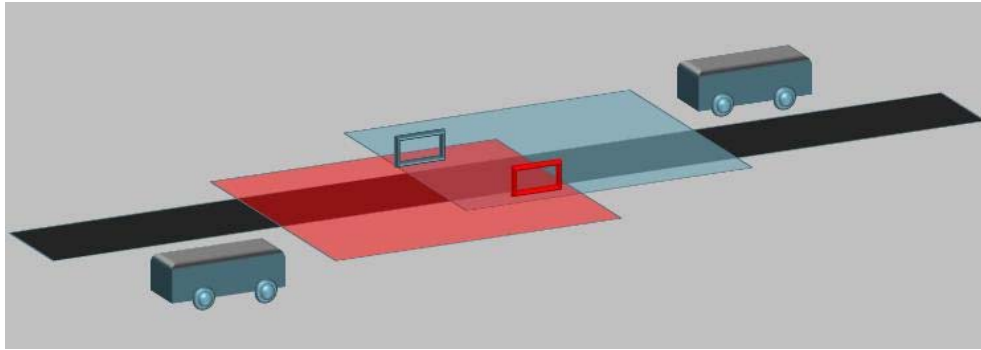


FIG. 4.4 – Vue d'artiste d'un arrêt de bus pour notre système

Diagramme cas d'utilisation

Le service est composé de trois domaines (figure ??) : Opérateur, Bus et Utilisateur. Le planificateur travaillant chez l'opérateur de transport gère des lignes de bus sur lesquelles il alloue des bus. Ensuite un conducteur est affecté à un bus sur une ligne pour son service. Finalement l'utilisateur emprunte un bus sur une ligne. Le service que nous souhaitons réaliser permettra un déploiement de l'application sur le terminal de l'utilisateur à partir du bus de (2) vers (1). Lorsque l'opérateur achète un nouveau bus pour le mettre dans sa flotte, il lui faut déployer le service dans le bus. Il y a donc un second déploiement de (3) vers (2).

Architecture sous forme de composants

A partir d'une description textuelle du problème, l'axe 1 a réalisé une modélisation de cette application sous forme d'*ApplicationPart* et de *Services*. Une transformation dans le monde des composants CCM est représentée figure 4.6.

Le composant *PositionGenerator* lit des données au format NMEA à partir d'une source de données (GPS réel, fichier texte de trace GPS, générateur aléatoire, ...) et les transforme en événements CORBA équivalents à la structure *Position*.

Le composant *EventDispatcher*. A partir d'une position GPS, ce composant génère des événements applicatifs : un changement de sens sur la ligne, le fait d'être dans une zone correspondant à un arrêt.

Le composant *ICAUServer*. Ce composant gère le dialogue avec les terminaux des usagers. Il fournit la liste des lignes ainsi que la liste des arrêts sur une ligne. Il se charge également de propager aux composants *ICAUGUI* le nom de l'arrêt courant sur la ligne à partir des événements reçus depuis *EventDispatcher*.

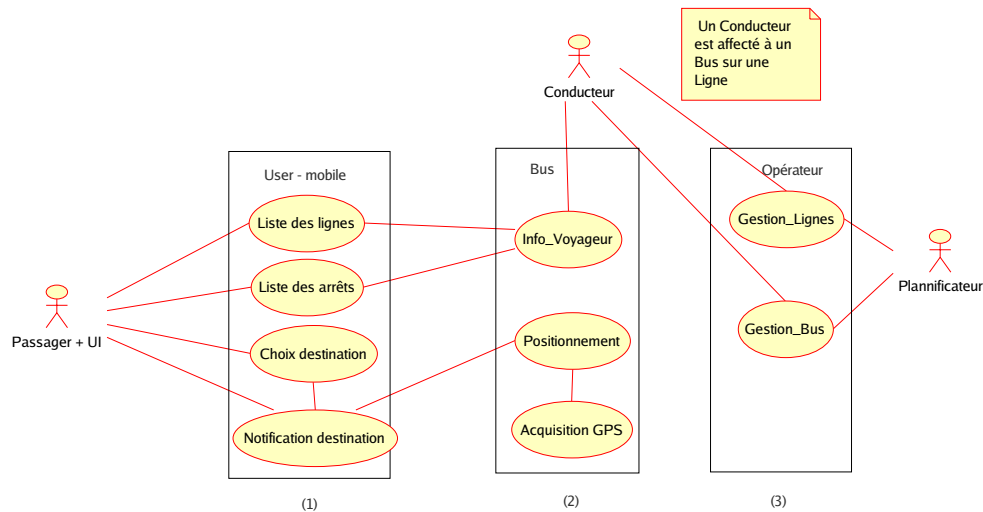


FIG. 4.5 – Diagramme de cas d'utilisation

Le composant *ICAUGUI* gère le dialogue avec le composant *ICAUServer* et interagit avec l'utilisateur.

Selon les capacités de calcul et de mémoire, il y aura plus ou moins de composants déployés sur le terminal de l'utilisateur. Dans le cas d'un assistant numérique, les composants *ICAUServer* et *ICAUGUI* sont déployés sur le PDA. Dans le cas d'un téléphone portable avec des ressources beaucoup plus restreintes, seul le composant *ICAUGUI* est déployé. Les autres composants sont déployés sur le serveur du bus.

Les figures 4.7 et 4.8 montrent les copies écrans de l'interface utilisateur sur assistant personnel. Afin de montrer le fonctionnement complet du système, des interfaces graphiques ont également été développées pour les autres composants. La figure 4.9 présente l'interface graphique réalisée pour le composant *EventDispatcher*. Nous pouvons voir sur cette interface le cheminement du bus ainsi que la définition des zones de notification des arrêts.

Cette application nous a permis d'appréhender le problème de la confidentialité et de la gestion de la vie privée. Selon l'endroit où sont déployés les composants (sur le terminal de l'utilisateur ou dans le bus) le système connaît plus ou moins d'informations sur la destination de l'utilisateur. Dans le cas de l'assistant personnel, aucune information relative au déplacement ne sort du PDA.

4.2.3 Conclusion

Les travaux dans cette action 4.1 du projet ont permis à l'équipe INRETS-LEOST de fournir des exemples réels d'applications avec des problèmes auxquels le monde des transports est confronté.

Ces exemples ont été étudiés sous divers angles par les autres axes du projet : pour la modélisation ou pour exprimer des nouvelles fonctionnalités pour des plate-formes ubiquitaires.

Finalement, ces exemples ont été implémentés et fonctionnent sur divers équipements mobiles.

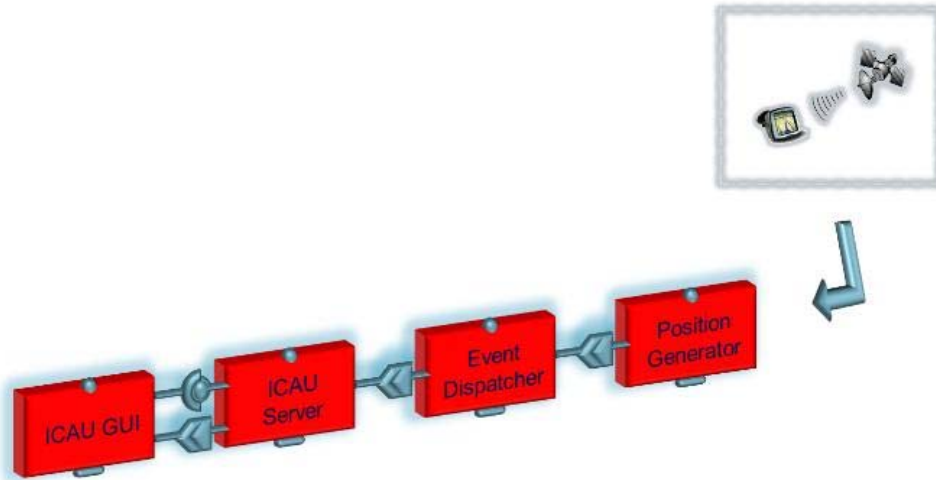


FIG. 4.6 – Architecture logicielle sous forme de composants

4.2.4 Retombées de l’axe

Les travaux de l’action 4.1 ont été présentés dans [FGM05a], [FGM05c], [FGM05b], [FGM05d], [CCG⁺06] que lors d’un workshop dans le cadre du réseau d’excellence EURNEX (European Railway Network of Excellence) lors d’une réunion du pôle 5 sur l’ “Intelligent Mobility” et lors d’un atelier de travail sur deux jours à Villeneuve d’Ascq du groupe mobilité du GDR.

4.3 Action 4.2 - Applications liées aux E-services

Dans le domaine des e-services, les démonstrateurs ont été essentiellement consacrés à la mise en oeuvre de services liés à la mobilité dans un cadre d’apprentissage.

On peut distinguer deux grandes classes de démonstrateurs : des démonstrateurs correspondant plutôt à des environnements ”augmentés” utilisant par exemple la géolocalisation ou de l’information contextuelle ; un démonstrateur orienté vers la diversification des modes d’interaction avec les environnements d’apprentissage, en particulier avec la possibilité d’utiliser des interactions vocales par téléphone lorsqu’un environnement de travail classique n’est pas disponible.

4.3.1 Environnement augmenté

Ces services portent sur l’utilisation de la localisation et l’authentification des utilisateurs afin de leur fournir des informations et services contextualisés et permettre par exemple de réaliser une vidéo communication de manière spontanée avec des personnes disponibles dans un bâtiment. Les scénarios et démonstrateurs définis reposent sur la présence d’une infrastructure permettant une communication fiable (Wifi). La figure 4.10 présente l’architecture globale permettant d’accéder aux services d’apprentissage et de collaboration à travers une infrastructure reposant sur des accès sans fil et des périphériques particuliers (PDA, écran tactile, identifiants RFID ...).

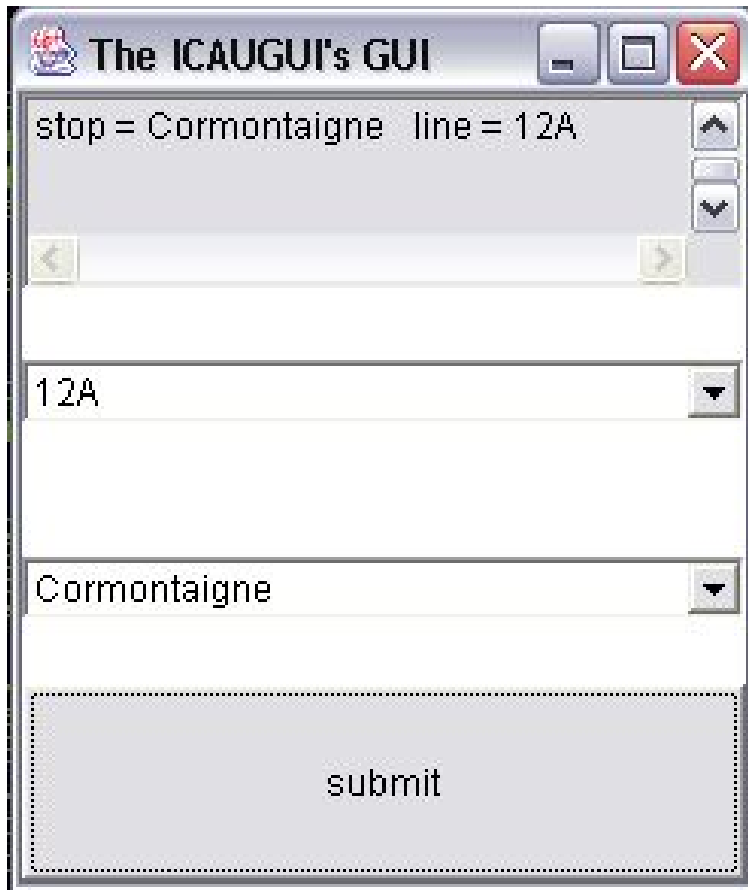


FIG. 4.7 – Copie d'écran de l'application sur PDA

La figure 4.11 présente un exemple de l'application de localisation sur PDA qui est fournie aux étudiants. Ces démonstrateurs ont permis d'étudier la mise en oeuvre de différentes technologies disponibles et les usages qui peuvent en être fait en lien avec le projet EUCUE.

4.3.2 Interactions multimodales

Un deuxième scénario est en cours d'expérimentation autour d'un service de blog mobile utilisé à des fins de suivi de projet et de collaboration entre étudiants et avec le tuteur. Dans ce cadre, nous explorons les différents canaux susceptibles de supporter cette activité (web, SMS/MMS et voix) [LLY⁺07][KVC07]. La figure 4.12 présente l'architecture du blog mobile. Cette architecture permet d'envisager l'ajout de nouveau canaux et repose sur une intégration et une orchestration de services dans une démarche architecture orientée service (SOA) pour l'intégration des environnements d'apprentissage.

4.3.3 Valorisation

Les travaux de l'axe 4.2 ont été présentés dans [LLY⁺07][KVC07]. Ils seront également présentés lors d'un atelier sur l'apprentissage mobile organisé par l'équipe LILF-NOCE [YTE07]. Ces démonstrateurs ont également permis de prendre des contacts avec des entreprises du pôle



FIG. 4.8 – Copie d'écran de l'application sur PDA

de compétitivité PICOM industries du commerce, ce qui a permis de déposer et d'obtenir le financement d'un projet ANR Télécoms (projet P-LearNet). Ce projet a été labellisé par le pôle de compétitivité.

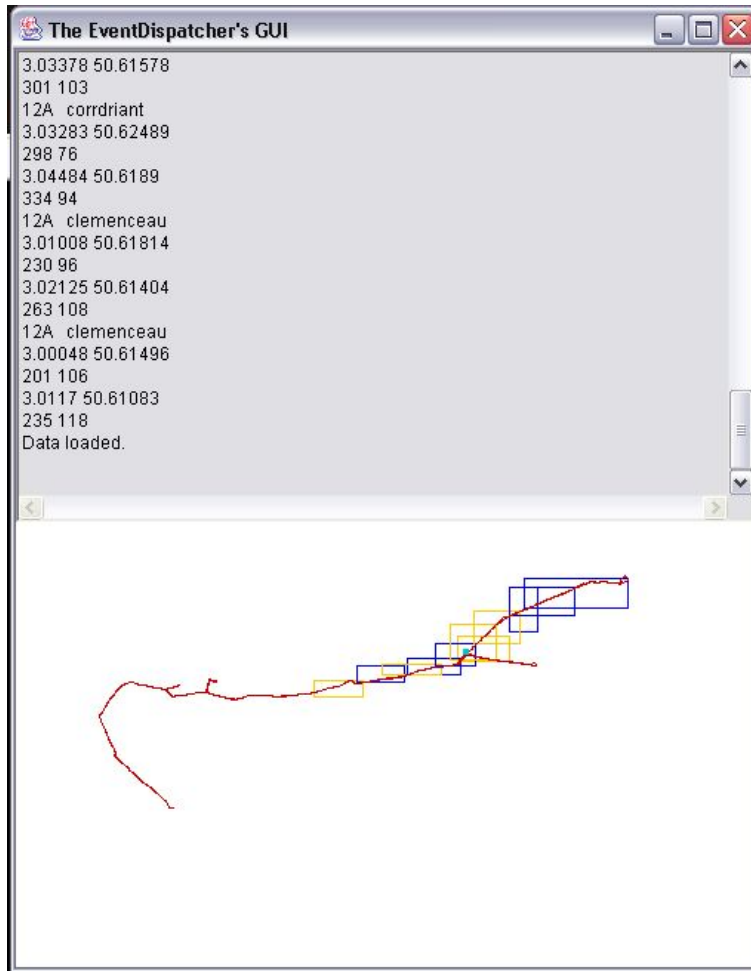


FIG. 4.9 – Interface graphique du composant *EventDispatcher*

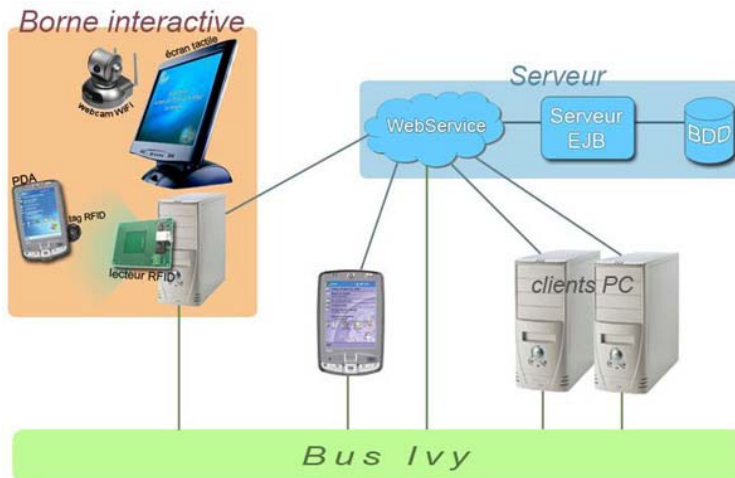


FIG. 4.10 – architecture de l'infrastructure de mobilité



FIG. 4.11 – Interface de localisation

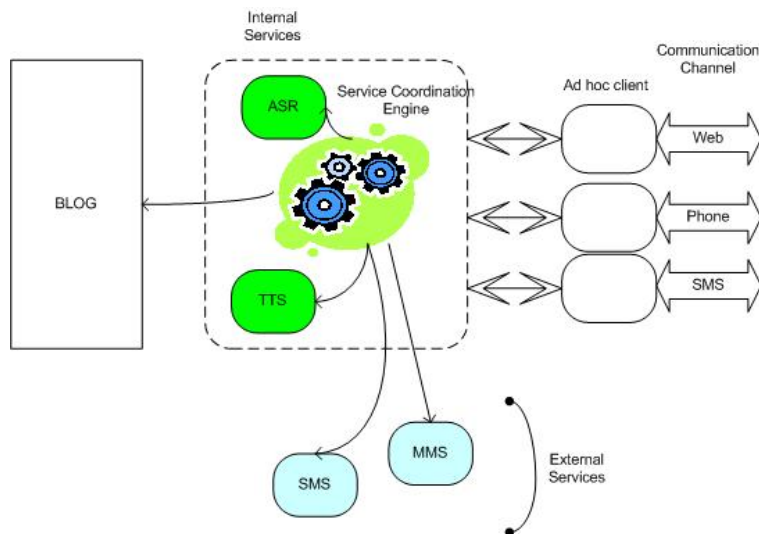


FIG. 4.12 – Modèle de localisation

Chapitre 5

Ecole des Mines de Douai

5.1 Liste des participants

- Noury Bouraqadi, Enseignant-Chercheur, 80%
- Abdelhak-Djamel Seriai, Enseignant-Chercheur, 80%
- Philippe Hasbroucq, Chef du département, 10%
- David Gille, Ingénieur, 20%
- Guillaume Grondin, Doctorant, 50%
- Gautier Bastide, Doctorant, 50%
- Houssam Fakih, Doctorant, 50%
- Ludovic Guégan, Master (5 mois en 2005) 100%
- Abbas Messad, Master (5 mois en 2006) 100%
- Liu Weigang, Master (4 mois en 2006) 100%
- Rémy Mouëza, Master (3 mois en 2006) 100%
- Misagh Shakeri, Master (4,5 mois en 2007) 100%

5.2 Contribution sur l'adaptation structurelle de systèmes à base de composants logiciels dans les environnements ubiquitaires

5.2.1 Problématique

Les applications ubiquitaires à base de composants posent de nombreux problèmes liés à leur adaptation. En fait, comme il est impossible de développer des composants répondant aux multiples variations concernant leurs utilisations, il était nécessaire de proposer des approches permettant d'adapter ces composants logiciels fournis sur étagères pour qu'ils puissent être ajustés à leurs contextes d'utilisation.

Pour cette contribution, notre intérêt a porté sur le problème de l'auto-adaptation des applications ubiquitaires par rapport à leurs infrastructures de déploiement. L'objectif est d'adapter ces applications aux propriétés des architectures matérielles de déploiement de ces applications, telles que les propriétés des composants matériels (e.g. les machines disponibles dans un contexte donné et leurs ressources) la configuration de l'architecture matérielle (e.g. comment ces machines sont connectées ensemble) et les propriétés des connexions entre ces machines (e.g. bande passante, distance, etc.).

Nous réalisons cette adaptation par l'étude d'une nouvelle facette d'adaptation que nous appelons l'adaptation structurelle. L'adaptation structurelle consiste en la possibilité de modifier la structure d'un composant logiciel en fonction des besoins de son utilisation. Le changement de la structure des composants adaptés est réalisé de manière transparente sans aucune modification des services ou des comportements de ces composants. Il s'agit, par exemple, de mettre à jour la structure d'un composant existant en le décomposant en différents sous-composants où chacun de ces nouveaux sous-composants offre une partie des fonctionnalités offertes par le composant initial et où l'assemblage de ces derniers fournit les mêmes services que le composant adapté. Il serait ainsi possible de pouvoir répartir, suivant les besoins, les services de ce composant sur les machines disponibles. Cela est possible par le déploiement de chacun des nouveaux sous-composants de manière séparée sur des sites différents qui peuvent être suivant la configuration, distribués ou non.

5.2.2 Résultats

1. Un état de l'art concernant les propriétés des applications ubiquitaires et context-aware. L'état de l'art est intégré dans le document représentant le premier livrable du projet MOSAIQUES.
2. Une approche d'adaptation structurelle de composants logiciels a été proposée. Cette approche étudie différentes facettes liées à ce problème [BSO07b]. Nous citons à titre d'exemple, l'adaptation structurelle de manière statique [BSO06b], [BSO06d], [BSO07b], [BSO06a], l'adaptation structurelle de manière dynamique [BSO06c], [BSO07a], la transformation de composants monolithiques en composants distribués [BSO06e], [SBO06a], [SBO06b] et le problème de l'auto-adaptation [BSO07c], etc.

5.3 Contribution sur l'auto-adaptation de composants logiciels pour une gestion efficace des ressources limitées dans les environnements ubiquitaires

5.3.1 Problématique

L'objectif de cette contribution était de développer un outil permettant d'expérimenter l'approche d'adaptation structurelle dans les environnements ubiquitaires.

5.3.2 Résultats

Cette partie a abouti au développement du prototype UbiBuildingSite. UbiBuildingSite est la réalisation d'un système " context-aware " pour le travail collaboratif. Il met en relation l'ensemble des personnes se trouvant sur un chantier de BTP. Il est développé en utilisant Java et le système expert JESS.

5.4 Contribution sur les architectures logicielles ouvertes et dynamiques pour les environnements ubiquitaires

5.4.1 Problématique

Les environnements ubiquitaires sont en général des environnements ouverts. En fait, l'ensemble des ressources matérielles et logicielles disponibles ne sont pas connues a priori. Il est ainsi difficile de préétablir une architecture et une configuration stable pour ce type d'applications.

5.4.2 Résultats

Pour pouvoir générer automatiquement la configuration logicielle qui répond le mieux à un contexte d'utilisation, nous avons proposé un modèle de composants et d'applications à base de composants permettant un assemblage " négocié " entre les composants. La configuration d'assemblage n'est pas définie mais " calculée " dynamiquement suivant les composants disponibles.

5.5 Contribution avec l'ensemble des équipes impliquées dans le cadre du projet MOSAIQUES à la définition d'un modèle structurel et dynamique d'une application ubiquitaire

L'objectif de ce travail est la proposition d'un modèle de structure et de la dynamique d'une application ubiquitaire. Ainsi, les propriétés de ce type d'applications a été étudié pour en déduire un modèle structurel et dynamique adéquat. Ce travail a abouti à la publication [CCG⁺06].

5.6 Contribution sur l'adaptation dynamique par ré-assemblage d'applications ubiquitaires à base composants

5.6.1 Problématique

L'adaptation est dite *dynamique* si ce processus est réalisé sans arrêter l'exécution du logiciel. Une telle dynamique s'avère nécessaire notamment dans les applications ubiquitaires où un arrêt peut rendre les services inexploitable dans la pratique, coûteux financièrement ou même dangereux du point de vue humain ou environnemental. Cette dynamique est d'autant plus délicate que les adaptations ne sont pas prévisibles.

Nous nous sommes intéressés dans cette partie à l'adaptation dynamique des applications à base de composants. Dans ce contexte, la réalisation d'une adaptation se traduit par une *reconfiguration* de l'assemblage de composants constituant l'application. Cette reconfiguration consiste à ajouter/supprimer/remplacer des composants ou des connexions entre composants. La reconfiguration de l'application englobe également la modification des attributs des composants qui la compose.

5.6.2 Résultats

Le travail réalisé dans cette partie a conduit à la définition de MADCAR. Il s'agit d'un modèle abstrait de moteur d'adaptation pour applications à base de composants logiciels. Nous qualifions MADCAR d'abstrait car il est indépendant de tout modèle de composant. Les hypothèses minimales que nous faisons sont : homogénéité des composants en terme de modèle, auto-documentation des composants par contrats, connecteurs entre composants par composants d'interaction. Les contrats documentant les composants concernent au minimum les attributs paramétrables et les interfaces fournies ou requises.

5.7 Contribution sur les architectures d'agents logiciels adaptables pour la gestion de ressources limitées et variables

5.7.1 Problématique

L'adaptation est l'une des propriétés les plus désirables dans la communauté des Systèmes Multi-Agents (MAS). Le but étant d'approcher un comportement optimal compatible avec les ressources matérielles et logicielles disponibles à chaque instant. Notre point de départ correspond aux architectures dites hybrides qui mixent une couche réactive assurant un temps de réponse court et une couche cognitive qui permet d'avoir un comportement intelligent.

5.7.2 Résultats

Nous avons étudié l'utilisation du modèle MADCAR pour faciliter la conception d'agents logiciels (au sens Systèmes Multi-Agents). Il s'agit de modéliser un agent autonome capable de se ré-assembler à la suite d'un changement contextuel. Dans cet agent, nous séparons son comportement normal (matérialisé par un assemblage de composants) de son comportement d'adaptation. Pour ce faire, nous intégrons à cet agent un moteur permettant de déclencher, décider et réaliser des ré-assemblages de composants. Le processus d'assemblage est piloté par une politique d'assemblage explicite, précise et ouverte, qui doit être spécifiée par le concepteur de l'agent. Les agents construits sur la base de MADCAR peuvent être considérés comme hybrides. Le moteur d'assemblage correspond à la couche cognitive alors que les traitements correspondent à des réflexes. Cependant, il est possible que les traitements de l'agent soient eux-mêmes découpés en deux parties, l'une réactive et la seconde délibérative.

Parallèlement au travail sur les agents à base de MADCAR, nous avons étudié la problématique de la gestion de ressources. Cela nous a conduit à l'introduction du concept "d'hybridation paramétrable". Il s'agit d'adapter un agent hybride de manière à ce qu'il réalise des tâches de manière réactive ou de manière cognitif. Le passage d'une couche à une autre est réalisé selon les ressources disponibles. Nous avons ainsi abouti à une architecture d'agent hybride générique complémentaire à celle utilisant MADCAR. En plus de permettre le paramétrage de "l'hybridation", notre architecture est réflexive. Cette propriété lui confère la spécificité de pouvoir adapter les stratégies mêmes de l'adaptation. Le coût de l'adaptation est ainsi régulé en fonction des ressources disponibles.

5.8 Logiciels développés

- Scorpio (<http://ocm.ensm-douai.fr/bastide/index.php?%20n=PmWiki.ScorpioProject>)
Scorpio (Software COmponent stRuctural adaPtatIOn) est un outil permettant l'adaptation structurelle statique et dynamique de composants et d'applications à base de composants. L'outil est développé en utilisant Julia, l'implémentation Java du modèle de composants Fractal.
- Auto-Fractal (<http://csl.ensm-douai.fr/grondin/AutoFractal>)
Basé sur le modèle MADCAR, Auto-Fractal est un moteur d'assemblage de composants Fractal qui permet la construction d'applications auto-adaptables. Une application auto-adaptable est capable de déclencher, planifier et réaliser automatiquement et dynamiquement le ré-assemblage des composants qui la constituent. Lorsque certains changements de contexte sont détectés (ex : apparition/disparition du réseau), des décisions sont prises automatiquement pour ré-assembler l'application. Pour ce faire, le concepteur doit décrire (grâce à un langage de contraintes) d'une part, les fonctionnalités désirées de l'application visée, et d'autre part, une politique d'adaptation pour piloter les (ré-)assemblages. Ces deux descriptions sont séparées l'une de l'autre pour faciliter leur compréhension/évolution. De plus, notre langage de description des fonctionnalités et des politiques d'adaptation est découplé des composants à assembler (pas de référence directe aux composants). Cette modularité simplifie la conception des applications auto-adaptables, mais aussi leur évolution.
- UbiquiTalk (<http://csl.ensm-douai.fr/UbiquiTalk>)
UbiquiTalk est une plate-forme pair à pair (P2P) ouverte qui permet la découverte automatique d'équipements sans nécessiter aucune infrastructure. Ainsi, UbiquiTalk peut être utilisé aussi bien au sein d'un réseau structuré (par exemple le LAN d'une entreprise) ou d'un réseau ad hoc à base d'une technologie sans fil (par exemple Wifi). Chaque pair UbiquiTalk, appelé "hôte" peut être fournisseur de service, client de service, ou les deux à la fois. Les utilisateurs disposent d'une interface graphique pour administrer et paramétrer leurs machines et choisir les services distants à importer (i.e. utiliser localement) et les services locaux à exporter (i.e. mettre à disposition à travers le réseau). Cette interface permet également d'observer les connexions et déconnexions des machines liées notamment aux déplacements des utilisateurs nomades. Deux déclinaisons d'UbiquiTalk ont été implantées : l'une pour des équipements mobiles de type PDA et la seconde pour des machines de bureau de type PC.

Chapitre 6

INRETS - Laboratoire LEOST

6.1 Liste des participants

- Christophe Gransart, CR, 50 %
- Aurélien Bocquet, doctorant, 10 %
- Stéphanie Berny, stagiaire, 4 mois
- Florent Demuynck, stagiaire, 4 mois
- Mossaab Melloul, stagiaire, 6 mois

6.2 Objectifs et problèmes développés

Le monde des transports est de par sa nature un monde mobile dans lequel l'informatique est de plus en plus présente. Dans ce contexte, nous avons besoin d'applications ubiquitaires/contextuelles qui vont permettre à l'utilisateur de voyager en toute sérénité sans avoir à se préoccuper de l'état de ses applications en fonction dans l'endroit où il se trouve.

De plus, avoir un environnement actif qui permet, par exemple, de présenter automatiquement à l'utilisateur de nouvelles applications en fonction du lieu où il se trouve nous permet d'imaginer de nouvelles applications.

L'utilisation des applications ubiquitaires dans le domaine des transports est très variée : cela va de l'affichage des documentations techniques lorsqu'un opérateur de maintenance est près d'un équipement jusqu'à de l'information voyageur en temps réel.

Dans le cadre du projet MOSAIQUES, nous avons défini et réalisé deux applications dans le domaine de l'information voyageur. La première application permet d'être notifié du quai de départ d'un train dès l'arrivée de l'utilisateur dans la gare. La seconde application permet à un usager de transports en commun tel qu'un autobus d'être notifié automatiquement et de manière personnelle que le bus arrive à proximité de l'arrêt choisi. Ces applications seront détaillées dans la section résultats.

Nous allons d'abord faire un rappel des besoins que nous avons soulevés au début de ce projet. Ces besoins se sont traduits par des développements logiciels en lien avec l'équipe LIFL-GOAL dans l'axe 2.



FIG. 6.1 – Cycle de vie d'une application contextuelle

6.2.1 Cycle de vie d'une application contextuelle

Une application contextuelle est une application qui permet d'apporter à l'utilisateur des informations ou des services à un moment déterminé et en un lieu bien précis. La figure 6.1 présente le patron en détail.

Lorsqu'un utilisateur muni d'un terminal mobile se trouve en dehors d'une zone où existe une application contextuelle (6.1.a), son terminal ne lui offre aucun service/application.

Dans un second temps, lorsqu'il arrive sous une couverture de réseau sans fil offrant des services contextuels, son terminal lui présente automatiquement la liste des services disponibles à l'endroit où il se trouve (6.1.b). L'utilisateur choisit alors un service à utiliser. L'infrastructure déploie alors automatiquement le service sur le terminal de l'utilisateur.

Tant que l'utilisateur se trouve sous la couverture du réseau sans fil, il peut utiliser les différents services mis à sa disposition (6.1.c).

Finalement, lorsque l'utilisateur se déplacera hors de la couverture du réseau, les applications contextuelles ne seront plus disponibles (6.1.d). Plus précisément, le comportement des services sera dépendant de la logique applicative. Certains services seront purement et simplement supprimés du terminal, d'autres pourront fonctionner en mode autonome sans le réseau ou sauvegarderont leurs données jusqu'à une prochaine activation.

6.2.2 Le mécanisme de découverte

Le service de découverte de service est une des briques logicielles clé pour la réalisation d'applications contextuelles. Ce service de découverte de services permet à un utilisateur muni

d'un terminal

- d'obtenir une information concernant sa position physique (là où il se trouve),
- d'obtenir la liste des services potentiellement disponible en ce lieu,
- finalement de choisir un service et de déclencher le déploiement et l'utilisation du service.

6.3 Les résultats

Nous avons défini deux scénarios applicatifs qui ont servi de point de départ pour une mise en application des travaux de modélisation effectués dans l'axe 1 et pour des travaux d'extension de la plate-forme à composants OpenCCM de telle sorte qu'elle puisse accueillir des applications répondant au cycle de vie décrit ci-avant.

6.3.1 Définition de scenario applicatifs

Le scénario “ trains au départ ” L'application est issue du monde ferroviaire. L'objectif de cette application est de permettre à un voyageur se déplaçant en train de connaître automatiquement le quai de départ de son train. Cette information est automatiquement fournie sur son terminal mobile personnel (PDA, smartphone, ...) lorsqu'il entre dans la gare de départ.

Cette application est composée de plusieurs éléments logiciels qui vont s'exécuter sur des machines soit fixes (installées à demeure dans la gare) soit des machines mobiles (les terminaux des passagers).

La figure 6.2 présente le cas d'utilisation de cette application.

Les différentes fonctions de l'application peuvent être résumées comme suit :

- *Database train schedule* Ce composant contient la liste des horaires de l'ensemble des trains de l'opérateur.
- *Train Component* Ce composant est installé dans la gare sur une machine fixe. Ce composant va extraire de la base de données de l'opérateur la liste des trains qui concerne la gare à laquelle il est associé. Il doit également répondre aux requêtes qui vont venir des terminaux mobiles (composant suivant).
- *Service mobile part GUI / Text to Speech UI* Ces composants s'exécutent sur les terminaux des utilisateurs. Certains utilisateurs préféreront une interface graphique (GUI), d'autres utiliseront une interface à base de synthèse vocale qui leur lira la liste des trains au départ ainsi que le quai associé.

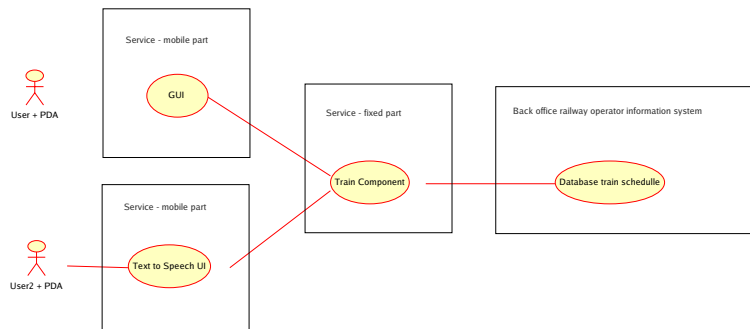


FIG. 6.2 – Diagramme de cas d'utilisation

Le scénario “ notification d’arrêt ” Actuellement, les usagers des transports en commun souffrent d’un manque d’informations pour les guider durant leur voyage. Lorsque cette information est disponible (*e.g.* affichage), son accès n’est pas toujours pratique (*e.g.* être assis dos au bandeau d’information), non personnalisé (*e.g.* mal-voyant), et souvent sous-exploitée (*e.g.* absence d’alerte pour usager distrait ou occupé).

L’application que nous proposons permet aux usagers d’être prévenus automatiquement et de manière individuelle dès que le bus approche de l’arrêt où ils souhaitent descendre [RGA04b]. Un prototype matériel de cet exemple a été réalisé et breveté [RGA04a].

Partant de cette expérience, nous pouvons exprimer les caractéristiques de cette application comme suit :

- Un usager est muni d’un terminal de type PDA communicant. Sur ce terminal est déployé un logiciel fournissant les informations relatives au réseau urbain qu’il utilise : plan du réseau de transport et horaires théoriques de passage des bus aux arrêts.
- Le service de notification d’arrêt (également appelé ICAU) lui permet de choisir sur son terminal l’arrêt où il souhaite descendre. Ce service de notification dialogue via Wi-Fi avec le calculateur embarqué dans le bus. Celui-ci, via un système de géo-localisation tel que le GPS, connaît à tout instant sa position sur le réseau de transport. Lorsque le bus approche d’un arrêt sur la ligne, le système embarqué envoie aux usagers du bus les informations relatives à cet arrêt (*e.g.* le nom de l’arrêt). Chaque terminal usager compare cette information avec le souhait exprimé par le possesseur du terminal. Si les informations correspondent, le terminal prévient son possesseur par une alarme de son choix (sonore, visuelle, vibreur, ...) qu’il doit se préparer à descendre.

Nous pouvons donc voir dans ces besoins que certaines fonctionnalités (plan, horaire) seront toujours disponibles sur le terminal de l’usager alors que le service de notification ne fonctionnera que dans un bus équipé d’un système de localisation et muni de l’applicatif de notification.

La figure 6.3 présente une vue d’artiste de la zone de pertinence dans laquelle le système doit prévenir l’usager que le bus est proche de l’arrêt. Les différentes couleurs représentent les zones par sens de circulation du bus.

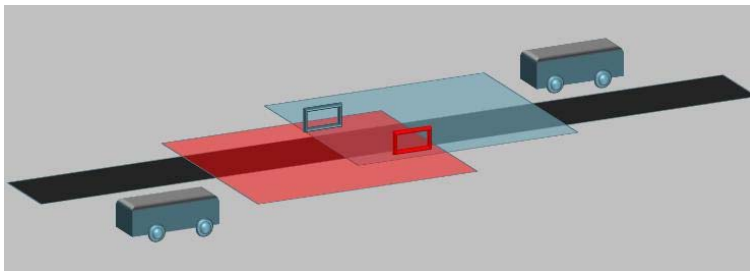


FIG. 6.3 – Vue d’artiste d’un arrêt de bus pour notre système

6.3.2 Exemple d’application pour l’axe 1

Le scénario “ notification d’arrêt ” a permis aux équipes impliquées dans l’axe 1 d’appliquer et d’affiner leurs travaux sur la méta-modélisation et l’approche MDA des applications ubiquitaires.

6.3.3 Proposition d’extension de la plate-forme OpenCCM dans l’axe 2

Le scénario “ trains au départ ” a permis à l’équipe GOAL impliquée dans l’axe 2 d’étendre leur plate-forme pour prendre en compte le contexte de l’utilisateur (au travers de sa localisation, son type d’équipement) afin de lui fournir une liste d’applications qui potentiellement peuvent l’intéresser et qui peuvent être déployées sur son terminal.

Des composants spécifiques ont été développés dans l’infrastructure OpenCCM pour permettre la découverte des services et le déploiement à la volée des services logiciels sur les terminaux des usagers.

6.3.4 Réalisation d’applications dans l’axe 4

Ces diverses applications ont été réalisées par l’INRETS-LEOST dans le cadre de son travail dans l’axe 4.1 de démonstration de faisabilité pour le monde des transports.

6.4 Logiciels développés

Les logiciels développés correspondent aux scénarios applicatifs présentés :

- l’application “ trains au départ ” dans le monde ferroviaire
Cette application a été réalisée à l’aide de composants CORBA. Les éléments présentés dans le cas d’utilisation en Figure 6.2 se traduisent en composants CORBA CCM (Figure 6.4).
- l’application “ notification d’arrêt ” pour tout mode de transport public de surface. De plus cette application a été réalisée sur téléphone portable et sur assistant numérique.

6.5 Retombées du projet

Les travaux ont été présentés dans [FGM05a],[FGM05c],[FGM05b],[FGM05d],[CCG+06] ainsi que lors d’un workshop dans le cadre du réseau d’excellence EURNEX (European Railway Network of Excellence) lors d’une réunion du pôle 5 sur l’ “Intelligent Mobility” et lors d’un atelier de travail sur deux jours à Villeneuve d’Ascq du groupe mobilité du GDR.

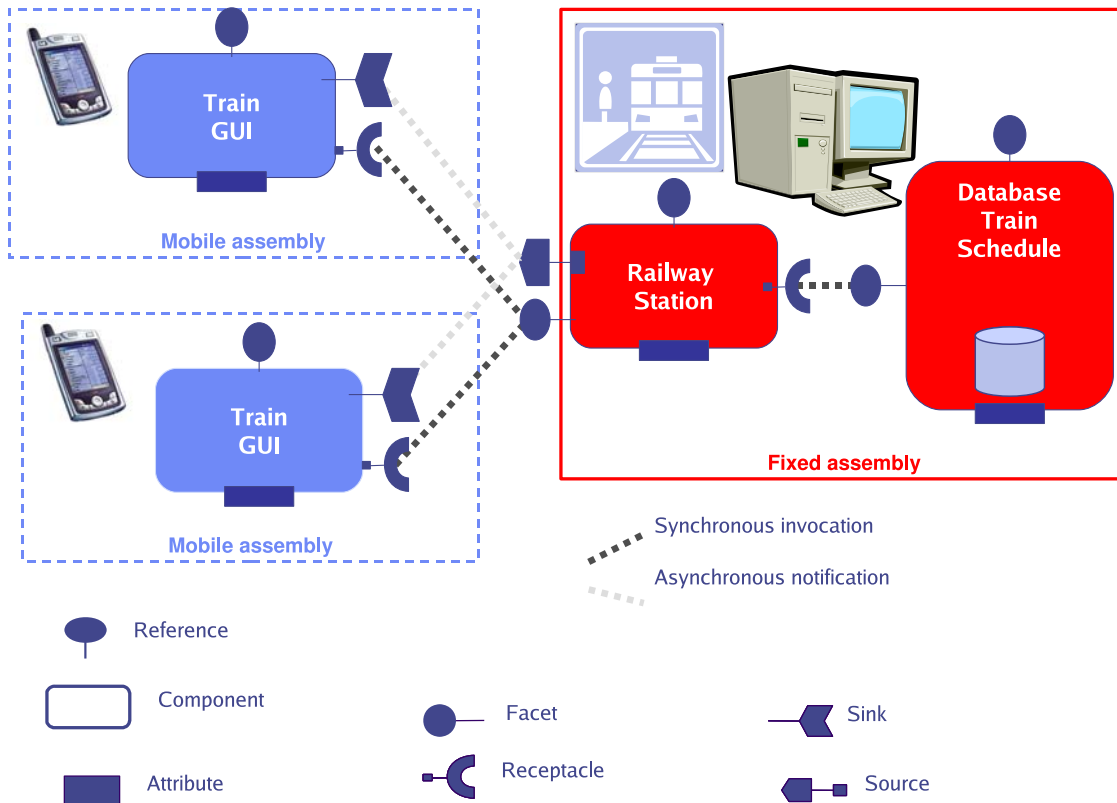


FIG. 6.4 – Architecture sous forme de composants CCM

Chapitre 7

USTL - Laboratoire LIFL - Equipe GOAL

7.1 Liste des participants

- Olivier Caron (MdC) 50%
- Bernard Carré (MdC) 50%
- Maja D'Hondt (CR2 INRIA), 10%
- Laurence Duchien (PR) 50%
- Jean-Marc Geib (PR) 10%
- Areski Flissi (IR CNRS) 50%
- Anne-Françoise Le Meur (MdC) 10%
- Raphaël Marvie (MdC) 50%
- Philippe Merle (CR1 INRIA) 30%
- Renaud Pawlak (CR2 INRIA) 10%
- Lionel Seinturier (PR) 50%
- Gilles Vanwormhoudt (Enseignant-chercheur) 50%
- Areski Flissi (IR CNRS) 50%
- Missi Tran (IE INRIA) 10%
- Nicolas Petitprez (IE INRIA) 10%
- Olivier Barais, doctorant, 50% (septembre 2004 – novembre 2005)
- Alexis Muller, doctorant, 50% (septembre 2004 – juin 2006)
- Romain Rouvoy, doctorant, 50% (septembre 2004 – décembre 2006)
- Carlos Noguera, doctorant, 50% (septembre 2005 – mai 2007)
- Jérémy Dubus, doctorant, 50% (septembre 2005 – mai 2007)
- Ales Plsek, doctorant, 50 % (septembre 2006- mai 2007)
- Nicolas Pessemier, doctorant, 10% (septembre 2004-mai 2007)
- Dolorès Diaz, doctorant 10% (septembre 2004-mai 2007)
- Denis Conan, invité 25% (janvier 2006 – juin 2006)
- Johan Fabry, Post-doc INRIA 25% (sept 2006 – août 2007)
- Johan Brichau, Post-doc CNRS 10% (janv 2006 – sept 2006)
- Xiaoning Sun, Stagiaire 100 % (Février 2006-Juin 2006)
- Guillaume Dufrene, Stagiaire, 100%(Février 2006-Mars 2006)

7.2 Problèmes abordés et résultats

L'équipe GOAL a participé à différentes actions dans les travaux des axes 1 et 2.

7.2.1 Cadre de développement

Les objectifs du thème modèle de l'équipe GOAL étaient dans l'axe 1 d'évaluer les possibilités d'application de nos travaux sur la modélisation (composition de modèles et processus de modélisation) dans le contexte des applications ubiquitaires.

Parmi les spécificités de ces applications nous avons plus particulièrement travaillé sur l'adaptation. En premier lieu comment modéliser les possibilités d'adaptation structurelle d'une application par rapport à son contexte. Ensuite, comment modéliser la dynamique de l'adaptation d'une application en fonction de son exécution (et donc par rapport à son contexte). Et enfin, comment intégrer cette modélisation dans un processus de développement dirigé par les modèles.

Les objectifs du thème intergiciels et architecture des logiciels étaient dans l'axe 2 de proposer des solutions permettant de modéliser et concevoir des plates-formes permettant la prise en compte des problèmes liés aux environnements ubiquitaires. Nous pouvons résumer nos deux sous-objectifs dans ce thème comme étant :

- La gestion des transactions réparties et hétérogènes en contexte ubiquitaire sachant que les participants à ces transactions sont répartis sur des machines volatiles et mobiles et que ces participants peuvent mettre en oeuvre différents modèles et standards de transactions ainsi que différents algorithmes de validation des transactions.
- L'automatisation du déploiement d'applications en contexte ubiquitaire sachant que les composantes d'une même application peuvent être réparties sur plusieurs sites et que ces composantes peuvent être réalisées au dessus de différents modèles de logiciel.

7.2.2 Les résultats dans l'axe 1

Nos propositions dans l'axe 1 concernent trois aspects de la modélisation des applications ubiquitaires : des moyens de modélisation (méta-modèles), une démarche de conception (processus) et des moyens d'exploitation (transformation de modèles). Finalement nous avons travaillé sur la transformation de programmes existants en environnement connecté dans des programmes pour des environnements ubiquitaires.

Méta-modèles pour les applications MOSAIQUES Notre première contribution est la définition de méta-modèles adaptés à la conception d'application ubiquitaires dans le cadre de Mosaïques. Ces applications ont comme spécificité de devoir s'adapter à leur contexte d'exécution, aussi bien en termes de services (logiciels) que de ressources (matérielles) disponibles. Il est donc important de pouvoir, dès la phase de conception d'une application de pouvoir spécifier ses éléments, mais aussi leurs dépendances vis-à-vis du contexte.

Pour cela, nous avons proposé un méta-modèle structurel permettant la modélisation des services (requis et fournis) d'une application et de contraintes d'usage d'un service de l'application. Ces contraintes sont exprimées abstraitement en fonction du contexte et seront évaluées à l'exécution : par exemple, le service *signaler l'arrêt* de l'application bus a comment contrainte la présence du réseau et du service *notification des arrêts*. Chaque service d'une application MOSAIQUES représente donc un *point d'adaptation* de l'application qui pourra

fonctionner en mode *normal* (tous les services sont disponibles) ou en mode *dégradé* (seuls certains services sont disponibles).

Enfin, nous avons proposé un modèle d'exécution pour les applications MOSAIQUES. Ce modèle précise quand peut se faire une adaptation et comment une application réagit aux changements de contexte. Il définit principalement quels sont les événements qui déclenchent l'adaptation de l'application par le calcul des contraintes en fonction des nouveaux éléments de contexte et la prise de décision quant à la possibilité d'utiliser ou non un service et/ou l'application (si aucun service n'est disponible l'application ne peut que attendre un nouveau changement de contexte).

Processus de conception Le méta-modèle structurel a été défini sur la base de la représentation par vues. Les objets métiers utilisés dans un équipement nomade sont modélisés au sein d'une vue. Il est donc possible d'identifier l'identité conceptuelle de ces objets métiers dans les différents équipements constituant une application ubiquitaire et cela favorise la synchronisation de ces équipements.

Cette représentation par vue a aussi été la base de la définition du processus de conception des applications MOSAIQUES. Les applications et leurs modèles étant complexes, il est important de pouvoir décomposer leur conception afin de se concentrer sur une préoccupation particulière. Le processus de conception est défini comme un enchaînement d'étapes de modélisation où chaque étape permet de concevoir une vue de l'application (une préoccupation). Le modèle complet d'une application est alors défini comme la fusion de ces différentes vues.

Du modèle d'une application à son déploiement Une fois le modèle d'une application réalisée, l'objectif est de l'exploiter pour réaliser concrètement cette application. Au vue de la multitude des plates-formes disponibles, l'axe 2 a retenu FDF comme service de déploiement. Afin de pouvoir exploiter dans l'axe 2 les modèles d'applications produits par l'axe 1 tout en restant à un haut niveau d'abstraction / d'indépendance, nous avons choisi de projeter le modèle d'une application vers le méta-modèle de déploiement de Jérémy Dubus [DM06a] à l'aide d'une transformation de modèle. L'avantage de ce méta-modèle de déploiement est qu'il est outillé pour piloter le service FDF mais aussi d'autres services technologiques de déploiement. Il représente ainsi le pivot entre les travaux de l'axe 1 et de l'axe 2.

La transformation de programmes pour environnements ubiquitaires Programmer des applications pour des environnements ubiquitaires est une tâche difficile. Plusieurs limitations inhérentes au domaine, telles que les connexions volatiles, ou encore la faible puissance des batteries, amènent le développeur à revoir ses méthodes de conception. Nous avons dans ce travail voulu partir de programmes existants et les transformer en programmes fonctionnant sur des environnements ubiquitaires. Nous avons travaillé sur deux particularités de ces programmes qui sont les connexions volatiles et la conscience du contexte. Nous avons basé notre approche sur les techniques de génération et de transformation de code sur lesquelles nous travaillons dans l'équipe, et plus particulièrement à l'aide notre outil Spoon (<http://spoon.gforge.inria.fr>). Nous avons proposé des annotations et des transformations associées permettant la transformation d'une application fonctionnant en mode connecté en une application fonctionnant dans un environnement ubiquitaire. Un ensemble d'annotations a été proposé et un outil de transformation de code appelé Graffiti a été développé. Ce travail se place dans le cadre de la thèse Carlos Noguera (Financement Réseau d'Excellence

AOSD/INRIA) et a donné lieu à deux publications [FN07, PDNP07].

7.2.3 Les résultats dans l'axe 2

Les propositions du thème intergiciels de l'équipe se sont concrétisées par plusieurs travaux décrits ci-dessous.

La définition et la réalisation de la plate-forme OpenCCM/MOSAIQUES Celle-ci est une plate-forme d'exécution à base de composants pour les applications ubiquitaires [FGM05d, FGM05b] permettant la découverte et le déploiement à la demande de ces applications sur des PDAs [FGM05c]. Cette plate-forme a été mise en oeuvre sur des démonstrateurs de l'axe 4 dans le domaine du transport en collaboration avec Christophe Gransart de l'INRETS [FGM05a].

La définition et la réalisation du canevas flexible GoTM pour la gestion des transactions dans le cadre de la thèse de Romain Rouvoy. Ce canevas a été conçu selon une approche à base de composants et de patrons de conception permettant de construire des services évolutifs de gestion de transactions [RM06, RM07]. Dans [SARM05, RSAM06b], nous avons montré comment ce canevas pour être mis en oeuvre pour implanter divers algorithmes de validation tels que 2PC, 2PC-PA et 2PC-PC ainsi que pour permettre l'auto-configuration d'un service de transactions afin qu'il choisisse l'algorithme de validation le plus approprié au contexte d'exécution. Dans [RSAM06a], nous avons montré comment ce canevas permet de construire un service de transactions supportant différents standards de gestion des transactions, à savoir les standards OMG Object Transaction Service (OTS), SUN Java Transaction Service (JTS) et Web Service Atomic Transaction (WS-AT). De plus, ce service permet alors de composer élégamment et efficacement des transactions avec des participants hétérogènes.

La définition et la réalisation du canevas Fractal Deployment Framework (FDF) permettant de déployer toute sorte de logiciel comme des composants applicatifs, des serveurs d'applications, des services intergiciels, des supports d'exécution langage et des systèmes d'exploitation. Ce canevas permet d'automatiser le déploiement de la plate-forme OpenCCM/Mosaiques sur un ensemble de stations fixes et de PDAs, mais aussi sur une grille de calculateurs [FM06].

La définition et le prototypage de l'environnement DACAR DACAR est un environnement pour le déploiement autonome et l'auto-adaptabilité d'applications en environnement distribué et ouvert [DM06c, DM06b] développé dans le cadre de la thèse de Jérémie Dubus. Il s'agit d'un environnement permettant aux applications ubiquitaires de s'auto-déployer, s'auto-configurer et s'auto-adapter en fonction des apparitions et disparitions de machines dans un contexte ubiquitaire. DACAR s'appuie sur une approche dirigée par les modèles [DM06a]. Le prototype actuel de DACAR utilise le canevas FDF pour déployer l'infrastructure intergicelle (par exemple OpenCCM/Mosaiques) et gère ensuite l'autonomie des applications ubiquitaires.

Les Indicateurs logiques pour applications ubiquitaires La construction d'application ubiquitaire nécessite de pouvoir collecter un certain nombre de données provenant de l'environnement physique comme les caractéristiques matérielles des équipements ou l'état des

ressources (présence/absence du réseau, débit, capacité batterie, ...). La capture de ces données doit pouvoir être réactualisée en permanence, les données doivent pouvoir être agrégées, lissées et synthétisées afin de fournir des indicateurs logiques de haut niveau sur lesquelles les applications peuvent se baser pour prendre des décisions d'adaptation. Le framework COSMOS a été conçu par LIFL/GOAL (L. Seinturier et R. Rouvoy) en collaboration avec Denis Conan (GET/INT Evry) pour permettre de concevoir et implémenter facilement ces indicateurs logiques. La difficulté réside dans le fait que de nombreuses données physiques et logiques doivent pouvoir être capturées via des sondes et que leur composition fait apparaître une combinatoire élevée. Pour cela COSMOS fournit un ensemble de composants logiciels (modèle Fractal) et de schémas d'assemblages (langage Fractal ADL) qui permettent de factoriser les développements et offrent la possibilité de développement de nouveaux indicateurs à moindre coût. Ce travail a donné lieu à une publication dans une conférence internationale [CRS07] et une publication dans une revue française [CRS08].

Les politiques de reconfiguration d'architectures logicielles . Les applications informatiques modernes sont de plus en plus conçues et réalisées comme un assemblage de composants logiciels. Dans le cadre de l'ubiquitaire, l'adaptation intervient non seulement au niveau des composants pris individuellement, mais également sur l'assemblage de ces composants. Cette étude a pour but d'étudier les différentes solutions existantes pour la définition de politiques de reconfiguration d'architecture logicielle et de prototyper une solution pour le langage de description d'architecture du modèle de composant Fractal. Cette étude a donné lieu au stage de master recherche de X. Sun sous la supervision de L. Seinturier et L. Duchien (LIFL/GOAL) [Sun06].

L'architecture de conteneurs intergiciels évolutive Les supports middleware actuels sont de plus en plus orientés composant, aussi bien en ce qui concerne les applications qu'ils hébergent qu'en ce qui concerne leur conception interne. Les applications ubiquitaires se caractérisent par une diversité importante de contextes d'exécution qui nécessitent des supports techniques variés. Le middleware doit donc pouvoir s'adapter à cette diversité en fournissant aux applications hébergées les services techniques permettant d'exploiter au mieux les ressources du contexte. Parallèlement, les services techniques doivent pouvoir être adaptés afin de répondre efficacement aux variations du contexte. Le but de cette étude est de définir un support intergiciel modulaire à base de composants, permettant d'accueillir et de gérer les services techniques et leur évolution afin de satisfaire les contraintes des applications embarquées. Cette étude donne lieu à la thèse de Ales Psek (financement INRIA CORDI) encadrée par L. Seinturier et P. Merle (LIFL/GOAL) [PSM07].

7.3 Logiciels développés

- L'outillage dans l'atelier de modélisation Objecteering du profil UML2 pour les applications MOSAIQUES.
- GoTM, <http://gotm.objectweb.org>, Logiciel libre licence LGPL
- OpenCCM/Mosaiques, <http://openccm.objectweb.org>, Logiciel libre licence LGPL
- Fractal Deployment Framework (FDF) <http://fdf.gforge.inria.fr>, Logiciel libre licence LGPL
- Divers prototypes DACAR non encore diffusés

- Spoon Graffiti, <http://spoon.gforge.inria.fr/>

7.4 Thèses et HDR soutenues et en cours

- Thèse d'Olivier Barais, *Construire et Maîtriser l'évolution d'une architecture logicielle à base de composants*, soutenue le 30 novembre 2005, dirigée par Laurence Duchien,
- Thèse d'Alexis Muller, *Construction de systèmes par application de modèles paramétrés*, soutenue le 26 Juin 2006, dirigée par Jean-Marc Geib, Olivier Caron et Bernard Carré,
- Thèse de Romain Rouvoy, *Une démarche à granularité extrêmement fine pour la construction de canevas intergiciels hautement adaptables : application aux services de transactions*, soutenue le 8 décembre 2006, dirigée par Jean-Marc Geib et Philippe Merle,
- Thèse de Carlos Noguera, *Transformation de programmes pour applications ubiquitaires*, à soutenir en septembre 2008, dirigée par Laurence Duchien et Renaud Pawlak,
- Thèse de Jérémy Dubus, *L'environnement DACAR pour le déploiement autonome d'applications en environnement distribué et ouvert*, à soutenir en septembre 2008, dirigée par Jean-Marc Geib et Philippe Merle, bourse ministère.
- Thèse d'Ales Plsek. *Architecture de conteneurs intergiciels évolutive pour les systèmes distribués embarqués*, à soutenir en juin 2009, dirigée par Lionel Seinturier et Philippe Merle,

7.5 Retombées du projet

Les acquis de ce projet nous ont permis de répondre à des appels à projet de type européen ou national. Nous participons aux projets suivants :

- Flex-eWare est un projet ANR TL (2007-09) de type plate-forme exploratoire qui a pour but de concevoir une infrastructure pour les applications embarquées à base de composants logiciels. Ces applications partagent de nombreuses caractéristiques avec le domaine de l'informatique ubiquitaire. Il s'agit de concevoir une chaîne d'outils pour le développement d'applications reconfigurables et extensibles à base de composants. Les technologies logicielles types ciblés par le projet sont par exemple Lw-CCM et Fractal/Think. Les démonstrateurs qui seront réalisés dans ce projet appartiennent aux domaines des systèmes électriques intelligents, de l'avionique, des télécommunications et des MPSoC (Multi-Processor System on Chip). Le projet regroupe Thales (leader), France Telecom, CEA, Teamlog, Trialog, STMicroelectronics, Schneider, ENST, LIP6, INRIA. L. Seinturier et P. Merle (LIFL/GOAL) participent à ce projet.
- Le projet ITEA S4ALL a pour objectif de fournir un environnement orienté service accessible à tout utilisateur (Services for ALL). Ce projet de deux ans a démarré en juillet 2005 et se termine fin juin 2007. Les partenaires sont Alcatel CIT (coordinateur), Bull, Capricode, Fraunhofer Fokus, HIIT, INRIA, Instituto de Telecomunicações, INT, mCENTRIC, Nokia, PT Inovação, Schneider Electric, Thales, Université Joseph Fourier, Universidad Politécnica de Madrid, University of Kassel, Vodafone et Xquark/Odonata.
- Le projet ANR TL JOnES a pour objectif de concevoir et réaliser une plate-forme répartie orientée service conforme à la spécification Java Business Integration (JBI). Ce projet de deux ans a démarré en janvier 2006 et se terminera en décembre 2007. Les partenaires sont INRIA (coordinateur), EBM WebSourcing, ENSTIMAC, France

Telecom R&D, Open Wide et ScalAgent Distributed Technologies.

- Le projet ANR TL SCOrWare a pour objectif de concevoir et réaliser une plate-forme orientée service conforme à la spécification Service Component Architecture (SCA). Ce projet de deux ans a démarré en janvier 2007 et se terminera en décembre 2009. Les partenaires sont Amadeus, Artenum, EBM WebSourcing, Edifixio, eXo Platform, INRIA (coordinateur), INT, IRIT, Obeo et Open Wide.
- le projet ANR TL FAROS (2006-2008) de type projet exploratoire a pour objectif de définir un environnement de composition pour la construction fiable d'architectures orientées services. Il complète les travaux sur l'intégration d'applications par la prise en compte d'éléments contractuels permettant une composition cohérente de services ainsi que par la définition d'une méthodologie permettant de rendre reproductibles les procédés d'intégration de contrats depuis des modèles métiers jusqu'à leur projection vers des plateformes d'exécution. Ces éléments contractuels seront de différents niveaux : fonctionnels, extra-fonctionnels, locaux ou globaux. Ils permettront l'élévation du niveau de confiance dans la composition de services. Les applications visées sont de type ubiquitaire. Le projet regroupe France Telecom R &D, EDF, INRIA, LIFL, I3S.

Chapitre 8

USTL- Laboratoire LIFL- Equipe NOCE

8.1 Liste des participants

- Bossaert Philippe, Stagiaire M2Pro e-services, 3 mois plein
- Cybulski Eric, Stagiaire M2Pro e-services, 3 mois plein
- Gruszecki Jean-Philippe, Stagiaire M2Pro e-services, 3 mois plein
- Le Pallec Xavier, Maître de conférences, 50%
- Thomas Vantroys, Maître de conférences, 50%
- Yvan Peter, Maître de conférences, 50%
- Sarra Kadoucci, Stagiaire école d'ingénieur, 4 mois temps plein
- Alain Derycke, Professeur, 20%
- Pierre Caron, Doctorant, 10%

8.2 Problèmes abordés et résultats

L'équipe NOCE a contribué au projet de recherche MOSAIQUES sur les axes modélisation des applications ubiquitaires (axe 1), services pour la mobilité (axe 2) et scénarios de démonstration dans le domaine de l'apprentissage mobile (axe 4). Les contributions réalisées sur ces différents thèmes sont décrites dans la suite.

8.2.1 Modélisation des applications ubiquitaires

Travail 1 Un cycle d'ingénierie dirigée par les modèles pour la conception d'applications ubiquitaires

Problèmes : Un tel cycle signifie amener un modèle représentant la structure abstraite d'une application ubiquitaire vers une réalisation logicielle, ceci au travers une succession de modélisations de plus en plus concrètes. La principale difficulté de ce travail réside dans les phases de traduction de la "dynamique d'adaptation au contexte" de l'application modélisée : pour chaque espace de modélisation ou de réalisation utilisée, il faut conserver les parties implicite (celle exprimée par les concepts de modélisation) et explicite (celle exprimée par le modèle) de cette dynamique.

Travail 2 Guider l'activité de modélisation d'une application ubiquitaire.

Problèmes : La modélisation implique un travail d'abstraction qui n'est pas un exercice simple. Le caractère récent de l'informatique ubiquitaire rend difficile la conception d'applications ubiquitaires. Il est donc très pertinent de proposer une assistance à la modélisation d'applications ubiquitaires. Le problème est double : en quoi peut consister un assistant de modélisation ? Sans expérience de la modélisation ubiquitaire, comment définir de bonnes pratiques associées ?

8.2.2 Résultats

Travail 1 [BCG⁺07b][BCG07a]

Nous avons défini un cycle IDM complet pour le méta-modèle MOSAIQUES commençant par la modélisation "MOSAIQUES" de l'application ubiquitaire. La seconde étape de notre cycle consiste à exprimer le précédent modèle dans une perspective orientée service. Pour cela, nous avons choisi la récente norme SCA (Service Component Architecture). L'expression du modèle en SCA permettra d'utiliser toutes les plateformes compatibles avec cette norme pour implémenter l'application ubiquitaire. Nous avons défini des règles de transformation de modèles MOSAIQUES → SCA. Enfin la troisième et dernière étape du cycle a été de projeter le modèle SCA vers XPDL 2.0, qui peut être considéré comme le méta-modèle standard des systèmes de workflow et qui a l'avantage d'intégrer le concept de services web. Nous avons défini des règles de transformation de modèles SCA → XPDL. Nous utilisons la plateforme COW pour exécuter tous modèles XPDL générés.

Travail 2 [PMM⁺06]

Nous avons collaboré avec l'équipe GOAL pour étudier et appliquer leurs travaux concernant les processus de modélisation incrémentale (PMI). Les PMI visent à guider l'activité de modélisation. Nous avons démarré ces travaux par l'utilisation des PMI pour le méta-modèle standard de la scénarisation pédagogique (IMS-LD) : nous avons défini les *best-practices* d'IMS-LD dans le cadre formel des PMI. Cette première étude nous a permis de maîtriser ce cadre formel et de développer un outil pour le supporter (cf. les logiciels développés). Nous avons donc ensuite appliqué ces résultats au méta-modèle MOSAIQUES, c'est-à-dire à la modélisation ubiquitaire, en définissant un processus de modélisation de type descendant (de macroscopique à microscopique) et un processus de type ascendant (démarrant par les services et remontant au système complet).

8.3 Logiciels développés

Evolutions de ModX ModX est un outil de modélisation développé dans notre laboratoire pour définir tout types de méta-modèles et modèles associés. Une de ses particularités est de pouvoir définir et associer plusieurs formalismes graphiques à un méta-modèle. Pour les travaux cités plus haut, nous avons utilisé ModX comme environnement de modélisation pour définir les méta-modèles MOSAIQUES et SCA, représenter des modèles de type MOSAIQUES, SCA, XPDL et implémenter les règles de transformations entre eux. Pour cela, nous avons dû développer plusieurs fonctionnalités supplémentaires.

- la fonctionnalité graphique de juxtaposition afin de pouvoir représenter les "ports" de composants de la notation SCA.

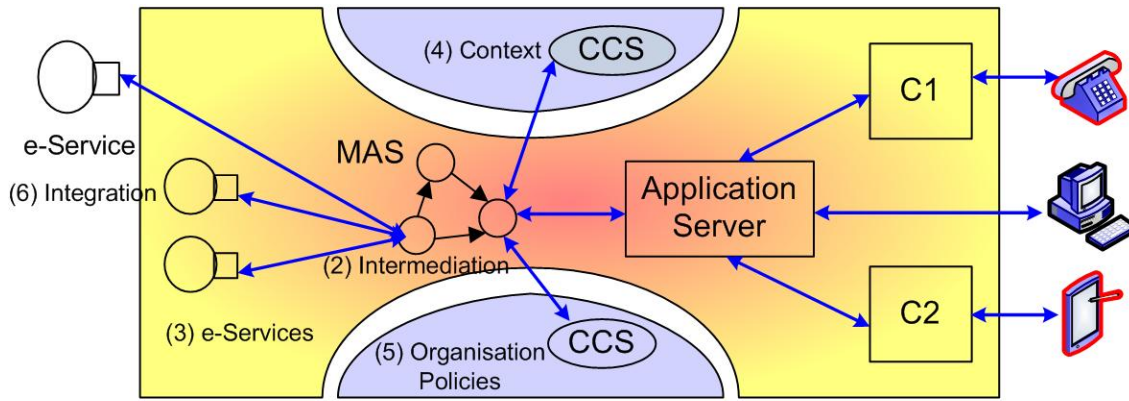


FIG. 8.1 – Architecture d'intermédiation

- l'intégration d'un langage de script basé sur Java au sein de ModX pour faciliter l'écriture des règles de transformation.
- fournir une implémentation du cadre formel des PMI. Cette fonctionnalité de ModX s'appelle "méthodologie".

Services pour la mobilité Dans le cadre de l'axe 2, l'équipe NOCE s'attache à développer une infrastructure d'intermédiation permettant la gestion de services ubiquitaires. Ce travail s'articule autour de l'orchestration des services en fonction du contexte et trouve son pendant dans le projet MIAOU en terme d'interaction homme-machine pour l'utilisation des services à travers différents canaux de communication. La Figure 8.1 présente une vision simplifiée de l'architecture développée avec à droite les différents canaux permettant d'accéder aux services dans différentes situations (traditionnelle, en mobilité) et à gauche les services qui fournissent les fonctionnalités applicatives.

L'architecture d'intermédiation est basée sur un système multi-agents JADE. Les différents agents qui composent le système gèrent :

- Les propriétés des canaux de communication ;
- Le contexte d'utilisation qui peut couvrir des aspects tels que la localisation des utilisateurs ou leurs profils ;
- Les règles organisationnelles qui doivent être appliqués à l'utilisateur ;
- Les propriétés et fonctions des services proposés.

En fonction du canal utilisé, du contexte et des règles organisationnelles, le système pourra proposer des services pertinents aussi bien en terme de support à l'activité en cours qu'en terme de facilité d'usage en fonction des propriétés des canaux utilisés [KVC07].

Scénarios de mobilité En terme de scénarios, l'équipe NOCE s'est concentrée essentiellement sur la définition de scénarios d'apprentissage mobile et le développement de démonstrateurs mettant en situation des services liés à la mobilité.

Ces services portent sur l'utilisation de la localisation et l'authentification des utilisateurs afin de leur fournir des informations et services contextualisés et permettre par exemple

de réaliser une vidéo communication de manière spontanée avec des personnes disponibles dans un bâtiment. Les scénarios et démonstrateurs définis reposent sur la présence d'une infrastructure permettant une communication fiable (Wifi).

Un deuxième scénario est en cours d'expérimentation autour d'un service de blog mobile utilisé à des fins de suivi de projet et de collaboration entre étudiants et avec le tuteur. Dans ce cadre, nous explorons les différents canaux susceptibles de supporter cette activité (web, SMS/MMS et voix) [LLY⁺07].

8.4 Retombées du projet

Les contributions de l'équipe NOCE ont des retombées qui ont un intérêt pour la région dans le cadre de la recherche aussi bien que de l'enseignement. En terme de recherche les scénarios mis en place autour du blog mobile font l'objet d'une collaboration avec l'Université de Picardie Jules Verne et le laboratoire d'informatique du Littoral afin d'expérimenter le dispositif, en lien avec le projet EUCUE du présent contrat de plan état-région.

D'autre part l'équipe NOCE a proposé et obtenu un financement lors de l'appel à projet ANR Télécoms 2006 sur le thème de l'apprentissage pervasif (projet p-LearNet). Ce projet a été labellisé par le pôle de compétitivité "industries du Commerce" et implique les sociétés Auchan et La Poste. Dans le cadre de ce projet des propositions seront faites en terme d'architecture et de service pour l'apprentissage pervasif. En parallèle, un volet expérimentation et usages permettra de valider ces propositions. Enfin suite aux travaux menés dans le cadre de MOSAIQUES, une nouvelle option de la licence professionnelle Réseaux et Télécommunications a été habilitée visant à former des professionnels compétents dans le domaine du développement d'applications mobiles.

Chapitre 9

USTL- Laboratoire LIFL- Equipe RD2P

9.1 Liste des participants

- Gilles Grimaud, MCF IEEA 30%
- David Simplot-Ryl, PR IEEA 10%
- Alexandre Courbot, doctorant 100% (Septembre 2004-Septembre 2006)
- Leman Florian, stagiaire
- Deblonde J-Philippe, stagiaire
- Benony Vincent, ingénieur, (mars 2007-Mai 2007)

9.2 Problèmes abordés et résultats

L'équipe RD2P a principalement participé aux actions de l'Axe 2 « Infrastructure logicielle avec des propriétés d'adaptabilité dynamique ».

9.2.1 Contexte des travaux

Nous avons basé nos travaux sur les infrastructures logicielles utilisées pour supporter l'exécution des applications mobiles. Dans ce cadre, nous nous sommes intéressés à la logique de déploiement des applications Java. Pour ce type de logiciels, différents cycles de vie applicatifs bien définis sont largement documentés et communément acceptés par l'industrie. Nous avons proposé un modèle d'infrastructure de déploiement et un ensemble de méthodes d'analyses appropriées au contexte ubiquitaire. Sur cette base, nous avons pu proposer une stratégie générique pour supporter les différentes logiques de déploiement. Sur cette base nous avons proposé un ensemble d'analyses permettant l'adaptation automatique des applications et de leurs supports aux environnements les plus contraignants de l'informatique ubiquitaire.

9.2.2 Logique de déploiement

Dans le cadre de l'axe 2, nous avons étudié les différentes infrastructures existantes offrant des mécanismes pour faciliter les déploiements d'applications sur des supports ubiquitaires. Parmi les infrastructures les plus pertinentes nous avons identifié, le modèle des Bundles

(OSGi), le modèle des Servlets (J2EE), et le modèle des Cardlets (JavaCard). L'ensemble de ces modèles reposent sur les mécanismes de chargement et de liaison dynamique fournis par l'environnement d'exécution de Java. Cependant ce sont ces mêmes mécanismes qui constituent l'essentiel de la complexité des supports d'exécutions Java, et qui rendent impossible le déploiement des applications Java au sein des infrastructures embarquées. Nous avons proposé un support d'exécution particulier qui nous permet de réaliser un « pré-déploiement » des applications quelle que soit la logique de déploiement (Bundle, Servlet, Cardlet) sur laquelle elles reposent. Nous avons montrés dans [RCG05, MGS05] que sur cette base, il était possible d'adapter les applications aux matériels ubiquitaires les plus contraints.

9.2.3 Outils de spécialisation

Le mécanisme de pré-déploiement que nous avons proposé, permet de réaliser les opérations liées à la logique de déploiement des applications en amont de l'exécution de ces mêmes applications. Selon le modèle de pré-déploiement que nous préconisons, les applications ne sont transférées sur la cible où elles seront exécutées qu'une fois que leur déploiement est achevé. Dans ce contexte, il devient possible de spécialiser l'application après son déploiement et avant son transfert vers la cible réelle. Cette spécialisation « à chaud » a largement été discutée dans le cadre du projet MOSAIQUES et elle a donné lieu à différentes publications scientifiques dont [CGV05, MCG05, ?]. En effet, nous avons montré que des analyses appropriées permettaient de spécialiser automatiquement les logiciels déployés vers des cibles contraintes, et qu'elles permettaient des rendements bien meilleurs que ceux usuellement utilisés, en tirant partie des informations issues de la logique de déploiement déjà achevée au moment de leurs mise en oeuvre.

9.2.4 Support d'exécution minimaliste

Enfin une fois les méthodes d'analyses que nous avons proposées sont mises en oeuvre, elles peuvent non seulement servir à spécialiser les applications déployées, mais aussi à montrer comment elles peuvent être utilisées pour spécialiser le support d'exécution ubiquitaire lui-même. Dans le cadre du projet MOSAIQUES, nous avons eu l'opportunité de réaliser un outil de spécialisation de la machine virtuelle Java. Les résultats probants que nous avons obtenus ont été publiés dans [ACSR06]. Ils démontrent que la spécialisation « à chaud » issue du modèle de pré déploiement que nous avons proposée est particulièrement profitable lorsqu'elle est appliquée à la spécialisation des machines virtuelles.

9.3 Logiciels développés

Les travaux promus dans le cadre du projet MOSAIQUES se sont traduits par des réalisations logicielles conséquentes. Ces réalisations regroupées autour de la plateforme JITS sont aujourd'hui exploitées au sein du consortium associé au projet ANR « SVP » (SurVeiller et Protéger). Ils sont accessibles (sous réserve d'agrément du consortium) par le biais de la forge logiciel de l'INRIA à l'URL <http://jits.gforge.inria.fr>.

9.4 Thèses et HDR soutenues et en cours

- Thèse d’Alexandre Courbot, « spécialisation tardive de systèmes Java embarqués pour petits objets portables et sécurisés » sous la direction de David Simplot-Ryl et Gilles Grimaud (Financement INRIA-Région), soutenue le 20 septembre 2006.
- Thèse de Yann Hodique, « Sûreté et optimisation par les systèmes de types en contexte ouvert et contraint », sous la direction d’Isabelle Simplot-Ryl et Gilles Grimaud (allocation couplée de l’ENS Cachan), soutenue le 13 avril 2007.

9.5 Retombées du projet

Dans la continuité des travaux initiés par le projet MOSAIQUES, l’équipe RD2P contribue actuellement aux projets ANR SVP et Mesure.

- SVP est un projet, du programme du Réseau National de la Recherche en Télé-communications, labellisé par l’ANR, qui propose l’étude, la réalisation et l’expérimentation d’une architecture ambiante intégrée pour faciliter la conception, le déploiement et l’exploitation optimale de services de surveillance et de prévention sur différents types de réseaux dynamiques. Les partenaires de ce projet sont Thales, le CEA, l’INRIA, le LIP6, Aphy-Care, Anact et U2S.
- Mesure est un projet précompétitif, du programme du Réseau National de la recherche en Technologies Logicielles, labellisé par l’ANR, qui propose l’étude, la réalisation et l’expérimentation d’un jeu de benchmarks pour quantifier les performances des différentes infrastructures logicielles proposées par l’industrie pour supporter le déploiement et l’adaptation des applications aux sein des cartes à microprocesseurs. Ce projet s’inscrit dans la continuité des études initiées par MOSAIQUES autour des infrastructures logicielles pour l’informatique ubiquitaire. Les partenaires sont : Le CEDRIC (CNAM), le LIFL (Université des Sciences et Technologies de Lille) et Trusted Labs (Sophia-Antipolis).

Chapitre 10

USTL-Laboratoire LIFL-Equipe SMAC

10.1 Liste des participants

- Philippe Mathieu (PU, IUT "A"), 30 %.
- Bruno Beaufls (MCF, IUT "A"), 30 %.
- Sébastien Picault (MCF, IUT "A"), 30 %.
- Jean-Christophe Routier (MCF HDR, IEEA), 30 %.
- Yann Secq (MCF, IUT "A"), 50 %.
- Maxime Morge (PostDoc), 50 %.
- Tony Dujardin (Stagiaire)
- Benjamin Noteau (Stagiaire)
- Joffrey Woets (Stagiaire)
- Mathieu Braure (Stagiaire)

10.2 Les objectifs et problèmes abordés

L'objectif du projet MOSAIQUES était l'étude des impacts produits lors de la conception de logiciels devant s'exécuter dans un contexte ubiquitaire, c'est-à-dire un environnement dans lequel les applications sont physiquement réparties sur des périphériques aux capacités très variables (en terme de calcul, de mémoire, de connexion réseau ou encore d'interaction homme machine). Dans ce contexte, la conception de logiciels est profondément modifiée car il n'est plus possible de supposer qu'un seul organisme développe l'ensemble des applications proposées à l'utilisateur. D'autre part, il est nécessaire lors de la création et surtout lors de l'exécution de l'application de prendre en compte les différences d'environnement d'exécution. Il est alors indispensable de représenter ces contextes d'exécution pour que les applications puissent s'adapter aux capacités disponibles à un instant donné.

L'équipe SMAC étudie depuis sa création la conception de système multi-agents distribués¹ et s'est intéressée depuis plusieurs années à la conception incrémentale de systèmes

¹**Dynamic Skill Learning : A Support to Agent Evolution**, *Jean-Christophe Routier and Philippe Mathieu and Yann Secq*, Proceedings of the Artificial Intelligence and the Simulation of Behaviour symposium on Adaptive Agents and Multi-agent systems (AISB'01), pp. 25–32, 2001

ubiquitaires². Plus récemment, l'émergence de nombreuses plateformes nous a mené à étudier plus particulièrement les problèmes d'interopérabilité³. Ces problèmes surgissent lorsque des systèmes conçus par différentes organisations doivent interagir. La solution habituelle consiste à définir des composants ad'hoc permettant d'établir un pont entre les deux systèmes. Cette solution n'est pas satisfaisante et totalement impraticable lorsque le nombre de systèmes augmente.

10.3 Les propositions effectuées et les résultats du projet

Pour illustrer ces problématiques et les solutions que nous y apportons, nous avons effectué deux propositions fortes dans le projet MOSAIQUES. La première proposition concernait un scénario "Foire de Paris" permettant d'illustrer les objectifs poursuivis en terme d'applications ubiquitaires. La seconde se positionnait en terme de méthode de conception de systèmes ubiquitaires et en terme de mécanismes d'interaction dans le but de découvrir, sélectionner et composer des services.

Dans le cas d'étude "Foire de Paris", un utilisateur vient à Paris pour se rendre à la Foire de Paris. Il dispose d'un périphérique (PDA, téléphone ...) qui héberge son assistant, un agent logiciel servant d'interface avec les services "ubiquitaires". Voici un exemple d'interaction pouvant se produire entre l'assistant de l'utilisateur et un agent médiateur pour les services de transport :

```
Assistant : "Quel service de transport puis-je utiliser pour me
rendre a la Foire de Paris?
AgST : Le taxi est un service de transport qui permet de se
rendre a la Foire de Paris.
Assistant : Pourquoi peut-il etre qualifie de service de
transport pour la Foire de Paris ?
AgST : Il permet de se rendre a Paris
Assistant : Selon moi, ce service de transport doit me permettre
de me rendre place de la porte de Versailles.
AgST : Selon moi, ce service de transport ne doit pas permettre
de se rendre place de la porte de Versailles mais avec un taxi
vous le pouvez.
Assistant : OK, je vais considerer les services de taxi".
```

Dans cet exemple, les deux agents ne partagent pas les mêmes représentations concernant les services de transport. Dans un environnement multi-fournisseurs, ce type de situation se produit souvent, il est donc nécessaire de proposer des mécanismes résolvant (dans une certaine mesure) ces incompréhensions.

Pour cela, nous avons proposé d'utiliser la notion d'ontologie, pour représenter les informations échangées par les agents, conjointement à un système dialogique, pour résoudre les conflits de représentation et effectuer la découverte et la sélection de services.

²**Ubiquitous Computing : vanishing the notion of application**, *Philippe Mathieu and Jean-Christophe Routier and Yann Secq*, Workshop on Ubiquitous Agents on embedded, wearable, and mobile devices (AAMAS'02), 2002

³**Bridging the gap between semantic and pragmatic**, *Philippe Mathieu and Jean-Christophe Routier and Yann Secq*, Proceedings of The 2003 International Conference on Information and Knowledge Engineering (IKE'03), pp. 308-314, 2003

Une ontologie définie formellement un vocabulaire commun pour partager l'information d'un domaine entre plusieurs agents. Une ontologie repose sur des définitions de concept et de leurs relations, interprétables par des machines. Le formalisme que nous utilisons est une logique de description ALC. Ce langage correspond au niveau intermédiaire utilisé par le W3C dans OWL (OWL-DL). L'utilisation de cette logique apporte des automatisations sur la vérification de la cohérence de la description ainsi que des capacités de classification automatique des ressources.

L'ontologie intervient dans la caractérisation des services et des ressources présentes dans le système. Comme il paraît peu probable qu'une ontologie unique s'impose (le Graal poursuivit par le projet CYC ...), il est nécessaire de traiter cette question d'interopérabilité des ontologies.

Dans notre solution nous n'avons pas créé de ponts entre les ontologies, mais nous avons utilisé des techniques d'argumentation. Un mécanisme dialogique permet aux agents d'aboutir à un consensus sur leurs représentations et permet la sélection de services répondant aux spécifications de l'utilisateur. Notre approche, DIALRAR, est un cadre formel dans lequel les agents argumentent pour s'accorder sur une représentation.

Dans ce système, un dialogue est une séquence cohérente de coups qui a pour but de faire évoluer une situation initiale pour atteindre les buts des participants. En l'occurrence, le but des dialogues consiste en la résolution d'une requête pour sélectionner un service ou découvrir les capacités d'un service. Dans la situation initiale, les deux agents impliqués ne partagent pas la même définition d'un objet, soit parce que l'un d'eux est ignorant, soit parce que leurs définitions ne proposent pas le même point de vue. Au terme du dialogue qui les confronte, les agents doivent atteindre un accord qui concerne la définition de cet objet ce qui permet de sélectionner ou non le service comme correspondant à la requête du client.

Bien que les étapes de découvertes et de sélection d'un service soient fondamentales, la composition d'un ensemble de services est d'autant plus importante. L'objectif ultime est d'avoir un enchaînement automatique d'un ensemble de services découverts dynamiquement pour résoudre une requête de l'utilisateur. Pour parvenir à cet objectif, il serait nécessaire de pouvoir représenter la sémantique d'un service et de son utilisation. Cet objectif étant très ambitieux, nous avons travaillé sur deux étapes nécessaires et préalables : un mécanisme semi-automatisé de composition de services et un système de négociation sur des planifications de composition de services.

Nous avons proposé une composition automatisée de services par fusion d'information et chaînage de services. La phase de fusion d'information consiste à intégrer les informations publiées par les services, afin de pouvoir raisonner sur plusieurs sources d'information simultanément. La phase de chaînage de services propose un enchaînement de services en se basant sur les informations consommées et produites par les services.

Pour que la phase de fusion soit possible, nous avons proposé un modèle de conception d'ontologie prescrivant une structuration des ontologies selon quatre niveaux :

- Le niveau I regroupe les connaissances les plus abstraites, partagées par l'ensemble des agents du système. Dans notre exemple, nous décrivons à ce niveau les concepts de **Graphe**, **Noeud** et **Arc** qui sont utilisables dans des domaines qui peuvent être très différents du transport.
- Le niveau II contient les connaissances du domaine, ces connaissances spécialisent des concepts de niveau I. Dans notre exemple nous décrivons à ce niveau un **Reseau** qui est un ensemble de **Point** (Noeud géographiquement situé par des coordonnées GPS) et des **Etapas** reliant les **Points**.

- Le niveau III concerne les connaissances spécifiques au service mais explicitées avec les connaissances communes. Ainsi le médiateur va pouvoir comprendre les informations (**Stations**, **Arrets**) à l'aide du niveau II (**Point**) qu'il partage avec les autres agents de domaine proche.
- Le niveau IV est dédié aux connaissances propres au service, n'ayant pas à être partagées avec les autres agents. Par exemple les noms des chauffeurs de bus ou de métro.

Une fois les ontologies définies, il est possible de caractériser les services. Chaque service possède deux parties : la description des opérations précise pour chaque opération du service le nom de l'opération et à l'aide des concepts du niveau I, II ou III, les informations consommées (ou requises) et produites par l'opération. Pour l'exemple des services de transport, trois opérations sont définies : **simuler** qui fournit un **Trajet** construit par le service de transport à partir de son réseau pour relier les deux **Points** fournis. L'opération **evaluerCout** va construire un **BilletFactice**, c'est-à-dire un devis sur le coût pour effectuer le **Trajet** fourni. L'opération **acheter** sera effectuée par l'assistant de notre utilisateur afin d'obtenir son **Billet** de transport à partir du **BilletFactice** fourni par le médiateur. Les informations publiques sont les connaissances que fournit le service afin d'être utilisé par un utilisateur et de permettre des raisonnements. Dans l'exemple de transport pour "la Foire de Paris", les services de **Bus** et de **Metro** fournissent un réseau d'arrêts ou stations (**Points** géographiquement situés) reliés par les routes pour le bus ou les voies de métro, ce qui équivalent aux informations reprises par notre utilisateur avec les annotations de localisation.

Ces travaux ont fait l'objet de publications avec comité de lecture : dans une conférence nationale JFSMA'06[MRSD06a], un workshop international FOCA'06[MRSD06b], une conférence internationale BNAIC'06[MRS06] et une revue internationale JOO'07[MR07] (à paraître). Ainsi, dans l'article publié aux JFSMA'06, nous avons présenté un dispositif formel au travers duquel deux agents dialoguent pour aboutir à un consensus ontologique. Nous avons proposé un modèle de représentation argumentatif qui permet de gérer les conflits entre des descriptions ayant des pertinences différentes selon l'agent qui les évalue. Nous avons également défini un modèle de raisonnement d'agents dans lequel chaque agent justifie la définition sur laquelle il s'engage en prenant en compte les définitions de ses interlocuteurs. Dans ce dispositif, deux agents dialoguent pour aboutir à un accord malgré leur représentation conflictuelle.

La deuxième proposition forte, que nous avons faite dans le cadre du projet MOSAIQUES, concernait un système de négociation sur des planifications de composition de services. Pour traiter cette problématique, nous avons supposé que les services étaient décrits à l'aide du langage ADL (Action Description Language) qui est une extension du langage STRIPS (Stanford Research Institute Problem Solver). Chaque action est définie en termes de pré et post conditions. Une composition de service correspond alors à une instance de plan, c'est-à-dire une séquence d'actions dont l'évaluation mène de l'état actuel à un état satisfaisant les objectifs définis par l'agent.

Nous nous plaçons alors dans le contexte où un agent **AcheteurClient** effectue auprès d'un ou plusieurs agents **VendeurFournisseur** de services une requête pour un service particulier. Pour satisfaire à cette requête ceux-ci doivent proposer à l'acheteur un plan. Ce plan doit être accepté par l'acheteur, c'est donc aux vendeurs de le convaincre de sa validité. Cette fois encore un mécanisme dialogique délibératif est utilisé. Les arguments que s'échangent les différentes parties correspondent aux descriptions des services, ces descriptions doivent être justifiées et les agents se construisent donc une représentation commune au cours du dialogue. Les différents arguments sont évalués par l'acheteur ce qui lui permet, une fois la

validité d'un plan établi, d'en établir la force. Parmi les différents plans proposés et qui sont donc en conflit, l'acheteur peut donc faire le choix de celui qui lui paraît le mieux adapté à ses besoins et préférences.

Ces travaux ont fait l'objet du mémoire de Master Recherche de Mr Dujardin et d'une publication dans une conférence nationale JFPDA'06[MR06]. Dans cette communication, nous avons proposé le modèle de dialogue entre agents DIALP (DIALP Is an Argumentative Labour for Planification), qui permet de formaliser un processus délibératif. Ce modèle circonscrit un dispositif formel au travers duquel les agents jouent et arbitrent pour atteindre un accord pratique. A cette intention, nous avons proposé un modèle de raisonnement argumentatif qui permet de gérer les conflits entre des plans ayant des forces différentes selon l'agent qui les évalue. Nous avons également défini un modèle de justification des plans produits qui prennent en compte les engagements des interlocuteurs. Dans ce modèle, la décision pratique finale est prise par un agent tiers en fonction de l'autorité de chacun des joueurs et des plans avancés.

10.4 Thèses et HDR soutenues et en cours

- Jean-Christophe Routier, HDR soutenue en décembre 2006.

Chapitre 11

USTL-Laboratoire LIFL- Equipe STC

11.1 Liste des participants

- Mireille Clerbout (PR) 100%
- Isabelle Simplot-Ryl (MCF,IUT) 50%
- Mirabelle Nebut (MCF) 100 %
- Arnaud Bailly, doctorant, 50% (septembre 2004 – décembre 2005)
- Dorina Ghindici, doctorante 50% (septembre 2005 – mai 2007)
- Yann Hodique, doctorant 40% (septembre 2004 – mai 2007)
- Issa Traoré, Associate professor, University of Victoria, Canada, invité en avril 2006.

11.2 Problèmes abordés et résultats

L'équipe STC a participé à différentes actions dans les travaux des axes 1 et 2.

11.2.1 Cadre de développement

Dans le cadre de l'axe 1 du projet, nous avons proposé un cadre de développement pour les applications ubiquitaires et les plates-formes d'exécution qui les prennent en charge. La seconde contribution dans ce domaine a porté principalement sur la formalisation des spécifications. Nous avons proposé d'utiliser pour la spécification du contexte des types booléens et énumérés pour permettre l'application ultérieure de techniques de validation. Nous avons aussi proposé l'utilisation d'expressions logiques (clauses de Horn) pour spécifier les contraintes d'utilisation d'une application en fonction du contexte. Enfin nous avons contribué à la définition du méta-modèle de la dynamique d'une application ubiquitaire [CCG⁺06].

11.2.2 Composition et extensibilité

Les applications ubiquitaires modernes sont distribuées dans des contextes hétérogènes et doivent évoluer sans compromettre leur disponibilité et sans possibilité de mise à jour synchronisée. Dans ce contexte, garantir la sûreté de fonctionnement des applications peut s'avérer délicat. Nous avons étudié la composition des composants et obtenu des propriétés

intéressantes qui garantissent la compositionnalité du modèle dans [BCSR06a, BCSR06b]. Nous avons également étudié dans [JOSR05] un mécanisme de mise à jour dynamique de classes qui permet de mettre à jour une hiérarchie de classes de telle manière que les objets existants, comme les nouveaux, des classes mises à jour et de leur sous-classes soient mis à jour graduellement durant l'exécution, sans arrêter un système en cours d'exécution, et sans nécessiter de synchronisation.

11.2.3 Autonomie et sécurité des systèmes

Les systèmes ubiquitaires reposent sur l'utilisation de nombreux petits objets tels les cartes à puce ou les PDA qui s'intègrent dans les systèmes d'information. Dans ce contexte, différentes optimisations des applications sont nécessaires pour qu'elles puissent être embarquées dans ces petits systèmes. Cependant, ces différentes interventions ne doivent pas se faire au détriment des propriétés de sécurité des systèmes. Nous avons donc proposé des techniques qui permettent à un système embarqué chargeant du code mobile de vérifier l'innocuité du code optimisé. Dans [GHSR05], nous avons présenté un système de types extensible pour langage intermédiaire qui unifie dans une hiérarchie unique différents systèmes de types de langages sources variés et assure la confidentialité et l'intégrité. Pour augmenter l'efficacité et la flexibilité du système, nous proposons un mécanisme de liaison dynamique sûr qui permet un grand nombre d'optimisations. Dans [GHSR06], nous discutons de l'utilité et de l'applicabilité de l'analyse d'échappement (une technique d'optimisation mémoire utilisée pour les langages objets) dans le cas des petits systèmes embarqués utilisés dans l'informatique ubiquitaire. Nous montrons dans [GHSR07], que l'utilisation de cette technique et son extension à des patrons de conception telles que les fabriques d'objets, permet d'utiliser les mêmes techniques de génie logiciel que pour d'autres systèmes cibles et d'utiliser a posteriori des optimisations sûres pour porter les applications sur des systèmes embarqués de petite taille. Ceci est important pour le développement d'applications ubiquitaires qui doivent être adaptées à des contextes hétérogènes.

11.2.4 Sécurité des données

L'informatique pervasive pose également le problème de la sécurité et de la confidentialité des données des utilisateurs mais aussi des différents fournisseurs de services, éventuellement concurrents. Les techniques liées à l'analyse de flot d'information sont étudiées depuis longtemps pour assurer la confidentialité des données dans le contexte de l'informatique « classique ». Nous nous sommes intéressés à l'utilisation de ces techniques dans le cadre d'applications ouvertes, embarquées et extensibles qui chargent du code mobile dans [GGSR06a, GGSR07, GGSR⁺06b].

11.2.5 Coopération des systèmes autonomes

Les systèmes ubiquitaires ont la particularité d'intégrer des plateformes fortement hétérogènes et des contextes d'utilisation variés. Nous nous sommes intéressés, toujours dans le contexte de la sûreté de fonctionnement, à la manière d'assurer les services du réseau lorsque le contexte d'utilisation ne permet pas l'utilisation d'infrastructures et nécessite la coopération de tous les usagers. Ceci a donné lieu aux publications [HSR06, HSR07].

11.3 Logiciels développés

Un prototype d'outil d'analyse statique vérifiable et embarquable a été développé (STAN pour STatic Alias aNalyser), il est actuellement disponible en ligne pour évaluation par la communauté.

11.4 Thèses et HDR soutenues et en cours

- Thèse d'Arnaud Bailly, « *Test et Validation de Composants Logiciels* » sous la direction de Mireille Clerbout et Isabelle Simplot-Ryl (Financement Région/Entreprise, Norsys), soutenue le 15 décembre 2005.
- Thèse de Yann Hodique, « *Sûreté et optimisation par les systèmes de types en contexte ouvert et contraint* », sous la direction de Isabelle Simplot-Ryl et Gilles Grimaud (allocation couplée de l'ENS Cachan), soutenue le 13 avril 2007.
- Thèse de Dorina Ghindici, « *Validation de mécanismes de sécurité dans les systèmes embarqués* », (Financement bourse ministère) sous la direction de Isabelle Simplot-Ryl, commencée en septembre 2005.
- HDR d'Isabelle Simplot-Ryl, « *Sûreté de fonctionnement en contexte objet, distribué et extensible* », sous la direction de Mireille Clerbout, soutenue le 4 décembre 2006.

11.5 Retombées du projet

- Une réponse à l'appel à projet 2007 ANR-07-SESUR (Sécurité et Sûreté Informatique) a été déposée en mars 2007 est actuellement en cours d'évaluation. Le projet SFINCS (Securing Flow of INformation for Computing pervasive Systems) s'inscrit dans la continuité des travaux sur la sécurité des informations en contexte ubiquitaire initiés dans Mosaïques et regroupe les partenaires : LIF (Université de Provence), LIFL (Université des Sciences et Technologies de Lille), Norsys (Enneveulin), SI3SI (Systèmes d'Information 3 Suisses International), Trusted Logic (Versailles) et VERIMAG (CNRS Grenoble).

Chapitre 12

UVHC- Laboratoire LAMIH- Equipe ROI

12.1 Liste des participants

- Sylvain LECOMTE, Professeur, 50%
- Vincent POIRRIEZ, Maître de conférences, 50%
- Dorian PETIT, Maître de conférences, 40%
- Thierry DELOT, Maître de conférences, 30%
- Marie THILLIEZ, Maître de conférences, 30%
- Nadia BENNANI, Maître de conférences, 20%
- Hocine GRINE, Stagiaire, 12 mois
- Guy KANDEM, Stagiaire, 12 mois
- Colombe Hérault, Doctorante, 50%
- Samuel Colin, Doctorante, 50%
- Hocine Grine, Doctorant, 50%

12.2 Objectifs et problèmes abordés

Dans le cadre du LAMIH, nous nous sommes attachés à atteindre plusieurs objectifs :

- définir une architecture d'adaptation pour services techniques ou applicatifs,
- définir des critères de garantie sur la qualité des assemblages de composants formant les services,
- et enfin, appliquer le tout à un service d'évaluation de requêtes en environnement ubiquitaire.

12.3 Les résultats

Les résultats sont de plusieurs ordres. Tout d'abord, dans le cadre de l'axe 1, plusieurs travaux de modélisation ont été effectués par quasiment l'ensemble des partenaires. Ces travaux ont été présentés lors de la conférence UbiMob'06 [CCG⁺06]. Dans ce cadre, il a été fourni plusieurs modélisations :

- la modélisation d'une application MOSAIQUES,

- la modélisation de son environnement,
- la modélisation de la dynamique d'exécution,
- un exemple d'application basé sur ces modèles.

Dans ce cadre, le LAMIH s'est particulièrement intéressé à la réalisation d'un évaluateur de requêtes adaptable pour environnements ubiquitaires. La spécification et le fonctionnement de cet évaluateur respectent les différentes modélisations obtenues. Ce travail a également fait l'objet de plusieurs publications [GDL05, HG07, HG06, GHL05] et de la réalisation d'un prototype (soumis à publication dans le cadre des démonstrations de la conférence BDA'07).

Dans le cadre de l'axe 2, le LAMIH a travaillé sur le développement d'un mécanisme de déploiement pour application à base de composants. Ce travail, en cours, a abouti à une soutenance de Master recherche en 2006. Enfin, des travaux ont débuté pour enrichir le langage d'assemblage des applications à base de composants, dans le but de vérifier un certain nombre de propriétés formelles et de garantir le bon fonctionnement de ces applications. Ce travail a été publié [CPP⁺05] et va se poursuivre avec des partenaires nationaux dans le cadre du projet ANR SETIN'06 TACOS.

Au cours de ces travaux, plusieurs difficultés ont été rencontrées. Une des principales difficultés rencontrées concerne les travaux liés au contexte et à la définition de celui ci pour la définition de la dynamique de l'adaptation. Encore aujourd'hui, la prise en compte de ce contexte pose problème, notamment au niveau de la définition des règles d'adaptation qui en résultent : des éléments différents du contexte peuvent déclencher des actions d'adaptation opposées l'une à l'autre (notamment dans la configuration de l'optimisateur de notre évaluateur de requêtes).

Une autre difficulté importante non encore surmontée est relative à la formalisation des propriétés d'adaptation, cette formalisation est nécessaire pour permettre de valider ces spécifications mais elle requière de pouvoir exprimer des propriétés non fonctionnelles de composants pour en déduire celles d'assemblages de composants adaptables.

12.4 Logiciels développés

La principale réalisation est un évaluateur de requêtes adaptable basé sur le modèle à composant Fractal (soumis à publication dans le cadre des démonstrations de la conférence BDA'07)

12.5 Thèses et HDR soutenues et en cours

- HDR de LECOMTE S. (2005). Conception et adaptation de services techniques pour l'informatique ubiquitaire et nomade. Mémoire d'HDR, LAMIH, UVHC, décembre.
- HDR de POIRRIEZ V. (2006). Contributions à la compréhension de problèmes d'optimisation combinatoire et études d'extensions de la méthode B.. Mémoire d'HDR, "Institut des Sciences et Techniques de Valenciennes", "Université De Valenciennes Et Du Hainaut-Cambrésis", décembre.
- Thèse de Colombe Hérault, "Adaptabilité des services techniques dans le modèle à composants", LAMIH, université de Valenciennes et du Hainaut Cambrésis, soutenue le 23 juin 2005
- Thèse de Samuel Colin, "Contribution à l'intégration de temporalité au formalisme B : Utilisation du calcul des durées en tant que sémantique temporelle pour B " soutenue

le 03 octobre 2006

- Thèse de Hocine Grine, « Evaluateur de requêtes adaptable pour environnements ubiquitaires », en cours.

12.6 Retombées du projet

Grâce à MOSAIQUES, l'équipe a tissé des liens importants au niveau national et international :

- Plusieurs projets ANR ont été soumis avec différents partenaires (LORIA, LIG, IRIT, etc). Actuellement, un de ses projets a été accepté et a débuté en septembre 2006 (<http://tacos.loria.fr>)
- Un atelier de travail a été monté en collaboration avec Florence SEDES dans le cadre de la conférence INFORSID (atelier GEDSIP : Gestion de données dans les systèmes d'information pervasifs)
- Au niveau international, des liens ont pu être tissés avec l'université de Saragosse en Espagne (ce qui a abouti à la venue de Sergio Harri comme professeur invité en juin 2007)

Chapitre 13

Les retombées du programme de recherche et les perspectives

13.1 Les retombées du programme

13.1.1 Retombées sur le plan scientifique

L'avenir de la conception des nouveaux terminaux informatique (PDA, téléphone) ainsi que des logiciels afférents pour les nouvelles applications des réseaux étendus et des télécommunications sera très différent de ce qui existe actuellement. Tous les acteurs en sont persuadés. L'industrie du logiciel est en train de devenir une industrie du composant logiciel par la même révolution que le matériel il y a 20 ans. Les futures applications seront fabriquées par génération et/ou par assemblage de composants trouvés sur des étagères virtuelles multi-fournisseurs, puis déployés chez des prestataires de services et chez les utilisateurs. Dans ce cadre, le besoin de pouvoir facilement adapter dynamiquement ces composants pour qu'ils correspondent aux besoins des terminaux sur lesquels ils vont s'exécuter est vital.

Ce projet est fondé sur des concepts et technologies qui sont à leur premiers babutiements. C'est en partie pour cela que les équipes impliquées dans le projet MOSAIQUES ont participé à cette évolution scientifique et ont apporté des éléments tels que :

- la prise en compte de l'adaptabilité dans la modélisation par composants des applications logicielles réparties avec une mise en avant d'une démarche et une expérimentation dans des ateliers de modélisation ;
- la conception et la mise en œuvre de plates-formes hautement adaptables ;
- la conception de services systèmes et utilisateurs pour ces plates-formes adaptables ;
- la validation et la vérification d'une relation service/ressource dans un tel environnement ;
- une meilleure compréhension de la relation entre la mobilité, la gestion des ressources et les composants logiciels. Une expérimentation sur des applications de transport et d'E-Services permettra de valider ce travail.

Le projet MOSAIQUES a permis de conforter le niveau scientifique des équipes partenaires et d'accroître leur visibilité dans le domaine de la production du logiciel, en leur fournissant un cadre commun de travail pour l'ensemble de leurs thèmes de recherche ainsi qu'un démonstrateur par la mise en œuvre d'une application dédiée au transport et aux E-Services dans un tel contexte.

Le projet MOSAIQUES a permis le développement des compétences dans le domaine de la conception d'applications ubiquitaires, dans la conception et la réalisation de plates-formes d'exécution et de services. Il a su articuler leur travaux pour démontrer la faisabilité d'une démarche de conception allant de la définition du modèle de l'application jusqu'à sa mise en œuvre par transformation de modèles et génération de code sur une plate-forme d'exécution. Les particularités des plates-formes ont été modélisées et intégrées dans cette démarche. Les équipes ont également montré leur savoir-faire sur la définition et la réalisation de plates-formes d'exécution pour un milieu ubiquitaire.

Les équipes du projet MOSAIQUES ont été des acteurs des groupes de recherche nationaux dans le domaine de l'intelligence ambiante et de l'ubiquitaire, tout particulièrement David Simplot-Ryl, Gilles Grimaud, Sylvain Lecomte et Christophe Gransart. Les seniors du projet ont souvent été sollicités dans le cadre de comités de programme de conférence du domaine ou encore dans le cadre d'expertise de travaux de thèses en lien avec le thème du projet.

Plusieurs thèses ont été soutenues dans les diverses équipes. Nous avons envisagé en début de projet que quinze thèses soutenues sur la thématique du projet. Aujourd'hui, nous dénombrons dix thèses soutenues, huit thèses en cours et quatre Habilitations à Diriger des Recherches soutenues réparties comme suit :

Equipes	Thèses soutenues	Thèses en cours	HDR soutenues
EMD	1	2	-
INRETS-LEOST	-	1	-
LIFL-GOAL	3	3	-
LIFL-NOCE	1	-	-
LIFL-RD2P	1	-	-
LIFL-SMAC	-	-	1
LIFL-STC	1	1	1
LAMIH-ROI	2	1	2
Total	10	8	4

TAB. 13.1 – Nombre de thèses et d'HDR dans le projet MOSAIQUES

Les doctorant ont été financés par des allocations régionales ou nationales ou sur d'autres projets en lien avec le projet MOSAIQUES. Nous n'avons pas eu de bourse de thèse régionale ou nationale dédiée directement à ce projet.

Nous détaillons les publications dans le tableau suivant. La légende de ce tableau est la suivante.

- Revues internationales avec comité de lecture : RI
- Revues nationales avec comité de lecture : RN
- Conférences internationales avec actes : CI
- Conférences nationales avec actes, : CN
- Ateliers internationaux : AI
- Ateliers nationaux : AN
- Brevets : B

Nous soulignons qu'une publication a été faite avec toutes les équipes ayant participé à l'axe 1 et que les équipes ont publiées également deux à deux, par exemple GOAL/NOCE, GOAL/INRETS, GOAL/STC, STC/RD2P. Le premier chiffre correspond au nombre de pu-

blications, le second après le "/" correspond au nombre de publications signées avec d'autres équipes. Le total de chaque colonne ne prend pas en compte les doublons, c'est-à-dire que si deux équipes sont co-auteurs d'une même publication, celle-ci apparaît dans la ligne de chaque équipe, mais n'est comptabilisée dans la ligne "Total" qu'une seule fois.

Equipes	RI	RN	CI	CN	AI	AN	B
EMD	1	1	5	1/1	1	5	
INRETS-LEOST			2/2		1/1	1/1	1
LIFL-GOAL	1	2	11/3	3/1	1/1	7/1	
LIFL-NOCE			3/1	1/1	2		
LIFL-RD2P			8/4		6/4		
LIFL-SMAC	1		1	2	1	1	
LIFL-STC	1		6/4	1/1	4/4		
LAMIH-ROI	1	1	1	3/1	3	2	
Total	5	4	30	9	15	16	1

TAB. 13.2 – Nombre de publications pour le projet MOSAIQUES

13.1.2 Retombées sur le plan de la structuration de la recherche

Le développement d'un projet commun entre deux universités de la région, une école d'ingénieurs et un institut national a permis un échange fructueux entre les différents acteurs du projet et a renforcé la visibilité extérieure de la région. En effet, ce projet a permis d'acquérir des compétences sur l'ubiquité numérique et plus particulièrement sur l'adaptabilité dynamique. Ce travail devrait servir de base pour le prochain CPER sur l'intelligence ambiante.

Le projet a contribué au développement et à la structuration de la recherche dans la région Nord - Pas de Calais par le rapprochement des équipes de domaines différents allant du système à l'ingénierie du logiciel, à la sûreté de fonctionnement jusqu'aux usages. Le projet a permis de mobiliser une soixantaine de chercheurs et d'ingénieurs venant des différentes équipes et de faire évoluer leurs compétences vers la maîtrise des modèles et infrastructures pour les environnements ubiquitaires. Les membres des équipes ont appris à mieux connaître les domaines des autres et ont partagé leurs connaissances. Ils également construit un savoir-faire commun. Des échanges fructueux au cours des réunions des différents axes et des réunions plénières ont abouti à des publications communes.

Les différents acteurs du projet MOSAIQUES ont également répondu ensemble à des appels régionaux et nationaux de type ANR. Ils continuent à échanger et à partager au travers des pôles de compétitivité de la région (notamment PICOM et I-Trans).

13.1.3 Retombées sur le plan social, économique et culturel

Les équipes participantes étaient déjà très impliquées dans les actions R &D avec des PME/PMI régionales. On peut par exemple citer deux bourses CIFRE et une bourse région/entreprise avec la société NorSys ainsi qu'un projet RNTL avec la société ALICANTE pour le LIFL. Les actions du projet MOSAIQUES ont permis d'augmenter cette dimension en fournissant la démarche et les technologies nécessaires au développement des compétences des PME/PMI

régionales. Les grands groupes de la région tels que AUCHAN, Les 3 Suisses, La Redoute ou encore Decathlon, au travers du pôle de compétitivité PICOM, se sont également intéressés à nos résultats. Les grands groupes nationaux, tels que Thalès, France Telecom R&D et EDF ou encore les organismes de recherche tels que le CEA avec lesquels nous avons des contacts à travers des coopérations ou des co-encadrements de thèses, sont également intéressés par les avancées dans le domaine de l'adaptabilité d'applicatifs. Plusieurs projets ANR de type technologies logiciels ou réseaux ont été soumis et acceptés pendant la durée du projet MOSAIQUES.

Les résultats du projet ont également été diffusés par nos interventions dans des cours de Master recherche ou séminaires en direction de nos étudiants, mais également des entreprises.

13.1.4 Retombées sur le plan du rayonnement

Le département Informatique et Automatique de l'Ecole des Mines de Douai (EMD) est membre du pôle GSP (Grille, Système et Parallélisme) animé par Bertil Foliot et Jean-Louis Pazat au sein du GdR 725 du CNRS (ASR : Architectures, Systèmes et Réseaux). En relation avec le projet MOSAIQUES, différents membres de l'EMD participent activement par des exposés et des démonstrations à l'action "Adaptation dynamique aux environnements d'exécution" animée au sein du GSP du GDR ASR par Françoise André (professeur à l'Université de Rennes1). Les résultats obtenus par l'EMD au sein du projet MOSAIQUES ont permis de nouer des collaborations qui se sont notamment traduites par eux projets concrets. Le premier, labellisé CARNOT-MINES et intitulé "Plate-form ouverte pour réaliser des expériences virtuelles sur les matériaux", est mené en collaboration avec un consortium international de cimentiers à travers le département Génie Civil et Environnement de l'EMD. Le second projet, mené avec différents partenaires nationaux, a été soumis auprès de l'ANR. Il mettrait en œuvre nos travaux sur l'informatique ubiquitaire pour l'aide aux secouristes, notamment dans le cas de catastrophes naturelles.

Le LAMIH est un des acteurs du groupe de recherche GTMOB du PRC I3. Dans ce cadre, Sylvain LECOMTE a été président du comité de programme de la conférence UBIMOB'05 et est membre du comité de programme depuis 2004. Sylvain LECOMTE et Vincent POIRRIEZ ont été membres (et rapporteurs) de jurys de thèse dans les domaines du projet MOSAIQUES dans plusieurs universités. Enfin, MOSAIQUES a permis de nouer des contacts avec plusieurs autres équipes, ce qui a conduit au montage de plusieurs projets et à l'acceptation par l'ANR du projet TACOS.

Le laboratoire LEOST de l'INRETS a des coopérations régionales avec la société Infodio sur l'utilisation de terminaux embarqués (PDA, portables, ...) dans les TERs, participe au projet européens InteGRail sur l'intégration des systèmes d'information ferroviaires avec les différents acteurs tels que Alstom, Bombardier, Siemens, SNCF, ... Le laboratoire participe également à une ARA nationale sur les composants embarqués en environnement hétérogène (projet REVE). Le laboratoire fait également partie d'un groupe de travail du GDR I3 sur la mobilité.

L'équipe GOAL du LIFL à travers de l'équipe-projet INRIA ADAM est l'un des membres du réseau d'excellence européen FP6AOSD (Aspect-Oriented Software Development) qui a pris comme cas d'application l'adaptabilité dans les environnements mobiles. L'équipe participe également aux projets européens de ITEA S4ALL, aux projets nationaux de type ANR Technologies logicielles Flex-eWare, FAROS, JOnES et de type ARA Sécurité Informatique avec le projet REVE. L'équipe collabore également avec l'équipe Prog de la VUB (Bruxelles)

sur l'intelligence ambiante.

L'équipe RD2P du LIFL collabore maintenant depuis 17 ans avec la société Gemplus, devenue GemAlto, leader mondial des solutions à base de cartes à puce. Elle participe au développement de nouvelles plates-formes d'exécutions embarquées sur des matériels fortement contraints dans le projet intégré WASP ainsi que dans le cadre du projet ANR SVP. Les travaux d'adaptabilité de la plateforme JITS, portés par Alexandre Courbot, ont été primés, par le "JavaCard Consortium" en 2006 puis par le "JAX Innovation Contest" en 2007.

L'équipe NOCE du LIFL est un membre du réseau d'excellence Européen Kaleidoscope sur le thème des Environnements Informatiques pour l'Apprentissage Humain (EIAH). Dans ce cadre, elle participe au groupe d'intérêt sur l'apprentissage mobile (Mobile Learning SIG). Elle est également responsable du projet ANR Télécoms exploratoire p-LearNet (2006-2009) sur l'apprentissage mobile qui a été labellisé par le pôle de compétitivité "Industries du Commerce". Les membres de l'équipe sont également impliqués dans la préparation et la réalisation de la prochaine école d'été CNRS de la communauté EIAH sur les mêmes thèmes. A travers ces différentes activités, l'équipe NOCE pourra diffuser les résultats du projet MOSAIQUES dans la communauté scientifique mais également au niveau industriel à travers les relations établies avec le pôle de compétitivité dans le cadre du projet p-LearNet.

L'équipe SMAC du LIFL fait partie de plusieurs groupes de travail du GDR I3 comme ASA, MFI et MIMOSA. Elle a réalisé plusieurs contrats de recherche notamment pour la société Cryo Interactive dans le cadre des jeux vidéo et la société La Redoute dans le cadre de l'identification de profils clients. Elle fait partie du programme TACT du CPER Nord-Pas de Calais à travers les projets NIPO sur les interactions, COLORS sur les outils logiciels et FormatSciences sur les nouveaux usages. Elle est membre du contrat Esprit AgentLinkII de la CEE.

L'équipe STC du LIFL participe à l'ACI sécurité SPOPS et Isabelle Simplot-Ryl également membre du projet CREOL (Concurrent Reflective Object-oriented Language) du département d'informatique de l'université d'Oslo.

Nous avons également accueilli des chercheurs invités dans nos équipes (1 chez LIFL-GOAL, 1 au LAMIH-ROI) ainsi que vingt-deux stagiaires répartis dans les différentes équipes.

Les différentes équipes ont formé stagiaires, doctorants et ingénieurs à ces techniques et ont donné des cours dans le cadre des masters régionaux sur l'informatique mobile.

13.1.5 Préoccupations de développement durable

Le projet MOSAIQUES participe également au développement durable. En effet en prenant en compte les besoins actuels en matière de conception d'applications pour des équipements mobiles et en rendant ces applications adaptables, les résultats du projet s'inscrivent dans la durée.

Les démonstrateurs correspondent à la mise en œuvre d'applications ubiquitaires dans les transports et les E-services, mais ils peuvent facilement être projetés dans le domaine de la formation, de la santé et du commerce qui seront des applications courantes dans l'avenir :

- l'amélioration de l'information auprès des usagers, les divertissements tout au long du trajet, et les possibilités de travail au cours des déplacements amènera l'utilisateur à passer du mode de transport individuel, qu'est la voiture, à un mode de transport en commun, plus attractif, économique et écologique ;
- l'éducation permanente implique toutes les étapes d'apprentissage depuis l'école maternelle jusque après la retraite. Ceci est actuellement considéré comme un aspect essen-

tiel pour les individus et la société. L'acquisition de connaissances et des compétences permet de trouver plus facilement un emploi, d'améliorer son statut professionnel, de s'épanouir et/ou de participer à l'évolution de la société ;

- l'utilisation d'applications pour la santé est également une avancée pour le suivi des malades et les échanges entre soignants ;
- finalement les services en liaison avec le commerce ont déjà prouvés leur utilité avec le développement de l'Internet et devraient se prolonger rapidement par l'ubiquité numérique, via le pôle Industrie du Commerce.

Facilitant la communication et la capitalisation de connaissances, les Technologies de l'Information et la Communication ont dès à présent un rôle majeur dans ce contexte. Les enjeux concernant l'informatique ubiquitaire sont donc cruciaux. La percée des objets nomades en fait aujourd'hui des dispositifs de premier plan : leur présence massive rend obligatoire l'adaptation des E-Services sur ces supports. Les travaux sur le «mobile learning» ont déjà montré des apports très importants pour l'enseignement : indépendance du lieu, authenticité des situations. Le paysage éducatif des futures générations intégrera forcément des objets nomades.

En démontrant la faisabilité (et l'utilité) du projet MOSAIQUES pour un ensemble d'applications de la vie courante, nous indiquons son intérêt fondamental : faciliter l'exploitation logicielle maximale des objets nomades pour étudier aisément les apports d'usage.

13.2 Les perspectives de développement

Le projet MOSAIQUES a permis de faire en sorte que huit équipes de deux universités, une école d'ingénieurs et d'un institut de recherche se retrouvent autour de l'ubiquité numérique et augmentent leur connaissance. Ce projet est le début d'une aventure dans la région Nord-Pas-de Calais. Plus particulièrement, les chercheurs de l'ingénierie du logiciel ont été sensibilisés dans leurs activités à la difficulté de concevoir et d'exécuter des applications dans des environnement ubiquitaires. La création d'un campus intelligence ambiante, mais également avec les pôles de compétitivité I-Trans et PICOM, devraient leur permettre de poursuivre leurs travaux. Par ailleurs, les travaux du projet MOSAIQUES seront poursuivis dans le cadre des collaborations établies avec les partenaires nationaux et internationaux. Plusieurs projets ANR et un projet Européen ont été acceptés, d'autres sont en cours d'évaluation.

13.3 Confidentialité

Ce rapport n'a pas de caractère confidentiel. La plupart des résultats ont été publiés ou sont en passe de l'être.

Bibliographie

- [ACSR06] J.-J. Vandewalle A. Courbot, G. Grimaud and D. Simplot-Ryl. Application-driven customization of an embedded java virtual machine. In *proceedings of the Second International Symposium on Ubiquitous Intelligence and Smart Worlds (UISW2005)*, 2006.
- [BCG07a] Philippe Bossaert, Eric Cybulski, and Jean-Philippe Gruszecki. Dconcevoir une application ubiquitaire sous forme de services. rapport de projet M2Pro e-services, 04 2007.
- [BCG⁺07b] Philippe Bossaert, Eric Cybulski, Jean-Philippe Gruszecki, Vantroys Thomas, and Xavier Le Pallec. Projection de modèles abstraits d’applications ubiquitaires vers les technologies sca et xpd. workshop Apprentissage Mobile, Conférence EIAH 2007, 06 2007.
- [BCSR06a] Arnaud Bailly, Mireille Clerbout, and Isabelle Simplot-Ryl. Component composition preserving behavioural contracts based on communication traces. In Jacques Farré, Igor Litovsky, and Sylvain Schmitz, editors, *Proc. 10th Int. Conference on Implementation and Application of Automata (CIAA 2005), Revised Selected Papers*, volume 3845 of *Lecture Notes in Computer Science*, pages 54–65, Sophia Antipolis, France, 2006. Springer-Verlag.
- [BCSR06b] Arnaud Bailly, Mireille Clerbout, and Isabelle Simplot-Ryl. Component composition preserving behavioural contracts based on communication traces. *Theoretical Computer Science*, 363(2) :108–123, 2006.
- [BLMD05] Olivier Barais, Julia Lawall, Anne-Françoise Le Meur, and Laurence Duchien. Providing support for safe software architecture transformations. In *Working Session of the 5th Working IEEE/IFIP Conference on Software Architecture (WICSA 2005)*, Pittsburg, PA, USA, nov 2005.
- [BSO06a] Gautier Bastide, Abdelhak-Djamel Seriai, and Mourad Oussalah. Adaptation of monolithic software components by their transformation into composite configurations based on refactoring. In *Proceedings of the 9th International SIG-SOFT Symposium on Component-Based Software Engineering (CBSE)*, pages 368–375, 2006.
- [BSO06b] Gautier Bastide, Abdelhak-Djamel Seriai, and Mourad Oussalah. Adapting software components by structure fragmentation. In *Proceedings of the 2006 ACM symposium on Applied computing (SAC)*, pages 1751–1758, New York, NY, USA, 2006. ACM Press.
- [BSO06c] Gautier Bastide, Abdelhak-Djamel Seriai, and Mourad Oussalah. Dynamic adaptation of software component structures. In *Proceedings of the 2006 IEEE*

International Conference on Information Reuse and Integration (IRI), pages 404–409, 2006.

- [BSO06d] Gautier Bastide, Abdelhak-Djamel Seriai, and Mourad Oussalah. Restructuration de composants logiciels : Une approche d’adaptation structurelle statique basée sur la refactorisation de code orienté-objet. In *Workshop OCM-SI (INFORSID 2006)*, 2006.
- [BSO06e] Gautier Bastide, Abdelhak-Djamel Seriai, and Mourad Oussalah. Transformation d’un composant logiciel centralisé en un composant logiciel distribué. In *Journées Composants*, pages 25–36, 2006.
- [BSO07a] Gautier Bastide, Abdelhak-Djamel Seriai, and Mourad Oussalah. Auto-adaptation de composants logiciels : Prise en compte du contexte d’exécution pour l’auto-adaptation structurelle de composants logiciels. In *Workshop GED-SIP (INFORSID 2007)*, 2007.
- [BSO07b] Gautier Bastide, Abdelhak-Djamel Seriai, and Mourad Oussalah. Restructuration de composants logiciels : Une approche d’adaptation structurelle de composants logiciels monolithiques basée sur leur refactorisation. *RSTI - L’Objet*, 13(1) :81–116, 2007.
- [BSO07c] Gautier Bastide, Abdelhak-Djamel Seriai, and Mourad Oussalah. Software component re-engineering for their runtime structural adaptation. In *Proceedings of the 31st Annual IEEE International Computer Software and Applications Conference (COMPSAC - 2007)*, 2007.
- [CCDG05] J. Coutaz, J. Crowley, S. Dobson, and D. Garlan. Context is key. *Communication of the ACM*, 48(3) :49–53, 2005.
- [CCG⁺06] O. Caron, B. Carré, C. Gransart, X. Le Pallec, S. Lecomte, R. Marvie, M. Nebut, D. Seriai, and G. Vanwormhoudt. Propositions pour la modélisation d’applications ubiquitaires. In *Ubimob’06, 3e Journées Francophones Mobilité et Ubiquité*, Conservatoire National des Arts et Métiers - Paris, september 2006. short paper.
- [CGV05] A. Courbot, G. Grimaud, and J.-J. Vandewalle. Romization : Early deployment and customization of java systems for restrained devices. In *International workshop on Construction and Analysis of Safe, Secure and Interoperable Smart devices (CASSIS05)*, 2005.
- [CPP⁺05] Samuel Colin, Dorian Petit, Vincent Poirriez, Jerome Rocheteau, Rafael Marcano, and Georges Mariano. Brilliant : An open source and xml-based platform for rigorous software development. In *SEFM ’05 : Proceedings of the Third IEEE International Conference on Software Engineering and Formal Methods*, pages 373–382, Washington, DC, USA, 2005. IEEE Computer Society.
- [CRS07] D. Conan, R. Rouvoy, and L. Seinturier. Scalable processing of context information with cosmos. In *Proceedings of the 7th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS’07)*, volume 4531 of *Lecture Notes in Computer Science*, pages 210–224. Springer, June 2007.
- [CRS08] D. Conan, R. Rouvoy, and L. Seinturier. Cosmos : composition de n œuds de contexte. *Techniques et Sciences Informatiques*, 2008.

- [DM06a] Jérémy Dubus and Philippe Merle. Applying OMG D&C Specification and ECA Rules for Autonomous Distributed Component-based Systems . In *Proceedings of the Models Workshop on Models@Runtime*, volume 4364 of *Lecture Notes in Computer Science*, pages 242–252, Genova, Italia, oct 2006. Springer-Verlag.
- [DM06b] Jérémy Dubus and Philippe Merle. Autonomous Deployment and Reconfiguration of Component-based Applications in Open Distributed Environments . In *Proceedings of the 8th International OTM Symposium on Distributed Objects and Applications (DOA'06)*, volume 4277 of *Lecture Notes in Computer Science*, pages 26–27, Montpellier, France, nov 2006. Springer-Verlag.
- [DM06c] Jérémy Dubus and Philippe Merle. Vers l'auto-adaptabilité des architectures logicielles dans les environnements ouverts distribués. In *Actes de la 1ère Conférence Francophone sur les Architectures Logicielles (CAL'06)*, pages 13–29, Nantes, France, sep 2006. Hermès Sciences.
- [FGM05a] Areski Flissi, Christophe Gransart, and Philippe Merle. A Component-Based Software Infrastructure for Contextual Transportation Applications . In *The 5th International Conference on Intelligent Transportation System - Telecommunication (ITS-T 2005)*, Brest, France, jun 2005.
- [FGM05b] Areski Flissi, Christophe Gransart, and Philippe Merle. A Component-based Software Infrastructure for Ubiquitous Computing . In *4th International Symposium on Parallel and Distributed Computing (ISPDC 2005)*, Lille, France, jul 2005.
- [FGM05c] Areski Flissi, Christophe Gransart, and Philippe Merle. A Service Discovery and Automatic Deployment Component-Based Software Infrastructure for Ubiquitous Computing . In *Proceeding of Ubiquitous Mobile Information and Collaboration Systems, CAiSE Workshop (UMICS 2005)*, Porto, Portugal, jun 2005.
- [FGM05d] Areski Flissi, Christophe Gransart, and Philippe Merle. Une infrastructure à composants pour des applications ubiquitaires. In *2ième conférence Ubiquité et Mobilité (UbiMob 2005)*, Grenoble, France, jun 2005.
- [FM06] Areski Flissi and Philippe Merle. A Generic Deployment Framework for Grid Computing and Distributed Applications . In *Proceedings of the 2nd International OTM Symposium on Grid computing, high-performance and Distributed Applications (GADA'06)*, volume 4279 of *Lecture Notes in Computer Science*, pages 1402–1411, Montpellier, France, nov 2006. Springer-Verlag.
- [FN07] Johan Fabry and Carlos Noguera. Ami : The future is now – a position paper. In *3rd Workshop on Object Technology for Ambient Intelligence and Pervasive Systems (OT4AmI) at ECOOP'07*, July 2007.
- [GDL05] Hocine Grine, Thierry Delot, and Sylvain Lecomte. Adaptive query processing in mobile environment. In *MPAC '05 : Proceedings of the 3rd international workshop on Middleware for pervasive and ad-hoc computing*, pages 1–8, New York, NY, USA, 2005. ACM Press.
- [GGSR06a] Dorina Ghindici, Gilles Grimaud, and Isabelle Simplot-Ryl. Embedding verifiable information flow analysis. In *Proc. 4th Annual Conference on Privacy, Security and Trust*, pages 343–352, Toronto, Canada, 2006. McGraw-Hill.

- [GGSR⁺06b] Dorina Ghindici, Gilles Grimaud, Isabelle Simplot-Ryl, Issa Traore, and Yanguo Liu. Integrated security verification and validation : Case study. In *Proc. of the Second IEEE LCN Workshop on Network Security (WoNS 2006), held in conjunction with the 31st Annual IEEE Conference on Local Computer Networks (LCN 2006)*, Tampa, Florida, 2006.
- [GGSR07] Dorina Ghindici, Gilles Grimaud, and Isabelle Simplot-Ryl. An information flow verifier for small embedded systems. In *Proc. Workshop in Information Security Theory and Practices 2007 Smart Cards, Mobile and Ubiquitous Computing Systems*, volume 4462, pages 189–201, Heraklion, Crete, Greece, 2007. Springer-Verlag.
- [GHL05] Hocine Grine, Colombe Herauld, and Sylvain Lecomte. Localisation de services techniques dans un modele a composants. In *Actes des 4emes Journees composants (JC), Le Croisic, avril., 2005*.
- [GHSR05] Gilles Grimaud, Yann Hodique, and Isabelle Simplot-Ryl. Secure extensible type system for efficient embedded operating system by using metatypes. In *Sanso 2005 : First International Workshop on System and Networking for Smart Objects (SaNSO 2005), in Proc. 11th International Conference on Parallel and Distributed Systems*, volume 2, pages 83–87, Fukuoka, Japan, 2005. IEEE computer Society. Best paper award.
- [GHSR06] Gilles Grimaud, Yann Hodique, and Isabelle Simplot-Ryl. Can small and open embedded systems benefit from escape analysis ? In *Proc. of ECOOP Workshop on Implementation, Compilation, Optimization of Object-Oriented Languages, Programs and Systems (ICOOOLPS'2006)*, 2006.
- [GHSR07] Gilles Grimaud, Yann Hodique, and Isabelle Simplot-Ryl. A verifiable light-weight escape analysis supporting creational design patterns for small embedded systems. In *Proc. of the 2007 IEEE International Symposium on Ubisafe Computing (UbiSafe-07)*, pages 440–447. IEEE Computer Society, 2007.
- [HG06] Sylvain Lecomte Hocine Grine, Thierry Delot. Un évaluateur de requêtes adaptable pour un environnement pervasif. In *3èmes Journées Francophones : Mobilité et Ubiquité 2006*. ACM Press, 2006.
- [HG07] Sylvain Lecomte Hocine Grine, Thierry Delot. Evaluation de requêtes adaptable dans les environnements pervasifs : une approche à base de composants récursifs. In *Actes de l'Atelier Gestion de Données dans les Systèmes d'Information Pervasifs (GEDSIP) organisé dans le cadre du XXVème congrès INFORSID*, Perros-Guirec, mai 2007.
- [HSR06] Michaël Hauspie and Isabelle Simplot-Ryl. Cooperation in ad hoc networks : enhancing the virtual currency based models. In *InterSense '06 : Proceedings of the first international conference on Integrated internet ad hoc and sensor networks*, volume 138 of *ACM International Conference Proceeding Series*, New York, NY, USA, 2006. ACM Press.
- [HSR07] Michaël Hauspie and Isabelle Simplot-Ryl. Local algorithms enforcing node cooperation in self-organized ad hoc networks. In *Proc. 4th Annual Conference on Wireless On demand Network Systems and Services (WONS 2007)*, pages 130–137, Obergurgl, Austria, 2007. IEEE Press.

- [JOSR05] Einar Broch Johnsen, Olaf Owe, and Isabelle Simplot-Ryl. A dynamic class construct for asynchronous concurrent objects. In M. Steffen and G. Zavattaro, editors, *Proc. 7th International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS'05)*, volume 3535 of *Lecture Notes in Computer Science*, pages 15–30. Springer-Verlag, June 2005.
- [KVC07] Sarra Kaddouci, Thomas Vantroys, and Vincent Chevrin. From task model to multi-channel access : Services integration in the ubi-learn platform. In *Conférence Web Based Education 2007*, France, mars 2007.
- [LLY⁺07] Dominique Lecllet, Eric Leprêtre, Peter Yvan, Céline Quénu-Joiron, Bénédicte Talon, and Thomas Vantroys. Améliorer un dispositif pédagogique par l'intégration de nouveaux canaux de communication. In *Actes de la conférence EIAH 2007*, page 10, Lausanne, Suisse, 06 2007.
- [MCC⁺07] A. Muller, O. Caron, B. Carré, G. Vanwormhoudt, and S. Bouzitouna. Ingénierie multi-modèles : Projection flexible d'assemblages de modèles. In *Conf. francophone Langages et Modèles à Objets (LMO'07)*, Toulouse, mars 2007.
- [MCG05] K. Marquet, A. Courbot, and G. Grimaud. Ahead of time deployment in rom of an embedded java-os. In *proceeding of International Conference on Embedded Soft-ware and Systems '05*, 2005.
- [MGSR05] K. Marquet, G. Grimaud, and D. Simplot-Ryl. Optimization of the root set for object-oriented memory management of smart devices. In *MOS Workshop / ECOOP*, 2005.
- [MM01] R. Marvie and P. Merle. CORBA Component Model : Discussion and Use with OpenCCM. Technical report, Laboratoire d'Informatique Fondamentale de Lille (LIFL), June 2001.
- [MN06] R. Marvie and M. Nebut. Processus de modélisation incrémentaux. In *2nd Journées sur l'Ingénierie Dirigée par les Modèles (IDM'06)*, Lille, juin 2006.
- [Mos06] Projet Mosaïques. Livrable 1- etat de l'art. Rapport de recherche, Laboratoire d'Informatique Fondamentale de Lille, LIFL, février 2006.
- [MP06] V. Mencl and M. Polak. Uml 2.0 components and fractal : An analysis. 5th Fractal Workshop Leveraging an European open source community around the Fractal component model, juillet 2006. WS à ECOOP'2006.
- [MR06] Maxime Morge and Jean-Christophe Routier. Système multi-agents délibératif. In *Actes des Journées Francophone Planification, Décision, Apprentissage pour la conduite de système (JFPDA'2006)*, June 2006.
- [MR07] Maxime Morge and Jean-Christophe Routier. Debating over heterogeneous descriptions special issue on formal ontologies for communicating agents. *Journal of Applied Ontology*, 2007.
- [MRS06] Maxime Morge, Jean-Christophe Routier, and Yann Secq. Argumentation to compose services. In *Proceedings of the 18th Belgian-Dutch conference of Artificial Intelligence (BNAIC'2006)*, pages 237–246, October 2006.
- [MRSD06a] Maxime Morge, Jean-Christophe Routier, Yann Secq, and Tony Dujardin. Comment atteindre un accord sur une représentation ? In Vincent Chevrin, edi-

tor, *Actes des 14e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2006)*. Hermès, 2006. JFSMA'2006 – Annecy (France) – 18-20 octobre 2006.

- [MRS06b] Maxime Morge, Jean-Christophe Routier, Yann Secq, and Tony Dujardin. A formal framework for inter-agents dialogue to reach an agreement about a representation. In Roberta Ferrario, Nicola Guarino, and Laurent Prevot, editors, *Proceedings of the Workshop on Formal Ontologies for Communicating Agents (FOCA'2006)*, August 2006.
- [PDNP07] Carlos Andreas Parra, Maja D'Hondt, Carlos Noguera, and Ellen Van Paeschen. Introducing context-awareness in applications by transforming high-level rules. In *3rd Workshop on Object Technology for Ambient Intelligence and Pervasive Systems (OT4AmI) at ECOOP'07*, July 2007.
- [PMM+06] Xavier Le Pallec, César Moura, Raphaël Marvie, Mirabelle Nebut, and Jean-Claude Tarby. Supporting generic methodologies to assist ims-ld modeling. In *Proceedings of the ICALT 2006 conference*, Kerkrade, Netherlands, 07 2006. IEEE Computer Society Press. PDF : <http://www.ask.iti.gr/icalt/2006/others/>.
- [PSM07] A. Plsek, L. Seinturier, and P. Merle. Ambient-oriented programming in fractal. In *3rd Workshop on Object Technology for Ambient Intelligence and Pervasive Systems (OT4AmI) at ECOOP'07*, July 2007.
- [RCG05] Christophe Rippert, Alexandre Courbot, and Gilles Grimaud. A low-footprint class loading mechanism for embedded java virtual machines. In *3rd ACM International Conference on the Principles and Practice of Programming in Java*, 2005.
- [RGA04a] J. Rioult, C. Gransart, and S. Ambellouis. Determination de l'arrêt souhaité. Demande de brevet français N° 04 12856 déposé le 3 décembre 2004, ALL/HC 116.623., décembre 2004. En cours d'extension à l'Europe.
- [RGA04b] J. Rioult, C. Gransart, and S. Ambellouis. Zut, j'ai loupé mon arrêt ! un nouveau service d'aide aux déplacements. In *Les nouvelles technologies dans la cité*, 2004. Rennes, France.
- [RM06] Romain Rouvoy and Philippe Merle. Using Microcomponents and Design Patterns to Build Evolutionary Transaction Services . In *Proceedings of the International ERCIM Workshop on Software Evolution (EVOL'06)*, Lille, France, apr 2006.
- [RM07] Romain Rouvoy and Philippe Merle. Using Microcomponents and Design Patterns to Build Evolutionary Transaction Services . *Electronic Notes in Theoretical Computer Science (ENCTS)*, 166 :111–125, jan 2007.
- [RSAM06a] Romain Rouvoy, Patricia Serrano-Alvarado, and Philippe Merle. A Component-based Approach to Compose Transaction Standards . In *Proceedings of the 5th International Symposium on Software Composition (SC'06)*, volume 4089 of *Lecture Notes in Computer Science*, pages 114–130, Vienna, Austria, mar 2006. Springer-Verlag.
- [RSAM06b] Romain Rouvoy, Patricia Serrano-Alvarado, and Philippe Merle. Towards Context-Aware Transaction Services . In *Proceedings of the 6th International*

Conference on Distributed Applications and Interoperable Systems (DAIS'06), volume 4025 of *Lecture Notes in Computer Science*, pages 272–288, Bologna, Italy, jun 2006. Springer-Verlag.

- [SARM05] Patricia Serrano-Alvarado, Romain Rouvoy, and Philippe Merle. Self-Adaptive Component-Based Transaction Commit Management . In *Proceedings of the 4th International Middleware Workshop on Adaptive and Reflective Middleware (ARM'05)*, volume 116 of *ACM International Conference Proceeding Series*, pages 1–6, Grenoble, France, nov 2005. ACM Press.
- [SBO06a] Abdelhak-Djamel Seriai, Gautier Bastide, and Mourad Oussalah. How to generate distributed software components from centralized ones? *Journal of Computers*, 1(11) :40–52, 2006.
- [SBO06b] Abdelhak-Djamel Seriai, Gautier Bastide, and Mourad Oussalah. Transformation of centralized software components into distributed ones by code refactoring. In *Proceedings of the 6th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS)*, pages 332–346, 2006.
- [Sun06] X. Sun. Reconfiguration d'architecture pour les applications mobiles. Mémoire de master recherche, Université des Sciences et Technologies de Lille, June 2006.
- [TV06] C. Tombelle and G. Vanwormhoudt. Dynamic and generic manipulation of models : From introspection to scripting. In *9th International Conference MoDELS 2006*, number LNCS 4199, Italy, octobre 2006.
- [YTE07] Peter Yvan, Vantroys Thomas, and Leprêtre Eric. Accès multimodal à un blog pour le suivi des stages. In *Actes de l'Atelier Apprentissage Mobile - Couplé à EIAH 2007*, Lausanne, Suisse, 06 2007.