PERCOMOM : une méthode de modélisation des applications interactives personnalisées appliquée à l'information voyageur dans le domaine des transports collectifs.

Arnaud Brossard

Numéro d'ordre : 08 / 43

THESE

pour l'obtention du

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

Spécialité Automatique et informatique des Systèmes Industriels et Humains

Mention informatique

Présentée par

Arnaud BROSSARD

Ingénieur

PERCOMOM :
Une méthode de modélisation des applications interactives personnalisées appliquée à l'information voyageur dans le domaine des transports collectifs

Soutenue publiquement le 10 décembre 2008 devant le jury composé de :

Gaëlle Calvary	MdC HDR, Université de Grenoble	Examinateur
Mourad Abed	Professeur à l'UVHC	Directeur de thèse
Bernard Espinasse	Professeur, Université d'Aix-Marseille	Rapporteur
Christophe Kolski	Professeur à l'UVHC	Co-directeur de thèse
Philippe Palanque	Professeur, Université de Toulouse 3	Président
Guillaume Uster	Chargé de Recherche, INRETS, Villeneuve d'Ascq	Examinateur
Jean Vanderdonckt	Professeur, Université catholique de Louvain-La-Neuve	Rapporteur

Université de Valenciennes et du
Hainaut-Cambraisis (UVHC)
Le Mont-Houy
F-59313 Valenciennes Cedex 9

Avant Propos

Le travail présenté dans ce mémoire a été réalisé au Laboratoire d'Automatique, de Mécanique et d'Informatique industrielles et Humaines (LAMIH) de l'Université de Valenciennes et du Hainaut-Cambraisis (UVHC), au sein de l'équipe Raisonnement Automatique et Interaction Homme-Machine (RAIHM), dirigée par le Professeur Christophe Kolski, dans le cadre d'un financement associé au projet Viatic.Mobilité sélectionné par l'Agence Nationale de la Recherche (ANR) dans le cadre du pôle de compétitivité I-Trans dédié au transport et piloté par l'Institut National de Recherche sur les Transports et leur Sécurité (INRETS) à Villeneuve d'Ascq.

Je remercie Messieurs Bernard Espinasse, Professeur à l'Université d'Aix-Marseille et Jean Vanderdonckt, Professeur à l'Université Catholique de Louvain-La-Neuve pour m'avoir fait l'honneur d'être rapporteurs de ce mémoire. Je suis également très reconnaissant envers Madame Gaëlle Calvary, Maître de Conférences HDR de l'Université de Grenoble, Monsieur Philippe Palanque, Professeur à l'Université de Toulouse 3, et Monsieur Guillaume Uster, Chargé de Recherche à l'INRETS de Villeneuve d'Ascq, pour avoir accepté d'examiner ce travail.

Cette thèse n'aurait pas pu avoir lieu sans Christophe Kolski qui m'a accueilli au sein de son équipe et qui m'a permis de surmonter de nombreuses difficultés administratives dues à mon statut un peu particulier. Je tiens aussi à remercier mon administration de tutelle, le Rectorat de l'Académie de Créteil, qui a bien voulu m'accorder une mise à disposition pour études et recherche à caractère d'intérêt général afin que je puisse réaliser cette thèse dans de bonnes conditions.

Je tiens aussi à remercier Mourad Abed, Professeur à l'UVHC, qui m'a guidé tout au long de mes travaux. Mes remerciements vont également à Guillaume Uster, chargé de recherche à l'INRETS, et à Christelle Petit-Rozé, chargé de mission à l'INRETS, pour l'ensemble de leur travail dans le cadre du projet ANR Viatic.Mobilité et aussi pour leurs avis et commentaires toujours forts pertinents concernant mon travail de recherche. J'associe également à ces remerciements l'ensemble des partenaires impliqués dans le projet ANR Viatic.Mobilité qui, à travers leurs problématiques métier diverses et variées, m'ont permis d'enrichir énormément mes réflexions.

Je remercie également l'ensemble du personnel du LAMIH, chercheurs, ingénieurs et administratifs, de leurs encouragements et de leur amitié. Je tiens à remercier plus particulièrement Philippe Dos Santos pour m'avoir apporté son aide et son support pour résoudre un certain nombre de problèmes techniques. J'associe également à ces remerciements, l'ensemble des personnes avec qui j'ai partagé mon bureau et plus particulièrement Stéphanie Bernonville, Arnaud Doniec, David Gacquer, Mohamad Anas Hariri et Djilali Idoughi. Nos discussions ont toujours été très enrichissantes et m'ont souvent permis d'apporter un regard neuf sur mes problématiques de recherche.

Je remercie les différents étudiants et stagiaires pour l'intérêt qu'ils ont porté à mes sujets d'études ou de recherche et surtout pour la contribution précieuse qu'ils m'ont tous apporté. Je pense notamment à Mohamed Hédi Hadroug, à Farid Ben Miled et à Bacha Firas.

Enfin, je remercie toute ma famille et tous mes amis pour leur soutien et leurs encouragements pendant toute la durée de mes travaux. Je leur exprime ma profonde reconnaissance pour m'avoir soutenu dans ce nouveau projet professionnel.

A ma femme,

A mes enfants,

A mes parents,

A ma famille,

A mes amis.

Table des matières

AVANT PROPOS ..... **I**

TABLE DES MATIERES ..... **V**

LISTE DES FIGURES ..... **IX**

LISTE DES TABLEAUX ..... **XIII**

LISTE DES ACRONYMES ..... **XV**

INTRODUCTION GENERALE **1**

CHAPITRE 1 LA MODELISATION DES APPLICATIONS INTERACTIVES DANS LES SYSTEMES D'INFORMATION VOYAGEUR DANS LE DOMAINE DES TRANSPORTS **5**

INTRODUCTION 7

1.1 LA MODELISATION DES APPLICATIONS INTERACTIVES : DEFINITION ET PRINCIPES 7

1.1.1 Définition et principes de la modélisation *7*

1.1.2 L'approche dirigée par les modèles *8*

1.1.3 Les principes de transformations de modèles, préconisés par l'OMG, pour passer du niveau CIM à l'application concrète de niveau PSM *10*

1.1.4 Synthèse et discussion *12*

1.2 LES OUTILS ET METHODES DE MODELISATION DES APPLICATIONS INTERACTIVES 13

1.2.1 Définitions et principes *13*

1.2.2 Étude des méthodes et outils de modélisation des applications interactives *14*

1.2.3 Synthèse et discussion *19*

1.3 LES PROCESSUS METIER DANS LE CADRE DE LA MODELISATION DES APPLICATIONS INTERACTIVES 21

1.3.1 Définition et principes *21*

1.3.2 L'utilisation des processus métier dans le cadre de la modélisation : comparaison par rapport aux autres approches *21*

1.3.3 Synthèse et discussion *25*

1.4 L'ONTOLOGIE DANS LE CADRE DE LA MODELISATION DES APPLICATIONS INTERACTIVES 26

1.4.1 Définition et principes *26*

1.4.2 Étude des outils et méthodes de description et de manipulation des ontologies *27*

1.4.3 Synthèse et discussions *30*

1.5 LA MODELISATION DES APPLICATIONS INTERACTIVES DANS LE DOMAINE DES TRANSPORTS DANS LE CADRE DE L'INFORMATION VOYAGEUR 31

1.5.1 Les typologies d'applications dans le domaine des transports *31*

1.5.2 Les travaux existants dans le domaine de l'information voyageur *32*

1.5.3 Synthèse et discussion *35*

CONCLUSION 36

CHAPITRE 2 DE LA PERSONNALISATION DANS LES SYSTEMES D'INFORMATION A CELLE DANS LES SYSTEMES D'INFORMATION VOYAGEURS DANS LE DOMAINE DES TRANSPORTS **39**

INTRODUCTION 41

2.1 LA PERSONNALISATION : DEFINITION, PRINCIPES ET METHODES 41

2.1.1 Définition *41*

2.1.2 Les facteurs qui influent sur la personnalisation *43*

2.1.3 Méthodes et techniques de personnalisation des contenus *44*

2.1.4 Synthèse et discussion *46*

2.2 LA MODELISATION DE L'UTILISATEUR DANS LE CADRE DE LA PERSONNALISATION 46

2.2.1 Définition *46*

2.2.2 Étude des outils et méthodes de modélisation de l'utilisateur *47*

2.2.3 Synthèse et discussion *49*

2.3 LA MODELISATION DU CONTEXTE DANS LE CADRE DE LA PERSONNALISATION 51

2.3.1 Définition *51*

2.3.2 Étude des outils et méthodes de modélisation du contexte *52*

2.3.3 Synthèse et discussion *53*

2.4 LA MODELISATION DES CONTENUS DANS LE CADRE DE LA PERSONNALISATION 54

2.4.1 Définition *54*

2.4.2 Étude des outils et méthodes de modélisation des contenus *55*

2.4.3 Synthèse et discussion *56*
2.5 LA PERSONNALISATION DANS LE DOMAINE DES TRANSPORTS 57
2.5.1 Les besoins dans le domaine des transports *57*
2.5.2 Les outils et méthodes existants adaptés au domaine des transports *58*
2.5.3 Synthèse et discussion *59*
CONCLUSION 60

CHAPITRE 3 PERCOMOM, UNE METHODE DE MODELISATION DES APPLICATIONS INTERACTIVES UTILISANT LES PROCESSUS METIER 63
INTRODUCTION 65
3.1 L'APPROCHE DE MODELISATION AU NIVEAU CONCEPTUEL 66
3.1.1 Introduction générale sur l'approche globale de PERCOMOM *66*
3.1.2 La modélisation conceptuelle dans PERCOMOM (niveau CIM) *67*
3.1.3 Les modèles CIM, dits modèles de support, dans le cadre de la modélisation conceptuelle des interactions. *73*
3.1.4 La modélisation des interactions au niveau conceptuel (niveau CIM) *82*
3.1.5 Synthèse et conclusion *95*
3.2 L'ARCHITECTURE DE TYPE MDA UTILISEE PAR PERCOMOM (NIVEAUX CIM, PIM ET PSM) 96
3.2.1 Introduction *96*
3.2.2 Le niveau PIM : un niveau composé de services fonctionnels *97*
3.2.3 Le niveau PSM : une architecture pour une génération semi-automatique des applications *103*
3.2.4 Synthèse et discussion *105*
3.3 LES PRINCIPES DE TRANSFORMATION DE MODELES POUR PASSER DU NIVEAU CIM A L'APPLICATION CONCRETE DE NIVEAU PSM 106
3.3.1 Introduction *106*
3.3.2 La transformation de modèles dans PERCOMOM *107*
3.3.3 Analyse et discussion *110*
CONCLUSION 111

CHAPITRE 4 PERCOMOM, UNE PRISE EN COMPTE DE LA PERSONNALISATION QUI S'APPUIE SUR UNE ONTOLOGIE DE DOMAINE 113
INTRODUCTION 115
4.1 LA PERSONNALISATION DES CONTENUS : UN SERVICE FONCTIONNEL DE NIVEAU PIM 116
4.1.1 Introduction *116*
4.1.2 Intégration du service fonctionnel de personnalisation dans les modèles conceptuels *116*
4.1.3 L'évolutivité du service de personnalisation dans l'architecture proposée par PERCOMOM *122*
4.1.4 Synthèse et discussion *125*
4.2 LA PRISE EN COMPTE DU CONTEXTE 126
4.2.1 Introduction *126*
4.2.2 Les contextes applicatifs, géographiques et temporels *127*
4.2.3 Les autres éléments contextuels *129*
4.2.4 Prise en compte du contexte et approche MDA *132*
4.2.5 Synthèse et discussion *134*
4.3 LA PRISE EN COMPTE DE L'UTILISATEUR 136
4.3.1 Introduction *136*
4.3.2 La prise en compte des préférences de l'utilisateur *136*
4.3.3 La prise en compte de l'expérience de l'utilisateur vis-à-vis de l'application *138*
4.3.4 Synthèse et discussion *140*
4.4 LA PRISE EN COMPTE DES CONTENUS 140
4.4.1 Introduction *140*
4.4.2 Les contenus et l'ontologie de domaine *141*
4.4.3 La notion de silo de données au niveau des applications *144*
4.4.4 Synthèse et discussion *146*
CONCLUSION 147

CHAPITRE 5 MISE EN ŒUVRE DE PERCOMOM DANS LE PROJET ANR VIATIC.MOBILITE 149
INTRODUCTION 151
5.1 LE PROJET ANR VIATIC.MOBILITE 151
5.1.1 Introduction *151*
5.1.2 Les différents espaces d'interaction et la notion de continuité de l'information dans le cadre d'un déplacement multimodal *152*
5.1.3 Les principaux besoins d'adaptations spécifiques identifiés en matière de modélisation *154*

5.1.4 Conclusion ... *156*
5.2 LA PRISE EN COMPTE DU VECU DES DEPLACEMENTS DANS LES APPLICATIONS TRANSPORT ... 156
5.2.1 Introduction ... *156*
5.2.2 La définition de la notion de vécu des déplacements ... *157*
5.2.3 La prise en compte du vécu des déplacements dans PERCOMOM ... *157*
5.3 PREMIERE APPLICATION : UNE BORNE INTERACTIVE D'ORIENTATION ... 163
5.3.1 Introduction ... *163*
5.3.2 Description de l'application ... *163*
5.3.3 La modélisation de niveau CIM de l'application table d'orientation (cf. chapitre 3, §3.1) ... *165*
5.3.4 Conclusion sur la première application ... *175*
5.4 UN CAS PRATIQUE DE TRANSFORMATION DE MODELES DANS PERCOMOM : DES MODELES CONCEPTUELS DE NIVEAU CIM DE L'APPLICATION A L'APPLICATION CONCRETE ... 175
5.4.1 Introduction ... *175*
5.4.2 Présentation de l'application ... *175*
5.4.3 Les modèles conceptuels associés à l'application ... *176*
5.4.4 Les choix techniques effectués : l'exemple des éléments de framework associés au modèle statique des interactions (IM2) ... *178*
5.4.5 Exemple de transformation d'un modèle statique d'interaction (IM2) ... *183*
5.4.6 Exemple de transformation d'un modèle de processus métier (IM1) ... *185*
5.4.7 Conclusion sur l'exemple de transformation de modèles ... *186*
5.5 TROISIEME APPLICATION : LA PERSONNALISATION DES CONTENUS DANS LE CADRE D'UNE APPLICATION PERMETTANT DE VISUALISER LES PROCHAINS DEPARTS EN GARE ... 187
5.5.1 Introduction ... *187*
5.5.2 Description de l'application ... *187*
5.5.3 Le service de personnalisation et la prise en compte du contexte dans la modélisation de niveau CIM associée à l'application ... *188*
5.5.4 Conclusion sur la troisième application ... *191*
5.6 LA PRISE EN COMPTE DE L'UTILISATEUR DANS PERCOMOM : APPARTENANCE SOCIALE ET ECHANGES VERBAUX ... 192
5.6.1 Un exemple d'utilisation d'un modèle d'organisation (IM2) et d'un modèle d'action (BM1) dans le cadre d'une adaptation d'une application à l'utilisateur ... *192*
5.6.2 Prise en compte des interactions verbales entre personnes dans un processus métier (modèle IM1) de niveau CIM ... *195*
5.6.3 Conclusion sur les exemples complémentaires ... *198*
CONCLUSION ... 198

CHAPITRE 6 EVALUATION GLOBALE DE PERCOMOM ET PERSPECTIVES DE RECHERCHE ... 199
INTRODUCTION ... 201
6.1 ÉVALUATION DE PERCOMOM ... 201
6.1.1 Comparaison de PERCOMOM par rapport aux approches similaires ... *201*
6.1.2 Les contributions principales de PERCOMOM ... *205*
6.2 PERSPECTIVES DE RECHERCHE ... 206
6.2.1 Outillage de PERCOMOM ... *206*
6.2.2 PERCOMOM : une méthode de modélisation conceptuelle à finaliser ... *206*
6.2.3 Vers une architecture ouverte et modulaire ... *207*
6.2.4 Les experts métier au cœur de la modélisation des applications ... *208*
6.2.5 Vers une prise en compte globale de la personnalisation ... *209*
6.2.6 La prise en compte du contexte dans PERCOMOM : entre pertinence et "superflu" ... *210*
6.2.7 Vers une prise en compte globale de l'utilisateur dans le cadre de la personnalisation des contenus 210
6.2.8 Les contenus : un élément central de la personnalisation ... *212*
6.2.9 Vers une généralisation de PERCOMOM ... *212*
CONCLUSION ... 213
CONCLUSION GENERALE ... 215

BIBLIOGRAPHIE ... 219

ANNEXE A : LE MOTEUR DE REGLES UTILISE DANS PERCOMOM ... 233
A.1. LES PRINCIPES ... 233
A.2. LA DEFINITION D'UNE REGLE METIER ... 233
A.2.1. Introduction ... *233*
A.2.2. La définition d'une clause ... *234*

A.2.3. La définition d'une expression *235*
A.2.4. La définition d'une règle *236*
A.3. LA DEFINITION DES DOMAINES DE VALIDITE DANS PERCOMOM 237
A.3.1. Introduction générale *237*
A.3.2. Les restrictions géographiques *238*
A.3.3. Les restrictions temporelles *239*
A.3.4. Les restrictions applicatives *241*
A.3.5. Les domaines de validité *242*
A.4. LES MOTEURS ASSOCIES A L'EVALUATION DES REGLES METIER ET DES DOMAINES DE VALIDIE 242
A.4.1. Le moteur d'évaluation des règles métier *242*
A.4.1. Le moteur d'évaluation des domaines de validité *245*

ANNEXE B : QUELQUES ELEMENTS UTILISES DANS LE CADRE DU FORMALISME DE MODELISATION DE PERCOMOM 247
B.1. LES PROPRIETES ASSOCIEES A UN EVENEMENT (MODELE EM3) 247
B.2. DESCRIPTION DES PRINCIPAUX ELEMENTS STATIQUES D'INTERACTION UTILISES DANS PERCOMOM AU NIVEAU DES MODELES STATIQUES D'INTERACTION (IM2) 247

ANNEXE C : L'OUTIL DE CREATION DE MODELES DE PROCESSUS METIER (IM1) ET DES MODELES STATIQUES D'INTERACTION (IM2) DEVELOPPE DANS LE CADRE DE PERCOMOM 251
B.1. LES PROPRIETES ASSOCIEES A UN EVENEMENT (MODELE EM3) 251
B.2. QUELQUES COPIES D'ECRAN DE L'OUTIL DE MODELISATION 251
C.2.1. L'écran d'accueil *251*
C.2.2. La création d'un nouveau projet de modélisation *253*
C.2.3. La validation d'un diagramme *254*
C.2.3. L'association d'artefact aux différents éléments composant les modèles *255*
C.2.4. Transformation de modèles de processus métier (IM1) du niveau CIM au niveau PIM *256*

Liste des figures

Figure 1.1. Modèle, métamodèle (formalisme de modélisation) et métamétamodèle (métaformalisme) (Blanc et Salvatori, 2005) 8
Figure 1.2. Représentation globale de l'Approche MDA vue sous l'angle de l'IHM 9
Figure 1.3. Modèle en "Y" du passage du niveau PIM au niveau PSM dans l'approche MDA (OMG, 2003) 10
Figure 1.4. La transformation de modèles vue par l'OMG dans le cadre d'une architecture MDA 11
Figure 1.5. Architecture globale de l'approche Wisdom (Jardim Nunes, 2001) 15
Figure 1.6. Modèle simplifié du framework associé aux interfaces graphiques en UsiXML (Vanderdonckt, 2005) 15
Figure 1.7. Flux de création d'une interface graphique indépendamment des plateformes d'exécution (Schattkowsky et Lohmann, 2005) 16
Figure 1.8. Architecture globale de l'approche Amacont (Hinz et Fiala, 2004) 17
Figure 1.9. Principe d'adaptation des interfaces dans Amacont (Hinz et Fiala, 2004) 17
Figure 1.10. Modèle de transformation pour la génération d'interfaces graphiques (Forbrig *et al.*, 2004) 18
Figure 1.11. Les problèmes de communication au sein d'un projet : exemple pédagogique 23
Figure 1.12. Exemple de processus métier mélangeant tâches et échanges de messages verbaux (exemple tiré de la documentation de la notation BPMN) 25
Figure 1.13. Exemple de représentation d'une classe "Route_Section" de l'ontologie OTN dans l'outil Protege 29
Figure 1.14. Graphe de dépendance de la classe "Route_Section" dans l'ontologie OTN 30
Figure 1.15. Exemple d'IHM sur une plateforme de type PDA dans le cadre du projet TramMate (Kjeldov *et al.*, 2003) 33
Figure 1.16. Exemple d'utilisation de PRIAM sur des panneaux d'affichage public dans un aéroport (Jacquet *et al.*, 2006) 33
Figure 1.17. Présentation globale de l'architecture AVANTI (Fink *et al.*, 1998) 34
Figure 1.18 Vue globale du système proposé par (Matsubara *et al.*, 2001) 35
Figure 2.1. Les facteurs externes influençant la personnalisation d'après (Van Setten, 2001) 43
Figure 2.2. Exemple de personnalisation par catégorie sur le site "Mon Yahoo!" 44
Figure 2.3. Exemple de filtrage social sur le site de vente en ligne de la société Amazon 45
Figure 2.4. L'ontologie de profil utilisateur proposé par (Golemati *et al.*, 2007) dans l'outil Protege 49
Figure 3.1. Etapes prise en compte par PERCOMOM dans le cadre d'une approche classique de développement suivant un cycle en V 67
Figure 3.2. La vision de l'évolution de la modélisation suivant l'OMG (Watson, 2005) 68
Figure 3.3. La vision globale, au niveau CIM, d'une application dans PERCOMOM 69
Figure 3.4. Processus d'analyse utilisé dans le cadre de PERCOMOM pour identifier les différentes catégories de modèles et les différents modèles au niveau conceptuel 71
Figure 3.5. Les dix grandes étapes de la méthode PERCOMOM 72
Figure 3.6. Schéma de l'ensemble des modèles conceptuels de niveau CIM dans PERCOMOM 73
Figure 3.7. Exemple d'ontologie, dans PERCOMOM, associée aux espaces géographiques (représentation provenant de l'outil Protege présenté dans le chapitre 1 (cf. §1.4.2) 75
Figure 3.8. Exemple simple de représentation d'un modèle d'organisation avec un formalisme spécifique à PERCOMOM (modèle SM2) 76
Figure 3.9. Exemple simplifié de représentation de la structure hiérarchique de la SNCF avec un formalisme spécifique à PERCOMOM (modèle SM2) 76
Figure 3.10. Exemple de définition de deux rôles d'accès à l'aide d'informations contenues dans le profil de l'utilisateur et d'un formalisme spécifique à PERCOMOM (modèle SM3) 77
Figure 3.11. Exemple d'utilisation de l'ontologie géographique dans un modèle de sécurité (modèle SM3) 78
Figure 3.12. Exemple d'utilisation d'une règle métier dans un modèle d'action en utilisant un formalisme spécifique à PERCOMOM (modèle BM1) 79
Figure 3.13. Exemple de transformation d'un modèle de relation dans une ontologie de domaine (a) vers un modèle entité-relation de définition d'une base de données (b) 79
Figure 3.14. Les principaux éléments statiques d'interaction dans PERCOMOM au niveau CIM (présentation sous la forme d'un diagramme de classes) 86
Figure 3.15. Exemple de modèle statique d'interface (a) (modèle IM2 de niveau CIM) avec une de ses représentations graphiques possibles (niveau PSM) sur un ordinateur de type PC (b) et sur un téléphone portable de type SmartPhone (c) 88
Figure 3.16. Exemple d'association de propriétés à un élément d'interaction de type *UIFieldStatic* dans un modèle statique d'interaction (IM2) 89

Figure 3.17. Exemple de fenêtres pouvant être associées à un processus d'identification (fenêtres issues du niveau PSM)....91
Figure 3.18. Exemple de contenu d'un UIUnit associé à un processus d'identification de l'utilisateur (niveau CIM)....91
Figure 3.19. Exemple de processus métier (modèle IM1) pouvant être associé à l'*UIUnit* (modèle IM2) de la Figure 3.18....92
Figure 3.20. Principe de création d'un modèle dynamique des interactions pour un *UIUnit* AA à partir de trois processus métier différents utilisant cet *UIUnit* (afin de faciliter la lecture de la figure, chaque tâche de l'*UIUnit* AA a été identifiée par une lettre de l'alphabet)....93
Figure 3.21. L'architecture globale à base de frameworks associée à la méthode PERCOMOM....97
Figure 3.22 Exemple d'association d'un service fonctionnel d'adaptation linguistique de niveau PIM dans des modèles IM2 ne niveau CIM, pour un *UIElement* de type *UIFieldStatic* (a) et pour un *UIElement* de type *UIGroup* (b)....98
Figure 3.23. Procesus de création d'un nouveau service fonctionnel de niveau PIM dans PERCOMOM....100
Figure 3.24 Exemple d'utilisation des propriétés d'un service fonctionnel de niveau PIM dans un modèle IM2 de niveau CIM....100
Figure 3.25. Exemple d'association de différents groupes de propriétés pour un même service fonctionnel de niveau PIM dans un modèle IM2 de niveau CIM (utilisation d'une règle métier pour savoir quel groupe de propriétés utiliser)....102
Figure 3.26. Exemple de répartition de quelques plateformes techniques de consultation sur les deux niveaux de framework PSM....103
Figure 3.27. Processus de sélection des informations de positionnement d'un élément d'interaction au niveau PSM....105
Figure 3.28. Principe global de transformation de modèles dans PERCOMOM....107
Figure 3.29. Exemple de transformation de modèles dans le cadre de la création d'une application finale dans PERCOMOM....108
Figure 3.30. Les différents types de transformation de modèles dans PERCOMOM pour passer du niveau CIM au niveau PSM....109
Figure 4.1. Exemple d'association d'un service de personnalisation à un élément d'interaction de type *UIUnit* dans un modèle statique d'IHM (modèle IM2) au niveau CIM....120
Figure 4.2. Utilisation d'un critère lié au concept dans le cadre de la personnalisation....121
Figure 4.3. Processus utilisé pour l'adaptation du service de personnalisation de niveau PIM en fonction de nouveaux besoins fonctionnels énoncés par les experts métier....123
Figure 4.4. Exemple d'utilisation de deux services différents de personnalisation au niveau PIM en fonction de la région où se trouve l'utilisateur au moment de son accès à l'application (définition au niveau de l'architecture globale PERCOMOM)....124
Figure 4.5. Exemple d'adaptation du service de personnalisation au niveau PSM en fonction de la plateforme technique utilisée (soit des PDA avec GPS soit des PDA sans GPS)....125
Figure 4.6. Exemple d'affichage de deux écrans d'une même application d'affichage des prochains départs en gare avec un affichage non personnalisé pour (a) et personnalisé pour (b) en fonction de la prochaine destination de l'usager....126
Figure 4.7. Exemple d'utilisation du contexte applicatif dans le cadre de la sélection de propriétés pour un UIElement de type UIFieldOut....127
Figure 4.8. Exemple d'utilisation d'une restriction géographique au sein d'un processus métier (modèle IM1) utilisé pour afficher des informations transport....129
Figure 4.9. Représentation d'une partie des éléments constituant l'environnement physique dans lequel évolue une application finale (seuls quelques éléments sont représentés sur cette figure)....130
Figure 4.10. Exemple d'utilisation d'un événement contextuel pour déclencher automatiquement le démarrage d'un processus métier dès que celui-ci survient....131
Figure 4.11. Exemple d'utilisation d'un test sur la validité d'un événement contextuel au sein d'un processus métier (modèle IM1)....131
Figure 4.12. Prise en compte d'un stimulus (modification du contexte externe) au niveau CIM, PIM et PSM dans le cadre de notre approche....132
Figure 4.13. Exemple d'adaptation d'une application à la vitesse de déplacement d'un utilisateur : déplacement normal (a) et déplacement rapide (b)....134
Figure 4.14. Présentation de du modèle IM2 associé à l'exemple de la Figure 4.13 sensible à un événement contextuel relatif à la vitesse de déplacement de l'utilisateur....135
Figure 4.15. Utilisation de propriétés spécifiques dans l'artefact d'un UIFieldInOut pour permettre l'affichage d'une préférence....138

Figure 4.16 Exemple d'utilisation de la notion d'expérience utilisateur pour sélectionner un sous-processus à exécuter au sein d'un processus métier (modèle IM1) 139
Figure 4.17. Exemple de représentation des liens entre concepts métier dans une ontologie de domaine pour trois concepts métier spécifiques dans le domaine des transports (formalisme disponible dans l'outil Protege qui permet de créer et de manipuler des ontologies) 142
Figure 4.18. Exemple de définition des propriétés pour un lien "est effectue en" dans une ontologie métier à l'aide de l'outil Protege 143
Figure 4.19. Exemple de définition de restriction au niveau d'un concept métier (d'une classe de l'ontologie) dans l'outil Protege 143
Figure 4.20. Exemple de modèle entité-relation (modèle BM2) obtenu à partir d'une ontologie de domaine 144
Figure 4.21. Exemple schématique d'association de silos de données à des applications dans PERCOMOM ... 146
Figure 5.1. Les trois grands espaces d'information identifiés dans le cadre du projet Viatic.Mobilité (INRETS, 2008) 152
Figure 5.2. Les services d'aide à la mobilité dans Viatic.Mobilité (INRETS, 2008) 153
Figure 5.3. Les services d'agrément dans Viatic.Mobilité (INRETS, 2008) 154
Figure 5.4. Les catégories de vécu des déplacements identifiées dans le cadre du projet Viatic.Mobilité (Juguet *et al.*, 2007) 157
Figure 5.5. La partie spécifique d'analyse associée à la prise en compte des catégories de vécu des déplacements lors de la création des modèles de processus métier (IM1) au niveau CIM 158
Figure 5.6. Arbre de décision utilisé dans PERCOMOM pour la prise en compte des changements de catégories de vécu des déplacements lors de l'exécution des applications 159
Figure 5.7. Exemple d'utilisation de la catégorie de vécu des déplacements dans un modèle de processus métier (modèle IM1) de niveau CIM 160
Figure 5.8. Exemple d'utilisation de la notion de catégorie de vécu des déplacements dans un modèle statique d'interaction (modèle IM2) de niveau CIM 161
Figure 5.9. Extrait du processus métier (modèle IM1 de niveau CIM) associé à l'affichage des informations sur le prochain déplacement de l'utilisateur 161
Figure 5.10. Exemple d'écrans associés à l'affichage du prochain déplacement (a) pour un usager de la catégorie de vécu des déplacements Castor, pour lequel on affiche des informations professionnelles, et (b) pour un usager qui n'est pas de la catégorie Castor 162
Figure 5.11 Borne interactive développée dans le cadre du projet ANR Viatic.Mobilité utilisant l'application table d'orientation à destination des usagers des transports en commun 163
Figure 5.12. Page d'accueil associée à la table d'orientation développée dans le cadre du projet Viatic.Mobilité 164
Figure 5.13. Ecran affiché, au niveau de la table d'orientation, lorsque l'usager effectue une recherche d'adresse 165
Figure 5.14. Extrait de l'ontologie géographique utilisée dans le cadre de l'application table d'orientation fonctionnant sur une borne interactive 166
Figure 5.15. Les deux groupes principaux de processus métier au niveau de la page d'accueil de la table d'orientation : (a) processus à valeur ajoutée qui apporte une valeur ajoutée par rapport à la consultation d'une simple carte, (b) processus directement lié aux opérations qui peuvent être effectuées sur l'élément graphique de type carte géographique 167
Figure 5.16. Modèle de processus métier (IM1) associé au processus métier de recherche d'une adresse sur une carte 168
Figure 5.17. Extrait du processus métier principal (IM1) "DémarrerTableOrientation" limité à l'activation des processus métier associé au premier groupe de processus métier (les processus à valeur ajoutée) de l'écran initial de l'application table d'orientation 170
Figure 5.18. Modèle statique de groupe logique d'interaction associé à la table d'orientation présentée dans le chapitre 5.3.1 171
Figure 5.19. Modèle statique d'UIUnit associé à la partie navigation sur la carte géographique de l'application table d'orientation 172
Figure 5.20. Ecran d'accès aux services de loisir dans Viatic.Mobilité 176
Figure 5.21. Extrait du modèle métier (IM1) associé aux trois premiers éléments du "menu graphique" permettant de sélectionner un service de loisir dans le projet ANR Viatic.Mobilité 177
Figure 5.22 *UIUnit* (modèle IM2) associé au menu graphique de sélection des services de loisir dans le projet ANR Viatic.Mobilité 177
Figure 5.23 Architecture en couches utilisée dans le cadre de l'exemple de transformation de modèles 178
Figure 5.24. Extrait de la partie de framework de niveau PIM associé au formalisme utilisé dans les modèles de niveau CIM 179

Figure 5.25. Représentation synthétique de la transformation de modèles du niveau CIM au niveau PIM pour un modèle de processus métier (IM1) 180
Figure 5.26. Modèle de classes simplifié associé à la partir de framework de niveau PIM associée aux services fonctionnels et techniques 181
Figure 5.27. Représentation synthétique de la transformation de modèles du niveau CIM au niveau PIM pour un modèle statique d'interaction (IM2) 182
Figure 5.28. Schéma global simplifié de passage du niveau CIM au niveau PSM d'un modèle de processus métier (IM1) et d'un modèle statique d'interaction (IM2) 183
Figure 5.29. Les trois *UIUnit* (a, b et c) contenus dans l'*UIGroup* contenant l'*UIUnit* "UIUnit_MenuImageServicesLoisir" 184
Figure 5.30. Exemple de page affichée pour un usager, sur PC, dans la gare de Lille Flandres. 187
Figure 5.31. Exemple de page affichée pour un usager sur son PC, dans la gare de Valenciennes. 188
Figure 5.32. Exemple de page affichée, sur un téléphone, dans la gare de Valenciennes. 188
Figure 5.33. Processus métier "HorairesDepart" associé à une application permettant d'afficher les horaires des prochains départ dans la gare où se trouve l'utilisateur 189
Figure 5.34. Modèle statique d'interaction (IM2) du groupe logique d'interaction "UIGroup_HorairesDepart" associé à l'application d'affichage des horaires des prochains départ dans la gare où se trouve l'utilisateur 190
Figure 5.35. Modèle statique d'interaction (IM2) de l'unité logique d'interaction "UIUnit_HorairesDepart" associée au groupe logique d'interaction "UIGroup_HorairesDepart" dans l'application d'affichage des horaires des prochains départ dans la gare où se trouve l'utilisateur 191
Figure 5.36. Modèle d'organisation SM2 associé à la répartition des utilisateurs dans deux groupes sociaux différents ("Etudiant" et "Non etudiant") 192
Figure 5.37. Utilisation d'une propriété "action" dans un artefact associé à un UIFieldAction permettant l'impression d'un ticket 193
Figure 5.38. Modèle d'action (SM2) associé à l'action "ImprimerTicket" et dont le contenu tient compte du groupe social d'appartenance de l'utilisateur 193
Figure 5.39. Exemple de ticket de paiement personnalisé (a) pour un étudiant, (b) pour un usager qui n'est pas un étudiant 194
Figure 5.40. Exemple de prise en compte des acteurs et des échanges d'informations verbaux au niveau de la définition des modèles conceptuels d'un processus métier associer à la demande d'horaires de train par un usager à un guichetier dans une gare 196
Figure 5.41. Exemple d'application à disposition du guichetier, accessible via un navigateur sur un ordinateur de type PC, associée au modèle métier décrit dans la Figure 5.40 196
Figure 5.42. Exemple d'application à disposition d'un guichetier (ou d'un agent de terrain), accessible via un navigateur web sur un téléphone portable de type Apple iPhone, associée au modèle métier décrit dans la Figure 5.40 197
Figure 5.43. Formalisme utilisé pour la prise en compte des interactions verbales dans les processus métier avec le formalisme utilisé pour les tâches manuelles réalisées par l'utilisateur (a) et pour les échanges verbaux réalisés entre deux intervenants (b). 197
Figure 6.1. Méthode d'amélioration continue appliquée PERCOMOM 209

Liste des tableaux

Tableau 1.1. Comparaison des différents outils MDA/IDM du marché 13
Tableau 1.2. Tableau de synthèse sur les approches Wisdom, UsiXML, Schattkowsky et Lohmann, Amacont et Forbrig 19
Tableau 2.1. Comparaison des différentes solutions de modélisation du profil utilisateur 50
Tableau 3-1. Les principaux avantages et inconvénients des modèles de support de niveau CIM dans le cadre de la PERCOMOM 81
Tableau 3.2. Eléments du formalisme général des processus métier utilisés dans le cadre de la méthode PERCOMOM dérivés de la notation BPMN (modèle IM1) 83
Tableau 3.3. Description globale des principaux éléments statiques d'interaction dans PERCOMOM au niveau CIM (modèle IM2) 86
Tableau 3.4. Liste des codes couleurs utilisés dans PERCOMOM dans les modèles IM2 de niveau CIM pour les principaux éléments d'interaction (format RGB) 87
Tableau 3.5. Fonctions standards associées à un *UIFieldStaticInOut* dans PERCOMOM 89
Tableau 3.6. Les principales propriétés associées à un élément d'interaction de type *UIField* et à ses descendants 90
Tableau 3.7. Principaux avantages et inconvénients des approches de type IDM lors du passage du niveau CIM au niveau PSM 96
Tableau 3.8. Exemple de propriétés utilisables au niveau CIM pour un service fonctionnel d'adaptation linguistique de niveau PIM 101
Tableau 3.9. Tableau récapitulatif des différents éléments architecturaux utilisés au niveau PSM dans PERCOMOM 104
Tableau 3.10. Avantages et inconvénients de l'architecture de PERCOMOM par rapport à une architecture classique de type MDA 106
Tableau 4.1. Tableau comparatif des principaux avantages et inconvénients des différentes possibilités de prise en charge de la personnalisation des contenus dans PERCOMOM 117
Tableau 4.2. Les opérateurs de sélection définis dans le cadre de PERCOMOM pour effectuer des sélections sur les individus de l'ontologie de domaine (OD) 118
Tableau 4.3. Les différentes propriétés définies au niveau de l'artefact associé au service de personnalisation de niveau PIM 120
Tableau 4.4. Exemple de définition d'un libellé de type erreur pour les trois niveaux de plateformes définit dans PERCOMOM 122
Tableau 4.5. Les différents niveaux de prise en charge du contexte dans l'architecture technique proposée dans PERCOMOM 133
Tableau 4.6. Comparaison des avantages et inconvénients de la prise en charge du contexte dans PERCOMOM par rapport à l'approche MDA préconisée par l'OMG 135
Tableau 5.1. Les principaux besoins d'adaptation spécifiques identifiés dans le cadre du projet ANR Viatic.Mobilité 155
Tableau 5.2. Exemple de représentation graphique d'un élément d'interaction de type *UIAction* en fonction de la localisation géographique de l'utilisateur et des capacités techniques de la plateforme utilisée pour accéder à l'application 172
Tableau 5.3. Ensemble des propriétés associées à un libellé de type texte dans le fichier de configuration annexe des libellés de type texte de niveau CIM 173
Tableau 6.1. Comparaison des approches PERCOMOM, CTT (UsiXML) et UML pour la modélisation des processus métier 202
Tableau 6.2. Comparaison des approches PERCOMOM, UsiXML et UML au niveau de la modélisation des interfaces interactives (niveau CIM) 203

Liste des acronymes

BPEL	-	Business Process Execution Language
BPL4WS	-	Business Process Execution Language for Web Services
BPM	-	Business Process Modeling
BPMN	-	Business Process Modeling Notation
CASE	-	Computer-aided Software Engineering
CIM	-	Computational Independent Model
CTT	-	Concur Task Tree
CWM	-	Common Warehouse Metamodel
DBM	-	Database Model
DSL	-	Domain Specific Language
IHM	-	Interaction Homme-Machine
INRETS	-	Institut National de Recherche sur les Transports et leur Sécurité
LAMIH	-	Laboratoire d'Automatique, de Mécanique et d'Informatique industrielles et Humaines
MDA	-	Model Driven Architecture
MDE	-	Model Driven Engineering
MOF	-	Meta Object Facility
OCL	-	Object Constraint Language
OMG	-	Object Management Group
PDA	-	Personal Digital Assistant
PERCOMOM	-	PERsonalization and COnceptual MOdeling Method
PIM	-	Platform Independent Model
PSM	-	Platform Specific Model
QVT	-	Query/View/Transformation
RAIHM	-	Raisonnement Automatique et Interaction Homme-Machine
RBAC	-	Role Based Access Control
RBACM	-	Role Based Access Control Model
SBVR	-	Semantics of Business Vocabulary & Business Rules
SOA	-	Service Oriented Architecture
UHVC	-	Université de Valenciennes et du Hainaut-Cambrésis
UIML	-	User Interface Markup Language
UML	-	Unified Modeling Language
UsiXML	-	USer Interface eXtensible Markup Language
W3C	-	World Wide Web Consortium
WIMP	-	Windows, Icons, Menu, Pointing device
XMI	-	XML Metadata Interchange
XML	-	eXtensible Markup Language

Introduction générale

Dans notre société actuelle, la généralisation de l'utilisation des outils informatiques et la multiplication des sources et des quantités d'informations disponibles ont entraîné de nouveaux besoins. Ainsi, le problème n'est plus de fournir de l'information mais de fournir la bonne information au bon moment et sur le bon support (PDA, ordinateur portable, téléphone portable, etc.). Et ceci, tout en étant capable de prendre en compte l'ensemble des contraintes et limites liées à chaque type de plateforme de consultation de l'information (taille d'écran, capacités graphiques, association du son et de l'image, etc.).

Ceci est d'autant plus vrai dans le domaine des transports où les usagers ont tous des besoins et des buts différents qui sont rarement mutualisables (Lyons *et al.*, 2007). Et, le niveau de complexité est encore plus élevé lorsque plusieurs moyens de transport différents sont utilisés, c'est-à-dire lorsque les déplacements sont dits multi-modaux (ATEC-ITS, 2002) (Perreau, 2002) (Uster, 2000) (Uster, 2001). Ainsi, par exemple, dans une gare, certains utilisateurs vont être à la recherche de leur correspondance, d'autres vont planifier leur prochain déplacement, d'autres encore voudront savoir où acheter un bouquet de fleurs avant de repartir à leur domicile. Dans ce contexte, la personnalisation est devenue un élément déterminant pour permettre de résoudre cette problématique surtout si la diffusion des applications se fait vers un très large public (ChoiceStream, 2007) (Vesanen, 2005). Si dans sa définition générale, personnaliser, c'est individualiser la relation entre l'utilisateur et l'application, derrière ce terme se trouvent de nombreuses problématiques techniques et méthodologiques même lorsqu'on se limite à la personnalisation des contenus fournis à l'utilisateur (Van Setten, 2001). Et, si on dispose aujourd'hui d'outils et de méthodes pour prendre en charge la personnalisation, ceux-ci sont bien souvent spécifiques à certains contextes ou à certains environnements techniques bien déterminés limitant d'autant leur réutilisation et/ou leur généralisation.

Par ailleurs, l'ingénierie dirigée par les modèles (IDM) est aujourd'hui en passe de devenir le nouveau paradigme en matière de développement d'applications (Favre *et al.*, 2006) (Hailpern et Tarr, 2006) (Schmidt, 2006). L'objectif de cette approche est de créer des applications à partir de modèles conceptuels et surtout à partir d'un assemblage de modèles conceptuels. Pour cela, l'IDM se base sur une architecture de type MDA dont le premier niveau est composé d'un modèle conceptuel de l'application indépendant des technologies qui seront utilisées pour réaliser l'application ; ce modèle est appelé CIM (Computational Independent Model). Si cette vision permet d'envisager une industrialisation des développements informatiques, elle présente, au niveau des outils et des méthodes existants aujourd'hui, un certain nombre de manques. Ainsi, la modélisation conceptuelle des applications se limite souvent à une modélisation fortement liée aux techniques de développement (Tariq et Akhter, 2004) et la notion de personnalisation n'est généralement pas prise en compte ou alors de manière incomplète.

L'objectif principal de nos travaux a été de proposer une méthode adaptée au développement des systèmes d'information personnalisés dans le domaine du transport. Mais plus qu'une méthode purement liée à la prise en compte de la personnalisation dans les applications, comme celle proposée par (Anli, 2006), nous avons étudié une approche plus générale permettant d'intégrer la personnalisation dans les modèles d'applications et, plus précisément, dans les modèles conceptuels d'applications. De cette manière, il devient possible d'envisager la création d'applications informatiques personnalisées de manière semi-automatique, c'est-à-dire avec un minimum d'interventions humaines, à partir de modèles conceptuels dans le cadre d'une approche de type IDM où chaque modèle est fortement réutilisable. Les problématiques dans le domaine des transports nécessitant d'être capable de créer des applications pour de multiples plateformes utilisateurs, nous avons aussi porté un intérêt tout particulier à la modélisation des interactions homme-machine (IHM). Tout ceci a donné naissance à la méthode PERCOMOM (PERsonalization and COnceptual MOdeling Method) qui englobe l'ensemble de nos travaux à ce sujet.

Le premier chapitre présente un état de l'art des travaux relatifs à la modélisation des applications. Il explicite les différents termes et définitions utilisés dans le domaine. Puis, à travers une présentation des principaux outils et méthodes existants, il cerne les limites et contraintes des approches actuelles. Il introduit ensuite la notion de processus métier à travers une vision fonctionnelle des problématiques informatiques. Puis, à travers une description générale des ontologies, il montre quel peut être leur apport dans le cadre de la modélisation des applications. Enfin, il présente aussi les différentes solutions développées dans le cadre de la prise en compte des problématiques spécifiques au domaine des transports.

Le deuxième chapitre présente un état de l'art des travaux relatifs à la personnalisation des contenus. Il explicite les principaux termes utilisés dans le domaine et présente les principes et méthodes pour la personnalisation des contenus. Pour personnaliser, il faut aussi, en-dehors de l'outil de personnalisation lui-même, être capable de prendre en compte un certain nombre de facteurs différents comme les préférences de l'utilisateur, le lieu où est consultée l'information, l'heure de consultation de l'information, l'état d'esprit de l'utilisateur au moment de la consultation, etc. Ainsi, le deuxième chapitre présente comment, dans les travaux et méthodes existants, les principaux facteurs sont utilisés et pris en compte. Pour cela, il s'attache à présenter les problématiques existantes, dans le cadre de la personnalisation, autour de la modélisation de l'utilisateur, puis celles qui gravitent autour de la modélisation du contexte et enfin de la modélisation des contenus eux-mêmes. Enfin, il présente les différents travaux et méthodes développés spécifiquement ou mis en œuvre dans le domaine du transport.

Le troisième chapitre présente la méthode PERCOMOM. Pour commencer, il décrit PERCOMOM du point de vue de la modélisation conceptuelle des applications et surtout montre comment cette modélisation, principalement fonctionnelle, permet d'obtenir un modèle CIM réellement indépendant des problématiques techniques. Il montre aussi comment PERCOMOM permet de prendre en compte les problématiques spécifiques aux interactions Homme-Machine. Puis, il montre comment, à travers l'utilisation d'une architecture et de transformations spécifiques, est effectué le passage des modèles conceptuels aux applications concrètes. Enfin, il présente les différentes solutions proposées pour faciliter la réutilisation des modèles.

Le quatrième chapitre présente pour sa part comment PERCOMOM permet de prendre en compte la personnalisation des contenus au niveau des modèles conceptuels. Pour cela, il introduit une notion spécifique à PERCOMOM qui est celle de service fonctionnel qui permet de prendre en compte des problématiques techniques, comme la personnalisation, au niveau conceptuel. Puis, il montre comment PERCOMOM gère les différents facteurs, présentés dans le chapitre 2, ayant une influence sur la personnalisation. Ainsi, il présente comment la prise en compte du contexte dans PERCOMOM peut avoir une interaction avec la problématique de personnalisation. Ensuite, il montre comment les spécificités de l'utilisateur, ou de l'usager dans le domaine du transport, sont gérées dans PERCOMOM. Enfin, à travers une analyse de la prise en compte des contenus dans PERCOMOM, il propose une nouvelle approche de la gestion des contenus dans le cadre de la personnalisation.

Le cinquième chapitre décrit plusieurs applications concrètes de la méthode PERCOMOM développées dans le cadre du projet Viatic.Mobilité (http://viatic.inrets.fr) financé par l'Agence Nationale de la Recherche et développé au sein du pôle de compétitivité I-Trans sous la direction de l'Inrets et plus particulièrement de M. Guillaume Uster. Après une présentation du contenu et des enjeux du projet, trois exemples sont présentés en détail :

- La "table d'orientation" permettant à des usagers de s'orienter dans un lieu public comme une gare. Celle-ci permet de montrer comment il est possible de créer des modèles conceptuels dans PERCOMOM à partir d'une analyse des besoins fonctionnels.
- Un exemple de transformation de modèles, qui se base sur une architecture technique de validation développée pendant nos travaux de recherche, permettant de valider le principe de trnsformation des modèles de niveau conceptuel en applications concrètes manipulables par les utilisateurs dans le cadre d'une approche de type MDA.

- Une application, disponible sur différents systèmes nomades, permettant, à un usager, de consulter simplement les prochains départs en gare en fonction de la gare où il se trouve. Cette application permettra de montrer comment PERCOMOM est capable d'intégrer des problématiques liées à la prise en compte du contexte dans les modèles conceptuels de niveau CIM.

Des exemples complémentaires seront aussi fournis pour présenter d'autres aspects importants de PERCOMOM comme l'utilisation des modèles sociaux ou encore la prise en compte des interactions verbales.

Le sixième et dernier chapitre commence par établir une comparaison entre PERCOMOM et d'autres approches similaires pour les aspects relevant directement de la génération d'IHM. Puis il présente, sous forme synthétique, les différentes contributions de PERCOMOM aussi bien dans le domaine de la modélisation des applications interactives que dans le domaine de la prise en compte de la personnalisation dans le cadre d'une approche de type IDM. Enfin, il se termine par une présentation de l'ensemble des perspectives de recherche ouvertes dans le cadre de nos travaux. Ces perspectives concernent aussi bien la méthode PERCOMOM que, de manière plus générale, la modélisation des applications ou encore la personnalisation des contenus.

Chapitre 1
La modélisation des applications interactives dans les systèmes d'information voyageur dans le domaine des transports

Sommaire

INTRODUCTION **7**

1.1 LA MODELISATION DES APPLICATIONS INTERACTIVES : DEFINITION ET PRINCIPES **7**
1.1.1 DEFINITION ET PRINCIPES DE LA MODELISATION 7
1.1.2 L'APPROCHE DIRIGEE PAR LES MODELES 8
1.1.3 LES PRINCIPES DE TRANSFORMATIONS DE MODELES, PRECONISES PAR L'OMG, POUR PASSER DU NIVEAU CIM A L'APPLICATION CONCRETE DE NIVEAU PSM 10
1.1.4 SYNTHESE ET DISCUSSION 12

1.2 LES OUTILS ET METHODES DE MODELISATION DES APPLICATIONS INTERACTIVES **13**
1.2.1 DEFINITIONS ET PRINCIPES 13
1.2.2 ÉTUDE DES METHODES ET OUTILS DE MODELISATION DES APPLICATIONS INTERACTIVES 14
1.2.3 SYNTHESE ET DISCUSSION 19

1.3 LES PROCESSUS METIER DANS LE CADRE DE LA MODELISATION DES APPLICATIONS INTERACTIVES **21**
1.3.1 DEFINITION ET PRINCIPES 21
1.3.2 L'UTILISATION DES PROCESSUS METIER DANS LE CADRE DE LA MODELISATION : COMPARAISON PAR RAPPORT AUX AUTRES APPROCHES 21
1.3.3 SYNTHESE ET DISCUSSION 25

1.4 L'ONTOLOGIE DANS LE CADRE DE LA MODELISATION DES APPLICATIONS INTERACTIVES **26**
1.4.1 DEFINITION ET PRINCIPES 26
1.4.2 ÉTUDE DES OUTILS ET METHODES DE DESCRIPTION ET DE MANIPULATION DES ONTOLOGIES 27
1.4.3 SYNTHESE ET DISCUSSIONS 30

1.5. LA MODELISATION DES APPLICATIONS INTERACTIVES DANS LE DOMAINE DES TRANSPORTS DANS LE CADRE DE L'INFORMATION VOYAGEUR **31**
1.5.1 LES TYPOLOGIES D'APPLICATIONS DANS LE DOMAINE DES TRANSPORTS 31
1.5.2 LES TRAVAUX EXISTANTS DANS LE DOMAINE DE L'INFORMATION VOYAGEUR 32
1.5.3 SYNTHESE ET DISCUSSION 35

CONCLUSION **36**

Introduction

Aujourd'hui, avec la généralisation des systèmes informatiques embarqués et nomades, l'utilisateur peut accéder à de multiples sources d'information à tout moment et quel que soit l'endroit où il se trouve ; et ceci à partir de plateformes techniques qui peuvent être très différentes (borne interactive, téléphone portable, PDA, ordinateur portable, etc.). Ceci est d'autant plus vrai dans le domaine des transports collectifs où, par définition, l'usager est mobile et passe par de nombreux lieux différents. Les applications doivent donc aujourd'hui devenir multiformes en étant capables de s'adapter à la plateforme utilisée, mais aussi en étant capables d'assurer une continuité de service pendant toute la durée d'un déplacement d'un usager. Tout ceci entraîne une augmentation de la complexité des applications qu'il devient de plus en plus difficile à gérer sans outils et méthodes adaptés.

Aujourd'hui, il semble, d'après les nombreux travaux de recherche dans le domaine et la multiplication des outils commerciaux, que la modélisation des applications constitue une solution prometteuse pour résoudre l'ensemble des problématiques liées à cette complexité. Mais, si la grande majorité des travaux réalisés et des outils proposés semblent être d'accord sur une vision à long terme où les applications ne seraient plus créées qu'à partir d'un simple assemblage de modèles déjà existants, le chemin est encore long pour arriver à ce résultat ; sans compter que les moyens proposés pour y arriver sont parfois fort différents et bien souvent peu compatibles entre eux.

Ce chapitre présente un état de l'art de la modélisation des applications et plus particulièrement de la modélisation des interactions homme-machine dans le cadre d'applications de type WIMP (Window, Icon, Mouse, Pointing device). Ainsi, la première partie présente la notion de modélisation dans le cadre des applications interactives ainsi que les évolutions actuelles et futures dans le domaine. La deuxième partie se focalise sur les principaux outils utilisés aujourd'hui pour modéliser les interfaces homme-machine. La troisième partie s'intéresse à un concept important de notre approche qui est la notion de processus métier. La quatrième partie introduit une autre notion importante qui est celle d'ontologie qui va permettre de faire la séparation entre les données physiques et les modèles. Enfin, la cinquième et dernière partie présente des exemples représentatifs d'outils et de méthodes de modélisation utilisés dans le cadre spécifique du domaine des transports collectifs.

1.1 La modélisation des applications interactives : Définition et principes

1.1.1 Définition et principes de la modélisation

Étymologiquement, le terme "modèle" provient du latin *modellus* qui est un diminutif de *modus* qui veut dire mesure. Ce terme, qui était utilisé à l'origine dans les beaux-arts pour désigner l'objet à l'origine de l'œuvre, a commencé à prendre son sens moderne grâce à l'artiste et savant B. Palissy (1510-1589) qui l'a utilisé dans le sens de "représentation en petit de ce qui sera reproduit en grand" comme, par exemple, une maquette. Dans le domaine scientifique, le mot modèle est utilisé pour la première fois par des cybernéticiens en 1950 (Wiener, 1950) et sa première définition scientifique apparaît en 1966 dans le dictionnaire "Le Trésor de la Langue Française" : "Système Physique, mathématique ou logique représentant les structures essentielles d'une réalité et capable, à son niveau, d'en expliquer ou d'en reproduire dynamiquement le fonctionnement".

Aujourd'hui, dans le domaine de l'informatique, un modèle est généralement associé à une notation et/ou à des outils graphiques permettant de représenter de manière synthétique et simplifiée, mais complète, une partie d'une application (OMG, 2003). Ainsi, chaque modèle représente une vue sur la réalité de l'application comme, par exemple, le modèle de composants en UML qui permet de représenter les différents composants qui constituent l'application ou encore le modèle de déploiement qui permet de représenter le déploiement de ces mêmes composants sur les dispositifs matériels (OMG, 2005). Ceci est d'autant plus vrai dans le cadre des applications interactives que celles-ci peuvent être utilisées sur un nombre de plateformes important et avec des interfaces homme-machine fort différentes (Kolski, 2001) : interfaces d'acquisition du type boutons ou molettes, interfaces de

restitution du type écrans ou LED témoins, interfaces combinées du type écrans tactiles ou commandes à retour d'effort, etc. S'il semble, aujourd'hui, y avoir un consensus autour du fait qu'il soit impossible de tout modéliser avec un seul type de modèle, la question qui se pose est de savoir comment faire interagir les différents modèles entre eux. Pour répondre à cette problématique, l'OMG (OMG, 2006) propose d'utiliser un métamétamodèle commun appelé *Meta Object Facility* (MOF) permettant de définir la syntaxe et la sémantique d'un langage de modélisation. Ensuite, pour faciliter la communication entre les modèles, l'OMG (OMG, 2007) propose d'utiliser un langage d'échange intitulé *XML Metadata Interchange* (XMI). Ces deux éléments étant définis, il devient alors possible de concevoir sur une même base commune (MOF) un ensemble de modèles différents traitant chacun d'un aspect particulier de l'application mais capables de communiquer entres eux. La Figure 1.1 donne un exemple de cette approche globale proposée par l'OMG.

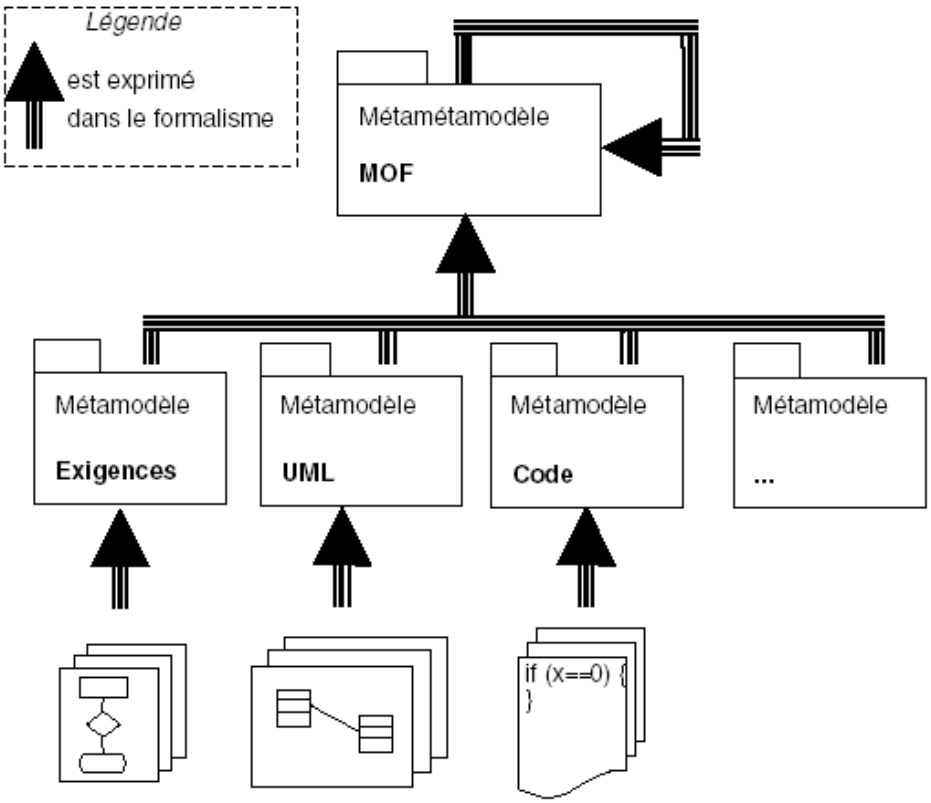


Figure 1.1. Modèle, métamodèle (formalisme de modélisation) et métamétamodèle (métaformalisme) (Blanc et Salvatori, 2005)

Mais modéliser ne sert à rien si on ne sait pas créer, à partir des modèles, des applications utilisables. C'est justement ce que propose l'approche dirigée par les modèles que nous allons maintenant présenter.

1.1.2 L'approche dirigée par les modèles

L'ingénierie dirigée par les modèles, appelée aussi IDM, est aujourd'hui en passe de devenir le nouveau paradigme en matière de développement d'applications informatiques (Favre *et al.*, 2006) (Hailpen *et al.*, 2006) (Schmidt, 2006). L'objectif de cette approche est de permettre la création d'applications informatiques à partir de modèles conceptuels et surtout à partir d'un assemblage de modèles conceptuels ; chaque modèle prenant en charge une ou plusieurs problématiques métier bien définies. Du point de vue de l'approche systémique des applications informatiques, pour une problématique métier bien déterminée, chaque modèle représente un point de vue différent sur un même processus métier comme la sécurité des accès, la répartition des tâches entre les intervenants, etc. Cette notion de processus métier étant importante dans le cadre de notre approche, elle sera présentée plus en détail dans le chapitre 1, §1.3.

Une notion importante de l'IDM est celle de modèle conceptuel. Pour notre part, nous avons retenu la définition donnée par l'OMG pour le niveau *Computation Independent Model* (CIM) de l'approche *Model Driven Architecture* (OMG, 2003) qui correspond à un niveau de modèles conceptuels : *"Le modèle CIM est une vue du système d'un point de vue indépendant des éléments informatiques permettant de le réaliser. Le modèle CIM ne montre pas les détails de la structure des systèmes. Un modèle CIM est parfois appelé un modèle de domaine et un vocabulaire qui est familier aux experts de ce domaine en question est utilisé pour le spécifier."*

L'objectif des modèles conceptuels est de permettre aux experts du domaine, qui ne sont pas des informaticiens mais des spécialistes métier, de définir eux-mêmes les modèles des processus métier qu'ils utilisent et qu'ils veulent voir repris dans une application informatique. Dans la suite de ce mémoire, pour plus de clarté, nous utiliserons le terme d'expert métier à la place d'expert de domaine.

Le passage des modèles conceptuels aux applications concrètes, manipulables par des utilisateurs finaux, se fait à travers une succession de transformations de modèles s'appuyant sur une approche de type MDA (cf. Figure 1.2). En fait, les transformations de modèles représentent la principale richesse de la méthodologie MDA car ce sont elles qui sont le reflet du savoir-faire et des méthodologies utilisées au sein d'une structure de développement. Voilà pourquoi l'OMG préconise de modéliser les transformations de modèles elles-mêmes afin d'en assurer la qualité et surtout afin d'assurer la pérennité des investissements consentis dans la mise en place d'une démarche MDA. Cette nécessité est d'ailleurs très bien expliquée dans (Clark *et al.*, 2004). Dans le cadre de notre travail, nous nous y intéresserons sous l'angle de l'Interaction Homme-Machine (IHM).

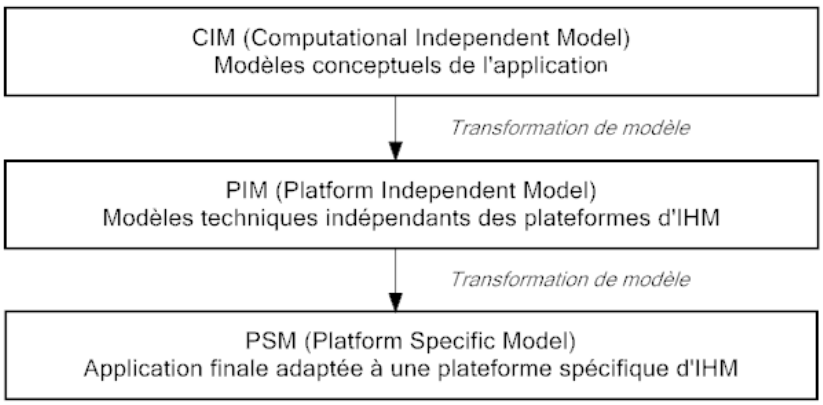


Figure 1.2. Représentation globale de l'Approche MDA vue sous l'angle de l'IHM

À travers l'utilisation de trois niveaux d'abstraction, l'approche MDA a pour but de séparer la logique du métier associée à l'application de la technologie qui sera utilisée pour la réaliser. L'objectif est de permettre de supprimer le lien direct entre les applications et le codage qui leur est associé, facilitant leur inter-opérabilité et les rendant ainsi moins sensibles aux évolutions technologiques. Le passage d'un niveau à un autre se fait à l'aide d'une transformation de modèles permettant à chaque étape d'enrichir les modèles du niveau précédent des informations techniques nécessaires et suffisantes. Cette transformation se fait aujourd'hui de manière automatique ou semi-automatique à l'aide d'outils comme le framework VIATRA (VIsual Automated model TRAnsformations) ou comme ATL (Atlas Transformation Language) qui sont tous les deux présentés en détail sur le site web http://dev.eclipse.org/viewcvs/indextech.cgi/gmt-home/index.html. Les outils permettant de faire des transformations de modèles étant nombreux, on se référera à (Czarrneski et Helsen, 2006) pour une présentation et une classification de ceux-ci. Dans l'approche MDA, les modèles de niveau CIM sont totalement indépendants des techniques et des plateformes qui seront utilisés pour leur réalisation. Les modèles de niveau PIM sont des modèles techniques indépendants de la plateforme finale sur laquelle sera exécutée l'application. Le niveau PSM représente pour sa part l'application finale adaptée à une plateforme d'utilisation bien spécifique (PDA, téléphone portable, PC, etc.). Pour pouvoir passer du niveau PIM au niveau PSM, on associe généralement, à travers une approche en Y (cf. Figure 1.3), au modèle de niveau PIM un modèle de plateforme et c'est la transformation des deux modèles qui donne le modèle de niveau PSM (OMG, 2003).

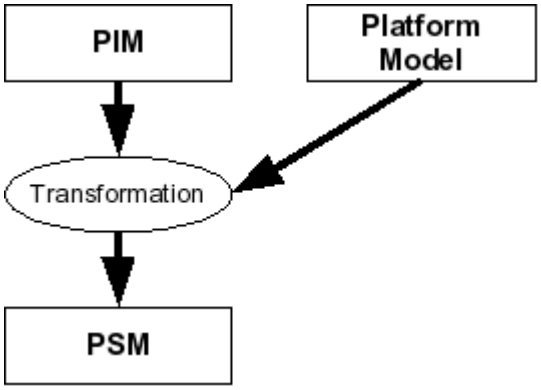


Figure 1.3. Modèle en "Y" du passage du niveau PIM au niveau PSM dans l'approche MDA (OMG, 2003)

1.1.3 Les principes de transformations de modèles, préconisés par l'OMG, pour passer du niveau CIM à l'application concrète de niveau PSM

Pour l'OMG, dans une approche MDA, les liens entre les différents niveaux de modèles sont établis à travers l'exécution de transformations de modèles qui se font de manière entièrement automatique. Les transformations principalement mises en avant sont les transformations CIM vers PIM et PIM vers PSM. Néanmoins, dans une approche MDA, les transformations inverses sont aussi envisageables bien que généralement peu étudiées et peu utilisées.

Si les modèles représentent un élément important dans le cadre d'une approche MDA, les transformations de modèles sont au coeur du processus méthodologique de création d'applications. Ce sont elles en effet qui permettent d'adapter les modèles à l'infrastructure technique réelle dans laquelle les applications vont être utilisées. Ainsi, elles contiennent l'ensemble du savoir-faire et de l'expérience associés au développement d'applications au sein de la structure, de la société, qui les utilise. Ceci signifie que si les transformations de modèles peuvent s'appuyer sur une base générique pour tous les utilisateurs, elles doivent être adaptées, de manière plus ou moins importante, à chaque environnement de conception d'applications et donc à chaque entreprise.

Aussi, pour formaliser ces transformations, l'OMG préconise de les modéliser en les considérant comme des applications particulières qui permettent de passer d'un modèle à un autre. De cette manière, il devient possible d'envisager une génération automatique du code de la transformation à travers une approche de type MDA. Ceci en facilite la maintenance et l'évolution dans le temps en permettant de prendre en compte de nouveaux environnements techniques ou de nouvelles contraintes techniques et/ou fonctionnelles.

Une des particularités de l'approche MDA, telle qu'elle est proposée par l'OMG, est de représenter l'ensemble des éléments associés à une application comme pouvant faire l'objet d'une modélisation. Ainsi, il est possible de définir des modèles de déploiement, des modèles d'intégration ou encore des modèles de test. En fait, dans une approche MDA tout est modèle, mais chaque modèle est spécifique à la problématique qu'il traite. Si ceci suppose la possibilité d'utiliser de nombreux types de modèles différents pour pouvoir prendre en compte l'ensemble des problématiques associées à la création et à la modélisation d'une application, ceci suppose aussi d'utiliser un formalisme de base commun au niveau de la modélisation de manière à permettre d'établir des liens entre les différents modèles.

Pour cela, dans le cadre de l'approche MDA, l'OMG propose de baser l'ensemble des modèles sur un formalisme de modélisation générique appelé MOF (Meta Object Facility). Celui-ci a pour but de permettre de définir l'ensemble des concepts ainsi que les relations entre les concepts indispensables pour l'expressivité des modèles.

Comme dans une approche MDA tout est modèle, MOF est aussi vu comme un modèle qui doit lui-même pouvoir être modélisé par un autre formalisme qui doit lui-même pouvoir être modélisé et ainsi de suite. En théorie, ce processus de création de formalisme de modélisation peut s'enchaîner de nombreuses fois jusqu'à obtenir un formalisme suffisamment générique qui soit capable de se modéliser lui-même. Et, c'est précisément le cas de MOF qui représente un méta-métamodèle dans le

sens où il permet de définir un formalisme de modélisation qui permet de créer des formalismes de modélisation.

Les formalismes de modélisation qui sont créés à partir de MOF sont des formalismes qui permettent de créer des modèles. En ce sens, ce sont des métamodèles car ils ne peuvent être directement utilisés pour modéliser une réalité. Ces métamodèles vont permettre la création de modèles qui pourront être utilisés pour représenter une problématique particulière d'une application informatique. Par transformation de modèles, ces derniers donneront lieu à la génération d'un code applicatif utilisable qui représentera une instance du modèle de l'application pour un contexte technique donné.

Pour permettre d'effectuer des liens entre les différents modèles, l'OMG propose d'utiliser, en partant de l'hypothèse que tous les métamodèles se basent sur MOF, un langage particulier d'échange entre modèles appelé XMI (XML Metadata Interchange). De cette manière, les différents modèles d'une application ne sont plus vus comme une simple juxtaposition de problématiques métier différentes mais comme un ensemble cohérent permettant de représenter l'intégralité d'une application.

Cette approche globale est représentée de manière synthétique dans la Figure 1.4, à travers une approche à quatre niveaux :

- M3 représente un méta-métamodèle de type MOF,
- M2 un métamodèle pour un domaine particulier,
- M1 un modèle associé à une application et/ou une problématique métier particulière,
- M0 une instance du modèle final, c'est-à-dire du code exécutable

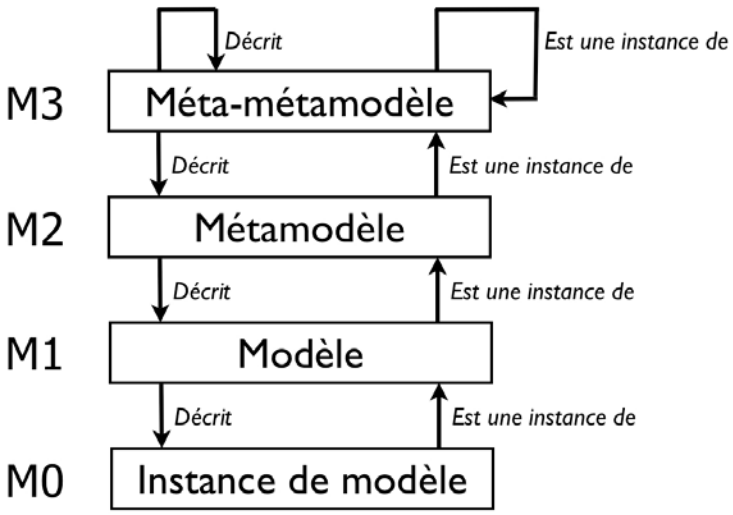


Figure 1.4. La transformation de modèles vue par l'OMG dans le cadre d'une architecture MDA

Dans le cadre de l'approche MDA telle qu'elle est généralement utilisée aujourd'hui, UML est très probablement le métamodèle le plus connu et le plus utilisé. Ainsi, ce métamodèle, qui permet d'élaborer des modèles décrivant des applications orientées objets, permet, par exemple, de préciser qu'une classe UML peut avoir des attributs et des méthodes. UML présente aussi la particularité d'avoir MOF comme méta-métamodèle ; ce qui explique aussi sa large utilisation dans le cadre de MDA.

Au niveau PIM, UML, à travers l'utilisation des différents modèles proposés dans la notation, permet de modéliser une application objet tout en restant totalement indépendant des plateformes d'exécution qui seront utilisée pour l'application finale. En ce sens, il constitue un véritable modèle d'analyse et de conception d'applications.

UML présente aussi la particularité de pouvoir créer et utiliser des profils UML qui permettent d'adapter UML à un domaine métier ou à un domaine technique particulier. Ainsi, il est possible de

définir des profils UML spécifiques à des plateformes d'exécution bien déterminées. Ces profils représentent alors des modèles dépendants des plateformes d'exécution qui correspondent alors parfaitement à des modèles de type PSM.

Afin de faciliter la transformation de modèles, l'OMG a élaboré un standard de transformation pour tous les formalismes basés sur MOF. Ce standard, appelé MOF QVT (Query, View, Transformation), définit le métamodèle permettant la création de modèles de transformation. Ainsi, il devient possible d'envisager des solutions permettant la transformation de modèles UML vers du code Java de manière automatique comme, par exemple, la solution *UML2Java* (disponible sur le site web http://javare.sourceforge.net/index.php).

De manière pratique, avec UML et MOF QVT, la transformation d'un modèle PIM vers du code Java se fera de la façon suivante :

- **Etape 1** : Le modèle UML de niveau PIM est transformé en un profil UML de niveau PSM (exemple de profil : UML pour EJB) à l'aide d'une première transformation.
- **Etape 2** : Le modèle UML de niveau PSM est transformé vers du code Java exécutable à l'aide d'une deuxième transformation en utilisant un outil comme *UML2Java*.

Ceci termine notre présentation rapide de la transformation de modèles dans le cadre d'une approche standard de type MDA en se basant sur les préconisations actuelles de l'OMG.

1.1.4 Synthèse et discussion

Dans ses principes, l'approche IDM apparaît comme un élément important pour arriver à une véritable industrialisation des développements informatiques et donc à une prise en compte globale de toute la complexité associée aux applications interactives. Néanmoins, elle possède aujourd'hui un certain nombre de contraintes et de limitations qu'il convient de prendre en compte.

La première de ces contraintes réside dans le fait que l'interprétation de la notion d'ingénierie dirigée par les modèles n'est pas la même pour toutes les personnes travaillant dans le domaine (Favre, 2004) ; ce qui entraîne à la fois des problèmes de communication entre les différents intervenants mais aussi des visions différentes des solutions possibles qui ne sont pas toujours compatibles. Dans le cadre de nos travaux, et en nous basant sur les travaux de (Favre, 2004) (Favre *et al.,* 2006), nous avons défini l'approche IDM comme devant avoir les caractéristiques suivantes :

- Elle permet de créer des modèles à partir de modèles déjà existants.
- Elle ne doit pas introduire de nouveaux formalismes mais compléter les formalismes existants.
- Elle permet une génération aussi automatisée que possible des applications en s'appuyant sur une architecture MDA.

Ces critères étant posés, il est important de noter que si les travaux sont nombreux dans le domaine, il n'existe pas, à notre connaissance, encore d'outils, que ce soit dans le domaine industriel ou dans le domaine de la recherche, permettant une véritable approche dirigée par les modèles des développements informatiques.

Au niveau de l'approche MDA, bien que celle-ci existe depuis plusieurs années (Oya, 2002) (OMG, 2001), elle n'est encore utilisée que trop rarement dans le cadre de la réalisation d'applications et encore moins dans le cadre du développement d'IHM. Et, lorsqu'elle est utilisée, c'est bien souvent de façon incomplète malgré le fait que de nombreux travaux en aient montré l'intérêt (Eyermann *et al.*, 2004) (Frankel, 2005) (Muller *et al.*, 2005) notamment dans le domaine des IHM (Pérez-Medina *et al.*, 2007) (Sottet *et al.*, 2006) (Sottet *et al.*, 2007) ; cf. Tableau 1.1. En fait, ce qui limite aujourd'hui son utilisation, c'est en partie le manque d'outils et de méthodes permettant de réaliser une modélisation conceptuelle des applications (Tariq et Akhter, 2004*)* indépendante des techniques utilisées pour leur réalisation ; la plupart des solutions existantes se contentant d'utiliser uniquement les niveaux PIM et PSM ou alors limitant l'approche à un simple assemblage de composants déjà définis. Étant donné qu'il n'existe pas, à notre connaissance, d'outillage complet d'une approche IDM dans le domaine de la recherche, on trouvera dans le Tableau 1.1, une rapide comparaison de quelques

outils commerciaux existants aujourd'hui sur le marché qui sont vendus comme étant compatibles avec une approche IDM.

Tableau 1.1. Comparaison des différents outils MDA/IDM du marché

Produit	Société	Approche IDM	Modélisation des IHM
Arc Styler	Interactive Objects	Oui mais ne permet que l'assemblage de composants déjà existants au niveau conceptuel à l'aide de modèles métier.	Non
Blu Age	Blu Age Software	Oui mais sans prise en compte des modèles conceptuels (uniquement PIM et PSM) : génération d'applications Java EE et .Net à partir de modèles UML.	Oui via la création de maquettes en XHTML
CodeFluent	SoftFluent	Oui mais ne permet que l'assemblage de composants déjà existants au niveau conceptuel à l'aide de modèles métier.	Non
Leonardi	Lyria	Oui mais fortement orienté vers les IHM ; de nombreuses autres fonctionnalités n'étant pas prises en compte.	Oui. Spécifiquement orienté vers la création d'IHM
Objecteering	Objecteering Software	Oui mais ne permet que l'assemblage de composants déjà existants au niveau conceptuel à l'aide de modèles métier.	Non
OptimalJ	Compuware	Oui mais uniquement au niveau PIM et PSM à travers l'utilisation de modèles de développement Java et de règles métier.	Non
Rational	IBM	Oui mais ne prend en compte la notion de modèles conceptuels qu'à travers la notion de patrons de conception et/ou à travers le principe d'assemblage de composants.	Non
Select Solution Factory	Select Business Solution	Oui mais ne permet que l'assemblage de composants déjà existants au niveau conceptuel à l'aide de modèles métier.	Non

En conclusion, si l'approche MDA et plus encore l'approche IDM paraissent prometteuses pour résoudre les nombreuses problématiques associées à la création d'applications interactives, elles ne disposent pas encore de l'outillage nécessaire pour exprimer pleinement leur potentiel. En fait, les outils sont aujourd'hui trop liés à la technique et bien souvent à l'approche par composants. Ainsi, les solutions existantes sont généralement associées aux architectures techniques de type Service Oriented Architecture (SOA) qui se limitent, bien souvent, à la génération d'applications web.

1.2 Les outils et méthodes de modélisation des applications interactives

1.2.1 Définitions et principes

Dans un monde où l'utilisateur peut accéder à une quantité sans cesse croissante d'informations et d'applications, il peut aussi très facilement passer d'un fournisseur d'applications à un autre si l'expérience personnelle qu'il a avec l'application ne s'avère pas positive. Or, comme le souligne (Raskin, 2000), "*pour l'utilisateur, le produit (l'application) c'est l'interface*". Développer une bonne interface homme-machine est donc aujourd'hui indispensable si on veut garder et fidéliser les utilisateurs d'une application et surtout si on veut permettre à l'utilisateur d'accéder facilement à l'information dont il a besoin au moment où il en a besoin. Aussi, il est aujourd'hui indispensable de

prendre en compte l'ensemble des éléments liés à l'interface homme-machine dans le cadre de la modélisation

Du point de vue applicatif, une interface peut se définir à deux niveaux (Kolski, 2001). Le premier niveau correspond aux éléments statiques d'interaction ; c'est-à-dire aux éléments physiques qui permettent d'effectuer les interactions (boutons, zones de saisie, zones tactiles, etc.). Le deuxième correspond à l'aspect dynamique des interactions, c'est-à-dire à la description de l'ensemble des interactions élémentaires qui sont nécessaires pour réaliser une action du point de vue métier comme, par exemple, remplir et valider un formulaire.

Modéliser une application interactive revient donc à être capable de prendre en compte à la fois les aspects statiques des interactions mais aussi les aspects dynamiques. Dans le cadre d'une approche dirigée par les modèles, ceci doit se faire en utilisant plusieurs types de modèles différents ; chaque modèle représentant un point de vue particulier sur l'application (Ribaud *et al.*, 2005). Aujourd'hui, dans le domaine des IHM, il existe un certain nombre de méthodes et d'outils qui permettent ce type de modélisation. Nous allons présenter les principaux dans la partie suivante.

1.2.2 Étude des méthodes et outils de modélisation des applications interactives

En matière de modélisation d'applications dotées d'une interface graphique, les derniers travaux dans le domaine montrent que la génération d'une IHM passe par l'utilisation d'outils adaptés. Ainsi, (Bardon *et al.*, 2003) proposent, à travers l'utilisation de la méthode OVID (Objects, Views and Interaction Design), de prendre en compte cinq vues différentes pour modéliser une application de manière abstraite : les objets utilisateur, les tâches, les flots de tâches, les états des objets et les vues abstraites de l'interface. (Sottet *et al.*, 2005) proposent, pour leur part, de prendre en compte cinq métamodèles différents : les tâches utilisateur, les concepts du domaine de l'application, l'espace de travail, les interacteurs et le programme lui-même. De son côté, (Hanumansetty, 2004) estime que le développement d'applications graphiques doit être capable de prendre en compte trois contextes complémentaires : la plateforme utilisateur, l'environnement et les caractéristiques de l'utilisateur. Or, selon, lui, il est impossible d'exprimer ces trois contextes en se basant uniquement sur l'utilisation d'une modélisation s'appuyant uniquement sur le langage UML ; ce qui traduit bien la complexité d'appréhender une application dans sa globalité. Il est intéressant de noter ici que (Bastide et Palanque, 2003) sont arrivés à la même conclusion sur les manques du langage UML pour modèliser une application interactive ; celle-ci nécessitant l'utilisation de plusieurs modèles qui ne peuvent pas tous être pris en charge par UML.

Pour ce qui est de la nature des modèles eux-mêmes, (Molina, 2004) propose d'utiliser des modèles simples basés sur un minimum de concepts plutôt que des modèles trop complexes censés être capables de prendre en compte toutes les spécificités qu'il est possible de rencontrer dans les différentes typologies d'application. Selon lui, il vaudrait mieux définir plusieurs modèles différents spécialisés en fonction du type d'application que de définir un seul modèle global censé être capable de permettre la modélisation de tous les types d'applications ; ce qui pose la question de savoir comment faire le lien entre les différents modèles.

Dans l'optique d'une vision plus globale des choses, parmi l'ensemble des travaux réalisés dans le domaine, un des modèles globaux les plus intéressants est le modèle Wisdom (Whitewater Interactive System Development with Object Models) proposé par (Jardim Nunes, 2001) dont les principaux apports sont les suivants :

- Un modèle architectural
- Une méthode de développement
- Une intégration complète des interfaces utilisateur

Un schéma synthétique des différents modèles de l'approche Wisdom et des relations entre eux est présenté dans la Figure 1.5.

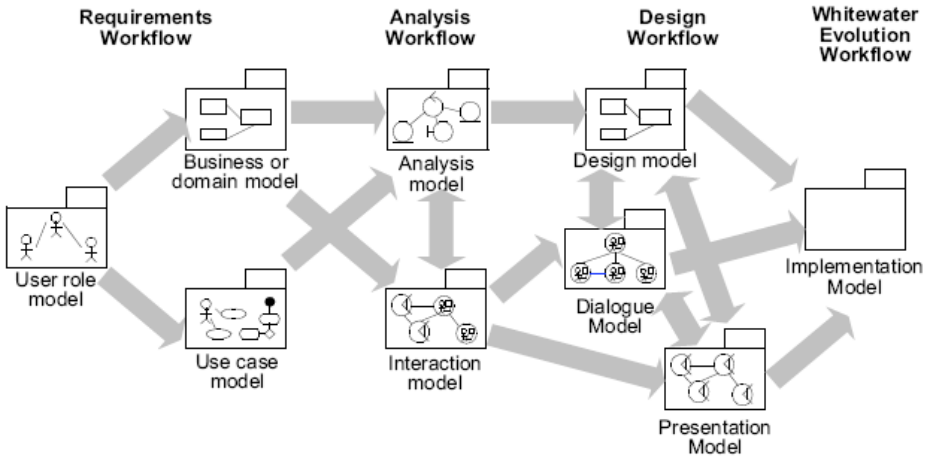


Figure 1.5. Architecture globale de l'approche Wisdom (Jardim Nunes, 2001)

Cette approche était une des premières propositions de prise en charge d'une modélisation complète d'une application. Elle a par la suite été complétée, afin de mieux gérer les aspects interactifs, par des profils UML basés sur l'utilisation de patrons d'éléments graphiques (Jardim Nunes, 2003). Elle a aussi donné lieu, devant la difficulté d'adapter des outils existants, au développement d'une application permettant de modéliser graphiquement une interface graphique indépendante de tout environnement d'exécution : c'est l'outil CanonSketch présenté par (Campos et Jardim Nunes, 2004).

Une autre approche bien établie aujourd'hui est celle proposée autour d'UsiXML (User interface eXtensible Markup Language) qui est un langage de type XML adapté à la modélisation d'applications utilisant des interfaces graphiques (http://www.usixml.org). Celui-ci s'appuie sur le framework CAMELEON (Calvary *et al.*, 2003) qui, en définissant un niveau « Tâches et concepts », permet de prendre en charge les tâches de l'utilisateur mais aussi tout un ensemble de concepts annexes permettant de modéliser les utilisateurs ou encore l'environnement. Dans UsiXML, la modélisation des aspects dynamiques des IHM est confiée à des modèles basés sur l'utilisation de ConcurTaskTree (CTT) proposée par (Mori *et al.*, 2002) (Paterno *et al.*, 1997). Au niveau de la modélisation statique des IHM, UsiXML propose une modélisation conceptuelle des interfaces graphiques afin de pouvoir s'affranchir de la diversité des langages, outils et plateformes pouvant être utilisés pour créer physiquement une interface graphique. Pour cela, UsiXML propose une approche compatible MDA (Vanderdonckt, 2005) se basant sur un ensemble de tâches et de concepts pour arriver, après un certain nombre de transformations, à une interface graphique directement utilisable. Un modèle simplifié de cette approche, présenté dans le cadre de la création d'une application multi contextes (pour plusieurs types de plateformes utilisateur), est présenté dans la Figure 1.6.

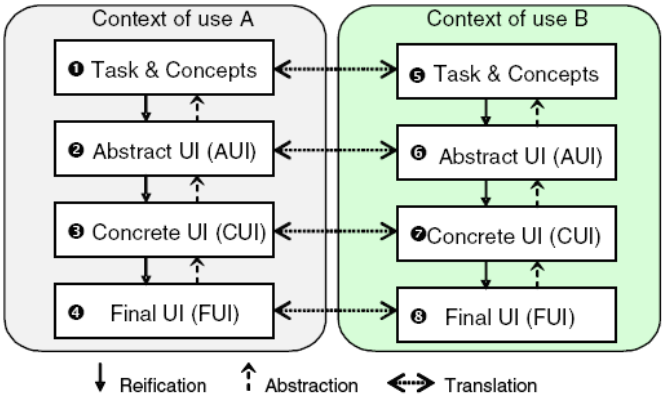


Figure 1.6. Modèle simplifié du framework associé aux interfaces graphiques en UsiXML (Vanderdonckt, 2005)

Aujourd'hui, un des grands intérêts d'UsiXML est de proposer un certain nombre d'extensions et surtout de nombreux outils permettant de modéliser et de générer des interfaces graphiques. Ainsi, on pourra citer l'outil IDEALXML (Montero, 2005) qui permet de prendre en compte un processus itératif de développement d'interfaces graphiques. Mais, cette solution n'est pas parfaite dans le sens où elle ne permet pas de prendre en compte l'intégralité des aspects liés au développement d'une application (gestion de la sécurité des accès, description des différents intervenants, modélisation des échanges d'informations entre intervenants, etc.). Pour une description plus précise d'UsiXML, on pourra se référer à la thèse de (Florins, 2006) et au site Internet http://www.usixml.org. Il est intéressant aussi de noter que des travaux sont actuellement en cours pour normaliser le langage UsiXML au niveau de l'organisme World Wide Web Consortium (W3C) en charge de la définition des standards de développement pour les applications web.

À ce modèle présenté à travers UsiXML, (Schattkowsky et Lohmann, 2005) ont opposé un autre modèle proposant une séparation stricte des aspects visuels de l'interface graphique de ceux liés au traitement des informations (cf. Figure 1.7).

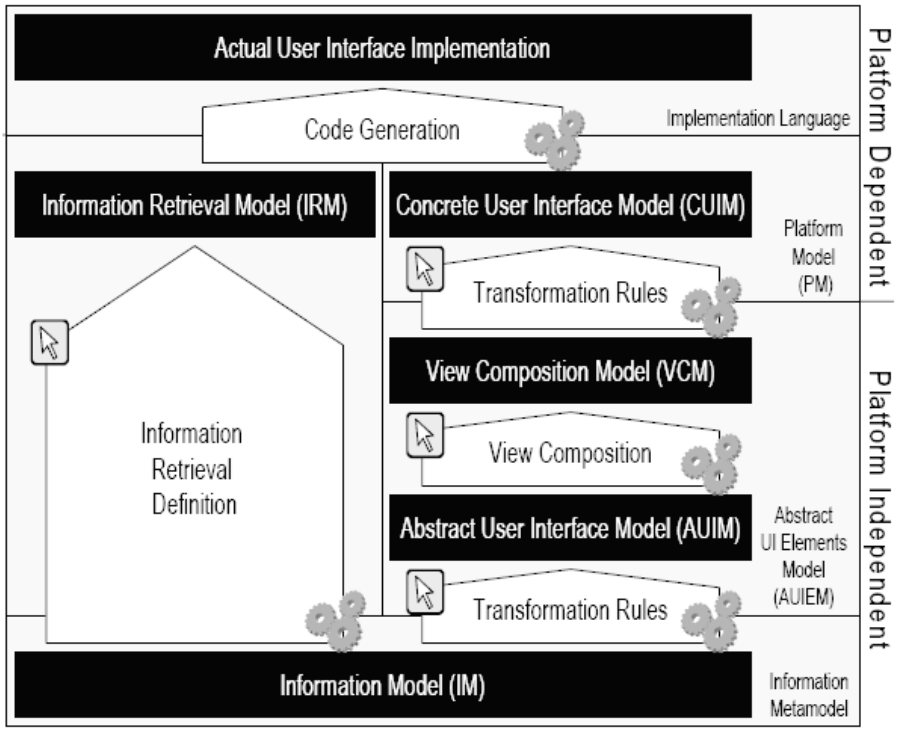


Figure 1.7. Flux de création d'une interface graphique indépendamment des plateformes d'exécution (Schattkowsky et Lohmann, 2005)

On pourra noter que ce modèle global correspond à une approche un peu plus classique de modélisation des interfaces graphiques (type Modèle-Vue-Contrôleur) mais surtout présente l'avantage de proposer des modèles différents pour chaque type de problématique rencontré (séparation des actions de l'interface graphique).

L'approche Amacont (System Architecture for Multimedia Adaptive Web CONTent) de (Hinz et Fiala, 2004), propose, pour sa part, en s'appuyant sur la notion de contexte d'utilisation et de composants d'interaction, une architecture permettant d'avoir des applications web s'adaptant automatiquement aux plateformes d'utilisation. Pour un exemple d'adaptation dans un contexte 3D, on pourra se référer à (Dachselt *et al.*, 2006). Le principe général de l'architecture Amacont est présenté à la Figure 1.8. De manière globale, Amacont propose une approche et une architecture permettant à des applications web de s'adapter à la plateforme utilisée, au lieu où se trouve l'utilisateur et enfin à l'utilisateur lui-même. Tout ceci passe par la création de trois types de modèles qui sont appliquées sur l'application elle-même à travers des transformations de type XML. Ainsi, toute page web affichée à un utilisateur provient d'une page web générique qui est d'abord adaptée au contexte global, puis à

l'utilisateur avant d'être générée dans le langage informatique le mieux adapté à la plateforme de consultation de l'utilisateur.

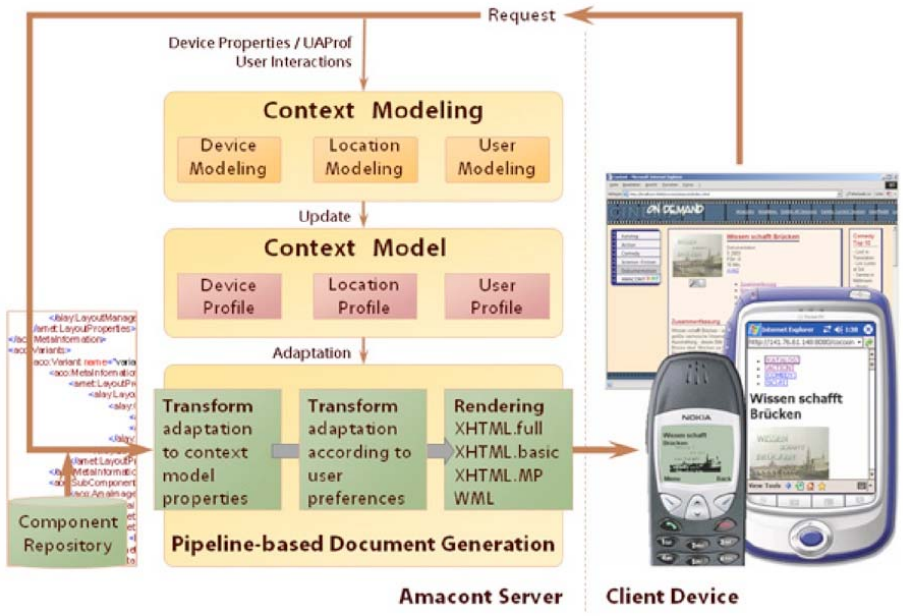


Figure 1.8. Architecture globale de l'approche Amacont (Hinz et Fiala, 2004)

Dans Amacont, qui ne prend en compte que des applications de type WIMP (Windows, Icon, Menu, Pointing device), les éléments statiques d'interactions sont définis suivant plusieurs niveaux de contenu pour tenir compte des différents types interfaces. Ainsi, une image n'est pas définie de manière unique mais sous la forme de plusieurs images différentes possédant des tailles et des niveaux de qualité différents (chaque type d'image étant utilisé pour un type d'interface bien déterminé). Ceci permet de créer pour chaque plateforme, des interfaces adaptées utilisant, pour chaque type d'élément d'interaction le niveau d'élément d'interaction le plus adapté (cf. Figure 1.9).

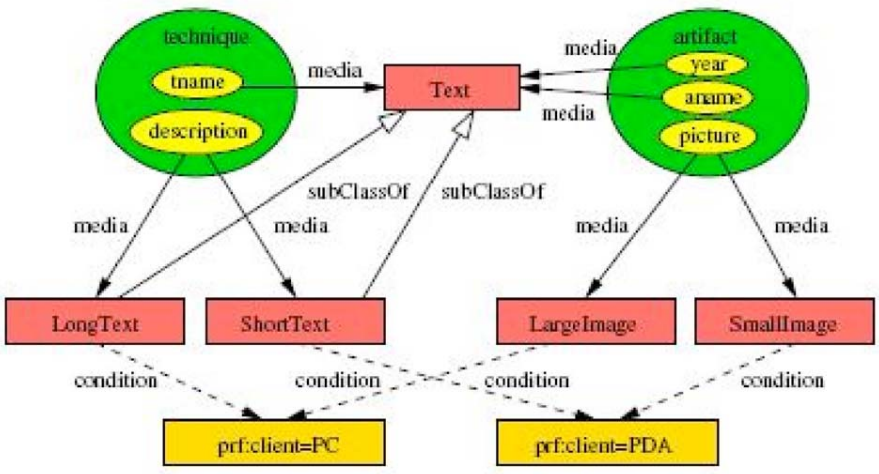


Figure 1.9. Principe d'adaptation des interfaces dans Amacont (Hinz et Fiala, 2004)

Enfin, on terminera cette présentation de l'existant en parlant d'un certain nombre de travaux qui se sont axés à démontrer toute l'importance de l'utilisation de patrons de développement pour la création d'interfaces graphiques. Dans ces travaux, la notion de patron se base sur la notion générique définie par (Alexander *et al.*, 1970) : "*chaque patron décrit un problème qui se manifeste constamment dans notre environnement, et donc décrit le cœur de la solution à ce problème, d'une façon telle que l'on puisse réutiliser cette solution des millions de fois, sans jamais le faire deux fois de la même manière*".

Ainsi :

- (Schlee et Vanderdonckt, 2004) ont montré qu'il est possible de créer des applications dotées d'interfaces graphiques de manière entièrement automatique en passant par de l'assemblage de composants réutilisables.
- (Nichols et Faulring, 2005) ont montré qu'une génération automatique d'interfaces graphiques est tout à fait réalisable si on se limite à des typologies bien particulières d'application.
- (Forbrig et *al.*, 2004) proposent une architecture de développement uniquement basée sur l'utilisation de patrons de développement (cf. Figure 1.10) ; cette dernière approche étant particulièrement indiquée dans le cadre d'une démarche de type IDM. Dans cette approche, chaque application est d'abord modélisée par un modèle de tâches, un modèle d'utilisateur et un modèle de plateforme. Ces modèles sont ensuite transformés de manière à séparer la partie directement liée aux interfaces (modèle de dialogue) de la partie purement applicative (modèle de classes). Les modèles de classes et de dialogue sont alors transformés à nouveau en se basant sur l'utilisation de patrons de conceptions en se basant sur le principe que chaque modèle initial correspond à un patron de conception déjà existant, à un assemblage de patrons de conception ou alors à une adaptation minime d'un patron existant. Une fois ces transformations effectuées, l'application est générée à partir d'éléments applicatifs déjà existants correspondant aux différents patrons de conceptions utilisés.

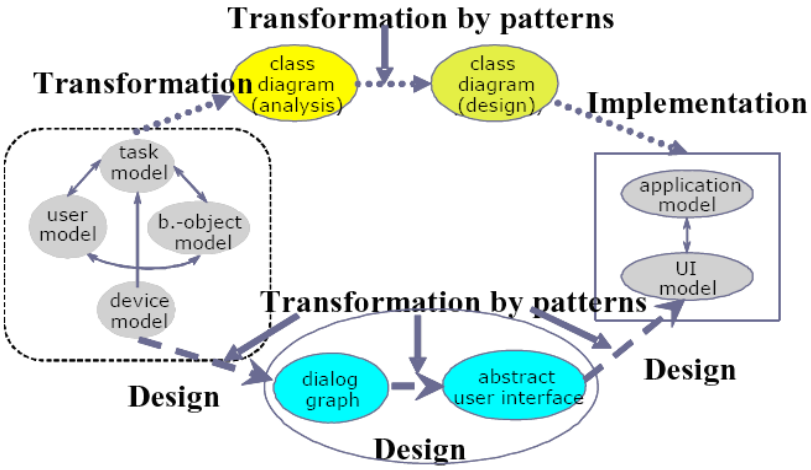


Figure 1.10. Modèle de transformation pour la génération d'interfaces graphiques (Forbrig *et al.*, 2004)

De ces différents travaux, basés sur les patrons de conception, on retiendra principalement :

- Qu'il est possible d'avoir une génération automatique d'interfaces graphiques à partir de patrons de conception mais que celle-ci n'est possible que si on dispose de suffisamment de patrons de conception pour représenter l'ensemble des problématiques pouvant exister au niveau d'une application (Forbrig et *al.*, 2004).
- Que l'utilisation de patrons de développement nécessite de séparer la partie applicative de l'interface graphique (Forbrig et *al.*, 2004) (Nichols et Faulring, 2005). Cette séparation permet de :
    - Limiter le nombre de patrons de conception à utiliser lors des transformations. Ainsi, un patron mélangeant partie applicative et interface graphique ne pourra être utilisé que pour un type de problématique bien précis alors que des patrons séparés au niveau graphique et au niveau applicatif pourront être assemblés de différentes manières pour résoudre plusieurs types de problématiques.
    - Simplifier les patrons de conception en ne les faisant traiter qu'un seul type de problématique.

Ces derniers travaux terminent notre tour d'horizon des outils et méthodes existants aujourd'hui pour modéliser des applications interactives. Ce domaine de recherche étant très riche, les outils et méthodes présentées dans cette partie ne sont que quelques exemples représentatifs des solutions de modélisation proposées pour résoudre les différentes problématiques liées aux environnements multiples et aux interfaces multiples. Pour une présentation un peu plus complète de l'ensemble des solutions ou pistes de solutions existantes, le lecteur pourra se reporter au document de synthèse de (Seffah et Javahery, 2004) qui reprend la plupart des travaux dans le domaine.

1.2.3 Synthèse et discussion

Si toutes les solutions présentées dans la partie précédente (cf. chapitre 1, §1.2.2) permettent de prendre en compte, dans une certaine mesure, l'ensemble des éléments associés aux IHM, elles présentent aussi un certain nombre de limites et de contraintes qui ne permettent pas de les qualifier comme étant des solutions universelles (cf. Tableau 1.2).

Tableau 1.2. Tableau de synthèse sur les approches Wisdom, UsiXML, Schattkowsky et Lohmann, Amacont et Forbrig

	Approche Wisdom	**Approche UsiXML**	**Approche Schattkowsky et Lohmann**	**Approche Amacont**	**Approche Forbrig**
Compatible avec l'approche MDA	Non	Oui pour les aspects relatifs à la modélisation des IHM statiques ; l'aspect dynamique n'étant que très partiellement pris en compte dans l'approche UsiXML du point de vue MDA.	Partiellement car le modèle d'information ne suit pas les principes de l'architecture MDA.	Non validé dans le cadre des IHM.	Non car plus centré sur l'utilisation des patrons de conception que sur une architecture bien précise.
Les types d'applications prises en compte (avec ou sans IHM)	En théorie tous les types d'applications disposant d'une interface de type WIMP.	En théorie tous les types d'application mais surtout les applications de type web dans le cadre des derniers travaux (Martinez, 2007).	N'a pas donné lieu, à notre connaissance, à une réalisation pratique.	Centrée principalement sur les applications de type WIMP orientées web.	Présentée uniquement, du point de vue d'un modèle de tâches, dans le cadre de la création de graphes de dialogues hiérarchiques ; ce qui ne permet pas de généraliser l'approche.
Les différents types de modèles pris en compte (de manière globale)	Ensemble des modèles liés aux applications interactives.	Très centrée sur les modèles d'IHM statiques et dynamiques. Ne prend pas en compte des points comme la sécurité des accès ou encore la modélisation des données.	Très centrée sur les modèles d'IHM statiques et dynamiques.	Principalement centrée sur les modèles permettant de faire une adaptation contextuelle des applications par rapport aux plateformes de consultation. Les modèles des utilisateurs, de sécurité, etc., ne sont pas pris en compte.	Ne propose pas une véritable approche par modèle mais une adaptation des IHM en utilisant des transformations basées sur des patrons de conception modifiables.

	Approche Wisdom	**Approche UsiXML**	**Approche Schattkowsky et Lohmann**	**Approche Amacont**	**Approche Forbrig**
Prise en compte des modèles statiques d'IHM (modèles permettant de définir l'ensemble des éléments d'interaction (boutons, zones de saisie, zone de texte, image, etc.))	Oui à travers le modèle de présentation.	Oui à travers le langage UsiXML.	Oui mais ne fournit pas de description sur les outils et langages utilisés.	Oui à travers les adaptations contextuelles aux plateformes d'utilisation.	Oui mais de manière indirecte car les modèles d'interfaces doivent respecter certains critères pour pouvoir être transformés, grâce à l'utilisation de patrons de conception, en interfaces directement utilisables.
Prise en compte des modèles dynamiques d'IHM (modèles permettant d'associer des tâches à l'ensemble des éléments d'interaction)	Oui à travers les modèles d'interactions et les modèles de dialogue	Oui, principalement à travers le modèle de tâche.	Oui mais ne fournit pas de description sur les outils et langages utilisés.	Pas spécifiquement car l'approche est surtout centrée sur l'adaptation des interfaces aux plateformes d'utilisation.	Non pas spécifiquement
Dispose d'une méthode spécifique de modélisation des IHM	Pas spécifiquement. S'appuie sur la notation UML et plus particulièrement sur les profils "UML" pour modéliser les IHM.	Oui car UsiXML est centré sur la modélisation des IHM.	Reprend en grande partie l'approche de modélisation des IHM proposée par UsiXML.	Oui mais limitée à la prise en compte des différents types de plateformes du point de vue statique.	Oui mais principalement basée sur l'adaptation des interfaces aux différents types de plateformes.
Dispose d'outils utilisables pour créer des modèles	Oui, à travers les outils CanonSketch (http://dme.uma.pt/projects/canonsketch) pour la création de modèles abstraits d'interfaces et TaskSketch (http://dme.uma.pt/projects/tasksketch) pour la création de modèles de tâche	Dispose de nombreux outils, sur le site web http://www.usixml.org, dont FlowiXML qui permet de modéliser des workflow applicatifs (des tâches métier) avec les IHM associées et de générer les applications pour des plateformes multiples.	Oui, dans le cadre des travaux de recherche, mais les outils ne sont pas librement disponibles.	Oui, dans le cadre des travaux de recherche, mais les outils ne sont pas librement disponibles.	Oui, dans le cadre des travaux de recherche, mais les outils ne sont pas librement disponibles.
Situation actuelle en matière de travaux de recherche	Travaux rattachés en partie à UsiXML.	Très actif : de nombreux travaux sont en cours.	A notre connaissance, plus de publication sur le sujet depuis plusieurs années.	Toujours actif principalement dans le domaine du 3D et des applications Web.	Toujours actif au niveau de la définition, de l'utilisation et de la réutilisation des patrons de conception.

De toutes les approches permettant de prendre en charge la modélisation des IHM, statique et dynamique, l'approche UsiXML est aujourd'hui la plus mature en proposant une méthode et un langage complet de modélisation des IHM. De plus, elle dispose de nombreux outils dont certains, comme FlowiXML (Guerrero et Vanderdonckt, 2008), permettent de créer des applications interactives complètes dans un domaine métier bien déterminé.

Néanmoins, le principal inconvénient de l'ensemble de ces méthodes réside dans le fait que les modèles proposés ne permettent de générer qu'un seul type d'applications suivant des schémas bien définis d'interaction (exemple : uniquement des applications web). D'autre part, certains aspects liés à l'environnement de l'application ne sont pas pris en compte comme, par exemple, la modélisation du contexte de l'application (prise en compte du temps, du lieu géographique, des événements extérieurs,

etc.). De la même manière, la modélisation des systèmes externes comme, par exemple, des applications tierces ou encore des systèmes techniques comme des imprimantes ne sont pris en compte dans aucune des approches présentées précédemment. Enfin, la modélisation des données au niveau conceptuel n'est que très peu abordée ou alors uniquement à travers des modèles très simplifiés. Quant à la personnalisation des informations et des contenus, aucune des approches présentées n'aborde la question.

Ces manques importants ne permettent pas de rendre ces approches utilisables dans un contexte général et surtout pas pour des applications où la personnalisation des informations fait partie des critères primordiaux du point de vue fonctionnel. En fait, ce qui manque c'est une approche de modélisation permettant de prendre en compte tous les aspects d'une application aussi bien du point de vue des IHM que du point de vue de la prise en compte des fonctionnalités métier, du contexte d'exécution de l'application ou encore de la personnalisation qui sera présentée plus en détail dans le chapitre 2 de ce mémoire. En ce qui concerne la prise en compte des fonctionnalités métier, une des solutions possibles pourrait consister, comme nous allons maintenant le présenter, à utiliser une première approche basée sur les processus métier.

1.3 Les processus métier dans le cadre de la modélisation des applications interactives

1.3.1 Définition et principes

Le processus métier étant au coeur de l'approche IDM (cf. 1.1.2), il est indispensable d'en donner une définition. Pour (Johansson *et al.*, 1993) : *"Un processus métier est un ensemble d'activités qui prend un élément en entrée et le transforme en un élément de sortie. Idéalement, la transformation qui se passe durant le processus doit ajouter de la valeur à l'élément en entrée et créer un élément de sortie qui est plus utile et plus adapté aux besoins du destinataire que celui-ci se trouve en amont ou en aval."*. Pour (Davenport, 1992) :*"Un processus métier est un ensemble d'activités structurées et mesurables conçues pour produire une sortie spécifique pour un client ou un marché déterminé. Ceci impose d'étudier en détail la façon de faire le travail au sein d'une organisation et pas le résultat produit par ce travail. Un processus est donc un assemblage ordonné d'activités dans le temps et l'espace avec un début, une fin, et des éléments d'entrée et de sortie clairement définis : c'est une structure pour l'action […]. Avoir une approche processus implique d'adopter un point de vue client."*

Du point de vue logique, un processus métier est une succession de tâches, devant chacune être réalisée par un individu ou un système technique, permettant d'atteindre un but métier comme, par exemple, la recherche d'un itinéraire pour se rendre d'un point A à un point B en utilisant des transports collectifs. L'exécution du processus métier lui-même se réalise dans le cadre d'un workflow métier tel que le défini le Workflow Management Coalition (WfMC). Ainsi, pour (WfMC, 1999) un workflow n'est rien d'autre que l'ensemble des éléments permettant d'automatiser un processus métier.

Dans la partie suivante, nous allons présenter l'ensemble des éléments différenciant l'approche à travers les processus métier des autres approches plus classique de modélisation des tâches utilisateurs (provenant particulièrement du domaine de l'interaction homme-machine).

1.3.2 L'utilisation des processus métier dans le cadre de la modélisation : comparaison par rapport aux autres approches

Dans le cadre de la modélisation d'une application interactive, les processus métier peuvent n'être vus que comme une extension des modèles de tâche comme :

- K-MADe (Baron *et al.*, 2006) qui permet à l'aide de modèles de tâche de représenter et d'évaluer du point de vue ergonomique des IHM.

- ConcurTaskTrees (CTT) (Paterno, 1999), qui permet de représenter les interactions homme-machine comme des successions de tâches qui s'effectuent dans le temps (agencées selon un ensemble d'opérateurs temporels).
- UAN (User Action Notation) de (Hartson *et al.,* 1990) qui permet d'associer aux tâches des éléments physiques d'interaction disponibles sur la plateforme technique étudiée. L'ensemble est représenté dans un tableau dans lequel la première colonne contient la tâche d'interaction effectuée par l'utilisateur, la deuxième le résultat obtenu au niveau de l'interface et la troisième l'état global du système.
- Goals, Operator, Methods and Selection Rules (GOMS) de (Card *et al.*, 1983) qui permet de décomposer les interactions avec les applications en actions élémentaires d'interaction (actions physiques, cognitives ou perceptives) ; ces actions élémentaires définissant un framework permettant une étude approfondie des interfaces.
- Hierarchical Task Analysis (HTA) de (Annett et Duncan, 1967) dont l'objectif est de décomposer chaque tâche en sous tâches et ceci jusqu'à arriver à des tâches élémentaires ; ce qui permet d'établir un arbre hiérarchique de tâches pour chaque application.

Mais, comme nous le verrons dans la suite de ce chapitre, les processus métier ont été conçus pour être plus complets que les modèles de tâches de manière à permettre, en théorie, de prendre en compte l'ensemble des éléments nécessaires pour réaliser un but métier bien déterminé : les intervenants humains, matériels et logiciels, les tâches élémentaires, les règles métier, les documents, les échanges verbaux, etc.

Dans un article de (Limbourg et Vanderdonckt, 2003), une comparaison des principaux modèles de tâche, utilisés dans le cadre de la modélisation d'applications interactives, a permis d'établir que chaque modèle présente des avantages et des inconvénients mais surtout que chaque modèle est adapté à un contexte déterminé. Ainsi, il n'existerait pas de modèles de tâche universels permettant de prendre en compte l'ensemble des tâches associées à une application interactive. C'est cette universalité que cherche à atteindre l'approche par les processus métier.

À côté de ces modèles de tâche spécifiques aux IHM, il existe d'autres modèles de tâche plus orientés vers l'utilisation de composants comme, par exemple, les modèles proposés par la notation UML (OMG, 2005) ou encore l'utilisation de réseaux de Petri dans le cadre de la modélisation de workflows applicatifs (Injun *et al.,* 2002). En fait le problème avec ce type d'approche réside dans le fait que les éléments d'interaction sont rarement identifiés en tant que tels dans les tâches métier et qu'il est impossible de les traiter de manière individuelle ni de les détailler. D'autre part, si l'utilisation d'une notation basée sur les réseaux de Petri permet de faire des simulations sur les modèles, celles-ci se limitent bien souvent à la simple vérification d'absence de point de blocage, de branche morte ou encore de boucle infinie. Quant aux outils basés sur le langage UML, ils ne permettent pas d'exécuter les modèles sans passer par la génération physique d'une application finale. Tout ceci en limite fortement l'utilisation dans le cadre d'une approche dirigée par les modèles qui serait centrée sur la conception d'applications interactives.

En fait, dans le cadre d'une approche utilisant une modélisation conceptuelle des applications, l'utilisation des processus métier présente de nombreux avantages qui seront présentés plus en détail par la suite :

- Facilité de prise en main par des experts métier.
- Durée de vie des processus métier plus longue que celle des applications.
- Possibilité de gérer les processus comme des transactions :
  - Chaque processus peut être arrêté à tout moment de son exécution pour être repris plus tard.
  - Chaque processus peut être encapsulé de manière à ne permettre de prendre en compte les modifications effectuées pendant son exécution que lorsque celui-ci se termine avec succès ; ce qui permet d'assurer la cohérence des données et l'exécution en tout ou rien.
- Possibilité de représenter l'ensemble des flux d'information des différents intervenants y compris les flux informels de type échanges verbaux.

D'après la société de conseil spécialisée en gestion de projets (Centreline Solutions Inc., 2005), parmi les dix principales causes d'échec dans un projet, on trouve une mauvaise définition des besoins. Ceci peut être dû soit à une mauvaise expression des besoins de la part des utilisateurs, soit à une mauvaise compréhension des besoins par l'équipe technique en charge du projet. La Figure 1.11 montre, à travers un exemple pédagogique simple, ce que peut donner un problème de communication dans un projet.

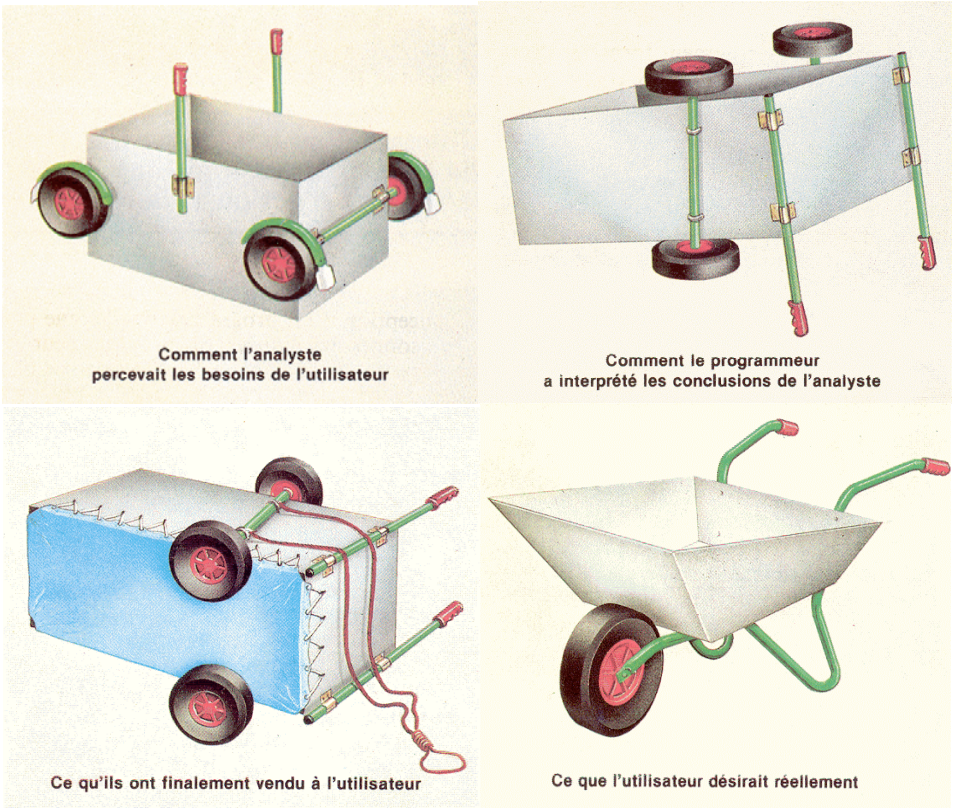


Figure 1.11. Les problèmes de communication au sein d'un projet : exemple pédagogique

Aujourd'hui, pour pouvoir appréhender la complexité des applications, et en théorie pour pouvoir faciliter la communication entre les différents intervenants des projets, on essaye d'utiliser au maximum des outils et techniques de modélisation qui puissent être partagés et compris par le maximum d'intervenants sur un projet. Un des problèmes des outils et méthodes de modélisation actuels réside dans le fait que les personnes qui connaissent le mieux les besoins du point de vue applicatif sont les experts métier qui ont souvent beaucoup de mal à comprendre des modèles qui ont été conçus par des informaticiens et sont bien souvent incapables de les manipuler (Selic, 2003) (Jardim Nunes et Campos, 2004). On pourra noter qu'UML, en étant trop lié aux concepts de la programmation orientée objets ne permet pas de créer des modèles vraiment indépendants des solutions techniques utilisées. De plus, il est loin d'être reconnu comme un langage compréhensible par l'ensemble des experts métier (Cook, 2004) (Henderson-Sellers *et al.*, 2005) (Palano *et al.*, 2006) même si l'utilisation des profils permet d'en simplifier la lecture. Pour faciliter la communication entre les experts métier, les ergonomes et les informaticiens, il est donc indispensable de créer des modèles et outils qui soient destinés aux experts métier et surtout qui soient facilement compréhensibles et manipulables par tous les intervenants (Bernonville *et al.*, 2005).

Pour pouvoir résoudre cette problématique, un groupe de travail regroupant plusieurs spécialistes issus de domaine différents ont créé le Business Process Modeling Notation ou BPMN (BPMI, 2004) qui est un langage, basé sur les réseaux de Petri, qui permet de modéliser, à travers un formalisme précis, tous les types de processus métier. Ce langage a été conçu pour être facilement compréhensible par des experts métier et par des informaticiens ; ce qui devrait, en théorie, limiter les problèmes de communication. Mais plus qu'un outil de modélisation à destination des informaticiens, BPMN a été conçu pour être utilisé par des experts métier ; l'objectif étant de permettre aux utilisateurs de créer eux-mêmes leurs applications.

L'utilisation des processus métier, si elle facilite la communication entre les différents intervenants d'un projet, permet aussi de pérenniser les investissements réalisés dans le cadre des projets informatiques. Ainsi, pour (Frankel, 2003), l'approche MDA et surtout la modélisation conceptuelle des applications sont des éléments indispensables pour pérenniser les investissements informatiques. Ainsi, à travers l'utilisation de modèles conceptuels indépendants des outils informatiques, il devient possible de séparer de manière formelle les aspects métier des technologies qui seront utilisées au niveau des applications. Ceci est d'autant plus intéressant que les technologies évoluent rapidement et que celles utilisées au moment de la conception de l'application ne seront pas les mêmes que celles qui seront utilisées quelques années plus tard lorsque l'application sera remise à jour du point de vue technique. Par contre, ce qui ne changera pas, ce sont les aspects métier liés à l'application. Pour donner un exemple, si l'application associée à l'achat d'un titre de transport dans un distributeur automatique peut utiliser différentes technologies au fil des ans, le processus métier qui est associé à cette opération restera le même pendant cette durée.

La modélisation des applications du point de vue métier, à travers les processus métier, représente donc un avantage indéniable par rapport aux autres types de modélisation car elle permet de capitaliser sur ce que fait une entreprise, c'est-à-dire sur ces processus métier. En dehors des aspects purement informatiques, cette formalisation des processus métier de l'entreprise, d'après (Jeston et Nelis, 2006) permet aussi une meilleure gestion de l'entreprise car on ne peut bien gérer que ce qu'on connaît bien. Elle permet aussi une meilleure productivité des employés ; chacun connaissant exactement ce qu'il a à faire. Enfin, elle permet d'engager facilement des cycles d'amélioration sur l'ensemble des processus métier de l'entreprise et ainsi de découvrir de nouveaux gisements de productivité tout en permettant une adaptation rapide de l'entreprise aux nouveaux besoins qui peuvent survenir sur son ou ses marchés.

Si l'utilisation des processus métier permet de représenter simplement l'enchaînement des tâches pour exécuter un but métier, elle présente aussi l'avantage d'assurer une cohérence logique au niveau de l'exécution de chaque processus métier (Eriksson et Penker, 2000) (Havey, 2005). En effet, un processus métier représente un tout sur lequel il est possible d'envisager un certain nombre d'actions comme :

- Démarrer le processus,
- Arrêter le processus,
- Mettre en veille le processus
- Reprendre le processus mis en veille.

Chaque action ainsi définie permet de s'assurer que le processus est toujours géré de manière cohérente et que les informations qui y sont gérées ne conduisent pas à une instabilité ou à des erreurs dans le système. En fait, ceci permet d'étendre la notion de transaction, telle qu'on la rencontre dans les bases de données, au niveau des processus métier ; ce qui représente une avancée certaine par rapport aux méthodes de modélisation classiques, comme UML qui ne gère pas, dans sa notation de base, la notion de transaction.

Pour donner un exemple dans le domaine des transports, l'achat d'un titre de transport pour un déplacement en train représente un processus métier qui forme un tout logique : l'usager n'étant débité que lorsqu'il reçoit le titre de transport. Ainsi, si le processus est interrompu en cours d'exécution, le titre de transport ne doit pas être édité et l'usager ne doit pas être débité. L'usager peut aussi décider d'interrompre le processus métier lié à l'achat du titre de transport pour aller vérifier une information dans son agenda puis revenir au processus métier et le continuer là où il l'avait interrompu ; ce qui lui permet d'éviter d'avoir à ressaisir des données tout en restant cohérent au niveau du processus métier associé à l'achat du titre de transport.

Enfin, dans les méthodes de modélisation classiques comme UML, les échanges verbaux entre les différents intervenants d'un processus métier sont rarement pris en compte et lorsqu'ils sont pris en compte c'est souvent de manière incomplète. Or, ces échanges verbaux sont parfois indispensables à la bonne compréhension et à la bonne exécution d'un processus métier ; l'ensemble des tâches le constituant ne pouvant pas toujours être automatisé. BPMN, dont une grande partie de la notation sera présentée dans le chapitre 3 de ce mémoire, propose pour sa part de modéliser ce type d'interaction à travers l'utilisation des messages et des lignes d'activités reliées aux différents types d'intervenants sur

un même processus métier. Ainsi, l'exemple de la Figure 1.12, tiré de la documentation du langage BPMN, présente, à travers l'exemple d'une visite d'un patient à un médecin, l'ensemble des messages échangés entre les différents intervenants de ce processus métier ainsi que les tâches réalisées par chacun des intervenants ; aucune de ces tâches n'étant automatisée. Du point de vue de la modélisation, le processus présenté est générique et peut s'appliquer à chaque visite d'un patient à son docteur. Il pourra aussi par la suite servir de base à une automatisation partielle du processus à travers l'automatisation de certaines tâches ; le reste du processus restant inchangé.

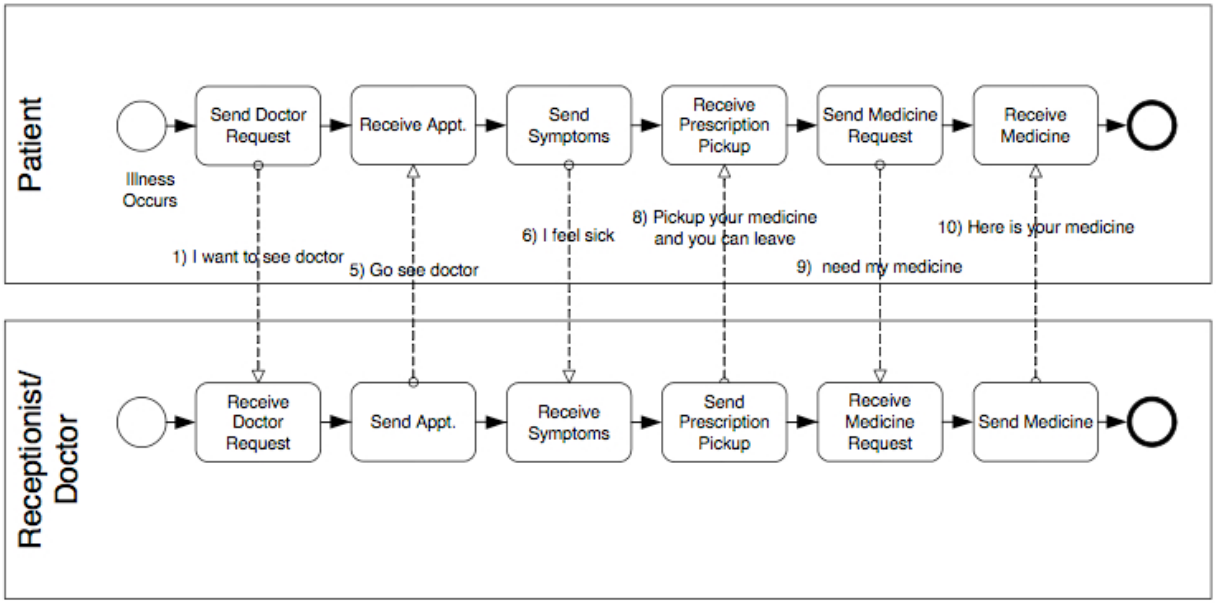


Figure 1.12. Exemple de processus métier mélangeant tâches et échanges de messages verbaux (exemple tiré de la documentation de la notation BPMN)

Si le pouvoir d'expression des processus métier et du langage BPMN au niveau des interactions représente un avantage indéniable par rapport aux autres méthodes et langages de modélisation en matière d'expressivité des modèles, tout n'est pas parfait et il existe un certain nombre de contraintes et de limites qu'il est important de présenter.

1.3.3 Synthèse et discussion

Du point de vue des fournisseurs commerciaux d'environnements de développement informatique, l'approche par les processus métier représente l'approche "idéale" qui permet de faciliter la communication entre les informaticiens et les utilisateurs et surtout qui permet de développer rapidement des applications réutilisables (Silver, 2007). Si ceci est vrai dans l'absolu si on s'en tient aux définitions des processus métier, la réalité est beaucoup plus nuancée.

Ainsi, si aujourd'hui, suite à sa normalisation par l'OMG, le langage BPMN est devenu un standard de fait largement utilisé par des produits commerciaux, son utilisation est très souvent limitée aux architectures techniques de type Service Oriented Architecture ou SOA (Oasis, 2006). Dans ce type d'utilisation, les services web sont vus comme des composants métier ; chaque composant métier permettant d'exécuter un ensemble de tâches métier élémentaire. Ainsi, les composants métier représentent des sous-processus métier capables de dialoguer entres eux ; le langage BPMN étant alors utilisé pour permettre l'assemblage de ces sous-processus métier afin de créer des processus métier complets. Cette approche par composants, si elle accélère les développements informatiques, présente le désavantage, du point de vue des IHM, de ne prendre en compte les interactions homme-machine que de manière très limitée. En fait, seules les interactions permettant de passer d'un composant à un autre sont représentées dans les modèles ; les interactions associées directement à chaque composant étant définies dans le composant lui-même et donc non modélisées par BPMN.

La notation BPMN présente aussi le désavantage de n'avoir été conçue que pour prendre en charge des applications fonctionnant suivant le principe des workflows applicatifs associés à des processus de gestion. Aussi, est-il très difficile de modéliser des applications temps réel en utilisant cette notation avec, par exemple, une prise en compte très limitée des processus fonctionnant en parallèle (difficulté à prendre en charge correctement les notions de processus synchrones et asynchrones).

Enfin, il n'est pas sûr que la notation BPMN ait un pouvoir d'expressivité suffisant pour pouvoir modéliser tous les types de processus métier de type workflow applicatif. Aujourd'hui, ce problème ne peut se résoudre qu'en utilisant une approche par composants métier qui, en permettant de masquer une partie de la complexité de la définition des processus métier dans les composants, permet de contourner les difficultés éventuelles. Aussi, les limites de la notation BPMN sont-elles aujourd'hui mal cernées ; ce qui représente très clairement une limitation dans le cadre d'une généralisation de son utilisation en-dehors des approches de type SOA.

En conclusion, si l'approche par les processus métier présente de nombreux avantages, elle est aujourd'hui trop limitée pour permettre la prise en charge de tous les aspects liés à la modélisation des IHM dans le cadre du développement d'applications dans le domaine de l'informatique de gestion. Néanmoins, comme la notation BPMN est, par nature, extensible, il semble tout à fait envisageable de l'étendre pour ne plus travailler uniquement au niveau de composants métier mais de descendre encore plus dans le détail pour travailler au niveau des éléments de base des interactions. En fait, du point de vue des IHM, cette notation représente une opportunité pour faire le lien entre les tâches directement liées aux interactions et toutes les autres tâches liées aux processus métier et ceci à l'aide d'un formalisme facilement compréhensible par tous. C'est ce que nous montrerons dans le chapitre 3 de ce mémoire où nous présenterons une nouvelle approche des processus métier basée sur le langage BPML. Mais, rendre un modèle compréhensible par tous ne repose pas uniquement sur une notation partagée par tous mais aussi sur un vocabulaire partagé par tous, parfaitement adapté au domaine métier traité. Si, l'utilisation d'un dictionnaire permet de résoudre cette problématique, l'utilisation d'une ontologie apporte pour sa part de nombreux avantages que nous allons maintenant aborder.

1.4 L'ontologie dans le cadre de la modélisation des applications interactives

1.4.1 Définition et principes

Une des grandes problématiques de la modélisation conceptuelle des applications informatiques réside dans la difficulté d'établir des modèles conceptuels pour l'ensemble des éléments de l'application. Si ceci est vrai au niveau de la modélisation des interactions homme-machine ou encore de la modélisation de tout ce qui touche à la sécurité des accès à l'application, c'est aussi vrai pour ce qui est la manipulation des données traitées par l'application. Ainsi, si les données peuvent n'être vues que comme des ensembles d'informations indépendants les uns des autres, dans le cadre d'une application déterminée, elles ont toutes un sens bien précis du point de vue métier. Mais surtout, elles disposent d'un certain nombre de relations entre elles pour permettre leur exploitation dans le cadre d'une application. Pour donner un exemple dans le domaine des transports, un horaire est toujours associé à une ligne de transport, et une ligne de transport est toujours desservie par une ou plusieurs compagnies de transport. Cette problématique, qui n'est pas récente, a donné lieu à la création de méthodes de modélisation des données comme la méthode Merise de (Tardieu *et al.,* 1983) (Nanci et Espinasse, 2001) ou encore la méthode entité-relation de (Chen, 1976) ; mais elle a aussi donné lieu à la création d'une nouvelle approche totalement indépendante des bases de données : les ontologies.

Aujourd'hui, la notion d'ontologie est largement utilisée dans ce qu'on appelle le web sémantique (Berners-Lee et Fischetti, 1999) (Berners-Lee *et al.,* 2001) dont l'objectif est de permettre une caractérisation unique des pages web afin de permettre de donner un sens à leur contenu. Dans ce cadre l'objectif recherché est de ne plus faire des recherches sur le web en fonction des mots contenus dans les pages web mais en fonction des concepts métier qui y sont associés permettant ainsi d'avoir des résultats de recherches plus pertinents en fonction des besoins des utilisateurs.

Par définition, le mot ontologie signifie l'étude des propriétés générales de ce qui existe. Dans le domaine de l'informatique, l'ontologie permet de définir l'ensemble des termes et concepts qui peuvent être manipulés dans le cadre d'une application. Elle permet aussi de définir les relations entre les termes et concepts se transformant ainsi en un modèle de données représentatif du domaine métier sur lequel il est possible ensuite d'effectuer des requêtes et des recherches. De manière générale, une ontologie est définie par les éléments suivants :

- Des classes qui permettent de définir des ensembles ou des collections d'objets
- Des individus qui sont des objets concrets d'une classe
- Des attributs qui permettent de caractériser les classes et, par la même occasion, les individus
- Des relations qui permettent de définir les liens entre les classes et entre les individus à la fois au niveau sémantique mais aussi au niveau hiérarchique entre les classes et les individus
- Des restrictions qui permettent de définir avec précision les types de relations possibles et donc les individus qui peuvent être insérés dans l'ontologie.

Pour une présentation plus détaillée des différents termes utilisés dans le cadre des ontologies et pour un exemple de définition d'une ontologie, on pourra se reporter à (Noy et McGuiness, 2001) ou à la Figure 1.13.

En matière de définition des ontologies, l'élément le plus important réside dans le langage utilisé pour décrire l'ontologie et plus précisément dans son pouvoir de description. Aussi, nous allons présenter dans la partie suivante des outils et méthodes représentatifs permettant de définir et d'utiliser des ontologies.

1.4.2 Étude des outils et méthodes de description et de manipulation des ontologies

Avant toute chose, il est important de noter que les ontologies ont d'abord été développées dans le cadre des travaux sur la représentation de la connaissance (Sowa, 2000) avant d'être utilisées dans le cadre du web sémantique. Aussi, les ontologies sont, aujourd'hui, très souvent à la base de la création de bases de connaissances sur lesquelles l'utilisateur peut effectuer un certain nombre de requêtes.

Si les langages utilisés dans le cadre de la définition des ontologies sont multiples, ils reposent tous sur les principes d'une logique de premier ordre dans laquelle les informations, les connaissances, sont représentées sous la forme de triplets du type : sujet, prédicat et objet. Dans le cadre de cette thèse, on ne fera pas une présentation exhaustive de tous les langages utilisables mais uniquement des principaux langages utilisés dans le cadre des ontologies.

Aujourd'hui, à la base de beaucoup de langages de description d'ontologies, on trouve le Resource Description Framework ou RDF (W3C (a), 2004) qui n'est autre qu'un langage conceptuel permettant de décrire des ressources (des éléments) de manière simple et sans ambiguïté. Ci-dessous, on trouvera un exemple de description, en RDF, du tableau suivant permettant de définir des compagnies de transport :

Nom	**Localisation**	**AnneeCreation**
Semurval	Valenciennes	1979
Transpole	Lille	1982

Le fichier RDF correspondant est donné à la page suivante.

```
<?xml version="1.0"?>

<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:transporteur="http://www.viatic.org/transporteur#">

        <rdf:Description  rdf:about="http://www.viatic.org/transporteur/Semurval">
            <transporteur:localisation>Valenciennes</transporteur:localisation>
            <transporteur:anneecreation>1979</transporteur:anneecreation>
        </rdf:Description>

        <rdf:Description rdf:about="http://www.viatic.org/transporteur/Transpole">
            <transporteur:localisation>Lille</transporteur:localisation>
            <transporteur:anneecreation>1982</transporteur:anneecreation>
        </rdf:Description>
</rdf:RDF>
```

Dans ce fichier, l'espace de nommage défini par "xmlns:rdf" permet d'indiquer qu'on va utiliser une syntaxe de type rdf. L'espace de nommage défini par "xmln:transporteur" permet d'indiquer qu'on va manipuler des individus de type "transporteur" dans la suite du fichier.

Si RDF permet de définir un certain nombre de notions du point de vue des ontologies, il ne permet pas de définir des notions de classes équivalentes, d'identité d'individus, de contraire, de symétrie, de transitivité dans les relations ou encore de cardinalité. Or ces éléments sont essentiels pour pouvoir décrire l'ensemble des concepts et des individus associés à un domaine métier.

Pour répondre à ces manques, le W3C a proposé un nouveau standard de description des ontologies à travers le Web Ontology Language ou OWL (W3C (b), 2004) qui prend ses fondations dans RDF. Celui-ci, en s'inspirant de langages comme le Darpa Agent Markup Language ou DAML (accessible sur le site http://www.daml.org), et plus spécifiquement de sa dernière version appelée DAML+OIL, et des fondements théoriques de la logique de description, propose un langage de description des ontologies beaucoup plus expressif que RDF. OWL présente aussi la particularité d'être à la fois un langage de description des ontologies et un langage d'échange entre les ontologies ; facilitant ainsi les échanges et les liens entre ontologies. Afin de pouvoir permettre une utilisation souple du langage, celui-ci a été défini suivant trois niveaux différents :

- OWL-Lite permet de décrire des classifications hiérarchiques simples (cardinalité de 0 ou 1) dans lesquelles il y a peu de contraintes (exemple : organisation des livres dans une bibliothèque). L'objectif de ce langage est de permettre d'effectuer une migration simple des thesaurus et autres taxonomies existantes vers OWL.
- OWL-DL permet de définir une ontologie avec le maximum d'expressivité en garantissant qu'il soit possible d'effectuer, sur celle-ci, des raisonnements qui donneront toujours un résultat et ceci dans un temps fini.
- OWL-Full permet, à partir d'une sémantique différente de OWL-Lite et OWL-DL, de définir des ontologies beaucoup plus riches pour lesquelles il est impossible de garantir que les raisonnements effectués le seront dans un temps fini et/ou donneront un résultat.

Si une ontologie OWL-Lite est aussi une ontologie valide au niveau OWL-DL et si une ontologie OWL-DL est aussi une ontologie valide au niveau OWL-Full, l'inverse n'est jamais vrai sauf cas particuliers. Aujourd'hui, OWL est devenu le standard de fait de définition des ontologies permettant de développer des ontologies sur une base commune mais surtout facilitant les échanges entre ontologies. Cette standardisation a permis de développer un ensemble d'outils de création et de manipulation d'ontologie dont l'outil Protege, développé à l'université de Stanford, est très probablement le plus avancé (http://protege.stanford.edu).

Pour modéliser les domaines métier, on dispose aujourd'hui d'un certain nombre d'ontologies génériques qui peuvent servir de base au développement d'ontologies plus spécifiques :

- OpenCyc (http://www.opencyc.org) qui est une ontologie mère dont l'objectif est de prendre en compte l'ensemble des domaines métier existants.
- Wordnet (http://www.wordnet.org) qui est une ontologie dont l'objectif est de prendre en compte l'ensemble des termes utilisés dans la langue anglaise.
- The United Nations Standard Products and Service Code (UNSPSC) (http://www.unspsc.org) dont l'objectif est de permettre une classification de tous les produits et services.

Dans le domaine des transports, un certain nombre d'ontologies plus spécifiques ont déjà été développées et/ou étudiées comme :

- The Ontology based Traffic Network ou OTN de (Lorenz *et al.*, 2005) qui propose de lier ontologies géographiques et ontologies transport.
- La proposition de (Timpf, 2002) où il compare, dans le cadre des déplacements urbains, une ontologie centrée sur les transports par rapport à une ontologie centrée sur le voyageur montrant la supériorité de cette dernière dans le cadre d'un déplacement de type voyage.
- La partie réservée aux transports dans OpenCyc (http://www.opencyc.org) qui a pour vocation d'être une ontologie universelle.

A titre d'exemple, on trouvera dans la Figure 1.13 une description de la classe "Route_Section" définie dans OTN telle qu'elle peut être affichée dans un outil comme Protege. Ainsi, une section de route ("Route_Section") est définie comme étant le chemin à parcourir entre deux arrêts sur une ligne de transport comme une ligne de bus par exemple. Ainsi, une section de route commence et se termine obligatoirement à un arrêt ("Stop_Point") et emprunte au moins une route physique ("Route_Link").

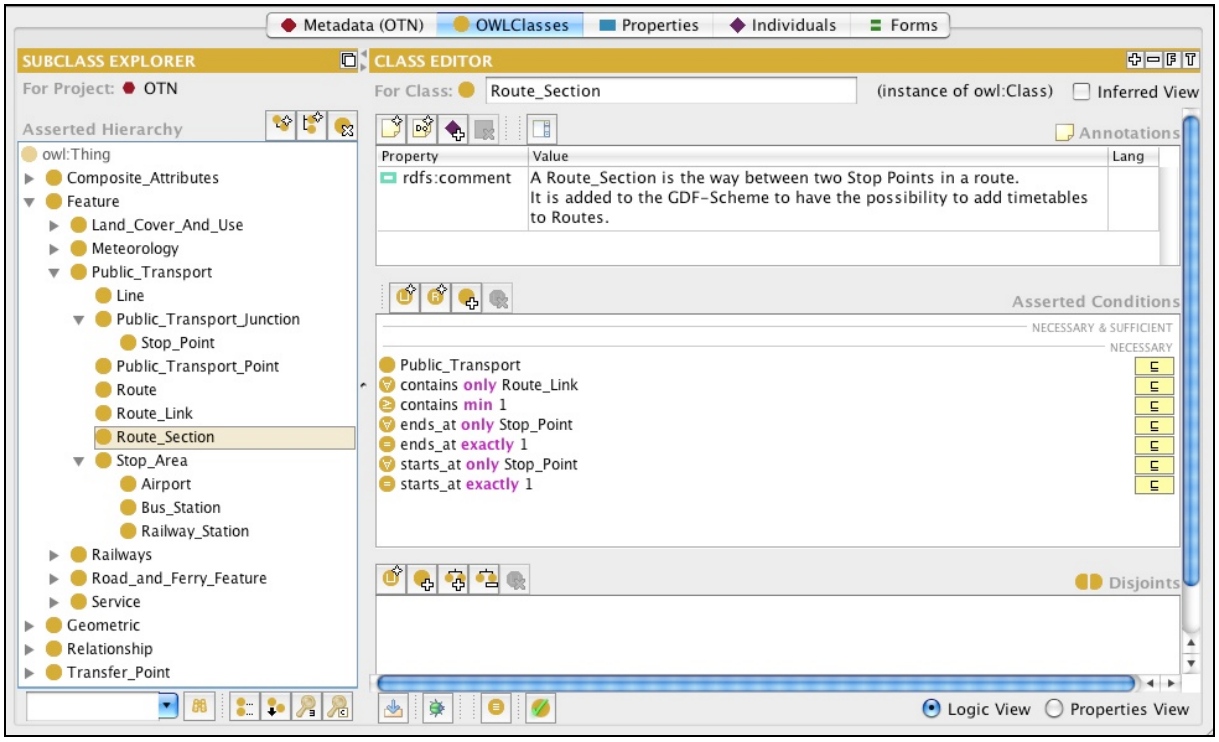


Figure 1.13. Exemple de représentation d'une classe "Route_Section" de l'ontologie OTN dans l'outil Protege

Il est intéressant de noter qu'une ontologie peut aussi être représentée sous la forme d'un graphe de dépendance entre classes comme dans la Figure 1.14 où on présente, dans l'ontologie OTN, les dépendances de la classe "Route_Section" par rapport aux autres classes de l'ontologie.

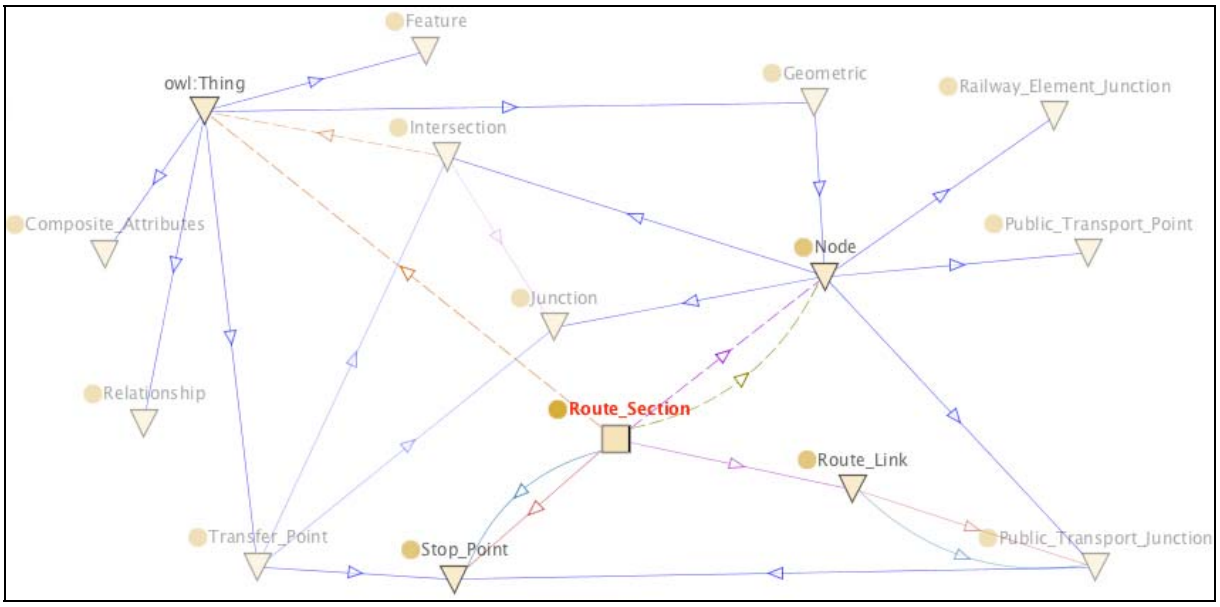


Figure 1.14. Graphe de dépendance de la classe "Route_Section" dans l'ontologie OTN

Malgré le fait qu'on dispose aujourd'hui d'un ensemble d'outils et de connaissances permettant de créer des ontologies, il n'existe pas aujourd'hui de véritable ontologie universelle reconnue et utilisée par tous. Ceci s'explique par les limitations et contraintes qui sont associées à la définition même des ontologies.

1.4.3 Synthèse et discussions

La première grosse problématique des ontologies réside dans le fait que pour un même domaine métier, il n'existe pas une seule ontologie possible (Noy et McGuiness, 2001). En fait, la définition d'une ontologie dans un domaine est très fortement dépendante de plusieurs éléments :

- De la définition des noms des classes, des relations et des propriétés ; ceux-ci devant être suffisamment clairs et précis pour être compréhensibles par tous et surtout ne pas introduire d'ambiguïté au niveau de leur contenu.
- Des connaissances et de la façon de percevoir le domaine des personnes qui créent l'ontologie.
- De l'utilisation qui va être faite de l'ontologie. Ainsi, certains éléments du domaine peuvent être vus comme des classes dans le cadre d'une certaine utilisation et comme des individus dans d'autres utilisations.

La deuxième problématique, qui est fortement liée à la première réside dans la difficulté à définir de manière pertinente les différentes classes, propriétés et relations de l'ontologie de manière à avoir une expressivité correcte par rapport au domaine ciblé. Ainsi, il n'existe aujourd'hui aucune méthode, et par extension aucun outil, permettant de s'assurer qu'une ontologie définit un domaine métier avec un minimum de classes, d'individus, de propriétés et de relations. De la même manière, il est impossible de pouvoir assurer qu'une ontologie est capable d'exprimer complètement un domaine sans en avoir testé toutes les possibilités.

Enfin il n'est pas sûr, comme l'a montré (Shirky, 2005) en prenant l'exemple du web sémantique, que les ontologies soient toujours la bonne solution pour caractériser un domaine ; d'autres techniques de caractérisation pouvant s'avérer beaucoup plus pertinentes et surtout plus efficaces pour permettre de manipuler l'ensemble des informations d'un domaine. Pour donner un exemple, le tableau périodique des éléments représente une classification stable et reconnue par tous qui permet de rapidement identifier chaque élément et surtout de rapidement en voir les caractéristiques. Dans ce

cadre, utiliser une ontologie pour représenter le tableau périodique des éléments introduirait une complexité inutile à travers une nouvelle représentation de ce tableau basée uniquement sur des liens entre les éléments ; ce qui est loin d'être pertinent du point de vue d'un chimiste.

Tout ceci ne doit pas occulter le fait que les ontologies font aujourd'hui partie des rares outils qui permettent d'appréhender un domaine métier de manière conceptuelle. Cette vision globale du domaine permet aussi d'englober l'ensemble des données qui y sont manipulées sous forme de classes et sous forme d'individus. À ce titre, il apparaît comme un des outils important à considérer dans le cadre d'une modélisation conceptuelle complète des applications interactives comme nous le verrons dans le cadre du chapitre 3 de ce mémoire. Et ceci d'autant plus que son utilité a été démontrée au niveau des applications créées dans le domaine des transports collectifs par (Wang *et al.,* 1997) dans le cadre de la recherche d'information transport et par (Becker et Smith, 1997) dans le cadre de la planification et de l'organisation des déplacements dans un contexte multi-modal (plusieurs modes de transport empruntés) qui sont deux aspects importants des besoins d'information transport comme nous allons maintenant le voir.

1.5 La modélisation des applications interactives dans le domaine des transports dans le cadre de l'information voyageur

1.5.1 Les typologies d'applications dans le domaine des transports

Le domaine des transports est un domaine assez vaste qui recouvre de nombreux aspects qui vont de la définition des moyens de transport à la gestion des usagers individuels, en passant par la gestion des infrastructures, les politiques de transport ou encore la supervision des déplacements. Dans le cadre de notre thèse, nous avons choisi de ne pas explorer l'ensemble de ces domaines mais de nous concentrer sur le domaine spécifique de l'information de l'usager dans le cadre de ses déplacements en transports collectifs impliquant l'utilisation de plusieurs moyens de transport (déplacements intermodaux). Et, dans ce domaine, nous avons décidé de nous focaliser plus particulièrement sur la problématique des applications interactives mise à la disposition des usagers pour préparer leurs déplacements et pour les accompagner pendant toute la durée de ceux-ci.

Comme l'a souligné (Lyons, 2006), les besoins en informations des voyageurs dans le cadre de leurs déplacements présentent les particularités suivantes :

- Les besoins ne sont pas constants dans le temps dans le sens où l'usager fait appel aux services d'information voyageur de manière ponctuelle en fonction d'une problématique bien déterminée comme, par exemple, un déplacement non habituel ou encore pour obtenir des informations lorsqu'il y a une perturbation sur le réseau de transport.
- L'usager recherche principalement des informations pour préparer ses voyages et/ou pour choisir le meilleur moyen pour se déplacer.
- Le besoin d'information est dépendant du contexte dans lequel se trouve l'usager (Lyons *et al.,* 2001)

Au niveau des sources d'information, l'information voyageur présente un certain nombre de particularités qu'il convient de prendre en compte. Ainsi, comme l'a souligné (Kenyon et Lyons, 2003), dans le domaine des transports, il existe trois types de systèmes d'informations :

- Des systèmes unimodaux centrés sur un seul moyen de transport (par exemple, uniquement le train).
- Des systèmes multimodaux permettant d'intégrer des informations de plusieurs moyens de transport pour un transporteur bien déterminé (comme, par exemple, l'ensemble des moyens de transports proposés par la société Transpôle sur la région de Lille (métro, bus, tram, navettes)).

- Des systèmes multimodaux intégrés permettant d'avoir des informations sur plusieurs moyens de transport provenant de différents transporteurs et ceci de manière totalement intégrée (comme par exemple, le site http:/transport-idf.com. qui permet, pour la région parisienne, de préparer des itinéraires utilisant les trois réseaux Optile, RATP et Francilien (SNCF))

Dans le cadre de notre thèse, le principe d'une méthode de modélisation générique capable de prendre en compte ces trois types de systèmes d'information a été retenu.

Enfin, parmi les dernières particularités associées au système d'information voyageur, on pourra retenir (Lyons, 2006) que du point de vue technique :

- L'information doit être accessible sur différents type de supports (ordinateur portable, téléphone portable, afficheurs publics, etc.)
- L'information doit pouvoir être consultée au cours des déplacements, quel que soit l'endroit où se trouve l'utilisateur, et que ceci est particulièrement important dans le cas où le réseau de transport est perturbé.
- L'information doit pouvoir être accessible à tous les types d'usagers (personnes jeunes, personnes âgées, personnes avec un handicap, etc.).

Il est important de noter que notre thèse étant orientée sur la génération d'applications interactives, on ne traitera pas des problématiques psychologiques et anthropologiques associées à l'utilisation des systèmes d'information voyageur. Pour ces questions, le lecteur pourra consulter les articles de (Verplanken *et al.,* 1997) et de (Wells et Horan, 1999) et pour les besoins plus spécifiques pour les personnes âgées et les personnes handicapées à (Bekiaris *et al.,* 2007). On ne traitera pas non plus de la problématique de la pertinence économique de la mise en place d'un système d'information voyageur pour laquelle on pourra se référer à (Chan *et al.,* 2006) ou encore à (Mapp *et al.,* 2000).

Après cette présentation des principales spécificités et problématiques liées à l'information voyageur, nous allons pouvoir étudier quelques solutions existantes, spécifiques au domaine de l'information voyageur, en matière d'IHM.

1.5.2 Les travaux existants dans le domaine de l'information voyageur

Aujourd'hui, dans le cadre de nos travaux de recherche, nous n'avons pas trouvé de méthodes de modélisation et/ou de conception d'applications permettant de prendre en charge l'ensemble des problématiques associées à l'information voyageur. Par contre, un certain nombre de travaux ont permis d'apporter des éléments de réponse à chaque problématique. Il est important de noter que dans la suite du document, on ne donnera pas une liste exhaustive de l'ensemble des travaux menés dans le domaine de l'information voyageur mais uniquement quelques applications représentatives des différentes problématiques traitées.

Pour ce qui est de la mise à disposition des informations transports sur un système mobile, on pourra citer les travaux de (Chevest et al., 2000) qui proposent un système d'information touristique sensible au contexte. On pourra aussi citer les travaux de (Maclean et Dailey, 2001) qui propose un système pour informer les usagers des prochains départs et des prochaines arrivées sur leur téléphone mobile. Les travaux de (Kjeldov *et al.,* 2003) proposent pour leur part, à travers TramMate, un système mobile d'aide aux déplacements dans un contexte professionnel où un utilisateur doit effectuer plusieurs rendez-vous dans des lieux différents dans une même journée. Dans TramMate, le contexte (temps, lieu, contenu de l'agenda, etc.) est utilisé afin d'optimiser les déplacements de l'usager et surtout afin qu'il puisse arriver à l'heure à tous ses rendez-vous. La Figure 1.15 présente un exemple d'IHM sur une plateforme mobile de type PDA de l'application TramMate.

Figure 1.15. Exemple d'IHM sur une plateforme de type PDA dans le cadre du projet TramMate (Kjeldov *et al.*, 2003)

Les travaux de (Jacquet *et al.*, 2006) qui, dans le cadre d'un outil PRIAM (PRésentation des Informations dans l'AMbiant), propose de nouveau types d'interactions entre les systèmes mobiles et les panneaux d'affichage public en vue de mieux guider les usagers en déplacement dans un aéroport. La Figure 1.16 présente un exemple de d'utilisation de PRIAM dans un aéroport où les écrans d'affichage généraux sont personnalisés en fonction de l'utilisateur situé à proximité. Ce type de personnalisation permet à l'usager de plus rapidement s'orienter dans un aéroport et surtout facilite l'accès à l'information ; celle-ci n'étant pas affichée suite à une demande explicite de l'usager mais parce que le système sait que l'usager en a probablement besoin.

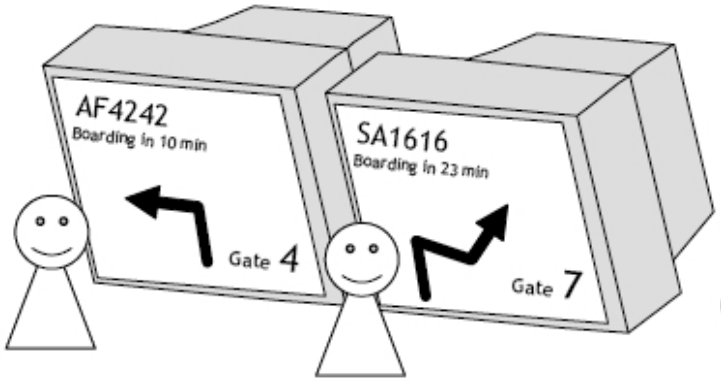


Figure 1.16. Exemple d'utilisation de PRIAM sur des panneaux d'affichage public dans un aéroport (Jacquet *et al.*, 2006)

Pour ce qui est de l'accès des utilisateurs aux systèmes d'information voyageur, on pourra citer les travaux de (Banâtre *et al.*, 2004) qui proposent de nouveaux moyens d'interaction pour permettre à des personnes non-voyantes de pouvoir identifier plus facilement les bus qu'elles doivent emprunter durant leurs déplacements. On pourra aussi citer les travaux de (Sperandio *et al.*, 2002) qui, à partir d'études expérimentales sur des sujets non-voyants, ont défini un certain nombre de règles à respecter dans l'élaboration du contenu d'un site web dans le domaine des transports. Toujours pour les personnes non-voyantes, on pourra citer les travaux de (Sanchez *et al.*, 2007) qui proposent, à partir d'une expérience sur le terrain, un certain nombre de règles pour la définition d'un système mobile de guidage des personnes non-voyantes pendant leurs déplacements. Ou encore les travaux de (Fink *et al.*, 1998), qui proposent, à travers une architecture particulière appelée AVANTI ("en avant" en italien), un système d'information touristique, prenant en compte l'information transport, capable de s'adapter à tout type d'utilisateurs et notamment aux personnes âgées et aux personnes avec un

handicap. La Figure 1.17 présente l'architecture AVANTI et son utilisation dans le cadre de la prise en compte des handicaps ; celle-ci étant basée sur l'utilisation d'un modèle d'utilisateur et de règles d'adaptation qui vont servir à générer et adapter les interfaces utilisateur.

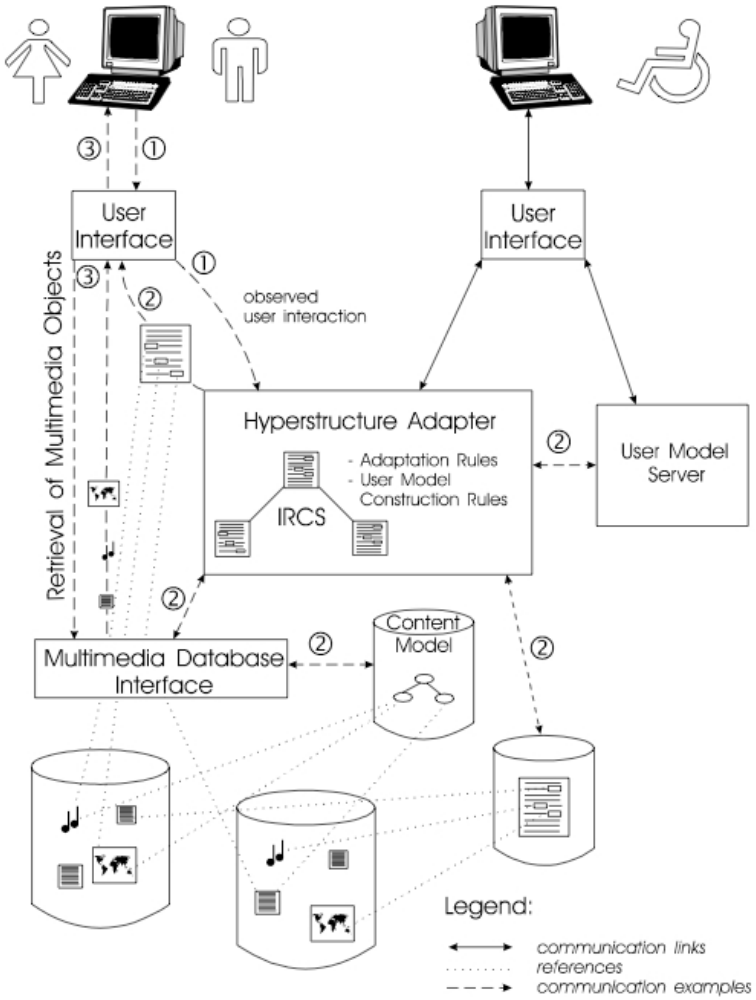


Figure 1.17. Présentation globale de l'architecture AVANTI (Fink *et al.*, 1998)

Enfin, on pourra terminer par les travaux de (Matsubara *et al.*, 2001) qui proposent de nouveaux moyens d'interaction pour guider les usagers pendant leurs déplacements. L'approche choisie dans le cadre de ces travaux est d'utiliser l'environnement physique, à travers différents types de capteurs, comme source d'information pour adapter les applications à l'utilisateur ; la relation entre l'utilisateur et les capteurs étant identifiée de manière unique (par exemple, pour une personne avec des problèmes de vue, avec l'utilisation d'une canne disposant d'un système de type radio-émetteur). Une présentation simplifiée du système proposé est présentée dans la Figure 1.18.

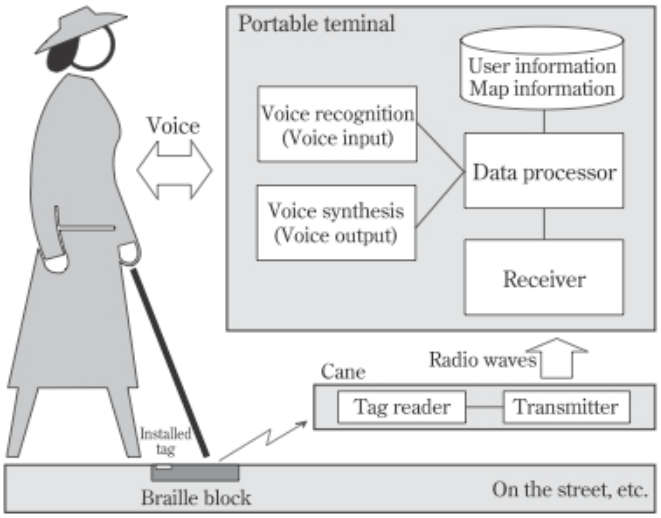


Figure 1.18 Vue globale du système proposé par (Matsubara *et al.*, 2001)

Si la partie personnalisation des informations est un élément important dans le domaine de l'information voyageur, celle-ci ne sera pas traitée dans le cadre de ce chapitre mais dans le cadre du chapitre suivant traitant spécifiquement de tous les aspects liés à la personnalisation des informations et des applications.

1.5.3 Synthèse et discussion

L'ensemble des travaux que nous venons de présenter, bien que ne concernant généralement qu'une petite partie des problématiques associées à la création d'applications interactives dans le domaine des transports, nous permettent néanmoins de déterminer un certain nombre de critères que devraient respecter un outil de modélisation des applications de type information voyageur :

- Les modèles conceptuels doivent être suffisamment génériques pour permettre de créer des applications pour de nombreux types de plateformes différentes (PDA, ordinateurs portables, smartphone, téléphone portable, afficheurs publics, etc.).
- Les modèles conceptuels doivent permettre de prendre en compte le contexte d'usage ; c'est-à-dire le contexte dans lequel l'application est utilisée de manière à être capable de s'adapter, si nécessaire, à tout changement de contexte.
- Les modèles conceptuels d'interactions doivent pouvoir être suffisamment souples pour permettre une certaine *plasticité* des interfaces, au sens de (Thévenin et Coutaz, 1999) et (Calvary et Coutaz, 2002) en permettant aux interfaces d'utiliser plusieurs plateformes techniques en simultané.
- Les modèles conceptuels doivent permettre de prendre en compte tous les types d'interaction (voix, toucher, capteur, etc.).
- Les modèles conceptuels doivent prendre en compte la notion de personnalisation afin de permettre de créer des applications capables d'offrir des contenus et des interfaces adaptés à chaque utilisateur.
- Les applications créées à partir des modèles conceptuels doivent être capables de s'adapter automatiquement à l'utilisateur et à ses handicaps.
- Les modèles doivent être capables de gérer les événements externes afin de pouvoir s'adapter en temps réel aux besoins des utilisateurs. Ainsi, dans le cadre d'une application de transport, un problème survenant sur une ligne de transport devrait être immédiatement pris en compte par les applications soit pour avertir les usagers, soit pour leur proposer des solutions de contournement pour leur permettre de continuer leur voyage.

Dans le cadre de ce mémoire, la solution présentée dans les chapitres 3 et 4 n'a pas pour ambition d'apporter une réponse complète à l'ensemble de ces critères mais plutôt des premiers éléments de réponse pour la majorité des critères. En fait, seuls deux critères, que nous avons jugés comme fondamentaux, seront traités en profondeur : par rapport à, d'une part la problématique de la modélisation générique des applications, et d'autre part la personnalisation des informations fournies à l'utilisateur.

Aujourd'hui, la problématique principale de la création d'une application dans le domaine de l'information voyageur réside dans le fait qu'il n'existe pas de méthode et/ou d'outil permettant de modéliser l'ensemble de l'application. Il est certes possible d'utiliser des outils existants comme, par exemple, la notation UML ; mais celle-ci, du fait de sa forte dépendance avec les technologies objets et son niveau d'abstraction limité par rapport aux problématiques métier, ne permet pas la création de modèles conceptuels véritablement indépendants des technologies et donc ne permet pas une approche de type IDM. Aussi est-il indispensable de réfléchir à une nouvelle méthode de modélisation conceptuelle des applications dans le domaine des transports capables de prendre en charge l'ensemble des critères présentés précédemment. Ces critères étant nombreux et variés, ils vont imposer à la méthode de modélisation d'avoir des modèles suffisamment génériques pour pouvoir prendre en compte un ou plusieurs critères simultanément. L'ensemble des critères n'étant pas trop éloignés de ceux qu'on peut rencontrer dans d'autres domaines métier, il sera par la suite tout à fait envisageable d'étendre la méthode de modélisation afin qu'elle puisse être utilisée dans d'autres domaines applicatifs.

Conclusion

À travers ce chapitre, nous avons montré que l'approche dirigée par les modèles était une approche très intéressante dans le cadre de l'industrialisation des développements informatiques dans le sens où elle permettait d'accélérer les développements, de les fiabiliser et surtout de faciliter leur réutilisation. Néanmoins, nous avons vu aussi que, pour l'instant, elle était rarement utilisée au maximum de ses possibilités dans le sens où la modélisation conceptuelle des applications était bien souvent oubliée ou prise en compte de manière limitée dans l'ensemble des outils existants.

Dans le domaine de la modélisation des applications interactives, il existe des solutions permettant de faire cette abstraction en vue d'une génération semi-automatique des interfaces sur des plateformes multiples. Mais ces solutions, en se centrant uniquement sur les problématiques directement liées aux interactions homme-machine ne permettent pas de prendre en compte les applications dans toute leur complexité. Ainsi, tous les aspects liés à la sécurité des accès aux applications où aux fonctionnalités purement métier sans interaction avec l'utilisateur sont souvent très difficiles voire impossibles à modéliser avec les différentes solutions proposées.

Une des solutions pour résoudre cette problématique serait de trouver des modèles qui soient pertinents aussi bien du point de vue des interactions homme-machine que du point de vue des autres fonctionnalités métier. Un des candidats les plus intéressants pour cela semble être l'approche par les processus métier qui, à travers un découpage en tâches, permet de modéliser l'ensemble des actions se déroulant au niveau d'une application. Mais, cette approche, qui est majoritairement utilisée dans le cadre d'architectures orientées services web, n'est aujourd'hui pas prévue pour modéliser directement des interfaces interactives bien que ses possibilités d'évolution permettent de l'envisager.

Une autre problématique de la modélisation, du point de vue conceptuel, réside dans la difficulté de modéliser les différents éléments manipulés dans le cadre du domaine métier de chaque application. S'il existe des outils de modélisation des données, ceux-ci, en se concentrant uniquement sur les données, ne permettent pas de prendre en compte tous les concepts du domaine. Aussi, les ontologies apparaissent comme une solution possible à ce problème même si, pour l'instant, elles ont été rarement utilisées dans ce sens.

Dans le domaine des transports, le fait que les utilisateurs soient mobiles et puissent utiliser de nombreuses plateformes techniques différentes pour accéder à l'information entraîne un certain

nombre de contraintes au niveau du développement d'applications. Ainsi, les applications devraient, dans l'idéal, être capables de s'adapter automatiquement à leur contexte d'usage et surtout être capables de fonctionner sur des architectures différentes. Or, aujourd'hui, les solutions existantes sont généralement centrées sur la prise en compte de quelques situations particulières et ne permettent pas de gérer toute cette diversité.

Dans le cadre de nos travaux, et en nous basant sur ces différentes constatations, nous avons développé une méthode de modélisation des applications interactives, basée sur les processus métier et les ontologies, permettant de modéliser complètement une application interactive au niveau conceptuel. Cette méthode, qui se base sur une approche dirigée par les modèles et qui sera présentée dans le chapitre 3 de ce mémoire, propose en outre une génération semi-automatique d'applications multi-plateformes permettant de mieux répondre aux problématiques particulières liées au domaine des transports collectifs.

À travers l'ensemble de ce chapitre, nous avons souvent évoqué une notion importante dans le cadre de l'information transport qui est celle de la personnalisation. Si celle-ci peut paraître simple à comprendre d'un point de vue sémantique (personnaliser c'est adapter à la personne), elle recouvre en fait de nombreuses problématiques que nous allons présenter dans le prochain chapitre.

Chapitre 2
De la personnalisation dans les systèmes d'information à celle dans les systèmes d'information voyageurs dans le domaine des transports

Sommaire

INTRODUCTION **41**

2.1 LA PERSONNALISATION : DEFINITION, PRINCIPES ET METHODES **41**
2.1.1 DEFINITION 41
2.1.2 LES FACTEURS QUI INFLUENT SUR LA PERSONNALISATION 43
2.1.3 METHODES ET TECHNIQUES DE PERSONNALISATION DES CONTENUS 44
2.1.4 SYNTHESE ET DISCUSSION 46

2.2 LA MODELISATION DE L'UTILISATEUR DANS LE CADRE DE LA PERSONNALISATION **46**
2.2.1 DEFINITION 46
2.2.2 ÉTUDE DES OUTILS ET METHODES DE MODELISATION DE L'UTILISATEUR 47
2.2.3 SYNTHESE ET DISCUSSION 49

2.3 LA MODELISATION DU CONTEXTE DANS LE CADRE DE LA PERSONNALISATION **51**
2.3.1 DEFINITION 51
2.3.2 ÉTUDE DES OUTILS ET METHODES DE MODELISATION DU CONTEXTE 52
2.3.3 SYNTHESE ET DISCUSSION 53

2.4 LA MODELISATION DES CONTENUS DANS LE CADRE DE LA PERSONNALISATION **54**
2.4.1 DEFINITION 54
2.4.2 ÉTUDE DES OUTILS ET METHODES DE MODELISATION DES CONTENUS 55
2.4.3 SYNTHESE ET DISCUSSION 56

2.5 LA PERSONNALISATION DANS LE DOMAINE DES TRANSPORTS **57**
2.5.1 LES BESOINS DANS LE DOMAINE DES TRANSPORTS 57
2.5.2 LES OUTILS ET METHODES EXISTANTS ADAPTES AU DOMAINE DES TRANSPORTS 58
2.5.3 SYNTHESE ET DISCUSSION 59

CONCLUSION **60**

Introduction

Depuis quelques années, la généralisation de l'utilisation des outils informatiques et la multiplication des sources d'informations disponibles ont entraîné de nouveaux besoins chez les utilisateurs d'applications interactives. Ainsi, aujourd'hui, le problème pour l'utilisateur n'est plus d'accéder à l'information mais d'accéder à la bonne information au bon moment et sur le bon support. Et, c'est particulièrement important dans le domaine des transports où les usagers ont tous des besoins et des buts différents et où les problématiques sont nombreuses et variées (Lyons *et al.*, 2007). Ceci est d'autant plus vrai que la maîtrise de l'information est un élément capital pour le développement des transports collectifs et plus encore lorsque les déplacements sont multimodaux et donc utilisent plusieurs moyens de transport différents (ATEC-ITS, 2002) (Perreau, 2002) (Uster, 2000) (Uster, 2001). Ainsi, par exemple, dans une gare, certains utilisateurs vont être à la recherche de leur correspondance pour continuer leur voyage, d'autres vont essayer de planifier leur prochain déplacement, d'autres vont vouloir se renseigner sur les prochains trains au départ ou à l'arrivée, d'autres encore voudront savoir où se trouve le commerce le plus proche afin d'effectuer des achats avant de repartir à leur domicile. Et ce ne sont que quelques exemples de toutes les attentes et de tous les besoins utilisateurs dans le domaine des transports et uniquement au niveau d'une gare.

Dans ce contexte, la personnalisation des contenus et des applications est devenue aujourd'hui un élément déterminant pour permettre de résoudre, en partie, cette problématique surtout si la diffusion des applications se fait vers un très large public (ChoiceStream, 2007) (Vesanen, 2005). Si dans sa définition générale, personnaliser, c'est individualiser la relation entre l'utilisateur et l'application, derrière ce terme se trouvent en fait de nombreuses problématiques techniques et méthodologiques (Van Setten, 2001). Et, si on dispose aujourd'hui d'outils et de méthodes pour prendre en charge la personnalisation, ils sont bien souvent spécifiques à certains contextes ou à certains environnements techniques bien déterminés limitant d'autant leur réutilisation et/ou leur généralisation.

Dans ce chapitre, nous allons commencer par définir la notion de personnalisation, telle qu'elle sera utilisée dans cette thèse, en fonction des différentes définitions qu'il est possible de trouver dans la littérature. Ensuite, nous montrerons comment il est possible de modéliser l'utilisateur qui est l'élément central de la personnalisation. Puis, nous ferons un rapide état de l'art sur les travaux portant sur les différents éléments contextuels devant être pris en compte dans le cadre de la mise en place d'un outil de personnalisation et nous montrerons comment il est possible de les intégrer dans une approche de type IDM. Ensuite, nous verrons la problématique d'accès aux contenus dans le cadre de la personnalisation. Enfin, nous terminerons en présentant quelques solutions existantes de personnalisation utilisée dans le cadre d'applications interactives dans le domaine des tranports.

2.1 La personnalisation : définition, principes et méthodes

2.1.1Définition

Actuellement, il n'existe pas de définition standard, dans le domaine de la recherche, sur la notion de personnalisation et chaque auteur l'adapte à ses besoins et à ceux de son domaine. Ainsi, pour (Kimball et Mertz, 2000) la personnalisation est une manière de développer une confiance, une familiarité et un investissement personnel avec l'application. Pour (Bazsalicza et Naïm, 2001), qui se limitent aux sites Web, la personnalisation est la capacité d'un site dynamique à produire des ressources en fonction de l'identité du demandeur. Pour notre part, en nous plaçant dans le cadre d'une modélisation conceptuelle des applications, c'est-à-dire d'une approche non technique des applications (OMG, 2003), nous avons retenu deux définitions complémentaires :

> *"La personnalisation est la capacité de fournir des contenus et des services adaptés aux individus en se basant sur la connaissance de leurs préférences et de leurs comportements "* (Hagen *et al.*, 1999).

> *"La personnalisation est la capacité d'adapter la communication client en se basant sur la connaissance des préférences et des comportements au moment de l'interaction (avec le client)"* (Dyche, 2002).

En partant de ces deux définitions, on peut dire que, pour une application informatique, la personnalisation, c'est la capacité de fournir à un utilisateur, à chaque instant, des contenus et des services adaptés à ses besoins et à ses attentes en utilisant des interactions homme-machine appropriées. Pour ce faire, cette adaptation utilise sa connaissance de l'utilisateur (caractéristiques et préférences), sa connaissance de l'environnement, ainsi que des informations sur le comportement de l'utilisateur au moment de l'interaction. Il en découle que la personnalisation peut se faire à trois niveaux :

- Au niveau de l'interface utilisateur ; celle-ci s'adaptant en fonction des besoins de l'utilisateur et du contexte dans lequel est utilisée l'application. Par exemple, une interface pour une personne ayant des problèmes de vue sera différente d'une interface pour une personne ayant toutes ses facultés visuelles. Du point de vue de l'utilisateur, cette adaptation peut se faire :
  - Soit de manière implicite par le système, c'est-à-dire sans son intervention, et, dans ce cas, on parle d'adaptativité (Oppermann et Simm, 1994) (Stephanidis *et al.*, 2001)
  - Soit de manière explicite suivant des choix effectués par l'utilisateur et, dans ce cas, on parle d'adaptabilité (Browne *et al.*, 1990), (Stephanidis *et al.*, 2001).
- Au niveau des processus métier associés aux applications ; ceux-ci pouvant être différents en fonction de l'utilisateur. Pour donner un exemple, l'achat d'un titre de transport à un distributeur automatique se fera différemment pour un utilisateur identifié dont on connaît déjà les besoins (zones de saisie pré-remplies et processus simplifiés) que pour un utilisateur occasionnel pour lequel on ne possède aucune information. Du point de vue de l'utilisateur, l'adaptation des processus peut se faire soit de manière implicite soit de manière explicite. Pour un exemple de ce type d'adaptation, on pourra se reporter aux travaux de (Balke et Wagner, 2003) qui proposent, pour les environnements de type internet, un algorithme personnalisé de sélection et d'aide à la sélection de services web
- Au niveau des contenus tels que ceux-ci ont été définis dans le cadre du projet ANR Viatic.Mobilité qui sera présenté en détail dans le chapitre 5. La personnalisation se fait soit au niveau des informations retournées soit au niveau du nombre et de la nature des services applicatifs disponibles. Pour un exemple de personnalisation implicite des contenus, on pourra se référer à (Baraglia et Silvestri, 2007) qui proposent à travers un outil de recommandations en ligne, appelé SUGGEST, une personnalisation automatique des contenus proposés à l'utilisateur. Pour un exemple explicite de personnalisation de contenu, on peut se référer à l'outil de recherche en ligne de la société Google (http://www.google.com) qui, à travers un formulaire de recherche où l'utilisateur entre manuellement des mot-clés, permet de retourner à l'utilisateur la liste des pages web qui sont les plus pertinentes et les plus consultées par rapport aux critères de recherche indiqués.

Dans le cadre de nos travaux, qui portent sur la création d'une méthode de modélisation des applications interactives, nous avons considéré que la personnalisation des interfaces utilisateur devait être prise en compte dans le cadre de la modélisation conceptuelle. De même, nous avons considéré que la personnalisation des processus métier faisait partie intégrante des modèles de processus métier. Aussi, nous ne traiterons dans la suite de ce chapitre que de la personnalisation des contenus. Des exemples de personnalisation des interfaces homme-machine et de processus métier prenant en compte la personnalisation seront donnés par la suite lors de la présentation de notre méthode de modélisation (cf. chapitre 4).

2.1.2 Les facteurs qui influent sur la personnalisation

Si la personnalisation peut se faire à plusieurs niveaux dans une application interactive, celle-ci ne peut se faire sans un certain nombre d'éléments que nous appellerons facteurs externes. Ainsi, pour (Van Setten, 2001), les quatre grands facteurs qui ont une influence sur la personnalisation des contenus sont :

- Les facteurs "utilisateur" : tous les facteurs qui concernent spécifiquement les utilisateurs comme, par exemple, leurs centres d'intérêt, leurs préférences, leurs caractéristiques physiques, leurs buts, leurs groupes sociaux d'appartenance, etc.
- Les facteurs de "contenu" ou "d'information" : tous les facteurs qui concernent les caractéristiques et propriétés des contenus permettant de définir comment l'information peut être personnalisée. Ainsi, pour donner un exemple, pour un document ayant été créé à l'aide d'un traitement de texte, les facteurs de contenu pourraient être : le nom de l'auteur, la langue utilisée, le titre, les mot-clés associés au document, etc.
- Les facteurs de "contexte" : tous les facteurs qui concernent l'environnement technique du système d'information ainsi que l'environnement de l'utilisateur et qui peuvent limiter ou augmenter les possibilités en matière de personnalisation. Ainsi, la plateforme technique utilisée par l'usager pour consulter l'information peut limiter le nombre d'informations qu'on peut lui retourner dans le cadre d'une personnalisation soit du fait de capacités techniques limitées (capacité mémoire interne de la plateforme, taille de l'écran, etc.) soit du fait de capacités limitées en matière de communication avec les serveurs de données (système embarqué dans un véhicule, communication en milieu perturbé, etc.).
- Les facteurs liés à la méthode de personnalisation : le choix de la méthode et des outils utilisés pour réaliser la personnalisation influence les possibilités et les résultats qu'il est possible d'obtenir à partir de cette personnalisation. Ainsi, si une méthode de personnalisation ne permet pas de prendre en compte les préférences de l'utilisateur, elle retournera probablement des résultats moins pertinents qu'une autre méthode qui serait capable de les prendre en compte.

Tout cela est résumé dans la Figure 2.1 proposée par (Van Setten, 2001). Dans la suite de ce chapitre nous présenterons les principales solutions existantes pour prendre en compte ces différents facteurs. Et, dans le chapitre 4, nous montrerons comment ces facteurs ont été intégrés dans notre approche.

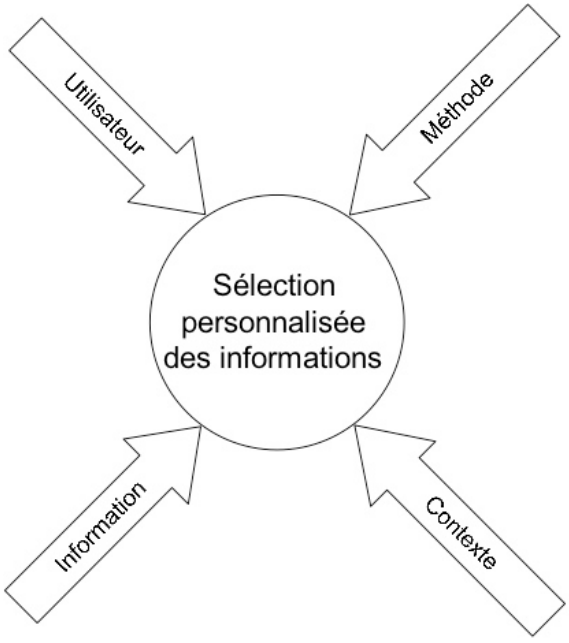


Figure 2.1. Les facteurs externes influençant la personnalisation d'après (Van Setten, 2001)

Il est intéressant de noter ici qu'il est possible d'établir un certain parallélisme entre les critères définis par (Van Setten, 2001) pour la personnalisation des contenus et les critères définis par (Gross et Specht, 2001) ou encore par (Schilit *et al.*, 1994) dans le cadre de l'adaptation des applications interactives à leur contexte d'usage tel que celui-ci a été défini par (Dey et Abowd, 2000) et (Dey, 2001). Ainsi, pour (Gross et Specht, 2001), le contexte est défini par quatre dimensions : la localisation, l'identité de l'utilisateur (à travers la connaissance de ses centres d'intérêts, de ses préférences, de ses connaissances, de l'historique de ses interactions avec le système), le temps et l'environnement ou les activités. Pour (Schilit *et al.*, 1994) un contexte est défini à partir des réponses aux trois questions suivantes : "Qui êtes-vous ? Avec qui êtes-vous ? Et quelles sont les ressources proches de vous ?". De cette manière, la personnalisation n'apparaît plus comme un élément spécifique à prendre en compte dans le cadre de la création d'une application interactive mais comme une adaptation contextuelle de l'application limitée aux contenus.

2.1.3 Méthodes et techniques de personnalisation des contenus

Les outils et méthodes de personnalisation des contenus étant nombreux, nous ne ferons, dans cette partie, qu'une description générale des principales méthodes et outils existants. Dans le cadre de la personnalisation des contenus, il est possible de regrouper ces différentes méthodes et techniques suivants cinq grandes familles (Van Setten, 2001) :

- La sélection par catégorie. C'est la technique la plus simple de personnalisation. De manière pratique, toutes les informations sont regroupées en catégories. L'utilisateur indique ensuite dans son profil les catégories qui l'intéressent et le système crée des interfaces homme-machine dont le contenu est adapté en fonction des catégories choisies. La Figure 2.2 montre ce type de personnalisation utilisée dans le site web "Mon Yahoo!" (http://cm.fr.my.yahoo.com) pour lequel (Manber *et al.*, 2000) ont analysé la facilité d'utilisation et l'efficacité du point de vue des utilisateurs comme, par exemple, la possibilité de gérer soi-même la personnalisation ou encore la possibilité de créer plusieurs types de pages différentes.

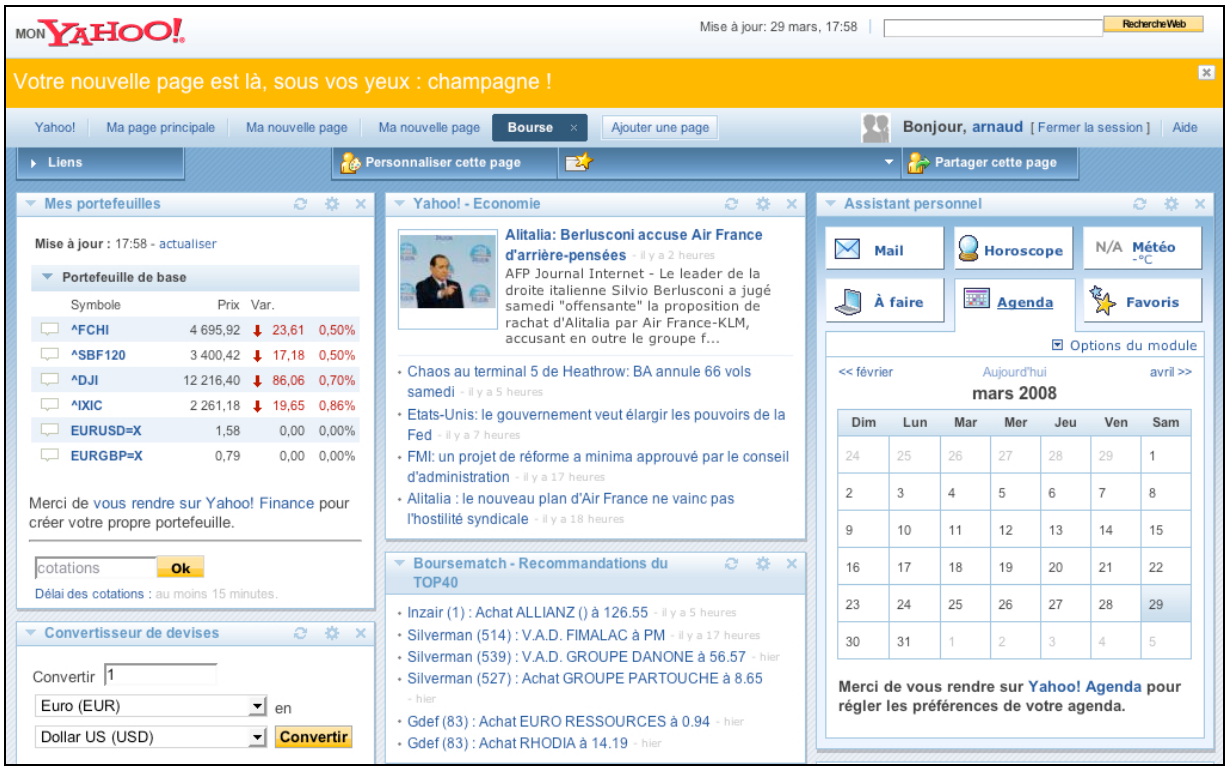


Figure 2.2. Exemple de personnalisation par catégorie sur le site "Mon Yahoo!"

- L'adaptation des requêtes. Dans cette technique, la requête de recherche d'information, qui est créée de manière explicite ou implicite par l'utilisateur, est adaptée en fonction de ce que le système connaît de l'environnement et des besoins des utilisateurs de manière à retourner les informations les plus adaptées au besoin supposé de l'utilisateur. Un exemple de ce type d'adaptation est présenté par (Khan, 1999) qui propose d'adapter les requêtes en fonction d'une ontologie de domaine afin de lever toute ambiguïté au niveau des termes utilisés dans la requête. De manière pratique, dans ce type d'approche, chaque information est associée à un ou plusieurs termes de l'ontologie de domaine. Lors d'une recherche d'information, chaque requête est analysée syntaxiquement de manière à l'associer à différents termes de l'ontologie. Une fois cette analyse effectuée, le système effectue une recherche à partir de la liste des termes de l'ontologie pour afficher les résultats les plus pertinents à l'utilisateur.
- Le filtrage d'information. Cette approche est basée sur les travaux de (Houseman et Kaskela, 1970) et est souvent utilisée lorsque le nombre d'informations sur lesquelles on peut effectuer une recherche est important. Elle consiste à retourner ou à proposer à l'utilisateur un certain nombre d'informations en fonction de ce que le système connaît des besoins et des préférences de l'utilisateur. Des exemples, de ce type de systèmes peuvent être trouvés dans (Van Setten et Moelaert-El Hahidy, 2000). La différence avec le principe d'adaptation des requêtes réside dans le fait que le filtrage est effectué sur le résultat retourné par la requête et pas au moment ou la requête est effectuée.
- Le filtrage social ou filtrage collaboratif. Dans cette approche, proposée par (Goldberg *et al.*, 1992), le système commence par chercher des personnes qui ont un profil et des goûts similaires à l'utilisateur. Puis, il va chercher l'ensemble des documents consultés par ces personnes pour faire des recommandations à l'utilisateur. C'est cette technique qui est utilisée, par exemple, sur le site de vente en ligne de la société Amazon (http://www.amazon.fr) dont un exemple est présenté à la Figure 2.3. Dans cet exemple, l'utilisateur a d'abord effectué une recherche sur un livre d'histoire sur la première guerre mondiale puis il a affiché la fiche d'information associée à ce livre. Le moteur de personnalisation d'Amazon, lui propose alors une liste d'autres livres susceptibles de l'intéresser car proches du livre dont la fiche d'information est affichée et surtout recommandés par des utilisateurs ayant un profil d'achat proche de celui de l'utilisateur. Pour une présentation un peu plus détaillée des différentes techniques de filtrage social, on pourra se reporter au document de synthèse de (Lumineau, 2003).

Figure 2.3. Exemple de filtrage social sur le site de vente en ligne de la société Amazon

- Le filtrage collaboratif informationnel. Cette technique combine les techniques de filtrage collaboratif (recherche des utilisateurs similaires) et les techniques de filtrage sur les contenus informationnels ; ce qui permet, d'après (Hirsh *et al.*, 2000) de combiner les avantages des deux méthodes. Pour des exemples de travaux sur ce type de technique, on pourra consulter les travaux de (Abhinandan *et al.*, 2007), de (Balabanovic et Shoham, 1997), de (Basu *et al.*, 1998) ou encore de (Sarwar *et al.*, 1998).

2.1.4 Synthèse et discussion

À travers cette rapide présentation de la notion de personnalisation et des outils qui lui sont associés, il apparaît que c'est un domaine de recherche assez vaste pour lequel des solutions techniques seules ne suffisent pas. En effet, la personnalisation doit avoir un but, elle doit servir l'utilisateur et/ou le fournisseur de l'application et surtout elle doit apporter quelque chose à l'application, au fournisseur de l'application et à l'utilisateur (Mamber, 2000) (Vesanen, 2005). Les travaux de recherche dans le domaine de la personnalisation doivent donc être pluridisciplinaires et ne pas se limiter uniquement à des solutions techniques si on veut arriver à mettre en place des personnalisations qui répondent à tous ces besoins parfois contradictoires.

Or, aujourd'hui, si on dispose d'outils et de moyens techniques pour mettre en place ces cinq catégories de personnalisation, on sait aussi que chacun présente des limites et des contraintes comme, par exemple, les "attaques malveillantes" possibles sur les systèmes de recommandations afin qu'ils fassent ressortir certaines informations plutôt que d'autres (Burke *et al.*, 2005). Dans ce type "d'attaque", l'objectif est de créer un grand nombre d'utilisateurs virtuels qui vont recommander certaines informations au détriment d'autres informations de manière à favoriser leur sélection par les algorithmes de recommandations et surtout de manière à la faire apparaître, le plus possible, en première position dans les résultats des recherches des autres utilisateurs. Aussi, les derniers travaux dans le domaine s'orientent-ils vers une intégration des différentes techniques ainsi que vers une conceptualisation de la personnalisation en vue d'offrir des systèmes de personnalisation fiables et efficaces (Anand et Mobasher, 2007).

Dans le cadre de nos travaux de recherche, nous avons pris comme hypothèse que, dans les modèles conceptuels des applications, la personnalisation des contenus devait être vue comme un ensemble homogène. Ceci provient du fait que les modèles conceptuels doivent être indépendants de toute notion de technique et que donc il est impossible d'y indiquer quelle technique de personnalisation va être utilisée au niveau de l'application. En fait, tout ce qu'il est possible de préciser au niveau des modèles conceptuels, c'est qu'on désire utiliser la personnalisation en précisant éventuellement les critères qui seront à prendre en compte (préférences utilisateurs, contexte, etc.). Cette séparation entre les modèles conceptuels et les techniques de personnalisation sera présentée en détail dans le chapitre 3.

2.2 La modélisation de l'utilisateur dans le cadre de la personnalisation

2.2.1 Définition

La première chose à connaître pour pouvoir personnaliser les contenus, c'est l'utilisateur. En effet, si on ne dispose d'aucune information sur lui, les contenus qui lui sont fournis ne peuvent être que génériques dans le sens où ils seront les mêmes pour un grand nombre d'utilisateurs. Dans le cadre de la mise en place d'une méthode de modélisation des applications, cette connaissance de l'utilisateur passe par l'utilisation d'un modèle utilisateur, tel qu'il a été défini par (Norman, 1983), à travers l'utilisation de modèles cognitifs, de modèles mentaux, de modèles de tâche ou encore de profils utilisateurs. Dans tous, les cas, comme l'a souligné (Allen, 1997), le modèle utilisateur n'est que l'image que le système informatique a de l'utilisateur et pas l'utilisateur lui-même.

En se plaçant dans le cadre de la personnalisation des contenus et d'après (Van Setten, 2001), connaître l'utilisateur c'est :

- Connaître ses centres d'intérêt ; c'est-à-dire savoir pourquoi il est intéressé par un certain type d'informations et pas par un autre.
- Connaître ses préférences ; c'est-à-dire savoir ce qu'il préfère pour chaque type et/ou chaque catégorie d'information. La différence avec les centres d'intérêt réside dans le fait qu'ils sont généralement définis de manière implicite par l'utilisateur alors que les préférences sont définies de manière explicite.

- Connaître ses tâches et ses buts ; c'est-à-dire savoir ce qu'il est en train de faire et pourquoi il le fait afin de lui apporter une personnalisation adaptée à ses besoins et à ses attentes métier au moment de l'interaction.
- Connaître ses caractéristiques ; c'est-à-dire les éléments objectifs qui permettent de caractériser un individu comme, par exemple, son genre (masculin ou féminin), sa date de naissance ou encore sa profession.

Pour un système d'information, cette connaissance de l'utilisateur implique de résoudre deux problématiques :

- Celle du stockage de l'ensemble des informations recueillies, sur chaque utilisateur, nécessaires dans le cadre de la personnalisation. Celle-ci se fait généralement à travers l'utilisation de profils utilisateurs qui contiennent, pour chaque utilisateur, l'ensemble des informations qui lui sont spécifiques (Amato et Straccia, 1999).
- Celle des outils et des méthodes à mettre en place pour recueillir cette information. Ce recueil des informations peut se faire de manière explicite à travers l'utilisation de questions auxquelles l'utilisateur doit répondre lui-même (Shardanand et Maes, 1995). Elle peut aussi se faire de manière implicite (automatique) par le système, c'est-à-dire sans aucune intervention de l'utilisateur (Goecks et Shavlik, 2000), à travers une analyse du comportement de l'utilisateur et aussi de l'historique de ses interactions avec l'utilisateur. Elle peut enfin se faire de manière mixte, l'utilisateur confirmant les informations proposées par le système.

Dans un système de personnalisation des informations basé sur un moteur de recommandations, la connaissance de l'utilisateur passe aussi par la notion de contexte identitaire partagé (Pomerol et Brézillon, 2001) où plus exactement par la notion de stéréotype qui est un ensemble de caractéristiques communes à un groupe ou à une classe d'utilisateurs. Dans le cadre de la personnalisation, les stéréotypes sont très souvent utilisés pour guider l'utilisateur à partir d'une reconnaissance de ses buts (Kobsa, 2001) (Virvou et Kabassi, 2002) ou pour inférer ses centres d'intérêt (Waern *et al.*, 1999) (Shardanand et Maes, 1995).

Il est intéressant de noter que la connaissance de l'utilisateur, dans le cadre de la personnalisation, passe aussi par la connaissance de son expérience avec le système c'est-à-dire par une exploitation de l'historique des interactions entre le système et l'utilisateur. Mais, elle passe aussi par l'expérience des autres utilisateurs avec le système (Mobasher *et al.*, 2000) de manière à pouvoir prédire, en comparant avec ce qu'ont déjà fait les autres utilisateurs, ce que va faire l'utilisateur.

La nature des informations permettant de caractériser l'utilisateur étant maintenant précisée, il convient de définir comment il est possible de modéliser l'utilisateur au niveau conceptuel.

2.2.2 Étude des outils et méthodes de modélisation de l'utilisateur

Nos travaux n'ayant pas pour but principal de définir un nouveau modèle d'utilisateur, nous ne ferons pas dans cette partie une présentation exhaustive de toutes les possibilités de modélisation de l'utilisateur. Ainsi, nous limiterons la présentation des outils et des méthodes de modélisation de l'utilisateur à ceux qui sont directement utilisables au niveau de la modélisation des applications. Nous ne nous intéresserons pas non plus aux différents facteurs humains comme les modèles mentaux et les modèles comportementaux de l'utilisateur ; voir plusieurs chapitres à ce sujet dans (Jacko et Sears, 2003). Nous nous limiterons aussi volontairement à ne présenter que les solutions permettant de modéliser un utilisateur unique et pas un groupe d'utilisateurs.

En matière de modélisation des utilisateurs, (Rich, 1983) identifie un espace à trois dimensions pour les modèles utilisateurs dans lequel chaque information sur l'utilisateur peut être placée suivant :

- Une dimension avec le modèle canonique (information pouvant être commune à plusieurs utilisateurs) d'un côté et le modèle individuel de l'autre (information spécifique à l'utilisateur).

- Une dimension avec le modèle explicite d'un côté (information recueillie de manière explicite) et le modèle implicite de l'autre (information recueillie de manière implicite).
- Une dimension avec le modèle à court terme d'un côté (information dont le contenu change très souvent) et le modèle à long terme de l'autre (information dont le contenu est stable dans le temps).

(Amato, 1999) propose pour sa part, dans le cadre d'une librairie digitale (électronique), un modèle de profil contenant cinq catégories (ce modèle ayant été repris et complété par (Kostadinov, 2003)):

- Données personnelles : contient les informations sur l'identité de l'utilisateur.
- Données collectées (contenu du document, structure du document, source du document) : contient les informations sur les préférences et restrictions sur les documents.
- Données de livraison (moyen de livraison, moment de livraison) : contient les informations sur la manière d'envoyer les documents à l'utilisateur.
- Données de comportement : contient les informations sur les interactions de l'utilisateur avec le système
- Données de sécurité : contient les informations sur les conditions d'accès aux données du profil.

Dans tous les cas, un point important à retenir est que le profil est dépendant du contexte dans lequel se trouve l'utilisateur (Rich, 1983) (Dinoff *et al.*, 2006). Aussi, pour chaque utilisateur, il n'existe pas un profil unique mais un ensemble de profils adaptés à chaque contexte dont certaines informations sont communes à l'ensemble des contextes comme le nom et le prénom de l'utilisateur par exemple.

En matière de stockage des informations de profil, la modélisation des utilisateurs a donné lieu, dans un premier temps, à la création d'arborescences plus ou moins simples, souvent spécifiques aux applications, dans lesquelles les informations sur l'utilisateur étaient stockées. La question qui s'est posée alors était de savoir comment permettre la réutilisation de ces profils par différentes applications. Pour cela, la solution a été de développer des outils génériques permettant de créer et de gérer des profils utilisateurs. Pour donner quelques exemples de ce type d'outils, on peut citer :

- Le "Believe, Goal, Plan – Management System" (BGP-MS) de (Kobsa, 1993) (Kobsa et Pohl, 1995) qui permet de construire un modèle schématique de l'utilisateur à travers des stéréotypes qui sont sauvegardés dans une partition système hiérarchisée.
- Le "General User Modeling Shell" (GUMS) de (Finin et Drager, 1986) (Finin, 1989) qui permet, en se basant sur une architecture indépendante des plateformes d'utilisation, de développer des modèles d'utilisateur individuels.
- Le "PROlog bases Tool for User Modeling" (PROTUM) de (Vergara, 1994) qui est une synthèse de BGP-MS et de GUMS.
- Le "Theory and Application for General User/learner modeling System" (TAGUS) (Paiva et Self, 1994) qui est un outil spécialement développé pour les applications interactives dans le domaine de l'apprentissage et de la formation.
- Le "User Modeling Toolkit" (UMT) de (Brajnik et Tasso, 1994) qui est un système générique flexible de développement de modèles utilisateurs.

De manière générale, l'ensemble de ces outils ont surtout été conçus pour permettre de gérer les informations implicites du profil utilisateur et de les faire évoluer dans le temps si nécessaire. Pour une présentation plus détaillée de ces différents outils, le lecteur pourra se référer à (Kobsa, 2001).

À partir du moment où ces moteurs ont commencé à exister, il est devenu possible de créer des profils utilisateurs contenant l'ensemble des informations sur l'utilisateur (informations explicites et informations implicites). Pour avoir une idée plus générale des méthodes de construction des profils utilisateur, on pourra consulter les travaux de (Rich, 1983), de (Cornelis, 2001) ou encore de (Kobsa, 1993). Afin de pouvoir gérer, dans les applications, toutes les informations du profil, (Gauch *et al*, 2003) et (Trajkova et Gauch, 2004) ont proposé l'utilisation d'une ontologie dans laquelle le profil utilisateur est structuré dans une hiérarchie de concepts pondérés. Cette pondération est effectuée en

fonction de la quantité d'informations rattachées à chaque concept, en partant du principe que plus il y a d'informations associées à un concept et plus ce concept doit être important pour l'utilisateur. Ces pondérations sont ensuite utilisées lors des recherches pour trier les informations retournées en favorisant les informations contenant les concepts possédant les pondérations les plus élevées. En parallèle de ces travaux, (Razmerita *et al.*, 2003) (Razmerita, 2005) ont proposé, dans le domaine de l'apprentissage et de la formation, une architecture générique de modélisation des profils utilisateurs se basant, elle aussi, sur une ontologie. À partir de ces travaux, (Golemati *et al.*, 2007) ont proposé une méthode générale d'utilisation d'une ontologie dans le cadre des profils utilisateurs. La Figure 2.4 présente un extrait de cette ontologie telle qu'elle peut être visualisée dans l'outil Protege. Dans cette ontologie, chaque utilisateur possède :

- Des éléments (des individus) généraux comme sa profession ou encore son niveau d'éducation qui permettent de définir son état-civil.
- Des éléments plus personnels comme ses centres d'intérêt, ses préférences ou encore ses niveaux d'expertises qui permettent de définir ses caractéristiques propres.
- Des éléments relationnels comme ses contacts qui permettent de définir son réseau relationnel.

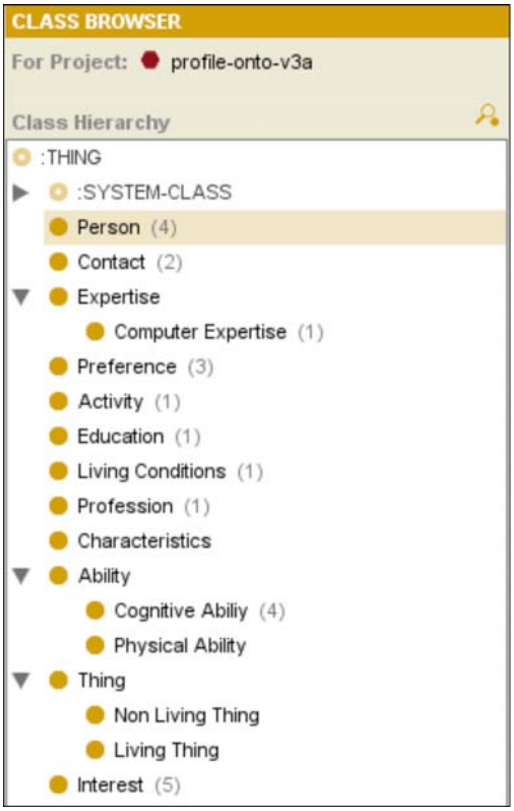


Figure 2.4. L'ontologie de profil utilisateur proposé par (Golemati *et al.*, 2007) dans l'outil Protege

Après ce tour d'horizon sur les différentes méthodes de modélisation de l'utilisateur, la question se pose de savoir si ces outils sont vraiment aussi génériques qu'ils le prétendent.

2.2.3 Synthèse et discussion

Pour pouvoir être utilisé dans le cadre d'une modélisation conceptuelle des applications, un modèle d'utilisateur doit pouvoir répondre à trois critères majeurs :

- Il doit être générique pour pouvoir être utilisé dans tout type d'application.
- Il doit être extensible pour permettre de prendre en compte de nouvelles propriétés ou de nouveaux critères.
- Il doit permettre la prise en compte de la notion de contexte.

Vis-à-vis de ces trois critères, les différentes solutions présentées apportent des réponses plus ou moins adaptées (cf. Tableau 2.1).

Tableau 2.1. Comparaison des différentes solutions de modélisation du profil utilisateur

Solution	Généricité	Extensibilité	Prise en compte du contexte
(Rich, 1983)	**Oui** Vision générale des profils	**Oui**	**Non** Chaque nouveau contexte conduit à devoir créer nouveau profil.
(Amato, 1999)	**Non** Spécialement dédié à la gestion d'une librairie digitale (électronique).	**Oui**	**Non** Pas de notion de contexte dans l'ensemble du modèle.
(Trajkova et Gauch, 2004)	**Non** Spécialement dédié à la recherche d'information sur le web.	**Oui**	**Non** Le seul contexte pris en compte est le contexte applicatif.
(Razmerita, 2005)	**Non** Spécialement développé pour le contexte de l'apprentissage.	**Oui**	**Oui** Prise en compte limité du contexte dans le cadre des éléments directement reliés à l'apprentissage.
(Golemati et al., 2007)	**Oui** Approche prévue pour être générique.	**Oui**	**Oui** L'utilisation d'une ontologie générique permet d'associer à chaque propriété un contexte d'usage limité aux informations contenues dans le profil

À la lecture de ce tableau, il apparaît que la solution la plus souple aujourd'hui pour modéliser l'utilisateur passe par l'utilisation d'une ontologie. Néanmoins, ceci pose de nombreuses questions dans le sens où les ontologies présentent encore aujourd'hui un certain nombre de problématiques non résolues (cf. chapitre 1, §1.4). D'autre part, se pose la question, au niveau de la modélisation conceptuelle des applications, des interactions entre le modèle utilisateur et les autres modèles. En effet, celles-ci doivent être suffisamment complètes pour permettre de modéliser tous les types d'applications et suffisamment indépendantes pour permettre de faire évoluer le modèle utilisateur sans avoir besoin de modifier tous les autres modèles. Ce point étant assez complexe, il sera traité dans les chapitres 3 et 4 de cette thèse.

Une autre question qui n'a pas été traitée jusqu'à maintenant est de savoir si l'utilisation d'un modèle utilisateur basé sur des informations de profil est réellement suffisante pour modéliser complètement les utilisateurs. La réponse est non car, dans le cadre d'une application, la connaissance de l'utilisateur est importante pour la personnalisation mais aussi pour tout ce qui touche à la sécurité d'accès aux différentes parties de l'application. Pour cela, on peut associer un modèle de sécurité de type Role Based Acces Control (RBAC) (Ferraiolo et Kuhn, 1992) (Sandhu *et al.*, 1996) qui, en se basant sur des rôles utilisateurs, permet de gérer l'ensemble des accès à l'application. Mais, dans ce cas, il faut s'interroger sur les liens à établir entre ce modèle de type RBAC et le modèle utilisateur défini par le profil utilisateur. Les modèles RBAC s'appuyant très souvent sur l'utilisation d'une hiérarchie sociale basée sur l'organisation interne d'une entreprise ou de la société civile, la question se pose aussi de savoir comment modéliser cette organisation sociale et comment l'intégrer dans les modèles conceptuels.

La modélisation de l'utilisateur, si elle peut paraître simple au premier abord, est en fait un élément multifacettes. Ainsi, du point de vue de la personnalisation, elle est abordée à travers une vision contextuelle des propriétés implicites et explicites caractérisant l'utilisateur. Du point de vue de la

sécurité d'accès aux applications, elle est vue comme une façon d'organiser les accès. Enfin, du point de vue organisationnel, elle est vue comme une façon d'avoir une image de la hiérarchie sociale dans laquelle l'utilisateur évolue. Ces visions complémentaires et pourtant disjointes nous imposent de voir la modélisation de l'utilisateur à travers différents modèles ; chaque modèle permettant de résoudre une problématique bien particulière. Ces modèles forment un tout qui permet de caractériser complètement l'utilisateur et c'est certainement là un des principaux défis d'une méthode de modélisation contextuelle.

Comme nous l'avons rapidement vu dans le cadre de cette partie, le profil de l'utilisateur est dépendant du contexte et on ne peut parler de personnalisation que si on prend en compte ce contexte. Aussi, est-il important de définir et surtout de caractériser cette notion de contexte.

2.3 La modélisation du contexte dans le cadre de la personnalisation

2.3.1 Définition

Avant de montrer l'influence du contexte dans le cadre de la personnalisation des contenus, il est essentiel de commencer par préciser la notion de contexte. Pour cela, on peut reprendre la définition la plus communément utilisée qui est celle de (Dey *et al*, 2000) et (Dey, 2001) : *"Le contexte représente toutes les informations qui peuvent être utilisées pour caractériser une situation ou une entité. Une entité est une personne, un lieu ou un objet qui est considéré comme significatif au niveau de l'interaction entre l'utilisateur et l'application, en y incluant l'utilisateur et l'application elle-même. Un système est sensible au contexte s'il utilise le contexte pour fournir des informations ou des services pertinents à l'utilisateur ; la pertinence étant évaluée en fonction de la tâche en cours de réalisation par l'utilisateur. La sensibilité au contexte est la facilité de s'adapter aux contextes. Les applications sensibles au contexte s'adaptent en fonction du lieu de leur utilisation, de l'ensemble des personnes présentes, des serveurs et moyens techniques utilisables et de leur évolution dans le temps. L'application examine l'environnement informatique et réagit aux changements. "*

Dans le cadre d'une application interactive, la prise en compte du contexte peut se faire à deux niveaux :

- Au niveau de l'interface et, dans ce cas, on parlera de Plasticité. Ainsi, pour (Thévenin et Coutaz, 1999) la plasticité, c'est "*La capacité des interfaces de s'adapter à leur contexte d'usage dans le respect de leur utilisabilité. Le contexte d'usage se définit comme le triplet utilisateur, plate-forme et environnement.*"
- Au niveau des contenus et, dans ce cas, on parlera de context awareness. Ainsi, pour (Abowd *et al.*, 97), le context awareness peut être défini comme "*l'utilisation du contexte pour fournir des informations appropriées et/ou des services à l'utilisateur ; le contexte étant n'importe quelle information qui peut être utilisée pour caractériser la situation d'une entité qui peut être un utilisateur, un environnement, un objet physique ou informatique*".

La notion de contexte étant maintenant définie de manière globale, il convient d'en préciser les différentes dimensions dans le cadre de la personnalisation des contenus. Ainsi, pour (Van Setten, 2001), le contexte est défini par :

- Une dimension applicative : la personnalisation effectuée peut être dépendante du domaine métier associé à l'application voire même à l'application elle-même.
- Une dimension technique : les capacités de la plateforme de consultation de l'utilisateur, la capacité des réseaux et l'infrastructure technique associée à l'application peuvent imposer un certain nombre de contraintes sur la personnalisation.
- Une dimension environnementale : l'environnement physique dans lequel l'application est utilisée (à domicile, dans une gare, dans un train, etc.), le lieu d'accès à l'application et le moment où l'application est accédée peuvent imposer un certain nombre de contraintes sur la personnalisation.

2.3.2 Étude des outils et méthodes de modélisation du contexte

Dans le cadre d'une approche de modélisation dirigée par les modèles, la question qui se pose est de savoir comment prendre en compte le contexte au niveau conceptuel ; c'est-à-dire, comment le rendre suffisamment abstrait pour qu'il soit indépendant de la technologie et suffisamment concret pour qu'il puisse être utilisé au niveau des modèles conceptuels.

Aujourd'hui, dans le cadre de nos recherches, nous avons trouvé peu de travaux prenant en compte la modélisation conceptuelle des contextes. Et, lorsque celle-ci est prise en compte c'est souvent de manière incomplète et, bien souvent, elle est limitée à des adaptations contextuelles liées au type de plateforme utilisée (Carton *et al.*, 2007) ou alors elle ne prend en compte que l'adaptation des IHM (Halin et Kubicki, 2005). Néanmoins, en partant des différentes solutions, nous avons pu déterminer plusieurs façons différentes d'aborder la caractérisation des contextes :

- La première, utilisée par exemple par (Tadj et Ngantchaha, 2006), consiste à ne définir qu'un nombre limité et déterminé de contextes. De cette manière, en se basant sur un ensemble de règles, il est possible, à chaque instant, d'associer à l'application un et un seul contexte bien déterminé.
- La deuxième, utilisée par exemple par (Chen *et al.*, 2004), propose, en se basant sur la solution précédente, d'introduire la notion d'ontologie au niveau des contextes. Dans cette approche, chaque élément du contexte (lieu, temps, capacités techniques, etc.) possède sa propre ontologie. C'est l'utilisation d'un moteur d'inférence sur l'ensemble de ces ontologies qui permet de déterminer le contexte ; celui-ci faisant partie lui-même d'une ontologie.
- La troisième consiste à lier la notion de contexte aux activités en cours de réalisation par l'utilisateur. Ainsi, (Bardram, 2004) propose de baser l'adaptation au contexte en fonction des processus métier en cours de réalisation par l'utilisateur. (Pascoe *et al.*, 1999), proposent, pour leur part, d'intégrer la notion d'intention au niveau de l'utilisateur dans l'adaptation au contexte. Ainsi, en présumant de ce que l'utilisateur souhaite faire, c'est-à-dire à partir des actions qu'il a réalisées, il est possible d'anticiper ses besoins et donc d'avoir une adaptation plus pertinente des applications au contexte en diminuant l'incertitude sur les besoins réels de l'utilisateur. C'est, selon eux, la seule manière d'avoir une adaptation contextuelle qui prenne vraiment en compte les besoins et les attentes des utilisateurs.

Si toutes les solutions proposent des visions différentes pour la prise en compte du contexte dans les applications, elles présentent néanmoins un certain nombre de similarités :

- Elles possèdent toutes un moteur de règles, où des composants "intelligents" permettant de définir les actions à réaliser en fonction des changements survenant au niveau du contexte.
- Elles se basent toutes sur une définition conceptuelle du contexte permettant de séparer les éléments techniques prenant en compte le contexte des éléments applicatifs qui y sont sensibles. Pour la majorité des solutions utilisées aujourd'hui, cette abstraction se fait à l'aide d'une ontologie de contexte.
- La prise en compte du contexte, dans une application, ne peut se faire qu'à travers la définition et l'utilisation de règles spécifiques définies à l'avance. Une application ne peut réagir à un contexte que s'il est prévu, dès sa conception, qu'elle y réagisse et si on a indiqué, lors de la création de l'application, comment elle doit y réagir.

2.3.3 Synthèse et discussion

Ces caractéristiques communes impliquent que les trois solutions pourraient être modélisées d'une manière très similaire, au niveau conceptuel, dans le cadre d'une approche IDM :

- Le contexte serait modélisé à travers une liste déterminée et limitée d'états possibles définis au sein d'une ontologie.
- Dans les autres modèles, on ne pourrait utiliser la notion de contexte qu'à travers l'utilisation de règles basées sur les différents états possibles du contexte.

Si cette utilisation de la notion de contexte au niveau conceptuel peut paraître simple et pertinente, elle présente néanmoins un inconvénient majeur. Ainsi, elle ne permet de prendre en compte les différents éléments du contexte que de manière globale et pas de manière individuelle. Pour donner un exemple, si on veut donner un comportement particulier à une application le premier lundi matin de chaque mois de 9H00 à 10H00 lorsque l'utilisateur se trouve en France et un autre lorsque l'utilisateur ne se trouve pas en France, on est obligé de définir deux états de contexte pour cette situation bien précise. Par la suite, si jamais on désire avoir un comportement par pays, il faudra avoir autant d'états de contexte que de comportements différents souhaités. Et, si jamais on désire un jour modifier la plage horaire d'activation du comportement, il faudra modifier individuellement l'ensemble des états ce qui pourra demander beaucoup de temps. Cette multiplication des états et cette problématique de maintenance sont aujourd'hui les limites principales de l'utilisation des états de contexte.

Pour répondre à ces deux limitations, une solution serait de définir chaque élément du contexte (temps, espace, niveau sonore, etc.) comme étant à l'origine d'un modèle spécifique à part entière. De cette manière, il devient possible d'utiliser chaque élément du contexte de manière individuelle en permettant, par exemple, une adaptation d'un processus métier uniquement en fonction du temps ou de l'espace. La question qui se pose alors est de savoir comment faire lorsqu'on a besoin d'une adaptation en fonction de plusieurs éléments de contexte différents comme le temps et l'espace. Une solution possible consiste à définir une notion de "contexte composé" sous la forme de règles de validation (règles retournant uniquement la valeur Vrai ou la valeur Faux) pouvant associer plusieurs éléments différents du contexte. De cette manière, il devient possible de créer de nombreux contextes différents parfaitement adaptés aux besoins de chaque application. Pour donner un exemple, à travers l'utilisation d'un "contexte composé", il devient possible, pour un processus métier, de définir un comportement particulier le vendredi matin lorsqu'on se trouve dans une gare et un autre le dimanche lorsqu'on se trouve dans un véhicule et qu'on est dans la tranche horaire 12H00 – 14H00. Ainsi, un processus métier associé à la recherche d'informations de loisir pourrait afficher, de manière préférentielle, la liste des spectacles et concerts du week-end le vendredi matin et la liste des restaurants ouverts le dimanche entre 12H00 et 14H00.

Si la solution proposée semble être intéressante, au niveau conceptuel, dans le cadre d'une approche IDM, elle présente néanmoins un inconvénient important. Ainsi, si dans une approche IDM, on impose que l'adaptation au contexte ne peut être définie qu'au niveau conceptuel, il devient alors impossible de faire des adaptations par rapport à des éléments techniques du contexte comme, par exemple, le type de plateforme puisque cette notion n'existe pas au niveau conceptuel. En fait, il est possible de contourner le problème en ne faisant pas ce type d'adaptation au niveau conceptuel mais en le prenant en compte au niveau PIM ou au niveau PSM de l'architecture MDA associée à l'approche IDM (cf. Chapitre 1, Figure 2). Ceci impose de revoir de manière globale l'approche MDA de manière à permettre ce type d'adaptation qui doit être totalement indépendant des modèles définis au niveau conceptuel.

Si, au niveau de la personnalisation des contenus, la prise en compte du contexte est un élément important pour obtenir une personnalisation pertinente et efficace, un autre élément doit aussi être pris en compte : les contenus eux-mêmes.

2.4 La modélisation des contenus dans le cadre de la personnalisation

2.4.1 Définition

Dans le cadre de nos travaux nous avons limité la prise en compte de la personnalisation uniquement au niveau des contenus qui peuvent être soit des informations soit des services. Au niveau des informations (des données), celles-ci peuvent être de trois types différents :

- Des données structurées (Codd, 1970). Ce sont des données qui respectent un certain schéma préétabli au niveau de leur contenu. Les contenus sont généralement répartis dans des tables dans lesquelles chaque enregistrement possède la même structure. C'est le type de données utilisées dans les bases de données relationnelles.
- Des données semi structurées (Buneman, 1997). Ce sont des données pour lesquelles ont ne dispose pas d'un schéma prédéfini ; celui-ci étant défini, *a posteriori*, dans les données elles-mêmes. Pour donner un exemple, les contenus des pages web et les fichiers XML sont des données de type semi structurées.
- Des données non structurées. Ce sont des données pour lesquelles il est impossible de définir une structure directement à partir de leur contenu. Par contre, il est possible de les caractériser en leur associant des métadatas qui sont représentées sous forme de données structurées. Une vidéo ou un document issu d'un traitement de texte sont deux exemples de données non structurées.

Si ces trois types de données semblent bien différents, un certain nombre de travaux ont montré qu'il était possible de les appréhender à travers une approche commune (Magnani et Montesi, 2004).

En ce qui concerne la notion de service, dans le cadre de la personnalisation des contenus, celle-ci peut être définie comme suit :

- Un service regroupe un ensemble de processus métier qui permettent à l'utilisateur de réaliser un ou plusieurs buts métier dans un domaine métier bien particulier. Un exemple de service pourrait être un service de musique en ligne qui regrouperait différents processus métier permettant de gérer, d'écouter et d'acheter de la musique en ligne.
- Un service est soit obligatoire soit optionnel. Dans le cas d'un service obligatoire, celui-ci est disponible pour l'utilisateur dans un contexte fonctionnel et environnemental particulier et l'utilisateur ne peut pas le rendre indisponible. S'il est optionnel, il est aussi disponible pour l'utilisateur dans un contexte fonctionnel et environnemental particulier, mais l'utilisateur peut choisir de ne pas y avoir accès à travers son IHM. Dans le cadre de l'information transport, un exemple de service obligatoire pourrait être l'accès à la liste des prochains départs lorsque l'utilisateur se trouve à une station (train, bus, tram ou métro). Un exemple de service optionnel pourrait être un service payant de formation en ligne qui pourrait donner accès, lorsque l'utilisateur a payé le droit d'entrée, à un certain nombre de formations et à la gestion de celles-ci.
- Un service peut être implicite ou explicite. Dans le cas où le service est explicite, celui-ci est directement accessible pour l'utilisateur. Dans le cas d'un service implicite, le service est rendu disponible par le système, mais l'utilisateur ne peut pas y accéder directement. Un exemple de service implicite pourrait être un service de recueil d'historique de navigation dans les IHM.

Dans le cadre d'une approche dirigée par les modèles, la question qui se pose ensuite et de savoir comment il est possible de modéliser conceptuellement les contenus et les données.

2.4.2 Étude des outils et méthodes de modélisation des contenus

Dans le cadre de notre thèse, nous n'avons pas trouvé de travaux spécifiques associés à la notion de service telle que nous l'avons définie dans la section précédente. En effet, dans la littérature, la notion de service est aujourd'hui souvent associée à celle de service web telle qu'elle est définie par exemple par Sun dans le cadre Java (http://java.sun.com/webservices/). Or ce type de service est beaucoup plus restrictif que ce que nous avons défini car, s'il contient bien des fonctionnalités qui peuvent être associée entres elles, il ne contient pas la notion de processus métier. Voilà pourquoi, il nous semble plus pertinent de définir ces services à l'aide de processus métier. Dans ce cadre, la personnalisation se traduit par l'utilisation de règles métier qui sont valides ou pas en fonction du contexte fonctionnel et/ ou environnemental. Cette approche sera détaillée dans le chapitre 3 de ce mémoire.

En ce qui concerne les données, on dispose historiquement, pour les données structurées, de deux grands types possibles de modélisation :

- Le modèle entité-relation de (Chen 1976) qui permet de définir des ensembles de données cohérents du point de vue métier ainsi que des règles permettant d'établir des relations entre les différentes entités.
- La modélisation Merise de (Tardieu *et al.,* 1983) (Nanci et Espinasse, 2001) qui permet de créer des modèles dit "conceptuels" de données sur un principe assez similaire à celui du modèle entité-relation.

Le problème de ces deux types d'approches réside dans le fait que les modèles créés le sont dans le but de permettre leur utilisation dans le cadre technique d'une base de données relationnelle. Aussi, ce manque d'indépendance vis-à-vis de la technique ne permet donc pas de parler de véritables modèles conceptuels tels qu'ils sont envisagés dans le cadre d'une approche MDA.

Pour contourner ce problème, on s'oriente, aujourd'hui, vers une abstraction beaucoup plus grande par rapport aux données structurées à travers l'utilisation d'ontologies (Aparicio *et al.,* 2005) (Trinkunas et Vasilecas, 2007). De cette manière, il devient possible de travailler sur des concepts, des propriétés, et des relations sémantiques directement reliées au domaine métier et plus sur des structures de données. À côté de cette grande tendance, d'autres travaux (Scappapietra, 2007) proposent d'utiliser des modélisations conceptuelles dédiées pour certain types d'informations : données multimédia, données ontologiques ou données dépendantes du contexte.

Pour les données non structurées, les premières modélisations envisagées considéraient que celles-ci pouvaient être manipulées comme des données structurées à travers l'utilisation des métadatas, Cette technique ne permettant pas de caractériser le contenu même de la donnée non structurée, d'autres travaux ont ensuite essayé d'extraire cette information de la donnée non structurée elle-même. Pour donner quelques exemples :

- (Adelberg, 1998) a proposé une technique pour extraire des données structurées et des données semi structurées d'un document sous format texte qui par nature est une donnée non structurée.
- (Chu *et al.,* 2007) ont proposé un système de recherche sur des informations non structurées en se basant sur le principe que chaque information non structurée contient une structure sous-jacente.
- (Embley *et al.,* 1998) ont proposé pour leur part d'utiliser une ontologie pour extraire des données structurées d'un ensemble de données non structurées.
- (Sarvas *et al.,* 2004) ont proposé une technique de création des métadatas de manière semi-automatique à partir d'images prises sur un téléphone portable en fonction du contexte et d'informations saisies par l'utilisateur.

À partir du moment où il est possible d'associer des données non structurées à des données structurées, il devient alors possible d'utiliser les mêmes modèles que pour les données structurées.

Pour les données semi structurées, qui sont souvent des données au format XML, il existe de nombreuses propositions de modélisation qui passent, par exemple :

- Par des modèles de type entité-relation adaptés (Loscio *et al.*, 2003)
- Par des extensions au langage UML (Liu *et al.*, 2006).
- Par des langages de description spécifiques comme XSEM (XML Scripture Encoding Model) (Necasky, 2007) qui permet d'associer un schéma de données à chaque type d'informations dans le domaine des documents littéraires.
- Par une intégration des fichiers XML dans des systèmes de gestion de données structurées (Florescu et Kossmann, 1999) à travers un mapping direct entre des données XML et des tables se trouvant dans une base de données relationnelles.
- Par la génération de métadatas (Stuckenschmidt et Van Harmelen, 2001) à partir d'ontologies ; ce qui permet de percevoir les données semi structurées comme des données non structurées particulières.

De manière globale, l'objectif de l'ensemble de ces travaux est de permettre de transformer les données semi structurées en données structurées ou non structurées sur lesquelles il devient alors possible d'effectuer des recherches.

En résumé, si on est capable de modéliser conceptuellement des données structurées, on est aussi capable de modéliser tous les types de données. La question qui se pose alors et de savoir de quels outils et de quelles méthodes nous disposons aujourd'hui pour effectuer cette modélisation.

2.4.3 Synthèse et discussion

En faisant la synthèse des différents travaux présentés ci-dessus, il apparaît qu'un des outils de modélisation utilisable pour tous les types de données est l'ontologie. Celle-ci peut être une ontologie métier ou une ontologie spécifique aux types de données dans le cadre de données non structurées ou de données semi structurées. Elle peut aussi associer les deux lorsque les métadatas sont reliées au métier. La question est alors de savoir comment utiliser cette ontologie dans le cadre de la modélisation conceptuelle et plus particulièrement dans le cadre de la personnalisation.

Dans cette partie, nous ne reviendrons pas sur la notion d'ontologie qui a déjà été présentée en détail (cf. chapitre 1, §1.4). Néanmoins, il est important de noter que les ontologies n'ont pour l'instant pas souvent été utilisées comme un outil de modélisation ou d'aide à la modélisation des données même si l'intérêt d'une modélisation conceptuelle des données a déjà été démontré (Turk *et al.*, 2003). Ainsi, la problématique généralement étudiée est celle de l'intégration de données structurées dans une ontologie en vue de créer une base de connaissance. Pour des exemples de ce type de travaux, on pourra se reporter à (Trinkunas et Vasilecas, 2007) ou encore à (Xu *et al.*, 06). Il existe cependant quelques travaux prometteurs qui ont été réalisés dans le domaine. Ainsi, (Sugumuran et Storey, 2006) ont proposé une méthode pour faciliter la conception des bases de données relationnelles en partant d'une ontologie du domaine métier. L'objectif de notre thèse n'étant pas de développer une nouvelle méthode de modélisation des données à partir d'une ontologie, nous n'entrerons pas plus dans le détail de ces différents travaux.

Concernant les aspects de personnalisation, il est important de noter qu'il n'est possible de personnaliser des informations que si celles-ci disposent des éléments nécessaires à cette personnalisation. Pour donner un exemple, si on veut personnaliser des données en fonction de la localisation géographique de l'utilisateur, ceci ne sera possible que si on associe à chaque donnée un ou plusieurs espaces géographiques de validité. Il est important de noter que ceci est valable au niveau de la donnée et pas forcément au niveau de la source de données. Ainsi, dans un même ensemble de données, comme, par exemple, des informations générales sur les perturbations transports, il sera possible de trouver des informations avec des restrictions géographiques et d'autre sans. Le contenu des données n'étant jamais connu au niveau conceptuel, ceci implique de développer, dans le cadre d'une démarche de type IDM, des outils de personnalisation capables de prendre en compte des informations possédant ou ne possédant pas des données permettant la personnalisation.

Au niveau de l'ensemble des modèles conceptuels, ceci impose de voir la personnalisation comme un service fonctionnel qu'il est possible d'appliquer sur n'importe quel élément d'information et pas comme un service technique. Ainsi, dans un modèle conceptuel, on pourra indiquer qu'on désire personnaliser certaines informations en fonction de la localisation géographique de l'utilisateur ou encore de ses préférences mais on ne pourra pas indiquer comment cette personnalisation sera effectuée. Ceci permettra :

- De rendre, du point de vue de la personnalisation, les applications entièrement portables d'un système technique à un autre.
- De s'affranchir totalement des capacités techniques de chaque système de personnalisation.

Dans le chapitre 3 de notre thèse, nous montrerons comment cette abstraction est effectuée et surtout ce qu'elle apporte concrètement dans le cadre d'une approche IDM.

Si, comme nous venons de le voir, la personnalisation des contenus est un domaine vaste et varié, l'utilisation de la personnalisation dans le domaine des transports présente elle aussi de nombreuses particularités et spécificités que nous allons maintenant présenter.

2.5 La personnalisation dans le domaine des transports

2.5.1 Les besoins dans le domaine des transports

Avec la multiplication et la généralisation des systèmes informatiques nomades, l'utilisateur peut consulter de l'information n'importe où et à n'importe quel moment. Dans le domaine de l'information transport, ceci se traduit pour l'usager par la possibilité de consulter de l'information transport aussi bien à son domicile que pendant ses déplacements. Si ceci peut présenter un champ plutôt vaste de possibilités, les critères attendus par les usagers ont déjà été identifiés à travers les travaux réalisés par (Lecomte et Patesson, 2000), par (Lyons, *et al.* 2001) ou encore dans le cadre du projet Viatic.Mobilité (http://viatic.inrets.fr) qui sera présenté en détail dans le chapitre 5. Ceux-ci sont au nombre de cinq :

- Avoir de l'information en temps réel sur les perturbations (retards et annulations) et sur les horaires (arrivées et départs).
- Avoir accès à des solutions de secours en cas de perturbation (utilisation d'autres lignes et/ou d'autres moyens de transport pour pouvoir continuer le déplacement).
- Disposer de solutions de transport qui prennent en compte tous les modes de déplacement (train, métro, tram, bus, taxi collectif, etc.).
- Obtenir des informations personnalisées en fonction de leur profil et de leurs habitudes de déplacement.
- Avoir accès à l'information transport pendant le déplacement sur tout type de système d'information (téléphone portable, PDA, borne interactive, etc.).

Ces critères étant posés, il est aussi important de voir que l'information transport représente un enjeu important dans le cadre des nouvelles politiques de déplacement mises en place au sein des différents pays.

Ainsi, aujourd'hui, pour lutter contre la congestion des villes au niveau de la circulation routière et aussi pour diminuer les émissions de gaz polluants, les villes cherchent de plus en plus à limiter la circulation automobile. Pour illustrer ceci, on pourra prendre l'exemple de la ville de Londres qui, depuis 2005, a établi un péage urbain (le London Congestion Charge), dissuasif au niveau financier, pour accéder au coeur de la ville. L'objectif recherché, dans ce type de politique est de favoriser les transports en commun et les véhicules moins polluants tout en facilitant la circulation des piétons et des véhicules au sein de la ville. Si ces solutions ont prouvé leur efficacité, aujourd'hui, toutes les villes ne sont pas prêtes pour mettre en place des systèmes aussi dissuasifs. D'un autre côté, les transports en commun sont bien souvent mal perçus par les usagers potentiels du fait d'*a priori* sur les difficultés d'organiser des déplacements et sur les conditions dans lesquelles s'effectuent ces déplacements. Du côté matériel, des moyens importants ont été investis ces dernières années pour moderniser les différents moyens de transport et pour améliorer leur confort. Par contre, de gros efforts restent à faire du côté de l'information transport pour rendre les déplacements beaucoup plus

simples (JPF Consultant, 1996) (Lecomte *et al.*, 2000) (Lyons *et al.*, 2001). Ceci est d'autant plus vrai que ces déplacements nécessitent d'emprunter plusieurs modes de transport différents (Perreau, 2002) (ITS, 2002). Aussi, est-il important d'identifier les réels besoins des usagers. Ainsi, en partant de ces différents travaux cités précédemment, il apparaît qu'un système d'information voyageur doit :

- Informer les utilisateurs des possibilités qui leur sont offertes.
- Inciter les utilisateurs à utiliser les transports collectifs pour leurs déplacements aussi bien pour le travail que pour les loisirs.
- Fournir aux utilisateurs le meilleur itinéraire pour un déplacement donné.
- Accompagner les utilisateurs pendant leurs déplacements.
- Fournir une information fiable et actualisée
- Être simple d'utilisation afin de pouvoir être consulté par tous les types d'usagers possibles (enfants, personnes âgées, personnes avec un ou plusieurs handicaps, etc.).

Dans ce cadre, la personnalisation des informations fournies représente un élément important qui peut influer sur le choix des utilisateurs d'emprunter ou pas les transports collectifs (Guo et Blythe, 2005).

L'ensemble des besoins et des contenus ayant été définis, nous allons maintenant faire un rapide tour d'horizon des différents systèmes et outils d'information voyageurs personnalisés existant aujourd'hui.

2.5.2 Les outils et méthodes existants adaptés au domaine des transports

Aujourd'hui, de nombreux systèmes commerciaux existent sur le marché de l'information voyageur avec des fonctionnalités plus ou moins avancées. Les solutions les plus généralement rencontrées sont des solutions d'information simples, non personnalisées, ne prenant en compte qu'un seul mode de transport et/ou qu'une seule société de transport. Ainsi, le site Star (http://www.star.fr) permet d'organiser ses déplacements en bus et en métro sur la ville de Rennes. Le site Transville (http://www.transvilles.com) permet, pour un voyageur, d'organiser ses déplacements dans la ville de Valenciennes et dans quelques villes environnantes. D'autres solutions proposent pour leur part quelques éléments de personnalisation. Ainsi, le site de la RATP (http://www.ratp.fr), qui regroupe la majorité des transports collectifs en Ile de France, permet, à travers l'outil "Ma RATP" de définir des raccourcis vers les lignes empruntées le plus souvent. Le site de la SNCF (http://www.voyages-sncf.com), qui recouvre la presque totalité des transports ferroviaires en France, permet pour sa part de pré-enregistrer ses déplacements préférés ainsi que quelques autres préférences comme, par exemple, le type de place et ou le type de train afin de faciliter les achats de titres de transport par l'usager. En matière d'accompagnement des usagers pendant leurs déplacements, la totalité de ces solutions ne dispose généralement que d'un nombre limité de services. Ainsi, le site de la RATP, propose de recevoir des informations sur les perturbations sur son téléphone portable via SMS. D'autres applications, comme celle de la société de transport de Washington (http://www.wmdata.com) proposent, pour leur part, à l'utilisateur de recevoir, sur son téléphone portable, des alertes sur des lignes précises. Elles permettent aussi de consulter, en temps réel, sur un ordinateur portable, les prochains départs à partir d'une station bien déterminée. Pour une vision un peu plus large des différents systèmes de transports existant aujourd'hui en Europe, on pourra se reporter au rapport de (Rupert *et al.*, 2003). De manière synthétique, aujourd'hui, dans le domaine de l'information transport, les solutions proposées présentent trois limitations importantes :

- Des possibilités très restreintes en matière de personnalisation de l'information.
- Un accès réduit aux informations pendant les déplacements.
- Une prise en compte limitée de l'intermodalité entre transporteurs et/ou entre les différents moyens de transport (Uster, 2002). Il est en effet très rare que des systèmes multimodaux soient déployés. Pour donner un exemple de ce type de système, on pourra citer le système développé sur le territoire de la Communauté Urbaine de Lille (http://www.transpole.fr).

Dans le cadre de cette thèse, nous ne prendrons pas en compte tous les aspects liés aux problématiques d'intermodalité qui ont déjà été abordés dans les travaux de (Gendre, 1999), (Mouloudi, 2007) et de (Petit-Rozé, 2003). Aussi, dans la suite du document, nous nous concentrerons sur les solutions de personnalisation dans le domaine de l'information transport.

Depuis de nombreuses années, différentes solutions ont été étudiées dans le cadre de travaux de recherche pour résoudre la problématique de la personnalisation des informations dans les transports. Ainsi, on pourra citer :

- Agenperso (AGENts logiciels PERSOnnels) de (Petit-Rozé *et al.*, 2003) dont l'objectif est de fournir une information personnalisée dans la préparation d'itinéraires utilisant plusieurs modes de transports.
- Persyst (personalization System) de (Anli, 2006), qui est bâti sur Agenperso, et dont l'objectif est de fournir un système de personnalisation évolutif permettant d'intégrer facilement de nouveaux besoins en matière de personnalisation.
- Piepser (Personalized information on disruption to public transport exclusive to user of public transport) de (Hoyer et Czogolla, 2002) dont l'objectif est de prévenir les usagers des perturbations susceptibles d'affecter le déplacement qu'ils sont en train d'effectuer. Le message d'information est envoyé via un SMS et des solutions alternatives sont proposées dans le cas où l'usager est susceptible de rater sa correspondance.
- Tisoni (Traveller information system on internet) de (Lam et Xie, 2002) dont l'objectif est de fournir des informations personnalisées sur les déplacements en fonction des préférences et des habitudes de déplacement de l'usager.

Si, comme nous venons de le voir, il existe de nombreux outils pour personnaliser l'information transport, peu de solutions de modélisation abordent la problématique de la modélisation de cette personnalisation et aucune ne traite du domaine des transports. Ainsi (Abrahão *et al.*, 2002) proposent d'utiliser, au niveau des modèles conceptuels, des patrons (patterns) de personnalisation. De cette manière, il devient possible d'intégrer la personnalisation dans le modèle de navigation. Pour (Ahmad *et al.*, 2007), dans le cadre d'un système de formation, l'utilisation d'un modèle du domaine de compétence métier de référence permet, à travers l'analyse des "erreurs" de recherche des utilisateurs, d'offrir une personnalisation adaptée aux réels besoins de l'utilisateur. (Bonnet, 2003) propose pour sa part une méthode de modélisation centrée sur l'utilisateur permettant de fournir des applications personnalisées ; la personnalisation des contenus n'étant pas prise en compte dans cette approche. Ces trois exemples permettent de voir qu'il n'existe pas aujourd'hui de consensus sur la façon dont il est possible d'intégrer la personnalisation dans les modèles conceptuels d'une application.

2.5.3 Synthèse et discussion

Si, dans le domaine de l'information transport, de nombreux besoins en matière de personnalisation de l'information ont été identifiés, les solutions proposées par les exploitants, ainsi que les différents travaux de recherche sur le sujet, ne prennent souvent en compte qu'un nombre limité de possibilités. Ce qui est sûr, c'est que les besoins en matière de personnalisation risquent d'évoluer à l'avenir pour répondre à des besoins qui ne sont pas encore connus et/ou identifiés. Aussi, il est indispensable de créer des systèmes de personnalisation performants et évolutifs capable d'intégrer rapidement et simplement de nouvelles fonctionnalités ; l'idéal étant que cette intégration ait un impact minimum sur les applications existantes.

Au niveau de la modélisation conceptuelle des applications, ceci implique d'être capable de prendre en compte toutes les possibilités de personnalisation utilisables dans le domaine des transports. Mais, cela implique aussi d'avoir une modélisation conceptuelle de la personnalisation qui puisse facilement prendre en compte de nouveau besoins. L'autre problématique à prendre en compte concerne la possibilité de permettre, à la personne en charge de la modélisation, de pouvoir associer librement les différentes fonctionnalités de personnalisation entre elles. Ainsi, il devrait être possible de demander une personnalisation uniquement en fonction de la localisation géographique pour une partie donnée d'une application et de demander une personnalisation en fonction de la localisation et des préférences de l'utilisateur pour une autre partie de l'application. Comment ceci peut-il se traduire au niveau conceptuel ? En fait, il existe deux possibilités :

- Ajouter des étapes de personnalisations dans chaque processus métier pour chaque type de personnalisation donnée (personnalisation en fonction de la localisation géographique, personnalisation en fonction des préférences de l'utilisateur, etc.).
- Utiliser un moteur conceptuel de personnalisation qui serait utilisable au sein des processus métier à travers l'activation ou non de propriétés ; chaque propriété correspondant à un type de personnalisation donnée.

Si la première solution semble être intéressante, elle présente l'inconvénient de ne pas prendre en compte les interactions possibles entre les différents types de personnalisation ; chaque type étant vu comme une personnalisation particulière. La deuxième solution, par contre, permet de prendre en compte cette problématique. De plus, à travers l'utilisation de propriétés, elle offre aussi plus de souplesse en permettant l'ajout d'un nouveau type de personnalisation par simple ajout d'une nouvelle propriété. L'ajout d'une propriété ne devant pas avoir d'impact sur les applications existantes, on devra prévoir d'associer à chaque propriété une valeur par défaut qui sera automatiquement prise en compte par toutes les applications.

Conclusion

À travers ce chapitre, nous avons fait une présentation générale de problématiques représentatives liées à la personnalisation et plus particulièrement à la personnalisation des contenus. Nous avons aussi fait un parallèle avec l'adaptation contextuelle des applications en montrant que la personnalisation pouvait être vue comme une adaptation particulière au contexte d'usage où l'utilisateur représente un élément principal de cette adaptation. Mais la personnalisation des contenus dépend aussi d'autres critères comme le contexte d'usage (plateforme technique, environnement physique, localisation, etc.), la nature des informations manipulées ou encore les algorithmes et les méthodes utilisés au niveau de la personnalisation. Et c'est l'ensemble de ces éléments qu'il faut être capable de prendre en compte dans le cadre d'une approche de modélisation des applications de type IDM.

Un des premiers points à prendre en compte est que la modélisation de l'utilisateur est une modélisation multifacettes. Aussi, elle ne peut être représentée à travers un seul et unique modèle. En fait, elle doit s'appuyer sur un ensemble de modèles spécifiques à différentes problématiques et capables d'interagir entres eux. Aujourd'hui, un des éléments permettant d'apporter de la souplesse au niveau de la modélisation des utilisateurs réside dans l'utilisation d'une ontologie. Mais alors se pose la question de la définition de cette ontologie et surtout des interactions à établir avec les autres modèles.

Pour ce qui est du contexte, la problématique principale qu'on peut rencontrer dans le cadre d'une modélisation conceptuelle réside dans le fait que toutes les adaptations ne peuvent être modélisées au niveau conceptuel. La question se pose alors de savoir comment intégrer ces adaptations qui ne peuvent être modélisées conceptuellement dans le cadre d'une architecture MDA. À côté de cela, se pose aussi la question de savoir comment modéliser le contexte au niveau conceptuel afin qu'il puisse être manipulé dans les différents modèles. Là aussi, l'utilisation d'une ontologie semble être une bonne solution même si elle entraîne un certain nombre de questions.

Au niveau des contenus, la complexité principale réside dans le fait que ceux-ci peuvent être de natures très différentes : services et données. Si pour les services, la personnalisation peut être, *a priori*, effectuée dans le cadre des processus métier associés à l'application, la prise en compte des données est un peu plus complexe. Ainsi, il faut être capable de prendre en compte tous les types de données (structurées, semi structurées et non structurées) ; ce qui est possible en utilisant des transformations en données structurées. Il faut ensuite être capable de manipuler ces données conceptuellement afin de pouvoir s'affranchir totalement de tout lien avec les aspects techniques. Dans ce cadre, l'utilisation d'une ontologie semble être un bon candidat pour permettre cette indépendance. Mais alors se pose la question du passage du niveau conceptuel au niveau technique pour laquelle on ne dispose pas aujourd'hui de réponse satisfaisante.

Dans le domaine des transports, un certain nombre de spécificités ont fait qu'il est bien souvent difficile d'utiliser des systèmes génériques de personnalisation sans devoir les adapter. Cela a entraîné le développement d'un certain nombre d'outils et de méthodes, plus ou moins compatibles et complémentaires, qui restent bien souvent très liés à la technique. Pour s'affranchir de toutes ces

contraintes et limitations techniques au niveau de la modélisation conceptuelle, une des solutions les plus prometteuses serait de pouvoir activer la personnalisation par simple sélection de propriétés. Ainsi, chaque propriété correspondrait à un type de personnalisation et l'utilisation de plusieurs propriétés permettrait de créer des personnalisations composites multi dimensionnelles.

En résumé, si on dispose aujourd'hui de nombreux outils et méthodes de personnalisation, on ne possède pas encore de méthode permettant de prendre en compte celle-ci dans le cadre d'une modélisation conceptuelle d'une application. Si, comme nous l'avons vu, des solutions existent, elles font appel à des modèles différents. Aussi, est-il indispensable de bien les définir aussi bien au niveau de leurs contenus qu'au niveau des interactions possibles avec les autres modèles. C'est ce que nous proposons, à un certain niveau, dans le cadre de notre méthode PERCOMOM que nous allons maintenant présenter.

Chapitre 3
PERCOMOM, une méthode de modélisation des applications interactives utilisant les processus métier

Sommaire

INTRODUCTION **65**

3.1 L'APPROCHE DE MODELISATION AU NIVEAU CONCEPTUEL **66**
3.1.1 INTRODUCTION GENERALE SUR L'APPROCHE GLOBALE DE PERCOMOM 66
3.1.2 LA MODELISATION CONCEPTUELLE DANS PERCOMOM (NIVEAU CIM) 67
3.1.3 LES MODELES CIM, DITS MODELES DE SUPPORT, DANS LE CADRE DE LA MODELISATION CONCEPTUELLE DES INTERACTIONS. 73
3.1.4 LA MODELISATION DES INTERACTIONS AU NIVEAU CONCEPTUEL (NIVEAU CIM) 82
3.1.5 SYNTHESE ET CONCLUSION 95

3.2 L'ARCHITECTURE DE TYPE MDA UTILISEE PAR PERCOMOM (NIVEAUX CIM, PIM ET PSM) **96**
3.2.1 INTRODUCTION 96
3.2.2 LE NIVEAU PIM : UN NIVEAU COMPOSE DE SERVICES FONCTIONNELS 97
3.2.3 LE NIVEAU PSM : UNE ARCHITECTURE POUR UNE GENERATION SEMI-AUTOMATIQUE DES APPLICATIONS 103
3.2.4 SYNTHESE ET DISCUSSION 105

3.3 LES PRINCIPES DE TRANSFORMATION DE MODELES POUR PASSER DU NIVEAU CIM A L'APPLICATION CONCRETE DE NIVEAU PSM **106**
3.3.1 INTRODUCTION 106
3.3.2 LA TRANSFORMATION DE MODELES DANS PERCOMOM 107
3.3.3 ANALYSE ET DISCUSSION 110

CONCLUSION **111**

Introduction

L'approche dirigée par les modèles sera très certainement conduite à devenir un nouveau paradigme en matière de développement d'applications dans les prochaines années (cf. chapitre 1). S'il existe aujourd'hui de nombreux outils et méthodes qui permettent d'aborder ce type d'approche, il y en a peu qui soient capables de la prendre en charge complètement. Et, lorsqu'ils le font, c'est souvent pour un domaine métier bien précis et uniquement pour des types d'applications bien déterminés. C'est en partant de ce constat, que nous avons développé une méthode de modélisation et une architecture générique capable de modéliser et de générer des applications interactives de type WIMP (Window, Icon, Mouse, Pointing device). La méthode s'appelle PERCOMOM (PERsonalization and COnceptual MOdeling Method). Dans ce chapitre, nous allons présenter l'ensemble des différents éléments de notre approche, du point de vue de la modélisation et du point de vue des IHM, en précisant à chaque fois les avantages et les inconvénients. L'ensemble des aspects concernant la prise en charge de la personnalisation dans le cadre de la modélisation des applications interactives seront, pour leur part, présentés dans le chapitre 4.

Dans la première partie, nous définirons l'approche dans sa globalité afin de préciser ce qu'est et ce que n'est pas PERCOMOM. Puis, nous inscrirons nos travaux dans le cadre d'une vision à plus long terme définie par l'OMG. Ensuite, en nous basant sur cette vision, nous présenterons une première ébauche de méthode de modélisation des applications au niveau conceptuel. Ainsi, nous verrons comment, de manière à permettre la création de modèles conceptuels par les experts métier, nous avons défini une vision orientée métier de la modélisation de niveau CIM. Celle-ci nous a amené à créer un ensemble de modèles conceptuels que nous avons répartis dans quatre grandes familles. Nos travaux étant principalement orientés vers la modélisation conceptuelle des IHM, nous présenterons ensuite, très rapidement, les différentes familles de modèles et les différents modèles associés qui servent tous de support aux modèles d'IHM avant de nous concentrer sur les modèles conceptuels d'IHM proprement dit. Ces modèles, au nombre de trois, permettent de gérer à la fois les aspects dynamiques (l'enchaînement des interactions) mais aussi les aspects statiques (le regroupement des éléments d'interaction en ensembles logiques) associés aux IHM. Nous terminerons alors cette partie par une synthèse résumant l'ensemble de l'approche PERCOMOM du point de vue de la modélisation conceptuelle des applications interactives.

Dans la deuxième partie, nous présenterons l'architecture de type MDA sur laquelle s'appuie notre méthode pour permettre la génération semi-automatique d'applications interactives. Nous verrons que celle-ci repose sur un ensemble de frameworks qui permettent de simplifier la modélisation de niveau CIM mais aussi d'apporter plus de souplesse dans l'adaptation et la prise en charge des applications. En premier lieu, nous présenterons le framework de niveau PIM et la notion de services fonctionnels et techniques. Nous verrons en quoi ceux-ci permettent de simplifier la modélisation de niveau CIM tout en apportant plus de fiabilité et de qualité aux applications réalisées. Puis, nous présenterons les frameworks de niveau PSM et leur apport au niveau de la prise en charge de multiples plateformes utilisateur. Enfin, nous terminerons par une synthèse présentant les avantages et les inconvénients de l'utilisation de la notion de frameworks dans une approche de type IDM.

Dans la troisième partie, nous présenterons comment, dans PERCOMOM, le passage des modèles conceptuels de l'application de niveau CIM à l'application concrète de niveau PSM est effectué avec, comme objectif, de minimiser les interventions humaines à effectuer pour générer des applications. Pour cela, nous présenterons les principes généraux de la transformation de modèle qui s'appuient principalement sur l'utilisation des frameworks définis au niveau PIM et au niveu PSM. Puis nous verrons comment ces principes sont appliqués dans PERCOMOM à travers l'utilisation des notions de tissage et de séparation de modèles lors des passages du niveau CIM au niveau PIM et lors des passages du niveau PIM au niveau PSM. Enfin nous terminerons par une synthèse présentant les avantages et les inconvénients des principes de transformation de modèle telle qu'elle est proposée dans PERCOMOM.

Mais avant toute chose, nous allons commencer par présenter ce qu'est PERCOMOM.

3.1 L'approche de modélisation au niveau conceptuel

3.1.1 Introduction générale sur l'approche globale de PERCOMOM

Nos travaux s'inscrivent dans le cadre de la réalisation d'une méthode de modélisation et d'une architecture générique permettant de générer des applications informatiques personnalisées interactives, dans le domaine de la diffusion d'informations et de services, à partir de leur modélisation conceptuelle et ceci de manière semi-automatique ; c'est-à-dire en limitant au maximum les interventions humaines. Ceci explique le nom de PERCOMOM (PERsonalization and COnceptual MOdeling Method) qui reprend les deux axes principaux de nos travaux : la modélisation conceptuelle des applications interactives, qui sera traitée dans ce chapitre, et la prise en compte de la personnalisation dans la modélisation, qui sera traitée dans le chapitre 4.

Il est important de noter ici que, pour l'instant, notre méthode n'est prévue que pour prendre en charge les interfaces interactives de type WIMP (Window, Icon, Menu, Pointing device).

Afin d'offrir une solution utilisable avec les outils et méthodologies existants, nous avons défini une méthode globale basée sur une approche MDA (cf. chapitre 1, §1.1.2). À terme, notre objectif est de proposer une véritable "fabrique" d'applications en nous basant sur une approche de type IDM (Ingénierie Dirigée par les Modèles) qui permette de créer des applications personnalisées à partir d'un assemblage de modèles existants.

Dans le cadre de nos travaux, le terme de méthode est utilisé dans l'optique d'une industrialisation des développements informatiques. Dans ce sens, PERCOMOM correspond plus à un procédé de fabrication qu'à une approche théorique de modélisation des applications informatiques. Aussi, est-il important de définir ce qu'est la méthode PERCOMOM :

- Un ensemble de modèles conceptuels permettant de modéliser complètement une application informatique.
- Une architecture technique permettant de passer de manière semi-automatique des modèles conceptuels aux applications finales.
- Un ensemble de services et fonctionnalités transverses permettant de faciliter le processus de création des applications.

Ce qu'elle n'est pas dans le cadre actuel d'avancement de nos travaux :

- Elle ne prend pas en charge les phases de définition et d'analyse des besoins.
- Elle ne prend en charge que de manière indirecte les problématiques associées aux architectures techniques sur lesquelles les applications vont être exécutées.
- Elle ne prend pas en charge les étapes de validation des applications aussi bien au niveau technique qu'au niveau fonctionnel.

La figure Figure 3.1 montre les étapes prises en compte dans PERCOMOM dans le cadre d'une approche classique de développement suivant un cycle en V.

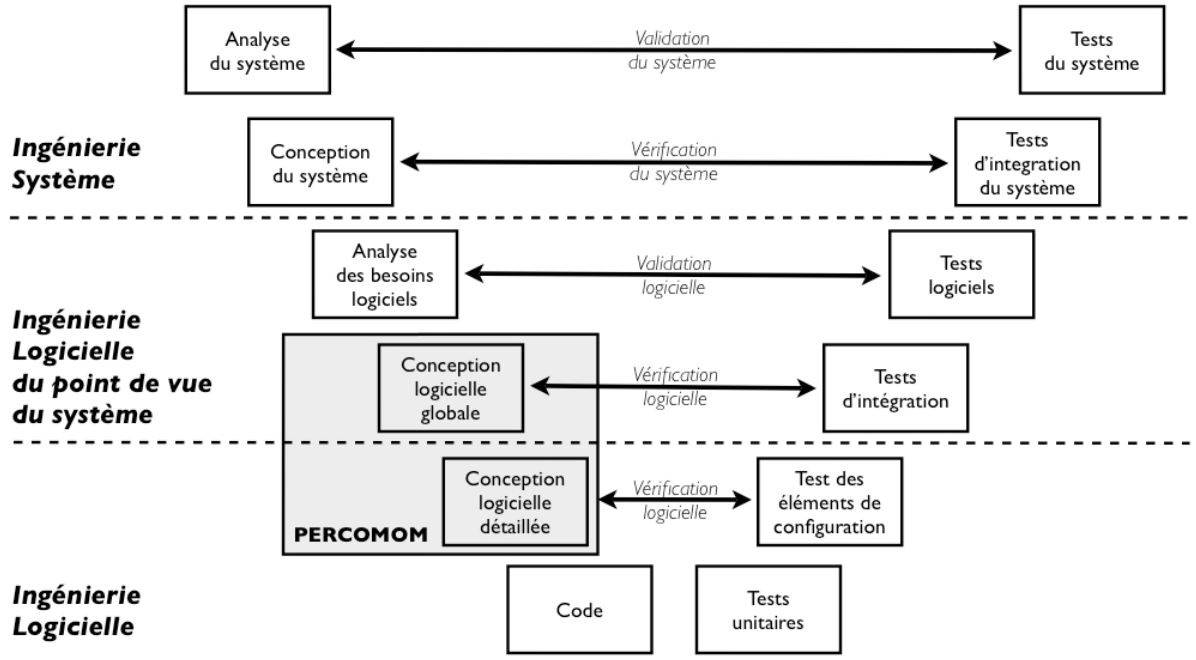


Figure 3.1. Etapes prise en compte par PERCOMOM dans le cadre d'une approche classique de développement suivant un cycle en V

Dans l'état actuel de nos travaux, nous avons principalement porté l'accent sur la modélisation des interactions homme-machine au niveau conceptuel (cf. chapitre 3, §3.1), sur la transformation automatique de modèles conceptuels en applications concrètes (cf. chapitre 3, §3.2) et sur la prise en compte de la notion de personnalisation des contenus dans les modèles conceptuels (cf. chapitre 4).

PERCOMOM s'appuyant sur une architecture de type MDA, nous allons montrer dans la suite de ce chapitre comment PERCOMOM utilise cette architecture à trois niveaux (CIM, PIM et PSM).

3.1.2 La modélisation conceptuelle dans PERCOMOM (niveau CIM)

3.1.2.1 La vision à moyen terme de la modélisation de l'OMG

Comme présenté dans le chapitre 1, il est aujourd'hui admis, dans le milieu de la modélisation, qu'il est impossible de représenter la globalité d'une application à travers l'utilisation d'un seul type de modèle même à travers une notation comme UML. Mais, ceci introduit de nouvelles problématiques au niveau de la modélisation :

- Quels sont les modèles à définir pour prendre en charge complètement une application ?
- Comment être sûr que les modèles soient bien complémentaires et ne se recouvrent pas au niveau de leurs contenus (impossibilité de définir deux fois la même chose dans deux types de modèles différents) ?

Pour répondre en partie à ces questions, l'OMG (Watson, 2005), qui est à l'initiative de la modélisation UML, a présenté sa vision de l'évolution de la modélisation globale des applications, au niveau conceptuel (CIM), à moyen terme (cf. Figure 3.2).

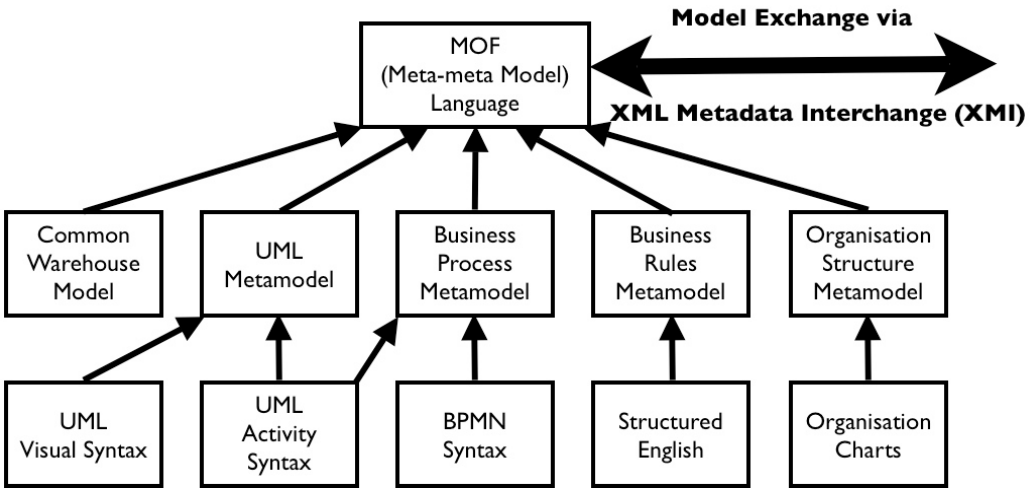


Figure 3.2. La vision de l'évolution de la modélisation suivant l'OMG (Watson, 2005)

Dans cette vision de l'OMG, les principaux éléments marquants sont les suivants :

- UML, même avec l'utilisation de profils, n'est pas suffisant pour permettre de décrire complètement une application.
- Le développement des architectures de type MDA et des approches de type IDM impose l'utilisation de modèles orientés vers les experts métier comme les modèles métier et les règles métier.
- Au niveau conceptuel, il est indispensable de prendre en charge les structures sociales dans lesquelles les utilisateurs évoluent. Pour cela, il convient de développer de nouveaux modèles (les modèles de structure d'organisation par exemple).
- Le métamétamodèle MOF (MetaObject Facility) est l'élément qui devrait permettre de relier l'ensemble des modèles entre eux, facilitant ainsi leurs interactions.
- Pour permettre la définition et l'utilisation standard de données entre tous les modèles, il est indispensable d'utiliser une base commune. Pour cela, l'OMG préconise d'utiliser le Common Warehouse Metamodel qui est une spécification décrivant les méta-données pouvant être partagées par des systèmes de Data Warehouse (entrepôts de données), de Business Intelligence (applications décisionnelles) et de gestion de connaissances.

Si cette vision semble être prometteuse, elle est aujourd'hui très en avance sur les outils et méthodes existants. Ainsi, la modélisation des règles métier repose aujourd'hui sur le langage *Semantics of Business Vocabulary and Business Rules* ou SBVR (OMG, 2005). Or ce langage ne permet, aujourd'hui, d'exprimer les termes et les règles métier qu'en langage courant difficilement utilisable dans une application informatique. Au niveau de la modélisation des processus métier, le langage proposé, *Business Process Modeling Notation* ou BPMN (BPMI, 2004), n'a pas de métamodèle totalement compatible avec MOF. Quant à la modélisation des organisations sociales, aucune modèle et aucun langage n'a pour l'instant été normalisé pour prendre en compte ce type de problématique. En conséquence, l'approche proposée par l'OMG n'a pas encore pu être utilisée dans le cadre du développement d'une application informatique. Aussi, il est difficile aujourd'hui de pouvoir en donner les avantages, les limites et les contraintes par rapport aux méthodes existantes.

3.1.2.2 Une première ébauche de méthode de modélisation orientée métier dans PERCOMOM

En partant de ce constat et en nous basant sur les travaux de (Watson, 2005) présentés précédemment, nous avons conçu une approche pour la modélisation conceptuelle des applications interactives qui repose sur les hypothèses suivantes :

- La modélisation conceptuelle de niveau CIM d'une application doit être totalement indépendante des technologies qui seront utilisées lors de sa réalisation technique.
- Les experts métier sont les personnes les plus à même pour définir et modéliser les modèles de niveau CIM car ce sont eux qui connaissent le mieux les besoins fonctionnels.
- Les experts métier n'étant pas, le plus souvent, des informaticiens, il faut leur fournir des modèles qu'ils puissent comprendre et manipuler et donc des modèles adaptés à leur métier à travers, principalement, des formalismes adaptés et des modèles ne prenant en charge que des parties bien spécifiques d'une application.

En partant de ces différents principes, nous avons défini une approche orientée vers les aspects fonctionnels de l'application, c'est-à-dire orientée métier, des applications interactives au niveau de la modélisation conceptuelle (cf. Figure 3.3).

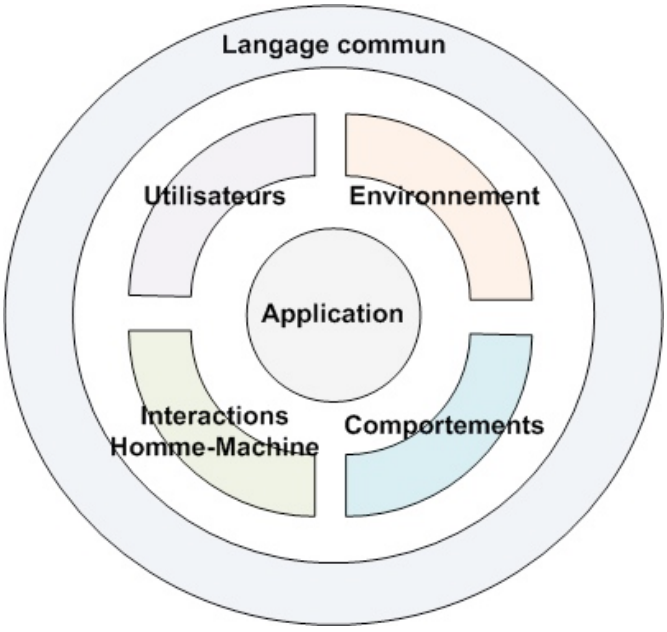


Figure 3.3. La vision globale, au niveau CIM, d'une application dans PERCOMOM

Ainsi, nous avons considéré, après étude de différentes applications existantes, qu'une application pouvait se modéliser, au niveau conceptuel, à travers quatre grands ensembles de modèles :

- Un ensemble de modèles permettant de représenter les utilisateurs. Celui-ci permet de caractériser l'utilisateur mais aussi de définir ses relations avec les autres utilisateurs ainsi que ses relations, en tant qu'individu, avec les applications. Ainsi, un utilisateur est défini à travers des caractéristiques qui lui sont propres comme, par exemple, son âge, sa taille, ses préférences ou encore son vécu. Mais, il est aussi défini comme faisant partie d'un ou de plusieurs groupes sociaux ayant un certain nombre de caractéristiques en commun. Et, par rapport à une application, il est défini comme ayant un rôle vis-à-vis de l'application qui se traduira, en pratique, par des niveaux d'accès aux différentes fonctionnalités de l'application.
- Un ensemble de modèles permettant de représenter l'environnement, c'est-à-dire l'ensemble des éléments extérieurs à l'application, du point de vue métier, pouvant éventuellement avoir une interaction avec celle-ci. Ainsi, si connaître le lieu géographique où se trouve l'utilisateur est important pour effectuer certains traitements, la définition de ce lieu

géographique fait partie de la définition de l'environnement dans lequel va être utilisée l'application ; cet environnement pouvant être commun à plusieurs applications. De même, les systèmes externes comme, par exemple, une imprimante, font partie de l'environnement de l'application, mais ne sont pas directement associés à celle-ci ; le changement physique de l'imprimante ne devant, en principe, pas impliquer de changement au niveau de l'application.

- Un ensemble de modèles permettant de représenter des comportements, c'est-à-dire l'ensemble des éléments, directement associés à l'application, capables de réagir à un stimulus, celui-ci pouvant consister en un simple appel à cet élément, et permettant de réaliser certaines tâches, de manière automatique, sans aucune interaction avec l'utilisateur.
- Un ensemble de modèles permettant de représenter les interactions homme-machine, c'est-à-dire l'ensemble des éléments permettant de caractériser toutes les interactions possibles entre les utilisateurs et l'application. Cette caractérisation se fait aussi bien au niveau statique à travers des agencements des différents éléments d'interaction pour former des ensembles cohérents d'interaction (écran de navigation ou de saisie par exemple) qu'au niveau dynamique pour préciser les interactions possibles entre les utilisateurs et les différents éléments statiques d'interaction.

Dans cette approche orientée métier, chaque modèle est prévu pour être pris en charge par un ou plusieurs experts métier en fonction du domaine d'expertise de chacun ; l'objectif étant que chaque modèle soit pris en charge par l'expert métier le plus qualifié pour le faire. Il est important de noter ici que le choix des différentes catégories de modèles a été fait de manière à :

- Limiter le nombre de catégories,
- Permettre une segmentation claire du rôle de chaque catégorie,
- Permettre de prendre en compte la majorité des problématiques liées à la modélisation conceptuelle d'une application interactive,
- Faciliter la répartition des modèles entre les différents experts métier en charge de la modélisation (spécialisation par domaine métier).

Pour identifier ces différentes catégories de modèles et les modèles associés, nous avons procédé suivant un processus d'analyse sur un ensemble d'applications existantes dans le domaine du transport (cf. Figure 3.4) utilisant différents types d'interfaces homme-machine et différents types de personnalisation. Les cinq applications principales analysées ont été :

- Le site web "Mon-Service-Transport.com "développé au LAMIH dans le cadre de la thèse de (Anli, 2006).
- Les applications fonctionnant sur les bornes automatiques SNCF situées dans les gares.
- Le site internet public de la SNCF (http://www.voyages-sncf.com)
- Le site internet public de la RATP (http://www.ratp.fr) et son service sur téléphone portable "Ma RATP dans la poche"
- Le site internet public de la société Transpole (http://www.transpole.fr)

Le choix de ces différentes applications a été effectué soit en fonction de leur représentativité nationale ou locale du point de vue du domaine transport soit en fonction des possibilités de personnalisation qu'elles offrent. Aussi, n'a-t-il pas la prétention d'être exhaustif par rapport à toutes les possibilités techniques et fonctionnelles qu'il est possible de rencontrer dans le domaine de l'information voyageur dans les transports.

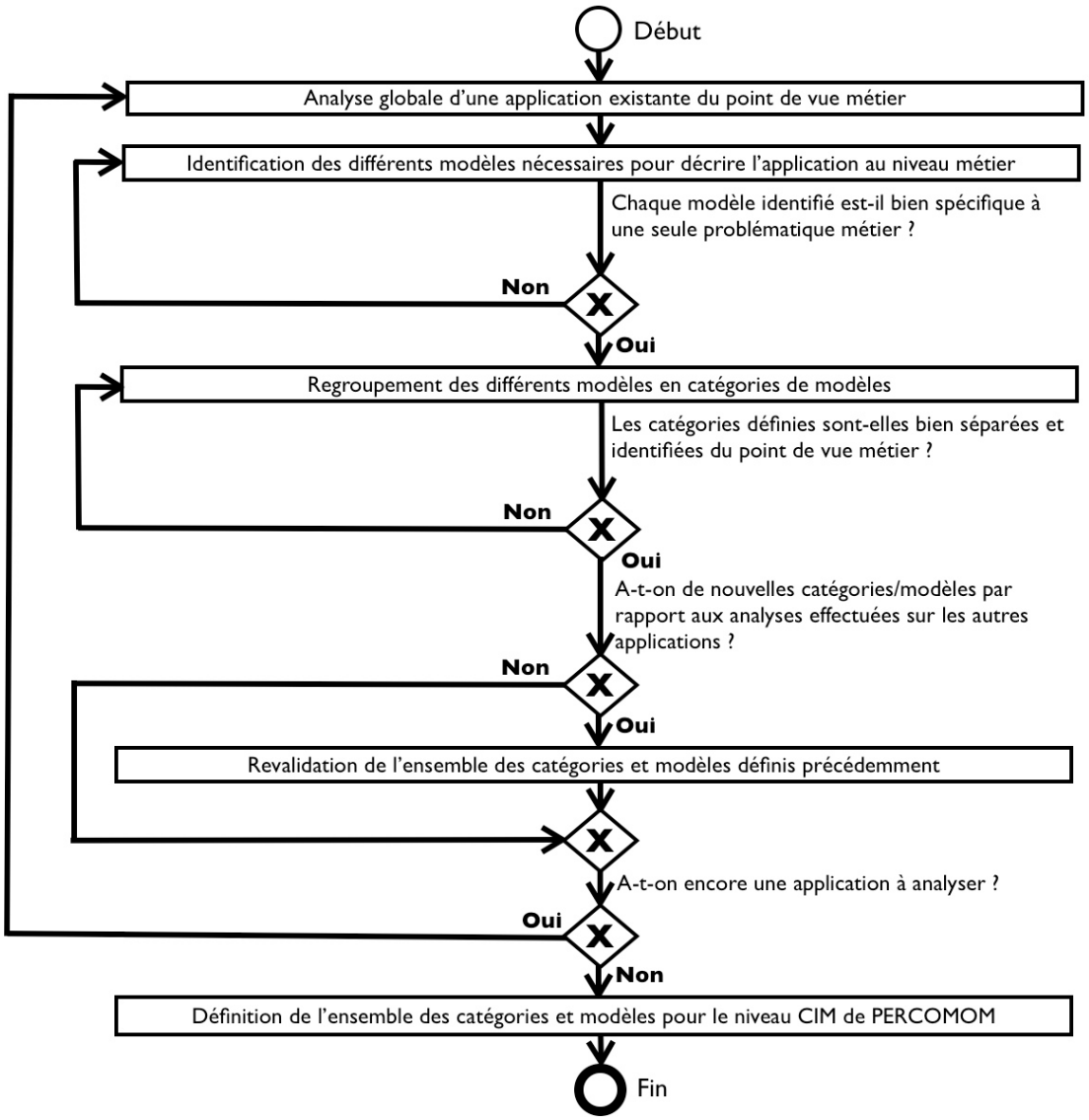


Figure 3.4. Processus d'analyse utilisé dans le cadre de PERCOMOM pour identifier les différentes catégories de modèles et les différents modèles au niveau conceptuel

L'approche pouvant faire intervenir un très grand nombre de participants, il est nécessaire de définir un cadre commun de travail afin de faciliter les échanges entre les différents intervenants. Dans le cadre de PERCOMOM, ce cadre commun est défini à travers l'utilisation d'un langage commun, défini spécifiquement pour le domaine métier associé à l'application. Celui-ci a pour but de permettre de lever toute ambiguïté au niveau des différents termes et concepts métier manipulés dans le cadre de l'application. Ainsi, dans le domaine du transport, on pourrait définir un "arrêt de bus" comme étant un terme métier et une "ligne de bus" comme étant un concept métier (une ligne de bus étant composée de plusieurs "arrêts de bus"). Dans PERCOMOM, cette définition d'un langage commun passe par l'utilisation d'une ontologie de domaine qui est une notion qui a déjà été présentée, de manière globale, dans le chapitre 1 (cf. §1.4).

À partir de cette approche fonctionnelle globale, nous avons conçu une première ébauche de méthode de modélisation conceptuelle et de génération d'applications en dix grandes étapes qui sont présentées dans la Figure 3.5. Le processus de création des modèles conceptuels et de génération de l'application étant linéaire, chaque étape est à considérer comme étant critique pour le démarrage de l'étape suivante et pour la réussite de l'ensemble du processus de création d'une nouvelle application.

Pour la partie modélisation de la méthode (étapes 1 à 7), l'ensemble des étapes doit être coordonné par un chef de projet, orienté fonctionnel, ayant une vision globale des différentes problématiques fonctionnelles. Cette coordination permettra d'assurer une certaine cohérence au niveau de la

modélisation en limitant les risques de mauvaise répartition des modèles, d'oubli de modèles ou de recouvrement de modèles (deux modèles différents modélisant en partie la même problématique).

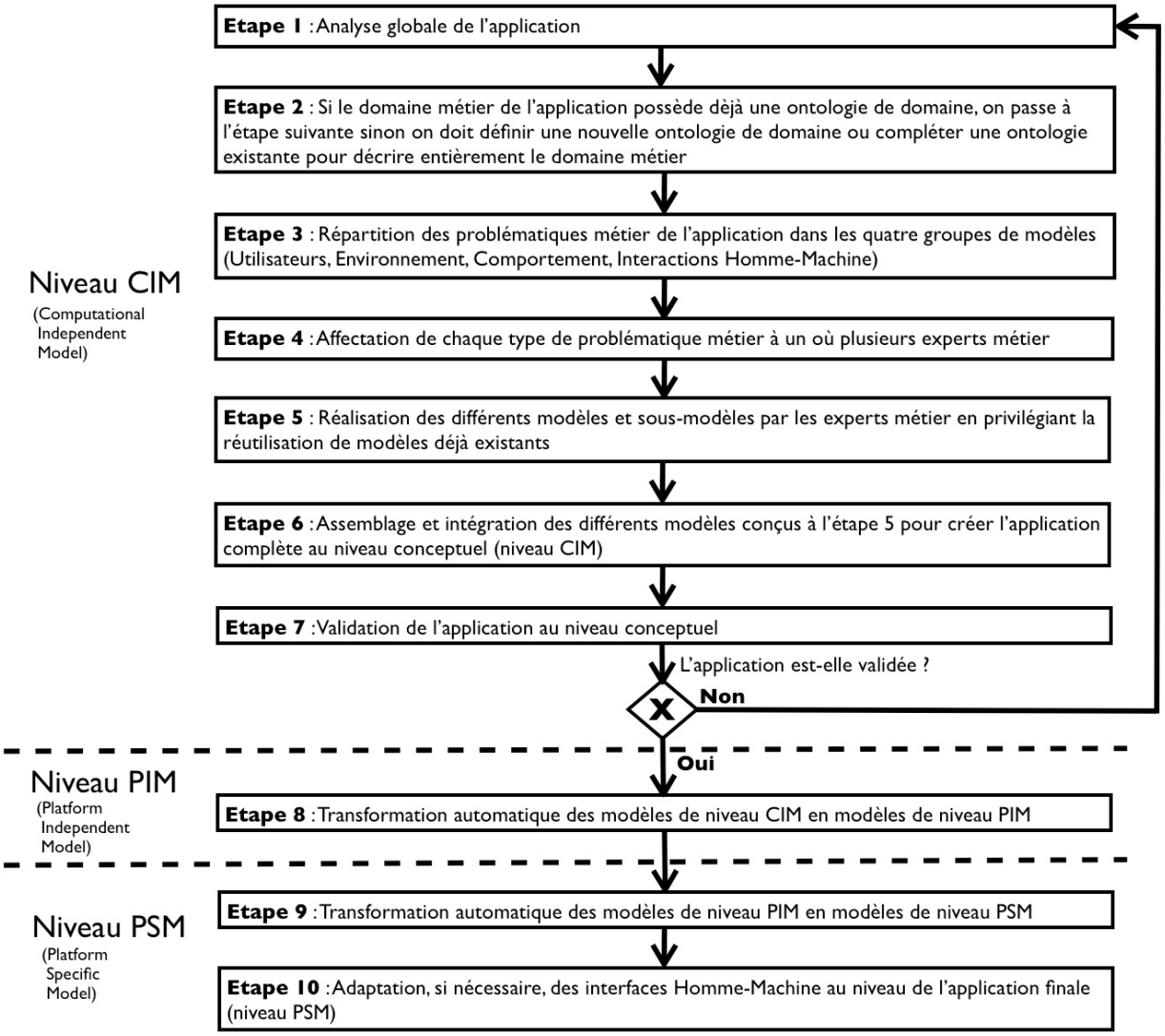


Figure 3.5. Les dix grandes étapes de la méthode PERCOMOM

Dans l'état actuel de nos travaux, la méthode de modélisation en dix étapes de PERCOMOM est dans un état de première ébauche ; toutes les étapes n'étant pour l'instant pas définies de manière formalisée (sauf les étapes 8 et 9 qui seront présentées dans le chapitre 3, §3.2). Néanmoins, elle représente la structure de la méthode cible sur laquelle repose l'ensemble du cadre méthodologique pour la modélisation conceptuelle, c'est-à-dire au niveau CIM, des applications interactives que nous allons maintenant présenter.

3.1.2.3 Le cadre méthodologique de PERCOMOM pour le niveau CIM

En partant de la vision globale d'une application, au niveau CIM, dans PERCOMOM, nous avons défini, pour chaque catégorie de modèles présentée sur la Figure 3.3 un ensemble de modèles permettant de prendre en charge les problématiques associées. Ces différents modèles, qui sont issus du processus d'analyse décrit dans la Figure 3.4, sont présentés dans la Figure 3.6 et sont décrits en détail dans le chapitre 1, §3.1.3 et §3.1.4.

La méthode ayant été développée dans le cadre du projet Viatic.Mobilité, son cadre d'application a volontairement été limité à ce projet afin de permettre une première utilisation et évaluation de la méthode.

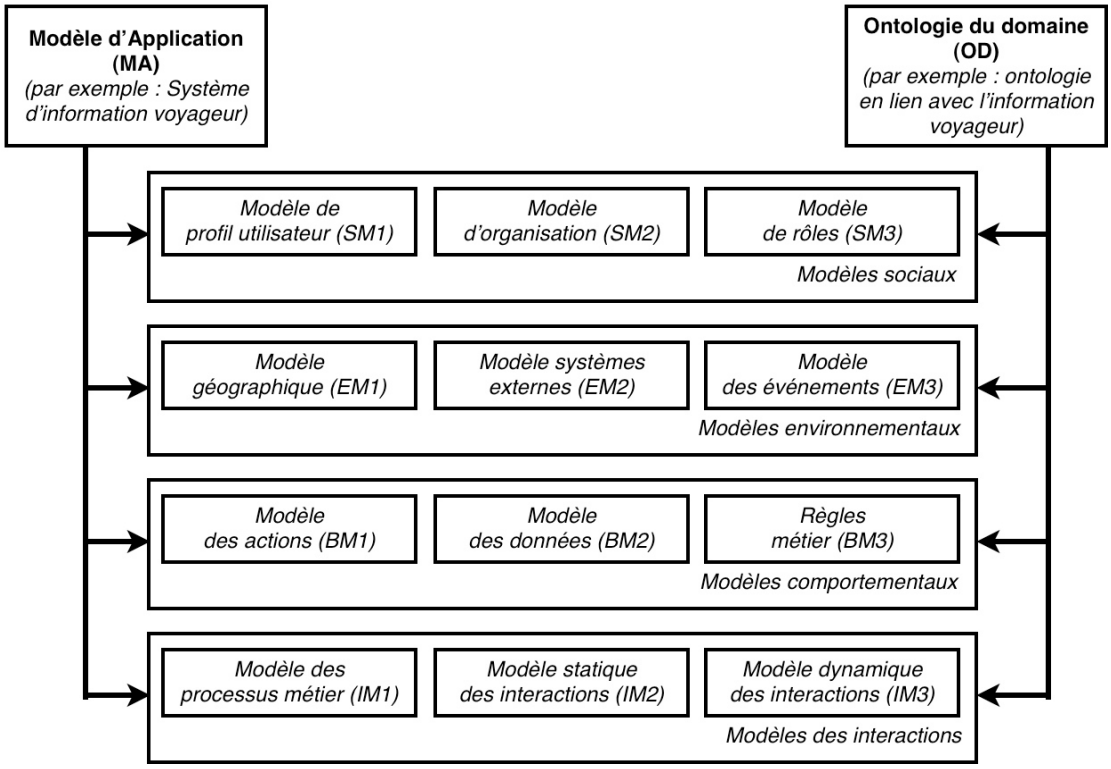


Figure 3.6. Schéma de l'ensemble des modèles conceptuels de niveau CIM dans PERCOMOM

Notre thèse étant principalement focalisée sur la modélisation des applications interactives, seuls ces modèles (IM1, IM2 et IM3) seront détaillés (cf. chapitre 3, §3.1.4). Les autres modèles, qui sont utilisés comme des modèles de support dans le cadre de la modélisation des interactions seront plus brièvement décrits (cf. chapitre 3, §3.1.3).

3.1.3Les modèles CIM, dits modèles de support, dans le cadre de la modélisation conceptuelle des interactions.

3.1.3.1 Le modèle d'application (MA)

Dans le cadre de notre approche, nous avons défini deux modèles globaux qui sont le modèle d'application (MA) et l'ontologie de domaine (OD). En fait, leur particularité réside dans le fait que ce sont des modèles transverses qui utilisent ou qui sont utilisés par l'ensemble des autres modèles.

Le modèle d'application, qui se présente sous la forme d'un profil, permet de définir des propriétés directement associées à l'application. Ainsi, on y trouve :

- La carte d'identité de l'application (identifiant, nom, catégorie d'application, etc.)
- Les propriétés générales sur l'application qui sont l'ensemble des propriétés qui ne sont pas directement reliées aux processus métier de l'application comme, par exemple, la langue utilisée par défaut dans l'application.
- Les propriétés applicatives qui sont directement reliées à l'application comme, par exemple, l'indication du processus métier à lancer lorsqu'on démarre l'application (modèle IM1), l'indication du processus métier à appeler par défaut en cas d'erreur dans l'application ou encore la liste les groupes d'accès (modèle SM3) pouvant accéder à l'application.

Dans PERCOMOM, nous avons choisi de représenter le modèle d'application sous la forme d'un fichier XML dans lequel se trouve l'ensemble des informations. À titre d'exemple, on trouvera, ci-après, un exemple de modèle d'application pour une application "InfoHoraireGare" qui fait partie de la

catégorie des applications "Information Horaire". Cette application utilise le français comme langue par défaut. Lorsqu'on la démarre, elle commence par lancer le processus métier "InfoHoraireGareDemarrage". Le processus métier qui sera appelé par défaut dans l'application sera "SelectionAffichageHoraireGare" et seuls les utilisateurs du groupe d'accès "ClientsSNCF" pourront accéder à l'application. Si l'utilisateur n'a pas le droit d'accéder à l'application, le processus métier "ErreurAccesInfoHoraire" sera exécuté.

```
<application>
   <!—Une application est caractérisée par un identifiant unique et par un nom unique -->
   <identite identifiant="12" nom="InfoHoraireGare">
      <categorie nom="Information Horaire"/>
   </identite>
   <proprietesgenerales>
      <languepardefaut code="FRE"/>
   </proprietesgenerales>
   <proprietesapplicatives>
      <businessprocessdemarrage nom="InfoHoraireGareDemarrage"/>
      <businessprocesspardefaut nom="SelectionAffichageHoraireGare"/>
      <controleacces erreur="ErreurAccesInfoHoraire">
         <groupeacces nom="ClientsSNCF"/>
      </controleacces>
   </proprietesapplicatives>
</application>
```

3.1.3.2 L'ontologie de domaine (OD)

À côté de ce modèle d'application, on trouve une ontologie de domaine métier (OD) qui permet de représenter l'ensemble des éléments métier utilisés dans le cadre de l'application ainsi que les différents liens sémantiques entre ces différents éléments. Pour la définition des différents termes utilisés pour manipuler des ontologies, on se reportera au chapitre 1, §1.4.1. Dans le cadre de la modélisation, l'utilisation d'une ontologie permet :

- de modéliser l'ensemble du vocabulaire et des concepts du domaine métier ce qui permet de mieux appréhender le domaine métier ;
- d'utiliser un socle de définition commun pour l'ensemble des applications qui seront développées dans le même domaine métier (toutes les applications utiliseront la même ontologie) ;
- de faciliter la communication entre les différents intervenants sur le projet à travers l'utilisation d'une terminologie commune ;
- de s'affranchir complètement des liens avec les sources de données (le lien entre l'ontologie et les sources de données étant réalisé au niveau du modèle de données BM2).

Pour donner un exemple d'utilisation de cette ontologie de domaine, la catégorie de l'application fait partie d'une classification des applications, définie au niveau de l'ontologie de domaine. Ainsi, il devient possible de regrouper les applications par catégorie mais aussi de définir et d'utiliser des liens sémantiques entre les différents types d'applications.

La Figure 3.7 présente un extrait d'une ontologie de domaine représentant l'ensemble des éléments géographiques utilisés dans le domaine sous forme de classe de l'ontologie. Dans cette ontologie géographique, on a défini que chaque individu de type "Zone_Géographique" et chaque individu héritant de ce concept possède un nom et une zone géographique associée de type polygone (notion issue des systèmes d'informations géographiques). Pour chaque classe héritant de "Zone_Géographique", on a ensuite défini des liens d'appartenance et d'exclusion de manière à avoir des relations du type : un "Continent" est composé de "Communauté_de_Pays" et de "Pays", un "Pays" appartient à un "Continent" ou à une "Communauté_de_Pays" qui appartient elle-même à un "Continent".

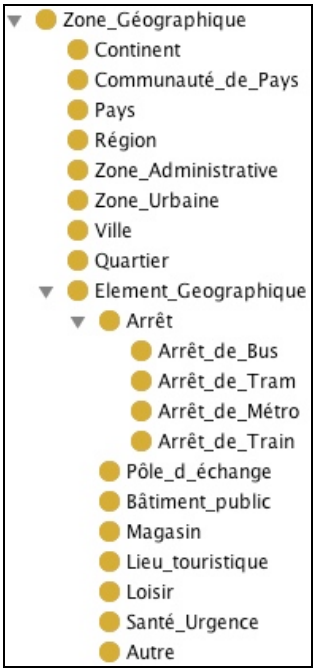


Figure 3.7. Exemple d'ontologie, dans PERCOMOM, associée aux espaces géographiques (représentation provenant de l'outil Protege présenté dans le chapitre 1 (cf. §1.4.2)

Dans l'ensemble des autres modèles, l'ontologie doit être obligatoirement utilisée lorsqu'on désire manipuler des informations du domaine comme, par exemple, un "Magasin" ou alors une "Ville". De cette manière, il devient possible d'avoir une abstraction totale des autres modèles par rapport aux données réelles qui seront manipulées dans l'application finale ; ce lien étant établi uniquement au niveau du modèle BM2 et ceci de manière indirecte (cf. chapitre 3, §3.1.3.5). Cette abstraction permet aussi d'utiliser les liens sémantiques entre les différentes classes de l'ontologie pour définir automatiquement des relations entre deux individus ou pour définir, à partir d'un individu d'une classe déterminée, quels sont les individus d'une autre classe qui lui sont associés.

Dans notre approche, l'ontologie de domaine est un élément très important dans le sens où elle est utilisée par tous les autres modèles dès que ceux-ci doivent manipuler des concepts du domaine métier. Ceci permet d'assurer une grande cohérence au niveau de l'ensemble des modèles mais aussi d'en faciliter la compréhension. L'ontologie n'étant pas au cœur de nos travaux de recherche, nous n'avons pas défini une nouvelle notation pour définir l'ontologie mais utilisé la notation standard OWL-DL (Ontology Web Language Description Logics). Dans la suite de ce chapitre, nous verrons comment cette ontologie est utilisée au sein des autres modèles et surtout ce qu'elle apporte à l'ensemble de ces modèles.

3.1.3.3 Les modèles sociaux (SM1, SM2 et SM3)

Les modèles sociaux (cf. Figure 3.6) servent à modéliser les utilisateurs à l'aide de trois types de modèles différents : un modèle de profil utilisateur, un modèle d'organisation et un modèle de sécurité.

Le modèle de profil utilisateur (modèle SM1) permet de gérer à la fois les informations spécifiques à l'utilisateur, comme, par exemple, son nom ou sa date de naissance, mais aussi ses préférences ou l'historique de l'ensemble de ses interactions avec l'application. Si les informations spécifiques sont définies à travers une structure de type hiérarchique, les préférences sont associées à l'ontologie de domaine. Ainsi, il est possible d'associer un niveau de préférence pour chaque classe de l'ontologie mais aussi pour chaque individu particulier de l'ontologie. Pour donner un exemple, l'utilisateur pourra indiquer qu'il préfère prendre le bus plutôt que de déplacer à pied (préférence de classe) et que lorsqu'il prend le bus, il préfère la société de transport X à la société de transport Y (préférence d'individus).

Le modèle d'organisation (modèle SM2) permet de définir la structure sociale dans laquelle les utilisateurs finaux de l'application évoluent. Il permet aussi d'indiquer l'ensemble des liens sociaux

entre les différents utilisateurs de l'application à travers la définition de groupes d'utilisateurs ayant entre eux des relations de type hiérarchique ("dirige", "gère", "fait partie de", etc.). Ceci permettra de définir au sein des autres modèles des règles de fonctionnement spécifiques en fonction du groupe social d'appartenance de l'utilisateur. Ce modèle permet de modéliser plusieurs structures humaines différentes comme, par exemple, une structure définissant l'organisation de la clientèle externe en groupes sociaux organisés et une structure définissant l'organisation hiérarchique d'une société industrielle comme une entreprise par exemple. Chaque utilisateur pourra alors faire partie d'une ou de plusieurs structures différentes et, au sein de chaque structure, il pourra appartenir à un ou plusieurs groupes sociaux différents. Ainsi, dans le projet ANR Viatic.Mobilité, le voyageur peut être pluriel : étudiant, retraité, touriste, etc. En Figure 3.8, on trouvera un exemple très simple de modèle d'organisation dans lequel on définit, par rapport à un groupe général contenant tous les utilisateurs, deux sous-groupes qui sont "Client SNCF" si l'utilisateur est un client référencé de la SNCF et "Pas Clients SNCF" s'il ne l'est pas. L'OMG n'ayant pour l'instant fait aucune préconisation sur le formalisme à utiliser pour les modèles d'organisation, le formalisme utilisé par PERCOMOM pour ce type de modèle est spécifique à PERCOMOM.

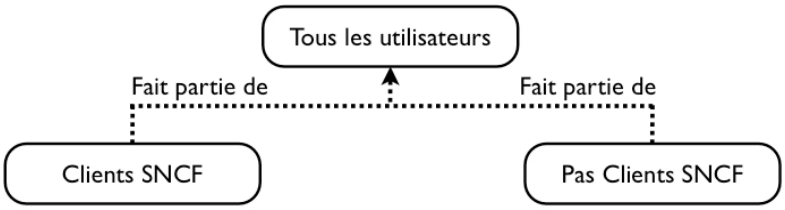


Figure 3.8. Exemple simple de représentation d'un modèle d'organisation avec un formalisme spécifique à PERCOMOM (modèle SM2)

En Figure 3.9, on trouvera un deuxième exemple de modèle social représentant une petite partie de l'organisation hiérarchique interne de la SNCF où la direction générale dirige directement (lien de subordination) trois autres directions :

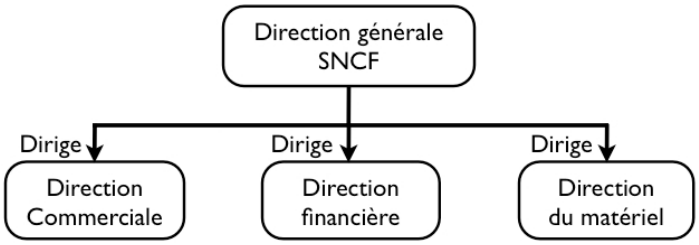


Figure 3.9. Exemple simplifié de représentation de la structure hiérarchique de la SNCF avec un formalisme spécifique à PERCOMOM (modèle SM2)

L'association des utilisateurs au modèle d'organisation est effectuée au niveau de chaque groupe social des modèles d'organisation. Cette association se fait à l'aide d'un fichier de définition XML dans lequel on indique la liste des utilisateurs individuels faisant partie du groupe social ainsi que la liste des autres groupes sociaux faisant partie eux aussi du groupe social considéré (lien d'appartenance). Par exemple, si on suppose qu'on définit un groupe social "Client Business SNCF" ; ce groupe social fera partie du groupe social "Clients SNCF" (pour la SNCF, des clients professionnels sont des clients comme les autres à qui ils offrent des services en plus). Par contre, l'inverse n'est pas vrai. Afin d'éviter d'avoir à redéfinir deux fois la liste des utilisateurs dans les deux groupes sociaux, on ne définira ceux-ci que dans le groupe social "Client Business SNCF" et l'on intégrera cette liste de définition dans le groupe social "Clients SNCF". Ainsi, le fichier de définition ressemblerait à :

```
<!-- Chaque groupe social est caractérisé par un identifiant unique et par un nom -->
<groupesocial id="667" nom="Clients SNCF">
    <!-- Chaque utilisateur peut être retrouvé par son identifiant unique ou par son nom -->
    <utilisateur id="123456"/>
    <!-- Chaque groupe social peut être retrouvé par son identifiant unique ou par son nom -->
    <groupesocial="Client Business SNCF"/>
</groupesocial>
```

Le modèle de rôles (modèle SM3) permet de définir, à partir du modèle d'organisation, les rôles utilisateurs associés à chaque application, ce qui permet ensuite d'autoriser ou pas l'accès à telle ou telle partie de l'application. Les rôles sont aussi utilisés au niveau du modèle d'application pour indiquer qui a accès à l'application. Il est important de noter ici la différence entre les groupes d'utilisateurs définis dans les modèles sociaux qui influent sur le comportement des applications et sur le contenu des informations retournées à l'utilisateur et les rôles d'accès qui contrôlent les accès de l'utilisateur à telle ou telle partie de l'application.

Dans notre approche, la définition des rôles se fait à l'aide d'un modèle graphique, spécifique à PERCOMOM, dans lequel il est possible d'utiliser des propriétés du modèle de profil utilisateur ou d'autres modèles sous forme de restrictions. Par exemple, si on suppose que la SNCF décide de ne donner l'accès à certains moyens de paiement et à certains services que si l'utilisateur est majeur (c'est-à-dire qu'il est âgé de plus de 18 ans), il faudra alors, à partir du groupe social "Clients SNCF" pouvoir définir deux types de rôles différents. Pour cela, on utilisera la propriété "age" du profil de l'utilisateur qui retourne l'âge de l'utilisateur. En fonction de cette propriété, on définira alors deux rôles d'accès "Personne majeure" et "Personne mineure" permettant de tenir compte de l'âge de l'utilisateur au niveau des accès aux différentes parties des applications (cf. Figure 3.10). Le formalisme utilisé dans le cadre de la création des modèles sera présenté plus en détail dans le chapitre 3, §3.1.4.1.

Figure 3.10. Exemple de définition de deux rôles d'accès à l'aide d'informations contenues dans le profil de l'utilisateur et d'un formalisme spécifique à PERCOMOM (modèle SM3)

Cette façon de faire permet de définir des rôles dynamiques qui ne sont pas uniquement dépendants de la liste des utilisateurs qui les composent. Ainsi, dans l'exemple de la Figure 3.10, le jour où une personne faisant partie du groupe social "Clients SNCF" atteint ses 18 ans, elle accède automatiquement au rôle d'accès "Personne majeure" sans nécessiter de modifications au niveau de l'application. Ceci signifie que les accès peuvent tenir automatiquement compte de l'évolution du profil de l'utilisateur mais aussi d'autres éléments comme, par exemple, le temps, la localisation de l'utilisateur où encore en fonction des événements survenant dans l'environnement (les événements sont définis dans le modèle EM3). Ce modèle étant un modèle de support dans le cadre de notre approche, nous ne développerons pas plus les nombreuses possibilités offertes. Néanmoins, il est important de noter que cette définition dynamique des rôles représente un élément important dans le cadre de la personnalisation et de l'adaptation des applications à l'utilisateur et au contexte d'usage aussi nous reviendrons dessus dans le chapitre 4.

3.1.3.4 Les modèles environnementaux (EM1, EM2 et EM3)

Les modèles environnementaux (cf. Figure 3.6) permettent de modéliser l'environnement dans lequel l'application va être utilisée. Dans ce cadre, seuls les éléments pouvant avoir une interaction au niveau du fonctionnement de l'application sont pris en compte. Dans le cadre de nos travaux, trois types de modèles ont ainsi été identifiés :

Le modèle géographique (modèle EM1) permet de modéliser l'environnement géographique dans lequel évolue l'application. Celui-ci est défini à travers une ontologie de classes géographiques (Continent, Pays, Région, Ville, etc.) reliés les uns aux autres par des propriétés d'inclusion et d'exclusion et par une base de connaissance contenant les individus associés à chaque classe géographique. À chaque individu, on associe une définition géographique compatible avec les systèmes cartographiques de type GIS (Geographic Information System). La Figure 3.7, présentée dans le chapitre 3, §3.1.3.2, montre un extrait d'une ontologie géographique adaptée au domaine des transports.

À partir de cette ontologie, on définit ensuite des individus qui seront utilisables comme critères dans l'ensemble des autres modèles. Par exemple, la gare ferroviaire de Lille ou la gare ferroviaire de

Valenciennes représentent deux individus de la classe "Gare_ferroviaire" de l'ontologie géographique. Une fois ces individus définis, il est possible de les utiliser dans des modèles de rôles pour définir, par exemple, des rôles différents dans chaque gare. De cette manière, l'utilisateur pourrait accéder à des services spécifiques disponibles uniquement dans certaines gares. De même, il est possible d'utiliser les classes de l'ontologie géographique comme critère dans les autres modèles. Ainsi, il devient possible, par exemple, de donner un rôle d'accès déterminé lorsque l'utilisateur se trouve dans une gare et un autre lorsqu'il se trouve dans un monument (cf. Figure 3.11).

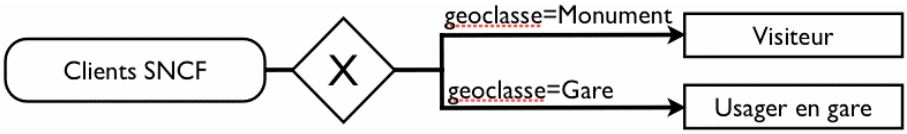


Figure 3.11. Exemple d'utilisation de l'ontologie géographique dans un modèle de sécurité (modèle SM3)

Le modèle des systèmes externes (modèle EM2) permet pour sa part de prendre en compte les systèmes externes avec lesquels l'application communique (imprimante, progiciel, etc.). Pour cela, on définit le système externe comme étant une boîte noire disposant d'entrées et de sorties, d'éléments de commande et d'un type de relation avec les applications (relation synchrone ou asynchrone).

Afin de pouvoir prendre en compte le fait que les systèmes externes ne sont pas toujours disponibles en fonction de la localisation de l'utilisateur, comme, par exemple, une imprimante, on associera à chaque système externe, à l'aide d'une ontologie des systèmes externes, une ou plusieurs classes d'appartenance. Chaque classe permettra de regrouper ensemble des systèmes externes ayant un comportement identique, par exemple des imprimantes, ce qui permettra de définir au sein des applications soit des associations avec des systèmes externes spécifiques soit des associations avec des classes de systèmes externes. Ainsi, lorsqu'un utilisateur en déplacement aura besoin d'imprimer une information, cette information se fera sur l'imprimante la plus proche physiquement et non sur une imprimante bien définie. Pour cela, on doit définir, pour chaque système externe la possibilité de lui associer une clause de validité géographique et/ou temporelle permettant de savoir à chaque instant les systèmes externes valides pour un utilisateur en fonction du lieu où il se connecte, de l'application et de l'heure et du jour.

Le modèle des événements (modèle EM3) permet de définir l'ensemble des événements pouvant être gérés par les applications. Chaque événement dispose d'un nom unique, d'une portée applicative permettant d'indiquer s'il est valable pour une ou pour plusieurs classes d'applications, d'une portée sociale permettant d'indiquer pour quels groupes sociaux l'événement est valide, d'une portée géographique permettant d'indiquer sa zone de validité géographique (utilise le modèle géographique) et d'une portée temporelle permettant d'indiquer sa durée de validité. Cette dernière peut aussi bien être définie par une durée en minutes, en heures ou encore en jours qu'être liée au traitement même de l'événement ; l'événement devenant, par exemple, inactif dès qu'un utilisateur l'a traité.

Pour donner un exemple, la survenue d'un problème technique empêchant les rames de circuler sur une ligne de métro pourrait générer un événement dont la durée de vie serait liée à la durée du problème technique ; cet événement pourrait être utilisé au sein des applications pour signaler le problème aux usagers et comme information pour le système de calcul d'itinéraires afin d'offrir aux usagers des solutions de substitution. Dans PERCOMOM, un événement est défini à l'aide d'un formalisme de type XML. Ainsi, si on veut définir qu'un événement "événement1", une fois déclenché, sera actif pendant 10 minutes, on définira le modèle d'événement (EM3) suivant :

```
<événement>
    <id valeur="12"/>
    <nom valeur="événement1"/>
    <dureedevie valeur="DuréeLimitée"/>
    <duréedevieenminutes valeur="10"/>
</événement>
```

À titre d'information, l'ensemble des propriétés associées à un événement, défini au niveau des modèles de type EM3, est défini dans l'annexe B de ce mémoire.

3.1.3.5 Les modèles Comportementaux (BM1, BM2 et BM3)

Les modèles comportementaux (cf. Figure 3.6) permettent de modéliser les éléments qui influent sur le comportement de l'application mais qui ne sont pas liés aux interactions homme-machine.

Le modèle des actions (modèle BM1) permet de modéliser des successions de tâches métier sans aucune interaction avec l'utilisateur. Chaque action peut avoir des valeurs en entrée qui sont fournies par l'interface utilisateur et peut modifier, en sortie, les valeurs affichées au niveau de l'interface utilisateur. Pour définir le contenu des actions, on utilise, au sein de chaque tâche, un pseudo langage spécifique dérivé du langage Pascal.

Un exemple d'action pourrait être la mise en forme et l'impression d'une fiche de déplacement pour un usager au niveau d'une borne d'information. L'utilisateur initie le démarrage de l'action, mais les différentes tâches contenues dans l'action sont toutes exécutées de manière automatique sans aucune intervention de sa part. Dans la définition du contenu des tâches et dans la définition des actions elles-mêmes, il est possible d'utiliser des informations provenant d'autres modèles. Si ces informations doivent permettre de faire des sélections, il est fortement recommandé d'utiliser des règles métier (modèle BM3). Par contre, si elles sont utilisées dans le cadre d'opérations, l'appel direct est possible. Par exemple, si on veut faire un calcul qui permet de compter le nombre de jours depuis la naissance de l'utilisateur, on fera directement appel à la propriété contenant la date de naissance de l'utilisateur dans le profil. Par contre, si, dans l'enchaînement des tâches, on veut faire, après une tâche initiale, une action si l'utilisateur est de sexe masculin et une autre s'il est de sexe féminin (par exemple, indiquer un chemin différent pour se rendre aux toilettes), on fera appel à une règle métier (modèle BM3) qui permet, par exemple, de déterminer si l'utilisateur est de sexe masculin. Ce dernier exemple est présenté dans la Figure 3.12.

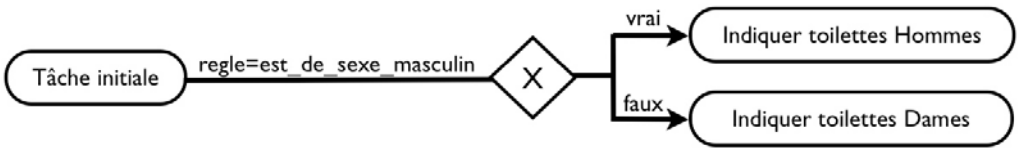


Figure 3.12. Exemple d'utilisation d'une règle métier dans un modèle d'action en utilisant un formalisme spécifique à PERCOMOM (modèle BM1)

Le modèle des données (modèle BM2) permet de relier les concepts de l'ontologie du domaine métier à un modèle conceptuel de données de type entité-relation (Chen, 1976) à travers un fichier de correspondance. Ainsi, si on suppose que dans une ontologie de domaine (OD) on dispose de deux classes "Société de transport" et "Moyen de transport" reliées entres-elles par des relations "est pris en charge par" et "gère" et que les deux classes ne possèdent que des propriétés n'acceptant qu'une seule valeur alors le modèle entité-relation correspondant sera composé de deux entités et de deux relations (cf. Figure 3.13). Une présentation plus détaillée des modèles BM2 est présentée dans le chapitre 4, §4.4.2.

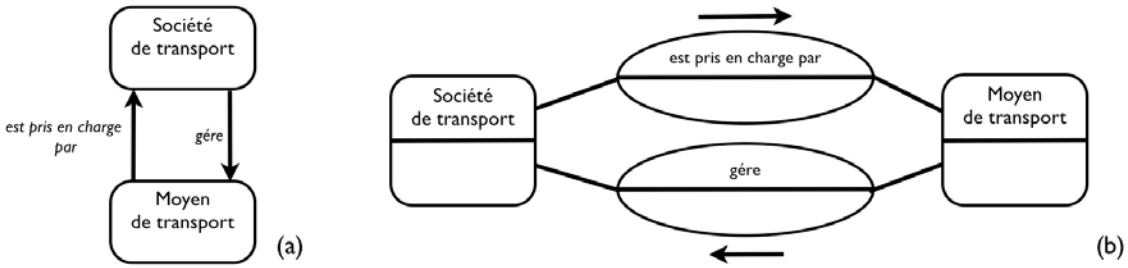


Figure 3.13. Exemple de transformation d'un modèle de relation dans une ontologie de domaine (a) vers un modèle entité-relation de définition d'une base de données (b)

Les règles métier (OMG, 2005) (modèle BM3) sont un des modèles de support les plus importants dans PERCOMOM. Elles permettent de sortir une partie de la logique métier des processus métier rendant ceux-ci plus faciles à modifier. Une définition des règles métier est donnée par (The Business Rules Group, 2000) :

"*A business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the business. The business rules that concern the project are atomic - that is, they cannot be broken down further.*"

Dans le cadre de notre approche, nous avons défini trois types de règles métier :

- les règles dites de validation dont l'évaluation retourne une valeur de type booléen ;
- les règles de sélection dont l'évaluation retourne une valeur qui peut être une chaîne de caractères, une date ou encore un nombre ;
- les règles d'action dont l'évaluation va déclencher l'exécution d'une action particulière.

La définition de ces règles peut utiliser l'ensemble des éléments définis dans les modèles présentés précédemment. Pour donner un exemple, dans le cadre de la diffusion de publicités sur la plateforme de l'utilisateur, une règle de validation pourrait être créée afin de savoir si une personne est ou non senior ; la définition du terme senior pouvant être, par exemple, une personne de plus de 60 ans. Au niveau du modèle de règle, celle-ci, dénommée "Est_Un_Senior", serait définie comme suit :

```
<!-- Une clause est l'élément de base du moteur de règles -->
<clause id="789" nom="Age_superieur_a_60_ans">
    <element type="profil" nom="age"/>
    <comparateur type=">="/>
    <membredroit type="valeurfixe" typevaleur="entier" valeur="60"/>
</clause>
<!-- Une expression représente une association logique de clauses et d'expressions -->
<expression id="12" nom="Expression_age_superieur_a_60_ans">
    <contenu type="clause" id="789"/>
</expression>
<!-- Une règle métier possède un type et est définie à partir d'une expression -->
<regle id="23" nom="Est_Un_Senior" idexpression="12" type="validation"/>
```

L'ensemble du formalisme associé à la définition des règles dans PERCOMOM ainsi que le moteur de règles utilisé sont précisés dans l'annexe A de ce mémoire.

Au niveau de la prise en compte du contexte, nous utilisons, dans le cadre de notre approche, des règles métier, de type validation, un peu particulières que sont les domaines de validité. De manière simple, un domaine de validité est composé d'un ensemble de restrictions qui sont des inclusions ou des exclusions sur des éléments contextuels. Par exemple, si on veut définir qu'une source de données est valable uniquement pour la ville de Lille, on doit définir un domaine de validité géographique de type inclusion dans lequel on effectuera une comparaison de la localisation actuelle de l'utilisateur par rapport à la ville de Lille. Le fichier de description du domaine de validité associé à cet exemple est donné ci-après :

```
<!—Un domaine de validité est défini par un identifiant unique et par un nom -->
<domainevalidite id="4" nom="geo_dans_la_ville_de_lille">
    <!-- Une restriction géographique est définie par un identifiant, par un nom et par son type -->
    <restrictiongeographique>
        <id valeur="987"/>
        <nom valeur="ville_de_lille"/>
        <type valeur="inclusion"/>
        <typedezone valeur="ontologie_individu"/>
        <definitionzone valeur="Lille"/ >
    </restrictiongeographique>
</domainevalidite>
```

Actuellement, dans PERCOMOM, nous ne prenons en compte, dans le cadre des domaines de validité, que trois types d'informations de contexte : l'espace géographique, le temps et le domaine applicatif. Ces trois types de restrictions, qui peuvent être assemblées pour former un domaine de validité, sont gérées de manière séparée à travers des restrictions géographiques, des restrictions temporelles et des restrictions applicatives qui sont présentées en détail dans l'annexe A.

3.1.3.6 Synthèse et discussion

Étant donné que notre domaine de recherche est centré sur la modélisation des IHM et sur la personnalisation, nous n'avons pris en compte qu'un nombre limité de modèles de support. Ces modèles représentent, d'après nos analyses préalables d'identification des modèles (cf. chapitre 3.1.2.2) les modèles minimum nécessaires pour la prise en compte des applications d'information des voyageurs dans le domaine des transports.

Suite à leur utilisation dans le cadre de la réalisation de différentes applications et suite aux contraintes fonctionnelles et techniques que nous nous sommes imposées dans le cadre de la thèse (applications dans le domaine des transports et interfaces de type WIMP), nous avons pu identifier les principaux avantages et inconvénients de ces différents modèles de support (cf. Tableau 3.1).

Tableau 3.1. Les principaux avantages et inconvénients des modèles de support de niveau CIM dans le cadre de la méthode PERCOMOM

Avantages	Inconvénients
• L'utilisation d'une ontologie de domaine permet d'avoir une certaine abstraction par rapport aux données physiques. En effet, on ne sait qu'au niveau du modèle BM2 à quel modèle de données est rattachée chaque classe de l'ontologie. • Mis à part le modèle d'application, les modèles sont conçus pour être indépendants des applications ; ce qui facilite leur réutilisation au sein d'autres applications. Ainsi, un modèle d'organisation, un modèle géographique ou encore un modèle de règle métier peuvent être partagés par plusieurs applications. • La spécialisation des modèles permet, en théorie, une prise en charge de chaque type de modèles par un expert métier dédié comme, par exemple, un spécialiste de la sécurité pour le modèle de sécurité (SM3). Ce point n'a pour l'instant pas pu être validé dans le cadre de la thèse. • La spécialisation des modèles permet d'utiliser une notation et/ou une représentation graphique adaptée à chaque problématique traitée et donc à chaque type d'experts métier. En l'absence de validation effective des différents modèles auprès d'experts métier, cet avantage reste théorique et repose sur les suppositions de l'OMG (Watson, 2005). • La plupart des modèles proposés sont, par nature, dynamiques et permettent d'envisager des adaptations automatiques au niveau des applications finales. Ainsi, les accès d'un utilisateur peuvent s'adapter automatiquement à des changements dans son profil ou à un changement de lieu pour l'utilisateur. • Les modèles proposés permettent de couvrir des problématiques rarement traitées dans les autres méthodes de modélisation comme, par exemple, la gestion du contexte géographique ou encore la gestion des accès ou de l'organisation sociale des utilisateurs.	• Le modèle des données (BM2) n'est pas un modèle totalement conceptuel dans le sens où il est très lié à des méthodes de modélisation spécifiques aux bases de données relationnelles. • Certains modèles peuvent se recouvrir au niveau de la définition de certains aspects. Aussi, il est indispensable de bien définir le cadre précis d'utilisation de chaque type de modèles à travers la mise en place de guides d'utilisation. • Les interactions entre les modèles pouvant être importantes, leur évolution peut devenir problématique dans le temps et surtout limiter la réutilisation des modèles. Pour cela, il est indispensable de définir avec précision l'ensemble des types d'interactions possibles tout en limitant leur nombre. • Les modèles proposés ne sont probablement pas complets dans le sens où ils n'ont été testés que dans le cadre d'un nombre limité d'applications. • Dans le cadre de la thèse, il n'a pas été possible, pour des raisons de temps, de vérifier que les modèles permettaient de prendre en charge l'ensemble des problématiques associées à la diffusion d'information dans le domaine des transports. • La compatibilité de l'ensemble des modèles avec le métamodèle MOF de l'OMG n'a pas été validée dans le cadre de la thèse.

En plus de ces modèles dits de support, notre approche propose d'utiliser un ensemble de trois types de modèles spécialement développés pour prendre en charge les IHM.

3.1.4 La modélisation des interactions au niveau conceptuel (niveau CIM)

3.1.4.1 Le modèle des processus métier (IM1)

Du point de vue général, l'utilisation d'une IHM s'inscrit toujours dans un processus global beaucoup plus large. Ainsi, on n'utilise pas une IHM pour le plaisir de l'utiliser mais pour faire quelque chose, pour atteindre un certain but (Norman, 1986). Aussi, dans le cadre de nos travaux de recherche, nous avons considéré que les IHM étaient un élément de support des processus métier tel que nous les avons décrits dans le chapitre 1.

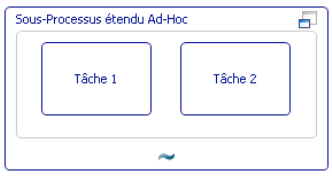
Cette hypothèse étant posée, le processus métier devient alors le cœur même de l'ensemble de la modélisation des interactions dans le sens ou :

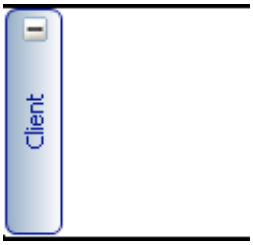
- Il permet de modéliser l'ensemble des buts métier de l'utilisateur dans une application ; chaque but métier donnant lieu à la création d'un ou de plusieurs processus métier.
- Il donne une raison d'être à chaque élément de l'IHM qui s'inscrit ainsi dans une logique métier bien précise ; chaque élément d'interaction pouvant être utilisé dans le cadre de l'exécution d'un ou de plusieurs processus métier.
- Il permet de prendre en compte l'ensemble des tâches permettant de réaliser un but métier (tâches interactives, tâches non interactives et tâches réalisées manuellement) et, en ce sens, est, en théorie, plus générique que les modèles de tâches plus classiques comme CTT (Concurrent Task Trees).
- Il permet de prendre en compte l'ensemble des échanges d'information effectués entre les différents intervenants (homme ou machine) dans le cadre d'un même processus métier.

Cette position centrale du modèle de processus métier, qui apparaît clairement dans les approches actuelles de type SOA (Service-Oriented Architecture), impose d'utiliser des outils et langages de modélisation capables de prendre en charge l'ensemble de ces points. Or, si on dispose aujourd'hui d'une notation, la notation BPMN (Business Process Modeling Notation), capable de définir l'ensemble d'un processus métier, son cadre d'utilisation reste limité. Ainsi, les outils proposés autour de cette notation sont très souvent orientés vers la manipulation de composants métier dans une architecture de type SOA et pas vers l'utilisation de tâches métier élémentaires. Aussi, à notre connaissance, aucun de ces outils ne permet de prendre en compte correctement les IHM qui sont composées, du point de vue des processus métier, d'un ensemble de tâches élémentaires. Voilà pourquoi, dans le cadre de nos travaux de recherche, nous avons défini, en partant du formalisme BPMN, un certain nombre d'éléments de notation spécifiques permettant de prendre en charge l'ensemble des éléments d'interaction.

Dans le Tableau 3.2, le lecteur trouvera les principaux éléments du formalisme général, dérivés du formalisme BPMN, utilisé dans le cadre des modèles IM1 de PERCOMOM. Les éléments de notation spécifiques à PERCOMOM ou utilisés d'une manière spécifique dans PERCOMOM sont, dans le tableau, suivi du signe *.

Tableau 3.2. Eléments du formalisme général des processus métier utilisés dans le cadre de la méthode PERCOMOM dérivés de la notation BPMN (modèle IM1)

Notation	Signification
	**Evénement de départ :** marque le début d'un processus métier. Il ne peut y avoir qu'un événement de départ par diagramme.
	**Evénement de fin :** marque la fin d'un processus métier.
	**Activité/Processus/Sous-Processus :** une activité représente un travail effectué. Si elle n'est pas composée d'autres activités, on dit que c'est une tâche élémentaire ; sinon, c'est un processus ou sous-processus standard dans lequel l'ordre d'exécution des activités est fixé. Remarque : chaque tâche doit avoir un nom unique permettant de l'identifier ; le nom étant indiqué sur la représentation graphique de la tâche.
 *	Représentation graphique d'une tâche "utilisateur" réalisée par un intervenant humain du processus. Dans le cadre de la méthode de modélisation proposée, ce type de tâche représente toutes les tâches interactives qui sont à la charge de l'utilisateur comme une entrée de données sur un formulaire ou un bouton à cliquer pour effectuer certaines tâches. Remarque : l'utilisation de ce formalisme est optionnelle et peut être remplacé par celle du formalisme plus général de représentation d'une activité/processus/sous-processus.
 *	Représentation graphique d'une tâche "système" qui est réalisée par l'application, par le système. Dans le cadre de la méthode de modélisation proposée, les tâches système peuvent être aussi bien des tâches interactives, comme un affichage de données à l'utilisateur, que des tâches non interactives comme un calcul d'un itinéraire. Remarque : l'utilisation de ce formalisme est optionnelle et peut être remplacée par celle du formalisme plus général de représentation d'une activité/processus/sous-processus.
 *	Représentation graphique d'une tâche "manuelle" qui est réalisée par un intervenant humain mais qui n'a pas d'interaction directe avec l'application. Dans le cadre de la méthode de modélisation proposée, les tâches manuelles permettent de modéliser l'ensemble des tâches réalisées par un acteur humain qui ne peuvent pas être prises en compte par l'application. Pour donner un exemple, lors de l'achat d'un billet de train au guichet d'une gare, il y a un échange verbal d'informations entre l'usager et l'employé au guichet. Cet échange verbal correspond, du point de vue de l'usager, à une succession de tâches réalisées suivant une certaine séquence. Ces tâches sont indispensables à la bonne réalisation du processus métier associé mais aussi à sa bonne compréhension. Modéliser les tâches manuelles revient donc à faciliter la compréhension des processus métier et donc à mieux les adapter aux besoins réels des utilisateurs et à mieux les faire évoluer dans le temps. Remarque : l'utilisation de ce formalisme est optionnelle et peut être remplacée par celle du formalisme plus général de représentation d'une activité/processus/sous-processus.
	**Processus « Ad-Hoc » :** une activité « Ad-Hoc » représente un processus dans lequel les sous-processus peuvent être appelés indépendamment les uns des autres (il n'y a pas d'ordre contrairement à un processus standard). Au niveau graphique, un processus "Ad-Hoc" est identifié à l'aide du signe "~".
	**Processus/sous-processus non étendu :** un processus non étendu est la représentation simplifiée d'un processus (sur le diagramme ne sera visible que le nom du processus). À tout processus non étendu est associé un processus/sous-processus « étendu » (représenté sous la forme d'un processus/sous-processus standard) qui permet d'en préciser le contenu. Remarque : l'affichage de l'icône représentant trois tâches liées ensemble est optionnel.

Notation	Signification
	Représente une association vers un élément qui, dans notre méthode, devra être précisé à travers l'utilisation d'un artefact.
	Représente un flux entre deux activités (tâches ou processus) : la fin de l'activité à l'origine du flux entraîne le démarrage de l'activité situé(e) à l'extrémité du flux.
	Représente un flux de type « envoi d'informations » entre deux tâches ou deux processus.
	Représente, dans les flux conditionnels, le flux par défaut qui indique l'activité à activer si aucune autre condition n'a été validée. Un flux conditionnel est un flux qui utilise des choix conditionnels pour définir la ou les tâches à exécuter.
	Représente, au sein d'un flux conditionnel, une porte logique de type OU exclusif permettant de faire des branchements conditionnels. Ceci permet, par exemple, de lancer des activités spécifiques après la fin d'une activité en fonction de la valeur d'une variable ou d'un paramètre.
	Représente, au sein d'un flux conditionnel, une porte logique de type OU permettant de faire des branchements conditionnels.
	Représente, au sein d'un flux conditionnel, une porte logique parallèle qui permet d'exécuter des tâches ou des sous-processus en parallèle.
	Un ensemble d'activités représente l'ensemble des processus/activités effectués par un intervenant (humain ou non). Par exemple, pour un processus d'achat d'un titre de transport à un automate, on aura, de manière simplifiée, deux ensembles de tâches : un pour toutes les tâches réalisées par l'utilisateur et un pour toutes les tâches réalisées par le système (l'automate dans le cas présent) ; le lien entre les deux ensembles se faisant à l'aide de flux. Par défaut, on recommande d'utiliser pour le nom de l'ensemble d'activité le nom du rôle métier associé aux tâches qu'il contient. Dans le cadre de PERCOMOM, l'utilisation des ensembles d'activités est recommandée mais non obligatoire.
	Un timer est une tâche particulière dont la seule action est d'attendre pendant un temps déterminé avant d'exécuter la tâche suivante. Le temps d'attente est précisé au niveau des propriétés du timer.
 *	Un trigger de message est une tâche particulière qui n'est activée qu'à la réception d'un message extérieur bien particulier. Dans le cadre de PERCOMOM, les seuls messages extérieurs gérés au niveau des modèles métier sont les événements définis dans les modèles EM3. Ainsi, tout événement survenant à un instant T déclenche automatiquement l'activation de tous les triggers de message qui lui sont associés
 *	**Artefact d'interaction :** un artefact d'interaction contient l'ensemble des propriétés spécifiques définies pour un élément d'interaction particulier. Dans le cadre de l'approche de modélisation, il est possible d'associer un artefact d'interaction à chaque type d'élément statique d'interaction. Au niveau graphique, un artefact d'interaction est identifié par son type xxxxArtefact ; xxxx étant remplacé par le type d'élément d'interaction comme par exemple une unité graphique d'interaction (cf. chapitre 3, §3.1.3.2).

Il est intéressant de noter qu'une des particularités de la notation BPMN est d'avoir été conçue en se basant sur les réseaux de Petri et donc de permettre, en théorie, une validation des modèles à deux niveaux :

- Vérification de l'absence de points de blocage dans les processus métier (un processus métier a toujours une fin).

- Vérification de l'accessibilité de toutes les tâches définies dans chaque processus métier (toutes les tâches métier définies sont accessibles à un moment ou un autre dans un processus métier).

Malheureusement, les outils disponibles actuellement ne permettent pas de faire ce type de validation bien que celle-ci soit un avantage important de BPMN par rapport aux autres notations existantes. À ce sujet, on pourra se reporter aux travaux de (Dikjman *et al.*, 2007) et à l'outil "*BPMN to Petri Net Transformer*" développé dans le cadre de leurs travaux de recherche et qui est disponible à l'adresse http://is.tm.tue.nl/staff/rdijkman/cbd.html. Dans le cadre de nos travaux, les outils de modélisation développés ne permettent pas encore de prendre en compte ces deux types de vérification de manière totalement automatique.

En fait, la vraie problématique qui se cache derrière ces aspects de vérification des modèles est la validation fonctionnelle des modèles au niveau conceptuel. Celle-ci est en effet indispensable dans le cadre d'une approche dirigée par les modèles où la définition et l'assemblage de modèles conceptuels conduisent à la création automatique d'une application directement utilisable. Or, comme dans toute approche de développement, il est indispensable de pouvoir effectuer un maximum de tests avant de livrer l'application aux utilisateurs. Et, si possible, ces tests doivent être réalisés le plus tôt possible dans le cadre du processus de développement afin de limiter au maximum l'impact de tout problème fonctionnel détecté. Dans une approche de type IDM, ces tests ne peuvent se faire qu'au niveau des modèles conceptuels ce qui impose de développer des outils et des méthodologies permettant de valider ces modèles au niveau fonctionnel. Dans ce cadre, l'utilisation des propriétés de type réseau de Petri des modèles métier ne représente qu'un élément de réponse pour résoudre cette problématique. Celle-ci représente, en fait, un axe de recherche important dans le cadre de la généralisation des approches de type IDM sur lequel nous espérons pouvoir nous pencher à l'avenir.

Les modèles de processus métier étant très fortement liés aux modèles statiques d'interaction, les exemples de définition et d'utilisation de processus métier seront donnés dans le chapitre 3, §3.1.4.2.

Si le modèle des processus métier permet de représenter l'ensemble des tâches nécessaires pour atteindre un but métier particulier, il s'appuie, au niveau des IHM, sur un modèle statique de représentation des IHM que nous allons présenter maintenant.

3.1.4.2 Le modèle statique d'interaction (IM2)

Pour rappel, il est important de noter que dans le cadre de nos travaux, nous n'avons pour l'instant pris en compte, du point de vue des IHM, que les applications interactives de type WIMP. Ce choix a été effectué afin de permettre la validation de l'approche dans sa globalité (de la modélisation à la création d'application). Ceci étant, la définition et la modélisation de tous les types d'IHM représentent, de notre point de vue, un axe important de recherche dans le cadre des approches de type IDM. Celles-ci, en séparant les aspects fonctionnels des aspects techniques, imposent la création de modèles conceptuels d'IHM valides pour tous les types d'IHM et ceci quels que soient les moyens d'interaction entre l'utilisateur et la ou les machine(s).

Pour revenir à notre approche de modélisation des IHM, le modèle statique d'interaction permet :

- de modéliser les éléments d'interaction utilisables dans le cadre d'une application ;
- de regrouper les éléments d'interaction en ensembles cohérents du point de vue de la logique métier ;
- de définir les liens entre les différents éléments d'interaction ;
- de définir les comportements spécifiques de chaque élément d'interaction.

Pour cela, le modèle statique d'interaction s'appuie sur une hiérarchie de composants d'interactions dont les principaux éléments sont représentés dans la Figure 3.14 ; une description de quelques éléments est fournie dans le Tableau 3.3. Une description complète de l'ensemble des éléments présentés sur la Figure 3.14 est donnée dans l'annexe B. De par ses principes généraux, ce modèle s'appuie beaucoup sur les propositions et idées proposées par UsiXML pour modéliser de manière conceptuelle les interfaces graphiques.

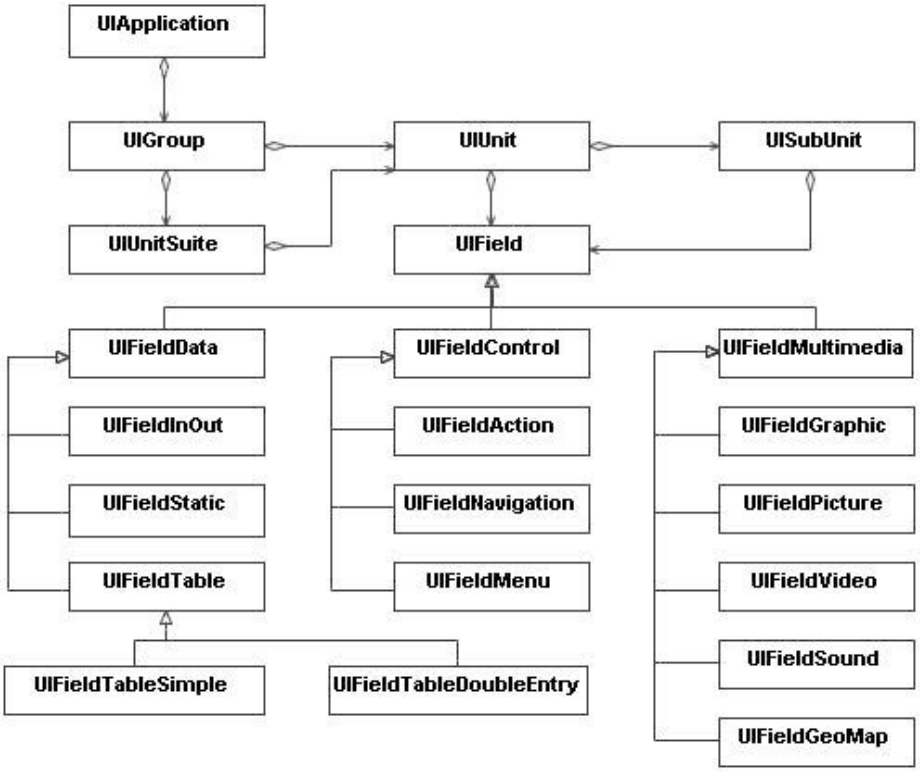


Figure 3.14. Les principaux éléments statiques d'interaction dans PERCOMOM au niveau CIM (présentation sous la forme d'un diagramme de classes)

Tableau 3.3. Description globale des principaux éléments statiques d'interaction dans PERCOMOM au niveau CIM (modèle IM2)

Élément d'interaction	**Description**
UIGroup	Un élément de type *UIGroup* permet de regrouper des *UIUnit* ou des *UIUnitSuite* en groupes logiques du point de vue métier. Cet élément sera présenté plus en détail dans la suite de cette section.
UIUnit	Un élément de type *UIUnit* permet de regrouper des éléments d'interactions en groupes logiques du point de vue des interactions ; l'ensemble des éléments d'interaction constitutifs de l'*UIUnit* ne pouvant être séparés les uns des autres du point de vue de la logique d'interaction. Cet élément sera présenté plus en détail dans la suite de cette section.
UIFieldInOut	Un *UIFieldInOut* représente un élément physique d'interaction permettant la saisie et l'affichage d'une propriété d'un individu ou de plusieurs individus de l'ontologie de domaine (modèle OD); cette propriété pouvant retourner une valeur unique ou une liste de valeurs.
UIFieldStatic	Un *UIFieldStatic* représente un élément physique d'interaction permettant l'affichage d'une information statique non modifiable à l'écran comme, par exemple, un libellé placé juste devant un champ de saisie de type *UIFiledInOut*. Le contenu d'un *UIFieldStatic* ne provient pas de l'ontologie de domaine mais est fixé, par l'expert métier, au moment de la création du modèle statique d'interaction.
UIFieldAction	Un *UIFieldAction* représente un élément d'interaction permettant de démarrer une action comme, par exemple, un bouton.
UIFieldNavigation	Un *UIFieldNavigation repésente* un élément d'interaction permettant de naviguer d'un *UIGroup* vers un autre *UIGroup* ou d'un *UIUnit* vers un autre *UIUnit* faisant partie du même *UIUnitSuite*.

Dans le cadre de la méthode PERCOMOM, les différents types d'*UIElement* sont identifiés à l'aide d'une couleur de fond spécifique qui permet de rapidement identifier le contenu d'un *UIUnit*. Pour des raisons de lisibilité sur support papier, ces codes couleurs n'ont été utilisés dans le cadre de la réalisation de ce mémoire. Ils sont donnés à titre d'indication dans le Tableau 3.4.

Tableau 3.4. Liste des codes couleurs utilisés dans PERCOMOM dans les modèles IM2 de niveau CIM pour les principaux éléments d'interaction (format RGB)

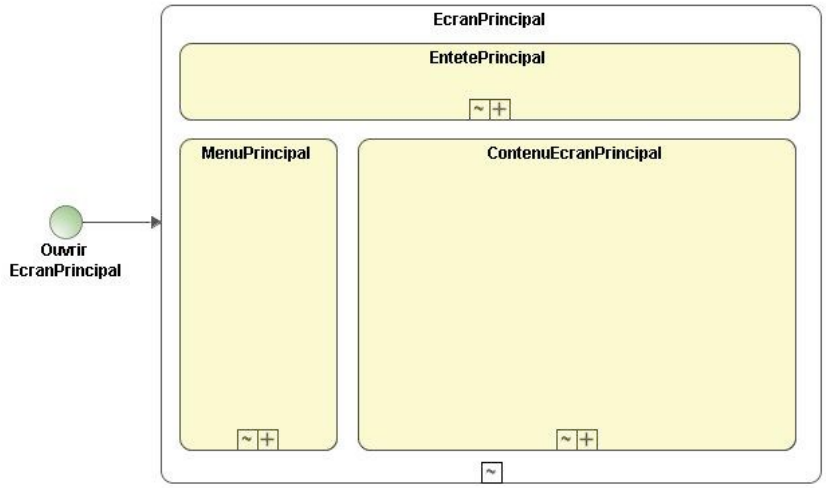
Élément d'interaction	Valeur R	Valeur G	Valeur B
UIGroup	255	255	255
UIUnit	255	255	0
UIUnitSuite	255	255	153
UISubUnit	255	255	204
UIFieldStatic	51	110	0
UIFieldInOut	51	160	0
UIFieldTableSimple	51	255	0
UIFieldTableDoubleLayer	51	204	0
UIFieldAction	33	99	255
UIFieldNavigation	33	140	255
UIFieldMenu	33	190	255
UIFieldGraphic	255	51	0
UIFieldPicture	210	51	0
UIFieldVideo	160	51	0
UIFieldSound	110	51	0

Par défaut, une application disposant d'une interface graphique est définie par un élément de type *UIApplication* auquel il est possible d'associer des propriétés dont les principales sont :

- L'identifiant du premier groupe d'éléments d'interactions à ouvrir au démarrage de l'application.
- Les groupes d'éléments d'interaction à ouvrir par défaut lorsque certains types d'erreurs sont détectés.

Dans le cadre de la modélisation globale des applications dans la méthode PERCOMOM, le modèle d'application permet de modéliser l'ensemble des éléments directement rattachés à une application, que celle-ci dispose d'interactions ou pas avec l'utilisateur. L'élément d'interaction *UIApplication* ne se substitue pas au modèle d'application mais le complète pour tous les aspects directement reliés aux IHM. Le lien se fait au niveau du modèle d'application à travers une propriété permettant d'indiquer si l'application est interactive et, si oui, quel est l'élément *UIApplication* utilisé. Ceci permet de faire une séparation stricte entre les éléments purement applicatifs associés à l'application et les éléments directement associés aux interactions.

Le premier regroupement d'éléments d'interaction se fait à travers l'utilisation d'*UIGroup* qui correspondent à des groupes logiques d'interactions du point de vue métier. Ceux-ci contiennent des unités logiques d'interactions dénommées *UIUnit* qui regroupent des éléments d'interactions qui ne peuvent pas être séparés du point de vue de la logique métier lorsqu'ils sont présentés à l'utilisateur contrairement aux *UIGroup* qui peuvent se présenter sous la forme de plusieurs écrans par exemple (un écran par *UIUnit*). Un exemple de définition du contenu d'un *UIGroup* et un exemple possible de représentation graphique de cet *UIGroup* sont donnés à la Figure 3.15.



(a)

(b)

(c)

Figure 3.15. Exemple de modèle statique d'interface (a) (modèle IM2 de niveau CIM) avec une de ses représentations graphiques possibles (niveau PSM) sur un ordinateur de type PC (b) et sur un téléphone portable de type SmartPhone (c)

Dans cet exemple, la plateforme cible choisie par défaut est une application de type Web (cas b). Si l'application devait être déployée sur un téléphone portable (cas c), chaque *UIUnit* pourrait être représenté, par exemple, par un onglet qui une fois sélectionné permettrait d'accéder au contenu de l'*UIUnit* en question. Ce qui signifie, que sur un téléphone portable, les trois *UIUnit* définis dans l'exemple précédent ne pourraient pas être vus simultanément mais uniquement de façon séparée. Par contre, dès qu'un *UIUnit* est sélectionné, le contenu de celui-ci est affiché en un bloc.

Dans un *UIGroup*, il est aussi possible de remplacer un *UIUnit* par un *UIUnitSuite* qui est un ensemble d'*UIUnit* regroupés du point de vue logique. De cette manière, il devient possible d'intégrer au sein d'un *UIGroup* un ensemble cohérent d'*UIUnit* dans lequel il est ensuite possible de définir une navigation sans avoir besoin de redéfinir un nouvel *UIGroup*.

Par exemple, un processus métier interactif standard pour se connecter à une application pourrait comporter un écran dans lequel l'utilisateur va indiquer son identifiant et son mot de passe, un écran de validation indiquant que l'identification s'est effectuée correctement et un écran de validation indiquant une erreur lors de l'identification. Ce qui représente, de manière simplifiée trois *UIUnit*. Sans l'utilisation d'un *UIUnitSuite*, l'utilisation du processus d'identification dans une application nécessiterait la création d'un minimum de trois *UIGroup* différents. Et, ceci serait à répéter au niveau de chaque application, ce qui limiterait fortement la réutilisation. Avec l'*UIUnitSuite*, il est possible de regrouper les trois *UIUnit* reliés au processus d'identification ensembles. La navigation entre les différents éléments constitutifs de l'*UIUnitSuite* se fait ensuite grâce à l'utilisation d'un formalisme spécifique de navigation. De cette manière, il devient possible d'intégrer le processus d'identification dans n'importe quel *UIGroup* sous la forme d'un *UIUnitSuite*. Du point de vue des IHM, le processus devient alors facilement modifiable, puisqu'il suffit de modifier le contenu de l'*UIUnitSuite*, et surtout

facilement réutilisable puisqu'il suffit d'intégrer l'*UIUnitSuite* dans n'importe quel *UIGroup* pour permettre à celui-ci d'utiliser le processus interactif d'identification.

Du point de vue des propriétés associées au *UIUnitSuite*, celles-ci sont très similaires aux propriétés associées aux *UIApplication* dans le sens où les *UIUnitGroup* peuvent être vus comme des petites applications interactives. De manière pratique, un *UIUnitSuite* est composé d'une liste d'*UIUnit* disposant d'un pointeur sur le premier élément d'*UIUnit* à afficher lors de l'affichage par défaut de l'*UIUnitSuite* à l'écran.

Les *UIUnit* sont constitués d'*UIElement*, qui sont des éléments d'interaction entre l'utilisateur et l'application, comme des éléments de navigation, des éléments d'entrée de données ou encore des éléments de type multimédia. Ils peuvent aussi être constitués d'*UISubUnit*, qui sont des regroupements d'*UIElement* réutilisables au sein de plusieurs *UIUnit* différents. L'ensemble de ces éléments (*UIGroup*, *UIUnit*, *UISubUnit* et *UIElement*) sont prévus pour être réutilisables par plusieurs applications permettant ainsi d'envisager la création d'applications par simple assemblage d'éléments existants. Ainsi, un *UIUnit* contenant un menu pourra être utilisé par plusieurs *UIGroup* différents. De même, un *UIUnitSuite* contenant une succession d'*UIUnit* permettant de s'identifier dans une application pourra être utilisé par plusieurs applications différentes.

Dans le cadre de PERCOMOM, chaque *UIElement* possède un certain nombre de fonctions standard qui peuvent être appelées à travers des modèles d'actions (modèles de type BM1). Chaque fonction standard permet d'effectuer une tâche bien précise sur l'*UIElement* qui permet de modifier le contenu de l'*UIElement*. Le Tableau 3.5 fournit quelques exemples de fonctions standard qui sont associées à un élément d'interaction de type *UIFieldStaticInOut*.

Tableau 3.5. Fonctions standard associées à un *UIFieldStaticInOut* dans PERCOMOM

Fonction standard	Description
RechargerContenu	Permet de remettre à jour le contenu de l'élément d'interaction *UIFieldStaticInOut* en fonction des dernières informations manipulées par le système.
EffacerContenu	Efface le contenu affiché par *l'UIFieldStaticInOut.*
SélectionnerSuivant	Permet, dans une liste, de sélectionner et d'afficher l'élément suivant.
SélectionnerPrécédent	Permet, dans une liste, de sélectionner et d'afficher l'élément précédent.
SélectionnerPremier	Permet, dans une liste, de sélectionner et d'afficher le premier élément de la liste.
SélectionnerDernier	Permet, dans une liste, de sélectionner et d'afficher le dernier élément de la liste.

Les *UIElement* disposent aussi de propriétés qui permettent de préciser leur comportement ainsi que, si nécessaire, leur lien avec l'ontologie du domaine. À titre d'exemple, le Tableau 3.6 fournit une liste des principales propriétés d'un élément d'interaction de type *UIField* et de tous ses descendants.

Un exemple d'association de ces propriétés à un élément d'interaction de type *UIFieldStatic* est donné à la Figure 3.16. Il est important de noter à ce niveau que toutes les propriétés sont rassemblées dans un artefact spécifique au type d'élément d'interaction manipulé. Ainsi, un *UIElement* de type *UIFieldStaticInOut* se verra associer un artefact de type "*UIFieldStaticInOut_Artefact*", un *UIGroup* se verra associer un artefact de type "*UIGroup_Artefact*", etc.

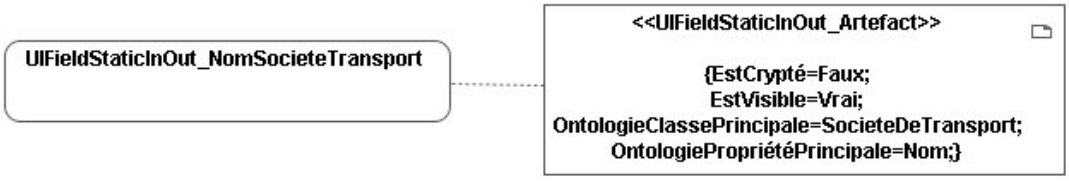


Figure 3.16. Exemple d'association de propriétés à un élément d'interaction de type *UIFieldStatic* dans un modèle statique d'interaction (IM2)

Tableau 3.6. Les principales propriétés associées à un élément d'interaction de type *UIField* et à ses descendants

Propriété	Type	Description
EstCrypté	Optionnel	Booléen permettant d'indiquer si l'*UIField* apparaît ou pas crypté à l'écran (c'est-à-dire lisible ou pas). Si la valeur n'est pas indiquée, c'est la valeur de l'*UIElement* supérieur qui contient l'*UIField* qui est utilisée (en principe un *UIUnit*) et si celui-ci ne contient pas non plus de valeur, on remonte dans la hiérarchie jusqu'à trouver une valeur.
EstVisible	Optionnel	Booléen permettant d'indiquer si l'*UIField* est visible ou pas à l'écran. Si la valeur n'est pas indiquée, c'est la valeur de l'*UIElement* supérieur qui contient l'*UIField* qui est utilisée (en principe un *UIUnit*) et si celui-ci ne contient pas non plus de valeur, on continue à remonter dans la hiérarchie jusqu'à trouver une valeur.
OntologieClassePrincipale	Optionnel	Permet d'indiquer la classe de l'ontologie de domaine à laquelle est rattaché l'*UIField*. Ceci permet d'affecter de manière automatique une valeur à l'*UIField* en fonction des individus de l'ontologie associés au niveau de l'*UIUnit* au moment de l'affichage de l'*UIField*.
OntologiePropriétéPrincipale	Optionnel	Permet d'indiquer la propriété de la classe de l'ontologie à laquelle est rattaché l'*UIField*.
OntologieNonLié	Optionnel	Permet d'indiquer que l'*UIFieldStatic* n'est pas relié, du point de vue de l'ontologie, à l'*UIUnit* qui le contient. La valeur par défaut est égale à "Faux".
CommentaireLibre	Optionnel	Permet à l'expert métier d'ajouter un commentaire textuel sur l'*UIElement* pour apporter des précisions qu'il jugerait utiles.

Au niveau des propriétés associées à un *UIElement*, celles-ci sont modifiables dans les modèles d'action (BM1), ce qui permet, par exemple, à l'aide d'une action (BM1) de rendre un *UIElement* visible ou invisible. L'association avec une classe de l'ontologie permet d'indiquer le contenu à afficher dans l'*UIElement* à partir des données associées à l'*UIUnit* contenant l'*UIElement* au moment ou l'*UIElement* est affiché ; ces données étant représentées par un ou plusieurs individus de l'ontologie de domaine (OD). Ainsi, si on reprend l'exemple de la Figure 3.16, si l'*UIUnit* contenant l'*UIFieldStatic* "*UIFieldStaticInOut_NomSociétéTransport*" manipule plusieurs individus différents de l'ontologie de type *SocieteDeTransport*, alors l'*UIFieldStatic* affichera une liste de valeurs correspondant à l'ensemble des valeurs de la propriété Nom de chaque individu. Par contre, si l'*UIUnit* ne manipule qu'un seul individu, alors l'*UIFieldStatic* affichera une seule valeur égale à la valeur de la propriété nom de l'individu manipulé. Maintenant, si l'*UIUnit* est associé à une autre classe de l'ontologie, alors l'*UIFieldStaticInOut* affichera la propriété *Nom* de l'ensemble des individus de la classe *SociétéDeTransport* qui ont une relation avec l'individu ou les individus manipulés par l'*UIUnit*. Enfin, si la valeur de la propriété *OntologieNonlié* est égale à "Vrai" alors l'*UIFieldStatic* affichera, sous la forme d'une liste, toutes les valeurs possibles pour la propriété *Nom* de la classe de l'ontologie *SociétéDeTransport*.

Dans PERCOMOM, le lien entre les modèles statiques d'interaction et les modèles de processus métier se fait à travers l'utilisation d'un formalisme particulier au niveau de la définition des noms des différentes tâches. Par exemple, supposons qu'on définisse un processus métier de login à travers un *UIUnitSuite* en utilisant deux *UIUnit* différents (cf. Figure 3.17) :

- Un *UIUnit* permettant à l'utilisateur d'entrer son identifiant et son mot de passe et affichant un message d'erreur dans le cas où l'identification est incorrecte (cf. Figure 3.17 a et b).
- Un *UIUnit* permettant d'afficher à l'utilisateur un message général dans le cas où l'identification est correcte (cf. Figure 3.17 c).

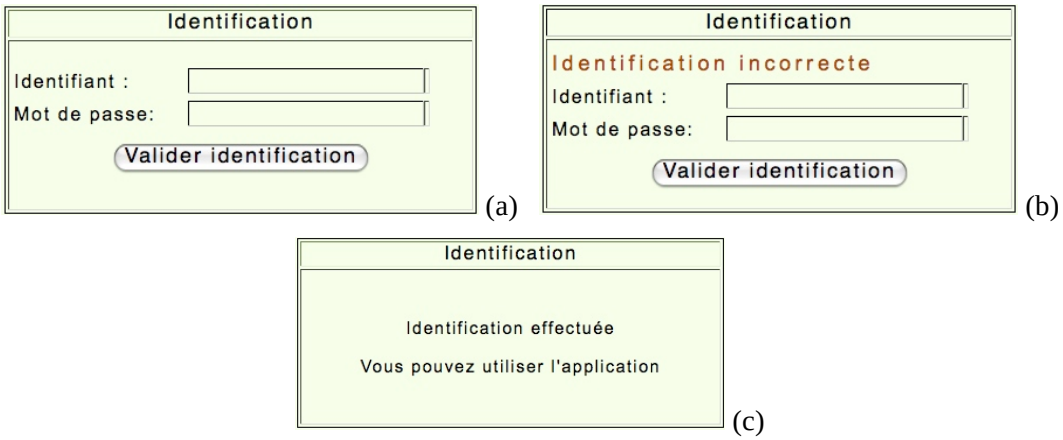


Figure 3.17. Exemple de fenêtres pouvant être associées à un processus d'identification (fenêtres issues du niveau PSM)

L'*UIUnit* associé aux copies d'écran (a) et (b) de la Figure 3.17 pourrait alors ressembler à l'exemple donné à la Figure 3.18 :

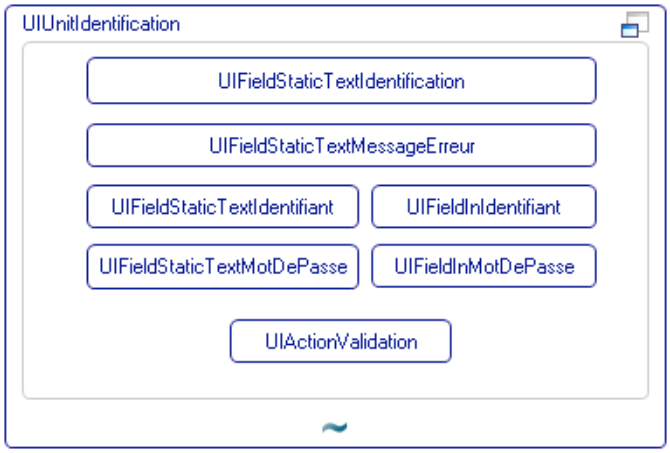


Figure 3.18. Exemple de contenu d'un UIUnit associé à un processus d'identification de l'utilisateur (niveau CIM)

On pourra noter que l'ensemble des éléments d'interaction pouvant apparaître au niveau de l'interface sont représentés sous la forme d'*UIElement*. Les éléments de type *UIFieldStatic* permettent de prendre en compte des champs texte non modifiables par l'utilisateur comme, par exemple, des titres. Les éléments *UIFieldInOut* représentent des champs de saisie. Quant au bouton de validation, il est représenté par un élément de type *UIFieldAction* portant le nom de "UIActionValidation". Pour des questions de lisibilité, les propriétés associées à chaque type d'élément n'ont pas été représentées sur la figure, mais chaque élément possède un ensemble de propriétés qui lui sont propres et qui permettent de le caractériser.

Le modèle statique de la Figure 3.18 est ensuite rattaché à un processus métier "*ProcessusIdentification*" dans lequel on décrit l'ensemble des interactions pouvant être réalisées (cf. Figure 3.19). Dans la définition de ce processus, le formalisme utilisé est le suivant ;

- Le mot-clé "*Ouvrir*" permet d'indiquer qu'on va afficher à l'utilisateur l'UIUnit correspondant et qu'on va éventuellement effectuer les actions qui sont associées à cette ouverture.
- Le mot clé "*Choix Utilisateur*" permet d'indiquer que le choix de la tâche suivante est effectué par l'utilisateur.
- Le mot clé "*Entrer*" permet d'indiquer que l'utilisateur peut entrer une valeur dans le champ indiqué en référence.

- Le mot clé "*Action*" permet d'indiquer qu'un élément de type *UIFieldAction* vient d'être sélectionné par l'utilisateur.
- Le mot clé "*Résultat Action*" permet d'indiquer que le résultat de l'action qui vient d'être exécutée va être utilisé dans le cadre du choix suivant.
- Le mot clé "*Fermer*" permet d'indiquer qu'on va fermer l'*UIUnit* correspondant et qu'on va éventuellement effectuer les actions qui sont associées à cette fermeture.

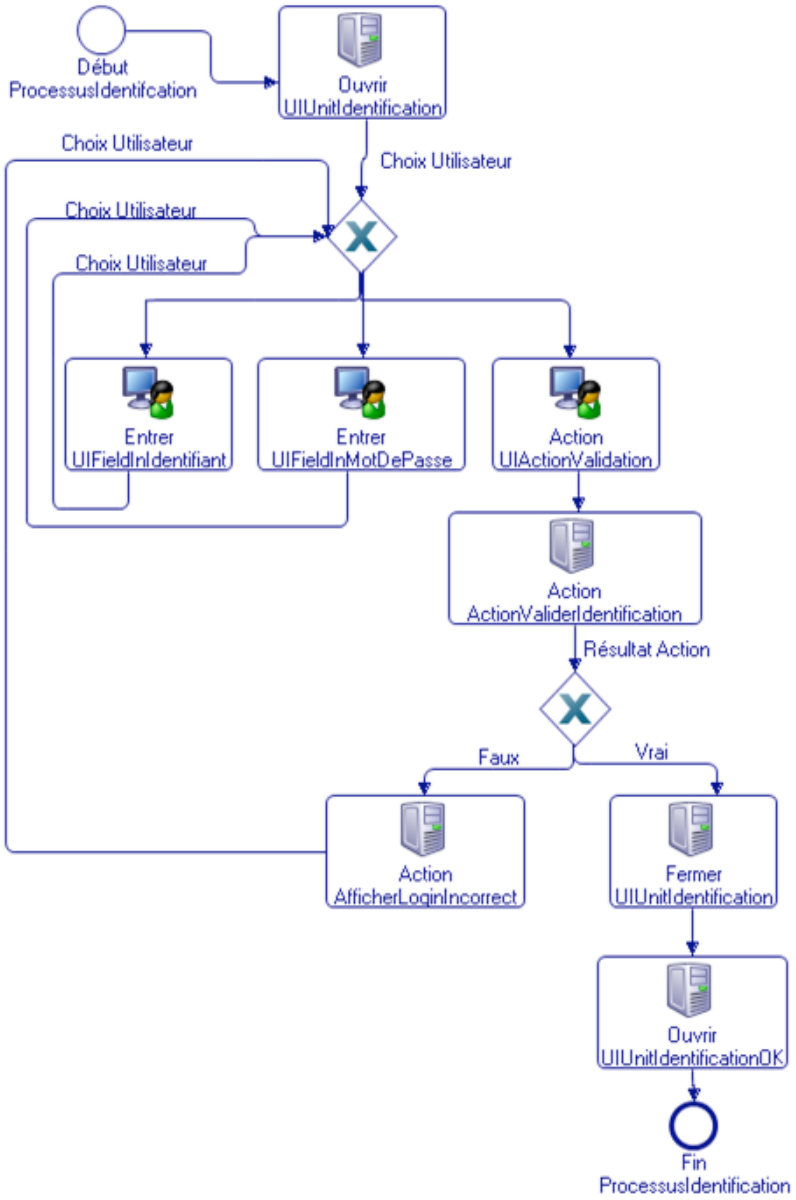


Figure 3.19. Exemple de processus métier (modèle IM1) pouvant être associé à l'*UIUnit* (modèle IM2) de la Figure 3.18

Le formalisme étant assez riche, nous nous limiterons dans cette partie à la présentation de cet exemple assez simple. D'autres exemples plus complexes seront présentés en détail dans les chapitres 5 et 6 de cette thèse.

Les processus métier n'utilisant pas systématiquement l'ensemble des possibilités d'interaction des *UIGroup* ou *UIUnit* manipulés et ces deux types d'éléments ne disposant pas de modèles de tâche directement associés, leur validation peut être difficile. Pour résoudre ce problème, nous avons développé, dans le cadre de notre méthode, un modèle de synthèse que nous avons appelé modèle dynamique d'interaction.

3.1.4.3 Le modèle dynamique d'interaction (IM3)

Dans PERCOMOM, le modèle dynamique d'interaction n'est pas un modèle à proprement parler dans le sens où ce n'est pas l'utilisateur qui le crée mais l'outil de modélisation lui-même. En fait, le modèle dynamique d'interaction reprend, pour un *UIUnit* déterminé et pour une *UIApplication* déterminée, toutes les interactions définies au niveau des processus métier utilisant cet *UIUnit*. Du point de vue de l'approche globale, le modèle dynamique d'interaction représente un processus métier particulier limité au niveau d'un *UIUnit* bien déterminé et à partir duquel on pourrait réaliser toutes les interactions possibles au niveau de l'*UIUnit*. De cette manière, il devient alors possible de valider l'ensemble des interactions définie pour un *UIUnit* déterminé de la même manière qu'il est possible de valider un processus métier. La Figure 3.20 présente un exemple de création d'un modèle dynamique des interactions (IM3) pour un *UIUnit* déterminé portant le nom AA à partir de trois processus métier différents (processus A, B et C) utilisant cet *UIUnit*.

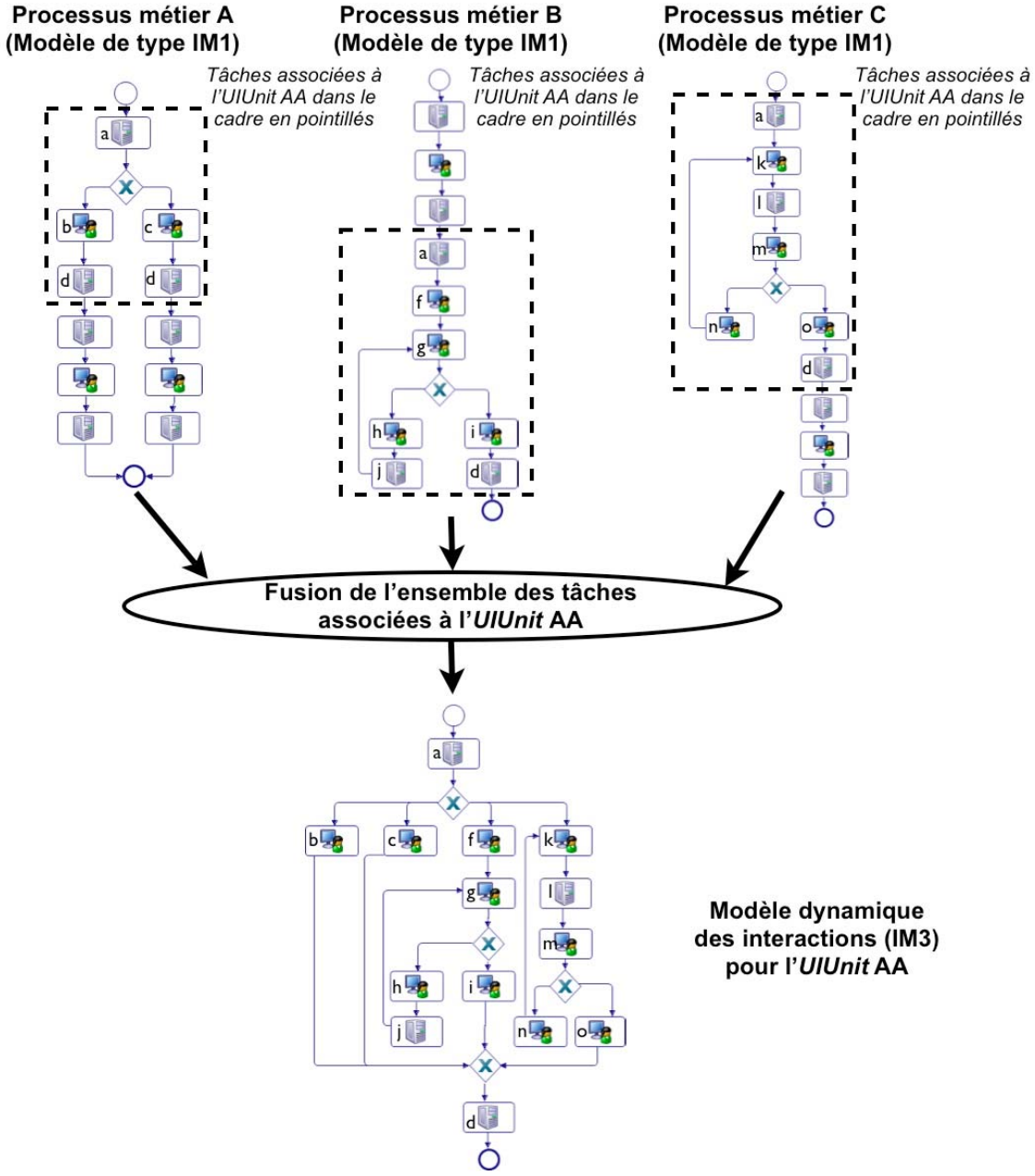


Figure 3.20. Principe de création d'un modèle dynamique des interactions pour un *UIUnit* AA à partir de trois processus métier différents utilisant cet *UIUnit* (afin de faciliter la lecture de la figure, chaque tâche de l'*UIUnit* AA a été identifiée par une lettre de l'alphabet)

Dans le cadre de la validation des applications créées à l'aide de la méthode PERCOMOM, les modèles dynamiques d'interaction représentent un élément important pour la validation des IHM. Ils permettent en effet de détecter :

- Les éléments d'interaction non utilisés dans une application.
- Les éléments d'interaction utilisés avec un sens métier différents (exemple : réalisation d'actions différentes dans deux processus métier pour un même élément de type *UIFieldAction* sur un même *UIUnit*).
- Les éléments d'interaction redondants sur un même *UIUnit* (exemple : deux éléments de type *UIFieldAction* ne portant pas le même nom mais réalisant la même chose dans des processus métier différents).

Dans le cadre de nos travaux, nous n'avons pas encore réalisé, par manque de temps, d'outils permettant de créer et de valider ces modèles d'interaction. Ces modèles sont donc, pour l'instant, validés de manière manuelle.

Dans le cadre d'une approche dirigée par les modèles, la validation de l'ensemble des modèles au niveau conceptuel est un élément essentiel à prendre en compte. En effet, sans cette validation, les problèmes de conception ne pourraient être détectés que tard dans le processus de développement et donc seraient coûteux à corriger. Le modèle dynamique d'interaction permet de prendre en compte le fait que les IHM servent de support à des processus métier et qu'en conséquence, il convient de les valider en les mettant dans leur contexte d'usage. En ce qui concerne les problématiques liées à l'ergonomie des IHM et/ou à l'apparence physique des IHM, celles-ci étant dépendantes, en partie, des interfaces physiques utilisées pour les visualiser, elles ne peuvent pas être évaluées au niveau conceptuel. En fait, leur évaluation sera effectuée à travers les niveaux PIM et PSM suivant des règles qui seront présentées dans le chapitre 3, §3.2.

Au niveau des modèles dynamiques d'interaction, afin de pouvoir prendre en compte les interactions entre *UIUnit*, il est possible de définir des modèles de validation au niveau des *UIGroup*. Ces modèles seront, par définition, un assemblage de modèles dynamiques d'interaction d'*UIUnit* dans lesquelles les liens entre *UIUnit* seront visuellement apparents.

3.1.4.4 Synthèse et discussion

Dans le cadre de la méthode PERCOMOM, les deux modèles conceptuels principaux liés à la modélisation des IHM sont le modèle des processus métier (IM1) et le modèle statique d'interaction (IM2). Le modèle dynamique d'interaction est utilisé, pour sa part, que pour la validation, au niveau conceptuel, des deux autres modèles. Par rapport aux autres approches de modélisation conceptuelle des IHM comme, par exemple, UsiXML, PERCOMOM présente un certain nombre de points communs mais aussi des différences marquées.

Ainsi, pour le modèle dynamique d'interaction, représenté par le modèle de processus métier (IM1) dans PERCOMOM, le choix d'utiliser une notation comme BPMN plutôt qu'un modèle de tâches plus classique comme CTT (Concurrent Task Trees) permet, à travers un seul modèle, de pouvoir prendre en compte aussi bien des tâches interactives que des tâches non interactives. Il permet aussi d'introduire la notion d'interaction comme faisant pleinement partie des processus métier liés à l'application. De cette manière, l'utilisateur est placé au centre des processus métier et n'est plus seulement un intervenant situé à la marge. A ce titre, il est intéressant de noter que cette logique de mettre l'utilisateur au centre des processus métier se retrouve aussi dans les approches proposées par les langages comme BPEL4People (Business Process Execution Language for People) dont l'objectif est d'introduire l'utilisateur comme un élément clé de l'exécution d'un processus métier.

A côté de ce modèle de processus métier, si le modèle statique d'IHM est similaire dans son approche globale, PERCOMOM propose une approche différente pour le regroupement logique des éléments d'interaction. Ainsi, ceux-ci sont groupés du point de vue de la logique métier et pas uniquement du point de vue de l'IHM. Ceci s'explique par le fait que le modèle de tâche associé au modèle statique d'IHM n'est pas un modèle purement lié à l'IHM mais un modèle métier centré sur les buts métier de l'utilisateur. Ainsi, PERCOMOM ne propose pas des modèles centrés sur l'IHM mais des modèles centrés sur le métier où l'IHM vient en support des objectifs métier des utilisateurs.

3.1.5 Synthèse et conclusion

Dans l'approche PERCOMOM, la modélisation conceptuelle de niveau CIM représente un élément essentiel de l'approche dans le sens où c'est à partir de cette modélisation que vont être générées les applications interactives.

Les modèles conceptuels étant des modèles totalement indépendants des technologies, PERCOMOM propose, à travers une vision holistique des applications, une approche de modélisation centrée sur les besoins métier et pas uniquement sur les besoins en matière de génération d'applications informatiques. Cette vision métier, permet d'envisager, à terme, une prise en charge de la modélisation conceptuelle de niveau CIM par des experts métier qui sont les personnes les plus à même de modéliser, de manière optimale, les besoins métiers associés à une application. Or, qui dit expertise métier dit aussi spécialisation métier ; ce qui signifie qu'un seul expert métier ne pourra, en théorie, pas modéliser l'ensemble d'une application mais uniquement certaines parties bien précises sur lesquelles il dispose d'une expertise.

C'est en partant de cette constatation, ou plutôt de cette hypothèse, que nous avons proposé de modéliser une application au niveau CIM à l'aide de plusieurs familles de modèles conceptuels différents. Ces familles, et les modèles qu'elles contiennent, ont été définis de manière à avoir :

- Un minimum de modèles,
- Une segmentation claire entre les différents modèles,
- Des modèles adaptés à des expertises métier bien précises.

D'où la création de quatre familles de modèles :

- Une famille de trois modèles (SM1, SM2 et SM3) permettant de prendre en charge les problématiques liées aux utilisateurs et à leur environnement social.
- Une famille de trois modèles (EM1, EM2 et EM3) permettant de prendre en charge les problématiques liées à l'environnement dans lequel l'application va être utilisée.
- Une famille de trois modèles (BM1, BM2 et BM3) permettant de prendre en charge les problématiques liées à des aspects comportementaux en rapport avec l'application
- Une famille de trois modèles (IM1, IM2 et IM3) permettant de prendre en charge les problématiques liées aux IHM.

Mais aussi la création de deux modèles plus généraux :

- Le modèle d'application permettant de prendre en compte l'ensemble des éléments spécifiques à l'application elle-même.
- L'ontologie de domaine servant de base commune à l'ensemble des modèles mais surtout permettant de modéliser l'ensemble du domaine métier du point de vue des concepts métier manipulés.

Tous ces modèles, mis à part le modèle d'application, sont conçus pour être facilement réutilisables dans d'autres applications permettant d'envisager la création de nouvelles applications par simple réutilisation de modèles déjà existants.

Nos travaux ayant principalement été centrés sur la prise en compte des IHM, les modèles d'interaction IM1 et IM2 ont particulièrement été développés dans l'optique de permettre la génération semi-automatique d'applications interactives de type WIMP (Window, Icon, Mouse, Pointing device). Les autres modèles ont pour leur part été développés, sous une forme basique, de manière à servir de support aux modèles d'interaction.

L'ensemble de ces modèles étant à destination des experts métier, leur nombre et leur contenu seront susceptibles d'évoluer dans le temps en fonction en fonction des résultats des tests de la méthode auprès d'experts métier.

Un des objectifs de la modélisation conceptuelle est de permettre de « simplifier » au maximum, tout en restant complet, la modélisation d'une application. Aussi, nous allons maintenant présenter un autre élément important de notre approche qui permet de faire cette simplification : l'architecture spécifique de type MDA de PERCOMOM permettant de générer de manière semi-automatique les applications à partir des modèles conceptuels.

3.2 L'architecture de type MDA utilisée par PERCOMOM (niveaux CIM, PIM et PSM)

3.2.1 Introduction

Dans le cadre des approches de type IDM, le passage par les niveaux PIM et PSM doit se faire, en théorie, de manière totalement automatique. Cela entraîne un certain nombre d'avantages mais aussi de nombreux inconvénients qui sont repris dans le Tableau 3.7.

Tableau 3.7. Principaux avantages et inconvénients des approches de type IDM lors du passage du niveau CIM au niveau PSM

Avantages	Inconvénients
• La génération étant automatique, il y a, en théorie, correspondance totale entre les applications créées et les modèles conceptuels. • Les modèles conceptuels créés au niveau CIM sont totalement indépendants des plateformes techniques réelles utilisées par les applications (implique la définition et l'utilisation d'un modèle de plateforme au niveau PIM pour prendre en compte les caractéristiques spécifiques de chaque type de plateforme d'application). • La réutilisation des modèles permet de fiabiliser les développements à travers la réutilisation de modèles éprouvés ; ce qui permet aussi d'accélérer les développements et donc de diminuer les coûts.	• Toute l'application doit être intégralement définie au niveau CIM car les modèles PIM et PSM ne sont obtenus que par transformation de modèles. • Toutes les interactions possibles et toutes les adaptations à l'environnement ne peuvent être définies qu'au niveau CIM. • Il n'existe pas de notion d'ensemble de fonctionnalités réutilisables dans les modèles mis à part à travers une réutilisation définie, au niveau CIM, à travers l'intégration de modèles.

Dans le cadre de PERCOMOM, afin d'avoir une approche dirigée par les modèles aussi efficiente que possible, nous avons développé une architecture globale permettant de limiter les inconvénients liés aux approches IDM. Ainsi, pour passer du niveau CIM à l'application concrète directement manipulable par l'utilisateur final, nous utilisons une architecture à trois niveaux (cf. Figure 3.21) de type MDA que nous avons enrichie au niveau PIM (cf. chapitre 3.2.2) et au niveau PSM (cf. chapitre 3.2.3).

Dans la figure 3.21, il est intéressant de noter l'utilisation de bases de règles externes au niveau des frameworks de niveau PIM et de niveau PSM. Ces bases de règles permettent de sortir une partie de la logique fonctionnelle et technique des frameworks de niveau PIM et de niveau PSM de manière à rendre ceux-ci adaptables sans avoir besoin de modifier le contenu des différents frameworks. Dans l'état actuel de nos travaux, les contenus des différentes bases de règles sont basés sur les principes du moteur de règles présenté dans l'annexe A de ce mémoire. Par contre, nous ne disposons pas pour l'instant d'une définition stricte des éléments de type règles qui doivent être définis ou pas dans le moteur de règles.

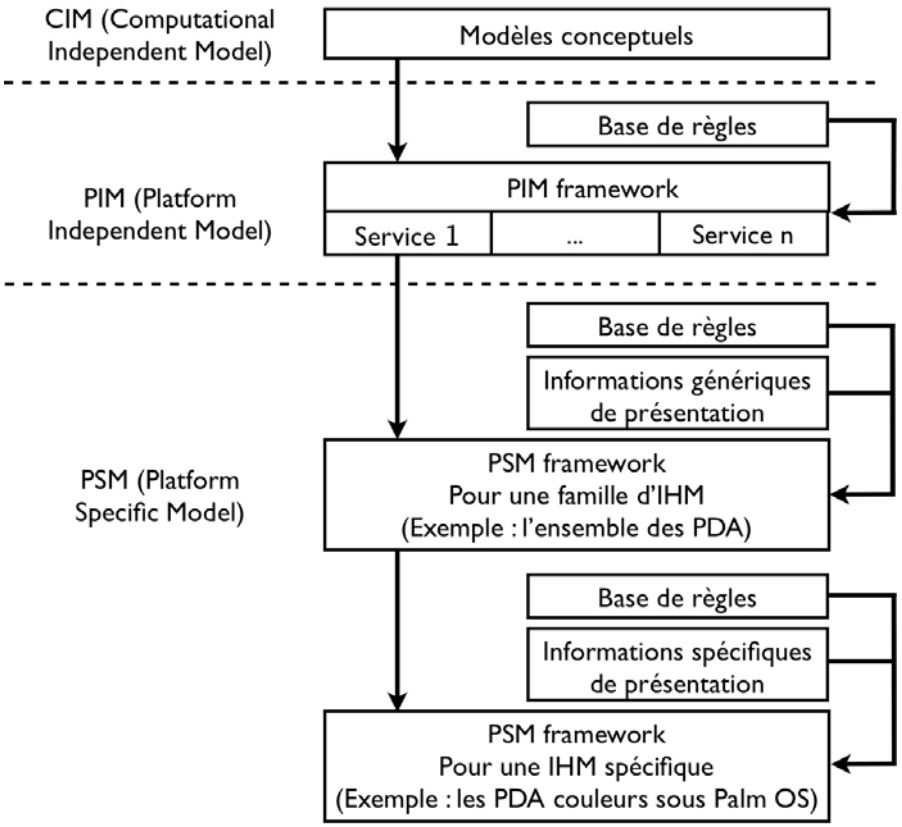


Figure 3.21. L'architecture globale à base de frameworks associée à la méthode PERCOMOM

Une des particularités du niveau PIM de notre architecture réside dans le fait que le framework associé contient un ensemble de services fonctionnels utilisables dans les modèles de niveau CIM. Cette notion de service fonctionnel étant spécifique à notre approche, nous allons la présenter en détail dans la partie suivante.

3.2.2 Le niveau PIM : un niveau composé de services fonctionnels

Dans PERCOMOM, un service fonctionnel de niveau PIM représente une fonctionnalité métier qu'il est possible d'appeler au niveau CIM sans en connaître le fonctionnement propre ; ce dernier étant défini au niveau PIM de l'architecture PERCOMOM.

Pour donner un exemple, la gestion des informations linguistiques, et plus précisément l'adaptation des éléments d'interaction à la langue de l'utilisateur, est un service fonctionnel de niveau PIM qui peut être utilisé au niveau CIM mais qui n'a pas besoin d'y être modélisé si on se place du point de vue métier. Ainsi, au niveau de la modélisation conceptuelle d'une application, il n'est pas important, du point de vue métier, de savoir comment fonctionne cette adaptation linguistique mais d'indiquer dans quel processus métier elle sera utilisée. De cette manière, il devient possible de simplifier les modèles conceptuels en permettant aux personnes en charge de la modélisation de se concentrer uniquement sur les buts et processus métier et pas sur les fonctionnalités annexes de support.

Dans PERCOMOM, un service fonctionnel de niveau PIM présente les particularités suivantes :

- Son fonctionnement est entièrement défini au niveau PIM de l'architecture PERCOMOM et pas au niveau CIM. Ainsi, du point de vue métier, il n'existe pas de modèle CIM pour un service fonctionnel de niveau PIM.
- C'est un ensemble de traitements permettant d'exécuter une opération précise sur l'élément auquel il est relié. Par exemple, si on associe le service fonctionnel d'adaptation linguistique des éléments d'interaction à un *UIFieldStatic*, l'adaptation linguistique ne sera effectuée que pour cet *UIFieldStatic*.

- C'est un ensemble de traitements qui n'est pas spécifique à une application particulière et qui peut se retrouver dans un nombre important d'applications. Par exemple, l'adaptation linguistique des éléments d'interaction n'est pas spécifique à une seule application mais peut concerner, *a priori*, toutes les applications.
- C'est un ensemble de traitements qui sont effectués de manière automatique sans aucune interaction avec l'utilisateur. Ainsi, l'utilisateur peut, au niveau d'une application, éventuellement définir s'il souhaite ou pas avoir une adaptation linguistique au niveau de l'application mais il ne peut pas définir où cette adaptation sera effectuée ; ceci étant effectué dans le modèle statique des interactions (IM2) de niveau CIM.
- Enfin, un service fonctionnel de niveau PIM peut être attaché à n'importe quel élément d'interaction de niveau CIM. Par exemple, l'adaptation des éléments d'interaction à la langue de l'utilisateur peut se faire uniquement au niveau d'un champ texte statique, un *UIFieldStatic*, bien précis dans un *UIGroup* et dans ce cas on l'associe à l'*UIFieldStatic* ou alors sur tous les éléments d'interactions contenus dans l'*UIGroup* et dans ce cas, on l'associe à l'*UIGroup*. La Figure 3.22 présente un exemple d'association d'un service fonctionnel de niveau PIM au niveau d'un modèle IM2 de niveau CIM pour un *UIElement* de type *UIFieldStatic* et pour un *UIElement* de type *UIGroup*.

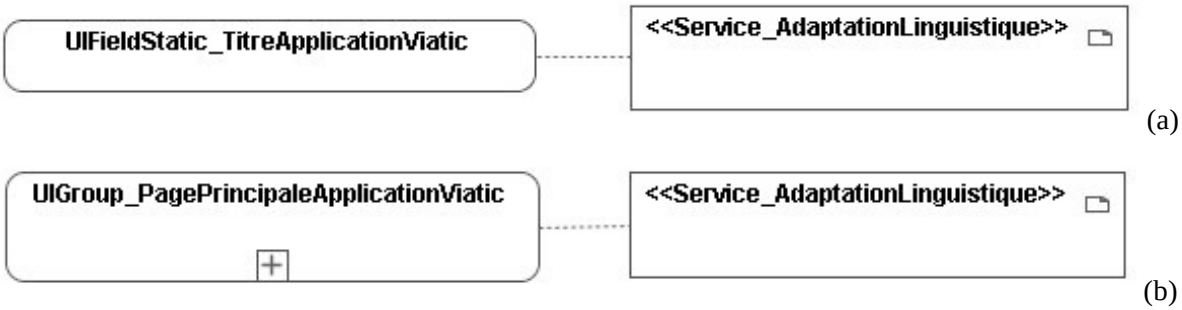


Figure 3.22 Exemple d'association d'un service fonctionnel d'adaptation linguistique de niveau PIM dans des modèles IM2 ne niveau CIM, pour un *UIElement* de type *UIFieldStatic* (a) et pour un *UIElement* de type *UIGroup* (b)

Par défaut, PERCOMOM ne propose qu'un nombre limité de services fonctionnels de niveau PIM principalement pour gérer la personnalisation et l'adaptation des contenus et pour gérer les sessions des utilisateurs ; une session étant l'ensemble des éléments relatifs à l'exécution d'une application pour un utilisateur donné. Ce dernier point ne sera pas présenté dans le cadre de ce mémoire. Par contre, il est possible d'ajouter par la suite de nouveaux services fonctionnels de niveau PIM, en fonction des besoins métier et en fonction du contexte métier. Dans ce cadre, pour qu'une opération ou qu'un ensemble d'opérations devienne un service fonctionnel de niveau PIM, il faut que :

- Il ne fasse pas obligatoirement partie d'un processus métier (modèle IM1) sinon c'est une action (modèle BM1). Dans PERCOMOM, tout ce qui est défini dans un processus métier (modèle IM1) doit pouvoir se retrouver dans un autre modèle conceptuel de l'application. Ainsi, si on souhaite effectuer une adaptation linguistique des éléments d'interaction dans le cadre d'un processus métier, on devra définir une action (modèle BM1) pour effectuer cette adaptation. Dans cette action (BM1), on devra décrire l'ensemble des étapes à effectuer pour réaliser cette adaptation.
- Il puisse être associé à un ou plusieurs éléments d'interaction. Dans PERCOMOM, un service fonctionnel de niveau PIM est appelé automatiquement au chargement ou lors de la modification du contenu d'un élément d'interaction. Aussi, tout ensemble de traitements qui ne peut être associé à un élément d'interaction ne peut être transformé en service fonctionnel de niveau PIM.
- Il soit susceptible d'être utilisé par plusieurs applications. Ainsi, si un ensemble d'opérations n'est utilisé que par une application, il n'est pas forcément pertinent de le transformer en service fonctionnel ; la notion de service fonctionnel étant surtout utile dans un contexte de réutilisation.

Il est important de noter ici que la définition et la création de services fonctionnels de niveau PIM ne sont pas obligatoires dans le cadre de la méthode PERCOMOM mais sont fortement conseillées. En effet, les services fonctionnels de niveau PIM :

- Favorisent, par leur principe même de réutilisation, la fiabilité des modèles : un service fonctionnel testé et validé dans le cadre d'une première application peut être directement utilisé dans une deuxième application sans avoir à subir une phase de test et de validation importante.
- Favorisent, la simplification des modèles conceptuels : l'utilisation d'un service fonctionnel de niveau PIM se faisant à travers une simple association dans les modèles de niveau CIM.
- Favorisent la maintenance des applications : un service fonctionnel de niveau PIM étant défini à un seul endroit, toute modification de ce service fonctionnel de niveau PIM est automatiquement prise en compte par l'ensemble des applications qui l'utilisent.

Au niveau de leur nature même, les services fonctionnels peuvent être classés en deux catégories :

- Les services explicites qui peuvent être appelés directement au niveau des modèles en fonction des besoins de la personne en charge de la modélisation comme, par exemple, la personnalisation des contenus. Pour information, ce type de personnalisation correspond à une adaptation des contenus d'informations retournés à l'utilisateur en fonction de son profil, de ses préférences ou encore en fonction de recommandations issues d'une communauté d'utilisateurs ayant un profil assez similaire à celui de l'utilisateur. On pourra donner, comme exemple de personnalisation, celle des contenus effectuée sur un site web commercial comme celui de la société Amazon (http://www.amazon.fr) (cf. chapitre 2, §2.1.3).
- Les services implicites qui sont automatiquement appelés par les applications comme, par exemple, la gestion des sessions utilisateurs. La définition de l'utilisation ou non des services implicites se fera au niveau PIM à travers la définition de règles d'utilisation de ces services.

De manière pratique, le processus de création d'un nouveau service fonctionnel de niveau PIM est décrit dans la Figure 3.23.

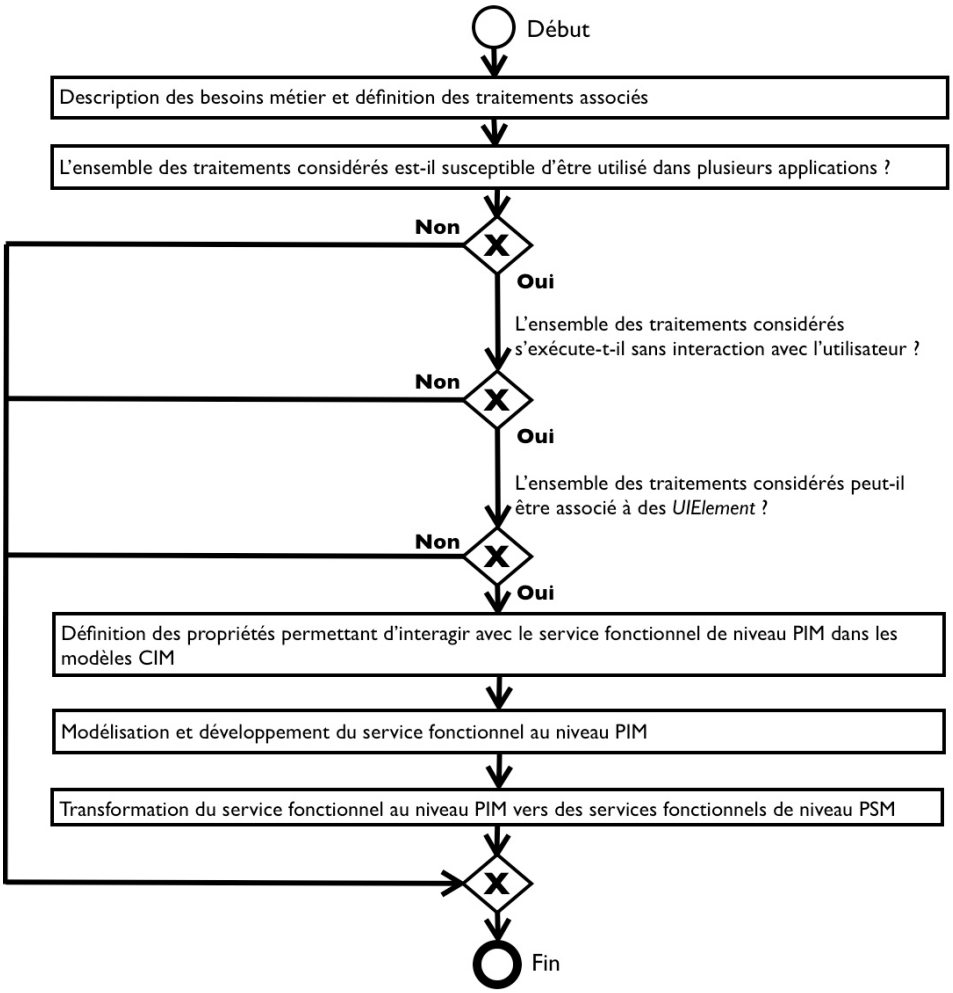


Figure 3.23. Procesus de création d'un nouveau service fonctionnel de niveau PIM dans PERCOMOM

Si le contenu et le fonctionnement des services fonctionnels de niveau PIM sont définis avec les experts métier, leur orientation technique implique une modélisation et un développement par des informaticiens (passage par des modèles de type UML de niveau PIM qui est transformé ensuite en modèles de niveau PSM). Dans le cadre de la création d'un nouveau service fonctionnel de niveau PIM, une des étapes importantes consiste à définir les propriétés fonctionnelles permettant, dans les modèles de niveau CIM, d'interagir avec le service fonctionnel de niveau PIM. Pour donner un exemple, pour un service fonctionnel de niveau PIM associé à l'adaptation des éléments d'interaction, on pourrait définir les quatre propriétés du Tableau 3.8 (la définition est donnée ici à titre d'exemple, elle n'a pas pour prétention d'être exhaustive).

Un exemple d'utilisation des propriétés d'un service de niveau PIM dans un modèle IM2 de niveau CIM est donné dans la Figure 3.24. Dans cet exemple, on indique qu'on désire faire appel au service d'adaptation linguistique sur l'*UIGroup* "*UIGROUP_PagePrincipaleApplicationViatic*" et qu'on désire que la langue utilisée pour cette adaptation soit obligatoirement la langue française.

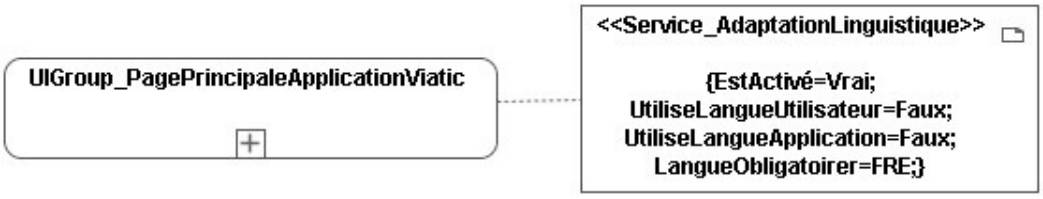


Figure 3.24 Exemple d'utilisation des propriétés d'un service fonctionnel de niveau PIM dans un modèle IM2 de niveau CIM

Tableau 3.8. Exemple de propriétés utilisables au niveau CIM pour un service fonctionnel d'adaptation linguistique de niveau PIM

Propriété	Type	Description
EstActivé	Obligatoire	Accepte deux valeurs (Vrai/Faux) et permet d'indiquer si on doit ou pas activer l'adaptation linguistique pour l'*UIElement* auquel est rattaché le service fonctionnel au niveau CIM. Dans le cas où aucune autre propriété n'est indiquée, l'adaptation linguistique se fait en fonction de langue par défaut définie au niveau du profil de l'utilisateur.
UtiliseLangueUtilisateur	Optionnelle	Accepte deux valeurs (Vrai/Faux) et permet d'indiquer si l'adaptation linguistique doit se faire en utilisant la langue par défaut définie au niveau du profil de l'utilisateur.
UtiliseLangueApplication	Optionnelle	Accepte deux valeurs (Vrai/Faux) et permet d'indiquer si l'adaptation linguistique doit se faire en utilisant la langue par défaut définie au niveau de l'application.
LangueObligatoire	Optionnelle	Accepte une seule valeur qui est le code ISO 639-2 sur trois positions de la langue dans laquelle le contenu associé à l'*UIElement* doit être affiché. Exemple : code "FRE" pour avoir un contenu affiché en français, code "ENG" pour avoir un contenu affiché en anglais ou encore code "GER" pour avoir un contenu affiché en allemand.

Dans le cadre du développement du service fonctionnel au niveau PIM, il faudra prendre garde à bien définir un comportement pour chaque type d'*UIElement* ainsi qu'une analyse fine des différentes interactions entre les propriétés permettant d'éviter des incohérences ou des blocages. Ainsi, dans le cadre du service fonctionnel d'adaptation linguistique, la propriété "*EstActivé*" est prioritaire par rapport à toutes les autres propriétés et, si sa valeur est égale à Non, les autres propriétés ne sont pas prises en compte. De même, la propriété "*UtiliseLangueUtilisateur"* est prioritaire par rapport à la propriété "*UtiliseLangueApplication"* qui est elle-même prioritaire par rapport à la propriété "*LangueObligatoire"*. L'ordre de traitement des propriétés et les relations entre les propriétés ne doivent pas être traités uniquement du point de vue technique mais principalement du point de vue métier. En effet, dans PERCOMOM, ce sont les besoins métier qui définissent le nombre et la nature de chaque propriété ainsi que la façon de les prendre en compte.

Afin de faciliter l'utilisation des services fonctionnels de niveau PIM, chaque service disposera de valeurs par défaut valables pour l'ensemble des applications et de valeurs par défaut valables spécifiquement pour une application ou un ensemble d'applications. De cette manière, au niveau des modèles de niveau CIM, on n'affichera que les propriétés qui ne correspondent pas aux valeurs par défaut afin d'en faciliter la lecture. Ainsi, si on définit pour une application que toutes les informations seront adaptées linguistiquement à la langue définie dans le profil de l'utilisateur, il ne sera plus nécessaire de l'indiquer dans les différents modèles d'interaction associés à cette application à moins qu'on veuille indiquer explicitement un autre type d'adaptation linguistique.

La définition des valeurs des propriétés des services fonctionnels de niveau PIM pouvant varier en fonction de règles métier prédéfinies ou de choix effectués dynamiquement par l'utilisateur, il est possible de définir pour un même élément d'interaction plusieurs ensembles de propriétés qui seront activés ou pas en fonction de l'évaluation des règles métier et/ou en fonction des choix de l'utilisateur. Afin d'assurer un fonctionnement cohérent des applications, notre méthode impose la création de valeurs par défaut qui seront sélectionnées dans le cas où aucune des règles ne serait valide. La Figure 3.25 présente un exemple de ce type d'adaptation pour un *UIElement* de type *UIGroup* sur lequel on effectuera une adaptation linguistique en fonction de la langue indiquée dans le profil de l'utilisateur lorsque celui-ci est identifié (évaluation de la règle métier de type validation "*estIdentifié*") et une adaptation linguistique à la langue française lorsqu'il ne l'est pas.

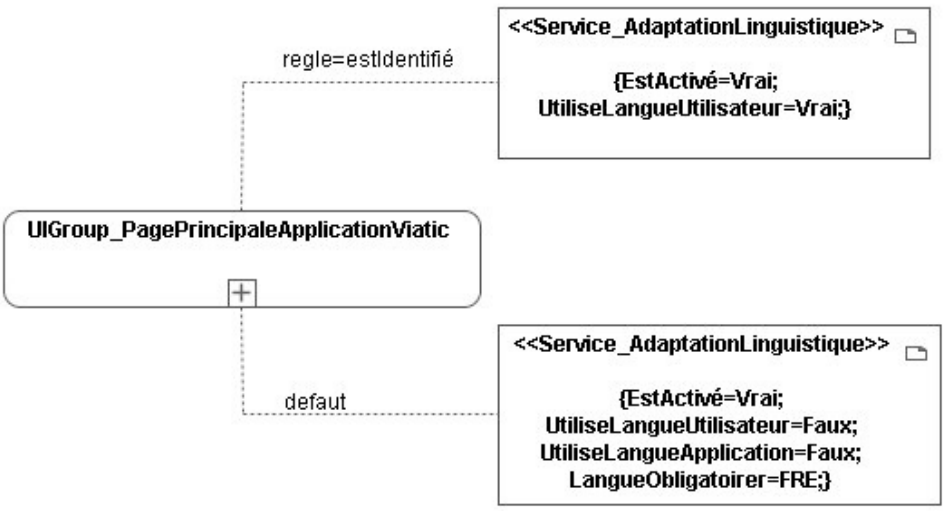


Figure 3.25. Exemple d'association de différents groupes de propriétés pour un même service fonctionnel de niveau PIM dans un modèle IM2 de niveau CIM (utilisation d'une règle métier pour savoir quel groupe de propriétés utiliser)

L'ensemble des *UIElement* faisant partie d'une hiérarchie d'éléments d'interaction, la définition d'une association d'un service fonctionnel de niveau PIM à un type d'*UIElement* entraîne automatiquement une notion d'héritage dans tous les éléments contenus dans cet *UIElement*. Ainsi, si on définit qu'un *UIGroup* est associé à une adaptation linguistique obligatoire à la langue française alors tous les éléments contenus dans cet *UIGroup* doivent utiliser cette même adaptation linguistique. Ainsi, tous les *UIUnit* contenus dans cet *UIGroup* auront aussi une adaptation automatique de leur contenu à la langue française ainsi que tous les *UIElement* contenus dans ces *UIUnit* et ainsi de suite. Dans le cas où un des éléments devant recevoir par héritage cette association et ces propriétés possède lui-même une association et des propriétés au niveau du service fonctionnel d'adaptation linguistique qui lui sont propres alors ce seront ces dernières qui seront utilisées ; les associations et propriétés existantes ne se substituant pas aux associations et propriétés déjà existantes.

Pour ce qui est du moteur de règles défini au niveau du framework de niveau PIM (cf. sa description détaillée en Annexe A), celui-ci est d'abord utilisé pour modifier le comportement des différents services en fonction du contexte, de l'application ou de caractéristiques de l'utilisateur. La définition des règles dans ce moteur de règles se base sur les mêmes principes que le moteur de règles métier de niveau CIM (modèle BM3). Ceci permet d'apporter une certaine souplesse au niveau du fonctionnement des services fonctionnels et techniques tout en fiabilisant les services eux-mêmes ; ceux-ci ne devant en principe plus être modifiés qu'à de très rares occasions. Pour donner un exemple, si on suppose qu'une contrainte technique impose d'utiliser des outils d'adaptation linguistique différents en fonction de la localisation géographique de l'utilisateur, on devra alors définir une règle permettant, au moment de l'appel du service fonctionnel d'adaptation linguistique d'indiquer quel outil d'adaptation linguistique utiliser. Ces règles pourront aussi être utilisées pour sortir une partie de la logique de traitement des traitements eux-mêmes afin de les rendre plus facilement modifiables. Ainsi, il serait possible d'envisager que la pondération des différentes propriétés du service fonctionnel d'adaptation linguistique soit effectuée à travers l'utilisation de règles ; ce qui permettrait de rapidement les modifier sans avoir besoin de modifier le service fonctionnel lui-même.

Dans le cadre de l'architecture PERCOMOM, le framework de niveau PIM est constitué, en premier lieu, des définitions de l'ensemble des services fonctionnels de niveau PIM que ceux-ci soit implicites ou explicites. Il est aussi constitué d'un framework. De manière plus générale, le framework de niveau PIM est utilisé lors de la transformation des modèles conceptuels de niveau CIM vers le niveau PIM afin d'enrichir ces modèles des différents éléments associés aux services fonctionnels de niveau PIM ; permettant ainsi d'avoir un modèle global de niveau PIM de l'application comprenant l'ensemble des fonctionnalités nécessaires. Les principes de la transformation de modèles du niveau CIM vers le niveau PIM seront détaillés dans le chapitre 3, §3.3.

3.2.3 Le niveau PSM : une architecture pour une génération semi-automatique des applications

Dans le cadre de l'architecture de PERCOMOM, le niveau PSM est constitué de deux niveaux comportant chacun un framework spécifique ; chaque framework représentant une adaptation du framework de niveau supérieur.

Ainsi, le premier niveau est utilisé pour caractériser une famille de plateformes d'IHM. Dans le cadre de notre approche, nous avons considéré qu'une famille de plateformes était composée d'un ensemble de plateformes disposant d'un certain nombre de caractéristiques communes. Celles-ci peuvent être physiques comme, par exemple, la taille de l'écran ou le nombre de couleurs affichables. Mais, elles peuvent aussi être fonctionnelles comme, par exemple, la façon d'afficher et de manipuler les listes déroulantes où encore la façon d'interagir avec l'utilisateur (utilisation d'un écran tactile ou d'un clavier par exemple). Par exemple, il serait possible d'envisager de définir une famille d'IHM pour l'ensemble des PDA qui disposent tous d'une taille d'écran limitée et de possibilités d'interactions avec un écran tactile.

L'utilisation du premier niveau de framework associé à une famille d'IHM permet d'effectuer une première série d'adaptations, génériques du point de vue de la plateforme finale, qui permettent de limiter les adaptations spécifiques à chaque plateforme finale. Ces dernières sont effectuées par le deuxième niveau de framework qui est spécifique à un type de plateforme bien précis comme, par exemple, les PDA couleur sous PALM OS 5.0. De cette manière, chaque plateforme spécifique de consultation peut être vue comme un bloc spécifique que l'on viendrait rattacher à une architecture déjà existante. Cette vision modulaire des plateformes techniques de consultation des informations, si elle peut paraître complexe au premier abord, permet de facilement envisager l'intégration de nouveaux types de plateformes dans le temps à travers l'ajout d'un nouveau "PSM framework pour une IHM spécifique". La Figure 3.26 présente un exemple de répartition de quelques plateformes techniques associées à des PDA et à des ordinateurs de type PC sur les deux niveaux de framework PSM.

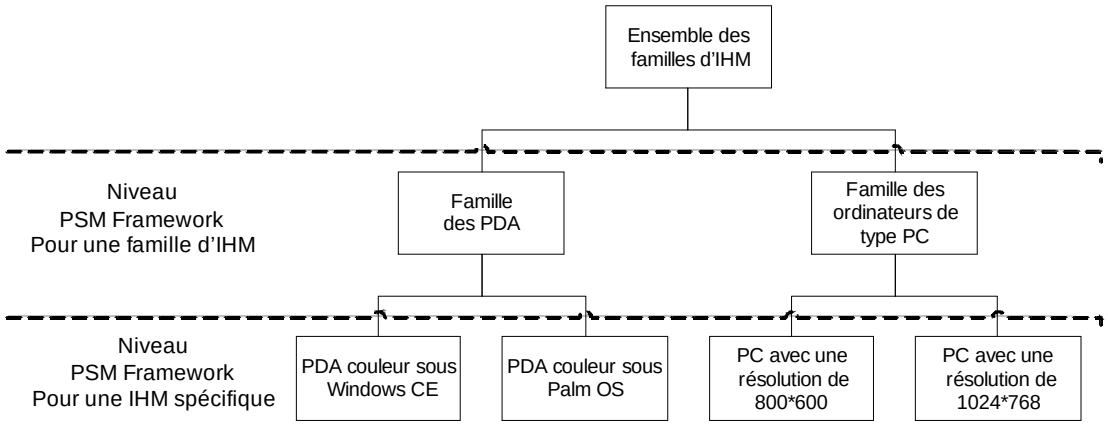


Figure 3.26. Exemple de répartition de quelques plateformes techniques de consultation sur les deux niveaux de framework PSM

L'utilisation d'un moteur de règles, au niveau de chaque framework, offre la possibilité, comme pour le framework de niveau PIM, d'adapter, facilement, le comportement de chaque élément du framework en fonction d'un certain nombre de critères (environnement, application, utilisateur, etc.). Ceci permet, comme pour le niveau PIM, de fiabiliser l'ensemble des éléments du framework tout en offrant un maximum de souplesse. De manière pratique, cette fiabilisation est assurée par le fait qu'une fois le framework développé, celui-ci n'a, en principe, plus de raison d'être modifié ; les changements de comportement étant gérés de manière externe à travers l'utilisation de règles. Il devient alors possible, du point de vue de l'architecture de PERCOMOM, de voir les frameworks de niveau PSM comme des boîtes noires qui ont suivi un processus rigoureux de développement et de validation permettant d'en assurer un fonctionnement avec un minimum d'erreurs.

Pour ce qui est du positionnement physique des différents éléments graphiques et de la définition et de l'utilisation de chartes graphiques, celles-ci se font à chaque niveau de framework PSM à travers

l'utilisation d'informations de présentation. Pour le premier niveau du framework, ces informations sont génériques et permettent de donner les grandes orientations de la présentation graphique des IHM aux utilisateurs. Pour le deuxième niveau, les informations permettent de préciser les spécificités graphiques de la plateforme finale. De cette manière, il devient possible d'avoir un pré-positionnement générique des éléments graphiques de l'IHM dans le cadre du premier framework et un positionnement finalisé dans le cadre du deuxième framework. Il est aussi possible d'associer le positionnement et/ou la charte graphique en fonction de l'application ou en fonction de l'élément d'interaction de type UIGroup ou UIUnit. De cette manière, il est par exemple possible de définir un positionnement identique pour tous les éléments d'interaction d'un UIGroup déterminé quelle que soit l'application tout en lui associant une charte graphique différente pour chaque application (police de caractères et couleurs différentes par exemple). De même, il est possible de définir, pour un même UIGroup, un positionnement différent des éléments d'interaction en fonction de chaque application.

Dans l'état actuel de nos travaux, la définition du positionnement des éléments d'interaction et la définition des chartes graphiques associées se font de manière manuelle, le pré-positionnement se limitant à reprendre le positionnement défini lors de la création des modèles statiques d'interaction au niveau conceptuel. De notre point de vue, le pré-positionnement automatique des éléments d'interaction devrait se faire en fonction de critères classiques liés à l'ergonomie des logiciels (Bastien et Scapin, 1993) (Vanderdonckt, 1994) (Vanderdonckt, 1999). Ceci permettrait de rendre les applications plus fonctionnelles mais aussi plus fiables en diminuant au maximum les interventions humaines (un mauvais positionnement où un mauvais choix de couleurs pouvant rendre un élément d'interaction difficilement utilisable). En couplant ceci à des méthodes et outils d'évaluation des interfaces, il deviendrait alors possible d'avoir un système de génération des interfaces capables de s'auto-évaluer et de s'auto-corriger afin de fournir un service de qualité aux utilisateurs. Le Tableau 3.9 présente, sous une forme synthétique l'ensemble des éléments de niveau PSM utilisés dans le cadre de PERCOMOM.

Tableau 3.9. Tableau récapitulatif des différents éléments architecturaux utilisés au niveau PSM dans PERCOMOM

Elément	Description
PSM framework pour une famille d'IHM	Représente une adaptation du framework de niveau PIM par rapport à une famille d'interface. Une famille d'IHM représentant un ensemble de plateformes d'affichage possédant un grand nombre de propriétés identiques (exemple : tous les PDA possèdent un certain nombre de caractéristiques en commun). De manière pratique, il n'existe pas qu'un seul framework de ce type mais autant de frameworks qu'il y a de familles d'IHM.
PSM framework pour une IHM spécifique	Représente une adaptation du framework "PSM framework pour une famille d'IHM" pour un type d'interface spécifique (exemple : les PDA couleur sous Palm OS version 5). De manière pratique, il existe un framework de ce type pour chaque type d'interface différente susceptible d'être manipulée par un utilisateur.
Informations génériques de présentation	Les informations génériques de présentation représentent des informations générales de présentation et de positionnement valables pour toutes les IHM des plateformes spécifiques qui sont rattachées à la famille d'interface. Ainsi, il est possible, pour un écran de sortie précis, de préciser le positionnement de chaque élément d'interaction qui le constitue ; ce positionnement étant alors commun à l'ensemble des plateformes spécifiques dérivées de la famille de plateformes.
Informations spécifiques de présentation	Les informations spécifiques de présentation représentent des informations de présentation et de positionnement qui sont spécifiques à une plateforme technique de consultation bien précise et donc qui ne sont pas partagées avec les autres plateformes de la même famille.
Règles pour le PSM framework pour une famille d'IHM	Cette base de règles permet de sortir une partie de la logique du "PSM framework pour une famille d'IHM" de manière à pouvoir modifier son comportement sans avoir besoin de redévelopper quoi que ce soit au niveau du framework. Ainsi, il serait possible de définir une règle de type validation précisant à partir de combien d'éléments, dans une liste affichée à l'écran, on dispose d'un ascenseur de navigation.
Règles pour le PSM framework pour une IHM spécifique	Cette base de règle permet de sortir une partie de la logique du "PSM framework pour une IHM spécifique de manière à pouvoir modifier son comportement sans avoir besoin de redévelopper quoi que ce soit au niveau du framework.

Au niveau de la transformation des éléments d'interaction, il est intéressant de noter que notre approche s'apparente à une spécialisation du modèle en Y (cf. Figure 1.3) de transformation proposé par l'OMG. En fait, la spécificité de notre approche réside dans l'utilisation de moteurs de règles et d'information externes de positionnement des éléments d'interactions qui permettent d'améliorer la "réutilisabilité" et la fiabilité des transformations tout en y apportant un maximum de souplesse. Pour illustrer cette "réutilisabilité", le processus de sélection des informations de positionnement des éléments est donné à titre d'indication dans la Figure 3.27.

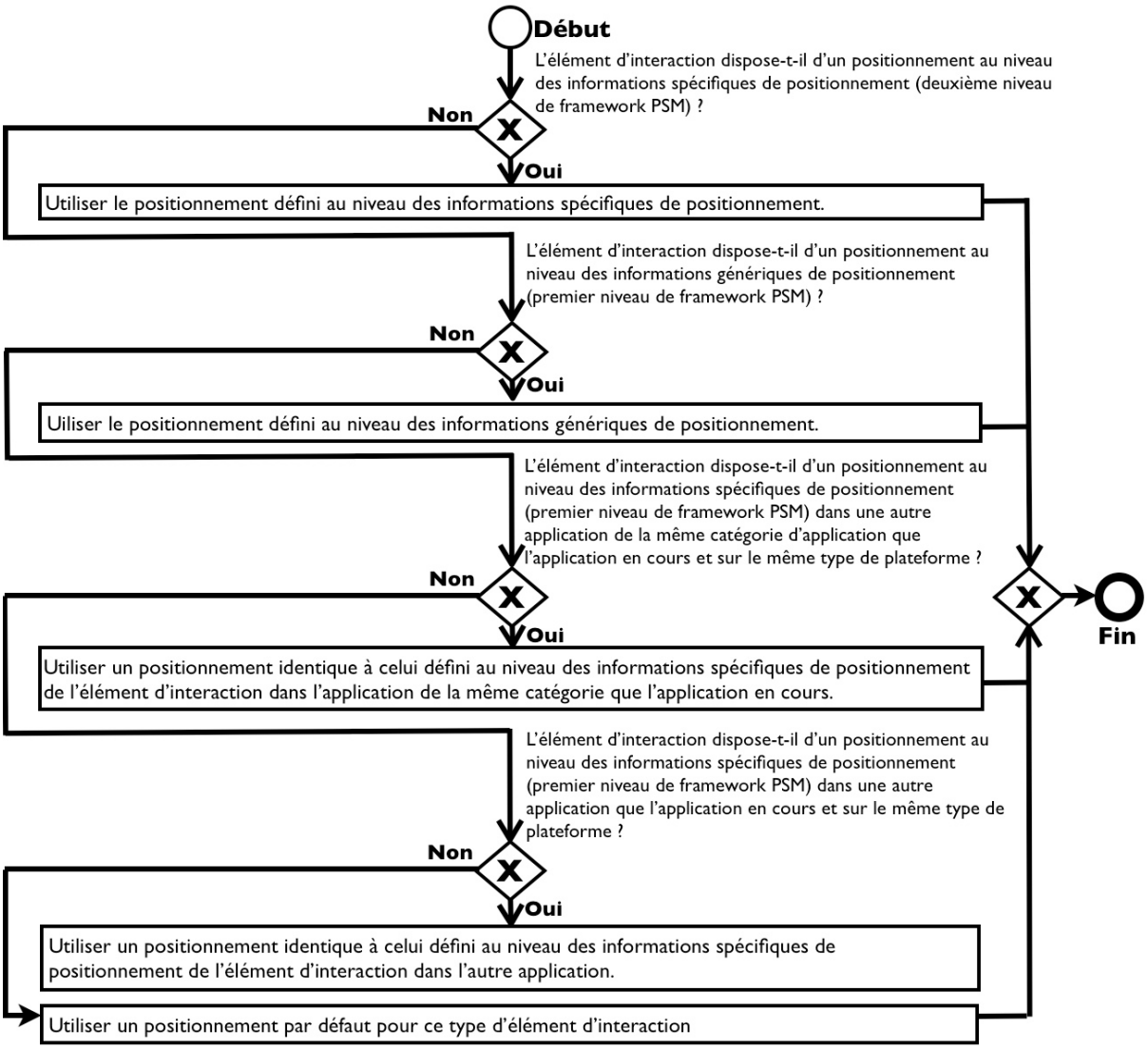


Figure 3.27. Processus de sélection des informations de positionnement d'un élément d'interaction au niveau PSM

3.2.4 Synthèse et discussion

Dans le cadre de PERCOMOM, l'utilisation d'une architecture MDA utilisant des frameworks aux niveaux PIM et PSM présente, par rapport à une approche classique de type MDA, un certain nombre d'avantages et d'inconvénients dont les principaux sont présentés dans le Tableau 3.10.

Tableau 3.10. Avantages et inconvénients de l'architecture de PERCOMOM par rapport à une architecture classique de type MDA

Avantages	Inconvénients
• L'utilisation de services fonctionnels et techniques de niveau PIM permet de simplifier les modèles de niveau CIM ; ceux-ci se concentrant uniquement sur les processus métier directement liés aux applications. • L'utilisation de frameworks permet de fiabiliser et d'homogénéiser les applications créées ; chaque élément du framework pouvant être utilisé par toutes les applications. • L'utilisation de moteurs de règles au niveau PIM et au niveau PSM permet d'adapter le fonctionnement des frameworks sans avoir besoin de redéfinir les différents composants qui les constituent. • L'utilisation d'une architecture à deux niveaux, pour la partie PSM, permet d'ajouter de nouvelles plateformes en ne devant redéfinir que les éléments vraiment spécifiques de la plateforme si du moins celle-ci peut se rattacher à une famille de plateformes existante.	• La séparation entre ce qui doit être passé en services fonctionnels et techniques de niveau PIM et ce qui doit être laissé au niveau des modèles conceptuels n'est pas encore très nette dans l'état actuel de l'approche. • L'utilisation des frameworks nécessite un travail plus poussé d'analyse lors de la création des éléments qui en font partie ; la modification de ces derniers ne devant être qu'exceptionnelle. • L'utilisation de règles, aux niveaux PIM et PSM, peut vite devenir très complexe à gérer si on ne dispose pas d'une dénomination standard et d'un guide de création de règles.

En conclusion, notre approche s'inscrit dans le cadre actuel du développement des fabriques d'applications. Dans celles-ci, on recherche, en effet, à diminuer les temps de développement en favorisant la réutilisation des éléments existants tout en fiabilisant les applications délivrées en diminuant au maximum les interventions humaines nécessaires à la création des applications. En ce sens, PERCOMOM présente une avancée dans le domaine de la création et de la génération automatique d'applications grâce à un enrichissement de l'approche MDA à travers l'utilisation de modèles conceptuels orientés métier mais aussi de frameworks fonctionnels et techniques aux niveaux PIM et PSM.

L'architecture MDA sur laquelle s'appuie PERCOMOM ayant été présentée, nous allons maintenant regarder comment la transformation de modèles est effectuée dans PERCOMOM pour passer des modèles de niveau CIM aux applications concrètes de niveau PSM.

3.3 Les principes de transformation de modèles pour passer du niveau CIM à l'application concrète de niveau PSM

3.3.1 Introduction

Pour l'OMG, dans une approche MDA (cf. chapitre 1, §1.1.3), les liens entre les différents niveaux de modèles sont établis à travers l'exécution de transformations de modèles qui se font de manière entièrement automatique. Les transformations principalement mises en avant dans MDA sont les transformations CIM vers PIM et PIM vers PSM. Néanmoins, dans une approche MDA, les transformations inverses sont aussi envisageables bien que généralement peu étudiées et peu utilisées. Dans le cadre de l'approche PERCOMOM, seules les transformations CIM vers PIM, PIM vers PSM et PSM vers code ont été étudiées.

PERCOMOM, en se basant sur une approche reposant sur des frameworks fonctionnels et techniques, n'utilise les préconisations de l'OMG que de manière partielle. Aussi est-il important de présenter les principes globaux associés à la transformation de modèles dans PERCOMOM qui sont sensiblement différents des approches classiques de type MDA.

3.3.2 La transformation de modèles dans PERCOMOM

La première chose à noter dans le cadre de l'approche PERCOMOM est que celle-ci se base sur un ensemble de modèles conceptuels de niveau CIM qui permettent de représenter une application du point de vue métier. En fait, chaque modèle de niveau CIM représente une vue métier différente de l'application et c'est l'association de l'ensemble des modèles de niveau CIM qui permet de créer l'application.

Dans le cadre de la transformation de modèles, une des spécificités de PERCOMOM réside dans l'existence des différents frameworks fonctionnels et techniques de niveau PIM et PSM. Pour les services de niveaux PIM, dans l'état actuel d'avancement de nos travaux et afin de pouvoir tester la validité de l'approche, ceux-ci sont modélisés suivant les préconisations de l'OMG à savoir à travers l'utilisation de modèles UML. Les services de niveaux PSM sont pour leur part modélisés à l'aide de profils UML adaptés à la plateforme technique sur laquelle sera déployée l'application.

En ce qui concerne les modèles CIM spécifiques à PERCOMOM, ceux-ci s'appuient, lors de leur transformation, sur des éléments de frameworks spécifiques définis aux niveaux PIM et PSM. Ainsi, si on prend l'exemple d'un modèle statique des interactions (IM2), les différents éléments des modèles utilisés au niveau CIM, les *UIElement*, sont définis de manière concrète dans les frameworks de niveau PIM et PSM. En fait lors de la création d'une application, au niveau CIM, pour les modèles statiques d'interaction (IM2), un modèle ne définit que des liens entre les différents éléments graphiques comme, par exemple, la composition d'un *UIUnit* ainsi que les propriétés associées à ces différents éléments graphiques. Par contre, la façon de prendre en charge ces propriétés et la façon de présenter physiquement les éléments graphiques à l'utilisateur au niveau de l'application finale sont définies au niveau des frameworks de niveau PIM et PSM. De même, l'association des services fonctionnels et techniques de niveau PIM à des modèles de niveau CIM, si elle permet de définir le comportement métier de l'application, n'est définie de manière effective et concrète que dans les frameworks de niveau PIM et PSM. Tout ceci est résumé dans la Figure 3.28.

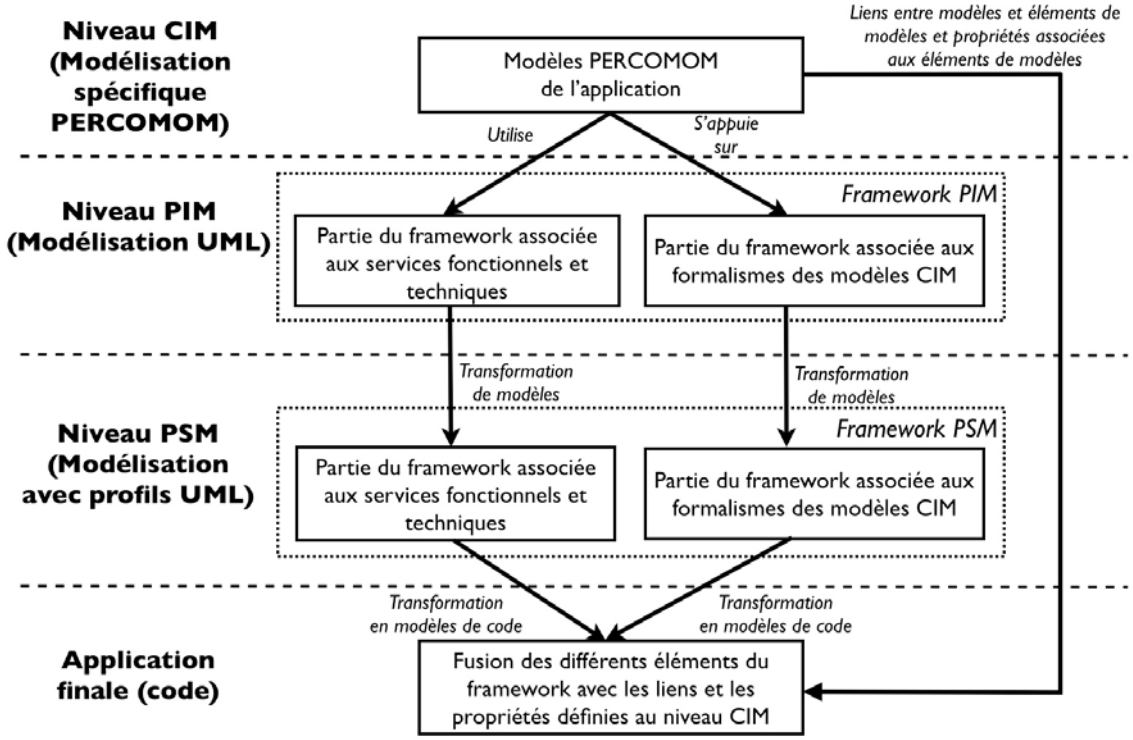


Figure 3.28. Principe global de transformation de modèles dans PERCOMOM

De manière globale, PERCOMOM peut être vue comme une approche de modélisation conceptuelle par composants métier pour lesquels on définirait, au niveau CIM :

- les relations entre composants (exemple : utilisation de règles métier (BM3) pour définir des conditions dans les processus métier (IM1)) ;

- les comportements de chaque composant en fonction d'un contexte métier particulier (exemple : utilisation de propriétés et de fonctions standards au niveau des *UIElement* dans les modèles CIM dans le cadre d'une application déterminée).

Une fois les frameworks de niveau PIM et PSM créés, le passage des modèles de niveau CIM au niveau PSM peut alors se faire en une seule transformation à travers un formalisme de transformation adaptée aux contraintes et spécificités techniques du niveau PSM ; ceux-ci étant définis dans chaque framework. En fait, seuls les liens entre les modèles et les éléments de modèles ainsi que les propriétés associées aux éléments de modèles sont transférés du niveau CIM au niveau PSM, l'application se basant alors sur une architecture technique et fonctionnelle de composants basée sur les différents frameworks. La Figure 3.29 donne un exemple de transformation de modèles dans le cadre de la création d'une application dans PERCOMOM qui sera utilisable sur deux plateformes techniques de consultation différentes : les PDA sous Palm OS et les PC avec une résolution d'écran de 600 * 800. Une fois les différents frameworks créés, la génération de l'application se résume à une transformation de modèles se basant sur trois types d'informations :

- les liens entre modèles et éléments de modèles et les propriétés associées aux éléments de modèles ;
- le PSM framework spécifique à la plateforme sur laquelle l'application va être utilisée ;
- les informations de présentation spécifiques à l'application et qui sont la synthèse des informations génériques et des informations spécifiques de présentation définis dans l'architecture PERCOMOM pour l'application considérée.

Ainsi, de manière concrète, lors de la génération d'une application, on aura une transformation de modèles par plateforme technique de consultation.

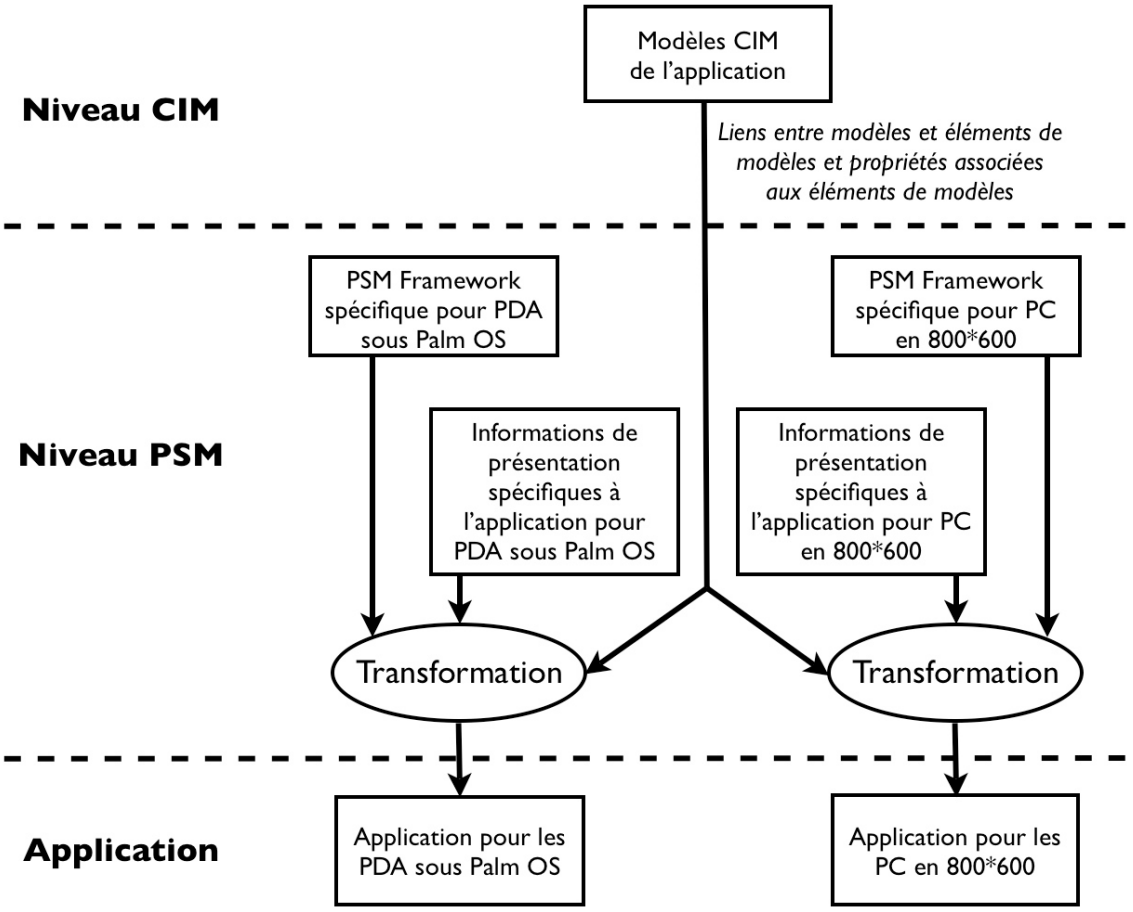


Figure 3.29. Exemple de transformation de modèles dans le cadre de la création d'une application finale dans PERCOMOM

Les modèles de niveau CIM étant par nature indépendants des plateformes techniques, cette transformation ne se fait pas forcément de manière directe ; chaque modèle de niveau CIM ne générant pas obligatoirement un seul modèle de niveau PSM. Ainsi, il sera possible d'avoir :

- Des tissages de modèles : deux ou plusieurs modèles de niveau CIM pouvant donner lieu à la création d'un seul modèle de niveau PSM. Par exemple, un modèle de profil utilisateur (modèle SM1) et un modèle de sécurité (modèle SM3) pourront donner lieu à la création d'un seul modèle d'utilisateur dans le cadre d'une application se basant sur un annuaire de type LDAP (Lightweight Directory Access Protocol) capable de stocker les informations du profil de l'utilisateur.
- Des séparations de modèles : un modèle de niveau CIM pouvant donner lieu à la création de plusieurs modèles de niveau PSM. Par exemple, un modèle de données (BM2) pourra donner lieu à la création de plusieurs modèles d'EJB (Enterprise JavaBeans) dans le cadre de la création d'une application Java.
- Des tissages et séparations de modèles : deux ou plusieurs modèles de niveau CIM pouvant donner lieu à plusieurs modèles de niveau PSM par tissage et séparation de modèles. Ainsi, si on reprend l'exemple de l'utilisation d'un annuaire LDAP, il serait possible d'envisager de gérer les informations statiques provenant du profil de l'utilisateur ainsi que les modèles de sécurité associés à l'application dans l'annuaire LDAP et les informations concernant les préférences de l'utilisateur dans une base de données.

La Figure 3.30 présente de manière synthétique les différentes possibilités de transformations de modèles du niveau CIM au niveau PSM.

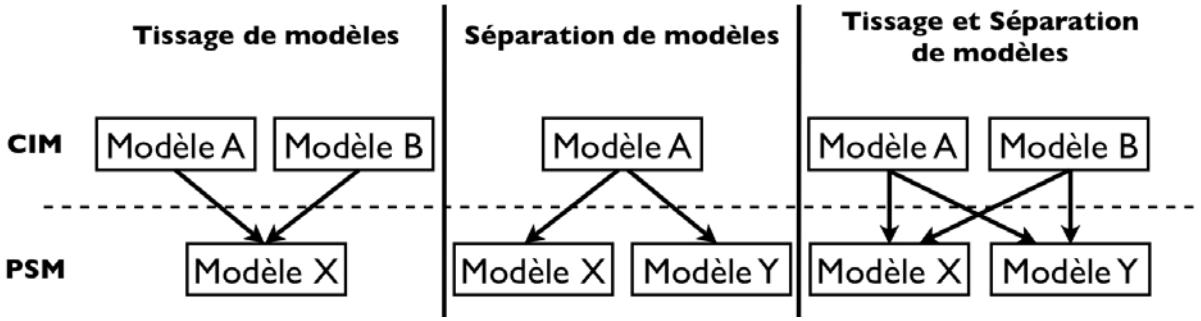


Figure 3.30. Les différents types de transformation de modèles dans PERCOMOM pour passer du niveau CIM au niveau PSM

En fait, le choix du type de transformation va fortement dépendre des choix de l'architecture technique (architecture SOA, client lourd, client léger, etc.) et des langages utilisés au niveau des applications finales (application Java, .NET, PHP, Ruby, etc.). En ce sens, si PERCOMOM se veut générique au niveau CIM à travers l'utilisation de modèles totalement indépendants des technologies utilisées dans les applications, le niveau PSM est par contre spécifique à l'environnement technique associé à l'application finale. Ceci est vrai :

- Pour les éléments de framework associés aux modèles conceptuels de niveau CIM dont certaines fonctionnalités peuvent ne pas être prises en compte ou alors peuvent être prises en compte de manière partielle dans certains environnements techniques. Ainsi, la prise en compte d'une adaptation de l'application au lieu géographique où se trouve l'utilisateur ne pourra se faire de manière effective que si la plateforme technique manipulée par l'utilisateur permet de le localiser.
- Pour les services fonctionnels et techniques de niveau PIM pour lesquels le mode de fonctionnement peut être très différent en fonction de l'environnement technique utilisé. Ainsi, la personnalisation des informations retournées à l'utilisateur lors d'une requête sera dépendante des capacités du service de personnalisation qui peuvent être différentes d'un environnement technique à un autre et d'une solution de personnalisation à une autre.

Si tout ceci semble, *a priori*, être un inconvénient, c'est aussi une véritable opportunité pour mettre en place de véritables fabriques d'applications. En effet, en se basant sur PERCOMOM, il est possible

d'envisager la création, par une société spécialisée, de librairies de "fonctionnalités" de niveau PIM et PSM contenant tout ou une partie des éléments des différents frameworks et diffusables auprès d'autres sociétés. Celles-ci pourraient alors être utilisées par des sociétés tierces dans un cadre technique bien défini. Ainsi, il deviendrait parfaitement envisageable de délivrer à une société utilisatrice un environnement PERCOMOM contenant l'ensemble des frameworks ainsi que toute l'architecture technique permettant de créer les applications sous la forme d'une "boîte noire" ; le client n'ayant plus qu'à créer des modèles de niveau CIM pour générer automatiquement des applications directement utilisables.

Dans le cadre de PERCOMOM, il convient aussi de ne pas oublier une des particularités de l'architecture globale qui réside dans la possibilité d'utiliser des moteurs de règles aussi bien au niveau PIM que PSM. Ces moteurs de règles vont permettre, dans le cadre de la transformation de modèles de pouvoir modifier le comportement de ces transformations en fonction de facteurs facilement modifiables. Ceci permettra de pouvoir faire évoluer et de pouvoir adapter les transformations de modèles, dans une certaine mesure, sans avoir besoin de redéfinir complètement l'architecture PERCOMOM. Ainsi, on pourrait imaginer, à travers l'utilisation d'une simple règle, permettre la génération d'une application soit dans un environnement technique java à base d'EJB soit dans un environnement technique java de type POJO (Plain Old Java Object)[1].

3.3.3 Analyse et discussion

Par rapport à la problématique de transformation de modèles dans le cadre d'une approche de type MDA, PERCOMOM présente les particularités suivantes :

- la possibilité de modéliser une application au niveau CIM et donc d'avoir une transformation de modèles à partir du niveau CIM ;
- l'utilisation de frameworks au niveau CIM et PSM qui permettent de simplifier les transformations de modèles ;
- une transformation "directe" des modèles CIM en modèles PSM s'appuyant sur les frameworks de niveau PIM et PSM permettant de générer de manière semi-automatique les applications.

Par rapport à une approche classique de type MDA, PERCOMOM introduit une notion de type "composant" (un composant représentant un élément du framework) dans les différents niveaux de l'architecture MDA (CIM, PIM et PSM). Ceci permet potentiellement de :

- simplifier les modèles de niveau CIM ;
- fiabiliser les transformations de modèles ; chaque composant pouvant être développé et testé de manière autonome ;
- faciliter les évolutions de l'environnement ; chaque composant pouvant être, dans une certaine mesure, modifié et adapté indépendamment des autres composants ;
- d'accélérer la mise en place d'un environnement de développement basé sur PERCOMOM ; les composants pouvant être développé par des sociétés tierces.

En fait, toute la richesse de PERCOMOM réside dans l'existence de ces différents composants qui représentent le cœur même de PERCOMOM.

Ceci étant, dans le cadre des transformations de modèles, PERCOMOM présente un inconvénient majeur dans le sens où il est impossible de générer une application et donc de faire une transformation de modèles tant que les différents frameworks n'ont pas été créés. L'utilisation de PERCOMOM nécessite donc une phase préparatoire beaucoup plus longue que les autres approches de type MDA.

[1] Le terme POJO a été inventé par Martin Fowler, Rebeccas Parsons et Josh MacKenzie en septembre 2000 : *"Nous nous sommes demandés pourquoi tout le monde était autant contre l'utilisation d'objets simples dans leurs systèmes et nous avons conclu que c'était parce que ces objets simples n'avaient pas un nom sympa. Alors, on leur en a donné un et ça a pris tout de suite"*. Le terme POJO définit un objet simple qui n'implémente pas d'interface spécifique à un framework comme c'est le cas pour un EJB (http:// http://www.martinfowler.com/bliki/POJO.html).

Elle nécessite surtout une réflexion préalable plus importante au niveau de la création des fonctionnalités reprises dans les différents frameworks car toute fonctionnalité oubliée pourra nécessiter des modifications plus ou moins importantes aux niveaux PIM et PSM. De même, la prise en compte de nouvelles contraintes techniques et ou de nouvelles contraintes fonctionnelles (services de niveau PIM) pourront nécessiter de profondes modifications des différents frameworks et éventuellement une modification des modèles de niveau CIM des applications. Ceci nécessiterait alors une reprise de l'ensemble des modèles existants pour toutes les applications afin de les adapter aux nouveaux modèles si on veut pouvoir les réutiliser lors de la création de nouvelles applications.

Conclusion

Aujourd'hui pour répondre à l'augmentation de la complexité des applications ainsi qu'aux exigences toujours plus fortes en termes de diminution des coûts de développement, diminution des délais de mise en production et augmentation de la qualité des applications fournies, les approches de type ingénierie dirigée par les modèles semblent être la solution idéale. Malheureusement, on ne dispose pas aujourd'hui, aussi bien dans le domaine de la recherche que dans le domaine commercial, d'outils ou d'approches permettant de prendre en compte la globalité d'une application.

À côté de cela, la généralisation des architectures de type SOA, entraîne une nouvelle façon de concevoir des applications par assemblage de composants métier. Ces assemblages sont réalisés à travers la création de processus métier et l'utilisation de règles métier qui permettent de modifier facilement et rapidement le comportement des processus métier.

Partant de ce constat, et en se basant sur la vision à moyen terme de l'OMG dans le domaine de la modélisation, il paraissait pertinent de développer une nouvelle méthode de conception des applications en essayant de capitaliser sur les avantages de chaque type d'approche. C'est cette réflexion qui a donné naissance à la méthode de modélisation PERCOMOM (PERsonalization and COnceptual MOdeling Method) que nous avons proposée et présentée, dans sa première version, dans ce chapitre.

Celle-ci est, à notre connaissance, la première méthode de modélisation des applications interactives qui soit capable de prendre en compte une application à travers des modèles de support et des modèles d'interactions. En se basant principalement sur les processus métier et en utilisant des services de niveau PIM, elle permet d'avoir des modèles conceptuels uniquement centrés sur le métier et donc plus facile à lire et à modifier. Elle permet aussi, à travers une architecture à plusieurs niveaux, de créer des applications plus facilement adaptables, à travers l'utilisation de règles, tout en augmentant l'homogénéité des comportements des applications.

Si PERCOMOM permet de modéliser les applications, c'est aussi une méthode capable de prendre en charge la personnalisation des applications au niveau de la modélisation conceptuelle de niveau CIM. C'est ce point que nous allons développer dans le chapitre suivant.

Chapitre 4
PERCOMOM, une prise en compte de la personnalisation qui s'appuie sur une ontologie de domaine

Sommaire

INTRODUCTION **115**

4.1 LA PERSONNALISATION DES CONTENUS : UN SERVICE FONCTIONNEL DE NIVEAU PIM **116**
4.1.1 INTRODUCTION 116
4.1.2 INTEGRATION DU SERVICE FONCTIONNEL DE PERSONNALISATION DANS LES MODELES CONCEPTUELS 116
4.1.3 L'EVOLUTIVITE DU SERVICE DE PERSONNALISATION DANS L'ARCHITECTURE PROPOSEE PAR PERCOMOM 122
4.1.4 SYNTHESE ET DISCUSSION 125

4.2 LA PRISE EN COMPTE DU CONTEXTE **126**
4.2.1 INTRODUCTION 126
4.2.2 LES CONTEXTES APPLICATIFS, GEOGRAPHIQUES ET TEMPORELS 127
4.2.3 LES AUTRES ELEMENTS CONTEXTUELS 129
4.2.4 PRISE EN COMPTE DU CONTEXTE ET APPROCHE MDA 132
4.2.5 SYNTHESE ET DISCUSSION 134

4.3 LA PRISE EN COMPTE DE L'UTILISATEUR **136**
4.3.1 INTRODUCTION 136
4.3.2 LA PRISE EN COMPTE DES PREFERENCES DE L'UTILISATEUR 136
4.3.3 LA PRISE EN COMPTE DE L'EXPERIENCE DE L'UTILISATEUR VIS-A-VIS DE L'APPLICATION 138
4.3.4 SYNTHESE ET DISCUSSION 140

4.4 LA PRISE EN COMPTE DES CONTENUS **140**
4.4.1 INTRODUCTION 140
4.4.2 LES CONTENUS ET L'ONTOLOGIE DE DOMAINE 141
4.4.3 LA NOTION DE SILO DE DONNEES AU NIVEAU DES APPLICATIONS 144
4.4.4 SYNTHESE ET DISCUSSION 146

CONCLUSION **147**

Introduction

Aujourd'hui, le client d'une entreprise ou d'une administration est de plus en plus demandeur de services personnalisés parfaitement adaptés à ses besoins et à ses attentes. À côté de cela, les entreprises cherchent à établir un lien de plus en plus étroit avec leurs clients de manière à :

- Mieux connaître les besoins de leurs clients afin de créer des produits et services les plus adaptés aux différents marchés.
- Mieux servir les clients afin de maximiser les relations commerciales et les bénéfices pouvant être tirés de cette relation.

D'où des besoins de plus en plus importants pour avoir des applications capables d'établir des relations individualisées avec les utilisateurs.

Au niveau des applications informatiques, une des solutions pour atteindre ces différents objectifs passe par la personnalisation aussi bien des contenus que des interfaces et/ou des processus métier. Mais aujourd'hui, s'il existe un certain nombre de solutions pour personnaliser les applications (cf. chapitre 2), il n'en existe quasiment aucune qui permette de prendre en compte cette personnalisation dans le cadre d'une approche de type IDM. Profitant de l'expérience du LAMIH dans le domaine de la modélisation et de la personnalisation des contenus (Petit-Rozé, 2003) (Anli, 2006), nous avons intégré cette problématique de la personnalisation dans le cadre du développement de PERCOMOM afin de vérifier sa capacité à prendre en compte différentes problématiques fonctionnelles. Cette précision étant apportée, nous allons, dans ce chapitre, montrer comment PERCOMOM gère la personnalisation dans le cadre de la modélisation.

Dans une première partie, nous allons expliquer pourquoi la personnalisation des contenus est un service de niveau PIM. Pour cela, nous allons commencer par présenter comment la personnalisation est intégrée dans les modèles conceptuels des applications. Puis, nous étudierons quels sont les avantages de ce type d'approche notamment du point de vue de l'évolutivité des outils de personnalisation dans le temps. Enfin, nous montrerons quels sont les avantages de notre approche par rapport aux autres approches existantes.

Dans une deuxième partie, nous allons expliquer comment, dans le cadre de PERCOMOM, il est possible de tenir compte du contexte pour personnaliser les applications aussi bien au niveau des contenus, des services que des processus métier. Dans une première partie, nous montrerons comment des éléments contextuels comme l'application, le temps et l'espace peuvent être pris en compte. Puis nous verrons comment il est possible d'intégrer d'autres éléments du contexte. Enfin, nous expliquerons pourquoi, de notre point de vue, la prise en compte du contexte dans une approche MDA doit se faire à tous les niveaux de l'architecture.

Dans une troisième partie, nous verrons comment l'utilisateur est pris en compte dans PERCOMOM au niveau de la personnalisation. Pour commencer, nous présenterons plus en détail le modèle de l'utilisateur. Puis, nous verrons que l'ontologie représente un élément important pour lier l'utilisateur au domaine métier. Enfin, nous verrons comment l'expérience de l'utilisateur peut, elle aussi, être prise en compte.

Dans une quatrième et dernière partie, nous verrons comment les contenus sont gérés dans le cadre de la personnalisation. Pour cela, nous verrons comment lier les contenus à l'ontologie de domaine. Puis nous aborderons la notion d'ensembles d'informations qui est utilisée au niveau des applications pour grouper les informations. Enfin, nous terminerons par la personnalisation des contenus même et plus précisément nous montrerons comment celle-ci est réalisée concrètement dans une application basée sur PERCOMOM.

Mais, pour commencer, il est indispensable de présenter comment la personnalisation des contenus est prise en compte, de manière globale, dans le cadre de PERCOMOM et plus exactement d'expliquer pourquoi elle doit être considérée comme un service fonctionnel de niveau PIM.

4.1 La personnalisation des contenus : un service fonctionnel de niveau PIM

4.1.1 Introduction

La notion de contenu pouvant être assez vaste (cf. chapitre 2), nous considérerons, dans la suite de cette partie, qu'un contenu est directement lié à la notion de données. En fait, il représente une vue sur un ensemble de données en fonction de critères (exemple : uniquement les données créées depuis 20 jours) et de limites (exemple : les 30 premiers enregistrements trouvés). En partant de cette définition, personnaliser un contenu consiste à retourner à l'utilisateur des informations en fonction de critères et de limites qui lui sont spécifiques.

En considérant le fait que les modèles conceptuels doivent être totalement indépendants des technologies, l'intégration de la personnalisation dans les modèles CIM ne peut se faire que du point de vue fonctionnel. Aussi ce qu'il est important de connaître pour personnaliser un contenu ce n'est pas comment cette personnalisation est effectuée mais quels sont les critères et les limites qui sont utilisés pour effectuer cette personnalisation. Si, par nature, il est possible d'imaginer tous types de critères et de limites, ceux-ci peuvent néanmoins se classer en trois catégories :

- Les critères et limites liés au concept métier manipulé. Par exemple, si on dispose d'une source de données contenant l'ensemble des horaires des trains au départ de toutes les gares de France, un critère pourrait être le nom d'une gare de manière à ne fournir à l'utilisateur que les horaires de départ pour une gare déterminée.
- Les critères et limites liés à l'utilisateur. Par exemple, pour une application de type agenda, on ne retournerait à l'utilisateur que les rendez-vous qui lui sont propres et pas tous les rendez-vous de tous les utilisateurs. Les critères utilisés à ce niveau étant directement reliés à l'utilisateur, ils prennent aussi en compte ses préférences et son historique de manipulation des données.
- Les critères et limites liés à l'environnement technique. Par exemple, pour une même IHM, le nombre d'informations pouvant être affichées sur un ordinateur fixe et sur un téléphone ne seront pas toujours les mêmes aussi bien au niveau du nombre de résultats affichés que du contenu affiché (nombre de propriétés du concept métier affichées). Pour prendre en compte cette spécificité nous avons défini, dans le cadre de PERCOMOM, trois niveaux possibles de plateforme : haute, moyenne et basse qualité. Des exemples d'utilisation de ces trois niveaux seront présentés dans la suite du document. Il est important de noter que le choix de définir trois niveaux de plateforme ne repose pas sur une étude approfondie des différents types de plateformes utilisables. En fait, ce choix repose sur une première approche permettant à des applications de fonctionner sur trois types de plateformes : un ordinateur PC fixe ou portable, un PDA et un téléphone portable (et ceci avec, avant tout, un objectif d'étude de faisabilité).

Si les critères et les limites sont multiples, leur intégration au niveau d'un modèle conceptuel peut aussi se faire à plusieurs niveaux comme nous allons maintenant le présenter.

4.1.2 Intégration du service fonctionnel de personnalisation dans les modèles conceptuels

4.1.2.1 Introduction

Lorsqu'on se place du point de vue métier, la personnalisation des contenus peut être vue de deux manières différentes (le Tableau 4.1 résume les avantages et les inconvénients de chaque solution) :

- Comme une fonctionnalité métier spécifique à chaque application. Dans ce cas, on définit au niveau de chaque application ce qu'on entend par personnalisation des contenus. Pour cela, on doit développer, pour chaque application, l'ensemble des modèles de niveau CIM, qui permettent de prendre en compte cette personnalisation.

- Comme une fonctionnalité métier qui peut être partagée par plusieurs applications. Dans ce cas, on définit celle-ci comme un ensemble de fonctionnalités réutilisables, du point de vue métier, par plusieurs applications. La personnalisation est alors vue comme un service fonctionnel de niveau PIM (cf. chapitre 3, §3.2.2).

Tableau 4.1. Tableau comparatif des principaux avantages et inconvénients des différentes possibilités de prise en charge de la personnalisation des contenus dans PERCOMOM

	Personnalisation vue comme faisant partie intégrante de modèles de niveau CIM	**Personnalisation vue comme un service fonctionnel de niveau PIM**
Avantages	• L'ensemble des fonctionnalités métier se trouvent dans les modèles de niveau CIM de l'application ; ce qui facilite leur prise en main et leur gestion (tout est regroupé en un seul endroit) par les personnes en charge de la modélisation des applications.	• Lorsqu'on souhaite modifier le comportement de la fonctionnalité métier de personnalisation des contenus, on ne doit le faire qu'à un seul endroit : dans le service fonctionnel de personnalisation de niveau PIM • À travers l'utilisation d'un seul service fonctionnel de niveau PIM de personnalisation des contenus, on s'assure de l'homogénéité de fonctionnement de la fonctionnalité de personnalisation des contenus.
Inconvénients	• Lorsqu'on souhaite modifier le comportement de la fonctionnalité métier de personnalisation des contenus, on est obligé de modifier l'ensemble des applications au niveau CIM. • L'homogénéité de fonctionnement de la fonctionnalité de personnalisation au sein des différentes applications ne peut être assurée ; chaque application pouvant utiliser une fonctionnalité de personnalisation différente.	• L'ajout d'une nouvelle possibilité fonctionnelle métier au niveau du service fonctionnel de personnalisation de niveau PIM, comme, par exemple, l'ajout de la possibilité de personnaliser les contenus en fonction de l'expérience des utilisateurs sur l'application, peut nécessiter de revoir l'ensemble des applications pour qu'elle soit prise en compte de manière optimale dans chaque application. Pour rappel, l'utilisation des services fonctionnels de niveau PIM se faisant à travers l'utilisation d'artefacts et de propriétés au niveau des modèles de niveau CIM, il est possible de donner un comportement par défaut à chaque nouvelle possibilité fonctionnelle métier de personnalisation sans avoir besoin de modifier chaque application.

Dans le cadre de PERCOMOM, en comparant les avantages et les inconvénients des deux types d'approches, nous avons considéré qu'il était opportun de prendre en compte la personnalisation des contenus comme un service fonctionnel de niveau PIM. La question qui se pose alors et de savoir si cette approche permet de prendre en compte les trois types de critères qui sont liés au métier, à l'utilisateur et à l'environnement.

4.1.2.2 Prise en compte des critères et limites liés aux concepts métier manipulés

Pour les critères et limites liés aux concepts métier manipulés, la prise en compte ne se fait pas directement au niveau du service de personnalisation mais à travers l'utilisation de critères sauvegardés. Du point de vue logiciel, les critères sont de deux ordres :

- **Des critères de sélection :**

  Un critère de sélection permet d'indiquer les valeurs des propriétés à prendre en compte dans le cadre de la requête de personnalisation. Ces critères peuvent être :

  - Des critères simples du type égalité de valeurs (exemple : "Train.TypeDeTrain==TGV" pour indiquer qu'on souhaite, au niveau du critère de sélection, que la valeur de la propriété "TypeDeTrain", de la classe "Train" de l'ontologie de domaine (OD), soit égale à la valeur fixe "TGV".)

- Des critères de comparaison permettant de ne sélectionner que les éléments dont la valeur de la propriété donne un résultat vrai à la comparaison. Par exemple, si on veut afficher tous les trains sauf les TGV, on utilisera un critère de comparaison du type "Train.TypeDeTrain!=TGV" pour indique qu'on souhaite, au niveau du critère de sélection, que la valeur de la propriété "TypeDeTrain" de la classe "Train" de l'ontologie de domaine (OD) soit différente de la valeur fixe "TGV".

Dans PERCOMOM, la définition un critère de sélection est la suivante :

classeOD.propriétéOD ***opérateur*** *valeurcritère*

Avec :

- *classeOD :* le nom de la classe de l'ontologie de domaine (OD) sur laquelle on va définir un critère de sélection
- *propriétéOD :* le nom de la propriété de l'ontologie de domaine (OD) sur laquelle on va définir un critère de sélection. Dans l'état actuel de nos travaux, on ne peut définir un critère de sélection que sur une propriété d'une classe de l'ontologie de domaine (OD).
- *opérateur :* l'opérateur de sélection utilisé (l'ensemble des opérateurs utilisés dans le cadre de PERCOMOM sont donnés dans le Tableau 4.2).
- *valeurcritère :* la valeur fixe utilisée dans le cadre de la sélection.

Tableau 4.2. Les opérateurs de sélection définis dans le cadre de PERCOMOM pour effectuer des sélections sur les individus de l'ontologie de domaine (OD)

Opérateur	Commentaire
==	La valeur de la propriété *propriétéOD* doit être strictement égale à la valeur de *valeurcritère*
!=	La valeur de la propriété *propriétéOD* doit être différente de la valeur de *valeurcritère*
<	La valeur de la propriété *propriétéOD* doit être strictement inférieure à la valeur de *valeurcritère*
<=	La valeur de la propriété *propriétéOD* doit être inférieure ou égale à la valeur de *valeurcritère*
>	La valeur de la propriété *propriétéOD* doit être strictement supérieure à la valeur de *valeurcritère*
>=	La valeur de la propriété *propriétéOD* doit être supérieure ou égale à la valeur de *valeurcritère*

Dans PERCOMOM, l'utilisation des critères de sélection peut se faire :

- Sur l'ensemble des individus de l'ontologie ; ce qui permet d'obtenir la liste de tous les individus de l'ontologie de domaine (OD) pour lesquels le critère de sélection est valide.
- Sur le résultat, c'est-à-dire l'ensemble des individus de l'ontologie de domaine (OD) d'une sélection effectuée à l'aide d'un autre critère de sélection.

- **Des critères de tri :**

Un critère de tri permet d'effectuer des tris sur les résultats retournés par une sélection. Les critères de tri sont indépendants les uns des autres et sont définis à partir du nom de la classe de l'ontologie de domaine, du nom de la propriété de l'ontologie et du sens du tri à appliquer. Par exemple, si on veut trier les types de trains par ordre ascendant (tri par ordre alphabétique croissant des libellés associés à chaque type de train (TER, TGV, Corail, etc.)), on utilisera un critère de tri de type "Train.TypeDeTrain ASC" pour indiquer qu'on souhaite faire un tri sur la valeur de la propriété "TypeDeTrain" pour les individus de la classe "Train" de l'ontologie de domaine (OD) par ordre ascendant ("ASC").

La définition d'un critère de tri se fait de la manière suivante :

classeOD.propriétéOD ***sensdutri***

Avec :

- *classeOD :* le nom de la classe de l'ontologie de domaine (OD) sur laquelle on va définir un critère de sélection
- *propriétéOD :* le nom de la propriété de l'ontologie de domaine (OD) sur laquelle on va définir un critère de sélection. Dans l'état actuel de nos travaux, on ne peut définir un critère de sélection que sur une propriété d'une classe de l'ontologie de domaine (OD).
- *Sensdutri :* permet d'indiquer le sens dans lequel le tri sera effectué. Il ne peut prendre que deux valeurs : "ASC" pour un tri ascendant, c'est-à-dire par ordre numérique ou alphabétique croissant, et "DESC" pour un tri par ordre descendant, c'est-à-dire par ordre numérique ou alphabétique décroissant.

Dans le cadre de PERCOMOM, l'ensemble des critères de sélection peuvent être définis dans les modèles d'actions (modèles BM1) au sein de tâches spécifiques qu'on appelle *sélection*. La définition de ces tâches se fait à travers un formalisme de type XML dont un exemple est donné ci-dessous :

```
<selection nom="trainsrapides" utiliseselection"traindenuit">
   <critereselection classeOD="Train" proprieteOD="TypeDeTrain" operateur="==" 
   critere="TGV"/>
   <criteretri classeOD="Train" propriétéOD="NumeroTrain" sensdutri="ASC"/>
</selection>
```

Cet exemple de *sélection*, qui porte le nom de "trainsrapides" permet de retourner l'ensemble des individus de la classe "Train" de l'ontologie de domaine (OD) qui ont la valeur de la propriété "TypeDeTrain" égale à "TGV" et qui proviennent d'une première *sélection* portant le nom de "traindenuit" permettant de sélectionner l'ensemble des individus de la classe de l'ontologie "Train" circulant la nuit.

Dans l'état actuel de nos travaux, dans le cas où, dans la définition d'une *sélection*, plusieurs critères de sélection seraient définis, ceux-ci seront reliés les uns aux autres avec un opérateur logique de type ET. Ainsi, si on définit un premier critère ne prenant que les trains de type TGV et un deuxième critère ne prenant que les trains circulant sur les lignes de l'ouest de la France, alors la sélection ne retournera que les trains de type TGV circulant sur les lignes de l'ouest de la France.

Dans une *sélection*, il est aussi possible de définir plusieurs critères de tri différents. Dans ce cas, chaque critère de tri s'ajoute au précédent et les tris sont effectués les uns après les autres dans l'ordre dans lequel ils sont rencontrés dans la définition de la sélection. Ainsi, si on demande d'effectuer un premier tri sur le type de train puis on définit un deuxième tri sur le numéro du train, le résultat retourné sera d'abord trié par type de train puis, pour chaque type de train, il sera trié sur le numéro du train.

4.1.2.3 Prise en compte des critères et limites liés à l'utilisateur

Pour ce qui est des critères et limites liés à l'utilisateur, leur prise en compte se fait à deux niveaux :

- À travers l'utilisation de propriétés au niveau de l'artefact lié au service de personnalisation (cf. Figure 4.1).

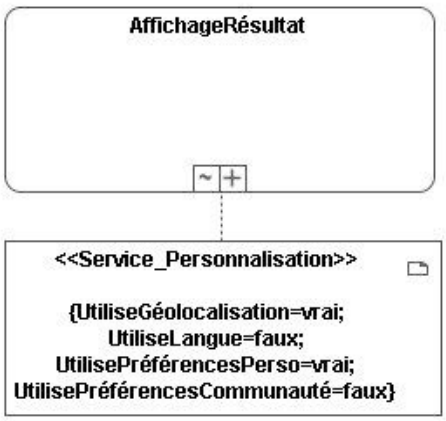


Figure 4.1. Exemple d'association d'un service de personnalisation à un élément d'interaction de type *UIUnit* dans un modèle statique d'IHM (modèle IM2) au niveau CIM

Dans cet exemple, on associe à un *UIUnit* "AffichageRésultat", un service de personnalisation des contenus qui personnalisera les contenus en fonction de la localisation géographique de l'utilisateur, au moment où la personnalisation est effectuée, et des préférences de communauté ; c'est-à-dire en se basant sur les choix effectués par d'autres utilisateurs ayant un profil similaire à l'utilisateur en cours (cf. chapitre 2.1.3).

À titre d'information, la signification des différentes propriétés de l'artefact associé au service de personnalisation de niveau PIM est donnée dans le Tableau 4.3. Il est important de noter, que, dans un artefact, il n'est pas obligatoire de donner une valeur à chaque propriété ; celles-ci utilisant automatiquement la valeur par défaut si celle-ci n'est pas indiquée.

Tableau 4.3. Les différentes propriétés définies au niveau de l'artefact associé au service de personnalisation de niveau PIM

Propriété	Commentaire
UtiliseGéolocalisation	Permet d'indiquer si la personnalisation des contenus doit tenir compte ou pas de la localisation géographique de l'utilisateur au moment où la personnalisation est effectuée. Peut prendre deux valeurs : Vrai ou Faux. La valeur par défaut est égale à Faux.
UtiliseLangue	Permet d'indiquer si la personnalisation des contenus doit tenir compte de la langue de l'utilisateur ou si elle ne doit pas en tenir compte. Peut prendre deux valeurs : Vrai ou Faux. La valeur par défaut est égale à Faux.
UtilisePréférencePerso	Permet d'indiquer si la personnalisation des contenus doit tenir compte ou pas des préférences de l'utilisateur (préférences indiquées dans son profil). Peut prendre deux valeurs : Vrai ou Faux. La valeur par défaut est égale à Faux.
UtilisePréférenceCommunauté	Permet d'indiquer si la personnalisation des contenus doit se faire en fonction de préférences communautaires c'est-à-dire en fonction de similarités avec les choix effectués par d'autres utilisateurs (cf. chapitre 2.1.3). Peut prendre deux valeurs : Vrai ou Faux. La valeur par défaut est égale à Faux.
PropriétésExclues	Permet d'indiquer la liste des propriétés des classes de l'ontologie qui ne seront pas prises en compte dans le cadre de l'ontologie de domaine. Pour donner un exemple, si on désire que la personnalisation ne soit pas effectuée sur la propriété "TypeDeTrain" de la classe de l'ontologie "Train", on indiquera la valeur Train."TypeDeTrain" dans la liste des propriétés exclues. Cette propriété peut prendre plusieurs valeurs différentes qui doivent être obligatoirement séparées par une virgule. Il n'y a pas de valeur par défaut pour cette propriété.

- À travers l'utilisation de propriétés de concepts de l'ontologie de domaine (OD) si ces propriétés peuvent être associées à un élément contenu dans le profil de l'utilisateur (modèle SM1). Ceci est présenté dans l'exemple de la Figure 4.2.

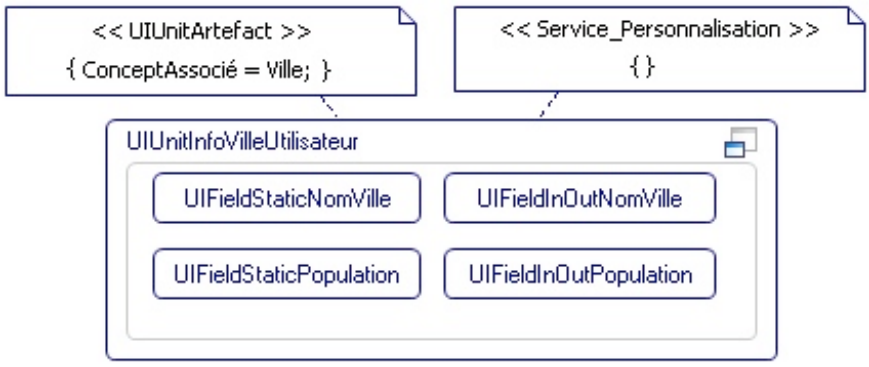


Figure 4.2. Utilisation d'un critère lié au concept dans le cadre de la personnalisation

Dans cet exemple, on désire afficher le nom ainsi que la population de la ville dans laquelle habite l'utilisateur. Pour cela, on définit un *UIUnit* ("UIUnitInfoVilleUtilisateur") permettant d'afficher deux champs statiques de type *UIFielsStatic* qui contiendront deux libellés "Nom de la ville : " et "Population :" et deux champs de type *UIFieldInOut* permettant d'afficher la valeur de la propriété "Nom" de la classe "Ville" de l'ontologie de domaine (OD) et la valeur de la propriété "Population" de la classe "Ville" de l'ontologie de domaine (OD) pour un individu de l'ontologie de domaine de type "Ville". Ensuite, on associe à l'*UIUnit*, à travers l'artefact associé, le nom de la classe de l'ontologie qui sera associée aux *UIField* contenus dans l'*UIUnit*. Puis, on associe à l'UIUnit "UIUnitInfoVilleUtilisateur", un service de personnalisation de niveau PIM afin d'indiquer qu'on désire personnaliser les informations affichées par l'UIUnit. Par contre, on ne définit aucune propriété dans l'artefact associé au service de personnalisation car on désire que la personnalisation soit effectuée à partir des informations contenues dans le profil de l'utilisateur (association par défaut effectuée au niveau des services de personnalisation).

4.1.2.4 Prise en compte des critères et limites liées à l'environnement technique

Pour ce qui est des critères et limites liées à l'environnement technique, la personnalisation des contenus, qui est vue ici uniquement du point de vue de l'adaptation des IHM au contexte d'utilisation de l'application, se fait à deux niveaux :

- Au niveau des données qui doivent contenir des informations adaptées aux différents types de plateformes.

  Ainsi, si on prend l'exemple d'un libellé statique affiché à l'utilisateur, celui-ci est susceptible de subir un certain nombre de contraintes en fonction de la plateforme technique de consultation utilisée ; la taille de l'écran au niveau de chaque plateforme ne permettant pas d'afficher un libellé comportant le même nombre de caractères de manière ergonomique pour l'utilisateur. La solution choisie au niveau de PERCOMOM pour prendre en charge cette problématique est d'offrir la possibilité de définir, pour chaque type de données manipulées, y compris les libellés statiques, 3 types de valeurs différentes adaptées chacune à un type de plateforme bien déterminé. Par exemple, pour un libellé de type erreur affichant un message si un mot de passe est incorrect, on pourrait avoir trois contenus différents en fonction de chaque niveau de plateforme (cf. Tableau 4.4).

Tableau 4.4. Exemple de définition d'un libellé de type erreur pour les trois niveaux de plateformes définit dans PERCOMOM

Niveau de plateforme	Taille maximale	Exemple de libellé
Haute qualité	Limite fixée arbitrairement à 1024 caractères dans le cadre de nos travaux.	*Le mot de passe que vous avez indiqué est incorrect. Veuillez le ressaisir.*
Moyenne qualité	20 caractères maximum	*Erreur mot de passe*
Basse qualité	8 caractères	*Err pass*

Il est important de noter que la problématique de personnalisation des contenus en fonction de l'environnement technique n'étant pas au cœur de nos travaux de recherche, le choix des trois niveaux de plateformes a été fait de manière arbitraire afin de pourvoir faire une première validation de l'approche proposée.

- Au niveau des frameworks PSM et des règles associées qui permettent, par exemple, d'indiquer, pour chaque type de plateforme spécifique, le niveau de plateforme pour chaque catégorie d'*UIElement* manipulée. Ainsi, il est possible de définir, pour un téléphone portable d'un modèle et d'une marque bien précis, que celui-ci peut afficher des vidéos (*UIFieldVideo*) en qualité moyenne et des libellés (*UIFieldStatic*) en qualité basse.

4.1.2.5 Conclusion

Comme nous venons de le voir, l'intégration du service fonctionnel de personnalisation de niveau PIM dans les modèles CIM permet de prendre en compte les trois principaux types de critères pouvant avoir un impact au niveau de la personnalisation réalisée :

- les données à travers la prise en compte des concepts métier,
- les utilisateurs à travers la prise en compte de différents éléments dont le profil de l'utilisateur,
- l'environnement à travers la prise en compte des contraintes techniques des différentes plateformes à travers une définition possible de chaque contenu suivant trois niveaux de "qualité" (Haute qualité, Moyenne qualité et Basse Qualité).

Ceci permet d'offrir de nombreuses possibilités différentes de personnalisation au niveau conceptuel permettant potentiellement de répondre à un grand nombre de problématiques fonctionnelles.

4.1.3 L'évolutivité du service de personnalisation dans l'architecture proposée par PERCOMOM

Dans le cadre de PERCOMOM, l'évolutivité du service de personnalisation, c'est-à-dire son adaptation aux besoins fonctionnels et aux besoins techniques, peut se faire aux trois niveaux de l'architecture MDA :

- Au niveau CIM, l'évolution du service de personnalisation se traduit par la définition de nouvelles propriétés et/ou critères au niveau de l'artefact associé au service de personnalisation. Il est aussi possible de supprimer des propriétés et ou critères existants mais, dans ce cas, une analyse d'impact fonctionnelle sur les applications existantes doit être menée. Les propriétés et critères devant être pertinents et non recouvrants, il conviendra aussi de vérifier la cohérence des ajouts par rapport à l'existant. Dans l'état actuel de nos travaux, nous n'avons pas créé d'outils permettant de réaliser ces différentes analyses mais c'est une perspective de recherche sur laquelle nous reviendrons dans le chapitre 6. La Figure 4.3 présente de manière simplifiée le processus de prise en compte

d'un nouveau besoin fonctionnel, en matière de personnalisation, défini par les experts métier (besoin d'évolution au niveau CIM), au niveau de l'architecture de PERCOMOM.

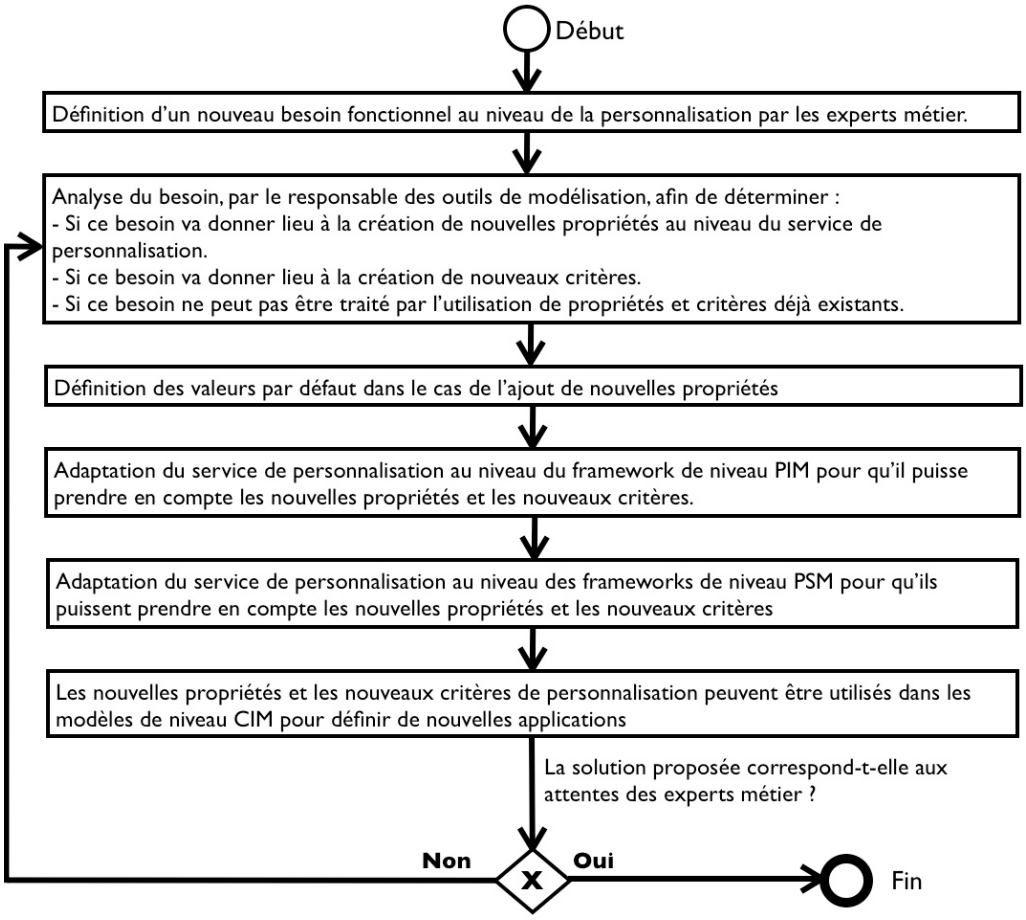


Figure 4.3. Processus utilisé pour l'adaptation du service de personnalisation de niveau PIM en fonction de nouveaux besoins fonctionnels énoncés par les experts métier

- Au niveau PIM, l'évolution du service de personnalisation se traduit par la définition de nouvelles fonctionnalités et/ou l'évolution de fonctionnalités techniques existantes.

  Pour rappel, dans PERCOMOM, la définition des algorithmes directement reliés à la réalisation technique effective de la personnalisation n'est effectuée qu'au niveau du framework de niveau PIM. Ceci permet de modifier ces algorithmes et de les faire évoluer de manière totalement indépendante des modèles d'applications définis au niveau CIM. En fait, la seule contrainte réside dans la capacité des nouveaux algorithmes à prendre en compte les propriétés et critères définis au niveau CIM dans le cadre du service fonctionnel de personnalisation. Etant donnés les différents outils et méthodes de personnalisation existants aujourd'hui (cf. chapitre 2, §2.3), cette prise en compte peut se faire de manière complète ou alors de manière partielle : certains critères étant pris en compte que de manière partielle. Ainsi, la prise en compte des préférences de l'utilisateur pourra se faire différemment en fonction des algorithmes utilisés au niveau CIM et donc pourra aussi se faire de manière incomplète en fonction des capacités respectives de ces algorithmes. Ainsi, suivant l'algorithme utilisé, la personnalisation des contenus en fonction des préférences communautaires pourra se faire de manière totalement différente, pouvant entraîner des résultats différents au niveau de la personnalisation. Par contre, les modèles de niveau CIM associés resteront inchangés.

  Il est important aussi de noter ici que la pertinence et l'efficacité de la personnalisation est fortement dépendante des algorithmes et méthodes utilisées pour réaliser celle-ci techniquement. En partant de ce postulat, nous avons défini au niveau PIM, à travers l'utilisation de règles définies à ce niveau, la possibilité d'utiliser des services physiques de

personnalisation différents en fonction du contexte et/ou du type d'application. Ainsi, il devient, par exemple, possible d'utiliser deux services de personnalisation de niveau PIM différents (c'est-à-dire, de manière simplifiée, deux algorithmes de personnalisation différents en fonction du lieu où se trouve l'utilisateur). La Figure 4.4 présente un exemple d'adaptation, pour le service de personnalisation, des différents frameworks en fonction de la région où se trouve l'utilisateur. Ceci passe par l'utilisation d'une règle de sélection "retourneregion" au niveau du framework de niveau PIM qui permet de retourner la région de l'utilisateur. Si l'utilisateur se trouve dans la région A, on utilise le service de personnalisation "PERSO_A". Dans tous les autres cas, on utilise le service de personnalisation PERSO_B. Si ceci permet la génération de frameworks de niveau PSM spécifiques, au niveau de la personnalisation, la région A par rapport aux autres régions, la règle de niveau PIM devra obligatoirement être reportée au niveau des frameworks de niveau PSM, et notamment au niveau des frameworks spécifiques à chaque type de plateforme. Ceci permettra à l'application finale de pouvoir utiliser les algorithmes de personnalisation appropriés en fonction de la localisation de l'utilisateur.

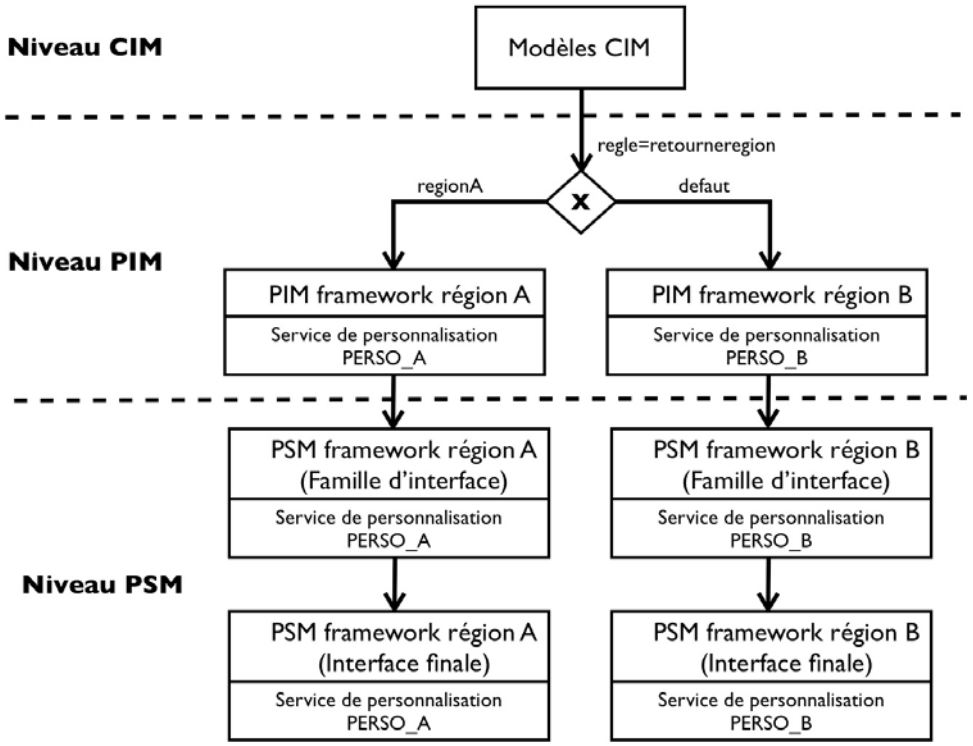


Figure 4.4. Exemple d'utilisation de deux services différents de personnalisation au niveau PIM en fonction de la région où se trouve l'utilisateur au moment de son accès à l'application (définition au niveau de l'architecture globale PERCOMOM)

- Au niveau PSM, l'évolution du service de personnalisation se traduit par une modification éventuelle des frameworks et/ou une adaptation des règles associées aux deux niveaux de framework PSM. Ceci permet, par exemple, de définir une limite pour un type de plateformes qui pourra être modifiée dans le temps en fonction de l'évolution technique de la plateforme comme, par exemple, une résolution d'écran plus importante. Ceci permet aussi de définir au niveau de chaque plateforme le nombre maximal d'informations pouvant être retournées lors de l'exécution d'une requête et/ou d'indiquer des critères de personnalisation supplémentaires spécifiques à chaque plateforme. Ainsi, la Figure 4.5 présente un exemple d'adaptation au niveau PSM d'un service de personnalisation en fonction de la plateforme finale, en l'occurrence dans cet exemple, des PDA dotés ou pas de systèmes de géolocalisation de type GPS. Dans le cas où le PDA est doté d'un GPS, le service de personnalisation est capable d'utiliser ces informations pour retourner des contenus adaptés à la localisation de l'utilisateur. Dans le cas où le PDA n'est pas doté d'un GPS, le service de personnalisation ne peut pas utiliser d'informations de géolocalisation.

De manière générale, la capacité de prise en charge des informations de type GPS est définie au niveau du service de personnalisation de niveau PIM ; par contre, l'activation de cette possibilité se fait au niveau de chaque plateforme et donc au niveau de chaque framework de niveau PSM spécifique à un type de plateforme bien déterminé.

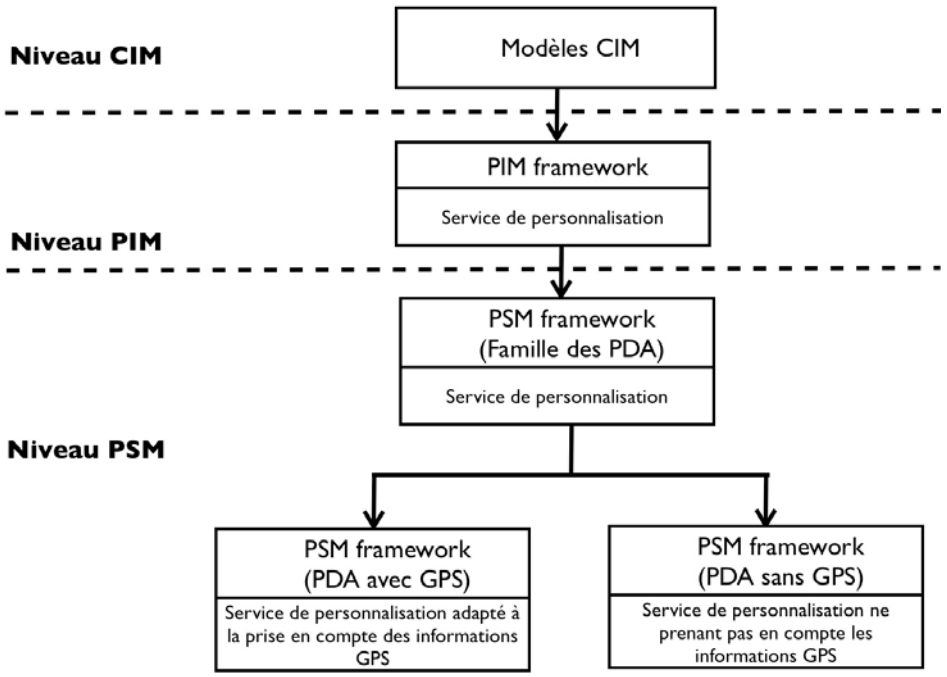


Figure 4.5. Exemple d'adaptation du service de personnalisation au niveau PSM en fonction de la plateforme technique utilisée (soit des PDA avec GPS soit des PDA sans GPS)

4.1.4 Synthèse et discussion

Aujourd'hui, les outils, méthodes et algorithmes de personnalisation ne couvrent souvent que quelques possibilités. Ainsi, si on prend l'exemple de PerSyst (Anli, 2006), du point de vue fonctionnel, le système de personnalisation n'est capable de prendre en compte que les préférences de l'utilisateur et le profil utilisateur. Par contre, l'adaptation à l'environnement n'est pas prise en compte et l'adaptation au concept du domaine est très limitée car la méthode et les algorithmes proposés ne permettent pas de créer des critères de comparaison. De plus, PerSyst n'est pas prévu pour prendre en compte des applications multiples mais pour fonctionner de manière individuelle pour chaque application. Aussi, nous baser sur PerSyst pour définir, au niveau CIM, les besoins fonctionnels en matière de personnalisation nous a semblé trop restrictif. Et, utiliser d'autres solutions de personnalisation comme, par exemple, Piepser (Hoyer et Czogolla 2002) ou Tisoni (Lam et Xie, 2002), présentés dans le chapitre 2, §2.5.2, nous aurait conduit à la même conclusion du fait de leurs limitations respectives. En fait, une des grosses limitations de ces outils, méthodes et algorithmes de personnalisation réside dans le fait que, dans le cadre d'une approche de modélisation conceptuelle des applications, s'ils peuvent correspondre à une réponse technique à des problématiques fonctionnelles, ils ne sont pas directement intégrables dans des modèles de niveau CIM à moins de les encapsuler dans la notion, spécifique à PERCOMOM, de service fonctionnel de niveau PIM.

Pour répondre à cette problématique, nous avons développé, dans PERCOMOM, une approche plus générique de la personnalisation au niveau CIM. Celle-ci est basée uniquement sur les besoins fonctionnels des experts métier en matière de modélisation de la personnalisation et plus sur les solutions techniques pour prendre en charge cette personnalisation. Dans cette approche, la problématique de personnalisation est encapsulée dans un service fonctionnel de niveau PIM dont le comportement peut être modifié, au niveau CIM, à travers l'utilisation de propriétés et/ou de critères.

Ceci permet d'avoir une séparation nette entre les modèles de niveau CIM et les méthodes et approches de personnalisation utilisées pour réaliser, de manière concrète cette personnalisation. Ainsi, il devient possible de faire évoluer ces méthodes et approches de personnalisation de manière

totalement indépendante des modèles conceptuels des applications qui vont les utiliser ; cette indépendance permettant de les adapter dans le temps sans avoir besoin de modifier, au niveau CIM, les applications qui les utilisent. Cette séparation représente, à notre avis, une avancée importante en matière de modélisation des applications personnalisées dans le sens où elle permet de prendre en compte la personnalisation dans les modèles conceptuels de niveau CIM de manière totalement abstraite à travers une vision purement métier de celle-ci.

4.2 La prise en compte du contexte

4.2.1 Introduction

Dans le cadre de la personnalisation, la prise en compte du contexte est un élément important pour apporter à l'utilisateur une personnalisation vraiment adaptée à ses besoins et ses attentes. Et ceci est particulièrement important dans le domaine des transports où, du fait de sa mobilité, l'utilisateur à besoin d'informations différentes, à des moments différents et ceci parfois rapidement. Pour donner, un exemple, lorsqu'un usager arrive dans une gare ferroviaire, il peut accéder à une application lui permettant de consulter les prochains départs. Cette application est identique pour toutes les gares et son contenu est automatiquement adapté en fonction de l'endroit où se trouve l'utilisateur, c'est-à-dire en fonction de la gare. Il est aussi adapté en fonction de ce que le système connaît sur le déplacement en cours de l'usager. La Figure 4.6 montre un exemple de prise en compte du contexte au niveau d'une application finale permettant d'afficher les prochains départs en gare en fonction de l'endroit où se trouve l'utilisateur.

Dans le cadre d'une application, le contexte peut être pris en compte à différents niveaux : au niveau des interfaces, au niveau des processus ou alors au niveau des contenus. Nos travaux ne nous ayant pas permis, pour l'instant, d'explorer toutes les possibilités, nous n'aborderons la prise en compte du contexte dans les applications que sous l'angle de la personnalisation des contenus proposés aux utilisateurs. Dans les parties suivantes, nous allons présenter les différents éléments contextuels pris en compte dans le cadre de PERCOMOM.

Prochains départs de Lille Flandres

Num	Heure	Destination	Situation	Voie
7040	11h00	PARIS NORD		8
848956	11h03	ROUEN		5
19715	11h09	ANVERS		10
843520	11h26	VALENCIENNES		
843814	11h30	CAMBRAI		
841231	11h33	LENS		
843110	11h37	VALENCIENNES		
842415	11h42	DOUAI		
843230	11h54	BETHUNE		
5403	11h56	LENS		

(a)

Prochains départs de Lille Flandres pour Paris Nord

Num	Heure	Destination	Situation	Voie
7040	11h00	PARIS NORD		8
7446	12h00	PARIS NORD		
7050	13h00	PARIS NORD		
7254	13h29	PARIS NORD		
7456	15h00	PARIS NORD		
7058	15h30	PARIS NORD		
7062	16h00	PARIS NORD		
7066	16h30	PARIS NORD		
7070	17h00	PARIS NORD		
7072	17h30	PARIS NORD		

(b)

Figure 4.6. Exemple d'affichage de deux écrans d'une même application d'affichage des prochains départs en gare avec un affichage non personnalisé pour (a) et personnalisé pour (b) en fonction de la prochaine destination de l'usager

4.2.2 Les contextes applicatifs, géographiques et temporels

Dans le cadre d'une application, le premier élément contextuel qui peut être pris en compte est l'application elle-même. Ceci est d'autant plus important que, dans le cadre d'une approche dirigée par les modèles où la réutilisation est, en principe, fortement développée, tous les modèles de niveau CIM peuvent être utilisés par plusieurs applications différentes.

De manière pratique, tenir compte du contexte applicatif, c'est être capable d'avoir, pour un même modèle conceptuel de niveau CIM, un comportement différent en fonction de l'application qui l'utilise. Si ceci permet d'apporter une plus grande souplesse au niveau de la modélisation, c'est aussi au prix d'une perte de généricité au niveau des modèles. Afin, de limiter cet inconvénient, la prise en compte de l'application dans les modèles implique de respecter deux règles :

- L'adaptation au contexte applicatif ne peut se faire qu'à travers l'utilisation d'une restriction applicative qui est une règle métier particulière de type validation (cf. chapitre 3, §3.1.3.5, pour la définition d'une restriction applicative).
- L'adaptation n'est possible au niveau de la catégorie d'applications ou au niveau de l'application même en fonction de ce qui est défini dans la restriction applicative utilisée.

Pour donner un exemple, supposons qu'on veuille utiliser dans plusieurs applications un même élément d'interaction *UIFieldInOut* permettant d'afficher un mot de passe à l'écran. Ce mot de passe devra :

- Être visible en clair dans les applications de type "Administration".
- Être visible de manière cryptée dans les applications qui ne sont pas du type "Administration".

Ceci ne sera possible qu'en ayant une valeur différente au niveau de la propriété "EstCrypté" de l'artefact associé à l'*UIFieldInOut* concerné. Pour faire cela, on utilisera une notation spécifique à PERCOMOM permettant d'introduire une sélection basée sur une restriction applicative permettant de savoir si l'application en cours appartient à la catégorie d'applications "Administration" ou pas (cf. Figure 4.7). Il est important de noter ici que les restrictions peuvent être utilisées dans l'ensemble des modèles utilisés dans le cadre de PERCOMOM de la même manière que les règles métier.

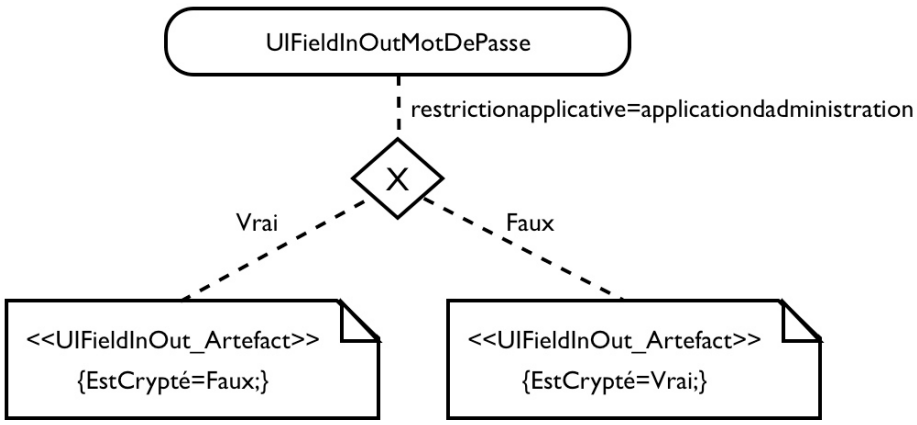


Figure 4.7. Exemple d'utilisation du contexte applicatif dans le cadre de la sélection de propriétés pour un UIElement de type UIFieldOut

La prise en compte du contexte géographique et du contexte temporel peut pour sa part se faire à deux niveaux :

- Au niveau des concepts métier manipulés et, dans ce cas, les informations de contexte (géographiques et temporelles) sont utilisés comme critères de filtrage lors de la sélection des données. Pour cela, il faut que les données manipulées, c'est-à-dire les individus de la classe de l'ontologie de domaine (OD), possèdent les informations géographiques et temporelles nécessaires pour que ce filtrage puisse effectivement se faire. L'association de ces informations peut se faire de deux manières différentes :

- De manière explicite. Dans ce cas, chaque individu du concept métier considéré, de la classe de l'ontologie considérée, possède une propriété permettant d'associer, de manière explicite par les utilisateurs, des informations géographiques et/ou temporelles. De manière pratique, ces informations sont stockées sous la forme d'un ou de plusieurs domaines de validité (cf. chapitre 3, §3.1.3.5).
- De manière implicite. Dans ce cas, chaque individu du concept métier considéré possède une propriété dans laquelle il est possible d'associer des domaines de validité. Cette association ne se fait pas explicitement par l'utilisateur mais de manière implicite par l'application à travers l'utilisation d'actions (modèle BM1).

Pour donner un exemple, supposons que dans une ontologie de domaine métier associé aux besoins d'un compagnie de transport ferroviaire comme la SNCF, on définisse un concept métier, une classe de l'ontologie de domaine (OD) pour définir chaque train circulant sur l'ensemble des voies et qu'on définisse, pour chaque individu de cette classe une propriété permettant d'indiquer la zone géographique dans laquelle chaque train va circuler. Il devient alors envisageable de pouvoir rechercher l'ensemble des individus de la classe "Train", pour une zone géographique bien déterminée. Ceci se fait en utilisant un critère de sélection particulier défini comme suit (ce critère s'ajoute aux critères définis dans le chapitre 4, §4.1.2.2) :

restrictionGeographique **==** *nomrestrictiongéographique*

Avec :

- *nomrestrictiongéographique* : le nom de la restriction géographique qu'on souhaite utiliser comme critère lors de la sélection d'individus dans l'ontologie de domaine.

Par exemple, si on suppose qu'on a défini une restriction géographique "regionnord" permettant de définir une zone géographique particulière et qu'on désire avoir tous les trains de type "TGV" circulant dans cette région, on devra alors définir le critère de sélection suivant :

```
<selection nom="trainsdanslaregionnord">
   <critereselection classeOD="Train" proprieteOD="TypeDeTrain"
   operateur="==" critere="TGV"/>
   <critereselection restrictionGeographique="regionnord"/>
</selection>
```

Si on souhaite utiliser une restriction temporelle comme critère de sélection, on devra utiliser, au niveau du critère de sélection la définition suivante :

restrictionTemporelle **==** *nomrestrictiontemporelle*

Avec :

- *nomrestrictiontemporelle* : le nom de la restriction temporelle qu'on souhaite utiliser comme critère lors de la sélection d'individus dans l'ontologie de domaine

- Au niveau des modèles de niveau CIM et, dans ce cas, les informations de contexte sont vues comme des critères de sélection particuliers. Pour rappel (cf. chapitre 3), la définition de contextes géographiques et temporels peut se faire à travers l'utilisation de restrictions. Par exemple, dans une application permettant d'afficher des informations transport, il pourrait être pertinent d'afficher le temps restant avant la prochaine correspondance, le prochain départ, de l'usager lorsque celui-ci se trouve au niveau d'un arrêt (arrêt de bus, gare, station de métro,…) en plus des informations transports générales ; alors que si l'usager n'est pas à un arrêt, on ne lui afficherait que les informations transports générales. La Figure 4.8 montre un extrait du processus métier qui pourrait être utilisé dans le cadre de ce type d'application.

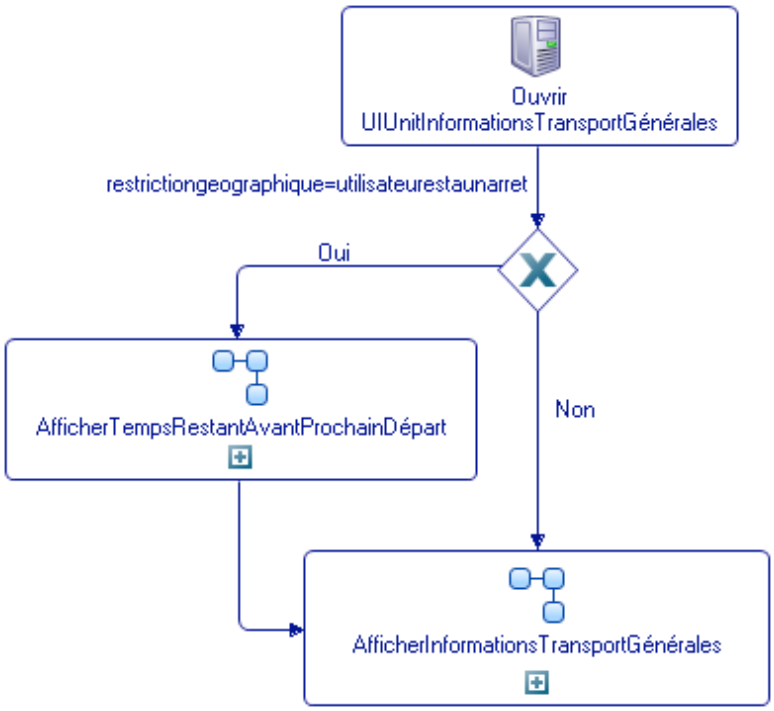


Figure 4.8. Exemple d'utilisation d'une restriction géographique au sein d'un processus métier (modèle IM1) utilisé pour afficher des informations transport

Il est important de noter que dans le cadre de PERCOMOM, il est possible d'utiliser, dans les modèles de niveau CIM, des assemblages de restrictions à travers l'utilisation de la notion de domaine de validité définie dans le chapitre 3, §3.1.3.5. Ainsi, il est possible de définir un domaine de validité qui ne retourne un résultat vrai que si l'utilisateur manipule une application de type "ServicesMultimédias" dans une gare le lundi matin.

4.2.3 Les autres éléments contextuels

Dans le cadre de PERCOMOM, les principaux éléments contextuels pris en compte sont le temps, l'espace et le type d'application en cours d'utilisation. Si ceci permet de résoudre beaucoup de problématiques au niveau d'une application dans le domaine des transports, il en découle aussi une vision trop restrictive de la notion de contexte.

Ainsi, si on ne prend en compte que l'environnement physique dans lequel évolue l'application et si on ne tient pas compte de l'utilisateur qui sera traité dans le chapitre 4, §4.3, un certain nombre d'autres éléments peuvent avoir une action sur l'application comme, par exemple :

- Le bruit ambiant qui peut entraîner, au niveau de l'application, une augmentation du volume sonore, un changement de type d'interface qui passe de sonore à visuel par exemple, un filtrage plus important des informations entrantes dans le cadre d'une application utilisant de la reconnaissance vocale ou d'une application de type téléphonie, etc.
- La luminosité ambiante qui peut entraîner un changement au niveau des couleurs utilisées pour l'interface comme, par exemple, un passage des caractères en rouge sur fond noir pour faciliter la vision nocturne de l'utilisateur, etc.
- La vitesse de déplacement de l'utilisateur qui peut nécessiter une adaptation de l'interface pour en faciliter la lecture lorsque l'utilisateur court lorsqu'il est en retard pour attraper sa correspondance, l'envoi d'un message d'alerte lorsque l'utilisateur dépasse la vitesse autorisée alors qu'il est en train de conduire un véhicule, etc.

- L'analyse des éléments entourant l'utilisateur via des technologies de type RFID (Radio Frequency IDentification) qui permettent de récupérer des informations dans l'interface en cours ou d'exécuter automatiquement des actions particulières.

De manière plus globale, l'environnement physique, dans lequel va être utilisée l'application peut être vu comme un ensemble de sous-éléments permettant chacun de caractériser une partie bien précise de l'environnement physique globale (cf. Figure 4.9)

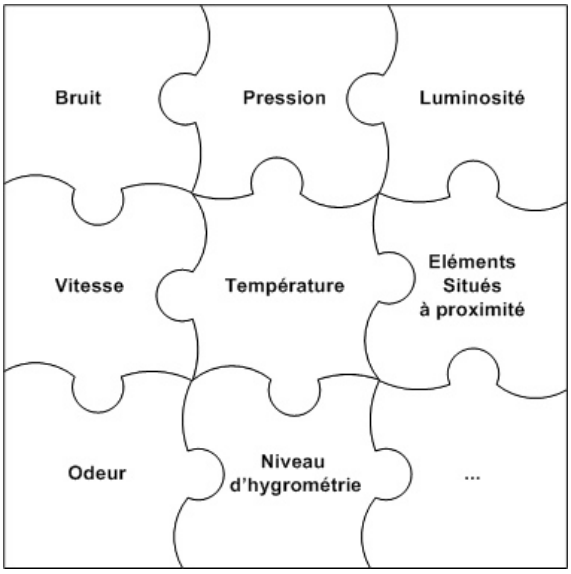


Figure 4.9. Représentation d'une partie des éléments constituant l'environnement physique dans lequel évolue une application finale (seuls quelques éléments sont représentés sur cette figure)

Si les éléments présentés dans la Figure 4.9 semblent être fort différents les uns des autres, ils présentent néanmoins tous un certain nombre de points communs au niveau de leur prise en compte :

- Celle-ci ne peut se faire qu'à travers l'utilisation de capteurs (transformation, par exemple, du niveau sonore extérieur en dB).
- Il est impossible de prévoir avec certitude le moment où leurs valeurs vont changer (le niveau de luminosité peut, par exemple, changer brutalement lorsque l'utilisateur passe de l'ombre au soleil).
- Lorsque qu'un changement survient, au niveau de l'environnement physique, sa durée n'est pas limitée : il est actif tant que l'événement est actif (par exemple, il sera possible de mesurer une vitesse de déplacement de l'utilisateur tant que celui-ci est en déplacement).

Tous ces éléments étant très fortement liés aux capacités de la plateforme technique manipulée par l'utilisateur et à l'environnement dans lequel l'utilisateur évolue, ils ne peuvent être directement pris en compte dans les modèles conceptuels de niveau CIM des applications. Or, il peut être parfois important de les prendre en compte du point de vue métier. Ainsi, si on développe une application pour des personnes chargées de vérifier des conduites de gaz, il est plus que pertinent de pouvoir les avertir d'une fuite de gaz si jamais le capteur, spécifique à ce type de détection, placé sur la plateforme technique utilisée par l'application en détecte une.

La question qui se pose alors est de savoir comment les prendre en compte au niveau CIM. La réponse proposée dans le cadre de PERCOMOM est de passer par les modèles d'événements (EM3). Ainsi, si on prend comme le niveau sonore extérieur mesuré en dB, celui-ci pourra, par exemple, être associé à quatre types d'événements différents :

- Un événement de type "niveausonorebas" si le niveau sonore est inférieur à 40 dB
- Un événement de type "niveausonoremoyen" si le niveau sonore est entre 40 et 75 dB.
- Un événement de type "niveausonoreélevé" si le niveau sonore est entre 76 et 100 dB.
- Un événement de type "niveausonoredangereux" si le niveau sonore est supérieur à 100 dB

Dans le cadre d'une application, il ne pourra y avoir qu'un seul événement, que nous qualifierons de contextuel, relatif au niveau sonore extérieur, actif en même temps.

Pour donner un exemple, on pourrait supposer, au niveau d'une application, qu'on envoie un message d'avertissement à l'utilisateur dans le cas ou le niveau sonore ambiant serait supérieur à 100 dB. Pour cela, on définira, par exemple, dans l'application un processus métier, qui n'est activé que par un événement de type "niveausonoredangereux", qui exécute un sous-processus "Envoi message d'alerte" avant de se terminer. Le modèle IM1 associé à ce processus est donné dans la Figure 4.10.

Figure 4.10. Exemple d'utilisation d'un événement contextuel pour déclencher automatiquement le démarrage d'un processus métier dès que celui-ci survient

Comme les autres événements, les événements contextuels sont aussi utilisables, dans l'ensemble des modèles conceptuels, de la même manière que les règles métier. Ainsi, si on suppose qu'on désire, au niveau d'un processus métier, exécuter un sous-processus A si l'événement "niveausonoreélevé" est actif et un sous-processus B si l'événement "niveausonoreélevé" est inactif, on devra définir le processus métier comme indiqué dans la Figure 4.11.

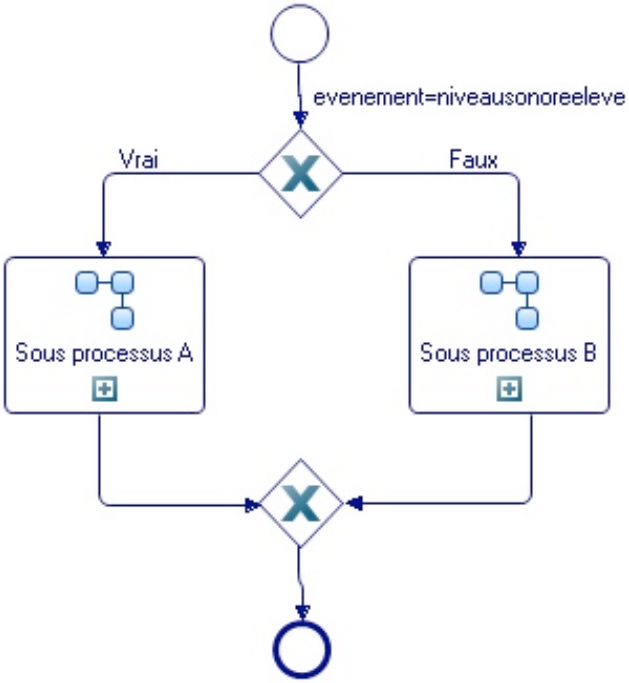


Figure 4.11. Exemple d'utilisation d'un test sur la validité d'un événement contextuel au sein d'un processus métier (modèle IM1)

La prise en charge des événements contextuels, au niveau CIM, étant maintenant définie, la question qui se pose est de savoir si la définition de tous les types d'événements possibles est vraiment pertinente au niveau d'une approche globale de type MDA.

4.2.4 Prise en compte du contexte et approche MDA

Dans le cadre d'une approche MDA, telle que préconisée par l'OMG, l'adaptation des applications à l'environnement ne peut se faire qu'au niveau des modèles de niveau CIM ; les modèles PIM et PSM étant obtenus par transformation de modèles à partir des modèles de niveau CIM. Si ceci est tout à fait normal dans le cadre de l'approche MDA préconisée par l'OMG, ce n'est pas forcément pertinent du point de vue métier. Ainsi, si on suppose que l'utilisateur manipule une plateforme technique de consultation capable de s'adapter automatiquement à la luminosité extérieure et que cette adaptation à été déterminée lors de la conception de la plateforme et qu'elle est non modifiable et qu'il est impossible de connaître quand cette adaptation est effectuée, celle-ci ne pourra alors pas être prise en compte dans les modèles de niveau CIM (les applications ne pouvant avoir aucune action sur cette adaptation automatique à la luminosité). Par contre, s'il est possible de récupérer une information sur cette adaptation à travers un événement et si celui-ci peut être utile du point de vue métier (sa prise en compte pouvant avoir une influence sur le déroulement de l'application), alors il devra être défini au niveau CIM à travers le modèle d'événement (EM3).

Aussi, dans le cadre de PERCOMOM, nous proposons une prise en charge de chaque élément du contexte (bruit, luminosité, …) différenciée en fonction de leur impact au niveau du modèle métier de l'application. Ainsi, si un événement contextuel peut avoir une influence du point de vue métier, il sera pris en compte au niveau CIM de l'architecture. Par contre, si son influence se résume à une adaptation automatique de l'IHM au niveau de la plateforme, il ne sera pris en compte qu'au niveau PSM. Tout ceci est résumé dans la Figure 4.12. La notion de stimulus représente le changement au niveau du contexte qui peut être perçu par le capteur.

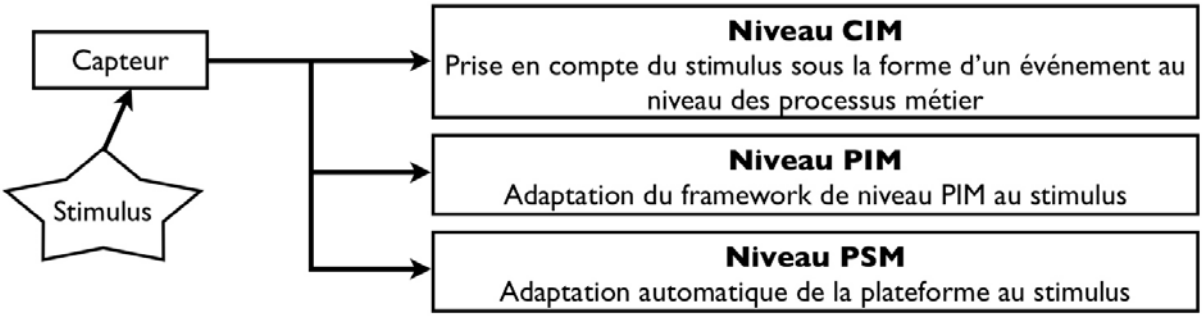


Figure 4.12. Prise en compte d'un stimulus (modification du contexte externe) au niveau CIM, PIM et PSM dans le cadre de notre approche

Ceci étant, si la prise en compte des éléments contextuels au niveau de l'architecture proposée dans PERCOMOM se fait dans les trois niveaux de l'architecture MDA, elle ne se fait néanmoins pas de la même façon à chaque niveau (cf. Tableau 4.5).

Tableau 4.5. Les différents niveaux de prise en charge du contexte dans l'architecture technique proposée dans PERCOMOM

Niveau MDA	Principe de prise en charge	Exemples d'adaptation
CIM	• Caractérisation du contexte à travers les modèles géographiques (EM1) et/ou les modèles des événements (EM3). Seules les informations contextuelles pouvant avoir un impact au niveau métier sont prises en compte. • Utilisation des informations sur le contexte dans le cadre de restrictions et/ou de règles métier.	Définition dans un *UIUnit* de deux affichages différents pour les informations sur les prochains départ suivant que l'utilisateur se trouve dans une gare où à un arrêt de bus (dans ce dernier cas, l'affichage du numéro de quai est inutile).
PIM	• Adaptation des services fonctionnels et techniques de niveau PIM à travers l'utilisation du moteur de règles de niveau PIM (cf. Figure 4.4 du chapitre 4, §4.1.3).	Définition de deux systèmes différents de personnalisation des informations en fonction de l'endroit où se trouve l'usager : • Un système situé sur un serveur central lorsque l'usager n'est pas dans un véhicule. • Un système situé sur un serveur embarqué lorsque l'usager se trouve dans un véhicule
PSM	• Définition et caractérisation des capteurs et des détecteurs ainsi que des caractéristiques physiques particulières de chaque plateforme. Ceci permet de savoir ce qui peut être mesuré, les valeurs qui peuvent être prises en compte (sensibilité et plage de valeurs des capteurs) et aussi les valeurs qui seront utilisées par défaut si la plateforme ne permet pas de prendre en compte certains éléments du contexte. Ceci permet aussi de définir l'ensemble des événements contextuels (modèles EM3) au niveau CIM. • L'adaptation de niveau PSM, qui correspond à une adaptation automatique des plateformes de consultation, se fait à travers la définition de règles de niveau PSM et à travers l'utilisation de ces règles dans les frameworks de niveau PSM.	• Utilisation des informations sur la luminosité ambiante pour adapter automatiquement la plateforme (adaptation de la luminosité de l'écran, rétro-éclairage du clavier, etc.). • Adaptation des frameworks de niveau PSM à chaque type de capteur disponible sur la plateforme ; chaque plateforme pouvant avoir des capteurs et des façons de mesurer les informations provenant du contexte différents (exemple : utilisation d'un capteur piezoélectrique pour mesurer la pression sur une plateforme et d'un capteur de type transformateur différentiel sur une autre plateforme).

Il est important de noter ici que le choix d'utiliser un élément contextuel à un niveau précis de l'architecture dépend principalement du domaine métier associé aux applications. Ainsi, un événement contextuel pris en charge au niveau CIM pour un domaine métier particulier pourra n'être pris en charge qu'au niveau PIM ou PSM dans un autre domaine métier et réciproquement. De même, la répartition des événements dans les différents niveaux pourra évoluer dans le temps en fonction des besoins métier mais aussi des possibilités et des besoins au niveau des frameworks PIM et PSM. Enfin, certains événements contextuels pourront être utilisés dans plusieurs niveaux en même temps. Ainsi, si on reprend l'exemple de l'adaptation à la luminosité, la détection d'une faible luminosité pourra entraîner une adaptation automatique de la plateforme au niveau PSM mais aussi une adaptation de l'IHM à travers la prise en compte de l'information sur la luminosité dans un modèle métier ou dans un modèle statique d'interface.

4.2.5 Synthèse et discussion

La prise en compte du contexte est un élément important de la personnalisation des applications qui permet d'adapter celles-ci à l'environnement dans lequel évolue l'utilisateur (cf. chapitre 2 § 2.3). En effet, à travers cette adaptation, on cherche à apporter à l'utilisateur, à chaque instant, soit les informations et services les plus pertinents soit les interfaces homme-machine les plus adaptées à ses besoins. Par exemple, si un usager vient d'arriver en train dans une gare et qu'il se déplace rapidement, on pourra en conclure qu'il est pressé et qu'il doit très certainement prendre une correspondance. Si en plus on connaît le déplacement qu'il doit effectuer, par exemple, par la connaissance de son agenda, on pourra alors adapter l'interface de manière à ne lui présenter que les informations utiles lui permettant de prendre sa correspondance et ceci de manière adaptée (cf. Figure 4.13).

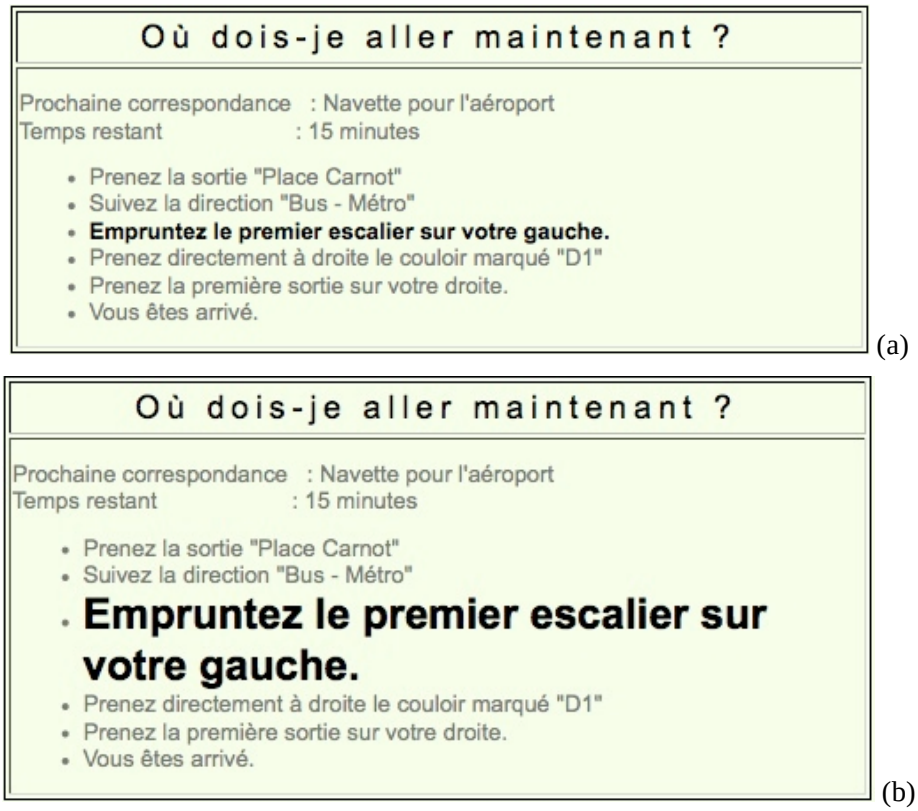


Figure 4.13. Exemple d'adaptation d'une application à la vitesse de déplacement d'un utilisateur : déplacement normal (a) et déplacement rapide (b)

Au niveau CIM, cette adaptation présentée dans la Figure 4.13 est définie au niveau du modèle statique d'interaction (IM2) associé aux deux écrans présentés. Dans ce modèle, l'ensemble des informations sur le chemin à suivre ("Prenez la sortie…..Vous êtes arrivé") sont affichées grâce à un *UIFieldInOut* qui porte le nom de "UIFieldInOutCheminASuivre" auquel on associe un artefact différent en fonction d'un événement "deplacementrapide" qui permet de savoir si l'utilisateur est en train de courir ou pas. Dans ces artefacts, on utilise deux propriétés spécifiques aux *UIFieldInOut* qui sont :

- *EstUneListe* : qui permet d'indiquer que l'ensemble des informations contenues dans l'*UIFieldInOut* vont être affichées sous la forme d'une liste.
- *ElementSelectionne* : qui permet d'indiquer comment l'élément actuellement sélectionné dans la liste doit être présenté ("Normal" s'il doit être présenté comme les autres éléments de la liste, "Important" si on doit le faire ressortir par rapport aux autres éléments de la liste).

La partie du modèle IM2 correspondant est présentée dans la Figure 4.14.

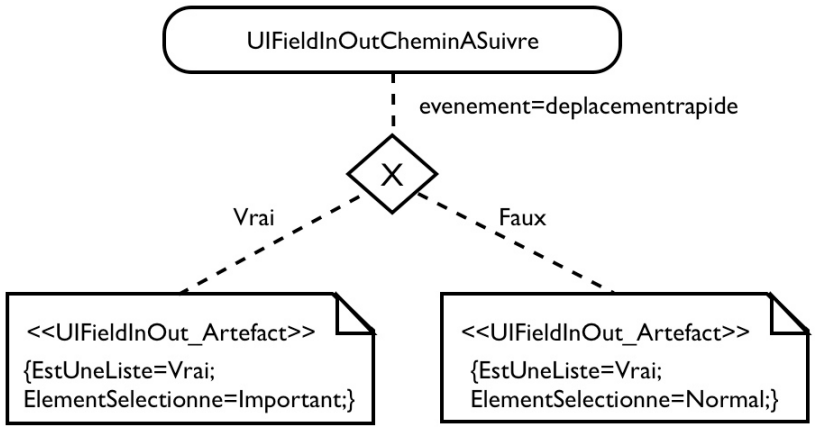


Figure 4.14. Présentation de du modèle IM2 associé à l'exemple de la Figure 4.13 sensible à un événement contextuel relatif à la vitesse de déplacement de l'utilisateur

Comme nous l'avons vu dans cette partie, PERCOMOM est capable, à travers notamment une gestion spécifique des événements contextuels, de prendre en compte l'environnement physique dans lequel évoluent les applications et ceci dans les différents niveaux de l'architecture (CIM, PIM et PSM). Par rapport à une approche classique de type MDA, ceci présente un certain nombre d'avantages mais aussi d'inconvénients dont les principaux sont résumés dans le Tableau 4.6.

Tableau 4.6. Comparaison des avantages et inconvénients de la prise en charge du contexte dans PERCOMOM par rapport à l'approche MDA préconisée par l'OMG

Critère	Avantages	Inconvénients
Modélisation de niveau CIM	• Simplification des modèles car seuls les éléments contextuels pertinents sont pris en compte. • L'utilisation des règles métier et des restrictions permet une adaptation rapide des applications sans avoir besoin de modifier les autres modèles.	• L'évolution dans le temps peut être complexe si on doit prendre en compte de nouveaux événements contextuels qui étaient avant pris en compte au niveau PIM ou PSM. • Pour un même élément contextuel (par exemple le bruit extérieur), le nombre d'événements différents pouvant être pris en compte peut évoluer en fonction des besoins métier (par exemple, définition de 4 niveaux de bruit au début puis définition de 9 niveaux de bruit par la suite) ; ce qui peut obliger à revoir toutes les applications existantes
Modélisation de niveau PIM	• La prise en compte du contexte au niveau PIM permet d'apporter plus de souplesse aux services fonctionnels mais aussi de simplifier les modèles de niveau CIM (l'adaptation n'apparaissant pas dans les modèles de niveau CIM)	La définition des événements contextuels à plusieurs niveaux : • Ne permet pas d'avoir facilement une vision d'ensemble de ceux-ci (chaque événement pouvant être traité dans les différents niveaux). • Complexifie les modèles de niveau PIM et PSM. • Ne permet pas de s'assurer de la cohérence des adaptations lorsque celles-ci se font à plusieurs niveaux.
Modélisation de niveau PSM	• La prise en compte du contexte au niveau PSM permet de tenir compte des adaptations automatiques des plateformes à leur environnement d'utilisation et des spécificités de chaque type de plateforme.	

Dans le cadre de la prise en compte des éléments contextuels dans le cadre de la personnalisation des informations, il nous reste un élément important à prendre en compte qui est l'utilisateur lui-même.

4.3 La prise en compte de l'utilisateur

4.3.1 Introduction

Pour pouvoir, dans une application, fournir des informations et des services pertinents à l'usager, il faut être capable de l'identifier et de le caractériser. Ceci est d'autant plus vrai dans les applications personnalisées, celles-ci devant pouvoir connaître les préférences et centres d'intérêt de l'utilisateur afin de lui fournir des informations et services adaptés à ses besoins et à ses attentes. Dans le cadre de PERCOMOM, la modélisation des utilisateurs se fait à travers l'utilisation des modèles sociaux et plus particulièrement à travers l'utilisation des modèles de profil utilisateur (SM1) et des modèles d'organisation (SM2) (cf. chapitre 3, §3.1.2.2).

Dans le profil utilisateur, les informations sont de deux types différents :

- Des informations explicites fournies par l'utilisateur qui permettent de le caractériser (âge, profession, etc.),
- Des informations implicites, collectées par le système, qui permettent de définir les préférences de l'utilisateur ; ces dernières étant reliées à l'ontologie de domaine comme nous le verrons dans le chapitre 4, §4.3.3.

Mais, si on veut être capable d'adapter les applications à l'utilisateur, ceci n'est pas suffisant. Il faut changer la façon dont on conçoit les applications en adoptant une approche des interactions centrée sur l'utilisateur et tout d'abord sur les buts qu'il poursuit lorsqu'il utilise chaque application (Cooper *et al.,* 2007). Il faut aussi trouver d'autres façons d'interagir avec lui de manière à permettre une adaptation plus rapide des applications et pour cela, une des solutions possibles est d'être capable de reconnaître les émotions de l'utilisateur (Cowie *et al.,* 2001) ; ceci constituant d'ailleurs un domaine de recherche difficile et très d'actualité (Brave et Noss, 2002).

Aujourd'hui, si de nombreux travaux tentent d'apporter des éléments de réponse pour ces différentes problématiques, peu, à notre connaissance, traitent du problème spécifique au domaine des transports de la prise en compte du vécu des temps de transports par les usagers ; c'est-à-dire de la façon dont ils utilisent leur temps pendant leurs déplacements (cf. chapitre 5, §5.2).

Au niveau de la prise en compte de l'utilisateur dans le cadre de la personnalisation dans PERCOMOM, nous allons, dans le cadre de ce mémoire, présenter deux éléments importants qui sont la prise en compte des préférences de l'utilisateur et la prise en compte de l'expérience de l'utilisateur vis-à-vis de l'application.

4.3.2 La prise en compte des préférences de l'utilisateur

L'objectif de PERCOMOM n'étant pas de définir de nouvelles méthodes ou de nouveaux algorithmes pour prendre en compte les préférences utilisateur, nous avons utilisé, dans le cadre de nos travaux, comme méthode et outil de référence pour la caractérisation des préférences de l'utilisateur ceux proposés dans le cadre de la méthode PerMet et de l'outil PerSyst (Anli, 2006). Dans cette méthode et dans cet outil, les préférences sont stockées au niveau du profil de l'utilisateur. Elles sont regroupées en catégories et chaque élément de la catégorie reçoit une pondération qui permet de déterminer son importance relative par rapport aux autres éléments de la catégorie.

En partant de cette approche, PERCOMOM, à travers l'utilisation d'une ontologie de domaine permet d'apporter une finesse supplémentaire au niveau de la définition des préférences. Ainsi, celles-ci ne sont pas définies uniquement par rapport à leur catégorie d'appartenance mais aussi par rapport au contexte métier dans lequel elles sont utilisées.

Pour donner un exemple, on suppose que dans l'ontologie d'un domaine métier, on définisse une classe "personne célèbre", une classe "écrivain" et une classe "chanteur". Une personne célèbre peut alors être soit un écrivain soit un chanteur ou alors les deux ; chaque caractérisation étant indépendante de toute autre caractérisation. Ces deux caractérisations représentant deux types de relations différentes, il devient alors possible, pour une même personne célèbre, d'avoir des préférences différentes en fonction de son type d'activité artistique (écrivain ou chanteur). Ainsi, un utilisateur

pourrait indiquer qu'il est un fan de Madonna lorsque celle-ci est une chanteuse mais qu'il n'apprécie pas les livres qu'elle écrit (des livres pour enfants dans ce cas précis). Ceci s'exprime au niveau du profil de la manière suivante (pour chaque contexte, la valeur totale des niveaux de préférence ne peut être supérieure à 1) :

```
<préférences>
    <classe="PersonneCélèbre">
        <contexte classe="Chanteur">
            <individu catégorie="identifié" identifiant="Madonna" niveaupréférence="0.8"/>
            <individu catégorie="générique" identifiant="Autre" niveaupréférence="0.2"/>
        </contexte>
        <contexte classe="Ecrivain">
            <individu catégorie="identifié" identifiant="Madonna" niveaupréférence="0.1"/>
            <individu catégorie="générique" identifiant="Autre" niveaupréférence="0.9"/>
        </contexte>
    </classe>
</préférences>
```

Dans cet exemple, l'utilisation de la catégorie "générique" permet d'indiquer une valeur générale pour tous les individus non identifiés de manière individuelle. Ainsi, dans notre exemple, tout autre chanteur que Madonna aurait un niveau de préférence égal à 0.2. Il est aussi possible de définir, pour un même individu, une valeur identique pour chaque contexte en définissant un tag *individu* directement sous le tag *classe*.

Dans PERCOMOM, la mise à jour des valeurs des préférences de l'utilisateur se fait uniquement à travers les applications. Cette mise à jour peut se faire de manière explicite en demandant à l'utilisateur d'indiquer lui-même ses préférences ou alors de manière implicite, c'est-à-dire automatique, à travers des actions au niveau CIM ou à travers les services fonctionnels de personnalisation de niveau PIM.

Au niveau, CIM, ceci pourra se faire de deux manières différentes :

- Dans les modèles d'action (BM1) où les données relatives aux préférences de l'utilisateur pourront être manipulées à l'aide d'un pseudo langage de traitement de données. Par exemple, si on désire associer la même valeur à tous les chanteurs qui sont des personnes célèbres, le code de traitement de données serait le suivant :

```
// Définition des variables utilisées
// Définition d'un compteur permettant de faire une boucle sur l'ensemble des individus
// associés à la classe PersonneCélèbre et au contexte Chanteur
Entier compteur;
// Variable permettant de stocker la valeur à affecter à chaque individu
Flottant valeurPreference;
// Variable permettant de manipuler une préférence en particulier
Preference preferenceEnCours;
// Calcul de la valeur de préférence à affecter à chaque individu pour qu'il ait chacun la
// même valeur de préférence
ValeurPreference = 1/preference.PersonneCelebre.Chanteur.Nombre();

// Boucle permettant d'affecter à chaque individu la même valeur de préférence
Pour compteur=1 a compteur=preference.PersonneCelebre.Chanteur.Nombre()
    // Récupération de l'individu en cours
    preferenceEnCours = preference.PersonneCelebre.Chanteur(compteur);
    // Affectation de la valeur de préférence à l'individu en cours de traitement
    preferenceEnCours.NiveauPreference = ValeurPreference;
FinPour
```

- Dans des modèles d'interaction, pour créer des interfaces permettant de lire et de modifier des valeurs de préférences. A titre d'exemple, la Figure 4.15 montre comment il est possible d'associer la valeur d'une préférence à un UIFieldInOut en utilisant des propriétés

au niveau de l'artefact associé à l'UIFieldInOut. Dans le cas présent, on affiche la valeur de la préférence associée à la personne célèbre Madonna lorsqu'elle est une chanteuse.

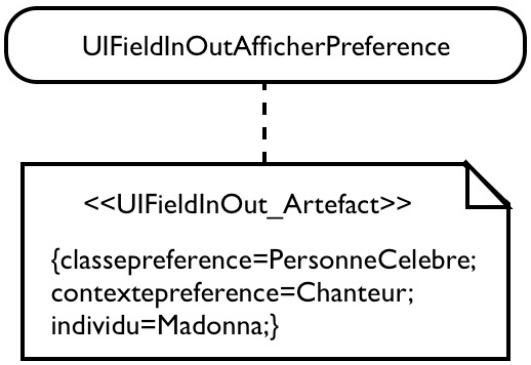


Figure 4.15. Utilisation de propriétés spécifiques dans l'artefact d'un UIFieldInOut pour permettre l'affichage d'une préférence

Au niveau PIM, la gestion des valeurs des préférences se fera directement par le service fonctionnel de personnalisation. PERCOMOM se voulant indépendant, dans son approche, des méthodes et algorithmes de personnalisation utilisés au niveau PIM, nous ne formulons aucune préconisation particulière pour ce type de prise en charge si ce n'est de garder une cohérence au niveau des différentes valeurs de l'ensemble des préférences. Cette indépendance vis-à-vis des méthodes et algorithmes de personnalisation implique aussi que les résultats obtenus à l'aide des modules fonctionnels de personnalisation utilisés pourront être différents en fonction de chaque module. Ainsi, un module de personnalisation pourra utiliser les préférences uniquement pour trier les résultats obtenus suite à l'exécution d'une requête alors qu'un autre pourra utiliser les préférences pour ne retourner que les informations avec un haut niveau de préférence ; les autres informations n'étant pas retournées.

Dans le cadre de la personnalisation des informations, un autre élément qui peut être pris en compte au niveau de la modélisation de l'utilisateur est ce qui est généralement appelé son expérience.

4.3.3 La prise en compte de l'expérience de l'utilisateur vis-à-vis de l'application

Avant de préciser comment la notion d'expérience est prise en compte dans le cadre de PERCOMOM, il convient d'abord d'en préciser la définition. Ainsi, dans PERCOMOM, l'expérience n'est prise en compte qu'à deux niveaux :

- Au niveau du profil utilisateur où il est possible de définir, pour chaque utilisateur, un niveau d'expérience parmi les quatre niveaux suivants : "Utilisateur occasionnel", "Utilisateur novice", "Utilisateur régulier" et "Utilisateur expert". Le choix de quatre niveaux a été fait de façon à permettre une différenciation simple des niveaux d'expérience. Par la suite, une analyse approfondie devra être réalisée afin de valider si ce nombre de niveaux est suffisant ou pas en fonction du type d'application.
- Au niveau du service fonctionnel de personnalisation de niveau PIM où la prise en charge de l'expérience utilisateur est gérée entièrement par celui-ci. À ce niveau, l'expérience utilisateur est utilisée par le service de personnalisation comme critère supplémentaire pour affiner les résultats retournés à l'utilisateur. La façon dont cette expérience est prise en compte étant fortement dépendante des algorithmes et méthodes de personnalisation utilisés, elle ne sera volontairement pas détaillée dans le cadre de ce mémoire car ce n'est pas le sujet principal de nos recherches.

PERCOMOM ayant pour but de permettre la modélisation, dans un même outil, d'un grand nombre d'applications différentes, la question se pose de savoir comment gérer des niveaux d'expérience différents en fonction du type d'application. En effet, un utilisateur peut être novice sur une application

et expert sur une autre dans un même domaine métier. La solution proposée dans le cadre de PERCOMOM est de pouvoir associer, dans le modèle de profil de l'utilisateur (SM1), à chaque niveau d'expérience, des domaines de validité applicative. Ainsi, si on suppose qu'un utilisateur est qualifié d'expert pour toutes les applications de type "InformationVoyageur" mais qu'il est qualifié de novice pour toutes les applications de type "Administration", on aurait alors, au niveau du profil utilisateur, la définition suivante :

```
<expérience>
    <niveau nom="UtilisateurExpert">
        <typeapplication catégorie="InformationVoyageur"/>
    </niveau>
    <niveau nom="UtilisateurRégulier">
    </niveau>
    <niveau nom="UtilisateurNovice">
        <typeapplication catégorie="Administration"/>
    </niveau>
</expérience>
```

Dans cet exemple, si le niveau d'expérience pour une application donnée n'est pas indiqué, le niveau "Utilisateur régulier" sera automatiquement sélectionné pour cette application. Dans le cadre de la modélisation conceptuelle des applications, le niveau d'expérience faisant partie du profil de l'utilisateur (SM1), il pourra être utilisé comme un critère dans l'ensemble des autres modèles conceptuels.

Cette expérience, comme tout élément du profil, pourra ensuite être utilisée dans les autres modèles de niveau CIM pour adapter le comportement de l'application à l'expérience de l'utilisateur. Ainsi, la Figure 4.16 présente un modèle de processus métier (modèle de type IM1) dans lequel des processus sont exécutés de manière conditionnelle en fonction du niveau d'expérience de l'utilisateur défini dans son profil (modèle SM1). Dans le cas où c'est un utilisateur expert, le sous-processus A sera exécuté. Dans tous les autres cas, c'est le sous-processus B qui sera exécuté.

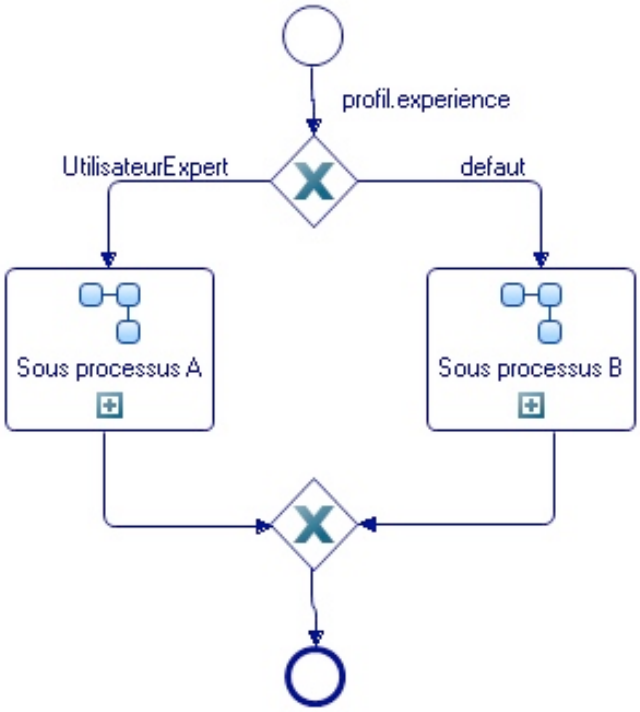


Figure 4.16 Exemple d'utilisation de la notion d'expérience utilisateur pour sélectionner un sous-processus à exécuter au sein d'un processus métier (modèle IM1)

4.3.4 Synthèse et discussion

Dans PERCOMOM, la modélisation de l'utilisateur dans le cadre de la personnalisation présente les particularités suivantes :

- Elle a été prévue à l'origine pour permettre de modéliser un grand nombre d'applications et ceci de manière différenciée à travers une prise en compte du contexte applicatif. Pour donner un exemple, l'expérience de l'utilisateur vis-à-vis de l'usager peut être différente en fonction de chaque catégorie d'applications.
- La notion de préférence est basée sur une ontologie de domaine applicatif ce qui permet d'associer à chaque préférence un contexte sémantique de validité.
- Elle est extensible au niveau de la prise en compte du contexte du moment que ces évolutions correspondent à un besoin métier. Ainsi, il est possible, par exemple, d'envisager d'associer à chaque élément du profil de l'utilisateur un domaine de validité temporel ou géographique si cette association a un sens du point de vue métier.
- Son utilisation n'est en théorie pas limitée au domaine des transports.
- Elle est capable de prendre en compte la notion de préférence utilisateur et de niveau d'expérience de l'utilisateur au niveau CIM et c'est une de ces grosses différences par rapport à toutes les autres solutions existantes.

Par contre, par rapport à une approche "idéale" elle présente, dans l'état actuel de nos recherches, les manques suivants :

- Elle ne prend pas en compte l'ensemble des aspects permettant de caractériser un utilisateur. Ainsi si on reprend la description proposée par (Van Setten, 2001), il manque aujourd'hui dans PERCOMOM : une prise en compte des centres d'intérêts, une prise en compte de l'humeur de l'utilisateur (joyeux, triste, en colère, etc.) ou encore une prise en compte de ses buts personnels qui peuvent être la composition de plusieurs buts métier différents.
- Elle peut vite devenir complexe à gérer à travers la multiplication du nombre de possibilités différentes lorsqu'on prendra en compte l'ensemble des informations sur le contexte.
- Elle ne dispose pas aujourd'hui d'un outillage de validation adapté permettant de limiter les risques d'avoir des définitions différentes pour un même objet comme, par exemple, la définition de deux niveaux d'expérience différents pour un même utilisateur et pour une même application.
- Enfin, son niveau de couverture de l'ensemble des besoins fonctionnels n'a pas été établi étant donné que PERCOMOM n'a été utilisée que pour réaliser un nombre limité d'applications dans le cadre bien particulier de la diffusion d'informations transport.

Ces restrictions étant posées, il reste un dernier point à traiter dans le cadre de la personnalisation : les contenus. Et plus précisément comment dans PERCOMOM ils sont modélisés et surtout comment ils sont pris en compte dans le cadre de la personnalisation.

4.4 La prise en compte des contenus

4.4.1 Introduction

Comme nous l'avons vu dans le chapitre 2, §2.4, les données peuvent être de trois types différents : non structurées, semi structurées et structurées. Nous avons aussi montré dans cette même partie que les données de tout type pouvaient être ramenées à des données structurées. Aussi, dans la suite du document, nous ne traiterons que les problématiques liées aux données structurées et ceci dans le cadre de la personnalisation.

Personnaliser des données c'est être capable de les adapter aux besoins et aux attentes de l'utilisateur. Si cela peut paraître simple lorsqu'on l'énonce ainsi, la personnalisation des contenus regroupe en fait de nombreuses problématiques qui sont :

- En premier lieu, on ne peut personnaliser que ce qu'on connaît. Ceci signifie que si on ne connaît rien de l'information manipulée, on ne pourra pas la personnaliser car on ne saura pas où récupérer les éléments d'information pertinents pour la personnalisation.
- Dans un contexte multi-applications, chaque application peut faire appel à des sources d'informations différentes contenant des données de nature identique du point de vue sémantique. La question se pose alors de savoir comment y accéder dans le cadre de la personnalisation et surtout comment créer des modèles de personnalisation indépendants des sources de données.
- La prise en compte d'informations générales comme, par exemple, le contexte géographique ou le contexte temporel, impose d'être capable de gérer ce type d'information au niveau des sources de données. Mais, la question qui se pose alors est de savoir si cette gestion doit se faire de manière implicite ou alors explicite, et sous quelle forme.

Ce sont toutes ces questions que nous allons traiter maintenant.

4.4.2 Les contenus et l'ontologie de domaine

Dans le cadre de PERCOMOM, l'ontologie de domaine (OD) (cf. chapitre 3, §3.1.3.2) est utilisée pour permettre de modéliser l'ensemble des notions métier faisant partie d'un ou de plusieurs domaines métier. Aussi, les informations qui sont manipulées dans l'ensemble des modèles conceptuels sont des informations associées à l'ontologie de domaine et pas des informations associées à une source de données bien identifiée. En fait, un seul modèle déroge à cette règle, c'est le modèle de données qui permet de faire le lien entre l'ontologie de domaine et une source physique contenant des données.

Dans le cadre de nos travaux, nous avons considéré que les sources physiques de données étaient créées à partir de l'ontologie de domaine dans le cadre de la mise en place de nouvelles applications. Ceci étant établi, le lien entre les sources physiques de données et l'ontologie est réalisé au niveau conceptuel à travers le modèle de données (BM2). Dans ce modèle, les données physiques sont modélisées à travers des modèles de type entité-relation sur lesquels il est possible d'associer des liens vers l'ontologie de domaine (vers les classes, les propriétés et les relations de l'ontologie). De manière simplifiée, les liens entre un modèle entité-relation et l'ontologie de domaine sont établis comme suit :

- Chaque classe de l'ontologie est reliée à une entité du modèle entité-relation.
- Pour une classe donnée, chaque propriété de l'ontologie est reliée à une propriété d'une entité avec quelques exceptions que nous verrons par la suite.
- Chaque relation de l'ontologie est reliée à une relation du modèle entité-relation.

Si ce modèle simplifié permet de facilement faire des liens entre un modèle entité relation et une ontologie, il impose néanmoins un certain nombre de limitations qu'il convient de lever. C'est ce que nous allons montrer maintenant à travers l'utilisation d'un exemple simple. Ainsi, supposons que nous avons défini une ontologie de domaine nous permettant de manipuler trois concepts :

- Le concept de société de transport qui permet de définir et de manipuler l'ensemble des propriétés associées à une société travaillant dans le domaine des transports collectifs (exemple : la société Transpole qui s'occupe des transports en bus, tram et métro sur l'agglomération lilloise).
- Le concept de moyen de transport qui permet de définir et de manipuler l'ensemble des propriétés associées à un moyen de transport collectif (exemple : bus, train, métro, etc.).
- Le concept de ligne qui permet de définir et de manipuler toutes les informations sur une ligne de transport (exemple : la ligne 1 du métro de Lille).

L'ensemble de ces concepts sont reliés par des relations sémantiques qui permettent d'établir des liens entre les différents individus de chaque concept métier (cf. Figure 4.17).

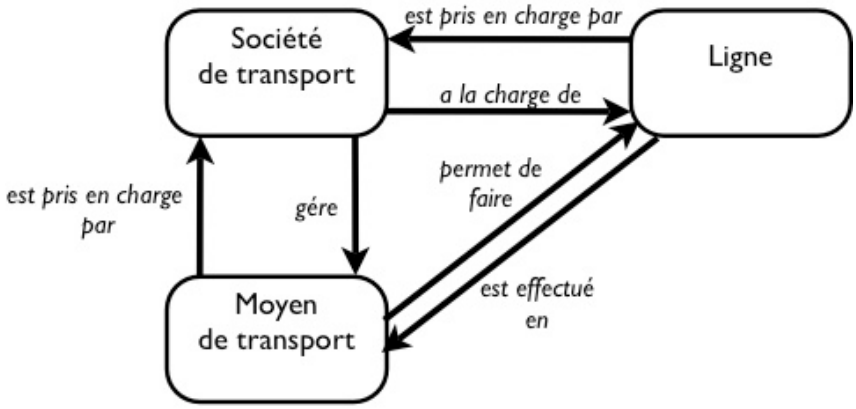


Figure 4.17. Exemple de représentation des liens entre concepts métier dans une ontologie de domaine pour trois concepts métier spécifiques dans le domaine des transports (formalisme disponible dans l'outil Protege qui permet de créer et de manipuler des ontologies)

À partir du modèle présenté à la Figure 4.17, il est possible de définir, pour l'instant de manière manuelle dans l'état actuel de nos travaux sur PERCOMOM, un modèle entité-relation (modèle BM2) en se basant sur les règles suivantes :

- Chaque classe de l'ontologie va donner lieu à la création d'une entité dans le modèle entité-relation. Ainsi, une classe "Société de transport" va donner lieu à la création d'une entité "Société de transport".
- Chaque propriété de l'ontologie va donner lieu à la création d'une propriété au niveau de l'entité dans le modèle entité-relation correspondant. Si la propriété peut avoir plusieurs valeurs, on crée alors deux entités complémentaires : une pour gérer l'ensemble des valeurs possibles et une pour faire le lien entre les valeurs et l'entité associée à la classe de l'ontologie. Ainsi, si la classe "Société de transport" possède une propriété multiple de type "Adresse du dépôt" alors, on va automatiquement créer une entité "Société de transport Adresse du dépôt" pour permettre la sauvegarde de l'ensemble des adresses de tous les dépôts de la société de transport. Et, on va créer une entité "Lien Société de transport Adresse du dépôt" pour permettre de faire le lien entre les société de transport et les adresses des dépôts.
- Chaque lien entre les classes de l'ontologie va être transformé en une relation entre les entités représentant ces classes. Ainsi, dans notre exemple, le lien "gère" entre le concept métier "Société de transport" et le concept métier "Moyen de transport" va être transformé en une relation "gère" entre l'entité "Société de transport" et l'entité "Moyen de transport" dans le modèle entité-relation.
- Chaque entité possédera une clé interne unique, de type identifiant, permettant de facilement établir des relations entre les différentes entités.
- Les cardinalités entre les différentes entités, du modèle entité-relation, seront déterminées en fonction des propriétés de chaque lien entre concepts au niveau de l'ontologie. Ainsi, si le lien est du type "fonctionnel", la cardinalité sera du type "1 vers n". Si le lien est de type "fonctionnel inverse", la cardinalité sera du type "n vers 1". Un exemple de définition de ces propriétés sur les liens, pour le lien "est effectué en" entre la classe "Ligne" et la classe "Moyen de transport" de notre ontologie d'exemple, est donné dans la Figure 4.18.

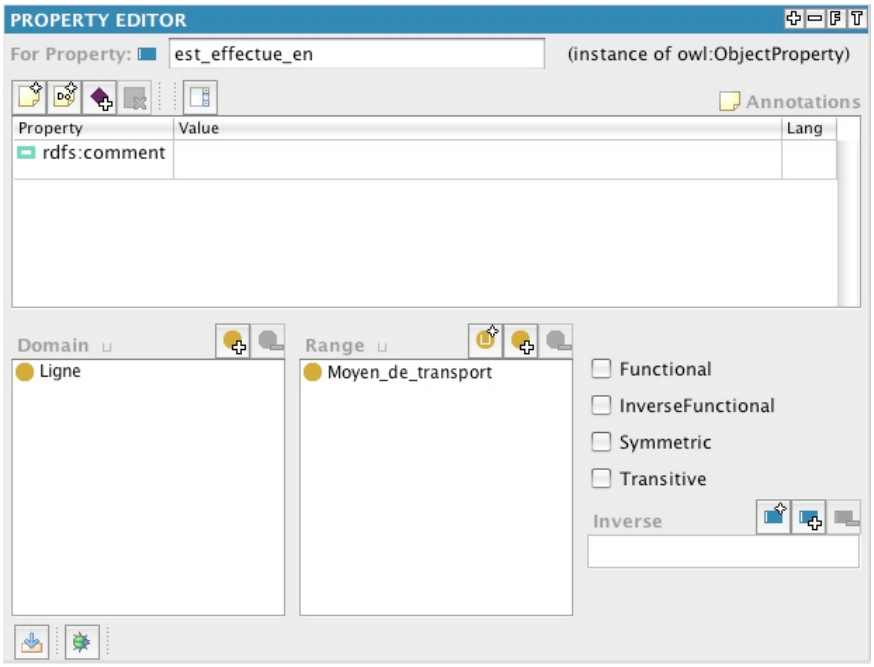


Figure 4.18. Exemple de définition des propriétés pour un lien "est_effectue_en" dans une ontologie métier à l'aide de l'outil Protege

Enfin, la cardinalité pourra être définie au niveau de chaque classe à travers l'utilisation de restrictions dans la définition de la classe de l'ontologie. Ainsi, dans une restriction, on peut indiquer si un lien est obligatoire ou optionnel ; on peut préciser aussi combien de liens au minimum et au maximum on peut avoir entre la classe manipulée et les autres classes de l'ontologie. La Figure 4.19 présente un exemple d'utilisation de restrictions au niveau du concept métier "Société de transport" pour lequel on indique qu'il doit y avoir au minimum un lien "a la charge de" vers le concept métier "Ligne" et entre 1 et 4 liens "gère" au maximum vers le concept métier "Moyen de transport".

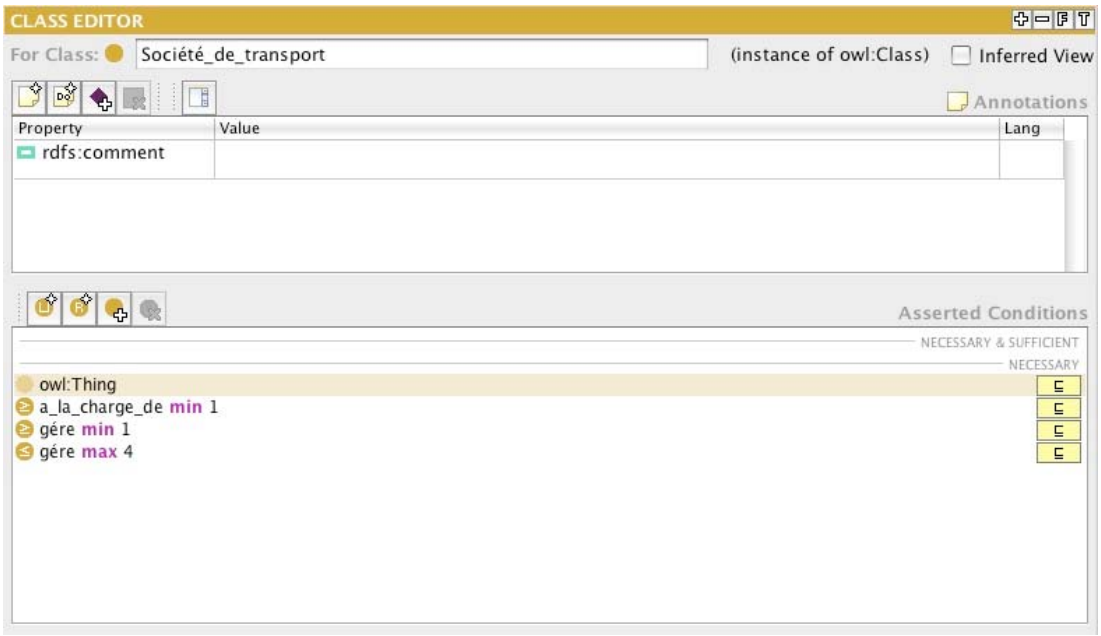


Figure 4.19. Exemple de définition de restriction au niveau d'un concept métier (d'une classe de l'ontologie) dans l'outil Protege

En utilisant l'ensemble de ces règles, on en déduit que le modèle entité-relation correspondant à l'ontologie de la Figure 4.17 serait alors celui présenté dans la Figure 4.20. Sur ce modèle, les cardinalités n'ont volontairement pas été indiquées afin d'en simplifier la lecture.

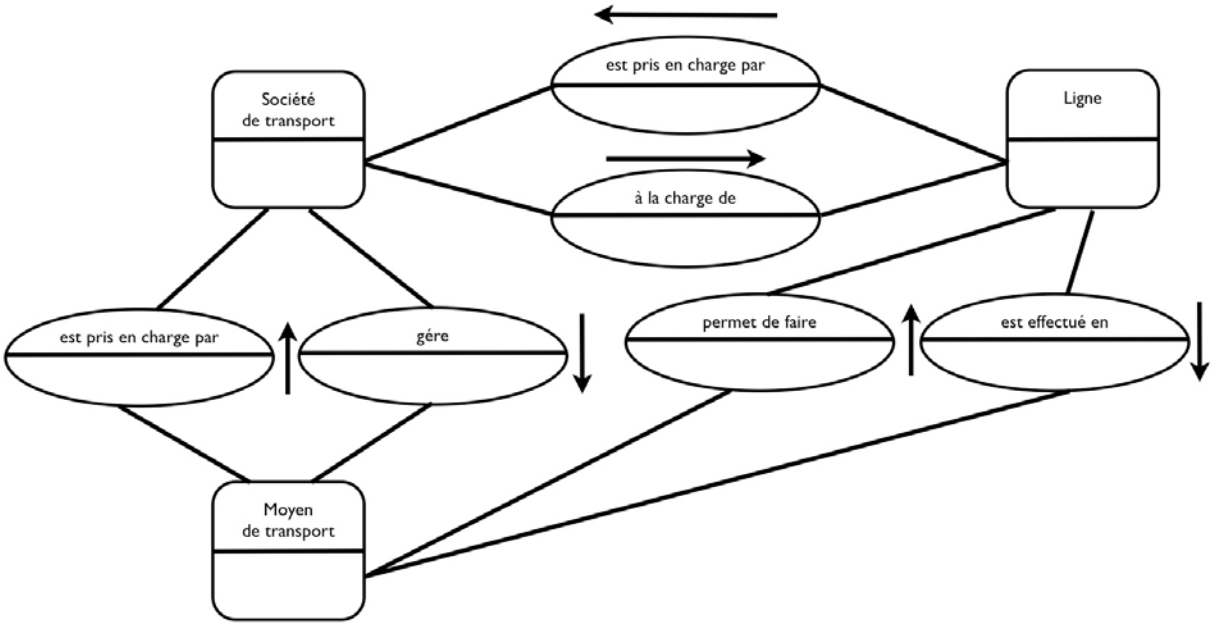


Figure 4.20. Exemple de modèle entité-relation (modèle BM2) obtenu à partir d'une ontologie de domaine

L'utilisation de la relation "est pris en charge par" à deux endroits différents dans le modèle entité-relation, s'il est correct du point de vue de l'ontologie métier, impose certaines contraintes au niveau de la relation physique. Ainsi, si on se met au niveau de la base de données physiques, une relation sera caractérisée par une table physique contenant les informations suivantes :

- L'identifiant de la classe de l'ontologie de départ.
- L'identifiant de l'individu de la classe de l'ontologie de départ.
- L'identifiant de la classe de l'ontologie d'arrivée.
- L'identifiant de l'individu de la classe de l'ontologie d'arrivée.

Pour ce qui est du lien avec les propriétés des classes de l'ontologie, il existe enfin un dernier cas à traiter : si la propriété de la classe de l'ontologie prend ses valeurs dans une liste pré-définie de valeurs, cette liste est alors modélisée dans le modèle entité-relation par la création d'une entité et d'une relation spécifique. Si la propriété de la classe peut prendre plusieurs valeurs dans cette liste, on devra alors créer une entité et une relation supplémentaire permettant de sauvegarder l'ensemble de ces valeurs.

Le lien entre les modèles de données (BM2) et l'ontologie de domaine (OD) étant établi, la question qui se pose maintenant est de savoir comment gérer la problématique d'une modélisation multi-applications nécessitant l'utilisation de bases de données différentes en fonction de la localisation de l'utilisateur au moment de son accès à l'application.

4.4.3 La notion de silo de données au niveau des applications

Pour donner un exemple, supposons qu'une entreprise de transport localisée sur deux zones urbaines décide de créer deux applications : une pour gérer l'ensemble des lignes dont elle a la charge dans la zone urbaine de Lille par exemple et une pour gérer l'ensemble des lignes dont elle a la charge dans la zone urbaine de Valenciennes. Comme son activité sur Valenciennes provient de l'acquisition récente d'une nouvelle société de transport, l'ensemble des informations concernant la gestion des lignes sur Valenciennes se trouve dans une base de données différente des informations concernant la

gestion des lignes sur Lille. Les concepts métier étant les mêmes dans les deux sources de données, l'entreprise décide d'utiliser une application commune pour accéder aux informations avec la particularité que la source de données utilisée sera différente en fonction de l'endroit où est utilisée l'application. La question qui se pose alors est de savoir comment accéder correctement à ces sources de données physiques. Dans PERCOMOM, la solution à cette problématique passe par l'utilisation de la notion de silos de données.

De manière pratique, chaque ensemble de données physiques est associé à un ou plusieurs silos de données ; chaque silo de données contenant l'ensemble des données nécessaires pour le bon fonctionnement d'une ou plusieurs applications. Afin de permettre de différencier les différents silos de données, chaque silo sera identifié par un nom unique.

Dans l'architecture PERCOMOM, les silos de données seront manipulés à deux niveaux :

- Au niveau CIM, où on effectuera l'association entre les silos de données et les applications dans les modèles d'application (MA). Ainsi, si on suppose que l'application "InfoHoraireGare" présentée dans le chapitre 3, §3.1.3.1, accède à un silo de données "Silo de données S1" quand l'utilisateur se trouve dans la ville de Valenciennes, un silo de données "Silo de données S2" quand il se trouve dans la ville de Lille et un silo de données "Silo de données S3" quand il ne se trouve ni à Lille ni à Valenciennes, alors le modèle (AM) correspondant sera le suivant :

```
<application>
 <identite identifiant="14" nom="InfoHoraireGare">
    <categorie nom="Information Horaire"/>
 </identite>
 <proprietesgenerales>
    <languepardefaut code="FRE"/>
 </proprietesgenerales>
 <proprietesapplicatives>
    <businessprocessdemarrage nom="InfoHoraireGareDemarrage"/>
    <businessprocesspardefaut nom="SelectionAffichageHoraireGare"/>
    <controleacces erreur="ErreurAccesInfoHoraire">
       <groupeacces nom="ClientsSNCF"/>
    </controleacces>
 </proprietesapplicatives>
 <silodonnees>
    <silo nom="Silo de données S1" domainevalidite="Ville de Valenciennes"/>
    <silo nom="Silo de données S2" domainevalidite="Ville de Lille"/>
    <silo nom="Silo de données S3" domainevalidite="NiLilleNiValenciennes"/>
 </silodonnees>
</application>
```

- Au niveau PSM, où on définira pour chaque silo de données l'ensemble des bases de données physiques le constituant. Ceci se fera à l'aide d'un fichier de configuration de type XML. Ainsi, si par exemple, on veut définir que le silo de données "Silo de données S1" est associé à une base de données de type MySQL, que le chemin pour accéder à ce silo de données est "jdbc:mysql//localhost:8889/applications" et que pour se connecter à cette base, il faut utiliser le nom d'utilisateur "root" et un mot de passe "root", le fichier de configuration sera alors le suivant :

```
<silodedonnees>
   <silo nom "Silo de données S1">
      <base>
         <basedriver valeur="MySQL"/>
         <basechemin valeur="jdbc:mysql//localhost:8889/applications"/>
         <basenomutilisateur valeur="root"/>
         <basemotdepasse valeur="root"/>
      </base>
   </silo>
</silodedonnees>
```

S'il est possible d'associer plusieurs silos de données à une même application, il est important de noter qu'un seul silo de données pourra être actif à la fois.

La Figure 4.21 présente, de manière très schématique, sans formalisme particulier, un exemple d'association de silos de données pour trois applications :

- Une application A ne permettant d'accéder qu'aux lignes de transport sur la zone urbaine de Valenciennes.
- Une application C ne permettant d'accéder qu'aux lignes de transport sur la zone urbaine de Lille.
- Une application B permettant d'accéder à l'un ou l'autre des deux silos de données en fonction de la localisation géographique de l'utilisateur (dans cet exemple, on utilise des règles métier mais on aurait aussi pu utiliser des restrictions).

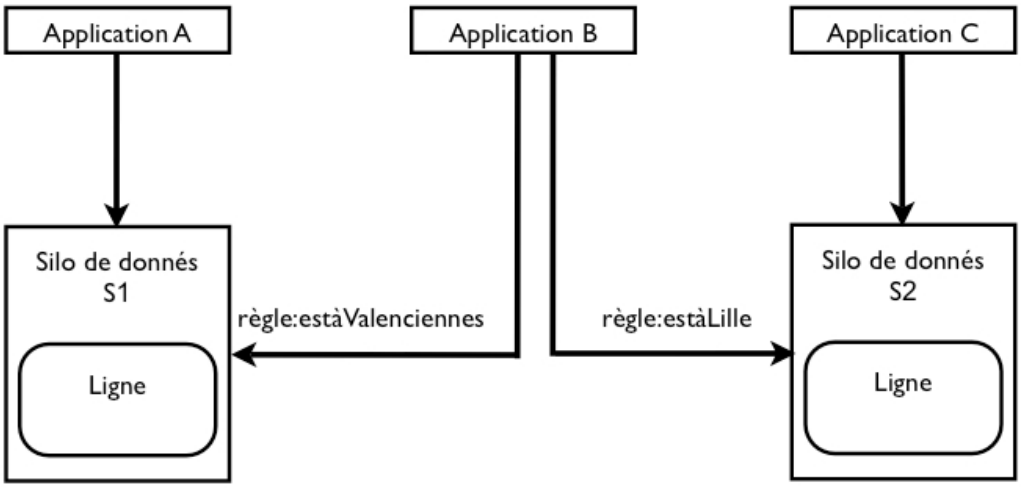


Figure 4.21. Exemple schématique d'association de silos de données à des applications dans PERCOMOM

4.4.4 Synthèse et discussion

Comme nous venons de le montrer, PERCOMOM propose, dans l'état actuel d'avancement de nos travaux, une première approche pour la prise en compte des contenus dans le cadre de la personnalisation des informations. Si celle-ci est encore loin d'être complète, elle présente néanmoins un certain nombre de particularités prometteuses :

- À travers l'utilisation d'une ontologie de domaine (OD), PERCOMOM permet d'associer aux modèles classiques de données, de type entité-relation, une sémantique métier qui permet de séparer la modélisation des sources de données des autres modèles conceptuels.
- L'utilisation de silos de données permet de faciliter la réutilisation des modèles et surtout permet de définir des applications capables de s'adapter à leur environnement au niveau de leur accès aux données physiques. Les silos de données permettent aussi de définir, pour un même concept, des sources de données différentes ce qui devrait, en principe, faciliter la reprise de données existantes.
- La prise en compte du contexte à travers l'utilisation de la notion de domaine de validité permet d'envisager de réaliser une véritable personnalisation contextuelle. Celle-ci est de plus extensible dans le sens où, à travers l'utilisation des restrictions, il serait facile de rajouter de nouveaux types de critères contextuels comme, par exemple, le niveau sonore ou la luminosité extérieure.

À travers toutes ces possibilités, PERCOMOM représente, à notre connaissance, une des premières approches capable de prendre en compte l'adaptation des contenus dans les modèles conceptuels d'applications dans le cadre d'une approche dirigée par les modèles.

Conclusion

Dans le cadre d'une approche de type MDA, telle que celle préconisée par l'OMG, la modélisation de l'ensemble d'une l'application est, en théorie, réalisée au niveau conceptuel. Et ceci est valable aussi pour tous les aspects liés à la personnalisation de l'information. Néanmoins, comme nous l'avons vu précédemment au niveau de la prise en compte du contexte, ceci n'est pas forcément l'approche la plus pertinente.

PERCOMOM permet, dans le cadre de la prise en charge de la personnalisation :

- d'offrir un point unique d'entrée pour la personnalisation d'applications multiples ;
- de ne définir à chaque niveau de l'architecture, grâce à l'utilisation des frameworks, que les informations réellement pertinentes pour ce niveau ;
- d'offrir une personnalisation qui peut être facilement rendue contextuelle ;
- de pouvoir apporter, à chaque niveau de l'architecture, de nouvelles fonctionnalités. Ainsi, il est possible de ne définir un filtrage collaboratif qu'au niveau PIM, l'ensemble des éléments nécessaires à ce filtrage étant gérés et pris en charge par le service de personnalisation. De même, il est possible d'avoir des adaptations automatiques des plateformes utilisateurs en fonction du contexte.

PERCOMOM permet ainsi de proposer une synthèse de l'ensemble des travaux déjà réalisés dans le domaine en proposant des réponses à un certain nombre de limitations et de contraintes.

Une des premières validations qui peut être effectuée lorsqu'on dispose d'une nouvelle approche et de nouveaux outils est de les utiliser dans le cadre de la réalisation d'une application concrète afin de pouvoir voir s'ils sont utilisables. C'est ce que nous avons fait dans le cadre du projet ANR Viatic.Mobilité comme nous allons le montrer dans le chapitre suivant.

Chapitre 5
Mise en œuvre de PERCOMOM dans le projet ANR Viatic.Mobilité

Sommaire

INTRODUCTION 151

5.1 LE PROJET ANR VIATIC.MOBILITE 151
5.1.1 INTRODUCTION 151
5.1.2 LES DIFFERENTS ESPACES D'INTERACTION ET LA NOTION DE CONTINUITE DE L'INFORMATION DANS LE CADRE D'UN DEPLACEMENT MULTIMODAL 152
5.1.3 LES PRINCIPAUX BESOINS D'ADAPTATIONS SPECIFIQUES IDENTIFIES EN MATIERE DE MODELISATION . 154
5.1.4 CONCLUSION 156

5.2 LA PRISE EN COMPTE DU VECU DES DEPLACEMENTS DANS LES APPLICATIONS TRANSPORT 156
5.2.1 INTRODUCTION 156
5.2.2 LA DEFINITION DE LA NOTION DE VECU DES DEPLACEMENTS 157
5.2.3 LA PRISE EN COMPTE DU VECU DES DEPLACEMENTS DANS PERCOMOM 157

5.3 PREMIERE APPLICATION : UNE BORNE INTERACTIVE D'ORIENTATION 163
5.3.1 INTRODUCTION 163
5.3.2 DESCRIPTION DE L'APPLICATION 163
5.3.3 LA MODELISATION DE NIVEAU CIM DE L'APPLICATION TABLE D'ORIENTATION (CF. PARTIE 3.1) 165
5.3.4 CONCLUSION SUR LA PREMIERE APPLICATION 175

5.4 UN CAS PRATIQUE DE TRANSFORMATION DE MODELES DANS PERCOMOM : DES MODELES CONCEPTUELS DE NIVEAU CIM DE L'APPLICATION A L'APPLICATION CONCRETE 175
5.4.1 INTRODUCTION 175
5.4.2 PRESENTATION DE L'APPLICATION 175
5.4.3 LES MODELES CONCEPTUELS ASSOCIES A L'APPLICATION 176
5.4.4 LES CHOIX TECHNIQUES EFFECTUES : L'EXEMPLE DES ELEMENTS DE FRAMEWORK ASSOCIES AU MODELE STATIQUE DES INTERACTIONS (IM2) 178
5.4.5 EXEMPLE DE TRANSFORMATION D'UN MODELE STATIQUE D'INTERACTION (IM2) 183
5.4.6 EXEMPLE DE TRANSFORMATION D'UN MODELE DE PROCESSUS METIER (IM1) 185
5.4.7 CONCLUSION SUR L'EXEMPLE DE TRANSFORMATION DE MODELES 186

5.5 TROISIEME APPLICATION : LA PERSONNALISATION DES CONTENUS DANS LE CADRE D'UNE APPLICATION PERMETTANT DE VISUALISER LES PROCHAINS DEPARTS EN GARE 187
5.5.1 INTRODUCTION 187
5.5.2 DESCRIPTION DE L'APPLICATION 187
5.5.3 LE SERVICE DE PERSONNALISATION ET LA PRISE EN COMPTE DU CONTEXTE DANS LA MODELISATION DE NIVEAU CIM ASSOCIEE A L'APPLICATION 188
5.5.4 CONCLUSION SUR LA TROISIEME APPLICATION 191

5.6 LA PRISE EN COMPTE DE L'UTILISATEUR DANS PERCOMOM : APPARTENANCE SOCIALE ET ECHANGES VERBAUX 192
5.6.1 UN EXEMPLE D'UTILISATION D'UN MODELE D'ORGANISATION (IM2) ET D'UN MODELE D'ACTION (BM1) DANS LE CADRE D'UNE ADAPTATION D'UNE APPLICATION A L'UTILISATEUR 192
5.6.2 PRISE EN COMPTE DES INTERACTIONS VERBALES ENTRE PERSONNES DANS UN PROCESSUS METIER (MODELE IM1) DE NIVEAU CIM 195
5.6.3 CONCLUSION SUR LES EXEMPLES COMPLEMENTAIRES 198

CONCLUSION 198

Introduction

Une des principales problématiques qui se pose lors de la création d'une nouvelle approche de modélisation consiste à pouvoir la valider sur des cas concrets afin d'en définir les limites et les contraintes. Aussi, dans le cadre de l'élaboration de PERCOMOM, nous avons particulièrement porté l'accent sur une validation pratique, à travers la réalisation d'applications concrètes. Étant donnés les travaux menés par notre laboratoire dans le domaine de la diffusion et de la personnalisation des informations dans le domaine des transports, c'est très naturellement que nous nous avons décidé d'utiliser toute cette expérience acquise au fil des années pour servir de support à nos travaux de recherche.

Dans ce chapitre, nous allons présenter le résultat d'applications modélisées avec PERCOMOM. Mais, avant d'étudier ces différents exemples, nous allons commencer par présenter le projet ANR Viatic.Mobilité du pôle de compétitivité I-Trans que nous avons eu la chance d'intégrer pendant toute la durée de notre thèse, auquel nous avons contribué, et qui nous a fourni de nombreux éléments de réflexion.

Puis nous présenterons un premier exemple de modélisation conceptuelle d'une application dans PERCOMOM à travers une application déployée sur une borne interactive du type table d'orientation.

Ensuite nous étudierons, à travers un exemple de transformation de modèles, comment, à l'aide de PERCOMOM, il est possible de générer des applications de manière semi-automatique.

Puis, nous étudierons une troisième application qui permet de consulter la liste des prochains départs en gare et qui présente la particularité d'être capable de s'adapter au lieu où se trouve l'usager.

Enfin, nous terminerons en présentant quelques exemples simples permettant de mettre en avant deux spécificités de notre approche qui sont la définition et l'utilisation des modèles d'actions (BM1) ainsi que la prise en compte des interactions verbales entre intervenants humains dans un processus métier.

Mais commençons par présenter le projet qui a servi de support à nos travaux.

5.1 Le projet ANR Viatic.Mobilité

5.1.1 Introduction

Aujourd'hui, les différentes politiques de transport cherchent à améliorer le transfert modal depuis l'automobile vers les transports collectifs afin de désengorger les villes et de limiter les nuisances liées à la pollution et au bruit. Pour cela, les efforts à effectuer doivent se porter sur deux niveaux : (1) Au niveau des infrastructures techniques pour augmenter les zones couvertes par les transports en commun, diminuer les temps de déplacement, rendre les véhicules de transport en commun plus agréables et plus confortables, embellir les gares et les arrêts, etc. (2) Au niveau de l'information fournie aux voyageurs en vue d'une simplification des usages des transports en commun.

C'est pour apporter des réponses sur ce deuxième niveau qu'a été lancé le projet ANR Viatic.Mobilité (INRETS, 2008). Ayant pris son nom du latin viaticum « provisions pour le voyage », ce projet a pour but de proposer des solutions pour accompagner le voyageur dans sa mobilité au quotidien. Ainsi, à travers différentes solutions techniques, il vise à proposer des services innovants et à simplifier leur paiement en mettant à profit les possibilités offertes par les technologies de l'information et de la communication. Ces services ont pour objectif de proposer de l'information d'accompagnement du voyageur pendant son déplacement (information multimodale) et de l'information d'agrément au cours de sa mobilité (actualités, culture, divertissement, tourisme, jeux, etc.). Ces informations doivent être accessibles à son domicile, à proximité des systèmes de transports et en embarqué dans les différents véhicules qu'il peut emprunter pendant ses déplacements.

De manière plus administrative, le projet Viatic.Mobilité a été lancé dans le cadre du pôle de compétitivité I-Trans (http://www.i-trans.org), dont l'objectif est de de concevoir, fabriquer et vendre les systèmes de transports du futur, et à été soutenu par l'ANR (Agence Nationale pour le Recherche)

(http://www.agence-nationale-recherche.fr). Ce projet, démarré en 2006, a regroupé quatorze partenaires différents du monde de la recherche et du monde de l'entreprise (sociétés What Time is IT ?, Canal TP, Visionor, W@LAN, Infodio, Archimed, IP4U, Atos Worldline, Worldspace, Inrets, Digiport, LAMIH, LAGIS, IEMN). Ce projet a été coordonné par Guillaume Uster de l'INRETS. Notre laboratoire de recherche est intervenu dans ce projet comme partenaire dans le cadre de l'étude et de la conception d'outils innovants pour la création et la génération d'IHM personnalisées. Notre contribution au projet a donné naissance à PERCOMOM.

Afin de bien comprendre certaines orientations et hypothèses qui ont conduit à différents choix lors de la création de PERCOMOM, il convient de présenter les différentes spécificités de ce projet.

5.1.2 Les différents espaces d'interaction et la notion de continuité de l'information dans le cadre d'un déplacement multimodal

Les premiers travaux, menés dans le cadre du projet ANR Viatic.Mobilité par la société What Time is IT ?, ont consisté à faire une étude anthropologique et sociologique permettant d'identifier les spécificités et les besoins des usagers des transports collectifs pendant leurs déplacements. Cette étude, menée sur le terrain à travers des observations et des entretiens avec des voyageurs, a d'abord permis d'identifier trois grands espaces d'information (cf. Figure 5.1) :

- Les points fixes où l'usager peut consulter facilement l'information pour préparer son ou ses déplacements.
- Les pôles d'échange où l'usager est en transit entre deux déplacements, entre deux moyens de transport, et où il est en recherche d'information pour faciliter son déplacement et/ou pour occuper son temps en attendant sa correspondance.
- Les véhicules de transport empruntés par l'usager dans lesquels il peut consulter tout type d'information mais surtout dans lesquels il peut accéder à de l'information lui permettant de faciliter la suite de son déplacement.

Si ces différents espaces permettent d'accéder à des informations en théorie similaires, ils présentent néanmoins chacun des particularités liées à la façon d'accéder aux informations (ordinateur fixe, PDA, afficheurs, borne interactive, etc.). Mais, ils présentent aussi des particularités vis-à-vis des attentes et besoins des usagers qui ne sont pas les mêmes en fonction de l'espace dans lequel ils se trouvent.

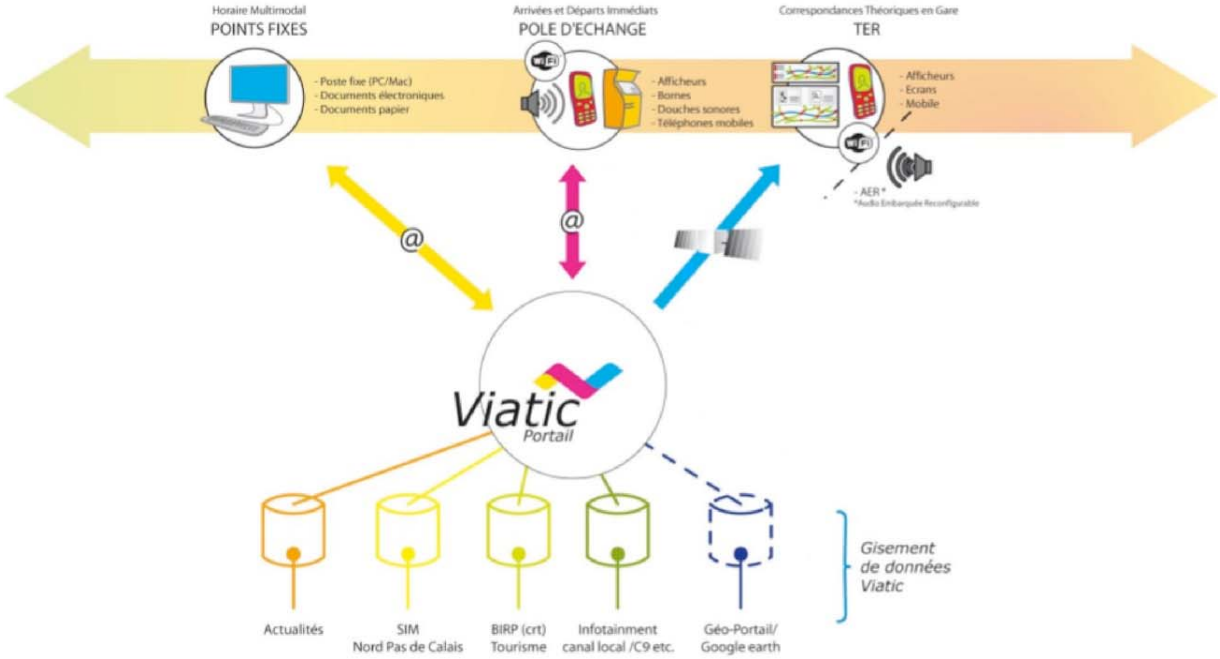


Figure 5.1. Les trois grands espaces d'information identifiés dans le cadre du projet Viatic.Mobilité (INRETS, 2008)

Au niveau de ces besoins, cette étude a permis d'identifier :

- Des attentes différentes en fonction du temps. Ainsi, pour donner un exemple volontairement simplifié, le matin un usager part au travail, il s'informe sur les dernières nouvelles. Le soir, il revient du travail et pense à ses occupations extra-professionnelles pour la fin de journée.
- Un changement d'attitude en fonction du lieu où l'usager se trouve et en fonction de son environnement. À titre d'exemple, on pourra prendre la définition de la notion de vécu de l'usager dans les véhicules de transport en commun que nous présenterons dans le chapitre 5, §5.2.
- Des besoins et un comportement qui évoluent fortement dans des conditions de stress comme, par exemple, lorsqu'une grève de transport survient.
- Des besoins d'accompagnement qui sont différents en fonction du lieu, du moment et du type d'usager (personne âgée, étudiant, etc.).

L'objectif de Viatic.Mobilité étant de favoriser l'utilisation des transports, la question qui s'est posée lors du lancement du projet était de savoir de quels types d'informations avait besoin l'usager. Si l'information transport permettant de faciliter les déplacements de l'usager apparaît comme la solution évidente, celle-ci englobe déjà une des problématiques principales de la diffusion de l'information dans le domaine des transports qui est l'accès à cette information. Ainsi, en fonction de l'endroit où l'usager se trouve au moment où il désire accéder à l'information, il aura éventuellement à sa disposition un où plusieurs supports différents pour le faire (ordinateur fixe, plan, téléphone portable, borne d'information, tableau d'affichage, table d'orientation, affiche, etc.) (cf. Figure 5.2) ; chaque support présentant l'information d'une manière différente (adaptation des informations au support et, si cela est possible, aux besoins de l'usager). Ainsi, un téléphone portable affichera les informations de déplacement en fonction du voyage en cours de l'usager alors qu'une affiche papier présentera, par exemple, l'ensemble des horaires pour une ou plusieurs lignes de transport données.

Figure 5.2. Les services d'aide à la mobilité dans Viatic.Mobilité (INRETS, 2008)

En fait, un des freins de l'utilisation des transports en commun réside dans l'appréhension de l'utilisation du temps pendant les déplacements. Pour la majorité des usagers, ce temps doit être mis à profit pour effectuer des activités qui correspondent à leurs besoins et à leurs attentes. Une des solutions proposées dans le cadre du projet consiste à mettre à disposition des usagers un ensemble services d'agrément lui permettant d'occuper ces temps de déplacement. Ainsi, lorsqu'il se rend de son domicile à son lieu de travail, on pourrait imaginer lui donner accès à des services lui permettant de consulter des informations de presse, des informations financières, de l'information audio, des vidéos, etc. (cf. Figure 5.3). Pour avoir une synthèse plus complète sur les problématiques liées à la mobilité et sur les problématiques liées à l'information dans le domaine des transports, on pourra se reporter à (Uster, 2008).

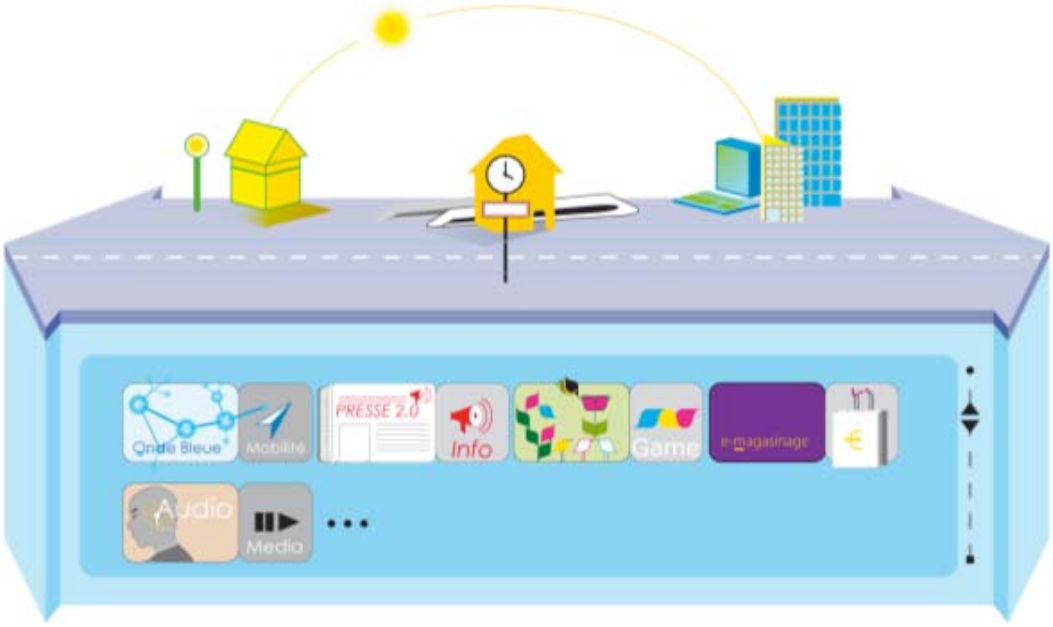


Figure 5.3. Les services d'agrément dans Viatic.Mobilité (INRETS, 2008)

Pour résumer, dans le projet ANR Viatic.Mobilité, la notion d'information recouvre des éléments différents qui vont de la manipulation d'informations simples à l'utilisation de services qui peuvent être liés à la mobilité ou aux loisirs. C'est ce point de vue que nous avons adopté dans le cadre de PERCOMOM et plus particulièrement au niveau de la personnalisation de l'information qui englobe cette dualité.

En matière de modélisation des interactions, la prise en compte de ces différents éléments et surtout de l'ensemble des diversités de situations possibles dans le cadre des déplacements entraîne un certain nombre de besoins spécifiques que nous allons maintenant présenter.

5.1.3 Les principaux besoins d'adaptations spécifiques identifiés en matière de modélisation

Dans le cadre de la création d'applications interactives dans le domaine des transports, le projet ANR Viatic.Mobilité a permis d'identifier un certain nombre de besoins spécifiques d'adaptation des systèmes d'information transport. Ceux-ci sont résumés dans le Tableau 5.1.

Tableau 5.1. Les principaux besoins d'adaptation spécifiques identifiés dans le cadre du projet ANR Viatic.Mobilité

Besoin d'adaptation	**Commentaire**	**Prise en compte dans PERCOMOM**
Adaptation aux différentes populations	Dans les transports en commun, les usagers proviennent de toutes les catégories sociales de la population et sont de tous âges. Les applications doivent donc être capables de prendre en compte cette diversité aussi bien au niveau des interactions qu'au niveau des contenus. Ainsi, il doit être, par exemple, possible d'adapter la publicité en fonction de la catégorie socioprofessionnelle de l'usager.	La prise en compte de ces différentes adaptations passe par l'utilisation du modèle de profil (SM1) pour renseigner les différentes informations sur l'utilisateur puis par l'utilisation de ces informations dans les autres modèles conceptuels comme les modèles de processus métier (IM1) ou encore les modèles d'action (BM1).
Adaptation à l'environnement	Dans les transports, la prise en compte du contexte permet d'apporter des informations pertinentes à l'usager en fonction de son environnement direct.	La prise en compte du contexte peut se faire de différentes manières comme présenté dans le chapitre 4.
Adaptation à de multiples plateformes	Afin de faciliter et d'encourager les déplacements, les usagers doivent pouvoir accéder à l'information et aux applications avec tout type de plateformes de consultation (téléphone, PDA, ordinateur portable, etc.). Ceci entraîne un nombre très important de plateformes possibles pour les applications qui ne peut être gérée qu'à travers une automatisation poussée de la génération des IHM.	L'adaptation à la plateforme se fait, au niveau PSM, à travers l'utilisation de deux niveaux de framework (cf. chapitre 3, §3.2.3).
Adaptation à la langue de l'usager	Les voyageurs pouvant provenir de différents pays et de différentes origines, les applications doivent être capables de s'adapter à la langue et à la culture de l'usager. Ainsi, on pourrait imaginer une application qui adapte automatiquement sa charte graphique en fonction de l'origine géographique et culturelle de l'utilisateur.	La langue de l'usager faisant partie des informations contenues dans le profil de l'utilisateur (SM1), elle peut être utilisée comme critère dans tous les autres modèles de niveau CIM.
Adaptation aux handicaps	Les applications doivent être capables de s'adapter aux handicaps des usagers. Exemple 1 : au niveau des IHM, pour une personne avec des problèmes de vue, les caractères affichés à l'écran doivent être de plus grande dimension. Exemple 2 : Au niveau des processus, pour une personne à mobilité réduite, le calcul des itinéraires doit tenir compte, au niveau des temps affichés, des difficultés de déplacement de ces personnes et, au niveau du chemin proposé, des aménagements spéciaux effectués.	Dans le cadre de PERCOMOM, l'adaptation aux handicaps n'a pas encore fait l'objet d'une étude approfondie. Aussi, est-elle pour l'instant gérée via des informations contenues dans le profil de l'utilisateur permettant de définir pour chaque type de handicap si l'usager le possède ou pas.
Adaptation au vécu des voyageurs	La prise en compte du vécu des voyageurs (cf. chapitre 5, §5.2) est quelque chose de spécifique au monde des transports qui permet d'apporter une personnalisation encore plus poussée au niveau des applications et surtout qui permet de prendre en compte les spécificités liées aux transports.	La prise en compte du vécu des voyageurs représente un élément spécifique au domaine des transports qui a été identifié dans le cadre de la réalisation du projet ANR Viatic.Mobilité. Sa définition et sa prise en compte dans le cadre de PERCOMOM sera présentée dans le chapitre 5, §5.2.

Besoin d'adaptation	**Commentaire**	**Prise en compte dans PERCOMOM**
Adaptation des infrastructures techniques	Dans le domaine des transports, l'utilisateur est par nature en mouvement empruntant des véhicules et traversant des lieux parfois fort différents. Au niveau technique, ceci signifie qu'il est susceptible d'utiliser des infrastructures différentes pour accéder à l'information en fonction du lieu où il se trouve. Ainsi, pour donner un exemple d'infrastructure choisie dans le cadre du projet, dans un train en mouvement un usager ne pourra accéder à une application de consultation des horaires qu'à partir d'un serveur embarqué, de capacité de stockage limitée, capable de recevoir des informations à partir d'un satellite mais incapable d'en émettre. Alors que s'il se trouve dans une gare, il pourra, par exemple, accéder à un serveur central contenant tous les horaires de manière bi-directionnelle. Dans les deux cas, il utilisera la même application, mais l'architecture technique utilisée pour accéder à l'application sera dépendante du contexte d'utilisation.	Dans PERCOMOM, ce type d'adaptation se fait au niveau CIM en définissant l'utilisation de services fonctionnels différents en fonction du lieu où se trouve l'utilisateur (utilisation des règles de niveau CIM pour indiquer quel service fonctionnel utiliser en fonction de quel lieu).

5.1.4 Conclusion

Pour résumer cette présentation rapide du projet ANR Viatic.Mobilité, ce projet a offert un cadre permettant d'identifier de nombreuses problématiques devant être prises en compte au niveau de la modélisation conceptuelle des applications. Ainsi, il a fait ressortir le besoin de prise en compte du contexte géographique et temporel ainsi que de la continuité d'accès à l'information. Il a aussi permis d'identifier des besoins importants en matière d'adaptation des applications.

Dans le cadre de nos travaux de recherche, le projet ANR Viatic.Mobilité nous a permis d'identifier, à travers l'expérience des différents partenaires ainsi qu'à travers différentes études menées sur le terrain, un grand nombre de besoins fonctionnels. Ainsi, il nous a été possible d'appliquer PERCOMOM à des cas concrets et ainsi de la valider partie par partie aussi bien du point de vue scientifique que du point de vue pratique, dans la mesure du possible. Ce sont quelques uns de ces cas concrets que nous allons présenter dans ce chapitre. Mais, avant toute chose, nous allons revenir sur un point particulier identifié dans le cadre du projet qui est le besoin de prise en compte, pour les usagers, du vécu des déplacements.

5.2 La prise en compte du vécu des déplacements dans les applications transport

5.2.1 Introduction

Le projet ANR Viatic.Mobilité, nous a permis d'identifier, dans le cadre de la personnalisation des contenus dans le domaine du transport, une notion importante et spécifique au domaine du transport qui est celle du vécu des déplacements par les usagers.

Dans cette partie, nous allons montrer comment PERCOMOM est capable de prendre en charge cette notion de vécu des déplacements dans le cadre de la modélisation d'une application interactive personnalisée dans le domaine du transport.

Pour cela, nous allons commencer par définir ce que recouvre la notion de vécu des déplacemements. Puis nous montrerons comment, dans PERCOMOM, il est possible d'utiliser cette notion dans le cadre de la modélisation de niveau CIM.

5.2.2 La définition de la notion de vécu des déplacements

Dans le domaine des transports collectifs, depuis longtemps, les temps de déplacement sont considérés comme des pertes de temps, comme des temps morts (Fichelet *et al.*, 1970) pendant lesquels les activités exercées, comme la lecture ou la discussion avec son voisin ne sont faites par le voyageur que pour passer le temps, que pour meubler un vide (Orain, 1997). Or, une étude datant de la fin des années 1990 (Marzloff et Glaziou, 1999) a montré que le temps de transport n'était perçu comme du temps perdu que pour moins de 40% des usagers. En partant de ce constat, des travaux de recherche ont cherché à différencier l'acte de déplacement des activités effectuées pendant les trajets (Mokhtarian et Salomon, 2001) puis à caractériser les activités effectuées pendant les temps de déplacement en essayant de les regrouper en grandes catégories.

Ainsi, (Flam, 2005) propose de répartir les activités suivant trois groupes :

- Les activités productives qui permettent de produire quelque chose comme la lecture, l'écriture, etc.
- Les activités de relâchement et de transition comme l'écoute de musique, l'observation du paysage, etc.
- Les activités de sociabilité qui permettent d'ouvrir une communication avec un ou plusieurs autres individus.

Une autre catégorisation possible, basée sur une étude sociologique et anthropologique réalisée dans le cadre du projet Viatic.Mobilité (Juguet *et al.*, 2007), propose une répartition en quatre groupes distincts répartis sur deux axes : un axe vertical représentant le degré de concentration du voyageur et un axe horizontal représentant son degré d'ouverture aux autres (cf. Figure 5.4) :

- Le castor représente la personne qui a une activité productive (lecture ou écriture d'un rapport, révision d'un cours, apprentissage d'une langue, etc.).
- La chouette représente la personne qui est perdue, qui recherche de l'information (consultation des plans de ligne, recherche de la prochaine correspondance, etc.).
- La marmotte représente la personne qui dort, qui se relâche (personne qui somnole, personne qui regarde le paysage défiler, etc.).
- Le paon représente la personne qui se montre, qui communique avec les autres (discussion avec son voisin, envoi de SMS, etc.).

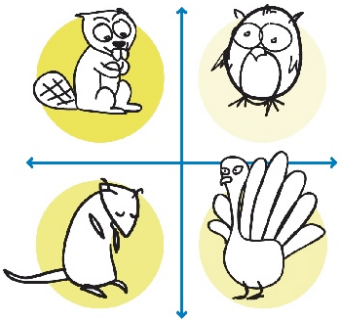


Figure 5.4. Les catégories de vécu des déplacements identifiées dans le cadre du projet Viatic.Mobilité (Juguet *et al.*, 2007)

5.2.3 La prise en compte du vécu des déplacements dans PERCOMOM

Pour prendre en compte le vécu des déplacements dans les modèles conceptuels d'une application, il est indispensable d'abord d'être capable d'identifier correctement les activités associées à chaque catégorie de vécu des déplacements. Puis, il faut définir comment il est possible de connaître, à tout moment, dans quelle catégorie de vécu de déplacement se trouve l'utilisateur. Enfin, il faut être capable de prendre en compte cette notion de catégorie de vécu des déplacements dans les modèles de l'application et notamment dans les premiers d'entre eux : les modèles conceptuels (niveau CIM).

Dans PERCOMOM, le principal modèle permettant de modéliser l'ensemble des activités des utilisateurs est le modèle des processus métier (modèle IM1) ; celui-ci reprenant l'ensemble des tâches effectuées en interaction avec l'utilisateur. Dans PERCOMOM, la réalisation de ces modèles est prévue pour être effectuée par des experts métier sur la base d'une analyse métier des besoins des utilisateurs pour chaque but et sous but fonctionnel identifiés au niveau de l'application. Prendre en compte le vécu des déplacements, dans ce type de modèle, revient donc à être capable d'identifier, pour chaque catégorie de vécu, l'ensemble des buts et sous-buts et donc l'ensemble des tâches qui lui sont associés.

Aussi, pour prendre en compte le vécu des déplacements, nous proposons d'enrichir la phase de conception des modèles de processus métier (IM1) d'une étape complémentaire à l'étape d'identification des processus et sous-processus métier (cf. Figure 5.5). Ainsi, cette phase de conception consistera, en premier lieu, une fois l'ensemble des tâches métier identifiées, à les regrouper en processus métier ; c'est-à-dire en ensembles de tâches permettant de réaliser un but métier du point de vue de l'utilisateur et/ou du point de vue du domaine métier (un processus métier pouvant faire appel à plusieurs utilisateurs). Une fois les processus identifiés, la deuxième étape consistera à les répartir dans les différentes catégories de vécu des déplacements ; chaque processus métier pouvant faire partie d'une ou de plusieurs catégories différentes de vécu des déplacements. Puis, si nécessaire, il faudra adapter chaque processus métier aux différentes catégories de vécu des déplacements ; c'est-à-dire redéfinir, éventuellement, le contenu de chaque tâche métier en fonction des différentes catégories de vécu des déplacements associées au processus métier. À ce niveau, on ne créera pas de nouveau processus métier, mais on utilisera l'information sur le groupe d'appartenance de l'utilisateur à une catégorie de vécu des déplacements pour adapter certaines tâches métier ou pour permettre l'exécution de certaines tâches métier associées uniquement à une catégorie de vécu des déplacements en particulier. Ainsi, si on prend pour exemple un processus métier associé à la création d'un nouveau déplacement, celui-ci pourra être automatiquement lié à l'agenda professionnel de l'utilisateur s'il est du type "Castor" alors que ce lien sera optionnel pour un utilisateur de type "Paon". Chaque adaptation devant correspondre à un réel besoin métier, celle-ci n'est pas réalisée de manière systématique sur chaque processus métier ; certains processus métier pouvant être identiques pour tous les utilisateurs comme, par exemple, l'impression d'un plan de ligne de métro.

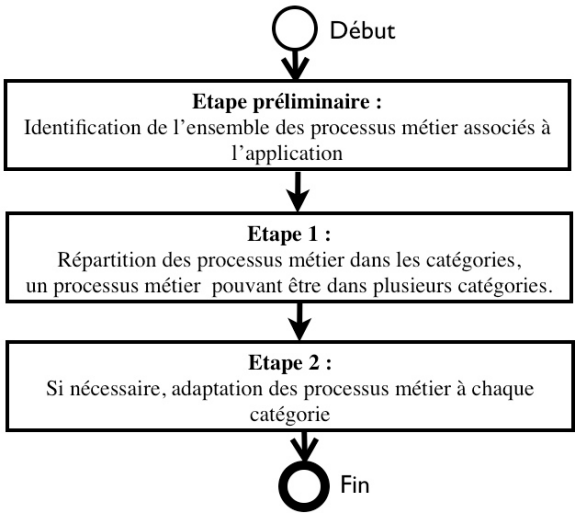


Figure 5.5. La partie spécifique d'analyse associée à la prise en compte des catégories de vécu des déplacements lors de la création des modèles de processus métier (IM1) au niveau CIM

Si la répartition des catégories et leur adaptation se fait au niveau de la phase d'analyse, lors de la création des modèles de processus métier (IM1), il est important de prévoir, lorsque l'application sera en production, une phase de rebouclage permettant de valider les choix effectués lors de la phase l'analyse. Pour cela, il faudra, une fois l'application déployée, mener des enquêtes sur le terrain permettant d'analyser les usages réels de chaque processus métier ; la catégorie choisie au moment de l'analyse initiale n'étant pas forcément la plus pertinente. En cas d'écart, il faudra revenir sur la phase d'analyse et adapter, si nécessaire, la répartition des processus métier dans les différentes catégories et/ou modifier l'adaptation aux catégories faite dans chaque processus métier. De ce fait, la répartition

des processus métier dans chaque catégorie et leur adaptation doit être un travail pluridisciplinaire faisant intervenir de nombreuses disciplines aussi bien dans le domaine des nouvelles technologies que dans le domaine des sciences humaines. On doit donc avoir, à ce niveau, un décloisonnement des disciplines et, c'est probablement là un des défis majeurs de la mise en place et de l'utilisation de PERCOMOM.

La principale problématique d'utilisation de la notion de vécu des déplacements réside dans le fait que pour l'utiliser, il faut être capable, à chaque instant, de connaître la catégorie de vécu de l'usager. Or, celle-ci peut changer de nombreuses fois dans le temps. Ainsi, un voyageur peut commencer son voyage en travaillant sur un dossier professionnel (catégorie "Castor") puis discuter avec son voisin (catégorie "Paon") et enfin somnoler un peu en attendant l'arrivée du train en gare (catégorie "Marmotte"). Malheureusement, aujourd'hui, on ne possède pas, à notre connaissance, de capteurs ou de modèles permettant de connaître de manière sûre le vécu des déplacements à chaque instant.

Aussi, dans le cadre de nos travaux, nous avons défini une technique de base d'identification du vécu des déplacements en fonction de sa "localisation" dans l'application. Cette identification se fait à partir de la connaissance du processus métier en cours d'utilisation par le voyageur et du contexte de vécu des déplacements dans lequel celui-ci est appelé. Ainsi, dans l'application, l'appel à chaque processus métier de l'application se fait toujours en association avec une catégorie de vécu des déplacements bien précise. Pour cela, l'application dispose de points d'entrée permettant d'accéder à un ensemble de processus métier associés à une seule catégorie de vécu du voyageur. Pour l'instant, étant donné que nous nous limitons à des applications de type WIMP, nous proposons d'utiliser les menus comme éléments permettant de faire cette répartition. Ainsi, lorsque l'utilisateur appelle un processus métier, celui-ci est systématiquement rattaché à un vécu des déplacements bien déterminé. Une fois la catégorie de vécu identifiée, celle-ci est stockée au niveau du profil de l'utilisateur permettant ainsi de l'utiliser au sein des processus métier comme toute autre propriété du profil de l'utilisateur. La première version de l'arbre de décision utilisé dans PERCOMOM pour la prise en compte des changements de catégories de vécu des déplacements est donnée dans la Figure 5.6.

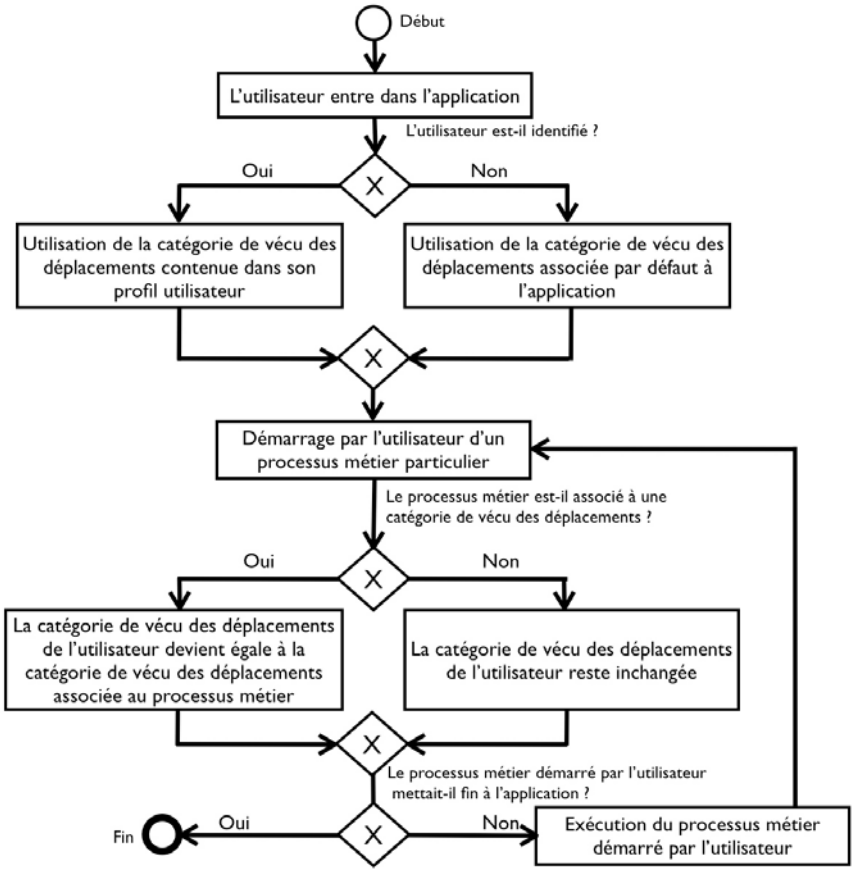


Figure 5.6. Arbre de décision utilisé dans PERCOMOM pour la prise en compte des changements de catégories de vécu des déplacements lors de l'exécution des applications

La valeur active de la catégorie de vécu des déplacements étant sauvegardée au niveau du profil de l'utilisateur, celle-ci peut être utilisée comme toute autre valeur du profil de l'utilisateur dans l'ensemble des modèles conceptuels de niveau CIM. Et, parmi l'ensemble des modèles de niveau CIM, les trois principaux modèles qui utiliseront cette information sont :

- Les règles métier (BM3),
- Le modèle de processus métier (IM1),
- Le modèle statique d'interaction (IM2).

Ainsi, au niveau des règles métier (BM3), l'utilisation de l'information sur la catégorie de vécu des déplacements se fait comme toute autre information du profil. Pour donner un exemple, on trouvera ci-dessous une règle métier "EstUnCastor" de type validation (qui ne peut retourner que les valeurs vrai ou faux) qui permet de savoir si l'utilisateur fait partie de la catégorie "Castor" :

```
<clause id="133" nom="UtilisateurEstUnCastor">
    <element type="profil" nom="CategorieUtilisateur"/>
    <comparateur type="=="/>
    <membredroit type="valeurfixe" typevaleur="chainedecaracteres" valeur="Castor"/>
</clause>

<expression id="19" nom="ExpressionUtilisateurEstUnCastor">
    <contenu type="clause" id="133"/>
</expression>

<regle id="56" nom="EstUnCastor" idexpression="19" type="validation"/>
```

Pour les modèles de processus métier (IM1), l'utilisation de l'information sur la catégorie de l'utilisateur se fait à travers des éléments de sélection des tâches à réaliser dans le cours du processus métier. Si le critère de sélection n'a que peu de chance d'évoluer dans le temps, il peut être utilisé directement au niveau du processus métier. Dans le cas contraire, il faudra utiliser une règle soit de validation, soit de sélection (règle qui retourne une valeur). L'exemple de la Figure 5.7 présente, dans un processus métier simplifié, un élément de sélection qui utilise, comme critère de sélection, la valeur de l'élément du profil de l'utilisateur contenant sa catégorie de vécu de déplacement pour déclencher une tâche qui est spécifique pour chaque catégorie de vécu des déplacements.

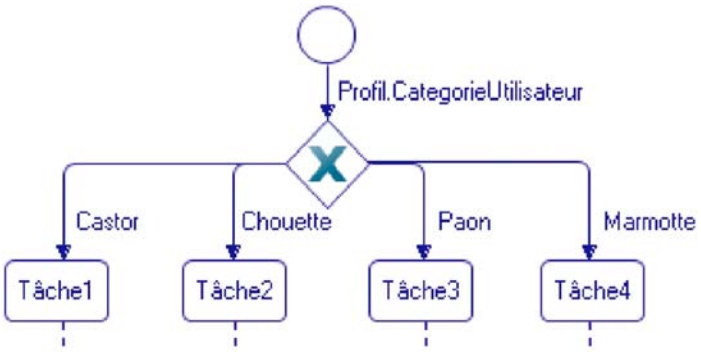


Figure 5.7. Exemple d'utilisation de la catégorie de vécu des déplacements dans un modèle de processus métier (modèle IM1) de niveau CIM

Pour les modèles statiques d'interaction (IM2), l'information sur le profil de l'utilisateur s'utilise uniquement via des règles métier de validation au niveau des propriétés d'affichage des éléments d'interaction :

- La propriété "EstVisibleRègle" qui est utilisée pour savoir si l'élément d'interaction est visible pour l'utilisateur.
- La propriété "EstUtilisableRègle" qui est utilisée pour savoir si l'élément d'interaction est manipulable ou pas par l'utilisateur.

La Figure 5.8 donne un exemple d'un champ de saisie de type texte ("NomUtilisateur") dont le contenu ne sera pas modifiable si l'évaluation de la règle "EstUnCastor", qui permet de savoir si l'utilisateur est ou pas de la catégorie Castor, retourne faux.

Figure 5.8. Exemple d'utilisation de la notion de catégorie de vécu des déplacements dans un modèle statique d'interaction (modèle IM2) de niveau CIM

Pour illustrer le cas de l'adaptation d'un processus métier (modèle IM1) à la catégorie de vécu des déplacements, nous allons présenter l'exemple d'un processus métier ayant pour but de permettre l'affichage des informations associées au prochain déplacement de l'usager. L'analyse des besoins menée par rapport à la prise en compte du vécu des déplacements a donné lieu à l'identification de deux adaptations différentes du processus métier. La première, qui est liée à la catégorie "Castor" permet de lier l'affichage des informations concernant le déplacement de l'usager avec les informations contenues dans son agenda professionnel afin de lui donner des informations sur le but du déplacement et sur le rendez-vous suivant. La deuxième, qui est commune à l'ensemble des autres catégories, n'affiche que les informations associées au déplacement lui-même.

Au niveau de la modélisation conceptuelle de niveau CIM, l'accès à ce processus pourrait se faire, dans une IHM de type WIMP, à l'aide d'une option dans un menu ou à l'aide d'un élément d'interaction de type "bouton". A partir de là, une action pourrait être lancée qui utiliserait un critère de sélection basé sur la catégorie de l'utilisateur (mot-clé "Profil.CategorieUtilisateur") pour afficher soit un groupe d'éléments d'interaction appelé "UIGroupProchDepartPro" si celui-ci est de la catégorie "Castor" soit vers un groupe d'éléments d'interaction "UIGroupProchDepart Gen" pour les autres catégories ; c'est le cas choisi par défaut si la catégorie de l'utilisateur n'est pas la catégorie "Casto. La Figure 5.9 montre un extrait du modèle de processus métier associé centré sur la sélection du groupe d'éléments d'interaction à afficher en fonction de la catégorie de vécu des déplacements. La Figure 5.10 présente deux exemples de sortie d'écran qui pourraient être associés à un usager de type Castor (exemple d'écran a) et à un usager qui n'est pas du type Castor (exemple d'écran b)

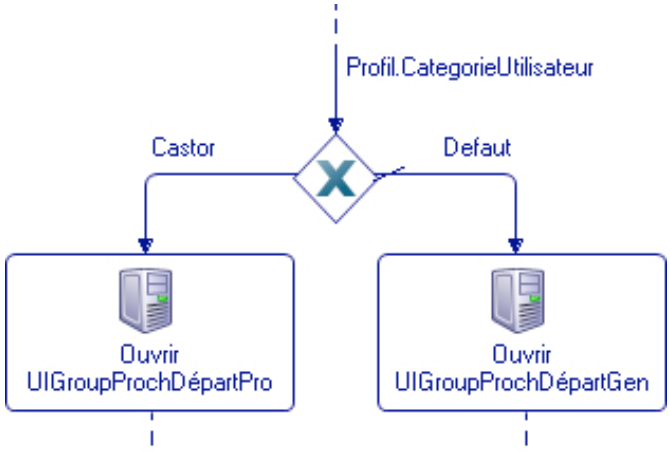


Figure 5.9. Extrait du processus métier (modèle IM1 de niveau CIM) associé à l'affichage des informations sur le prochain déplacement de l'utilisateur

(a)

(b)

Figure 5.10. Exemple d'écrans associés à l'affichage du prochain déplacement (a) pour un usager de la catégorie de vécu des déplacements Castor, pour lequel on affiche des informations professionnelles, et (b) pour un usager qui n'est pas de la catégorie Castor

Sur les deux exemples d'écrans de la Figure 5.10, afin de tenir compte du fait qu'il est impossible d'être certain de la catégorie de vécu des déplacements l'utilisateur, on a ajouté la possibilité d'appeler une action, à travers l'utilisation du bouton "Modifier affichage", permettant d'exécuter un nouveau processus métier qui permet de changer le contenu de l'affichage sans changer la catégorie de

l'utilisateur. Grâce à celle-ci, il devient, par exemple, possible à un utilisateur de type Castor de demander à avoir un affichage des informations suivant le format générique et réciproquement.

Cette problématique spécifique au domaine des transports étant prise en compte, nous allons maintenant voir, à travers un exemple concret, comment il est possible d'utiliser PERCOMOM pour modéliser une application concrète.

5.3 Première application : une borne interactive d'orientation

5.3.1 Introduction

Dans le projet ANR Viatic.Mobilité, l'application table d'orientation développée pour les bornes interactives (cf. Figure 5.11), est une application qui présente la particularité d'être personnalisée en fonction du contexte mais pas en fonction de l'utilisateur. Cette application va nous permettre de valider l'approche PERCOMOM dans le cadre de la modélisation d'une application concrète relativement simple.

Figure 5.11 Borne interactive développée dans le cadre du projet ANR Viatic.Mobilité utilisant l'application table d'orientation à destination des usagers des transports en commun

Pour cela, après une description fonctionnelle rapide de l'application, nous montrerons comment, à partir des besoins fonctionnels, il est possible de définir les différents modèles conceptuels de niveau CIM. Nous montrerons plus particulièrement comment il est possible de définir les proccesus métier (modèle IM1) et les modèles statiques d'interaction (IM2) associés à l'application à partir des besoins fonctionnels. Puis nous verrons, à travers cette application, comment les différents modèles de niveau CIM peuvent être reliés entre eux.

Il est important de noter que l'application déployée sur les bornes interactives dans le cadre du projet ANR Viatic.Mobilité n'est pas une application qui a été modélisée à l'aide de PERCOMOM. En conséquence, l'ensemble des éléments qui seront présentés dans cette partie relèvent d'un travail de rétro ingénierie sur cette application.

5.3.2 Description de l'application

Lors de ses déplacements, un usager des transports en commun va parfois avoir besoin d'accéder à des informations de mobilité alors qu'il n'est pas doté d'un système informatique nomade (PDA, téléphone portable, etc.) ou qu'il en est doté mais qu'il ne désire pas l'utiliser. Pour répondre à ce besoin, il convient de développer des systèmes physiques permettant à l'usager d'accéder facilement à

des informations de mobilité. Ainsi dans le cadre du projet ANR Viatic.Mobilité, une borne interactive de consultation d'informations à destination des usagers des transports en commun a été développée par Visionor et l'Inrets (cf. Figure 5.12 et Figure 5.13). À l'aide de cette borne, les usagers peuvent, à l'aide d'un menu accèder à une application, appelée table d'orientation, qui permet de :

- rechercher un itinéraire, utilisant au maximum les transports en commun, pour se rendre d'un point A à un point B ;
- rechercher les services de proximité (commerces, administrations, etc.) qui se trouvent autour d'un point géographique donné (par défaut l'endroit où se trouve la borne) ;
- rechercher une adresse et la localiser sur la carte (cf. Figure 5.13).

Les usagers peuvent aussi naviguer sur la carte géographique à l'aide d'un menu spécifique qui permet de se déplacer sur la carte, de faire des zooms avant et arrière ou encore de changer le type d'affichage (vue satellite, mode plan, mode mixte).

Cette table d'orientation présentant de nombreuses fonctionnalités, nous ne développerons dans cette partie que les modèles associés à la recherche d'informations.

Il est important de noter que, dans le cadre de la borne interactive, la modélisation de l'application associée à la borne nécessite l'utilisation d'objets d'interaction particuliers permettant la manipulation des cartes géographiques. Ces objets, ce sont les *UIFieldGeoMap* (cf. chapitre 3.1.4.2.) qui présentent la particularité d'avoir un comportement qui est défini de deux manières différentes :

- À travers l'utilisation d'un artefact dédié qui permet de définir les propriétés de l'objet comme la taille du zoom utilisé par défaut, l'endroit géographique sur lequel doit être centrée la carte, ou encore le type d'affichage par défaut (carte, satellite, mixte ou 3D).
- À travers un langage d'interaction spécifique, utilisable dans les actions, qui permet d'utiliser un certain nombre de fonctionnalités spécifiques aux éléments d'interaction de type géographique (affichage d'itinéraires, ajout d'icônes, etc.).

Figure 5.12. Page d'accueil associée à la table d'orientation développée dans le cadre du projet Viatic.Mobilité

Figure 5.13. Ecran affiché, au niveau de la table d'orientation, lorsque l'usager effectue une recherche d'adresse

5.3.3 La modélisation de niveau CIM de l'application table d'orientation (cf. chapitre 3, §3.1)

5.3.3.1 Identification des modèles supports

Les besoins généraux au niveau de l'application étant connus (cf. chapitre 5, §5.3.1), l'étape suivante, dans le cadre de la création des modèles de niveau CIM avec PERCOMOM (cf. chapitre 3, §3.1.2.2) consiste à identifier l'ontologie de domaine qui sera utilisée par l'application.

La problématique de création des ontologies n'étant pas au cœur de notre thèse, nous ne détaillerons volontairement pas la création de l'ontologie dans le cadre de notre exemple. Aussi, nous supposerons que l'application associée à la table d'orientation utilise une ontologie de domaine métier (OD) déjà existante.

L'étape suivante dans PERCOMOM consiste à répartir l'ensemble des problématiques métier associées à l'application au niveau des différents experts métier qui seront utilisés pour modéliser l'application. Si certaines problématiques peuvent être facilement identifiées lors de la phase d'analyse initiale des besoins, d'autres n'apparaîtront qu'après une analyse plus détaillée des besoins et plus précisément au moment où on aura à exprimer ces besoins dans des modèles. Ceci signifie que le processus de création des différents modèles de niveau CIM ne sera pas un processus linéaire mais un processus itératif. Ainsi, en fonction des besoins de modélisation identifiés dans le cadre du travail de modélisation des experts métier, il est possible qu'il soit nécessaire de créer de nouveaux modèles ou de modifier des modèles déjà créés. Afin de simplifier la présentation de notre exemple, nous considérerons que la création de l'ensemble des modèles métier se fera par un seul expert métier. Celui-ci, en faisant une première analyse des besoins décide de commencer par créer l'ensemble des modèles supports de l'application. Ceci donnera :

Pour les modèles sociaux :

L'application devant être disponible en libre-service, elle n'utilisera aucune identification de l'utilisateur ; aussi l'application n'aura-t-elle besoin d'aucun modèle de type SM1 (modèle profil utilisateur), SM2 (modèle d'organisation) ni SM3 (Modèle de sécurité).

Pour les modèles environnementaux :

Pour identifier l'ensemble des types de bâtiments et de services, l'application utilisera une ontologie géographique qui sera modélisée dans un modèle géographique EM1 dont un extrait est donné au niveau de la Figure 5.14.

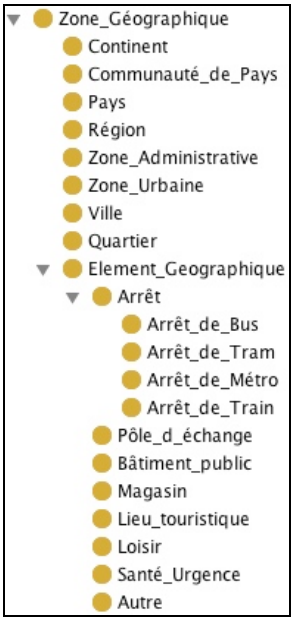


Figure 5.14. Extrait de l'ontologie géographique utilisée dans le cadre de l'application table d'orientation fonctionnant sur une borne interactive

Dans cet exemple de modélisation, la table d'orientation sera considérée comme étant une application ne disposant d'aucun lien avec un système externe et donc comme n'ayant besoin d'aucun modèle de type EM2 (le seul lien avec l'extérieur étant, pour l'application table d'orientation, l'accès à une base de données qui sera définie à travers un modèle de données BM2).

De même, l'application table d'orientation sera considérée comme ne réagissant à aucun événement et donc comme n'ayant besoin d'aucun modèle d'événement.

Pour les modèles comportementaux :

L'ensemble des modèles comportementaux (modèles des actions (BM1), modèles des données (BM2) et règles métier (BM3)) étant liés aux modèles d'interaction associés à l'application, ceux-ci seront définis au fur et à mesure de la création de ces modèles.

Une fois les principaux modèles de support créés, l'étape suivante consiste à identifier l'ensemble des processus métier qui constituent l'application.

5.3.3.2 Identification et répartition des processus métier

Dans PERCOMOM, une des premières étapes, dans la création des modèles de processus métier (modèles IM1), consiste à identifier, du point de vue fonctionnel, l'ensemble des processus métier associés à l'application (cf. chapitre 3, §3.1.2.2). Cette identification des processus métier se fait par des experts métier du domaine connaissant l'ensemble des attentes fonctionnelles des utilisateurs finaux vis-à-vis de l'application. Pour l'application table d'orientation présentée dans le chapitre 5, §5.3.1, l'application peut être vue, comme possédant deux groupes principaux de processus métier :

- Le premier est associé à l'ensemble des fonctionnalités qui présentent une valeur ajoutée du point de vue de l'utilisateur par rapport à une carte classique. Dans la suite de cette section, nous appellerons ces processus les "processus à valeur ajoutée".
- Le deuxième est associé à l'ensemble des fonctionnalités directement liés à la carte géographique (déplacement agrandissement, changement de format, etc.).

Ce premier découpage étant déterminé, il faut ensuite définir le contenu exact de chaque groupe de processus métier ; c'est-à-dire définir l'ensemble des buts métier que pourra réaliser l'utilisateur en passant par ces groupes métier.

Pour le premier groupe de processus métier, en partant de l'analyse des besoins, par les experts métier du domaine, il apparaît que celui-ci devra permettre :

- De rechercher un itinéraire ; c'est-à-dire de chercher comment se rendre d'un point A à un point B avec les transports en commun. Cette recherche devra retourner une "fiche" itinéraire contenant l'ensemble des informations nécessaires et suffisantes permettant à l'usager d'effectuer son trajet. Elle devra aussi afficher l'ensemble du trajet (du point de vue des lieux géographiques à traverser) sur la carte.
- De recherche des services de proximité ; c'est-à-dire chercher l'ensemble des services (administratifs, commerciaux, etc.) disponibles à proximité d'un point déterminé sur la carte géographique. Le type de service (exemple : cinéma, restaurant, hôpital, etc.) devra pouvoir être sélectionné par l'utilisateur et les lieux correspondant être affichés sur la carte à l'aide de pictogrammes. L'utilisateur devra aussi pouvoir afficher plusieurs types différents de services de proximité en même temps sur la carte.
- De rechercher une adresse sur la carte ; c'est-à-dire de pouvoir entrer une adresse et d'afficher le lieu géographique correspondant à cette adresse sur la carte. Une fois cette information affichée, l'usager pourra demander au système de rechercher l'itinéraire pour se rendre, en transports en commun, du lieu où il se trouve au lieu géographique associé à l'adresse qu'il a indiqué.

Pour le deuxième groupe de processus métier, en partant de l'analyse des besoins, par les experts métier du domaine, il apparaît que celui-ci devra permettre :

- de se déplacer sur la carte avec des fonctionnalités simples du type "en haut", "en bas", "vers la droite" ou encore "vers la gauche" ;
- de pouvoir zoomer en avant et en arrière sur la carte ;
- de pouvoir recentrer la carte sur le lieu où est localisée la borne interactive ;
- de pouvoir sélectionner des types d'affichage de cartes différents : mode plan, mode satellite ou mode mixte (satellite et plan).

À titre d'information, la Figure 5.15 présente les deux menus associés aux deux groupes de processus métier dans l'application finale de la table d'orientation présentée dans le chapitre 5.3.1.

(a)

(b)

Figure 5.15. Les deux groupes principaux de processus métier au niveau de la page d'accueil de la table d'orientation : (a) processus à valeur ajoutée qui apporte une valeur ajoutée par rapport à la consultation d'une simple carte, (b) processus directement lié aux opérations qui peuvent être effectuées sur l'élément graphique de type carte géographique

Pour les deux groupes principaux de processus métier, l'ensemble de ces processus métier devant être accessible, de manière identique, pour l'ensemble des usagers, la phase d'analyse et répartition des différents processus métier entre les différentes catégories de vécu des déplacements n'est pas nécessaire pour cette application.

Une fois les principaux processus métier identifiés, il convient d'analyser chaque processus métier de manière à déterminer l'ensemble des sous-processus qui le constitue. Ceci nous permettra de définir, dans PERCOMOM, le contenu du modèle de processus métier associé (IM1) associé à chaque processus métier identifié au niveau fonctionnel par les experts métier. Ainsi, si on prend l'exemple du processus métier associé à la recherche d'une adresse sur la carte, celui-ci est constitué de trois sous-proccessus s'exécutant dans cet ordre :

- un premier sous-processus permettant d'entrer une adresse que nous appellerons sous-processus "Entrer_Adresse" ;
- un deuxième sous-processus permettant de visualiser un lieu géographique sur une carte à partir d'une adresse que nous appellerons sous-processus "Visualiser_Lieu_A_Partir_Adresse" ;
- un troisième sous-proccessus permettant de calculer un itinéraire à partir d'une adresse que nous appellerons "Calculer_Itinéraire_A_Partir_Adresse".

Dans le cadre de la modélisation de niveau CIM, la décomposition d'un processus métier en sous-processus métier est modélisée dans un modèle de processus métier (IM1). La Figure 5.16 présente ainsi le modèle IM1 associé au processus métier associée à la recherche d'une adresse sur la carte.

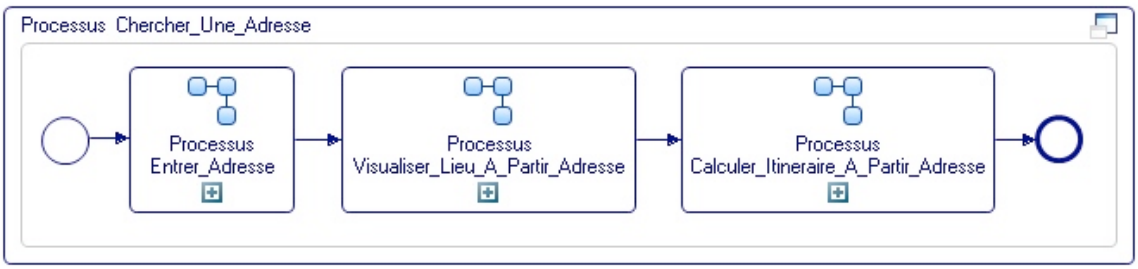


Figure 5.16. Modèle de processus métier (IM1) associé au processus métier de recherche d'une adresse sur une carte

Dans PERCOMOM, la détermination des sous-processus métier se fait sur les bases suivantes :

- Un sous-processus est composé de plusieurs tâches élémentaires. Uune tâche élémentaire est une tâche qui ne permet de réaliser qu'une seule action qui ne peut plus être décomposée du point de vue des modèles de processus métier.
- Un sous-processus doit permettre de réaliser un but métier bien précis et doit contenir l'ensemble des tâches permettant de l'atteindre. Par exemple, le sous-processus "Entrer_Adresse" doit contenir l'ensemble des tâches permettant d'écrire et de valider une adresse dans le cadre d'une application de type WIMP.
- Un sous-processus doit être réutilisable (un sous-processus doit pouvoir être utilisé par un autre processus que le processus dans lequel il a été identifié à l'origine). Ainsi, si on prend le sous-processus "Entrer_Adresse", celui-ci est utilisable dans le processus métier "Chercher_une_adresse" mais aussi dans tous les autres processus métier nécessitant d'entrer une adresse comme, par exemple, un processus métier de recherche d'un numéro de téléphone à partir d'une adresse.

Dans cet exemple de modélisation d'une application avec PERCOMOM, nous ne décrirons pas l'ensemble des sous-processus de manière à simplifier la compréhension de la méthode. Néanmoins, il est important de noter que la phase d'identification des processus et sous-processus est une phase essentielle de la modélisation dans PERCOMOM et, qu'à ce titre, elle doit faire l'objet d'un intérêt tout particulier des personnes en charge de la modélisation.

Une fois l'ensemble des processus et sous-processus métier identifés, la phase suivante de la phase d'analyse consiste à définir les liens entre les processus et les sous-processus métier du point de vue des interactions afin de créer l'ensemble des modèles métier (IM1) de l'application.

5.3.3.3 Création des modèles de processus métier (IM1) de l'application

Dans le cadre l'application table d'orientation, nous avons déterminé, dans la section précédente, qu'il y avait deux groupes de processus métier. La question qui se pose maintenant est de savoir comment se fera l'accès à chaque groupe de processus métier pour l'utilisateur. Pour répondre à cette question, les experts métier en charge de la modélisation doivent étudier avec précision les besoins

fonctionnels énoncés au niveau l'application. Dans le cadre de notre exemple, en nous basant sur notre travail de rétro-ingénierie, nous posons comme hypothèse que les besoins sont les suivants :

- L'application doit permettre d'accéder directement à une carte géographique centrée sur l'endroit où se trouve l'utilisateur.
- L'application doit permettre, à tout moment, d'accèder aux deux groupes principaux de processus métier.

De ces besoins, il en découle que l'application, à son ouverture, devra afficher une carte géographique tout en permettant l'accès à l'ensemble des sous-processus associés aux deux groupes principaux de processus métier.

Du point de vue des interactions, ceci signifie que l'ouverture de l'application va afficher un *UIGroup* contenant l'ensemble des processus métier associés aux deux principaux groupes d'interaction. La question qui se pose alors et de savoir comment activer chaque processus métier ? Aucun processus n'étant, *a priori*, prioritaire, ceux-ci doivent tous être accessibles de la même manière et ne s'exécuter que sur une demande explicite de l'utilisateur. Cette demande explicite sera réalisée à l'aide d'éléments d'interaction de type *UIAction* qui permettront chacun de démarrer un processus métier bien particulier.

Ces éléments étant définis, la Figure 5.17 présente un extrait du modèle métier (IM1) (processus "ProcessusPrincipalTableOrientation") associé au démarrage de l'application et dans lequel n'est représentée que la partie concernant le premier groupe principal de processus métier de manière à en faciliter la lecture. Ce processus métier commence par une action de type démarrage. La première tâche réalisée consiste à activer l'*UIGroup* ("UIGroup_TableOrientation") contenant l'ensemble des éléments d'interaction associés aux différents processus associés au premier groupe principal de processus métier. Une fois l'*UIGroup* activé, l'utilisateur peut sélectionner, en fonction de son choix propre, une action ("UIActionItineraire", "UIActionServicesDeProximité" ou "UIActionAdresse") permettant d'activer un des processus métier du premier groupe principal de processus métier. Ainsi, l'action "UIActionItineraire" permet d'activer le processus métier "Itinéraire" qui permet de rechercher un itinéraire. L'action "UIActionServiceDeProximité" permet d'activer le processus métier "ServicesDeProximité" qui permet de rechercher un service de proximité sur la carte. Enfin, l'action "UIActionAdresse" permet de l'activer le processus métier "Adresse" qui permet de rechercher une adresse sur la carte et à partir de cette adresse de calculer comment s'y rendre avec les transports en commun. Une fois qu'un processus se termine l'utilisateur peut choisir un autre processus à exécuter.

L'application étant déployée sur des bornes interactives, elle n'est pas prévue pour être arrêtée par un utilisateur ; aussi, le processus "ProcessusPrincipalTableOrientation" ne dispose pas d'événement de fin. L'arrêt de l'application se fera à l'aide d'une autre application, dite application d'administration, permettant de gérer l'ensemble des applications en cours d'exécution sur la borne interactive et d'éxécuter, pour chacune de ces applications, un processus métier spécifique mettant fin à l'exécution de l'application. Ceci se traduira, au niveau des modèles statiques d'interaction (IM2) associés à l'application d'administration par la définition d'*UIFieldNavigation* ; ceux-ci permettant de démarrer un processus métier sur une autre application.

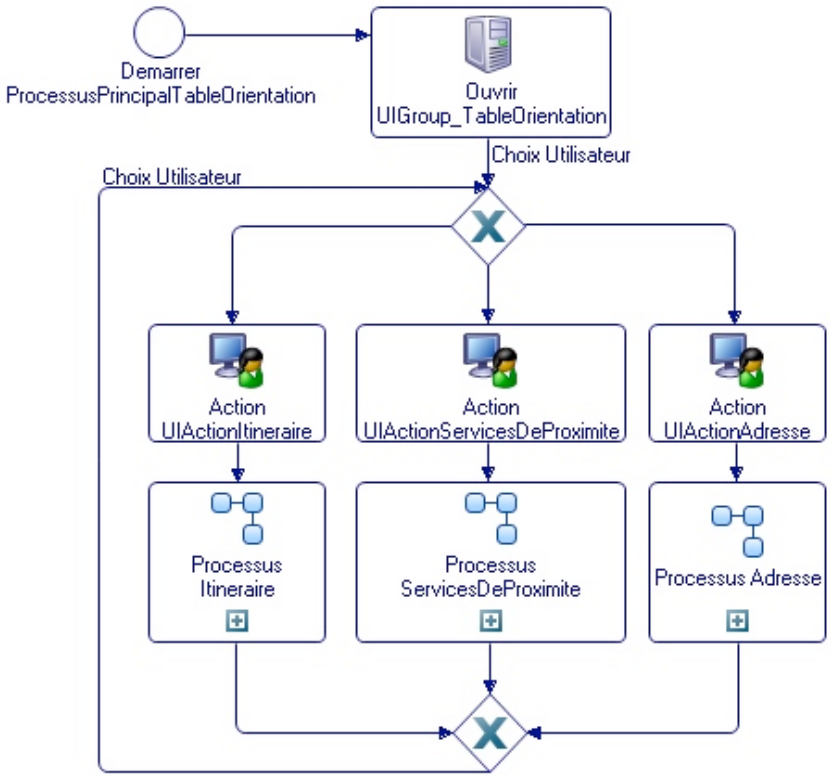


Figure 5.17. Extrait du processus métier principal (IM1) "DémarrerTableOrientation" limité à l'activation des processus métier associés au premier groupe de processus métier (les processus à valeur ajoutée) de l'écran initial de l'application table d'orientation

La création des autres modèles de processus métier (IM) étant basée sur les mêmes principes, nous nous limiterons à la présentation de ce processus au niveau de notre exemple de modélisation d'une application dans PERCOMOM.

Dans PERCOMOM, la création des processus métier (modèles IM1) étant fortement liée aux modèles statiques d'interaction (modèles IM2), ceux-ci sont bien souvent créés en parallèle des modèles de processus métier. Il est important de voir maintenant comment ceux-ci sont créés pour l'application table d'orientation.

5.3.3.4 Création des modèles statiques d'interaction (IM2) de l'application

Dans la table d'orientation, les besoins fonctionnels définis par les experts au niveau des interactions sont les suivants :

- L'application doit permettre d'accéder directement à une carte géographique centrée sur l'endroit où se trouve l'utilisateur.
- L'application doit permettre, à tout moment, d'accéder aux deux groupes principaux de processus métier.

Il est important de noter ici que, dans PERCOMOM, les experts métier définissant les besoins fonctionnels ne sont pas forcément les mêmes que ceux réalisant la modélisation. Par contre, dans les deux cas, ce sont obligatoirememnt des experts métier du domaine métier associé à l'application.

En partant des hypothèses définies plus haut, il en découle que l'application, au niveau de ses interactions, devra, lors de son démarrage, ouvrir un ensemble logique d'éléments d'interaction permettant de visualiser la carte géographique et permettant d'accéder à l'ensemble des processus métier. Cet ensemble logique d'éléments d'interaction sera un *UIGroup* qui portera le nom de "UIGroup_TableOrientation".

La question qui se pose ensuite est de déterminer les différents *UIUnit* qui vont composer cet *UIGroup* ; c'est-à-dire les différents ensembles d'éléments d'interaction qui forment chacun un tout logique du point de vue métier. Après une première analyse des besoins, les experts métier en charge de la modélisation de l'application table d'orientation identifient cinq *UIUnit* différents qui sont :

- *UIUnit_AffichageCarte* : contient les éléments d'interactions présents sur la page d'accueil.
- *UIUnit_Itinéraire* : contient les éléments d'interaction permettant d'effectuer une recherche d'itinéraire.
- *UIUnit_ServiceDeProximité* : contient les éléments d'interaction permettant d'effectuer une recherche de services de proximité.
- *UIUnit_Adresse* : contient les éléments d'interaction permettant d'effectuer une recherche à partir d'une adresse.
- *UIUnit_NavigationCarte* : contient les éléments d'interaction permettant de naviguer sur la carte (haut, bas, droite, gauche, etc.).

Le modèle statique d'interaction (IM2) pour l'UIGroup "UIGroup_TableOrientation" est donné à la Figure 5.18.

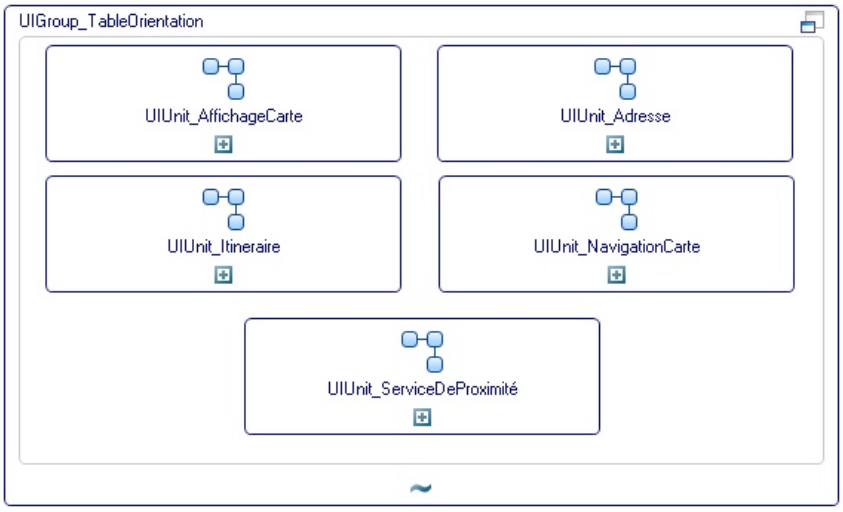


Figure 5.18. Modèle statique de groupe logique d'interaction associé à la table d'orientation présentée dans le chapitre 5.3.1

Une fois les *UIUnit* déterminés, il convient de définir l'ensemble des éléments d'interaction faisant partie de chaque *UIUnit.* La Figure 5.19 présente un exemple de définition statique de l'unité logique d'interaction "UIUnit_NavigationCarte" contenant l'ensemble des éléments d'interaction permettant de naviguer sur une carte ; dans le cas présent des éléments de type *UIAction* permettant de réaliser une action précise sur la carte du point de vue métier (se déplacer vers le haut, vers le bas, vers la droite, vers la gauche, etc.).

Figure 5.19. Modèle statique d'UIUnit associé à la partie navigation sur la carte géographique de l'application table d'orientation

Il est intéressant de noter au niveau de cet exemple que les éléments d'interaction de type *UIFieldAction* peuvent, à partir des hypothèses fixées au moment de l'analyse des besoins par les experts métier, être représentés, au niveau de la plateforme technique d'interaction (PDA, ordinateur portable, etc.), de deux manières différentes :

- soit à l'aide d'un bouton contenant une image,
- soit à l'aide d'un bouton contenant un texte.

Afin de pouvoir s'adapter aux différentes contraintes liées aux différents types de plateformes techniques et au contexte, chaque *UIFieldAction* pourra avoir, dans l'application finale, une représentation différente en fonction de la plateforme et du contexte. Le Tableau 5.2 présente un exemple de ce type d'adaptation pour un élément d'interaction de type *UIFieldAction* permettant de rechercher et d'afficher la liste des activités sportives actualisées pour un endroit donné.

Tableau 5.2. Exemple de représentation graphique d'un élément d'interaction de type *UIAction* en fonction de la localisation géographique de l'utilisateur et des capacités techniques de la plateforme utilisée pour accéder à l'application

Représentation graphique	Commentaire
Afficher la liste des activités sportives à Lille	Représentation graphique de l'UIFieldAction à Lille pour une plateforme de niveau "Haute qualité"
Afficher la liste des activités sportives à Valenciennes	Représentation graphique de l'UIFieldAction à Valenciennes pour une plateforme de niveau "Haute qualité"
Le sport à Valenciennes	Représentation graphique de l'UIFieldAction à Valenciennes pour une plateforme de niveau "Moyenne qualité"
SportVal	Représentation graphique de l'UIFieldAction à Valenciennes pour une plateforme de niveau "Basse qualité"

Ceci est rendu réalisable par la possibilité, au niveau CIM, via l'artefact associé à chaque élément d'interaction de type *UIFieldAction*, de pouvoir lui associer un élément d'interaction de type Image (*UIFieldPicture*) et/ou un élément d'interaction de type texte statique (*UIFieldText*) à travers l'utilisation de deux propriétés particulières.

- La première propriété *TexteAssocié* permet d'associer à l'élément *UIFieldAction* un identifiant numérique qui permet de retrouver, dans un fichier annexe de configuration, la chaîne de caractères qui sera associée à l'élément *UIFieldAction* en fonction de la langue et du contexte (représenté par un domaine de validité). Par exemple, si on suppose qu'on désire associer un texte à l'*UIFieldAction* permettant de se déplacer vers la droite sur une carte, et qu'on définisse que :
  - Ce texte correspond à l'identifiant 88.
  - Ce texte n'aura que des libellés avec un niveau de qualité "Haute qualité".
  - Ce texte sera disponible en français, en anglais et en allemand.
  - Ce texte n'aura aucun domaine de validité associé

  Alors la définition de ce texte, sous la forme d'un "libelletexte" dans le fichier annexe de configuration sera la suivante :

```
<libelletexte id="88">
   <libelle niveauqualite="Haut" langue="FRE" texte="Vers la droite" domainevalidite=""/>
   <libelle niveauqualite="Haut" langue="ENG" texte="To the right" domainevalidite=""/>
   <libelle niveauqualite="Haut" langue="GER" texte="Nach Recht" domainevalidite=""/>
</libelletexte>
```

  Le Tableau 5.3 présente l'ensemble des propriétés associées à un libellé de type texte dans le fichier de configuration annexe des libellés de type texte au niveau CIM.

Tableau 5.3. Ensemble des propriétés associées à un libellé de type texte dans le fichier de configuration annexe des libellés de type texte de niveau CIM

Propriété	Type	Description
Niveauqualite	Optionnel	Permet de définir à quel niveau de qualité de plateforme sera associé le texte ("Haut" pour niveau de qualité élevé, "Moyen" pour niveau de qualité "Moyenne" et "Bas" pour niveau de qualité bas). La valeur par défaut est égale à "Haut"
Langue	Optionnel	Permet d'indiquer le code de la langue sur 3 lettres au format ISO 639-2. S'il n'y a pas de valeur indiquée, le libellé est considéré comme étant valide et identique pour toutes les langues.
Texte	Obligatoire	Permet d'indiquer la chaîne de caractères associée au libellé.
domainevalidité	Optionnel	Permet d'associer au libellé un domaine de validité spécifique à l'aide de son identifiant.

- La deuxième propriété *ImageAssociée* permet d'associer à l'élément *UIFieldAction* un identifiant numérique qui permet de retrouver, dans un fichier annexe de configuration, l'identifiant de l'image qui sera associée à l'élément *UIFieldAction* ; cette association se faisant en fonction de la langue et du contexte (représenté par un domaine de validité). Les images sont, pour leur part, définies au niveau PIM dans une base de données où il est possible de retrouver l'image physique à partir de l'identifiant indiqué au niveau de la propriété *ImageAssociée*. Ainsi, supposons qu'on désire associer à l'*UIAction* une image associée portant l'identifiant 108 et ayant trois images physiques différentes en fonction de la langue de l'utilisateur. Le fichier de configuration annexe associé aux images, qui est basé sur un formalisme similaire au formalisme utilisé pour les libellés de type texte, sera alors le suivant :

```
<imageassociee id="108">
   <image niveauqualite="Haut" langue="FRE" idimage="14"  domainevalidite=""/>
   <image niveauqualite="Haut" langue="ENG" idimage="15"  domainevalidite=""/>
   <image niveauqualite="Haut" langue="GER" idimage="16"  domainevalidite=""/>
</libelletexte>
```

Dans le cas où la propriété ImageAssociée n'est pas renseignée, et/ou, au niveau CIM, la plateforme technique ne peut afficher les images, c'est l'élément de type texte qui sera utilisé au niveau de l'application finale.

La création des autres modèles statiques d'interaction (IM2) étant basée sur les mêmes principes, nous ne décrirons pas dans ce mémoire, pour des questions de place, l'ensemble des autres modèles IM2 définis au niveau de l'application associée à la table d'orientation.

Une fois les modèles statiques d'interaction (IM2) et les modèles de processus métier (IM1) définis, la modélisation de niveau CIM de l'application est en principe terminée (les autres modèles support ayant été créés et/ou modifiés pendant la création de ces deux modèles principaux). L'étape suivante de PERCOMOM consiste alors à intégrer l'ensemble de ces modèles pour créer l'application complète au niveau CIM puis à valider l'ensemble de l'application au niveau CIM.

5.3.3.5 Intégration des différents modèles et validation de l'ensemble de la modélisation de niveau CIM.

Dans l'état actuel d'avancement de nos travaux, il est important de noter que la phase d'intégration et de validation des modèles ne s'appuie pas encore sur des outils mais doit se faire manuellement. Ce manque, clairement identifié, fait partie de nos perspectives de recherche que nous présenterons dans le chapitre 6.

La première étape, au niveau de l'intégration des différents modèles de niveau CIM, consiste à créer le modèle d'application (MA) associé à l'application. L'analyse des besoins fonctionnels de l'application table d'orientation par les experts métier, à travers notre travail de rétro ingénierie, nous fournit les hypothèses suivantes :

- L'application n'aura pas de contrôle d'accès (tous les utilisateurs pourront y avoir accès)
- L'application utilisera le français comme langue par défaut pour l'ensemble de ses contenus.
- L'application démarrera automatiquement le processus métier "ProcessusPrincipalTableOrientation" afin d'accéder directement au plan et aux actions associées.
- L'application utilisera comme processus par défaut le processus métier "ProcessusPrincipaleTableOrientation". Pour rappel, le processus métier par défaut est le processus métier qui est appelé lorsque l'utilisateur tente d'exécuter un processus métier non disponible.
- L'application utilisera un seul silo de données (cf. chapitre 4, §4.4.3), qui portera le nom de "Silo de données TableOrientation", quel que soit le contexte dans lequel l'application est utilisée.

Le modèle d'application (MA) associé à l'application de la table d'orientation sera alors le suivant :

```
<application>
  <identite identifiant="03" nom="TableOrientation">
     <categorie nom="Information"/>
  </identite>
  <proprietesgenerales>
     <languepardefaut code="FRE"/>
  </proprietesgenerales>
  <proprietesapplicatives>
     <businessprocessdemarrage nom="ProcessusPrincipalTableOrientation"/>
     <businessprocesspardefaut nom="ProcessusPrincipalTableOrientation"/>
  </proprietesapplicatives>
  <silodonnees>
     <silo nom="Silo de données TableOrientation" domainevalidite=" "/>
  </silodonnees>
</application>
```

La validation des modèles de niveau CIM se faisant, pour l'instant, de manière entièrement manuelle, nous ne la présenterons pas dans ce mémoire mais la considérons comme une perspective de recherche (cf. chapitre 6).

5.3.4 Conclusion sur la première application

Comme nous l'avons vu à travers ce premier exemple, la méthode PERCOMOM permet de modéliser l'ensemble d'une application au niveau CIM à travers la création de tout un ensemble de modèles spécifiques à chaque type de problématique rencontrée. À travers son approche globale de la modélisation des applications interactives, PERCOMOM permet d'enrichir les modèles d'interface avec des informations sur le contexte, des informations sur les utilisateurs ou encore des informations sur les concepts métier manipulés dans l'application.

De notre point de vue, ceci est un des avantages principaux de PERCOMOM car, si la séparation de l'interface des autres éléments de l'application s'explique au niveau technique dans le cadre d'une approche de type MVC, elle ne se justifie pas au niveau conceptuel. En effet, à ce niveau, les interactions ne sont pas une finalité mais un moyen d'établir un dialogue entre l'homme et la machine. En ce sens, elles servent à enrichir les processus métier liés à l'application mais ne sont pas au cœur de l'application.

C'est dans ce sens qu'a été conçue PERCOMOM qui, au niveau CIM, à travers l'utilisation de propriétés dans les artefacts associés aux différents types d'éléments d'interaction, permet d'établir des liens avec les autres modèles de niveau CIM. Une des problématiques des méthodes et outils de modélisation réside dans leur capacité à transformer les modèles conceptuels en applications concrètes. PERCOMOM, dans l'état actuel de nos travaux, apporte un certain nombre de réponses à cette problématique comme nous allons maintenant le voir.

5.4 Un cas pratique de transformation de modèles dans PERCOMOM : des modèles conceptuels de niveau CIM de l'application à l'application concrète

5.4.1 Introduction

PERCOMOM est une méthode de modélisation des applications interactives personnalisées qui présente la spécificité de s'appuyer sur une architecture de type MDA disposant de frameworks techniques au niveau PIM et au niveau PSM (cf. chapitre 3, §3.2). Ainsi, en s'appuyant sur les principes de l'approche MDA, il est, en théorie, possible de générer, de manière semi-automatique, à partir des modèles conceptuels de niveau CIM d'une application, une application physique, manipulable par des utilisateurs, pour une plateforme technique déterminée comme un PC portable par exemple (cf. chapitre 3 §3.3). C'est ce que nous allons montrer dans cette partie à travers la génération d'une application permettant aux usagers des transports en commun d'accèder à des services de loisir.

Avant de commencer, il est important de noter que, dans l'état actuel de nos travaux de recherche, la transformation des modèles de niveau CIM aux applications concrètes ne s'appuie pas encore sur des outils. Ceci fait partie de nos perspectives de recherche qui seront présentées dans le chapitre 6.

5.4.2 Présentation de l'application

Afin de faciliter la compréhension du processus de transformation de modèles, nous allons faire de la rétro-ingénierie sur une application de sélection des services de loisir dont nous allons définir les principales caractéristiques ci-dessous. Dans le cadre de cette partie, nous allons considérer que toutes les analyses fonctionnelles ont déjà été réalisées par les experts métier. Ceux-ci ont défini un certain nombre d'hypothèses qui ont permis de définir le contenu de l'ensemble des modèles de niveau CIM.

Dans le cadre du projet ANR Viatic.Mobilité, les usagers peuvent, pendant leurs déplacements, accéder à différents services de loisir en ligne à partir, de ce que nous appelerons par la suite, un "menu graphique" présenté dans la Figure 5.20. Ce "menu graphique" est composé de 9 options

différentes ("Musique", "Vidéos", "Jeux", "Informations", "Votre région", "Commerce", "Immobilier", "Sport" et "Météo") qui permettent chacune d'ouvrir une nouvelle application.

Figure 5.20. Ecran d'accès aux services de loisir dans Viatic.Mobilité

Dans le cadre de notre exemple de transformation de modèles, nous n'allons pas présenter l'ensemble des modèles associés à l'application mais uniquement ceux permettant de sélectionner des options dans le "menu graphique".

5.4.3 Les modèles conceptuels associés à l'application

Pour simplifier notre présentation, nous ne détaillerons que les transformations de modèles associés aux modèles d'interaction suivant : le modèle de processus métier (IM1) et le modèle statique d'interaction (IM2).

Du point de vue des processus métier, le "menu graphique" est vu comme un ensemble d'éléments d'interaction de type *UIFieldAction*, sélectionnables de manière indépendante les uns les autres et permettant chacun de lancer un processus bien défini. La Figure 5.21 présente un extrait du processus métier associé au menu graphique pour les trois premières options du menu ("Musique", "Vidéos" et "Jeux") ; les autres options étant basées sur le même principe de sélection.

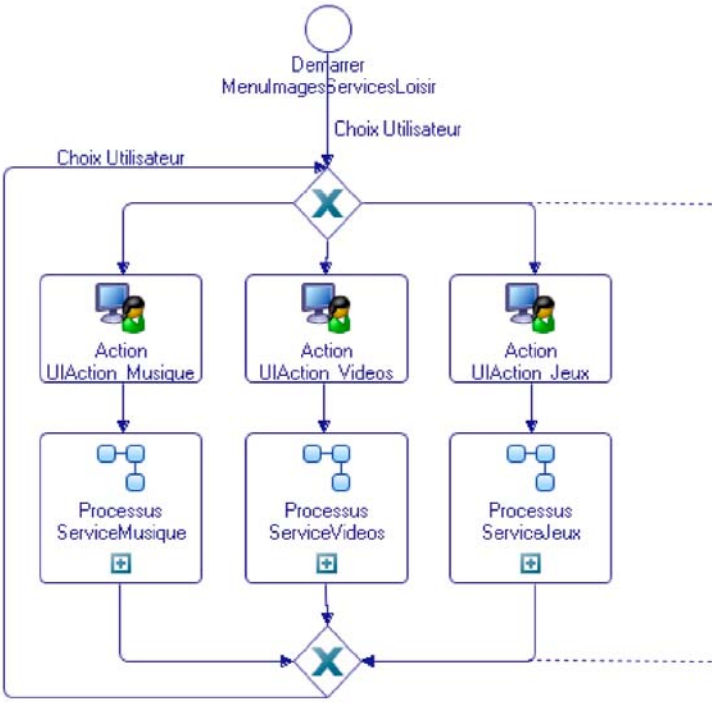


Figure 5.21. Extrait du modèle métier (IM1) associé aux trois premiers éléments du "menu graphique" permettant de sélectionner un service de loisir dans le projet ANR Viatic.Mobilité

À partir de ce processus métier et des besoins fonctionnels exprimés, il est possible de définir le modèle statique d'interaction associé au "menu graphique" permettant de sélectionner un service de loisir. L'ensemble de ce menu présentant une cohérence au niveau fonctionnel, l'ensemble des *UIField* le constituant feront partie d'un même *UIUnit* que nous appellerons "UIUnit_MenuImagesServicesLoisir". Dans cet *UIUnit*, on trouvera, associé à chaque option du menu, un *UIFieldAction* permettant de sélectionner l'option du menu graphique et un *UIFieldStaticText* permettant d'afficher un texte statique en-dessous de chaque *UIFieldAction*. La Figure 5.22 présente le modèle IM2 associé à l'*UIUnit* "UIUnit_MenuImagesServicesLoisir".

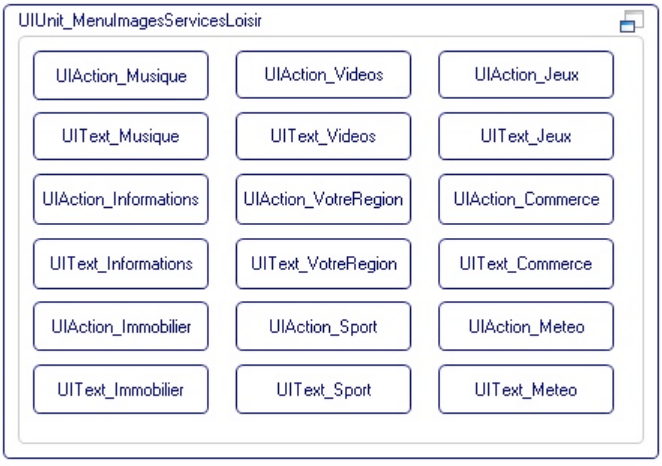


Figure 5.22 *UIUnit* (modèle IM2) associé au menu graphique de sélection des services de loisir dans le projet ANR Viatic.Mobilité

Ces deux modèles de niveau CIM étant établis, nous allons maintenant montrer comment, dans PERCOMOM, pour le "menu graphique", il est possible de passer du niveau CIM à l'application concrète. Pour rappel, l'ensemble des principes de la transformation de modèles ont été présentés dans la chapitre 3, §3.3.3.

5.4.4 Les choix techniques effectués : l'exemple des éléments de framework associés au modèle statique des interactions (IM2)

5.4.4.1 Les choix techniques effectués

Afin de permettre une présentation pratique des différentes transformations de modèles utilisées dans le cadre de PERCOMOM, nous avons choisi d'utiliser comme support l'environnement technique suivant basé sur le langage Java (cf. Figure 5.23) :

- Base de données : *MySQL*
- Serveur d'application *JBoss* entièrement écrit en Java, capable de fonctionner sur tout type de machines disposant d'une JVM
- Framework *Hibernate* version 3.2 qui permet de gérer la persistance et l'accès aux données dans les applications de type Java.
- Framework *Spring* Web MVC version 2.5 qui est un framework permettant de créer des applications web basées sur une architecture de type MVC (modèle – vue – contrôleur).
- Couche de présentation des éléments graphiques pour la plateforme cible visée : des pages *JSP*.
- Framework *Spring WebFlow* version 2.0 qui est un framework permettant la création de clients riches en langage Java utilisant des workflows fonctionnels.

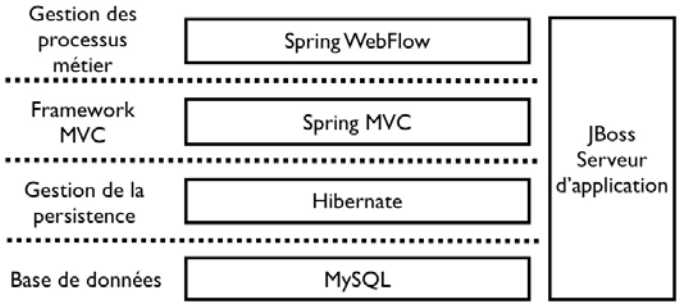


Figure 5.23 Architecture en couches utilisée dans le cadre de l'exemple de transformation de modèles

PERCOMOM étant par nature indépendant des technologies, les choix d'infrastructures techniques effectués, dans le cadre de cet exemple, ont été faits de manière arbitraire, en fonction des infrastructures techniques mises à notre disposition pendant la réalisation de nos travaux de recherche.

5.4.4.2 Le passage du niveau CIM au niveau PIM pour un modèle de processus métier (IM1)

Comme nous l'avons présenté dans le chapitre 3, §3.3, les différents modèles de niveau CIM s'appuient sur un framework de niveau PIM dans le cadre de la transformation de modèles. Ce framework est composé d'une partie spécifique à l'ensemble des services fonctionnels de niveau PIM et d'une partie associée à la traduction, au niveau CIM, du formalisme utilisé dans le cadre de la définition des modèles de niveau CIM. Ainsi, si on prend les modèles conceptuels associés aux processus métier (modèles IM1), ceux-ci sont définis principalement par :

- Des actions sur des *UIElement* au niveau des différentes tâches utilisées dans le processus métier (exemple : utilisation de l'action "Action" sur l'*UIElement* "UIAction_Musique" pour indiquer que l'*UIElement* "UIAction_Musique" est sélectionné par l'utilisateur).
- Des liens entre les différentes tâches qui peuvent soit être des liens simples (liens entre deux tâches) soit être des liens composés (plusieurs liens arrivant au même endroit) ou alors des liens de sélection (sélection de type "Choix Utilisater" par exemple).

L'ensemble des actions faisant partie du formalisme utilisé dans le cadre de la définition des modèles conceptuels de niveau CIM, celles-ci sont définies dans la partie du framework de niveau PIM prenant en charge ce formalisme. La Figure 5.24 présente, à l'aide du formalisme UML, un extrait du diagramme de classes définissant l'ensemble du formalisme associé aux *UIElement* au niveau PIM. Les actions sur les *UIElement* utilisables dans les modèles de processus métier au niveau CIM sont définies au niveau PIM sous la forme de méthodes dans les différentes classes correspondant aux différents types d'*UIElement* utilisés dans le cadre de PERCOMOM. À titre d'exemple, la Figure 5.24 représente les actions "Ouvrir" et "Fermer" associées aux *UIElement* de type *UIGroup* et l'action "Action" associée aux *UIElement* de type *UIField* ainsi qu'à l'ensemble des *UIElement* dérivés de UIField comme, par exemple, les *UIAction*.

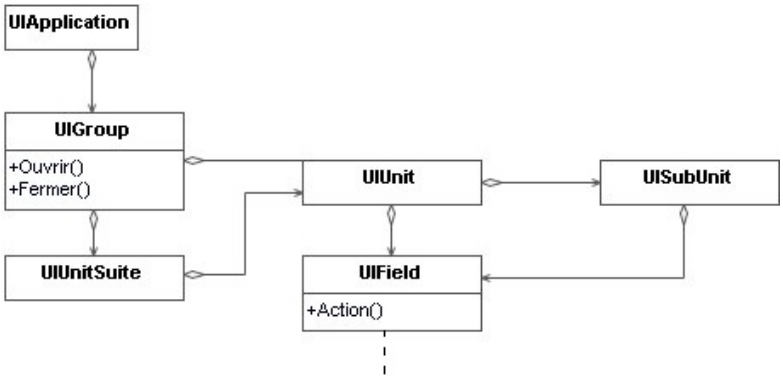


Figure 5.24. Extrait de la partie de framework de niveau PIM associé au formalisme utilisé dans les modèles de niveau CIM

Lors du passage du niveau CIM au niveau PIM, les liens entre les différentes tâches d'un modèle de processus métier (IM1) sont transformés en un fichier XML de définition du processus métier dont un exemple, pour le processus métier "MenuImagesServicesLoisir" présenté dans la Figure 5.21 est donné ci-dessous.

```
<!—Dans le fichier de définition de niveau PIM des processus métier, un processus métier -->
<!—est défini par un nom identique au nom du processus métier au niveau CIM et par un id unique -->
<processusmetier nom="MenuImagesServicesLoisir" id="11">
    <!—Chaque tâche est définie de manière individuelle -->
    <!—D'abord on indique son type qui peut-être selection pour indiquer une porte logique, -->
    <!—element si la tâche se rapporte à une action sur un UIElement ou encore sous-processus-->
    <!—pour indiquer que la tâche correspond au démarrage d'un sous-processus -->
    <!—Dans le cas d'une selection, on doit indiquer le type de critere utilise -->
    <!—Une tâche est identifiée par un modèle unique -->
    <tache id="1" type="selection" criteretype="ChoixUtilisateur">
        <!—Indication des tâches suivantes  -->
        <!—Si on utilise un critere, les tâches suivantes sont éxécutées de manière conditionnelle-->
        <!—en fonction du résultat de l'évaluation du critères associé à la sélection -->
        <suivant id="2" resultatevaluation=""/>
        <suivant id="3" resultatevaluation=""/>
        ….
    </tache>
    <!—Définition de la tâche associée à l'action "Action" sur l'UIElement "UIAction_Musique"-->
    <tache id="2" type="element" nomelement="UIAction_Musique" fonction="action">
        <suivant id="11"/>
    </tache>
    <tache id="3" type="element" nomelement="UIAction_Videos" fonction="action">
        <suivant id="12"/>
    </tache>
    ….
    <!—Exemple de définition d'exécution d'un sous-processus -->
    <tache id="11" type="sousprocessus" nomsousprocessus="ServiceMusique">
        <suivant id="20"/>
```

```
    </tache>
    <tache id="12" type="sousprocessus" nomsousprocessus="ServiceVideos">
        <suivant id="20"/>
    </tache>
    …
    <!—Définition d'une tâche correspondant à un lien composé -->
    <!—Correspond à la tâche permettant de reboucler sur la tâche de sélection 1 -->
    <tache id="20" type="selection">
        <suivant id="1"/>
    </tache>
</processusmetier>
```

La Figure 5.25 représente de manière synthétique comment se fait la transformation du niveau CIM vers le niveau PIM d'un modèle de processus métier (IM1).

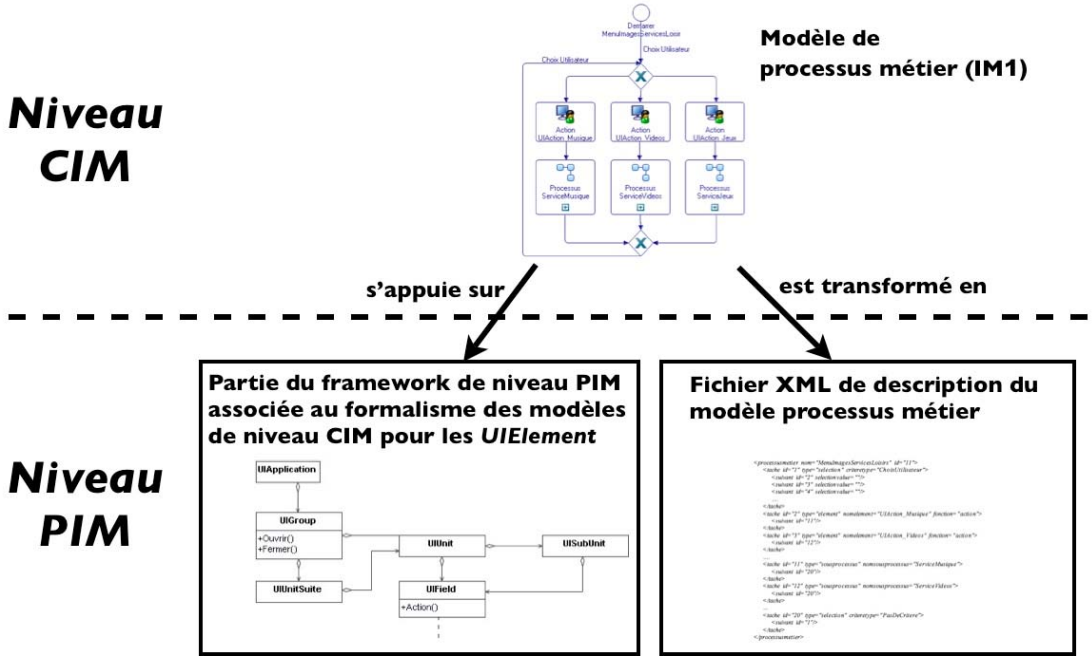


Figure 5.25. Représentation synthétique de la transformation de modèles du niveau CIM au niveau PIM pour un modèle de processus métier (IM1)

Maintenant que nous avons vu comment il est possible de passer du niveau CIM au niveau PIM pour un modèle de processus métier (IM1), nous allons présenter comment il est possible de passer du niveau CIM au niveau PIM pour un modèle statique d'interaction (IM2).

5.4.4.3 Le passage du niveau CIM au niveau PIM pour un modèle statique d'interaction (IM2)

De manière générale, un modèle statique d'interaction (IM2) de niveau CIM est basé sur les caractéristiques suivantes :

- il est basé sur l'architecture d'*UIElement* définie dans le cadre de PERCOMOM ;
- il utilise des fichiers externes de configuration pour les images et les libellés de type texte (cf. chapitre 5, §5.3.3.4) ;
- il est composé d'*UIElement* ;
- il utilise des services fonctionnels et techniques de niveau PIM à travers des propriétés ;
- il utilise des artefacts pour chaque *UIElement* pour en préciser les propriétés.

Lors du passage du modèle statique d'interaction du niveau CIM au niveau PIM, celui-ci s'appuiera sur la partie du framework de niveau PIM associée au formalisme des modèles de niveau CIM pour les *UIElement.*

Pour le passage des fichiers de configuration des images et des libellés, ceux-ci ne subissent aucune transformation lors de leur passage du niveau CIM au niveau PIM.

Au niveau PIM, la partie de framework associée aux services fonctionnels et technique est composée d'un ensemble de classes dérivées d'une classe générique "Service" dont un modèle de classe simplifié, au format UML, est présenté à la Figure 5.26.

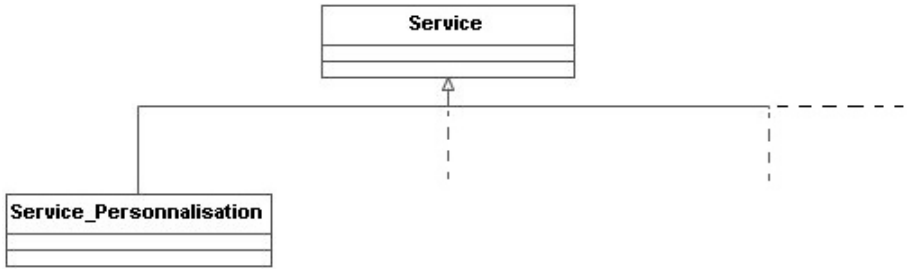


Figure 5.26. Modèle de classes simplifié associé à la partir de framework de niveau PIM associée aux services fonctionnels et techniques

Pour ce qui est de la composition du modèle statique d'interaction, c'est-à-dire des propriétés contenues dans les artefacts et des propriétés des services associés, celles-ci sont transformées lors de leur passage du niveau CIM au niveau PIM en un fichier de définition au format XML. Un exemple simplifié de ce type de fichier XML pour le modèle statique d'interaction "UIUnit_MenuImagesServicesLoisir" de la Figure 5.22 est donné ci-dessous :

```
<!—Chaque modèle statique d'interaction est défini par un nom et un id -->
<modelestatiqueinteraction nom="UIUnit_MenuImagesServicesLoisir" id="123">
    <!—Association des services fonctionnel de niveau PIM associés à l'ensemble du modèle -->
    <!—L'appel du service se fait à partie du nom du service -->
    <services nom="service_personalisation">
        <!—Définition de la valeur des propriétés définies au niveau du service fonctionnel -->
        <!—Comme l'association de ses propriétés peut être conditionnelle on définit -->
        <!—la condition associée (valeur vide s'il n'y a pas de condition) -->
        <condition type="" nom="">
            <!—Association des propriétés en fonction de la valeur du résultat de l'évaluation  -->
            <!—de la condition -->
            <resultatevaluation valeur="">
                <!—Définition de la valeur de chaque propriété du service fonctionnel -->
                <propriete nom="UtiliseLangue" valeur="Vrai"/>
                ….
            </resultatevaluation>
        </condition>
    <services>
    <!—Définition de chaque UIElement contenu dans le modèle statique d'interaction -->
    <!—en indiquant le type d'UIElement ainsi que son nom -->
    <uielement type="UIAction" nom="UIAction_Musique">
        <!—Définition éventuelle du service fonctionnel associé à l'UIElement-->
        <service nom="">
        …
        </service>
        <!—Définition de la valeur des propriétés définies au niveau de l'UIElement -->
        <!—Comme l'association de ses propriétés peut être conditionnelle on définit -->
        <!—la condition associée (valeur vide s'il n'y a pas de condition) -->
        <condition type="" nom="">
            <resultatevaluation valeur="">
```

```
            <!—Définition de la valeur des propriétés de l'artefact associé à l'UIElement -->
            <propriete nom="EstVisible" valeur "Vrai"/>
            …
        </resultatevaluation>
    </condition>
</uielement>
…
</modelestatiqueinteraction>
```

La Figure 5.27 montre de manière synthétique comment se fait la transformation du niveau CIM vers le niveau PSM d'un modèle statique d'interaction IM2.

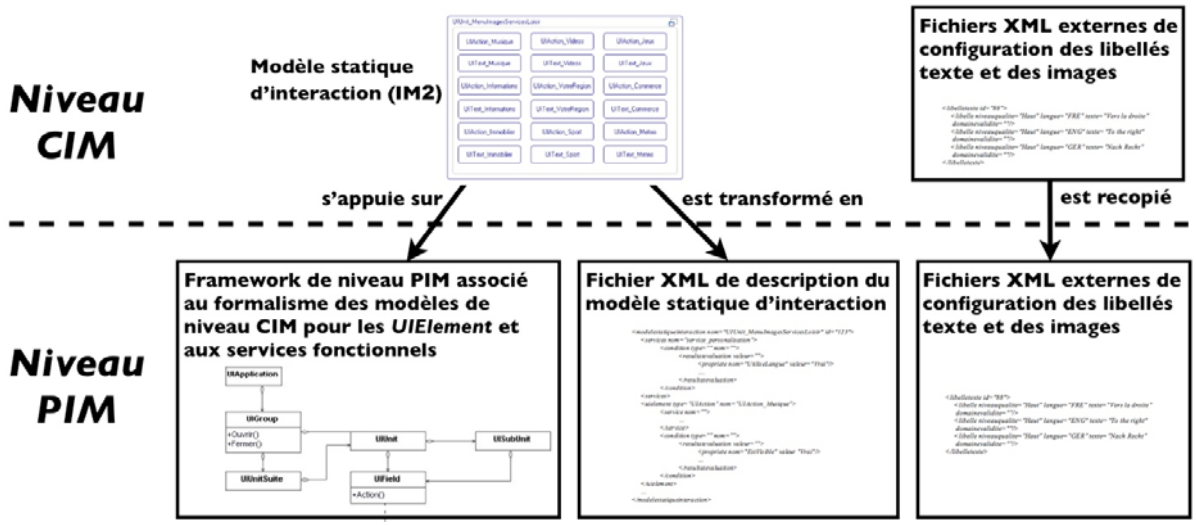


Figure 5.27. Représentation synthétique de la transformation de modèles du niveau CIM au niveau PIM pour un modèle statique d'interaction (IM2)

Après avoir vu le passage du niveau CIM au niveau PIM pour le modèle de processus métier (IM1) et pour le modèle statique d'interaction (IM2), il nous reste à étudier comment passer des modèles de niveau PIM aux modèles de niveau PSM et donc à l'application concrète.

5.4.4.4 La réalisation technique du passage du niveau PIM au niveau PSM

Dans le cadre de notre exemple, nous allons considérer que l'application ne sera générée que pour un seul type de plateforme technique de consultation : un navigateur internet disposant d'une résolution d'écran de 1024 x 768 points (la consultation est supposée se faire sur un ordinateur de type PC).

Le passage du niveau PIM au niveau PSM étant assez complexe, nous n'allons présenter dans la suite que les éléments principaux utilisés dans le cadre de cette transformation de modèles sans rentrer dans les détails.

Les éléments les plus simples à passer, du niveau PIM au niveau PSM, sont les fichiers XML externes de configuration des libellés texte et des images qui sont simplement recopiés du niveau PIM au niveau PSM.

Le framework de niveau PIM associé au formalisme des modèles de niveau CIM pour les UIElement et aux services fonctionnels de niveau PIM subit lors de son passage du niveau PIM au niveau PSM une adaptation aux spécificités de la plateforme technique de consultation.

Le fichier de XML de description du modèle de processus métier, lors de son passage du niveau PIM au niveau PSM, va donner lieu à la création d'un fichier XML de définition du workflow associé au processus métier (on suppose dans notre cas que la plateforme technique de consultation dispose d'un moteur de gestion des workflows).

Les fichiers XML de description du modèle de processus métier et du modèle statique d'interaction, lors de leur passage du niveau PIM au niveau PSM, vont donner lieu, par tissage entre les deux modèles (cf. chapitre 3, §3.3.2), à la création des pages écrans adaptées à la plateforme technique, à la création du code associé à la prise en charge de la logique métier associée aux pages écrans. Sur une plateforme technique de type PC, chaque élément de type *UIGroup* donnera lieu à la création d'une ou de plusieurs pages écrans.

La Figure 5.28 présente un schéma global simplifié de transformation de modèles du niveau CIM vers le niveau PSM pour un modèle de processus métier (IM1) et un modèle statique d'interaction (IM2). Dans un souçi de lisibilité, un seul niveau PSM a été représenté sur la figure.

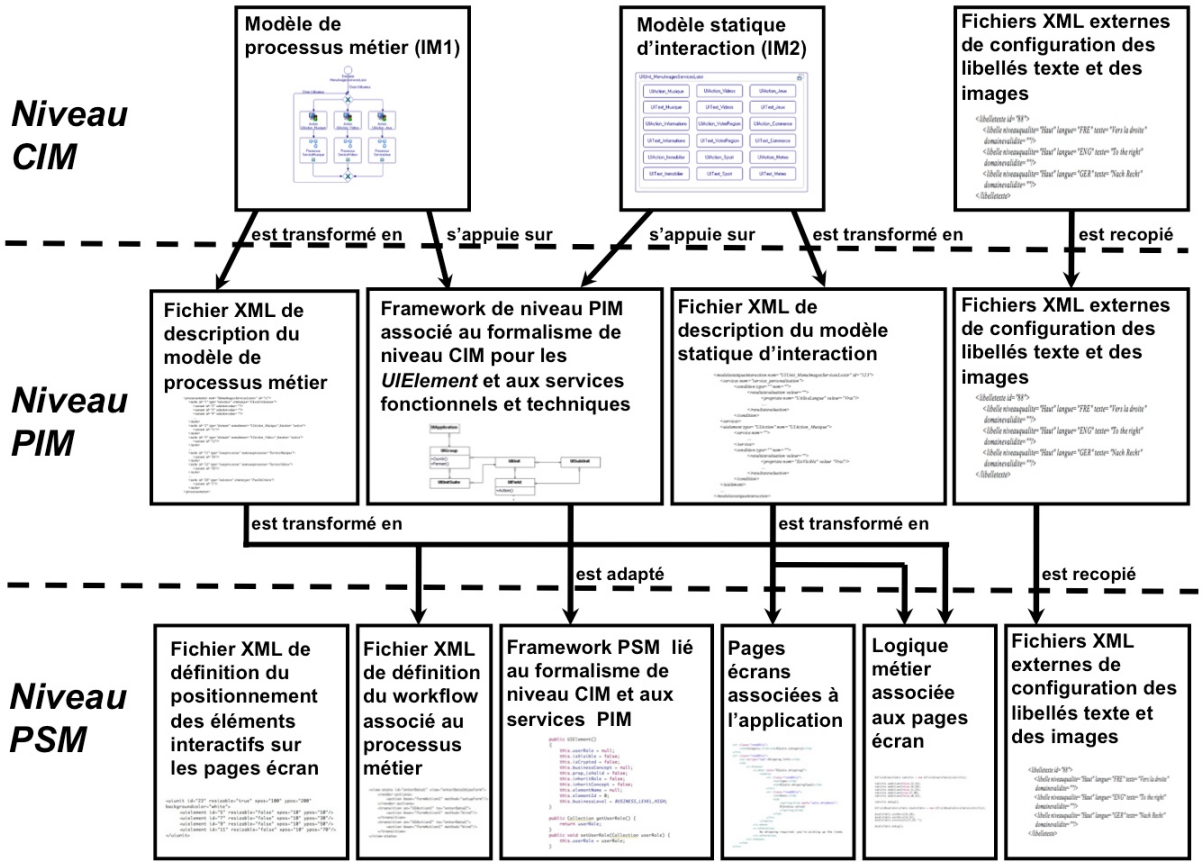


Figure 5.28. Schéma global simplifié de passage du niveau CIM au niveau PSM d'un modèle de processus métier (IM1) et d'un modèle statique d'interaction (IM2)

Le principe de transformation de modèle ayant été présenté dans sa globalité, nous allons maintenant l'appliquer au niveau de "menu graphique".

5.4.5 Exemple de transformation d'un modèle statique d'interaction (IM2)

Dans le cadre de cet exemple, il est important de noter que les transformations de modèles, d'une part du niveau CIM au niveau PIM et d'autre part du niveau PIM au PSM, ont été réalisés de manière manuelle et pas à l'aide d'outil. D'autre part, les choix techniques effectués au niveau PSM reposent sur une vision empirique du framework de niveau PSM de manière à pouvoir valider les principes généraux de la solution de transformation de modèles proposée par PERCOMOM. De manière générale, la solution technique retenue dans le cadre de l'exemple est celle présentée dans le chapitre 5, §5.4.4.1. Pour rappel, les choix techniques effectués au niveau de l'architecture relèvent de choix effectués en fonction des outils disponibles au moment de la réalisation de nos travaux de recherche.

La transformation du niveau PIM au niveau PSM des modèles IM1 (processus métier) et IM2 (modèle statique d'interaction) ayant été présentés dans le chapitre 5, §5.4.4.2 et §5.4.4.3, nous nous

intéresserons ici à une présentation pratique de la transformation de modèle du niveau PIM au niveau PSM.

En nous basant sur ce qui a été présenté dans le chapitre 5, §5.4.4, pour afficher notre "menu graphique", nous devons créer une page de type JSP associée à l'*UIGroup* ("UIGroup_AfficherMenuGraphiqueLoisir") contenant l'*UIUnit* dans lequel se trouve le "menu graphique" ("UIUnit_MenuImagesServicesLoisir"). Cet *UIGroup* est composé lui-même de trois *UIUnit* présentés, pour faciliter la compréhension du découpage en *UIUnit*, au niveau de l'application finale dans la Figure 5.29 :

- L'*UIUnit* (a) pour le groupe d'éléments d'interaction associés au menu général contenant le titre de l'application (ici Viatic.Mobilité). Le menu général regroupe, dans notre application d'exemple, toutes les fonctions métier de l'application dont les buts métier ne sont pas directement reliés aux objectifs métier principaux de l'application comme, par exemple, l'accès à l'aide en ligne ou l'envoi d'un message au responsable de l'application.
- L'*UIUnit* (b) pour le groupe d'éléments d'interaction associés au menu contextuel permettant de démarrer des processus métier directement reliés aux objectifs métier de l'application comme par exemple, donner l'accès aux informations transport.
- L'*UIUnit* (c) pour l'*UIUnit* "UIUnit_MenuImageServicesLoisir".

Figure 5.29. Les trois *UIUnit* (a, b et c) contenus dans l'*UIGroup* contenant l'*UIUnit* "UIUnit_MenuImageServicesLoisir"

En se basant sur le framework de niveau PSM associé au formalisme des modèles de niveau CIM pour les *UIElement*, cet écran sera donc géré à travers quatre objets java principaux :

- Un objet java associé à l'*UIGroup* permettant de gérer l'ensemble de la page
- Un objet java associé à l'*UIUnit* (a)
- Un objet java associé à l'*UIUnit* (b)
- Un objet java associé à l'*UIUnit* (c)

L'objet associé à l'*UIUnit* (c) sera lui-même composé de 18 objets java :

- 9 objets java associés aux images qui sont des *UIFieldAction* affichés par défaut sous forme d'images.
- 9 objets java associés aux libellés affichés sous les images qui sont des *UIFieldStaticText* de type text.

Dans le cadre de l'architecture technique, les objets Java sont construits à partir des classes Java définies dans la partie de framework associée au formalisme des modèles de niveau CIM pour les *UIElement* et à partir des propriétés définies pour chaque *UIElement*, à travers l'utilisation d'artefacts, dans les modèles statiques d'interaction de niveau CIM de l'application. Au niveau PSM, ces propriétés sont stockées dans une base de données définie dans le cadre de la "logique métier associée aux pages écran" (cf. la Figure 5.28 dans le chapitre 5, §5.4.4)

Ceci étant défini, on dispose alors, au niveau PSM, d'un ensemble de modèles et de codes permettant d'exécuter l'application définie au niveau CIM du moins du point de vue du modèle statique d'interaction (IM2). Dans la partie suivante, nous allons voir comment est utilisé un autre type de modèle à travers la transformation d'un modèle de processus métier (IM1) en modèle, en code, directement exécutable.

5.4.6 Exemple de transformation d'un modèle de processus métier (IM1)

Par définition, les modèles de processus métier sont des successions de tâches qui peuvent être effectuées aussi bien par le système que par l'utilisateur. En fonction de la plateforme technique utilisée et de leur nature, certaines tâches vont entraîner l'exécution d'actions tandis que d'autres ne vont donner lieu à aucune action au niveau de l'application. Ainsi, cliquer sur un bouton dans une interface va entraîner l'exécution d'un certain nombre d'actions, alors qu'afficher des informations à l'utilisateur dans un champ texte ne va pas entraîner systématiquement une action.

Aussi, dans le cadre de la solution technique retenue, il convient de répartir les éléments du formalisme de définition des modèles de processus métier en deux catégories : ceux qui vont exécuter des actions et les autres. Une fois cette séparation effectuée, il devient possible de définir des workflows applicatifs ne comportant que des actions. Dans le cadre de notre exemple, c'est ce que nous avons fait ; la création des workflow d'actions se faisant à l'aide du framework *Spring WebFlow*. De manière très simplifiée, les fonctionnalités et avantages de *Spring WebFlow* sont les suivants :

- Dans *Spring WebFlow*, il est possible de créer des workflows de navigation à partir d'un ensemble de pages *JSP* (une page *JSP* correspondant dans notre exemple à un écran du point de vue de l'application).
- Dans *Spring Webflow*, dans les workflows, les liens entre les pages *JSP* sont réalisés en fonction d'actions exécutées au niveau de chaque page. Ces liens peuvent être regroupés de manière à permettre de créer des workflows applicatifs très similaires aux processus métier décrits dans les modèles de processus métier (IM1). Dans ce cadre, chaque processus métier est défini sous la forme d'un workflow de navigation spécifique. Afin de faciliter la réutilisation, chaque page *JSP* peut faire partie d'un ou plusieurs processus et donc être utilisée dans un ou plusieurs workflows de navigation.
- Dans *Spring Webflow*, il est possible d'ajouter des actions à certains événements génériques associés à la page *JSP* (initialisation des contenus au chargement de la page, action à effectuer lors de la fermeture de la page, etc.).
- Dans *Spring Webflow*, il est possible de définir des workflows de navigation comme étant constitués d'un ensemble de sous-workflows de navigation spécifique reprenant une partie du workflow de navigation global.
- L'utilisation de workflow de navigation et de sous-workflow de navigation permet de faciliter la réutilisation des différents éléments définis au niveau PSM mais aussi simplifie la création de nouvelles applications. Ainsi, si un workflow de navigation est déjà défini pour une application, son utilisation dans le cadre d'une nouvelle application ne nécessitera quasiment aucune génération de code ; seules les propriétés des *UIElement* étant éventuellement différentes entre les deux applications.

Ci-dessous, on trouvera un exemple de définition d'un workflow de navigation dans *Spring Webflow*. Cet exemple correspond à une partie du fichier de configuration du processus métier de navigation associé au menu d'accès aux applications de loisir présenté dans le chapitre 5, §5.4.2. Afin de limiter le contenu de ce fichier, on ne présentera ici que les éléments directement associés à l'*UIFieldAction* permettant de démarrer le processus métier associé au menu "Musique" :

```
<!--Définition d'un état dans le worklow. Ici cet état permet l'affichage de -->
<!--l'UIGroup "UIGroup_AfficherMenuGraphiqueLoisir -->
<view-state id="afficherMenuGraphiqueLoisir" view="UIGroup_AfficherMenuGraphiqueLoisir">
        <!--Action permettant d'initialiser l'UIGroup -->
        <render-actions>
            <action bean="afficherMenuGraphiqueLoisirBean" method="initialisation"/>
        </render-actions>
        ….
        <!--Définition de l'action associée à un clic de la souris sur l'image "Musique" -->
        <!--Dans le cas présent, l'action va déclencher l'ouverture d'une autre page -->
        <!—Cette autre page étant associée au démarrage du processus ServiceMusique -->
        <transition on="UIAction_Musique" to="afficherLoisirMusique">
        </transition>
        ….
    </view-state>
```

Avec la définition des workflows dans *Spring Webflow* et la définition de pages *JSP* et la définition d'objets associés aux framework de niveau PSM pour la définition des *UIElement*, on dispose alors d'une application fonctionnelle manipulable par des experts métier.

5.4.7 Conclusion sur l'exemple de transformation de modèles

À travers cet exemple simple, nous avons montré la faisabilité de la génération automatique d'applications à partir de modèles conceptuels de niveau CIM conçus dans le cadre de PERCOMOM. Celle-ci se base principalement sur des frameworks (au niveau PIM et au niveau PSM) qui, en définissant techniquement l'ensemble des éléments associés à la notation utilisée dans les modèles de niveau CIM permettent de réduire la complexité de chaque transformation de modèle ; celle-ci se résumant, bien souvent, à un simple passage de définition et de propriétés à l'aide de fichiers XML de configuration.

Dans notre exemple, cette transformation a été effectuée en s'appuyant sur un environnement technique de type Java qui est un langage qui permet de créer des applications pour tout type de plateforme technique ; ce qui devrait garantir l'extensibilité de l'approche globale de transformation de modèles.

Maintenant que nous avons vu comment PERCOMOM permettait de modéliser une application au niveau CIM et comment il est possible de passer des modèles de niveau CIM aux applications concrètes, il nous reste une problématique à présenter qui est la prise en compte de la personnalisation des contenus dans les modèles conceptuels de nivau CIM. C'est ce que nous allons voir maintenant à travers un troisième exemple d'application.

5.5 Troisième application : la personnalisation des contenus dans le cadre d'une application permettant de visualiser les prochains départs en gare

5.5.1 Introduction

Un des éléments important de la méthode PERCOMOM réside dans sa capacité à prendre en charge des problématiques de personnalisation des contenus en même temps que des problématiques de prise en compte du contexte. Dans l'application que nous allons décrire dans cette partie, nous allons montrer, à travers une application permettant de visualiser les prochains départs en gare, comment il est possible d'intégrer ces deux problématiques dans les modèles conceptuels d'interaction (IM1) et (IM2).

5.5.2 Description de l'application

Une des premières préoccupations d'un voyageur lorsqu'il arrive en gare est de se renseigner sur les prochains départ afin de savoir quand il pourra prendre son train et surtout sur quel quai. Pour répondre à cette problématique, on va créer une application permettant d'afficher les prochains départs des trains, accessibles à travers une interface web. Cette application tiendra compte du lieu où se trouve l'utilisateur pour lui afficher les informations pertinentes. Ainsi, si l'utilisateur se trouve dans la gare de Lille Flandres, l'application lui affichera les 10 prochains départs à partir de la gare de Lille Flandres alors que s'il est dans la gare de Valenciennes, ce seront les 10 prochains départs à partir de la gare de Valenciennes qui seront affichés. Afin de pouvoir faciliter l'accès à l'information, cette application sera accessible à travers différentes plateformes de consultation. Afin de permettre une meilleure compréhension de l'application, les Figure 5.30, Figure 5.31 et Figure 5.32 présentent trois exemples d'utilisation de l'application sur un ordinateur de type PC et sur un téléphone portable de type smartphone.

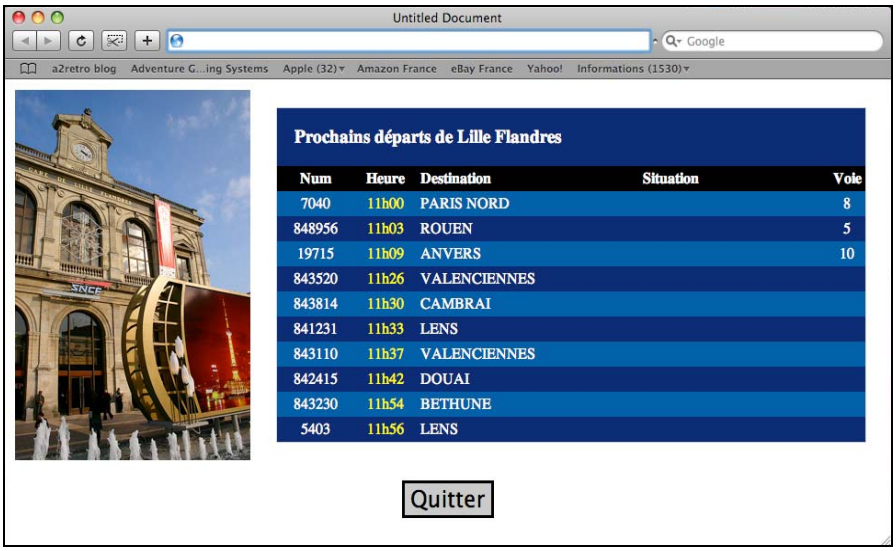


Figure 5.30. Exemple de page affichée pour un usager, sur PC, dans la gare de Lille Flandres.

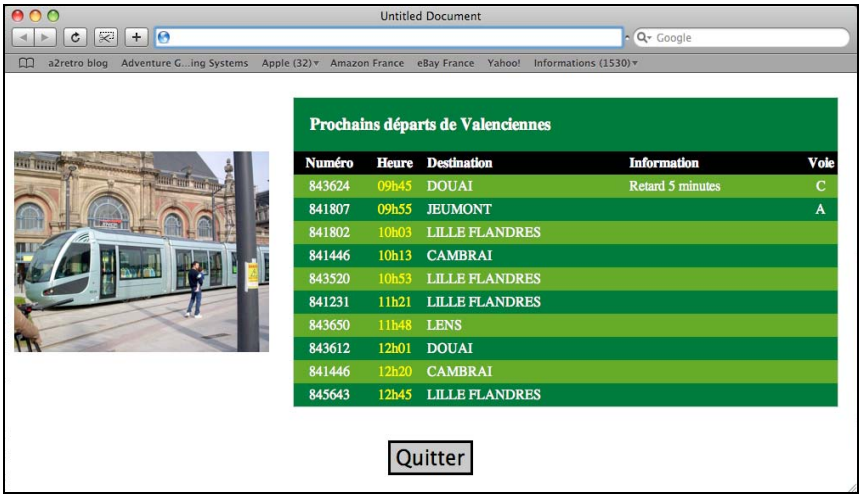


Figure 5.31. Exemple de page affichée pour un usager sur son PC, dans la gare de Valenciennes.

Figure 5.32. Exemple de page affichée, sur un téléphone, dans la gare de Valenciennes.

5.5.3 Le service de personnalisation et la prise en compte du contexte dans la modélisation de niveau CIM associée à l'application

5.5.3.1 Introduction

La méthode pour la modélisation de niveau CIM utilisée dans PERCOMOM ayant été présentée en détail dans le cadre de la première application, nous ne referons pas, dans cette troisième application, une modélisation complète de l'application mais ne présenterons que l'adaptation au contexte du point de vue des modèles d'interaction (cf. chapitre 4, §4.2).

Pour cela, nous montrerons dans les sections suivantes comment il est possible de prendre en compte le contexte dans des modèles conceptuels d'interaction de type modèle de processus métier (IM1) et de type modèle statique d'interaction (IM2).

Il est important de souligner que dans toutes les sections suivantes, on considère que l'analyse des besoins est effectuée par des experts métier du domaine et que l'ensemble des hypothèses posées l'ont été suite à des choix effectués par ces experts métier.

5.5.3.2 Le principal modèle de processus métier (IM1) de niveau CIM associé à l'application

Du point de vue métier, l'application peut se résumer à un seul processus métier, que nous appellerons "HorairesDepart" permettant d'afficher, à travers différents éléments d'interactions, les informations sur les prochains départs en gare. Ces informations évoluant dans le temps, le processus métier devra être capable de les remettre à jour périodiquement de manière entièrement automatique. En partant de ces différents éléments, la définition du modèle de processus métier (IM1) associés à "HorairesDépart" devient assez simple à réaliser.

En dehors des étapes de démarrage et d'arrêt du processus, celui-ci comprendra, d'après l'analyse effectuée par les experts métier, cinq étapes. La première consistera à ouvrir l'*UIGroup* contenant l'ensemble des éléments d'interaction ("Ouvrir UIGroup_HorairesDepart"). Puis, on exécutera, de manière automatique, une action "UIFieldAction_InitialiserHorairesDepart" qui permet de rechercher l'ensemble des horaires de départ pour le lieu où se trouve l'utilisateur. Ensuite, l'utilisateur pourra décider de terminer le processus en cours en passant par une action "UIFieldAction_HorairesDepart" qui déclenchera la fermeture de l'*UIGroup* "UIGroup_HorairesDepart" avant de mettre fin au processus. S'il ne fait rien, un timer permettra, de manière automatique, de faire une mise à jour des informations affichées toutes les 3 minutes (cf. Figure 5.33).

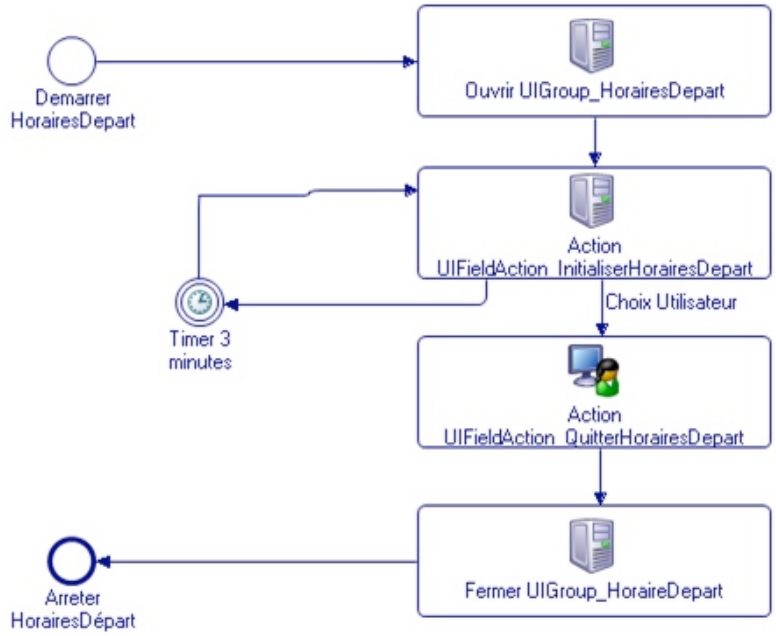


Figure 5.33. Processus métier "HorairesDepart" associé à une application permettant d'afficher les horaires des prochains départs dans la gare où se trouve l'utilisateur

Dans le cadre de la modélisation de l'application, une question qui se pose, à la personne en charge de la modélisation, est de savoir si le modèle de processus métier "HorairesDepart" est sensible au contexte ou pas. En fait, la vraie question à se poser est de savoir si l'enchaînement des tâches composant le processus métier sera différent en fonction du contexte, dans le cas présent en fonction du lieu où se trouve l'utilisateur. Dans le cas présent, la réponse est non : le processus métier sera identique ; seuls les contenus seront sensibles au contexte.

5.5.3.3 Les modèles statiques d'interaction (IM2) de niveau CIM associés à l'application

Au niveau des modèles statiques d'interaction (IM2), l'application est définie comme possédant un seul groupe logique d'interaction "UIGroup_HorairesDepart" contenant une seule unité logique d'interaction "UIUnit_HorairesDepart" dans laquelle sont définis l'ensemble des éléments d'interaction (cf. Figure 5.34). Ceci s'explique par le fait que l'ensemble des éléments d'interaction utilisés par l'application forment un tout logique qu'il n'est pas possible de séparer (c'est la définition même de l'*UIUnit* présentée dans le chapitre 3).

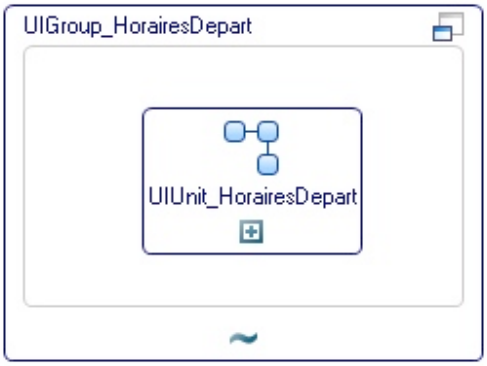


Figure 5.34. Modèle statique d'interaction (IM2) du groupe logique d'interaction "UIGroup_HorairesDepart" associé à l'application d'affichage des horaires des prochains départs dans la gare où se trouve l'utilisateur

Dans le cadre de l'application d'affichage des prochains départs, après analyse des besoins par les experts métier, ceux-ci ont défini que l'unité logique d'interaction devait contenir quatre éléments d'interactions :

- Une image représentant la photo de la gare où se trouve l'utilisateur (*UIFieldPicture_ImageGareHoraireDépart*).
- Un champ texte permettant d'afficher le nom de la gare (*UIFieldText_TitreHoraireDepart*).
- Une table d'affichage simple permettant d'afficher sous forme d'un tableau les 10 prochains départs de train (*UIFieldTableSimple_HorairesDepart*).
- Une action qui permet de mettre fin au processus métier associée (*UIFieldAction_QuiterHorairesDepart*).

Par rapport aux besoins fonctionnels exprimés par les demandeurs de l'application, les trois premiers éléments d'interaction seront sensibles au contexte géographique dans lequel l'application sera utilisée.

Comme nous l'avons vu dans le chapitre 4, §4.2, rendre un modèle statique d'interaction sensible au contexte peut se faire de différentes manières :

- Soit en utilisant un service de personnalisation des contenus.
- Soit en utilisant, pour les différents éléments d'intercation, des associations de propriétés, via les artefacts, sensibles à des informations de contexte.

Dans notre application d'exemple, l'objectif recherché au niveau de l'adaptation au contexte est d'avoir l'ensemble des informations contenues dans le modèle statique d'interaction associé à l'*UIUnit* "UIUnit_HorairesDepart" qui soient sensibles au contexte. Pour cela, on va associer à l'*UIUnit* le service de personnalisation de niveau PIM en indiquant, à travers l'utilisation de la propriété "UtiliseGéolocalisation", qu'on désire avoir une personnalisation des contenus en fonction de la localisation géographique de l'utilisateur (cf. Figure 5.35). À travers la notion d'héritage de propriétés, l'ensemble des *UIField* contenus dans l'*UIUnit* "UIUnit_HorairesDepart" vont se voir associer le service de personnalisation. Ceci ne signifie pas que chaque *UIField* sera automatiquement adapté à la

localisation de l'utilisateur mais qu'il le sera s'il dispose de contenus sensibles au contexte géographique. Ainsi, l'application ayant été définie comme étant en français, le libellé associé à l'*UIFieldAction* "UIFieldAction_QuitterHoraireDepart" n'étant pas sensible au contexte (le libellé restant toujours égal à "Quitter"), elle affichera toujours le même contenu quel que soit le lieu où se trouve l'utilisateur. Par contre, tous les contenus des autres champs étant sensibles au contexte, ceux-ci s'adapteront automatiquement à tout changement de contexte. Dans le cas où aucun contenu associé à un de ces éléments d'interaction ne serait disponible, pour un contexte particulier, l'application utilisera le premier contenu disponible permettant ainsi d'afficher systématiquement un contenu à l'utilisateur. Il est important de noter que la problématique de la cohérence du contenu fourni à l'utilisateur par rapport à la localisation ne relève pas d'une problématique de modélisation mais d'une problématique liée à la cohérence et à la disponibilité des contenus.

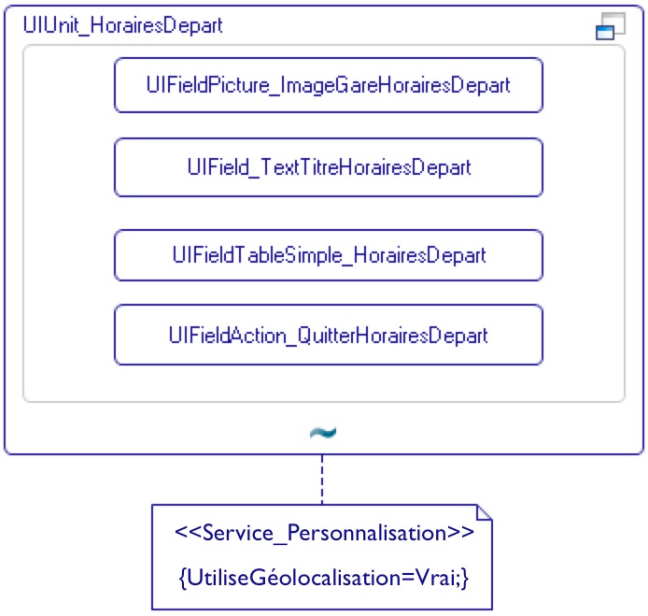


Figure 5.35. Modèle statique d'interaction (IM2) de l'unité logique d'interaction "UIUnit_HorairesDepart" associée au groupe logique d'interaction "UIGroup_HorairesDepart" dans l'application d'affichage des horaires des prochains départs dans la gare où se trouve l'utilisateur

5.5.4 Conclusion sur la troisième application

Dans cette troisième application, assez simple, nous avons présenté comment il était possible de prendre en compte une partie de la problématique de l'adaptation au contexte, au niveau des modèles CIM, dans le cadre de la phase de modélisation de PERCOMOM.

Dans la phase de modélisation de niveau CIM, rendre une application sensible au contexte consiste, dans PERCOMOM, à analyser, pour chaque modèle, quels sont les impacts du contexte sur celui-ci. Ainsi, si dans un modèle de processus métier (IM1) la prise en compte du contexte peut avoir une influence sur l'enchaînement des différentes tâches alors le modèle devra prendre en compte le contexte. Si par contre, le contexte n'a aucune influence sur l'enchaînement des différentes tâches alors le modèle ne sera pas considéré comme étant sensible au contexte. Ceci permet de n'introduire la notion de sensibilité au contexte que dans les modèles qui sont effectivement sensibles au contexte et pas au niveau de l'application globale.

Une des caractéristiques de PERCOMOM, à ce niveau, est de permettre la création de modèles conceptuels sensibles au contexte à travers une seule association d'un service fonctionnel de niveau PIM dans un modèle statique d'interaction (modèle IM2). Ceci permet d'indiquer qu'on souhaite que l'ensemble des contenus des éléments d'interaction soient adaptés au contexte sans pour autant faire de lien direct avec ces contenus. Ainsi, ce qui importe au niveau CIM, ce n'est pas de savoir si un contenu est sensible ou pas au contexte mais d'indiquer que s'il est sensible au contexte, on veut qu'il soit

adapté à ce contexte. Cette séparation entre les modèles d'interaction et les contenus utilisés par les différents éléments d'interaction permet d'avoir une séparation entre les deux permettant de les faire évoluer indépendamment les uns des autres. Ainsi, si une application est sensible au contexte et que les contenus associés aux éléments d'interaction ne le sont pas, l'application sera capable de fonctionner. Si par la suite certains contenus deviennent sensibles au contexte, ceux-ci seront automatiquement pris en compte par l'application sans avoir eu besoin de modifier les modèles conceptuels associés.

Pour terminer ce chapitre, nous allons maintenant présenter, de manière moins détaillée, quelques parties d'applications complémentaires permettant de présenter d'autres aspects de PERCOMOM.

5.6 La prise en compte de l'utilisateur dans PERCOMOM : appartenance sociale et échanges verbaux

5.6.1 Un exemple d'utilisation d'un modèle d'organisation (IM2) et d'un modèle d'action (BM1) dans le cadre d'une adaptation d'une application à l'utilisateur

Dans le cadre de la personnalisation des applications interactives, l'utilisateur est très certainement un des éléments les plus importants à prendre en compte au nveau de la personnalisation. Si, nous avons déjà montré comment il était possible, dans la modélisation de niveau CIM, de prendre en compte les informations provenant du profil (cf. chapitre 3, §3.1.2), les préférences de l'utilisateur (cf. chapitre 4, §4.3.2) et l'expérience de l'utilisateur (cf. chapitre 4, §4.3.3), nous n'avons pas encore montré comment il était possible de prendre en compte les informations provenant des modèles sociaux (SM2) dans les autres modèles. C'est ce que nous allons faire dans cette section de notre mémoire. Le langage associé au modèle d'action n'ayant été qu'abordé (cf. chapitre 3, §3.1.2), nous verrons aussi comment ce type de langage peut être utilisé dans le cadre de la modélisation d'une application.

Supposons que dans le cadre d'un processus métier permettant l'impression d'un ticket, on décide d'avoir un comportement différent en fonction du groupe social d'appartenance de l'utilisateur et plus précisément un comportement différent si l'utilisateur est un étudiant.

Au niveau de la modélisation, la première question est de savoir comment, du point de vue conceptuel, identifier un étudiant. Dans notre exemple, la solution retenue est de définir, dans un modèle d'organisation, deux groupes sociaux différents ("Etudiant" et "Non etudiant") dans lequel seront répartis l'ensemble des utilisateurs [2] (cf. Figure 5.36).

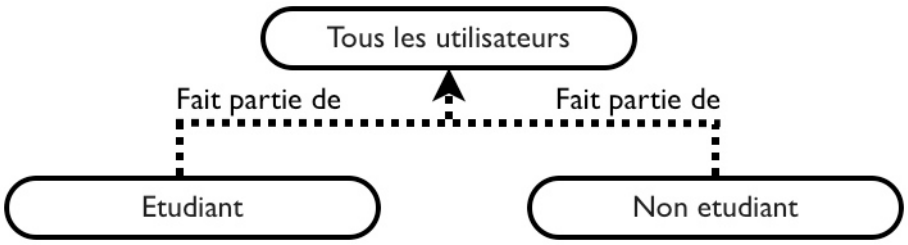


Figure 5.36. Modèle d'organisation SM2 associé à la répartition des utilisateurs dans deux groupes sociaux différents ("Etudiant" et "Non etudiant")

[2] Dans le cadre de l'exemple, il aurait aussi été possible de faire le choix de passer par le modèle de profil utilisateur (SM1) et par la création ou l'utilisation d'une propriété de ce modèle permettant d'identifier l'utilisateur comme étant un étudiant. Ce choix n'a pas été retenu ici afin de permettre de montrer d'autres types de liens entre modèles dans PERCOMOM.

Dans le cadre de notre l'exemple, l'impression d'un ticket sera réalisée volontairement par l'utilisateur à travers l'utilisation d'un élément d'interaction de type *UIFieldAction* "UIFieldAction_ImprimerTicket". Afin de pouvoir faire le lien entre cet élément d'interaction et l'action (modèle BM1) à exécuter, on définira au niveau de l'artefact associé à cet élément d'interaction un lien vers les modèles d'action à l'aide de la propriété "action" ; celle-ci permettant d'indiquer le nom du modèle d'action à exécuter lorsque l'utilisateur activera l'*UIFieldAction*. La Figure 5.37 présente un extrait du modèle statique d'interaction (SM2) correspondant uniquement centré sur l'*UIFieldAction* "UIFieldAction_ImprimerTicket".

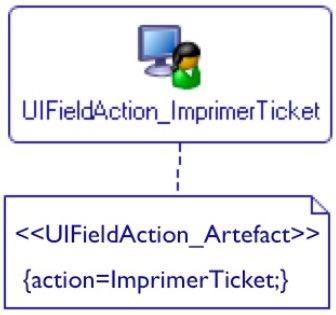


Figure 5.37. Utilisation d'une propriété "action" dans un artefact associé à un UIFieldAction permettant l'impression d'un ticket

Ce lien étant établi, le modèle suivant à définir est le modèle d'action correspondant à "ImprimerTicket". Ce modèle sera composé d'une première tâche "ValiderPaiement" permettant de calculer l'ensemble des éléments communs à tous les tickets comme le montant total et le numéro de référence de la transaction. Une fois cette première tâche réalisée, l'exécution de la tâche suivante sera dépendante du groupe d'appartenance de l'utilisateur. Ainsi, si l'utilisateur appartient à la catégorie "Etudiant" la tâche "TicketEtudiant" sera exécutée. Dans le cas contraire, ce sera la tâche "TicketGénéral" qui sera exécutée. Une fois l'une de ces deux tâches exécutée, le processus métier "ImprimerTicket" sera terminé. La Figure 5.38 présente le modèle d'action (SM2) associé à l'action "ImprimerTicket".

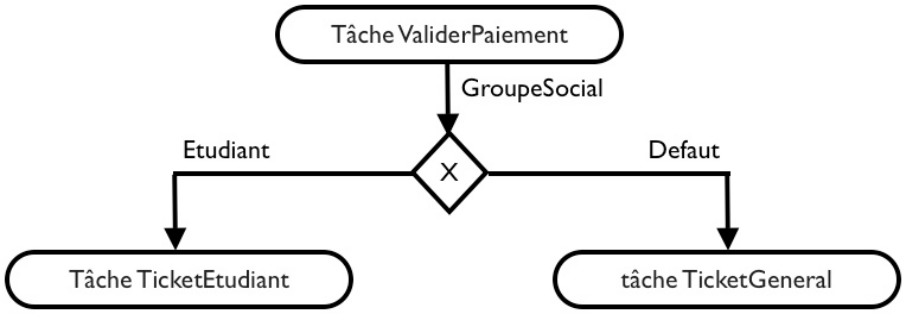


Figure 5.38. Modèle d'action (SM2) associé à l'action "ImprimerTicket" et dont le contenu tient compte du groupe social d'appartenance de l'utilisateur

À titre d'information et afin de faciliter la compréhension de l'exemple, la Figure 5.39 présente deux exemples différents de ticket imprimés en fonction du groupe social d'appartenance de l'utilisateur : "Etudiant" pour le ticket (a) et autre(s) groupe(s) pour le ticket (b).

(a)

(b)

Figure 5.39. Exemple de ticket de paiement personnalisé (a) pour un étudiant, (b) pour un usager qui n'est pas un étudiant

Une fois les tâches identifiées au niveau du modèle d'action, il convient ensuite de définir le modèle d'action (BM1) associé à chaque tâche (le nom de la tâche étant identique au nom du modèle d'action correspondant). De manière pratique, chaque modèle d'action peut être défini à l'aide d'un pseudo langage proche du langage Pascal permettant de décrire une succession d'opérations à réaliser. Ce pseudo langage s'appuie sur un ensemble de librairies de fonctions permettant de manipuler l'ensemble des objets définis dans les autres modèles comme, par exemple, l'ontologie de domaine ou encore le profil utilisateur. Dans le cadre de son exécution, une action peut faire appel à des informations spécifiques à l'environnement d'exécution à travers l'utilisation de variables externes ou à travers l'utilisation de paramètres qui lui ont été fournis lors de son appel. L'objet de ce chapitre n'étant pas de définir dans le détail les modèles d'action, nous terminerons sur le sujet en présentant, à titre d'exemple, le contenu du modèle d'action associé à l'action "TicketEtudiant" qui permet d'imprimer sur un ticket l'ensemble des informations spécifiques à un étudiant ; ces informations étant sensibles au contexte géographique et temporel.

```
{ Le mot-clé Action est utilisé pour définir une nouvelle Action (modèle BM1) }
action TicketEtudiant();
    { Définition des variables pour la gestion de la date et de l'espace géographique }
    var
        dt;datetime;
        gs:geographicalspace;
    begin
        { Initialisation de la date courante }
        setdatetime(dt,currentdatetime);
        { Initialisation de l'espace géographique courant (celui où se trouve l'utilisateur) }
        setgeographicalspace(gs,currentspace);
        { Impression du ticket }
        { Le mot-clé printer permet d'indiquer qu'on va utiliser l'imprimante la plus proche de l'usager }
        { Initialisation de l'imprimante avec le mot-clé open}
        { Le second paramètre permet d'indiquer le type d'imprimante à utiliser }
        { Type 'generic' pour utiliser une imprimante générique (celle imprimant le reçu)}
        { Type 'specific' pour utiliser une imprimante spécifique (celle imprimant le ticket)}
        { Même si l'utilisateur bouge, il utilisera la même imprimante jusqu'à ce que la }
        { fonction close soit utilisée}
        open(printer,generic);
        writeln(printer,CARTE DE CREDIT);
        { le format utilisé pour la date utilise des informations spécifiques à la localisation}
        { de l'utilisateur au moment de l'appel de la fonction }
        writeln(printer,'DATE: ',printformat(dt,uselocal,'dd-mm-yy hh-mm');
```

```
    { la fonction de formatage de l'espace géographique utilise le nom associé à l'espace }
    { géographique le plus petit qui contient l'endroit où se trouve l'utilisateur (provient du modèle EM1)}
    writeln(printer,'LIEU:',printformat(gs,uselocal,'firstconceptualname');
    writeln(printer,'');
    writeln(printer,'REFERENCE:');
    { Utilisation de la variable paymentreference qui a été initialisée dans une autre action }
    { Dans notre exemple, l'initialisation est faite dans l'action ValiderPaiement }
    writeln(printer,variable.paymentreference);
    writeln(printer,'');
    writeln(printer,'TOTAL');
    { Utilisation de la variable totalpaid qui a été initialisée avant dans l'action ValiderPaiement }
    writeln(printer,'        ',variable.totalpaid);
    writeln(printer,'--------------------------------------------------------------------------------');
    writeln(printer,'');
    { Impression de la dernière publicité valide pour les étudiants en tenant compte des }
    { propriétés de personnalisation définies au niveau de l'UIUnit utilisé }
    { Premier paramètre de la recherche : le nom du concept métier dans l'ontologie.}
    { Dans notre exemple, le concept métier manipulé est le concept Message_Publicitaire qui est }
    { associé à la notion de message publicitaire de type textuel }
    { Deuxième paramètre : liste des critères de recherche à utiliser. Dans notre exemple, le critère }
    { utilisé sera la valeur de la propriété categorie de la classe de l'ontologie qui permet d'indiquer }
    { à quelle catégorie socioprofessionnelle est associée le message publicitaire }
    { dans notre cas, ce sera la catégorie Etudiant}
    { Troisième paramètre : liste des critères de tri (chaque critère est associé à une propriété }
    { du concept manipulé ) }
    { Quatrième paramètre : nombre maximal de valeurs retournées par la fonction }
    { L'apparence finale de l'impression est géré automatiquement par la fonction writeln }
    { qui permet d'adapter l'impression à l'imprimante physique en cours d'utilisation }
    writeln(printer,searchconceptvaluewithinheritedpersonalization('Message_Publicitaire','Message_Pu
    blicitaire.categorie==Etudiant,'date desc',1);
    close(printer);
end
```

5.6.2 Prise en compte des interactions verbales entre personnes dans un processus métier (modèle IM1) de niveau CIM

Dans le cadre de la modélisation conceptuelle des applications (niveau CIM), il est souvent très difficile de modéliser l'ensemble des acteurs ainsi que l'ensemble des échanges d'informations pouvant intervenir dans le cadre d'un processus métier. Et, ceci est d'autant plus vrai lorsque les échanges sont difficilement formalisables comme dans le cadre d'échanges verbaux.

Dans le cadre de PERCOMOM, l'utilisation d'une notation dérivée de BPMN pour la définition de l'ensemble des processus métier nous permet de définir de manière claire l'ensemble des intervenants ainsi que l'ensemble des échanges pouvant survenir entre les intervenants (cf. chapitre 1, §1.3.2). Or, pour certains procesus métier, ceux-ci sont très importants pour permettre de modéliser l'enchaînement entre toutes les tâches du procesus métier et donc, dans PERCOMOM, il doit être possible de les définir dans les modèles de processus métier (IM1) de niveau CIM.

À titre d'exemple, la Figure 5.40 présente un processus métier simplifié qui pourrait être associé à une demande d'horaire de train par un usager à un guichetier dans une gare (l'application à la disposition du guichetier est présentée dans la Figure 5.41 dans sa version guichet et dans la Figure 5.42 dans sa version mobile). La version mobile de l'application est destinée à des agents de terrain afin de leur permettre de renseigner les usagers quelle que soit leur localisation dans la gare. Sur ce processus, chaque intervenant est représenté à travers la définition d'ensembles de tâches qui lui sont propres. Afin de simplifier la représentation graphique du processus, l'ensemble des tâches associées au guichetier a été regroupé dans un sous-processus métier *RechercherHoraireDeTrain*

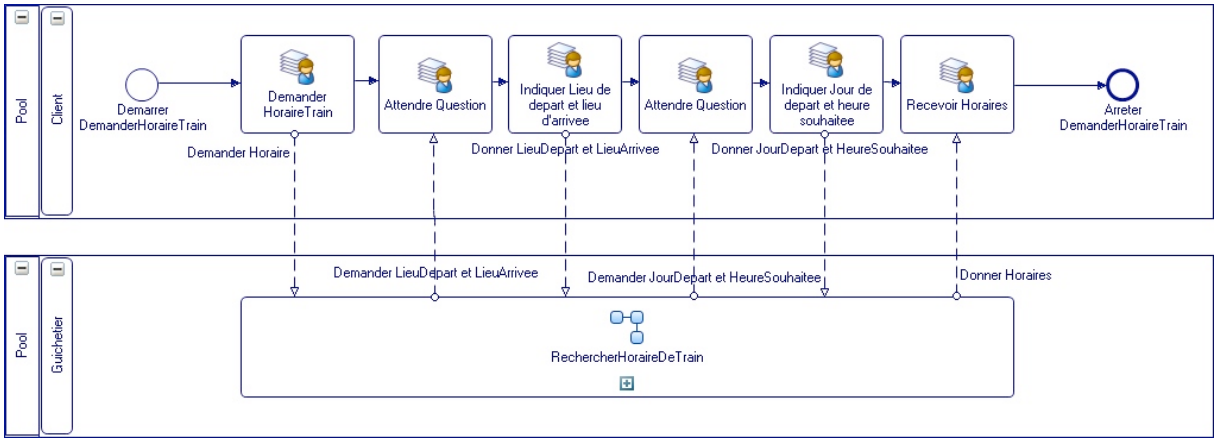


Figure 5.40. Exemple de prise en compte des acteurs et des échanges d'informations verbaux au niveau de la définition des modèles conceptuels d'un processus métier associer à la demande d'horaires de train par un usager à un guichetier dans une gare

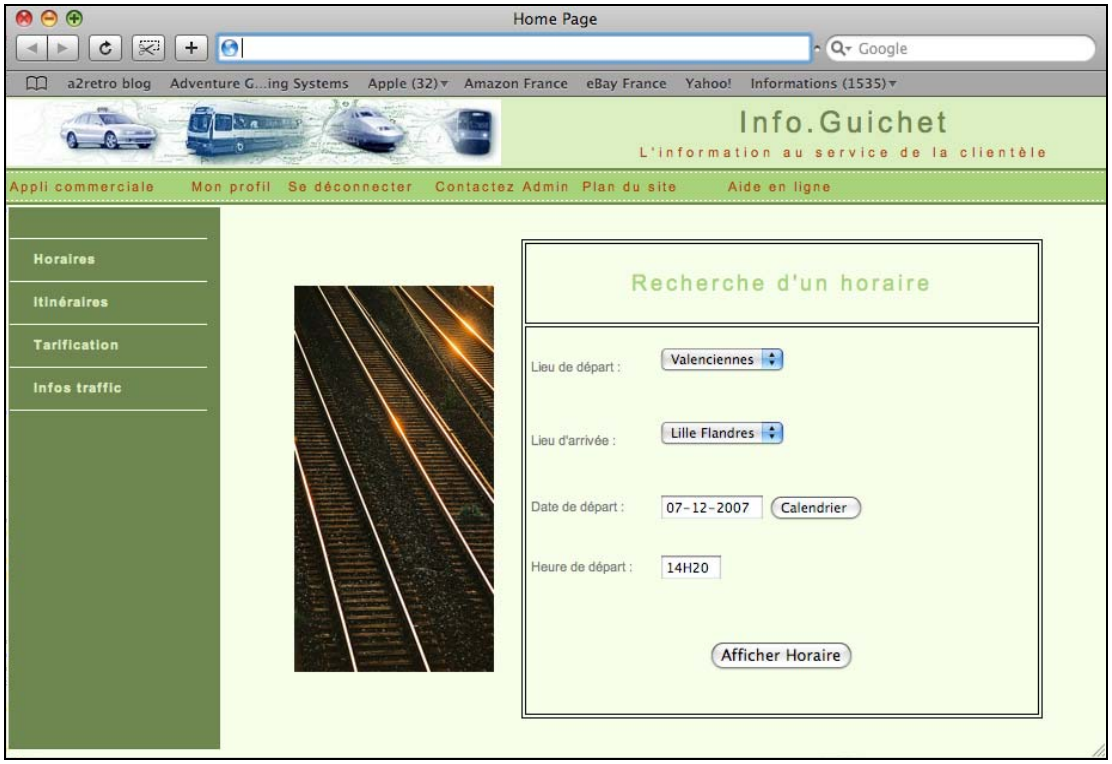


Figure 5.41. Exemple d'application à disposition du guichetier, accessible via un navigateur sur un ordinateur de type PC, associée au modèle métier décrit dans la Figure 5.40

Figure 5.42. Exemple d'application à disposition d'un guichetier (ou d'un agent de terrain), accessible via un navigateur web sur un téléphone portable de type Apple iPhone, associée au modèle métier décrit dans la Figure 5.40

Sur ce processus métier, on pourra remarquer que :

- Les tâches réalisées par l'utilisateur sont des tâches manuelles qui sont identifiées comme telles à l'aide d'un pictogramme spécifique (cf. Figure 5.43 (a)).
- L'ensemble des échanges d'informations entre les deux intervenants sont des échanges verbaux qui sont identifiés comme tel et qui sont tous répertoriés (cf. Figure 5.43 (b)).

(a) (b)

Figure 5.43. Formalisme utilisé pour la prise en compte des interactions verbales dans les processus métier avec le formalisme utilisé pour les tâches manuelles réalisées par l'utilisateur (a) et pour les échanges verbaux réalisés entre deux intervenants (b).

Au niveau de cet exemple, il est important de noter que, dans PERCOMOM, l'utilisation des ensembles d'activités (regroupement des activités par intervenant) n'est pas obligatoire au niveau de la définition de chaque processus métier. En fait, c'est une possibilité de notation qui est proposée, aux personnes en charge de la modélisation, pour faciliter la lecture et la compréhension des modèles métier en répartissant l'ensemble des tâches réalisées au niveau de chaque processus par intervenant (que celui-ci soit un intervenant humain ou pas).

Dans le cadre de PERCOMOM, nous avons considéré que les échanges verbaux et les tâches manuelles faisaient pleinement partie des processus métier même s'ils ne donnent pas forcément lieu à une traduction technique au niveau PIM ou au niveau PSM de notre architecture. Ceci permet de replacer les applications informatiques dans leur contexte et surtout de mieux comprendre leur utilisation et donc de mieux évaluer leur impact au niveau métier.

5.6.3 Conclusion sur les exemples complémentaires

Ces deux exemples complémentaires nous ont permis de vérifier la validité de deux éléments supplémentaires au niveau de l'approche PERCOMOM :

- Le principe de définition et l'utilisation des modèles d'actions dans le cadre d'une personnalisation en fonction du groupe social d'appartenance de l'utilisateur.
- La capacité de prise en compte des échanges verbaux dans les modèles de processus métier (IM1) de niveau CIM.

Ils ont aussi permis de voir, surtout dans le cadre du premier exemple concernant la prise en compte du groupe social d'appartenance de l'utilisateur, que le contenu et l'organisation des modèles de niveau CIM étaient dépendants des choix effectués par les experts métier au niveau de la modélisation. Ainsi, pour une même problématique, il est possible de définir plusieurs solutions de modélisation différentes permettant ainsi d'adapter la modélisation plus aux besoins des experts métier en charge de la modélisation qu'aux besoins purement liés à la problématique traitée.

Conclusion

Dans ce chapitre, nous avons, à travers différentes applications, validé les points suivants dans PERCOMOM :

- La validité de l'approche de modélisation pour la prise en charge d'une application interactive au niveau des différents modèles de niveau CIM ; et ceci uniquement à partir des besoins fonctionnels initiaux définis pour l'application.
- La validité de l'approche de transformation de modèles proposés dans le cadre de PERCOMOM pour passer des modèles de niveau CIM aux applications concrètes.
- La capacité de PERCOMOM à prendre en charge des problématiques liées à l'adaptation des applications interactives au contexte.
- La capacité de PERCOMOM à gérer la notion d'appartenance d'un utilisateur à un groupe social dans les applications.
- La capacité de PERCOMOM à permettre la modélisation des échanges verbaux dans les processus métier associés aux applications interactives.

À travers les différentes applications et exemples présentés, nous avons aussi montré que, dans PERCOMOM, les modèles conceptuels de niveau CIM, s'ils étaient bien séparés et s'ils traitaient chacun d'une problématique particulière, étaient capables d'interagir entre eux, permettant ainsi de prendre en compte de nombreuses problématiques différentes pour répondre aux besoins fonctionnels des applications.

À côté de cela, nous avons aussi montré que les modèles d'interaction créés au niveau CIM étaient bien des modèles conceptuels indépendants des problématiques techniques ; ce qui permet d'envisager d'avoir une véritable approche de type IDM à travers l'utilisation de PERCOMOM.

Cette séparation des modèles et les nombreuses intégrations possibles de modèles entre eux permettent de confier chaque type de modèle à un expert métier de la problématique considérée et d'ainsi optimiser la modélisation. Ceci représente, à notre avis, une avancée importante en matière de modélisation des applications et permet d'envisager, à terme, la création d'approches dirigées par les modèles qui se baseraient sur l'utilisation de modèles graphiques facilement manipulables par des experts métier. Ceci représente une des nombreuses perspectives de recherche que nous allons présenter dans le chapitre suivant.

Chapitre 6
Évaluation globale de PERCOMOM et perspectives de recherche

Sommaire

INTRODUCTION **201**

6.1 EVALUATION DE PERCOMOM **201**
6.1.1 COMPARAISON DE PERCOMOM PAR RAPPORT AUX APPROCHES SIMILAIRES 201
6.1.2 LES CONTRIBUTIONS PRINCIPALES DE PERCOMOM 205

6.2 PERSPECTIVES DE RECHERCHE **206**
6.2.1 OUTILLAGE DE PERCOMOM 206
6.2.2 PERCOMOM : UNE METHODE DE MODELISATION CONCEPTUELLE A FINALISER 206
6.2.3 VERS UNE ARCHITECTURE OUVERTE ET MODULAIRE 207
6.2.4 LES EXPERTS METIER AU CŒUR DE LA MODELISATION DES APPLICATIONS 208
6.2.5 VERS UNE PRISE EN COMPTE GLOBALE DE LA PERSONNALISATION 209
6.2.6 LA PRISE EN COMPTE DU CONTEXTE DANS PERCOMOM : ENTRE PERTINENCE ET "SUPERFLU" 210
6.2.7 VERS UNE PRISE EN COMPTE GLOBALE DE L'UTILISATEUR DANS LE CADRE DE LA PERSONNALISATION DES CONTENUS 210
6.2.8 LES CONTENUS : UN ELEMENT CENTRAL DE LA PERSONNALISATION 212
6.2.9 VERS UNE GENERALISATION DE PERCOMOM 212

CONCLUSION **213**

Introduction

Dans le cadre des différents chapitres présentés dans ce mémoire, nous avons :

- dressé un état de l'art des problématiques dans le domaine de la modélisation des applications interactives et de la personnalisation ;
- présenté notre méthode PERCOMOM de modélisation des applications interactives personnalisées ;
- mis en application PERCOMOM sur des cas concrets d'applications interactives dans le domaine des transports.

Si, beaucoup de points différents ont pu être abordés, il reste à effectuer une synthèse sur PERCOMOM dans sa globalité et aussi à faire le point sur ses manques et ses limites actuelles.

Dans la première partie de ce chapitre, nous allons commencer par montrer, à travers une comparaison de PERCOMOM vis-à-vis d'autres approches existantes, ce qui la différencie de celles-ci. Mais nous montrerons aussi ce qu'elle apporte de plus par rapport aux autres méthodes existantes. Une fois cette comparaison effectuée, nous ferons une synthèse sur les contributions principales de PERCOMOM du point de vue de la recherche dans les domaines de l'ingénierie logicielle, de la modélisation des IHM et de la personnalisation.

PERCOMOM n'existant pour l'instant que dans sa première version, un certain nombre de points restent à étudier dans les années à venir. C'est ce que nous présenterons dans la deuxième partie de ce chapitre à travers différentes perspectives de recherche aussi bien au niveau de la modélisation des applications que de la personnalisation des contenus.

Mais, avant cela, commençons par situer PERCOMOM par rapport autres méthodes existantes.

6.1 Évaluation de PERCOMOM

6.1.1 Comparaison de PERCOMOM par rapport aux approches similaires

Dans l'état actuel d'avancement de nos travaux, au niveau de la modélisation conceptuelle des applications, nous avons principalement mis l'accent sur les modèles conceptuels de modélisation des interactions et plus particulièrement sur le modèle de processus métier (IM1) et le modèle statique d'interaction (IM2).

Pour ces deux types de modèles, PERCOMOM propose un certain nombre d'éléments qui, comme nous le verrons par la suite, se démarquent des approches déjà existantes.

Le nombre d'approches permettant de modéliser une application interactive étant assez important (cf. chapitre 1), nous avons décidé de ne sélectionner que les deux approches les plus pertinentes par rapport aux problématiques traitées par PERCOMOM. Ainsi, dans la suite de cette section, nous allons comparer PERCOMOM, pour les aspects liés directement aux IHM (modèles IM1 et IM2), à :

- UsiXML qui est un langage de définition des interfaces pour les applications interactives et pour lequel il existe des outils et des méthodes de modélisation permettant de prendre en charge de nombreux aspects d'une application.
- UML qui est la notation de référence préconisée actuellement par l'OMG pour modéliser les applications informatiques.

Il est intéressant de noter ici qu'UML, qui se veut être un langage universel de modélisation, est prévu pour être extensible à travers l'utilisation de "profils UML" ; ces derniers étant des extensions au langage de modélisation. Dans le cadre de notre comparaison, n'ayant pas trouvé de profils UML, validés par l'OMG, pour la modélisation des processus métier et des IHM, nous nous sommes restreints à comparer PERCOMOM à la version 2 d'UML.

Pour ce qui est d'UsiXML, nous avons considéré, dans le cadre de notre comparaison, l'ensemble des travaux réalisés disponibles sur le site de référence d'UsiXML (http://www.usixml.org). Par

contre, nous n'avons pas pris en compte les travaux et réflexions en cours, présentés par (Vanderdonckt, 2008), pour lesquels nous ne disposions pas d'informations suffisantes au moment de la rédaction de ce mémoire.

Dans le Tableau 6.1, on trouvera une comparaison entre PERCOMOM, UsiXML et UML du point de vue de la modélisation des processus métier (modèle IM1). À ce titre, il est important de noter l'utilisation de CTT (Concur Task Tree) dans le cadre des modèles de tâches lors d'une modélisation d'une application interactive avec UsiXML.

Tableau 6.1. Comparaison des approches PERCOMOM, CTT (UsiXML) et UML pour la modélisation des processus métier (modèles IM1 de niveau CIM dans PERCOMOM)

	PERCOMOM (modèle IM1)	**CTT (UsiXML)**	**UML**
Possède une notation standardisée pour représenter les interactions	Oui	Oui	Non car passage par un profil utilisateur qui, dans le cas présenté, est spécifique aux applications de type web.
Permet de représenter le type d'interaction effectué	Oui à travers l'utilisation d'un certain nombre de mots-clés prédéfinis au niveau des tâches réalisées (exemple : action se rapportant à un élément d'interaction permettant de démarrer une action).	Non. CTT permet d'indiquer qu'il y a interaction avec l'utilisateur mais ne permet pas d'en préciser le type.	Oui mais uniquement à travers l'utilisation de profils spécifiques.
Permet de différencier les tâches effectuées par le système de celles effectuées par les utilisateurs	Oui à travers l'utilisation des mots-clés et d'un code couleur spécifique pour chaque type d'élément.	Oui à travers la notation CTT.	Oui à l'aide des mots-clés définis dans un profil spécifique.
Facilite la réutilisation des modèles existants pour la création de nouveaux modèles	Oui à travers la définition et l'utilisation de sous-processus métier.	Non. La réutilisation n'a pas été explicitement prévue dans le cadre de la notation CTT (ce qui peut constituer un goulot d'étranglement relatif à la réutilisation).	Oui mais de manière incomplète car la réutilisation des modèles reste limitée au niveau des diagrammes d'interaction.
Permet de représenter tout type d'interaction	Oui en théorie mais pour l'instant la validation reste limitée à quelques applications de type WIMP.	Oui pour les applications de types WIMP.	Dépend de ce qui est défini dans le profil UML.
Possibilité de créer des modèles conceptuels indépendants des aspects techniques	Oui par définition.	Oui par définition.	Non car les modèles restent liés à une approche globale de type objets qui est une approche technique.
Permet de prendre en charge des processus métier.	Oui à travers l'utilisation d'un formalisme qui permet de modéliser les tâches interactives et les tâches non interactives.	Non. CTT ne permet de prendre en charge que les interactions.	Oui à l'aide des mots-clés définis dans un profil spécifique.

De cette comparaison, il est possible d'en déduire trois choses :

- En standard, UML est peu adapté à la prise en charge de modèles de processus métier ou de modèles de tâches associés à la modélisation d'applications interactives (ce n'est possible qu'en utilisant des profils UML).
- UsiXML, à travers l'utilisation de CTT, permet de prendre en charge les modèles de tâches associés aux applications interactives. Par contre, il n'a pas été créé dans une optique de réutilisation des modèles et ne permet pas non plus d'identifier systématiquement le type d'interactions.
- PERCOMOM propose une prise en charge des processus métier du point de vue des applications interactives (modèles IM1) et surtout permet d'envisager une réutilisation des modèles dans le cadre d'une approche de type IDM (Ingénierie Dirigée par les Modèles).

Cette première comparaison sur les modèles de type (IM1), si elle permet de voir qu'UML n'est pas l'outil idéal pour modéliser les tâches et/ou processus métier associés à des applications interactives, elle montre aussi que PERCOMOM et UsiXML, à travers l'utilisation de CTT, sont très proches l'un de l'autre. En fait, une des grandes particularités du modèle de processus métier (IM1) de PERCOMOM par rapport à CTT (et donc UsiXML) réside dans la possibilité d'intégrer des éléments provenant des autres modèles conceptuels. Ainsi, dans PERCOMOM, il est possible, à partir d'un même processus métier, d'avoir plusieurs comportements différents en fonction, par exemple, du résultat de l'évaluation d'une règle métier ou alors en fonction du contexte.

L'autre modèle que nous avons détaillé dans ce mémoire est le modèle statique d'interaction (IM2) qui permet de modéliser les interfaces des applications interactives. Afin d'en montrer les différences par rapport aux solutions déjà existantes, nous l'avons comparé à UsiXML et UML (cf. Tableau 6.2).

Tableau 6.2. Comparaison des approches PERCOMOM, UsiXML et UML au niveau de la modélisation des interfaces interactives (modèle IM2 de niveau CIM dans PERCOMOM)

	PERCOMOM (Modèle IM2)	**UsiXML**	**UML**
Permet de modéliser une interface statique d'interaction au niveau conceptuel	Oui à travers l'utilisation des modèles statiques d'interface.	Oui à travers l'utilisation des modèles abstrait d'interface utilisateur (AUI).	UML ne disposant pas d'une approche complète au niveau de la modélisation des IHM, aucune comparaison n'est vraiment possible aussi bien avec UsiXML qu'avec PERCOMOM.
Permet de prendre en compte les problématiques d'interaction dans les applications d'information voyageur dans les transports (applications de type WIMP)	Oui, en théorie, à travers l'utilisation de la notion de *UIElement* qui est par définition extensible.	Non, pas pour l'instant car, à notre connaissance, UsiXML est encore très centré sur les applications classiques WIMP disposant d'un nombre réduit d'éléments d'interaction.	
Facilite la réutilisation	Oui car tous les éléments d'interactions sont prévus pour être réutilisables	Oui car tous les éléments sont prévus pour être réutilisables.	
Extensibilité des modèles.	Oui à travers la possibilité de rajouter de nouveaux éléments et surtout la possibilité de définir de nouvelles propriétés pour chaque type d'*UIElement.*	Oui.	
Lien avec les concepts métier (abstraction par rapport aux données manipulées dans les interfaces)	Lien effectué à travers l'utilisation des propriétés qui permet d'effectuer des liens vers l'ontologie de domaine.	UsiXML n'utilisant pas la notion de concept métier pour la manipulation des informations, ce lien n'existe pas. Seul un lien direct avec les données est possible.	

	PERCOMOM (Modèle IM2)	**UsiXML**	**UML**
Possède des outils pour créer des modèles d'interfaces interactives	Ne possède, pour l'instant, qu'un outillage limité.	Possède de nombreux outils adaptés à la prise en charge de plusieurs types de problématiques différents (voir les outils disponibles sur le site http://www.usixml.org)	
Mâturité et diffusion de l'outil	Outil expérimental non encore diffusable.	Le langage est en cours de normalisation auprès de l'OMG et de nombreux travaux de recherche sont actuellement effectués autour d'UsiXML	Outil standardisé par l'OMG et outil de référence pour la modélisation des applications développées avec un langage objets.

De cette comparaison, il est possible d'en déduire trois choses :

- UML, en standard, n'est pas adapté à la modélisation des interfaces d'une application interactive.
- UsiXML, est un langage qui permet de définir complètement les interfaces des applications interactives. Par contre, il ne permet aujourd'hui de prendre en compte que les interfaces de type WIMP (Window, Icon, Mouse, Pointing device). De plus, il ne permet de faire que des liens directs vers les modèles de données sans passer par une abstraction de niveau supérieur ; ce qui ne permet pas, aujourd'hui, d'en faire un véritable modèle conceptuel de niveau CIM.
- PERCOMOM propose quasiment les mêmes choses qu'UsiXML au niveau de la modélisation des interfaces interactives en permettant en plus d'accéder aux données à travers la notion de concept métier qui permet une véritable abstraction par rapport aux données manipulées dans l'interface.

Cette deuxième comparaison sur les modèles statiques d'interaction (IM2), nous permet de confirmer qu'UML n'est pas la notation la plus adaptée pour modéliser une application interactive surtout au niveau conceptuel. Elle nous permet aussi de voir qu'UsiXML et PERCOMOM sont très proches l'une de l'autre. Néanmoins, PERCOMOM présente, à notre avis, un certain nombre d'avantages complémentaires en-dehors du fait qu'elle permet d'avoir une abstraction plus grande au niveau conceptuel qu'UsiXML. Ainsi, PERCOMOM (cf. chapitre 3, §3.1.3), à travers l'utilisation d'artefacts spécifiques pour chaque élément d'interaction (les *UIElement*) et l'utilisation de règles métier ou de domaine de validité, permet de définir, pour chaque élément d'interaction, des propriétés sensibles au contexte et à l'utilisateur.

À côté de cela, une des grandes différences de PERCOMOM par rapport à UsiXML réside dans sa capacité de prendre en compte la notion de personnalisation ; ce que n'est pas capable aujourd'hui de faire UsiXML. D'autre part, PERCOMOM a été conçu dans une optique de forte réutilisabilité des modèles d'IHM ; chaque modèle pouvant servir à plusieurs applications différentes et, si nécessaire, avec des comportements différents à travers l'utilisation de restrictions applicatives. UsiXML, pour sa part, n'a été conçu que pour prendre en charge une seule application à la fois et, si les modèles sont réutilisables, ceux-ci ne sont, à notre connaissance, pas sensibles au contexte applicatif.

Enfin, PERCOMOM, contrairement à UsiXML, n'a pas été conçue que pour prendre en charge la modélisation des interfaces des applications interactives mais a aussi été conçue pour prendre en charge les applications interactives dans leur globalité ; ce qui explique l'existence des modèles sociaux (SM1, SM2 et SM3), des modèles environnementaux (EM1, EM2 et EM3) et des modèles comportementaux (BM1, BM2, BM3) au niveau CIM ; autour desquels subsistent encore de nombreuses perspectives de recherche (cf. dans ce chapitre, §6.2.5 à 6.2.8).

UsiXML étant plus avancé que PERCOMOM sur un certain nombre de points comme l'outillage, la génération semi-automatique d'applications interactives ou encore sur la validation des modèles

créés et étant donné que les modèles statiques d'interaction (IM2) sont très proches des modèles d'IHM créés par UsiXML, nous pensons qu'une synthèse des deux approches serait très intéressante à court ou moyen terme afin de pouvoir offrir un kit complet de modélisation, comprenant les outils et la méthode, pour les experts métier.

Cette comparaison rapide des deux modèles principaux d'interaction de PERCOMOM (IM1 et IM2) étant réalisée, il nous est maintenant plus aisé de montrer les contributions principales de PERCOMOM dans le domaine de la recherche.

6.1.2 Les contributions principales de PERCOMOM

Si PERCOMOM permet de prendre en charge de nombreuses problématiques différentes et complémentaires, elle nous permet, dans l'état actuel de nos travaux, d'apporter un certain nombre de contributions dans les domaines du génie logiciel, de la modélisation des IHM et de la personnalisation, dont les principales sont données ci-dessous :

- PERCOMOM est, à notre connaissance, la première méthode de modélisation des applications interactives qui permettent d'envisager de modéliser les différents aspects de ce type d'application au niveau CIM, et ceci à travers l'utilisation d'un ensemble de 14 modèles (3 modèles sociaux, 3 modèles environnementaux, 3 modèles comportementaux, 3 modèles d'interaction, 1 modèle d'application et une ontologie de domaine) (cf. chapitre 3, §3.1).
- PERCOMOM est, à notre connaissance, la première méthode de modélisation capable de prendre en charge, pour l'instant de manière limitée, la personnalisation des contenus au niveau de la modélisation conceptuelle des applications interactives (cf. chapitre 4).
- PERCOMOM propose une ébauche de solution architecturale innovante pour la génération semi-automatique d'applications à partir de modèles conceptuels (cf. chapitre 3, §3.2 et §3.3).
- À terme, PERCOMOM devrait pouvoir être utilisée par des experts métier et pas uniquement par des informaticiens (cf. chapitre 3, §3.1) ; ce qui devrait faciliter la création d'applications informatiques au sein des entreprises.
- PERCOMOM est, à notre connaissance, la première méthode de modélisation des applications interactives personnalisées à proposer l'utilisation de la notion d'ontologie au niveau des contenus manipulées dans les IHM permettant ainsi d'avoir un niveau d'abstraction supplémentaire par rapport aux données manipulées (cf. chapitre 3, §3.1.2). Même si cette utilisation n'est que très partielle aujourd'hui, elle est néanmoins très prometteuse.
- PERCOMOM, à travers l'utilisation d'un moteur de règles métier et la notion de domaine de validité, propose une solution intéressante pour la prise en charge du contexte au niveau des modèles conceptuels (cf. chapitre 3, §3.1.2, et annexe A).
- PERCOMOM, à travers son utilisation dans le cadre du projet ANR Viatic.Mobilité, est la première méthode de modélisation à introduire la notion de vécu des déplacements dans le cadre de la modélisation des applications interactives personnalisées (cf. chapitre 5, §5.2) ; même si aujourd'hui la solution proposée ne permet de résoudre cette problématique que très partiellement.

Si PERCOMOM présente, en théorie et d'après nos premières validations, un certain nombre d'avancées en matière de modélisation conceptuelle des applications interactives personnalisées, elle présente aussi, dans son état actuel, un certain nombre de limites et de points d'amélioration que nous allons maintenant présenter à travers nos principales perspectives de recherche.

6.2 Perspectives de recherche

6.2.1 Outillage de PERCOMOM

Dans le cadre de nos travaux de recherche, nous n'avons pas encore eu le temps de finaliser l'ensemble des outils nécessaires à la modélisation et à la génération semi-automatique des applications et seuls quelques éléments existent aujourd'hui (outil de création des modèles métier (IM1) et des modèles d'interaction (IM2) (cf. Annexe C), frameworks techniques de niveau PIM et de niveau PSM ou encore quelques services techniques de niveau PIM comme, par exemple le service de personnalisation des contenus).

Dans le cadre d'une validation à plus large échelle de PERCOMOM, la création de ces outils est indispensable pour permettre (cf. chapitre 3) :

- la prise en charge de l'ensemble des modèles de niveau CIM ;
- la prise en charge des liens entre les différents modèles de niveau CIM afin de faciliter l'appel d'un modèle à partir d'un autre modèle ;
- la gestion d'une bibliothèque de modèles conceptuels permettant de faciliter la réutilisation des modèles ;
- la validation des modèles conceptuels au niveau CIM à travers l'utilisation des réseaux de Petri, pour les modèles d'interaction IM1 (processus métier) et IM3 (modèle dynamique d'interaction), et à travers l'utilisation d'autres outils et méthodes, qui restent à déterminer, pour les autres modèles ;
- la création et l'intégration de services fonctionnels et techniques au niveau PIM et faciliter leur utilisation au nivau PSM ;
- une génération semi-automatique des applications du niveau CIM au niveau PSM ;
- une gestion des règles au niveau PIM ;
- une gestion des informations de présentation et des règles au niveau PSM ;

Sans ces outils, PERCOMOM restera une méthode de modélisation plus théorique que pratique ne permettant pas ainsi d'en valider complètement la pertinence et surtout ne permettant pas d'en mesurer complètement les avantages et les inconvénients par rapport aux autres méthodes existantes (cf. chapitre 1).

L'outillage ne servant que de support à la méthode PERCOMOM, celui-ci ne pourra être efficace que si la méthode est définie avec précision et rigueur.

6.2.2 PERCOMOM : une méthode de modélisation conceptuelle à finaliser

Comme nous l'avons précisé dans le chapitre 3, §3.1.1, PERCOMOM n'est pas aujourd'hui une véritable méthode de modélisation dans le sens ou seuls les grands principes de la méthode ont été définis. Ceci représente un axe de recherche important car, pour que PERCOMOM puisse être réellement validée, il faut qu'elle s'appuie sur une véritable méthode de modélisation conceptuelle des applications interactives.

Pour répondre à cette problématique, il faudra principalement travailler autour de quatre axes :

- la définition complète de ce qu'est une application interactive du point de vue conceptuel et la création des modèles CIM qui lui sont associés ;
- la définition des liens et des interactions entre les différents modèles de niveau CIM, du point de vue conceptuel ;
- la définition d'une méthode permettant de passer des besoins fonctionnels aux modèles conceptuels ;

- la définition d'une méthode permettant de répartir les différents modèles entre les différents experts métier en charge de la modélisation de l'application.

Une fois finalisée, l'approche PERCOMOM devra être comparée de nouveau aux autres approches de modélisation existantes afin d'analyser en détail ses avantages et ses inconvénients. Elle devra aussi faire l'objet d'une évaluation par rapport à sa capacité à prendre en charge les différentes approches de création d'applications que sont, par exemple, les approches classiques de type cycle en V (présentée dans le chapitre 4, §4.1.1) et les approches de type itératives ou agiles (Vickoff, 2005). Dans ce dernier type d'approche, les applications sont créées par itérations successives au niveau de leurs fonctionnalités conduisant à des mises en production successives.

Enfin, elle devra être complétée par les outils et architecture permettant de la prendre en charge.

6.2.3 Vers une architecture ouverte et modulaire

Une des caractéristiques de PERCOMOM, dans le cadre de la génération semi-automatique d'applications interactives personnalisées, est de se baser sur un ensemble de frameworks aussi bien au niveau PIM qu'au niveau PSM (cf. chapitre 3, §3.2).

Du point de vue technique, ces frameworks sont constitués d'un certain nombre de composants spécialisés capables d'interagir entres eux mais surtout ne prenant en charge, chacun, qu'un seul type de problématique technique. Dans le cadre de nos travaux, ceci nous a amené à nous poser de nombreuses questions pour lesquelles nous ne disposons aujourd'hui que de réponses partielles :

- Quelles sont les caractéristiques qui permettent de définir un composant ?
- Quels sont les différents types de composants ?
- Quelles sont les normes et standards à définir pour permettre la création de composants ?
- Quelles sont les normes et standards à définir pour permettre la communication entre composants ?
- Comment pouvoir s'assurer de l'unicité de chaque composant ?
- Comment faciliter la gestion de l'ensemble des composants ?

Une fois que nous aurons des réponses à ces questions, il sera possible d'envisager la création de librairies de composants permettant de répondre à un certain nombre de problématiques particulières. Ceci permettra d'envisager la création de ces librairies de composants par des sociétés tierces spécialisées dans la prise en charge de problématiques bien spécifiques ou de plateformes techniques particulières.

L'utilisation de composants pose néanmoins un certain nombre de questions sur la validation de leur contenu technique et de leur contenu fonctionnel. Pour répondre à ces questions, il faudra créer les méthodes et les outils permettant de pouvoir assurer un certain niveau de qualité pour chaque composant créé. De même, les composants étant ouverts sur l'extérieur, il faudra étudier comment pouvoir assurer que l'association de différents composants entres eux n'entraîne pas des comportements indésirables au niveau de l'application.

Au niveau PSM, chaque famille de plateformes d'interaction et chaque plateforme spécifique disposant de son propre framework, la question qui se pose est de savoir comment pouvoir mutualiser certains composants entre les différents frameworks (certains composants pouvant être identiques sur plusieurs plateformes différentes).

Enfin, l'utilisation de composants au niveau des frameworks pose la question de la gestion de l'évolutivité de ces composants dans le temps. Ainsi, comment, lors d'une modification du contenu d'un composant, suite à une évolution des technologies par exemple, pouvoir s'assurer que ce composant fonctionne toujours de la même façon et surtout qu'il se comporte toujours de manière identique avec les autres composants.

Une méthode n'étant conçue que pour faciliter et améliorer le travail des personnes qui l'utilise, nous allons maintenant voir que celles-ci ont aussi un impact sur la façon dont la méthode est conçue.

6.2.4 Les experts métier au cœur de la modélisation des applications

Un des grands principes mis en avant par PERCOMOM, au niveau de la modélisation conceptuelle des applications, réside dans la possibilité, pour les experts métier, de pouvoir modéliser eux-mêmes les applications (cf. chapitre 3).

Dans le cadre de nos travaux de recherche, nous n'avons pu valider ce point qu'auprès d'un nombre limité d'experts métier ; ce qui ne nous a pas permis, pour l'instant, de tirer de conclusion même si les premiers retours sont plutôt positifs. Dans un proche avenir, nous avons comme objectif de réaliser une étude auprès d'un nombre significatif d'experts métier, dans le domaine des transports, afin de valider :

- la facilité de compréhension de la méthode ;
- la facilité de compréhension et de prise en main de chaque type de modèle conceptuel ;
- la pertinence et le niveau d'abstraction de chaque type de modèle conceptuel ;
- la capacité des modèles proposés à prendre en charge la majorité des problématiques fonctionnelles posées dans le cadre de la modélisation des applications interactives personnalisées dans le domaine de la diffusion d'informations à destination des usagers des transports en commun ;
- les interactions nécessaires et suffisantes entre chaque type de modèle ;
- le principe de répartition des modèles conceptuels entre les différents types d'experts métier ;
- la pertinence de l'utilisation des services fonctionnels de niveau PIM du point de vue des experts métier ;
- l'ensemble des services fonctionnels nécessaires et suffisants pour permettre la modélisation de la majorité des applications interactives personnalisées dans le domaine de la diffusion d'informations à destination des usagers.

Cette étude entraînera probablement un certain nombre de modifications à apporter à PERCOMOM qu'il conviendra par la suite de valider dans le cadre d'une méthode d'amélioration continue de PERCOMOM (cf. Figure 6.1). Du point de vue pratique, cette méthode d'amélioration continue se découpera en quatre phases :

- Phase 1 : on planifie l'ensemble des modifications à effectuer dans PERCOMOM au niveau de la modélisation de niveau CIM pour que celle-ci soit utilisable par les experts métier.
- Phase 2 : on intégre les modifications dans PERCOMOM.
- Phase 3 : on met PERCOMOM en test auprès des experts métier et on observe leurs réactions tout en prenant note de leurs commentaires.
- Phase 4 : on analyse les résultats de la phase 3 afin de définir les éléments qu'il faut modifier, compléter, créer ou alors supprimer. On repart à l'étape 1 pour un nouveau cycle.

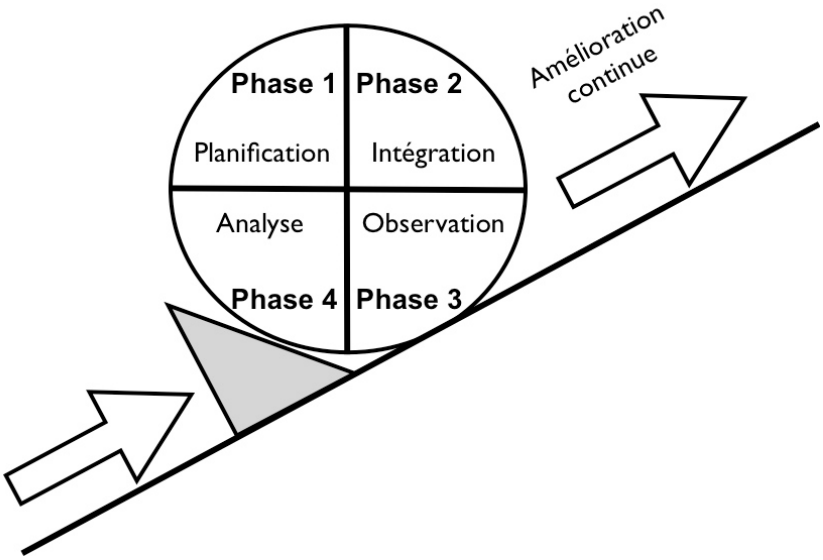


Figure 6.1. Méthode d'amélioration continue appliquée PERCOMOM

Dans le cadre de nos travaux, nous avons vu que, dans le domaine des transports, les applications interactives avaient un fort besoin de personnalisation des contenus que nous n'avons fait qu'aborder.

6.2.5 Vers une prise en compte globale de la personnalisation

Dans le cadre de nos travaux (cf. chapitre 4), nous nous sommes principalement focalisés sur la personnalisation des contenus et pas sur la personnalisation des interfaces même si certaines possibilités sont offertes par PERCOMOM pour rendre les applications interactives adaptatives ; c'est-à-dire capable de s'adapter à l'utilisateur.

Par contre, nous n'avons pas abordé la problématique de l'adaptabilité des interfaces ; c'est-à-dire de la possibilité pour les utilisateurs d'adapter eux-mêmes les interfaces en fonction de leurs besoins propres. Ceci représente un axe de recherche important au niveau de PERCOMOM si on veut la rendre capable de prendre en charge tous les types d'applications interactives personnalisées.

À côté de cela, nous n'avons considéré, au niveau des éléments pouvant avoir une influence sur la personnalisation que les éléments associés au contexte et à l'utilisateur. La question qui se pose à partir de là est de savoir si ce sont les seuls éléments à prendre en compte dans le cadre d'une personnalisation des applications ou s'il faut prendre d'autres éléments en compte. Et, si oui comment les prendre en compte ?

Dans PERCOMOM, l'utilisation de la personnalisation au niveau conceptuel passe par l'utilisation d'un service fonctionnel de niveau PIM et par des adaptations directement prises en compte au niveau des modèles CIM à travers, par exemple, l'utilisation de domaines de validité. La question qui se pose ici est de savoir si ces deux possibilités de prise en compte de la personnalisation sont pertinentes du point de vue de la modélisation et surtout si elles ne risquent pas de se recouvrir sur certains aspects. Pour cela, il faudra définir de manière stricte, en matière de personnalisation, ce qui doit être pris en charge au niveau des modèles de niveau CIM et ce qui doit être pris en charge à travers le service de personnalisation de niveau PIM.

Enfin, nous avons indiqué dans les chapitres 4 que la séparation entre le service fonctionnel de niveau PIM utilisé au niveau CIM et les méthodes et outils utilisés pour effectuer de manière concrète la personnalisation permettait de s'affranchir de ces derniers. Si ceci présente un avantage indéniable, cela présente aussi un inconvénient majeur dans le sens où, du point de vue fonctionnel, il n'est pas sûr qu'on puisse garder une cohérence applicative si certaines propriétés du service fonctionnel de personnalisation de niveau PIM, définies au niveau CIM, ne sont pas prises en compte par le composant associé à ce service de personnalisation au niveau PIM.

6.2.6 La prise en compte du contexte dans PERCOMOM : entre pertinence et "superflu"

Dans le cadre de la personnalisation des applications interactives, nos travaux nous ont permis de définir un certain nombre de problématiques au niveau de la prise en compte du contexte (cf. chapitre 4, §4.2) dont les principales sont :

- Comment définir un contexte ?
- Comment caractériser un élément contextuel ?
- Comment qualifier un élément contextuel du point de vue applicatif (quelles sont les valeurs possibles pour cet élément, comment passe-t-on d'une valeur à une autre, etc.) ?
- Comment prendre en compte le contexte (capteurs, senseurs, etc.) ?
- Quels sont les éléments contextuels pertinents à prendre en compte pour les applications interactives personnalisées ?
- Comment pouvoir s'assurer de l'intégrité fonctionnelle d'un processus métier en cours d'exécution lors de la prise en compte d'un élément contextuel ?
- Un élément contextuel est-il toujours pertinent en fonction, par exemple, de l'endroit où se trouve l'utilisateur, de ce qu'il fait, etc. ? En bref, la pertinence de la prise en compte d'un élément contextuel n'est-elle pas liée elle-même au contexte et à l'utilisateur ?

Une fois que nous aurons des réponses à ces questions se posera le problème de la gestion simultanée de plusieurs éléments contextuels dans le cadre d'une application interactive :

- Est-il pertinent de prendre en compte plusieurs éléments contextuels en simultané ?
- Faut-il gérer des priorités entre les différents éléments contextuels ?
- Si oui, comment fixer les priorités entre les différents éléments contextuels ?
- Comment définir les priorités entre la réalisation d'une tâche fonctionnelle et la prise en compte d'un élément contextuel au niveau d'une application ?

Les réponses à ces questions nécessitant de travailler sur de nombreux domaines de recherche différents, il serait intéressant de créer une équipe pluridisciplinaire pour permettre d'apporter la complémentarité nécessaire à la prise en compte de la problématique posée par le contexte dans le cadre de la personnalisation des contenus.

Un autre élément important pour la personnalisation des contenus est la prise en compte de l'utilisateur que nous n'avons fait qu'aborder dans le cadre de nos travaux.

6.2.7 Vers une prise en compte globale de l'utilisateur dans le cadre de la personnalisation des contenus

Ainsi, au niveau de la prise en compte de l'utilisateur nous n'avons considéré, dans le cadre de nos travaux sur la partie concernant la personnalisation des contenus (cf. chapitre 4, §4.3), qu'un nombre limité d'éléments :

- le profil de l'utilisateur ;
- les préférences de l'utilisateur ;
- l'expérience de l'utilisateur vis-à-vis de l'application.

Pour chacun de ces éléments, PERCOMOM propose une solution de prise en charge limitée de chaque problématique en se basant sur le fait que, pour un utilisateur, la majorité des informations associées à ces éléments seront fournies par les services fonctionnels de niveau PIM (pour la gestion de l'expérience de l'utilisateur) ou par l'utilisateur lui-même pour les informations contenues dans le profil de l'utilisateur.

Dans le cadre d'une prise en compte plus large de la personnalisation, il conviendra d'étudier en détail l'incidence du recueil des données au niveau des différents éléments (profil, préférences et expérience) sur la personnalisation elle-même. De même, il faudra étudier les interactions entre ces différents éléments afin de voir comment les associer et les utiliser au mieux dans le cadre d'une personnalisation des contenus dans une application. Enfin, il faudra regarder si d'autres éléments, directement rattachés à l'utilisateur, ne devraient pas être aussi pris en compte au niveau de la modélisation conceptuelle d'une application interactive personnalisée.

Dans le chapitre 5 (cf §5.2), nous avons aussi vu que, pendant son déplacement, l'usager des transports en commun pouvait avoir différents comportements et différentes attentes en fonction du vécu de son déplacement. Si ceci est assez simple à percevoir, à travers des expériences que nous avons pu tous vivre en tant qu'usager des transports en commun, l'utilisation de la notion de vécu des déplacements dans les applications informatiques introduit de nombreuses questions. La première question qui reste aujourd'hui en suspens réside dans la définition même des catégories de vécu des usagers.

Ainsi, dans le cadre du projet ANR Viatic.Mobilité, le résultat des recherches effectuées, par des anthropologues et des ethnologues, a permis d'identifier quatre grandes catégories de vécu des déplacements. Si cette catégorisation a le mérite d'être simple, aucune étude n'a été menée pour garantir qu'un usager des transports en commun se trouve systématiquement dans une des quatre catégories identifiées pendant ses déplacements. Pour cela, il conviendrait de mener des études plus approfondies sur la caractérisation du comportement des usagers pendant leurs déplacements afin de savoir s'il n'existe pas d'autres catégories et surtout afin de bien identifier l'ensemble des critères permettant de caractériser l'appartenance ou pas d'un usager à une catégorie plutôt qu'à une autre.

Ensuite se pose la question de savoir comment être capable de prendre en compte, au sein d'une application, les changements de catégorie de vécu des déplacements en "temps réel" et surtout sans erreur. Pour cela, il conviendrait :

- de faire des recherches sur la définition et l'utilisation de capteurs pour déterminer, en "temps réel" les changements de comportement des usagers en fonction de leurs gestes, de leurs postures ou encore de leurs expressions faciales ;
- d'étudier comment il est possible de déterminer le changement de catégorie de vécu des déplacements à partir des actions de l'utilisateur au niveau de l'application et aussi en fonction de ce qu'on connaît des habitudes de l'utilisateur ;
- de déterminer l'ensemble des critères permettant de savoir à quel moment il est pertinent d'effectuer un changement de catégorie de vécu des déplacements et à quel moment ce n'est pas pertinent.

Enfin, se pose la question de savoir comment généraliser cette notion de vécu des usagers à d'autres domaines métier et à d'autres contextes. Ainsi, on pourrait imaginer la prise en compte d'une notion de vécu des utilisateurs quand ils utilisent des applications domotiques ou alors quand ils sont dans leur voiture. Mais alors se pose la question de savoir comment passer d'une catégorisation de vécu des utilisateurs à une autre lorsque l'utilisateur change de domaine métier comme, par exemple, lorsqu'il termine un déplacement pour effectuer un autre type de tâche (travail, loisir, etc.).

Si tout ceci ouvre de nombreuses perspectives au niveau de l'adaptation des applications informatiques aux besoins et aux attentes des utilisateurs, apporter des réponses pertinentes à la problématique de prise en compte des vécus des utilisateurs nécessitera un décloisonnement des domaines de recherche et un travail en équipe pruridisciplinaire et c'est probablement là le principal défi.

Pour qu'il puisse y avoir une personnalisation des contenus, il faut aussi qu'on dispose de contenus. Or, comme nous avons pu le voir dans nos travaux (cf. chapitre 2, §2.4, et chapitre 4, §4.4), ceux-ci sont loin d'être simples à prendre en compte.

6.2.8 Les contenus : un élément central de la personnalisation

Dans PERCOMOM, nous avons volontairement limité la prise en compte de la notion des contenus à deux domaines :

- les informations affichées à l'utilisateur ;
- les processus métier proposés à chaque utilisateur (chaque processus étant vu comme étant un contenu applicatif).

Si la notion de contenu est assez simple à gérer dans le cadre de la création d'une application ne prenant pas en compte la notion de personnalisation telle qu'elle a été présentée dans le chapitre 4, elle devient nettement plus complexe une fois celle-ci intégrée. En fait, la vraie problématique réside dans la question de savoir comment intégrer les contenus dans la personnalisation ou plutôt comment intégrer la notion de personnalisation dans les contenus.

Si on se limite à la personnalisation telle qu'elle a été présentée dans le cadre de PERCOMOM (cf. chapitre 4), la prise en compte de la personnalisation au niveau des contenus consiste à rendre ceux-ci sensibles à l'utilisateur et sensibles au contexte. Les questions qui se posent alors, et pour lesquelles nous n'avons pas encore de réponses, sont les suivantes :

- Qu'est-ce que la notion d'utilisateur du point de vue des contenus ?
- Comment rendre un contenu sensible à l'utilisateur ?
- Comment gérer la notion de sensibilité à l'utilisateur des contenus au niveau de la modélisation conceptuelle des applications ?
- Qu'est-ce que la notion de contexte du point de vue des contenus ?
- Comment rendre un contenu sensible au contexte ?
- Comment gérer la notion de sensibilité au contexte des contenus au niveau de la modélisation conceptuelle des applications ?
- Comment gérer, en simultané, une sensibilité à l'utilisateur et une sensibilité au contexte ?
- Est-il toujours pertinent de rendre les contenus sensibles à l'utilisateur et au contexte ?

Au niveau des processus métier, la question qui se pose lorsqu'on parle prise en compte du contexte est de comment garantir le bon fonctionnement, du point de vue métier, du processus métier. Par exemple, si on utilise des domaines de validité dans un processus métier pour permettre l'exécution de telle ou telle tâche ou de tel ou tel sous-processus, comment vérifier que le processus métier ainsi modifié permet bien à l'utilisateur d'atteindre le but métier associé au processus métier ? Une des solutions pourrait être de créer des outils de validation des modèles conceptuels capables de prendre en compte la notion de contexte et donc capable de simuler celui-ci. Mais ceci assurera-t-il l'exhaustivité de la prise en compte du contexte ? Et comment prendre en charge la notion de continuité fonctionnelle dans le cadre de cette validation ?

Pour l'instant, nous n'avons travaillé que sur des contenus relatifs au domaine métier du transport et plus précisement à la problématique de la diffusion d'informations transport aux voyageur ; ce qui crée une certaine limitation sur les possibilités de généralisation de PERCOMOM.

6.2.9 Vers une généralisation de PERCOMOM

Dans le cadre de nos travaux de recherche, nous avons capitalisé sur l'expérience de notre laboratoire de recherche (Petit-Rozé, 2003) (Anli, 2006) et sur notre participation au projet ANR Viatic.Mobilité (cf. chapitre 5, §5.1) pour créer une méthode capable de prendre en charge les applications interactives personnalisées dans le cadre de la diffusion d'informations aux usagers des transports en commun. Si ceci nous a permis de valider notre méthode sur de nombreux points, elle ne nous a pas permis d'en valider la pertinence dans un cadre plus général.

Ainsi, il ne nous est, pour l'instant, pas possible de dire si PERCOMOM est généralisable à d'autres domaines métier ou pas. Et, si oui, quelles sont les modifications éventuelles qu'il faudra lui apporter pour être généralisable.

La première problématique qui devra être traitée dans le cadre d'une généralisation de PERCOMOM sera celle de la pertinence des différents modèles de niveau CIM dans chaque domaine métier. Ainsi, il est possible que certains modèles ne soient plus adaptés pour un autre domaine métier et que d'autres doivent être conçus spécifiquement pour un domaine métier bien particulier.

À côté de cela se pose une autre problématique importante : la définition et l'utilisation des domaines métier. Ainsi, si pour l'instant nous avons défini quelques ontologies simples pour valider PERCOMOM et pour réaliser les différentes applications de validation, il nous reste à définir une véritable ontologie, dans le domaine du transport, capable de prendre en charge l'ensemble des problématiques métier. Pour cela, nous devrons travailler avec des experts métier du domaine et des spécialistes des ontologies afin de limiter les risques de ne créer une ontologie qui ne serait qu'une vue tronquée du domaine et surtout afin de pouvoir s'assurer de son exhaustivité.

La généralisation de PERCOMOM à d'autres domaines métier pose aussi la question de la création d'autres ontologies de domaine mais aussi et surtout la problématique des liens entre les différentes ontologies :

- Comment passer d'une ontologie de domaine à une autre ?
- Comment créer des applications utilisant plusieurs ontologies de domaines différentes ?
- Est-il possible de créer une ontologie "universelle" qui serait capable de couvrir l'ensemble des domaines métier ?

Pour terminer sur les ontologies, la question se pose aussi de savoir si l'utilisation d'ontologies est vraiment pertinente dans tous les domaines métier ou si elle n'est envisageable que dans quelques domaines métier bien particulier.

Enfin, le dernier axe de généralisation de PERCOMOM réside dans sa capacité à prendre en charge d'autres problématiques que les applications interactives de type WIMP. Ainsi, il serait intéressant d'étudier comment généraliser PERCOMOM afin qu'elle puisse prendre en charge d'autres types d'interactions et surtout afin qu'elle puisse permettre de créer des applications interactives utilisant plusieurs types d'interaction différents en même temps comme, par exemple, des interactions de type WIMP avec des interactions de type vocales. À plus long terme, il faudra aussi étudier la possibilité d'étendre PERCOMOM à d'autres types d'applications que les applications interactives afin de permettre, à travers une seule méthode et un seul ensemble d'outils de modéliser et de créer le plus grand nombre possible de types d'applications différents.

Conclusion

Si PERCOMOM a permis d'apporter une nouvelle méthode de modélisation des applications interactives personnalisées de type WIMP et si sa première validation a permis d'en vérifier la pertinence, il n'en reste pas moins qu'aujourd'hui PERCOMOM est loin d'être une approche finalisée. Aujourd'hui, un de ses principaux manques réside dans l'absence d'une méthode de modélisation conceptuelle finalisée et d'outils permettant de prendre en charge l'ensemble des modèles et surtout permettant de générer de manière semi-automatique les applications conçues à partir de ces modèles. À notre avis, ce sont certainement là les deux axes principaux sur lesquels il conviendra de travailler, dans un proche avenir, si on veut pouvoir valider la pertinence de l'ensemble de l'approche auprès d'experts métier. À côté de cela, nos travaux sur PERCOMOM, nous ont permis d'ouvrir de nombreuses pistes de recherche différentes qui sont toutes importantes à explorer dans le cadre de la modélisation et de la personnalisation des applications interactives. C'est, très probablement, là un des autres apports de nos travaux de recherche. Ceux-ci nous ont aussi permis de voir les nombreux besoins en matière de décloisonnement des disciplines de recherche en montrant l'indispensable complémentarité des différentes disciplines pour permettre, à travers un enrichissement mutuel, la création de solutions pertinentes par rapport aux différentes problématiques posées par les applications interactives personnalisées.

Conclusion générale

Les travaux de recherche présentés dans ce mémoire se situent dans les thématiques de la modélisation des applications et de la personnalisation des contenus. Ils contribuent au développement des systèmes d'information personnalisés dans le domaine des transports.

Après une étude, dans le premier chapitre, des différentes méthodes et outils de modélisation des applications existants aujourd'hui, nous en avons vu les deux principales limites qui sont :

- L'absence d'une méthode de modélisation conceptuelle des applications qui soit capable de prendre en compte l'ensemble des problématiques associées au développement d'une application informatique.
- L'absence d'une méthode qui permette de manipuler des modèles conceptuels totalement indépendants des problématiques techniques.

Nous avons aussi montré l'intérêt d'utiliser la notion d'ontologie dans le cadre de la modélisation des applications informatiques et plus particulièrement dans le cadre d'une approche dirigée par les modèles.

Dans le deuxième chapitre, notre étude des systèmes de personnalisation des contenus, nous a permis de définir les principaux termes associés au domaine mais aussi de présenter les différentes solutions existantes aujourd'hui. Elle nous a aussi permis de voir que la personnalisation pouvait être soumise à de nombreuses contraintes et facteurs externes dont les principaux, dans le domaine du transport, sont :

- La prise en compte des spécificités de l'usager.
- La prise en compte du contexte,
- La prise en compte des spécificités liées aux types de contenus.

Cette étude nous a aussi permis de montrer qu'aujourd'hui, il n'existait pas de méthode ou d'outils permettant de prendre en charge la personnalisation des contenus dans le cadre d'une approche dirigée par les modèles. Et cela est d'autant plus vrai lorsqu'on désire prendre en compte les facteurs externes dans le cadre d'une personnalisation de contenu.

Dans le troisième chapitre, nous avons présenté notre méthode globale de modélisation, PERCOMOM (PERsonalization and COnceptual MOdeling Method), conforme à l'approche MDA, à travers une présentation d'un ensemble de modèles conceptuels, spécialisés chacun sur une problématique particulière. Nous avons montré comment ces modèles étaient reliés les uns aux autres permettant ainsi de modéliser, de manière aussi complète que possible, une application interactive dans le domaine des transports. Nous avons aussi présenté tout l'intérêt d'utiliser une approche basée sur les processus métier pour permettre de rendre les modèles plus facilement lisibles mais aussi plus facilement réutilisables. Nous avons aussi décrit une architecture permettant de prendre en charge les modèles conceptuels pour créer des applications concrètes directement utilisables par des usagers. Cette architecture, en introduisant la notion de services fonctionnels et techniques de niveau PIM, permet de simplifier les modèles conceptuels de niveaux CIM tout en introduisant plus de souplesse au niveau de la gestion des éléments communs aux différentes applications. Enfin, nous avons montré comment le passage des modèles de niveau CIM aux modèles de niveau PSM pouvait être réalisé de manière à permettre une génération semi-automatique des applications.

Dans le quatrième chapitre, nous avons proposé une nouvelle approche pour la prise en compte de la personnalisation des contenus dans le domaine des transports à travers l'utilisation de la notion de service fonctionnel. À côté de cela, nous avons montré comment il était possible de prendre en compte la problématique des facteurs externes liés au contexte, à l'utilisateur et à la nature des contenus aussi bien au niveau des modèles conceptuels qu'au niveau du service fonctionnel de niveau PIM. Nous avons montré que ceci permettait de rendre, au niveau conceptuel, la personnalisation complètement indépendante des solutions techniques utilisées pour la réaliser. Enfin, la solution proposée présente l'avantage de pouvoir facilement évoluer dans le temps pour prendre en compte de nouveaux facteurs de personnalisation ou pour étendre la prise en compte de certains facteurs de personnalisation.

Dans le cinquième chapitre, nous avons présenté quelques exemples d'utilisation de PERCOMOM dans le cadre de la modélisation d'applications d'informations voyageur dans le domaine de l'information voyageur. Nous avons aussi montré, à travers un exemple concret, comment le passage des modèles de niveau CIM aux modèles de niveaux PSM pouvait être réalisé dans PERCOMOM. Les résultats de ces expérimentations ont permis de valider l'approche dans sa globalité et surtout d'en appréhender certaines limites et contraintes.

Celles-ci ont été présentées dans le chapitre 6 à travers une évaluation globale de l'approche et à travers la définition de plusieurs axes de recherche dans les différents domaines couverts par PERCOMOM. Ceci nous a permis de montrer que si PERCOMOM apportait de nombreuses solutions intéressantes pour un grand nombre de problématiques de personnalisation elle était loin d'être finalisée et que de nombreux travaux doivent encore être menés pour la rendre utilisable et surtout pour lui permettre de prendre en compte un plus grand nombre de problématiqes fonctionnelles.

Pour conclure, nous dirons que PERCOMOM, en se basant sur une approche de type ingénierie dirigée par les modèles et en étant tournée vers les experts métier, représente, à notre avis, une évolution intéressante de la problématique de prise en charge des applications interactives personnalisées de type WIMP (Window, Icon, Mouse, Pointing Device). Surtout, elle ouvre la porte à la création de fabriques d'applications qui, à partir de modèles conceptuels créés par les utilisateurs, permettront de générer de manière semi-automatique des applications interactives personnalisées pour un grand nombre de plateformes techniques différentes.

Bibliographie

Aas K., *A Survey on Personalised Information Filtering Systems for the World Wide Web*. Research report n° 922, Oslo, Norway, Norwegian Computing Center, Norvège, 1997.

Abhinandan S. D., Mayur D., Ashutish F., Shyam R., Google news personalization: scalable online collaboratire filtering. *In Proceedings of 16th International Conference on World Wide Web*, Banff, Canada, 2007, pp. 271-280.

Abowd G., Atkeson C., Hong J, Long S., Kooper R., Pinkerton M., Cyberguide: A Mobile Context-Aware Tour Guide. *ACM Wireless Network*, Volume 3, 1997, pp. 421-433.

Abrahão S. M., Fons J., González M., Pastor O, Conceptual Modeling of Personalized Web Applications. *In Proceedings of the Second international Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, AH'2002*, Espagne, 2002, pp. 358-362.

Adelberg B., NoDoSE – a tool for semi-automatically extracting structured and semistructured data from text documents. *ACM SIGMOD Record*, Vol. 27, N° 2, 1998, pp. 283-294.

Ahmad F., de la Chica S., Butcher K., Sumner T., Martin J. H., Towards automatic conceptual personalization tools. *In Proceedings of the 7th ACM/IEEE Joint Conference on Digital Libraries, JCDL '07*, Vancouver, Canada, 2007, pp. 452-461.

Alexander C., Ishikawa S., Silverstein M., Jacobson M., Fiksdahl-King I., Angel S., *A Pattern Language Towns - Buildings – Construction*. Oxford University Press, Angleterre, 1977.

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

Amato G., Straccia U., User Profile Modeling and Applications to Digital Libraries. *In Proceedings of the third international conference on Research and Advanced Technology for Digital Libraries*, Paris, France, 1999, pp. 184-197.

Anand S. S., Mobasher B., Introduction to Intelligent Techniques for Web Personalization. *ACM Transactions on Internet Technology*, Vol. 7, N° 4, pp. 18, 2007.

Anli A., *Méthodologie de développement des systèmes d'information personalisés : Application à un système d'information au service des usagers des transports terrestres de personnes*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, 2006.

Annet J., Duncan K, Task Analysis and Training Design. *Occupational Psychology*, n° 41, 1967, pp. 211-227.

Aparicio A. S., Farias O. L. M., Dos Santos N., Applying ontologies in the integration of hérerogeneous relational databases. *In Proceedings of 2005 Australasian Ontology Workshop*, Sydney, Australie, 2005, pp. 11-16.

ATEC-ITS, *Pour un développement de l'information multimodale en agglomération : freins et perspectives*. Rapport du groupe de projet ITS-France Information multimodale en agglomération, France, 2002.

Balabanovic M., Shoham Y., Fab: Content-based collaborative recommendation. *Communication of the ACM*, Vol. 40, 1997, pp. 66-72.

Banâtre M., Couderc P., Pauty J., Becus M., Ubibus: Ubiquitous Computing to Help Blind People in Public Transport. *In Proceedings of the 6th International Symposium on Mobile Human-Computer, Mobile HCI 2004*, Glasgow, Grande Bretagne, 2004, pp. 310-314.

Balke W.-T., Wagner M., Toward Personalized Selection of Web Services. *In Proceedings of the 12th annual ACM International World Wide Web Conference, WWW2003*, Budapest, Hongrie, 2003, pp. 104-107.

Baraglia R., Silvestri F., Dynamic Personalization of Web Sites Without User Intervention. *Communications of the ACM*, vol. 50, n° 2, 2007, pp. 63-67.

Bardram J. E., Activity-Based Support for Mobility and Collaboration in Ubiquitous Computing. *In Proceedings of the Second International Conference on Ubiquitous Mobile Information and Collaboration Systems*, Riga, Lituanie, 2004, pp. 101-115.

Bardon D., Bjerke C., Vredenbourg K., *OVID Tutorial : Mastering the complexity of creating highly satisfying user experiences*. IBM, USA, 2003.

Baron M., Lucquiaud V., Autard D., Scapin D.L., K-MADe : un environnement pour le noyau du modèle de description de l'activité. *In Proceedings of the 18th French-speaking conference on Human-computer interaction*, Montreal, Canada, 2006, pp. 287-288.

Bastide R., Palanque P., UML for Interactive Systems: What is Missing. *In proceedings of INTERACT 2003 - Workshop on Software Engineering and HCI*, Zürich, Suisse, 2003, pp. 96-99.

Bastien J.M.C, Scapin D.L., Critères ergonomiques pour l'évaluation d'interfaces utilisateurs. *Rapport technique INRIA n°156*, INRIA, Le Chesnay, France, 1993.

Basu C., Hirsch H., Cohen W., Recommandation as Classification: Using social and content based Information in Recommandation. *In Proceedings of the 15th National Conference on Artificial Intelligence, AAAI'98*, Madison, USA, 1998, pp. 714-720.

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

Beale, R., Bordbar, B., Using modelling to put HCI design patterns to work. *In proceedings of HCI International. 11th International Conference on Human-Computer Interaction* Associates, Las Vegas, Nevada, USA, 2005.

Becker M., Smith S. F., An Ontology for Multi-Modal Transportation Planning and Scheduling. *Technical Report CMU-RI-TR-98-15*, Robotics Institute, Carnegie Mellon University, Pittsburgh, USA, 1997.

Bekiaris E., Panou M., Moussadakou A., Elderly and Disabled Travelers Needs in Informobility Services. *Lecture Notes in Computer Science*, Vol. 4554, Springer, Berlin, Allemagne, 2007, pp. 853-860.

Berners-Lee T, Fischetti M., *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. Harper, San Francisco, USA, 1999.

Berners-Lee T., Hendler James, Lassila O., The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American Magazine*, numéro du mois de mai, 2001.

Bernonville S., Kolski C., Beuscart-Zéphir M., Contribution and limits of UML models for task modelling in a complex organizational context: case study in the healthcare domain. *In Proceedings of The 5th International Business Information Management Association Conference – IBIMA 2005*, Le caire, Egypte, 2005, pp. 119-127.

Blanc X., Salvatori O., *MDA en action - Ingénierie logicielle guidée par les modèles*. Editions Eyrolles, Paris, 2005.

Bonnet S., Model Driven Software Personalization, *In Proceedings of Smart Objects Conference. SOC 2003*, Grenoble, France, 2003, pp. 114-117.

BPMI, *Business Process Modeling Notation version 1.0*. 2004, accessible à http://www.bpmn.org.

Brajnik G., Tasso C., A Shell for Developing Non-monotonic User Modeling Systems. *International Journal of Human-Computer Studies*, N° 40, 1994, pp. 31-62.

Brave S., Nass C., Emotion in human-computer interaction. In J.A. Jacko, A. Sears, *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications, Human Factors And Ergonomics*, L. Erlbaum Associates Inc., New Jersey, USA, 2002, pp. 81-96.

Brossard A., *L'ingénierie dirigée par les modèles appliquée au développement d'interfaces graphiques personnalisées : une approche à travers les processus metier*. Mémoire de Master Recherche, Université de Valenciennes, 2006.

Brossard A., Abed M., Modélisation des IHM : Une approche basée sur les processus metier. *In Proceedings of Nouvelles tendances technologiques en génie électrique & informatique, GEI'2007*, Sfax, Tunisie, 2007, pp. 91-101.

Brossard A., Abed M., Kolski C., *Modélisation conceptuelle des IHM : Une approche globale s'appuyant sur les processus metier*. Ingénierie des Systèmes d'Information - Revue des sciences et technologies de l'information, Numéro 2007/5, France, 2007, pp. 69-108.

Brossard A., IHM et IDM : un modèle statique d'IHM au service des processus metier. *In Proceedings of 19ème Conférence de l'Association Francophone d'Interaction Homme-Machine, IHM 2007*, Paris, France, 2007, pp. 221-224.

Brossard A., Abed M., Kolski C., Context Awareness and Model Driven Engineering: A multi-level Approach for the Development of Interactive Applications in Public Transportation. *In proceedings of 27th European Annual Conference on Human Decision-Making and Manual Control,* EAM'08, Delft, Hollande, 2008.

Browne D., Totterdell P., Norman M., *Adaptive User Interfaces*. Academic Press, Computer And People Series, USA, 1990.

Buneman P., Semistructured data. *In Proceedings of ACM Symposium on Principles of Database Systems (PODS)*, Tucson, Arizona, 1997, pp. 117–121.

Burke R., Mobasher B., Zabicki R., Bhaumik R., Identifying attack models for secure recommendation. *In Proceedings of Workshop on the Next-Generation of Recommender Systems*, San Diego, USA, 2005.

Calvary G., Coutaz J., Plasticité des interfaces : une nécessité. *Information-interaction-intelligence, In Proceedings of deuxièmes Assises nationals du GDR I3*, Cépaduès Editions, Nancy, France, 2002, pp 247-261.

Calvary G., Coutaz J., Daassi O., Balme L., Demeure A., Towards a new Generation of widgets for supporting software plasticity: the comet. *In Proceedings of 9th IFIP Working Conference on Engineering for Human-Computer Interaction*, Hambourg, Allemagne, 2003.

Campos P. F., Jardim Nunes N., CanonSketch : a User-Centered Tool for Canonical Abstract Prototyping. *In Proceedings of the 9th IFIP Working Conference on Engineering for Human-Computer Interaction Jointly with The 11th International Workshop on Design, Specification and Verification of Interactive Systems, EHCI/DSVIS 2004*, Hambourg, Allemagne, 2004, pp. 146-163

Card S.K., Moran T.P., Newell A., *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, 1983.

Carton A., Clarke S., Senart A., Cahill V., Aspect-Oriented Model-Driven Development for Mobile Context-Aware Computing. *In Proceedings of 29th International Conference on Software Enginnering Workshop, ICSE'07*, Mineapolis, USA, 2007.

Chan L., Darido G., Jackson D., Laver R., Schneck D., Hamilton B. A., *Real-time Bus Arrival Information Systems - Retun-on-Investment Study – Final Report*. U.S. Department of Transportation, Federal Transit Administration, Document OMB n° 0704 0188, Washington, USA, 2006.

Chen H., Finin T, Joshi A., Semantic Web in the Context Broker Architecture. *In Proceedings of Second IEEE international Conference on Pervasive Computing and Communication, PerCOM'04*, Orlando, USA, 2004, pp. 277-286.

Chen P., The Entity-Relationship Model – Toward a Unified View of Data. *ACM Transactions on Database Systems*, Volume 1, Issue 1, 1976, pp. 9-36.

ChoiceStream, *2006 ChoiceStream Personalization Survey*. 2006, accessible à http://www.choicestream.com.

Chu E., Baid A., Chen T., Doan A., Naughton J., A relational approach to incrementally extracting and querying structure in unstructured data. *In Proceedings of the 33rd international Conference on Very Large Data Bases*, Vienne, Autriche, 2007, pp. 1045-1056.

Clark T., Evans A., Sammut P., Willans J., *Applied Metamodelling: A Foundation for Language Driven Development Version 0.1*. Société Xactium, 2004.

Centreline Solutions Incorporated, *10 Major Cause of Project Failure*. support de cours pour le Project Management Institute Greater Toronto, Toronto, Canada, 2005, accessible à http://gtislig.org/Documents/10_Major_Causes_of_Project_Failure.pdf.

Codd E. F., A relational model of data for large shared data banks. *Communications of the ACM*, Vol. 13, N° 6, 1970, pp. 377–387.

Cook S., Domain-Specific Modeling and Model Driven Architecture. *MDA Journal*, January 2004, pp. 2-10.

Cooper A., Reimann R., Cronin D., *About Face 3: The Essentials of Interaction Design*. Wiley Publishing Inc., 2007.

Cornelis B., *Personalizing search in digital libraries*. Masters of Science Thesis, Université de Maastricht, Pays-Bas, 2001.

Cowie R., Douglas-Cowie E., Tsapatsoulis N., Votsis G., Kollias S., Fellenz W. et Taylor JG., Emotion recognition in human-computer interaction. *IEEE Signal Processing Magazine*, Vol. 18, No. 1, 2001, pp. 32-80.

Czarmeski K., Helsen S., Feature-based survey of model transformation approaches. *IBM System Journal*, Vol. 45, N° 3, 2006, pp. 621-645.

Dachselt, R., Hinz, M., Pietschmann, S., Using the AMACONT architecture for flexible adaptation of 3D web applications. *In Proceedings of the eleventh international conference on 3D web technology, Web 3D '06,* Columbia, USA, 2006, pp. 75-84.

Davenport H., *Process Innovation: Reengineering Work Through Information Technology*. Harvard Business School Press, Boston, Etats-Unis, 1992.

Dey A, Understanding and Using Context. *Personal and Ubiquitous Computing*, Vol. 5, 2001, pp. 4-7.

Dey A., Abowd G., Towards a Better Understanding of Context and Context-Awareness. *In Proceedings of CHIA'00 workshop on Context-Awareness*, 2000.

Dikjman R. M., Dumas M., Ouyang C., *Formal Semantics and Analysis of BPMN Process Models using Petri Nets.* Technical Report, Queensland University of Technology, 2007, accessible à https://eprints.qut.edu.au/archive/00007115/.

Dinoff R., Hull R., Kumar B., Lieuwen D., Santos P., Learning and managing user context in personalized communications services. *In Proceedings of International Workshop on Context in Advanced interfaces,* Venise, Italie, 2006.

Dyche J., *Model Driven Architecture*. Addison-Wesley Educational Publishers, USA, 2002.

Embley D. W., Campbell D. M., Smith R. D., Liddle S. W., Ontology-based extraction and structuring of information from data-rich unstructured documents. *In Proceedings of the seventh international conference on Information and Knowledge Management*, Bethesda, USA, 1998, pp. 52-29.

Eriksson H.E., Penker M., *Business Modeling with UML*. John Wiley & Sons Ltd, 2000.

Eyermann L.J., Smith A.B., Roberts E.F., *What Senior Management Needs to Know about the Value of MDA*. 2004, société IMI, accessible à http://www.omg.org/news/whitepapers/OMG-MDA-Final-Ariticle_June-2004v6.pdf.

Favre J.-M., Toward a Basic Theory to Model: Model Driven Engineering. *Workshop on Software Model Engineering, Wisme 2004*, Lisbonne, Portugal, 2004.

Favre J.-M., Estublier J., Blay-Fornarino M., *L'ingénierie dirigée par les modèles*. Hermès-Lavoisier, Paris, France, 2006.

Ferraiolo D.F., Kuhn D.R., Role Based Access Control. *In proceeding of 15th National Computer Security Conference*, USA, 1992.

Fichelet M., Fichelet R., May N., *Contribution à une psychosociologie des comportements urbains*. Ministère de l'Equipement et du Logement, Paris, 1970.

Finin T. W., GUMS: A General User Modeling Shell. *In User Models in Dialog Systems*, Kobsa A. and Wahlster W. Editeurs, Berlin, 1989, pp. 411-430.

Finin T. W., Drager D., GUMS1: A General User Modeling System. *In Proceedings of Sixth Canadian Conference on Artificial Intelligence*, Montreal, Canada, 1986, pp. 24-29.

Fink J., Kobsa A., Nill A., Adaptable and Adaptive Information Provision for All Users, Including the Disabled and the Elderly. *The New Review of Hypermedia and Multimedia*, Vol. 4, 1998, pp. 163-188.

Firas B., *Conception et développement des outils graphiques prenant en charge les modèles conceptuels dans la plateforme PERCOMOM*, Mémoire de fin d'étude, Ecole Nationale des Sciences de l'Informatique, Université de la Manouba, Tunisie, 2008.

Flam M., A qualitative perspective on travel time experience. *5th Swiss Transport Research Conference*, STRC-2005, Suisse, 2005.

Florescu D., Kossmann D., Storing and Querying XML Data Using an RDBMS. *IEEE Data Engineering Bulletin*, Vol. 22, N° 3, 1999, pp. 27-34.

Florins M., *Graceful Degradation: a Method for Designing Multiplatform Graphical User Interfaces*. Thèse de doctorat. Université Catholique de Louvain, Belgique, 2006.

Forbrig P., Dittmar A., Reichart D., Sinnig D., From Models to Interactive Systems Tool Support and XIML. *In Proceedings of the First International Workshop on Making model-based user interface design practical: usable and open methods and tools, MBUI 2004*, Funchal, Portugal, 2004.

Frankel D., *Model driven architecture: applying MDA to enterprise computing*. Wiley, 2003.

Frankel D., *Reflection of the State of MDA, MDA Journal*, Meghan-Kiffer Press, 2005, pp. 1-9.

Gauch S., Chaffee J., Pretschner A., Ontology-Based User Profiles for Search and Browsing. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, Special Issue on User Modeling for Web and Hypermedia Information Retrieval, 2003, pp. 1-3.

Gendre P., *Systèmes d'information multimodale : une bibliographie commentée*. Rapport technique, CERTU, Paris, France, 1999.

Goecks J., Shavlik J., Learning Users' Interests by Unobtrusively Observing Their Normal Behavior. *In Proceedings of 2000 International Conference on Intelligent User Interfaces*, 2000, pp. 129-132.

Goldberg D., Nichols D., Oki B., Terry D., Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, Vol. 35, 1992, pp. 61-70.

Golemati M., Katifori A., Vassilakis C., Lepouras G., Halatsis C., Creating an Ontology for the User Profile: Method and Applications. *In Proceedings of the First IEEE International Conference on Research Challenges in Information Science, RCIS*, Maroc, 2007.

Gross T., Specht M., Awareness in Context-Aware Information Systems. *In Proceeding of Mensch & Computer 01*, Bonn, Allemagne, 2001, pp. 173-181.

Guerrero J., Vanderdonckt J., FlowiXML: a step towards designing workflow management systems. *International Journal Web Engineering*, Vol. 8, N° 2, 2008, pp. 163-182.

Guo, W., Blythe P.T., Intelligent Personalised Traveller Information: Impacts on the Choice of Travel Mode. *In Proceedings of 12th World Congress on Intelligent Transport Systems and Services*, San Francisco, USA, 2005.

Hagen P., Manning H., Souza. R., *Smart Personalization*. Forrester Research, USA, 1999.

Hailpern B., Tarr P., Model-driven development: The good, the bad and the ugly. *IBM System Journal*, Vol. 45, N° 3, 2006, pp. 451-461.

Hanumansetty R., *Model Based Approach for Context Aware and Adaptive Interface Generation*. Ph.D. Thesis, Virginia Polytechnic Institute and State University, USA, 2004.

Halin G., Kubicki S., Architecture dirigée par les modèles pour une representation multi-vues du contexte de cooperation. *In Proceedings of IHM 2005*, Toulouse, France, 2005, pp. 211-214.

Hartson H. R., Siochi A. C., Hix D., The UAN: a User-Oriented representation for direct manipulation interface design. *ACM Trans. Inf. Syst*., Vol. 8, N° 3, 1990, pp. 181-203.

Havey M., *Essential Business Process Modeling*. O'Reilly, 2005.

Henderson-Sellers B., Cook S., Mellor S., Miller J., Selic B., *UML the Good, the Bad or the Ugly, Perspective from a panel of experts*. Software and Systems Modeling, Vol. 4, N° 1, 2005, pp. 4-13.

Hinz, M., Fiala, Z., AMACONT: A System Architecture for Adaptive Multimedia Web Applications. *In Proceedings of the Berliner XML Tage 2004*, Berlin, Allemagne, 2004, pp. 65-74.

Houseman E. M., Kaskela D. E., State of the art of selective dissemination of information. *IEEE Trans. Eng. Writing Speech*, Vol. 3, 1970, pp. 78-83.

Hoyer R., Czogolla O., Approach to personalized information services in public transport. *In Proceedings of 9th world congress of intelligent information systems*, Chicago, USA, 2002.

Injun C., Park C., Lee C., Task net: Transactional workflow based on colored Petri net. *European Journal of Operational Research*, Vol. 136, N° 2, 2002, pp. 383-402.

INRETS, Viatic.Mobilité : Les services d'aide à la mobilité en Région Nord Pas de Calais. *Rapport final, INRETS/RR-08-701-FR*, France, 2008.

ITS, Pour un développement de l'information multimodale en agglomeration : freins et perspectives. *Rapport 2002 du groupe de projet ITS France "information multimodale en agglomeration"*.

Jacko J.A., Sears A. (Eds.). *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*. Lawrence Erlbaum & Associates, 2003.

Jacquet C., Bellik Y., Bourda Y., PRIAM : Affichage dynamique d'informations par des écrans coopérants en environnement mobile. *In Proceedings of Conférence francophone ubiquité et mobilité, Ubimob 2006*, France, 2006, pp. 33-40.

Jardim Nunes D.N., *Object Modeling for User-Centered Development and User Interface Design: The Wisdom Approach.* Ph.D. Thesis, Université de Madeira, Portugal, 2001.

Jardim Nunes D.N., Representating User-Interface Patterns in UML. *In Proceeding of 9th International Conference on Object-Oriented Information Systems, OOIS 2003*, Genève, Suisse, 2003, pp. 142-163.

Jardim Nunes D.N., Campos P., Toward Usable Analysis, Design and Modeling Tools. *In Proceedings of the First International Workshop on Making model-based user interface design practical: usable and open methods and tools*, ISSN 1613-0073. Vol. 103., accessible à http://CEUR-WS.org/Vol-103, 2004.

Jeston J., Nelis J., *Business Process Management: Practical Guidelines to Successful Implementations*. Butterworth-Heinemann, 2006.

Johansson H., McHugh P., Pendlebury A., Wheller W., *Business Process Reengineering: Breakpoint Strategies for Market Dominance*. Wiley, Chichester, Angleterre, 1993.

Juguet J., Chevrier S., Petit-Rozé C., Uster G., Radiographie du voyageur : de l'(in)activité aux services à la mobilité. *Congrès ATEC-ITS*, Issy les Moulineaux, France, 2007.

Khan L., Structuring and querying personalized audio using ontologies. *In Proceedings of the seventh ACM International Conference on Multimedia*, Orlando, USA, 1999, pp. 209-210.

Kenyon S., Lyons G., The Value of Integrated Multimodal Information and its Potential Contribution to Modal Change. *Transportation Research Part F – Traffic Psychology and Behaviour*, Grande-Bretagne, 2003, pp. 1-21.

Kimball R., Mertz R., *Le Data Web House : Analyser les comportements clients*. Editions Eyrolles, Paris, 2000.

Kjeldskov J., Howard S., Murphy J., Carroll J., Vetere F., Graham C., Designing TramMatena: a context-aware mobile system supporting use of public transportation. *In Proceedings of the 2003 conference on Designing for User Experiences*, 2003, pp. 1-4.

Kobsa A., User Modelling: Recent work, prospects and hazards, Adaptive User Interfaces: Principles and Practices. In Schneider-Hufschmidt, T. Kuhme, U. Malinowski, *Adaptive User Interfaces: Principles and Practice*. Elsevier Science Publishers B.V., Amsterdam, 1993, pp. 111-128.

Kobsa A., Generic User Modeling Systems. *User Modeling and User-Adapted Interaction*, N° 11, 2001, pp. 49–63.

Kobsa A., Pohl W., The User Modeling Shell System BGP-MS. *User Modeling and User-Adapted Interaction*, Vol. 4, N° 2, 1995, pp. 59-106.

Kolski C., Analyse et conception de l'IHM. *Interaction Homme-Machine pour les Systèmes d'information*, volume 1, Hermès, Paris, 2001.

Korfhage R. R., *Information Storage and Retrieval*. Wiley Computer Publishing, 1997.

Kostadinov D., *Personnalisation de l'information et gestion des profils utilisateurs.* Rapport de master recherche, Université de Versailles, France, 2003.

Lam S.H., Xie F., Provision of Personnalised Transit Travel Information System and Architecture. *In Proceedings of 9th world congress on intelligent transportation systems*, Chicago, USA, 2002.

Nanci D., Espinasse B. *Ingénierie des systèmes d'information : Merise - Deuxième génération, 4ème edition*, Vuibert, Paris, France, 2001.

Lecomte N., Patesson R., Le panel des voyageurs: une étude des activités et des besoins d'informations des utilisateurs des transports publics. *In Proceeding of Conférence ERGO-IHM, ERGO-IHM'00,* Biarritz, France, 2000, pp. 129-135.

Limbourg Q., Vanderdonckt J., Comparing Task Model for User Interface Design. In J.A Jacko, 1. Sears (Eds), *The Handbook of Task Analysis for Human-Computer Interaction*, Lawrence Erlbaum Associates, 2003, pp. 135-154.

Liu H., Lu Y., Yang Q., XML Conceptual modeling with XUML. *In Proceedings of the 28th International Conference on Software engineering*, Shanghai, China, 2006, pp. 973-976.

Lorenz B., Ohlbach H. J., Yang L., *Ontology of Transportation Network. Délivrable A1-D4 du projet REasoning on the WEb with Rules and SEmantic*, REWERSE, 2005.

Losio B. F., Salgado A. C., Galvdo L. R., Conceptual modelling of XML schemas. *In Proceedings of the 5th ACM international workshop on Web information and data management*, New Orleans, 2003, pp. 102-105.

Lumineau N., *Un tour d'horizon du filtrage collaboratif.* Document réalisé dans le cadre de l'AS personnalisation, Laboratoire d'Informatique de Paris 6, LIP6, France, 2003.

Lyons, G., Harman, R., Austin, J. Duff, A., Traveller Information Systems Research: A Review and Recommendations for Transport Direct. *Report to the Department for Transport*, Local Government and the Regions, August, England, 2001.

Lyons G., The role of information in decision-making with regard to travel. *IEEE Proceedings, Intelligent Transport Systems*, Vol. 153, N° 3, 2006, pp. 199-212.

Lyons G., Avineri E., Farag S., Harman R., Strategic Review of Travel Information Research. *Final Report to the Department for Transport*, University of the West of England, Bristol, England, 2007.

Manber U., Patel A., Robison J., Experience with Personalization on Yahoo!. *Communications of the ACM*, Vo. 43, n° 8, 2000, pp. 35-39.

Maclean S. D., Dailey D. J., Real-Time bus information on mobile devices. *In Proceedings of Intelligent Transportation Systems 2001, ITS 2001*, Oakland, USA, 2001, pp. 988-993.

Magnani M., Montesi D., A Unified Approach to Structured, Semistructured and Unstructured Data. *Technical Report UBLCS-2004-9*, University of Bologna, Italie, 2004.

Mapp D., Edmondson D., Lafferty J., Third Party Retailing. *Report to the Association of Train Operating Companies*, MVA, Grande Bretagne, 2000.

Martinez R, *A Development Method for User Interfaces of Rich Internet Applications.* Rapport de DEA, Université Catholique de Louvain, Louvain-la-Neuve, 2007.

Marzloff, B. et Glaziou, S., *Le temps des puces.* Editions Carnot, Bourges, 1999.

Matsubara H., Fukasawa N., Goto K., Kawai K., Sato T., Aoki T., Mizukami N., Fujinami K., Shinomiya A., Interactive Guidance System for Passengers. *Quarterly Report of RTRI*, Vol. 42, N° 4, Japon, 2001, pp 201-206.

Mobasher B., Cooley R., Srivastava J., Automatic personalization based on Web usage mining. *Communications of the ACM*, Vol. 43, N° 8, 2000, pp. 142-151.

Mokhtarian P.L., Salomon I., How derived is the demand for travel? Some conceptual and measurement considerations. *Transportation Research Part A.35*, 2001, pp. 695-719.

Mouloudi A., *Intégration des besoins des utilisateurs pour la conception de systèmes d'information interactifs : Application à la conception d'un système d'information voyageurs multimodal (SIVM).* Thèse de doctorat, Université de technologie Compiègne, 2007.

Muller P.-A., Studer P., Fondement F., Bezivin J., Platform independent Web application modeling and development with Netsilon. *Software & System Modeling*, Vol. 4, N° 4, 2005, pp. 424-442.

Molina P., A Review to Model-Based User Interface Development Technology. *In Proceedings of the First International Workshop on Making model-based user interface design practical: usable and open methods and tools*, MBUI 2004, Funchal, Portugal, 2004.

Montero F., Solving the Mapping Problem in UI Design by Seamless Integration in IDEALXML. *In Proceedings of Design Specification and Verification of Interactive Systems*, DSVIS'05, Newcastle, Royaume-Uni, 2005.

Mori G., Paternò F., Santoro C., CTTE: Support for Developing and Analysing Task Models for Interactive System Design. *IEEE Transactions on Software Engineering*, Vol. 28, N° 9, 2002, pp. 797-813.

Necasky M., XSEM: a conceptual model for XML. *In Proceedings of the 4th Asia-Pacific conference on Conceptual Modeling*, Ballarat, Australie, 2007, pp. 37-48.

Nichols J., Faulring A., Automatic Interface Generation and Future User Interface Tools. *In Proceedings of the ACM CHI 2005 Workshop: The Future of User Interface Design Tools*, Pittsburgh, USA, 2005.

Norman D. A., Some observations on mental models. *In Mental models*, D. Gentner and A. L. Stevens (Eds.), New Jersey, USA, 1983, pp. 7-14.

Norman D.A., Cognitive engineering. Norman D.A. and Draper S.W. (Eds), *User centered system design: New perspectives on human-computer interaction*, New Jersey, USA, pp. 32-65.

Noy N. F., McGuiness D. L., Ontology Development 101 : A Guide to Creating Your First Ontology. *Stanford Knowledge System Laboratory Technical Report KLS-01-05 and Standford Medical Informatics Technical Report SMI-2001-0880*, Standford, USA, 2001.

Oasis, Reference *Model for Service Oriented Architecture 1.0*. 2006, accessible à http://xml.coverpages.org/OASIS-SOA-RM-CS.pdf .

OMG, *Introduction To OMG's Unified Modeling Language (UML)*. 2005, accessible à http://www.omg.org/gettingstarted/what_is_uml.htm .

OMG, *MDA Guide v1.0.1*. 2003, accessible à http://www.omg.org/cgi-bin/doc?omg/03-06-01.

OMG, *Meta Object Facility (MOF) Core Specification Version 2.0*. 2006, accessible à http://www.omg.org/spec/MOF/2.0/PDF.

OMG, *Model Driven Architecture*. Model Driven Architecture (MDA), Document number ormsc/2001-07-01. Technical report. 2001.

OMG, *MOF 2.0 / XMI Mapping Specification Version 2.1.1*. 2007, accessible à http://www.omg.org/technology/documents/formal/xmi.htm .

OMG, *Semantic of Business Vocabulary and Business Rules (SBVR)*. 2005, accessible à http://www.omg.org/technology/documents/bms_spec_catalog.htm.

Opperman R., Simm H., Adaptability: User-initiated individualization. In R. Oppermann (Ed.), *Adaptative User Support*, Lawrence Earlbaum, 1994, pp. 14-66.

Orain H., Du côté des trajets. In: Juan, S., Largo-Poirier, A., Orain, H. et Poltorak, J.- F. (eds.), *Les sentiers du quotidien - Rigidité, fluidité des espaces sociaux et trajets routiniers en ville*. L'Harmattan, Paris, 1997, pp. 97-119.

Oya M., MDA and System Design. *MDA Information Day*, OMG Technical Meeting, 2002, accessible à http://www.omg.org/mda/mda_files/E_MDADay_Oya.pdf.

Paiva A., Self J., TAGUS: A User and Learner Modeling System. *In Proceedings of the Fourth International Conference on User Modeling*. Hyannis, USA, 1994, pp. 43-49.

Pascoe J., Ryan N., Morse D., Issues in Developing Context-Aware computing. *In Proceedings of International Symposium on Handled and Ubiquitous Computing, HUC'99*, Karlsruhe, Allemagne, 1999, pp. 208-221.

Palano R., Pandurino A., Guido A.L., Conceptual design of web application families : the BWW approach. *In Proceedings of 6th Workshop on Domain Specific Modeling*, Portland, USA, 2006, pp. 23-32.

Paterno F., *Model-Based Design and Evaluation of Interactive Applications.* Springer-Verlag, Londres, Angleterre, 2000.

Paternò F., Mancini C., Meniconi S., ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. *In Proceedings of INTERACT 1997*, Sydney, Australie, 1997, pp. 362-369.

Pérez-Medina J.L, Dupuy-Chessa S., Front A., A Survey of Model Engineering Tools for User Interface Design. *In Proceedings of 6th International Workshop on TAsk Models and DIAgrams TAMODIA'2007*, Toulouse, France, 2007.

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

Petit-Rozé C., *Organisation multi-agents au service de la personnalisation des informations. Application à un système d'information multimodale pour le transport terrestre des personnes*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, 2003.

Petit-Rozé C., Anli A., Grislin-Le Strugeon E., Abed M., Kolski C., *AGENPERSO - Interfaces Homme-Machine à base d'AGENts logiciels PERSOnnels d'information aux usagers des TC*. Rapport final du projet PREDIT, Valenciennes, France, LAMIH, 2003.

Pomerol J.C., Brézillon P., *About some relationships between Knowledge and Context*. Modeling and Using Context (CONTEXT-01), Lecture Notes in Computer Science, Springer Verlag, 2001, pp. 461-464.

Raskin J., *The Human Interface, New Direction for designing Interactive System*. Addison Wesley, 2000.

Razmerita L., Services Contextualisés pour Utilisateurs et la Modélisation des Utilisateurs à base d'Ontologie: Défis et Perspectives. *l'Atelier sur la Modélisation Utilisateur et la Personnalisation d'Interfaces Homme-Machine, en conjonction avec la conférence Extraction et Gestion des Connaissances*, Paris, France, 2005.

Razmerita L., Angehrn A., Maedche A., Ontology based user modeling for Knowledge Management Systems. *In Proceedings of the 2003 User Modeling Conference*, Pittsburgh, USA, 2003, pp. 213-217.

Reisig W., *Petri-Net: an introduction*. Springer-Verlag, New York, USA, 1985.

Ribaud V., Saliou P., Kerboeuf M., Model Driven Engineering: Two Approaches through the same Case Study. *In proceedings of 13th Interdisciplinary Information Management Talks, IDIMT-2005*, Budweis, République Tchèque, 2005, pp. 215-234.

Rich E., Users are individuals: individualizing user models. *International Journal of Man-machine Studies*, N° 18, 1983, pp. 199-214.

Roques P., *UML modèliser un site e-commerce*. Editions Eyrolles, Paris, France, 2002.

Rupert B., Wright J., Pretorius P., Cook G., Hutchinson K., Kell W. T., Lister H. M., Nevarez M., Sanders L., Schuman R., Taylor R., Almborg J., Traveler Information Systems in Europe. Office of International Programs, U.S. Department of Transportation, *Report n° FHWA-PL-03-005*, 2003

Salton G., McGill M., *Introduction to modern information retrieval*. McGraw-Hill, USA, 1983.

Sandhu R. S., Coyne E. J., Feinstein H. L., Youman C. E., Role Based Access Control Models. *IEEE Computer*, Vol. 29, USA, 1996, pp. 38-47.

Sarvas R., Herrarte E., Wilhelm A., Davis M., Metadata creation system for mobile images. *In Proceedings of the 2nd international Conference on Mobile Systems, Applications, and Services, MobiSys '04*. Boston, USA, 2004, pp. 36-48.

Sarwar B. M., Konstan J. A., Borchers A., Herlocker J., Miller B., Riedl J., Using filtering agents to improve prediction quality in the GroupLens research collaborative filtering system. *In Proceedings of Conference on Computer Supported Cooperative Work*, CSCW'98, Seattle, USA, 1998, pp. 345-354.

Schattkowsky T., Lohmann M., Towards Employing UML Model Mappings for Platform Independent User Interface Design. *Satellite Events at the MoDELS 2005 Conference: MoDELS 2005 International Workshops OCLWS, MoDeVA, MARTES, AOM, MTiP, WiSME, MODAUI, NfC, MDD, WUsCAM*, Montego Bay, Jamaïque, Volume 3844, 2005, pp. 201 – 209.

Schilit B., Adams N., Want R., Context Aware Computing Applications. *In Proceedings of First International Workshop on Mobile Computing Systems ans Application*, 1994, pp. 85-90.

Scappapietra S., Exploring New Directions in Conceptual Modeling. *In Proceedings of Fourth Asia-Pacific Conference on Conceptual Modeling, APCCM2007,* Ballarat, Australie, 2007.

Schlee M., Vanderdonckt J., Generative Programming of Graphical User Interfaces. *In Proceeding of the working conference on Advanced Visual Interfaces, AVI 2004*, Gallipoli, Italie, 2004, p.403-406.

Schneiderman B., *Designing the User Interface*. Addison Wesley, 2004.

Seffah A., Javahery H., *Multiple User Interfaces - Cross Platform Application and Context-Aware Interfaces.* John Wiley & Sons Ltd Editor, 2004.

Selic B., The Pragmatics of Model-Driven Development. *IEEE Software*, Vol. 20, N° 5, 2003, pp. 19-25.

Shardanand U., Maes P., Social Information Filtering: Algorithms for Automating 'Word of Mouth'. *In Proceedings of the CHI-95 Conference*, Denver, USA, 1995.

Shirky C., *Ontology is Overrated: Categories, Links, and Tags*. Synthèse de deux présentations faites à O'Reilly ETech conference en mars 2005 et à IMCExpo en avril 2005, disponible à : http://www.shirky.com/writings/ontology_overrated.html.

Silver B., *The BPMS Value Proposition. Industry Trend Report*, Bruce Silver Associates, USA, 2007.

Sottet J.-S., Calvary G., Favre J.-M., Ingénierie de l'Interaction Homme-Machine Dirigée par les modèles. *In Proceeding of IDM'05*, Paris, France, 2005, pp 67-82.

Sottet J.-S., Calvary G., Favre J.-M., Coutaz J., IHM & IDM : Un tandem prometteur. Poster, *Ergo'IA 2006*, Biarritz, France, 2006.

Sottet J.-S., Ganneau V., Calvary G., Favre J.-M., Coutaz J., Favre J.-M., Demumieux R., Model-Driven Adaptation for Plastic User Interface. *In proceedings of Interact 2007*, Rio de Janeiro, Brésil, 2007, pp. 397-410.

Sperandio J.-C., Uzan G., Jobard N., *Difficultés rencontrées par les aveugles et déficients visuels dans la consultation de sites Web sur les transports*. Rapport d'étude, Institut pour la Ville en Mouvement, Paris, France, 2002.

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

Stuckenschmidt H., Van Harmelen F., Ontology-based metadata generation from semi-structured information. *In Proceeding of the 1st International Conference on Knowledge capture*, Victoria, Canada, 2001, pp. 163-170.

Sowa J. F., *Knowledge Representation: Logical, Philosophical and Computational Foundations*. Brools/Cole, New-York, USA, 2000.

Sugumaran V., Storey V. C., The role of domain ontologies in database design: An ontology management and conceptual modeling environment. *ACM Trans. Database Syst.*, Vol. 31, n° 3, 2006, pp. 1064-1094.

Tadj C., Ngantchaha G., Context handling in a pervasive computing system framework. *In Proceedings of the 3rd international conference on Mobile Technology Application & Systems*, Article n° 13, Bangkok, Thaïlande, 2006.

Tardieu H., Rochfeld A., Colletti R., *La méthode Merise – Tome 1 : Principe et outils*. Editions d'organisation, Paris, 1983.

Tariq N.A., Akhter N. Comparison of Model Driven Architecture (MDA) based tools. *In Proceedings of the 13th Nordic Baltic Conference on Biomedical Engineering and Medical Physics, IFMBE,* Umea, Suède, 2004, pp. 43-44.

Timpf S., Ontologies of Wayfinding: a Traveler's Perspective. *Networks and Spatial Economics*, Vol. 2, N° 1, 2002, pp. 9-35.

Thévenin D., Coutaz J., Plasticity of User Interfaces : Frameworks and Research Agenda. *In Proceedings of IFIP 13th International Conference on Human-Computer Interaction, Interact'99*, Edinburgh, Royaume-Uni, 1999, pp. 110-117.

Trajkova J., Gauch S., Improving Ontology-based User Profiles. *In Proceedings of RIAO 2004*, Avignon, France, 2004, pp. 380-389.

Trinkunas J., Vasilecas O., Building ontologies from relational databases using reverse engineering methods. *In Proceedings of 2007 International Conference on Computer systems and technologies*, Bulgarie, Article n° 13, 2007.

Turk D. E., Vijayasarathy L. R., Clark, J. D., The value of conceptual modeling in database development: An experimental investigation. *In Proceedings of the Evaluation of Modeling Methods in Systems Analysis and Design Conference*, Velden, Autriche, 2003.

Uster G., *État des réflexions sur le développement de l'information multimodale en France*. Congrès UITP, Hanovre, Allemagne, 2000.

Uster G., Développement de l'information multimodale en France : quels leviers actionner ?. *Annales des Ponts et Chaussées*, N° 98, Paris, France, 2001, pp. 22-26.

Uster G., Service de mobilité et d'information : innovation et recherché. La documentation Française, Collection "Le point sur", *Programme de recherché et d'innovation dans les transports terrestres (PREDIT)*, Paris, France, 2008.

Vanderdonckt J., *Guide ergonomique des interfaces homme-machine*. Presses Universitaires de Namur, Namur, Belgique, 1994.

Vanderdonckt J., Development Milestones toward a Tool For Working With Guidelines. *Interacting with Computers*, 12 (2), 1999, pp. 81–118.

Vanderdonckt J., A MDA-Compliant Environment for Developing User Interfaces of Information Systems. *In Proceedings of the 17th Conference on Advanced Information System Engineerings, CAiSE 2005*, Porto, Portugal, 2005, pp. 16-31.

Vanderdonckt J., Model-Driven Engineering of User Interfaces: Promises, Successes, and Failures. *In proceedings of 5th Annuual Romanian Conference on Human-Computer Interaction*, ROCHI'2008, Iasi, Roumanie, 2008, pp. 1-10.

Van Setten M., *Personalized Information Systems*. Giga CE project part of Gigaport Project, Telematica Instituut, Netherlands, 2001.

Van Setten M., Moelaert-El Hahidy F., Collaborative Search and Retrieval. *Report n° TI/RS/2000/050*, Telematica Instituut, Pays Bas, 2000.

Vesanen J., What is Personalization: A Literature Review And Framework. Helsinki School “Of Economics, *Working Papers, W-391*, 2005.

Vergara, H., PROTUM: A Prolog Based Tool for User Modeling. WG Knowledge-Based Information Systems, Department of Information Science, University of Konstanz, Allemagne, *WIS-Report 10*, 1994.

Verplanken B., Aarts H., van Knippenberg A., Habit, information acquisition, and the process of making travel mode choices. *European Journal of Social Psychology*, Vol. 27, N° 5, 1997, pp. 539-560.

Vickoff J.P. *Estimation et architecture des développements agiles*. Editions Hermes-Lavoisier, Paris, 2005.

Virvou M., Kabassi K., IFM: An Intelligent Graphical User Interface Offering Advice. *In Proceedings of 2nd Hellenic Conference on Artificial Intelligence*, Thessalonique, Grèce, 2002, pp. 155–164.

Xu Z., Zhang S., Dong Y., Mapping between Relational Database Schema and OWL Ontology for Deep Annotation. *In Proceedings of the 2006 IEEE/WIC/ACM international Conference on Web intelligence*, Hong Kong, Chine, 2006.

Waern A., Tierney M., Rudström A., Laaksolahti J., ConCall: Edited and Adaptive Information Filtering. *In Proceedings of 1999 International Conference on Intelligent User Interfaces, IUI'99,* Los Angeles, USA, 1999.

Wang J., Ding Z., Jiang C., An Ontology-based Public Transport Query System. *In Proceedings of First International Conference on Semantics Knowledge and Grid, SKG'05*, Beijing, Chine, 2005.

Watson A., *OMG's new modelling specifications***.** First European Conference on Model Driven Architecture – Foundation and Application, ECMDA 2005, Nuremberg, Allemagne, 2005.

W3C (a), *OWL Web Ontology Langage Overview*. W3C Recommendation, 2004.

W3C (b) , *RDF Primer*. W3C Recommendation, 2004.

Weiser M., The Computer for the Twenty-First Century. *Scientific American,* Septembre, 1991, pp. 94-104.

Wells K., Horan, T., Toward a consumer demand-driven intelligent transportation system policy: findings from Southern California, *Transportation Research Record,* 1679, USA, 1999, pp. 64-72.

WfMC, Terminology & Glossary. Workflow Management Coalition, *Document number WFMC-TC-1011*, 1999.

Wiener N., *The Human Use of Human Being*. Cybernetic and Society, Houghton Mifflin Company, 1950.

Annexe A : Le moteur de règles utilisé dans PERCOMOM

A.1. Les principes

Les règles métier (OMG, 2005) (modèle BM3) : elles sont un des modèles de support les plus importants dans PERCOMOM, permettent de sortir une partie de la logique métier des processus métier rendant ceux-ci plus faciles à modifier. Une définition des règles métier est donnée par (The Business Rules Group, 2000) :

> "*A business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the business. The business rules that concern the project are atomic - that is, they cannot be broken down further*."

Dans le cadre de notre PERCOMOM, nous avons défini trois types de règles métier :

- les règles dites de validation dont l'évaluation retourne une valeur de type booléen ;
- les règles de sélection dont l'évaluation retourne une valeur qui peut être une chaîne de caractères, une date ou encore un nombre ;
- les règles d'action dont l'évaluation va déclencher l'exécution d'une action particulière.

La définition de ces règles peut utiliser l'ensemble des éléments définis dans les modèles présentés dans le chapitre 3.

A.2. La définition d'une règle métier

A.2.1. Introduction

Dans le cadre de nos travaux, nous avons défini un moteur de règles spécifique qui permet d'évaluer des règles métier. Ce moteur de règles, qui est défini dans cette annexe, travaille sur trois types d'éléments que sont les clauses, les expressions et les règles.

- Une clause est l'élément de base du moteur d'évaluation de règles, une clause est définie sur la base de l'existence d'un membre droit, d'un comparateur et d'un membre gauche. Une clause ne peut retourner que deux valeurs : Vrai ou Faux. De manière pratique, la définition d'une clause passe par la définition d'un certain nombre de propriétés qui sont décrites dans la section A.2.2 de cette annexe.
- Une expression est le deuxième élément utilisé dans le cadre du moteur de règles de PERCOMOM. Une expression est composée d'un ensemble de clauses ou d'expressions, appelés de manière générique des contenus, qui sont liées entres-elles à l'aide d'un opérateur logiques de type ET ou de type OU. L'évaluation d'une expression ne peut retourner qu'une valeur booléenne de type Vrai ou Faux. Les propriétés d'une expression sont décrites dans la section A.2.3 de cette annexe.
- Une règle est le troisième et dernier élément utilisé dans le cadre du moteur de règles de PERCOMOM. De manière pratique, une règle permet, en fonction de son type, après l'évaluation d'une expression de retourner un booléen pour une règle de validation, un nom de modèle d'action (BM1) à exécuter pour une règle d'action ou une valeur pour une règle de sélection. Afin de permettre, l'utilisation du principe des sélections imbriquées, une règle peut être chaînée vers une autre règle dans le cas où l'évaluation du résultat de l'expression retourner un résultat faux. Ceci permet de créer une évaluation de règles en cascade permettant, par exemple, de pouvoir retourner plus de deux valeurs différentes lors de l'évaluation d'une règle de sélection ou lors de l'évaluation d'une règle d'action. L'ensemble des propriétés associées à une règle dans PERCOMOM sont décrites dans la section A.2.4 de cette annexe.

Un exemple de règle métier de type validation permettant de savoir, en fonction d'une information sur l'âge stockée dans son profil, si l'utilisateur a plus de 60 ans est donné ci-dessous :

```
<!-- Une clause est l'élément de base du moteur de règle -->
<clause id="789" nom="Age_superieur_a_60_ans">
    <element type="profil" nom="age"/>
    <comparateur type=">="/>
    <membredroit type="valeurfixe" typevaleur="entier" valeur="60"/>
</clause>

<!-- Une expression représente une association logique de clauses et d'expressions -->
<expression id="12" nom="Expression_age_superieur_a_60_ans">
    <contenu type="clause" id="789"/>
</expression>

<!-- Une règle métier possède un type et est définie à partir d'une expression -->
<regle id="23" nom="Est_Un_Senior" idexpression="12" type="validation"/>
```

A.2.2. La définition d'une clause

Une clause est l'élément de base du moteur d'évaluation de règles, une clause est définie sur la base de l'existence d'un membre droit, d'un comparateur et d'un membre gauche. Une clause ne peut retourner que deux valeurs : Vrai ou Faux. De manière pratique, la définition d'une clause passe par la définition d'un certain nombre de propriétés qui sont décrites dans le tableau A.1.

Tableau A.1. Ensemble des propriétés d'une clause dans le moteur de règles de PERCOMOM

Propriété	Type	Description
Id	Obligatoire	Identifiant de la clause. Cet identifiant est un nombre de type entier qui doit être unique pour l'ensemble des clauses.
Nom	Obligatoire	Nom de la clause. Ce nom doit être unique pour l'ensemble des clauses. Il permet de retrouver une clause à partir de son nom.
estinverse	Optionel	Permet d'indiquer, à l'aide d'un booléen, si le résultat de l'évaluation doit être inversé (correspond au placement d'un élément logique de type NOT devant l'ensemble de la clause). Par défaut, la valeur de la propriété *estinverse* est égale à "faux".
element.type	Obligatoire	Type d'élément qui sera utilisé pour effectuer la comparaison au niveau du membre gauche. Ce type peut être : "profil" pour un élément provenant du profil de l'utilisateur, "ontologie" pour un élément provenant d'un individu de l'ontologie en cours de manipulation au moment de l'évaluation de la règle, ou "variable" pour un élément de type valeur temporaire (une variable).
element.nom	Obligatoire	Permet de définir le nom de l'élément qui sera utilisé pour la comparaison au niveau du membre gauche. Pour un individu de l'ontologie, c'est le nom de la classe qu'on veut utiliser pour effectuer la comparaison.
element.secondnom	Optionnel	Permet de définir, pour un élément de type "ontologie", le nom de la propriété qui sera utilisée dans le cadre de la comparaison au niveau du membre gauche.

Propriété	Type	Description
comparateur	Obligatoire	Permet de définir le comparateur qui sera utilisé pour effectuer la comparaison. Ce comparateur peut être ">", "<", ">=", "<=" et "==" pour tester une égalité entre les deux membres. Lors des comparaisons entre deux éléments de types chaînes de caractères, la comparaison se fait caractère par caractère en utilisant l'ordre alphabétique comme base de comparaison. Ainsi "avion" est inférieur à "bus" et "métro" est supérieur à "météo".
membredroit.type	Obligatoire	Type d'élément qui sera utilisé pour effectuer la comparaison au niveau du membre droit. Ce type peut être : "profil" pour un élément provenant du profil de l'utilisateur, "ontologie" pour un élément provenant d'un individu de l'ontologie en cours de manipulation au moment de l'évaluation de la règle, "variable" pour un élément de type valeur temporaire (une variable) ou "valeurfixe" pour une valeur fixe qui sera définie dans la clause.
membredroit.nom	Optionnel	Permet de définir le nom de l'élément qui sera utilisé au niveau du membre droit pour la comparaison pour les éléments de type "profil" ou "variable". Pour les ontologies, permet de définir le nom de la classe de l'individu qu'on veut effectuer dans le cadre de la comparaison
membredroit.secondnom	Optionnel	Permet de définir, pour un élément de type "ontologie", le nom de la propriété qui sera utilisé dans le cadre de la comparaison eu niveau du membre droit.
membredroit.typevaleur	Optionnel	Permet d'indiquer le type de valeur pour un membre droit de type "valeurfixe". Ce type de valeur peut-être : "entier", "date", "heure", "nombreflottant" ou "chainedecaracteres"
membredroit.valeur	Optionnel	Permet d'indiquer la valeur du membre droit pour un élément de type "valeurfixe".

A.2.3. La définition d'une expression

Une expression est le deuxième élément utilisé dans le cadre du moteur de règles de PERCOMOM. Une expression est composée d'un ensemble de clauses ou d'expressions, appelées de manière générique des contenus, qui sont liées entres elles à l'aide d'un opérateur logiques de type ET ou de type OU. L'évaluation d'une expression ne peut retourner qu'une valeur booléenne de type Vrai ou Faux. Les propriétés d'une expression sont décrites dans le tableau A.2.

Tableau A.2. Ensemble des propriétés d'une expression dans le moteur de règles de PERCOMOM

Propriété	Type	Description
Id	Obligatoire	Identifiant de l'expression. Cet identifiant est un nombre de type entier qui doit être unique pour l'ensemble des expressions.
Nom	Obligatoire	Nom de l'expression. Ce nom doit être unique pour l'ensemble des expressions. Il permet de retrouver une expression à partir de son nom.
contenu.type	Obligatoire	Type de contenu : "clause" pour une clause et "expression" pour une expression. L'utilisation de contenu de type "expression" permet de gérer la notion de parenthèse dans le cadre de l'évaluation globale d'une expression. Ainsi, il est possible de créer une première expression, que nous appellerons expression A, contenant deux clauses reliées par un opérateur logique de type ET et de l'intégrer dans une deuxième expression, que nous appellerons expression B, permettant de l'associer avec une troisième clause à l'aide d'un opérateur logique de type OU. Une expression étant toujours évaluée en premier, lors de l'évaluation de l'expression B, c'est le résultat de l'évaluation de l'expression A qui sera évaluée et qui sera comparée à la troisième clause à l'aide de l'opérateur logique OU pour donner le résultat de l'expression B.
contenu.id	Obligatoire	Permet d'indiquer l'identifiant du contenu (identifiant de la clause ou identifiant de l'expression).
contenu.ordre	Optionnel	Numéro d'ordre du contenu pour l'évaluation de l'expression. Dans une expression, chaque contenu possède un ordre bien précis. Si l'ordre n'est pas indiqué, il est considéré comme étant automatiquement égal à 1. Le numéro d'ordre permet d'ordonner les différents contenus constituant l'expression entre eux ; l'évaluation de l'ensemble des contenus se faisant par numéro d'ordre croissant.
contenu.lienlogique	Optionnel	Lien logique vers le prochain contenu (numéro d'ordre suivant). Ce lien logique peut être soit un "ET" logique soit un "OU" logique. Si le lien logique n'est pas indiqué, on suppose que le lien avec l'élément suivant est un "ET" logique.
contenu.estinverse	Optionnel	Permet d'indiquer, avec un booléen, si le résultat de l'évaluation du contenu doit être inversé (correspond à un élément logique NOT). Par défaut, *estinverse* est égal à "faux".

A.2.4. La définition d'une règle

Une règle est le troisième et dernier élément utilisé dans le cadre du moteur de règles de PERCOMOM. De manière pratique, une règle permet, en fonction de son type, après l'évaluation d'une expression de retourner un booléen pour une règle de validation, un nom de modèle d'action (BM1) à exécuter pour une règle d'action ou une valeur pour une règle de sélection. Afin de permettre l'utilisation du principe des sélections imbriquées, une règle peut être chaînée vers une autre règle dans le cas où l'évaluation du résultat de l'expression retourner un résultat faux. Ceci permet de créer une évaluation de règles en cascade permettant, par exemple, de pouvoir retourner plus de deux valeurs différentes lors de l'évaluation d'une règle de sélection ou lors de l'évaluation d'une règle d'action. L'ensemble des propriétés associées à une règle dans PERCOMOM sont décrites dans le tableau A.3.

Tableau A.3. Ensemble des propriétés d'une règle dans le moteur de règles de PERCOMOM

Propriété	Type	Description
Id	Obligatoire	Identifiant de la règle. Cet identifiant est un nombre de type entier qui doit être unique pour l'ensemble des règles.
nom	Obligatoire	Nom de la règle. Ce nom doit être unique pour l'ensemble des règles. Il permet de retrouver une règle à partir de son nom.
idexpression	Obligatoire	Identifiant de l'expression associée à la règle
type	Obligatoire	Permet d'indiquer le type de règle avec les valeurs "validation" pour une règle de validation, "selection" pour une règle de sélection et "action" pour une règle d'action.
liensifaux	Optionnel	Permet d'indiquer un lien vers une autre règle à évaluer lorsque l'évaluation de l'expression associée à la règle a retournée le booléen "Faux". Par défaut, aucun lien vers une autre règle n'est effectué.
action.true	Optionnel	Pour une règle de type "action", permet d'indiquer le nom de l'action a exécuter si l'évaluation de l'expression retourne le booléen "Vrai"
action.faux	Optionnel	Pour une règle de type "action", permet d'indiquer le nom de l'action a exécuter si l'évaluation de l'expression retourne le booléen "Faux"
selection.vrai	Optionnel	Pour une règle de type "selection", permet d'indiquer la valeur à retourner si l'évaluation de l'expression retourne le booléen "Vrai"
selection.faux	Optionnel	Pour une règle de type "selection", permet d'indiquer la valeur à retourner si l'évaluation de l'expression retourne le booléen "Faux"

A.3. La définition des domaines de validité dans PERCOMOM

A.3.1. Introduction générale

Au niveau de la prise en compte du contexte, nous utilisons, dans le cadre de notre approche, des règles métier, de type validation, un peu particulières que sont les domaines de validité. De manière simple, un domaine de validité est composé d'un ensemble de restrictions qui sont des inclusions ou des exclusions sur des éléments contextuels. Par exemple, si on veut définir qu'une source de données est valable uniquement pour la ville de Lille, on doit définir un domaine de validité géographique de type inclusion dans lequel on effectuera une comparaison de la localisation actuelle de l'utilisateur par rapport à la ville de Lille. Le fichier de description du domaine de validité associé à cet exemple est donné page suivante.

```
<!—Un domaine de validité est défini par un identifiant unique et par un nom -->
<domainevalidite id="4" nom="geo_dans_la_ville_de_lille">
    <!-- Une restriction géographique est définie par un identifiant, par un nom et par son type -->
    <restrictiongeographique>
        <id valeur="987"/>
        <nom valeur="ville_de_lille"/>
        <type valeur="inclusion"/>
        <typedezone valeur="ontologie_individu"/>
        <definitionzone valeur="Lille"/ >
    </restrictiongeographique>
</domainevalidite>
```

Actuellement, dans PERCOMOM, nous ne prenons en compte, dans le cadre des domaines de validité, que trois types d'informations de contexte : l'espace géographique, le temps et le domaine applicatif. Ces trois types de restrictions, qui peuvent être assemblées pour former un domaine de validité, sont gérés de manière séparée à travers des restrictions géographiques, des restrictions temporelles et des restrictions applicatives qui sont présentées en détail dans les sections suivantes.

A.3.2. Les restrictions géographiques

Dans PERCOMOM, une restriction géographique est définie par un type de zone géographique sur laquelle l'évaluation va être effectuée de deux manières différentes : de manière inclusive permettant d'évaluer si l'utilisateur se trouve bien dans la zone et de manière exclusive permettant d'évaluer si l'utilisateur ne se trouve pas dans la zone géographique. Ces différentes zones géographiques peuvent être :

- Un individu de l'ontologie géographique (CM1) et dans ce cas, lors de l'évaluation de la restriction géographique, on comparera l'endroit où se trouve l'utilisateur à l'individu de l'ontologie. Par exemple, si on suppose que "Lille" est un individu de la classe "Ville" de l'ontologie géographique (CM1) et qu'on crée une restriction géographique sur cet individu alors le résultat de l'évaluation des restrictions géographiques ne retournera la valeur "Vrai" que si l'utilisateur se trouve physiquement dans la ville de Lille.
- Une classe de l'ontologie géographique (CM1) et dans ce cas, lors de l'évaluation de la restriction géographique, on comparera l'endroit où se trouve l'utilisateur par rapport à la classe de l'ontologie. Ainsi, si on décide de créer une restriction géographique sur la classe de l'ontologie géographique "Ville", l'évaluation de l'ontologie ne retournera la valeur "Vrai" que si l'utilisateur se trouve dans une ville au moment de l'évaluation et ceci quel que soit la ville (Lille, Paris, Londres, …).
- Un objet géométrique au format WKT (Well-Known Text), format utilisé dans les Systèmes d'Information Géographique (GIS), et dans ce cas, au moment de l'évaluation de la restriction géographique, la valeur "Vrai" n'est retournée que si l'utilisateur se trouve physiquement dans la zone géographique définie par l'objet géométrique au format WKT.

Les propriétés d'une restriction géographique sont données dans le tableau A.4.

Tableau A.4. Ensemble des propriétés d'une restriction géographique dans le moteur de règles de PERCOMOM

Propriété	Type	Description
Id	Obligatoire	Identifiant de la restriction géographique. Cet identifiant doit être un nombre entier qui doit être unique pour l'ensemble des restrictions géographiques.
Nom	Obligatoire	Nom de la restriction géographique. Ce nom doit être unique pour l'ensemble des restrictions géographiques. Il permet de retrouver une restriction géographique à partir de son nom.
Type	Obligatoire	Permet d'indiquer le type de restriction géographique avec les valeurs "inclusion" si l'évaluation de la restriction géographique doit retourner Vrai si l'utilisateur se trouve dans la zone géographique considérée, ou "exclusion" si l'évaluation de la restriction géographique doit retourner Vrai si l'utilisateur ne se trouve pas dans la zone géographique considérée.
Typedezone	Obligatoire	Permet d'indiquer le type de zone géographique à considérer. Peut prendre trois valeurs : "ontologieindividu" pour un individu provenant de l'ontologie géographique (ex : Lille, Paris, France, etc.), "ontologieclasse" pour un type de classe provenant de l'ontologie géographique (ex: Gare, Ville, Pays, etc.), "objetgeometrique" si l'élément géographique doit être défini sous la forme d'un fichier de définition d'un objet géométrique (format wkt (Well-Known Text) utilisé dans les Systèmes d'Information Geographique (SIG)).
definitionzone	Obligatoire	Permet d'indiquer le nom d'un individu de l'ontologie, pour un type de zone "ontologieindividu". Permet d'indiquer le nom de la classe de l'ontologie, pour un type de zone "ontologieclasse". Permet de définir l'objet géométrique pour un type de zone "objetgeometrique"

A.3.3. Les restrictions temporelles

Dans PERCOMOM, une restriction temporelle permet de définir des périodes de temps qui peuvent être fixes ou répétitives. Avant de présenter les restrictions temporelles en détail, il est important de noter qu'une restriction temporelle est toujours évaluée par rapport à l'heure locale de l'endroit où se trouve l'utilisateur. Sinon, une restriction temporelle possède toujours une date et une heure de début de validité et une date et une heure de fin de validité permettant d'indiquer à partir de quand et jusqu'à quand la restriction temporelle pourra être évaluée. En dehors de cette période, le résultat de l'évaluation temporelle retournera systématiquement la valeur "Faux". Sinon, une restriction temporelle peut porter :

- Sur un "jour" et, dans ce cas, la restriction temporelle retournera la valeur "Vrai" tous les jours pendant la tranche horaire de validité qui a été définie (la tranche horaire de validité est définie par une heure de début et une heure de fin).
- Sur une "semaine" et, dans ce cas, la restriction temporelle retournera la valeur "Vrai" uniquement les jours de semaine définis explicitement (exemple : le jeudi et le vendredi) et ceci uniquement pendant la tranche horaire de validité définie.
- Sur un "mois" et, dans ce cas, la restriction temporelle retournera la valeur "Vrai" uniquement pendant une tranche de jours bien précis et ceci uniquement pendant la tranche horaire de validité définie. Ainsi, il est possible de définir qu'une restriction temporelle est valide entre le 10 et le 15 de chaque mois et ceci de 14H00 à 16H00. Si des jours de semaine sont définis et si une périodicité au niveau des jours est indiquée, la restriction temporelle retournera "Vrai" que lorsque la périodicité de ce jour sera valide dans le mois. Par exemple, si on indique que la restriction temporelle est valable le mercredi et le jeudi et si on indique que la périodicité au niveau des jours est égale à 1 alors l'évaluation de la restriction temporelle retournera la valeur "Vrai" tous les premiers mercredi et jeudi de chaque mois et ceci uniquement pendant la tranche horaire de validité définie.
- Sur un "trimestre" et dans ce cas, on fonctionnera de la même façon que pour le "mois" sauf qu'il faudra préciser en plus les mois de validité pour le trimestre. Pour donner un exemple, il est possible de définir qu'une restriction temporelle n'est valable que les 5 premiers jours du premier mois de chaque trimestre (un trimestre démarrant le 1^{ér} janvier, le 1er avril, le 1er juillet ou le 1er octobre).
- Sur un "semestre" et dans ce cas, on fonctionnera de la même façon que pour le "trimestre". Ainsi, il sera possible de définir une restriction temporelle qui ne soit valable que pour le 15ème jour du dernier mois de chaque trimestre (un trimestre démarrant le 1er janvier ou le 1er juillet).
- Sur une "année" et dans ce cas, on fonctionnera de la même façon que pour le "trimestre". Ainsi, il sera possible de définir une restriction temporelle qui ne soit valable que le 1er jour du mois de janvier de chaque année ou alors que le 5ème mercredi de chaque année.

L'ensemble des propriétés associées à une restriction temporelle sont données dans le tableau A.5.

Voici un exemple de définition d'une restriction temporelle qui ne retourne la valeur "Vrai" que le premier jour de chaque mois :

```
<restrictiontemporelle>
    <id valeur="11"/>
    <nom valeur="premierjourdumois"/>
   <type valeur="inclusion"/>
   <typeperiode valeur="mois"/>
   <periodejourdebut valeur="1"/>
   <periodejourfin valeur="1"/>
</restrictiontemporelle>
```

Tableau A.5. Ensemble des propriétés d'une restriction temporelle dans le moteur de règles de PERCOMOM

Propriété	Type	Description
Id	Obligatoire	Identifiant de la restriction temporelle. Cet identifiant doit être un nombre entier qui doit être unique pour l'ensemble des restrictions temporelles.
nom	Obligatoire	Nom de la restriction temporelle. Ce nom doit être unique pour l'ensemble des restrictions temporelles. Il permet de retrouver une restriction temporelle à partir de son nom.
type	Obligatoire	Permet d'indiquer le type de restriction temporelle avec les valeurs "inclusion" si l'évaluation de la restriction temporelle doit retourner vrai si l'heure locale de l'utilisateur se trouve dans la tranche horaire définie par la restriction temporelle, "exclusion" si l'évaluation de la restriction temporelle doit retourner vrai dans le cas contraire.
datedebut	Optionnel	Date de début de validité de la restriction temporelle
datefin	Optionnel	Date de fin de validité de la restriction temporelle
heuredebut	Optionnel	Heure de début de validité de la restriction temporelle
heurefin	Optionnel	Heure de fin de validité de la restriction temporelle
typeperiode	Obligatoire	Définition du type de période qui peut prendre six valeurs : "jour" pour une restriction temporelle valable chaque jour. "semaine" pour une restriction temporelle valable chaque semaine. "mois" pour une restriction temporelle valable chaque mois. "trimestre" pour une restriction temporelle valable chaque trimestre. "semestre" pour une restriction temporelle valable chaque semestre. "année" pour une restriction géographique valable chaque année.
periodejour	Optionnel	Permet d'indiquer, lorsqu'on a indiqué des jours de la semaine dans la propriété "*SemaineJour*" la périodicité à utiliser lorsque le type de période n'est pas égal à "jour" ou à "semaine". Ainsi, si la périodicité est égale à 2 et si la propriété "*SemaineJour*" indique uniquement le mercredi, alors la restriction temporelle ne sera valide que le deuxième mercredi de chaque mois.
periodejourdebut	Optionnel	Permet d'indiquer, pour les périodes autres que "jour", le jour précis de la période à considérer pour le début de la restriction temporelle.
periodemoisdebut	Optionnel	Permet d'indiquer pour les périodes autres que "jour", "semaine" et "mois", le mois précis de la période à considérer pour le début de la restriction temporelle.
periodejourfin	Optionnel	Permet d'indiquer, pour les périodes autres que "jour", le jour précis de la période à considérer pour la fin de la restriction temporelle.
periodemoisfin	Optionnel	Permet d'indiquer pour les périodes autres que "jour", "semaine" et "mois", le mois précis de la période à considérer pour la fin de la restriction temporelle.
semainejour	Optionnel	Permet d'indiquer le jour de la semaine sous la forme d'une chaine de caractères ou chaque jour est défini à l'aide de ses deux premières lettres (exemple : "JE" pour jeudi ou "SA" pour samedi.
periodeheuredebut	Optionnel	Permet d'indiquer l'heure de début de validité pour la période associée à la restriction temporelle. Si celle-ci n'est pas indiquée, on considère que l'heure de début est égale à "00H00".
periodeheurefin	Optionnel	Permet d'indiquer l'heure de fin de validité pour la période associée à la restriction temporelle. Si celle-ci n'est pas indiquée, on considère que l'heure de fin est égale à "23H59".

A.3.4. Les restrictions applicatives

Dans PERCOMOM, une restriction applicative permet de définir une restriction portant sur une application ou sur un type d'application. La valeur retournée par l'évaluation de la restriction est égale à "Vrai" si l'application en cours est égale à la liste des applications ou aux catégories d'applications définies dans la restriction applicative sinon, la valeur rétournée est égale à "Faux". De manière générale, dans PERCOMOM, chaque application porte un nom unique et est associée à une catégorie ou plusieurs catégories d'applications. Ces différentes catégories d'applications sont définies dans une ontologie d'applications permettant de définir une hiérarchie au niveau des différentes catégories utilisées. Un exemple simple d'ontologie de catégorisation des applications est donné à la figure A-1.

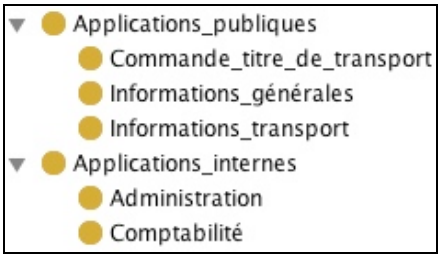


Figure A-1. Exemple d'ontologie de catégorisation des applications

L'ensemble des propriétés associées à une restriction applicative sont données dans le tableau A.6.

Tableau A.6. Ensemble des propriétés d'une restriction temporelle dans le moteur de règles de PERCOMOM

Propriété	Type	Description
id	Obligatoire	Identifiant de la restriction applicative. Cet identifiant doit être un nombre entier qui doit être unique pour l'ensemble des restrictions applicatives.
nom	Obligatoire	Nom de la restriction applicative. Ce nom doit être unique pour l'ensemble des restrictions applicatives. Il permet de retrouver une restriction applicative à partir de son nom.
applicationnom	Optionnel	Nom de l'application à laquelle la restriction applicative s'applique. Remarque : il est possible d'indiquer plusieurs noms d'applications si la restriction doit être valable pour plusieurs applications différentes.
applicationcategorie	Optionnel	Nom de la catégorie d'applications à laquelle la restriction applicative s'applique. Remarque : il est possible d'indiquer plusieurs catégories si la restriction doit être valable pour plusieurs catégories d'applications différentes.

Exemple de définition d'une restriction applicative qui ne retournera la valeur "Vrai" que si l'application en cours est associée à la catégorie d'applications "Administration" qui fait elle-même partie de la catégorie d'applications "Applications_internes" comme indiqué dans l'exemple d'ontologie de catégorisation des applications de la figure A-1.

```
<restrictionapplicative>
    < id valeur="12"/>
    <nom valeur="applicationdadministration"/>
    <applicationcategorie valeur="Administration"/>
<restrictionapplicative/>
```

A.3.5. Les domaines de validité

Au niveau global, un domaine de validité est composé d'un ensemble de restrictions géographiques de restrictions temporelles et de restrictions applicatives qui sont reliées entre elles par l'opérateur logique ET. Lors de son évaluation, un domaine de validité est vu comme étant une règle de validation qui ne peut retourner qu'une valeur booléenne (règle de type validation). Les différentes propriétés d'un domaine de validité sont données dans le tableau A.7

Tableau A.7. Ensemble des propriétés d'un domaine de validité dans le moteur de règles de PERCOMOM

Propriété	Type	Description
id	Obligatoire	Identifiant du domaine de validité. Cet identifiant doit être un nombre entier qui doit être unique pour l'ensemble des domaines de validité.
Nom	Obligatoire	Nom du domaine de validité. Ce nom doit être unique pour l'ensemble des domaines de validité. Il permet de retrouver un domaine de validité par son nom.
restrictiongeographique	Optionnel	Permet d'indiquer la liste des restrictions géographiques faisant partie du domaine de validité. Dans le cadre de PERCOMOM, les restrictions géographiques sont reliées entre elles par un opérateur logique de type ET.
restrictiontemporelle	Optionnel	Permet d'indiquer la liste des restrictions temporelles faisant partie du domaine de validité. Dans le cadre de PERCOMOM, les restrictions temporelles sont reliées entre elles par un opérateur logique de type ET.
restrictionaplicative	Optionnel	Permet d'indiquer la liste des restrictions applicatives faisant partie du domaine de validité. Dans le cadre de PERCOMOM, les restrictions applicatives sont reliées entre elles par un opérateur logique de type ET.

A.4. Les moteurs associés à l'évaluation des règles métier et des domaines de validité

A.4.1. Le moteur d'évaluation des règles métier

Pour rappel, une règle métier peut être de tris types différents : une règle de validation, une régle de sélection ou une règle d'action.

Le moteur d'évaluation des règles métier fonctionne sur le modèle suivant :

- **Etape 1 :** On commence par rechercher la définition de la règle métier à partir de son nom ou de son identifiant. Si on ne trouve pas de domaine de validité pour le nom ou l'identifiant fourni, on retourne une erreur. Sinon, on passe à l'étape suivante.
- **Etape 2 :** On récupère la définition de l'expression associée à la règle métier. Si celle-ci n'existe pas, on retourne une erreur. Dans le cas contraire, l'expression en cours devient égale à l'expression définie au niveau de la règle. On se positionne ensuite sur le premier élément de l'expression en cours (qui peut être une expression ou une clause) et on passe à l'étape 3.
- **Etape 3 :** On évalue le membre courant de l'expression en cours. Si celui-ci est une expression, l'expression en cours devient égale à cette expression, on se positionne sur le premier élément de l'expression en cours et on repasse à l'étape 3 pour faire l'évaluation de l'expression en cours. Si c'est une clause, on passe à l'étape 4.

- **Etape 4 :** On évalue la clause en cours (pour rappel, une clause ne peut retourner que la valeur Vrai ou la valeur Faux). Si la clause est définie comme étant inversée, on inverse le résultat obtenu de l'évaluation (Vrai devient Faux et Faux devient Vrai). Deux cas possibles ensuite :
    - Si la clause n'est pas précédée d'un autre élément dans l'expression en cours, la valeur de l'évaluation de l'expression en cours devient égale à la valeur de l'expression et on passe à l'étape 5.
    - Si la clause était précédée d'un ou de plusieurs autres éléments (clauses ou expressions) dans l'expression en cours, on calcule la nouvelle valeur de l'évaluation de l'expression en cours comme étant égale à l'évaluation logique de la dernière évaluation de l'expression en cours avec la valeur de l'évaluation de la clause ; le lien entre les deux éléments étant réalisé à l'aide de l'opérateur logique associé au membre précédent de l'expression en cours. On passe ensuite à l'étape 5.
- **Etape 5 :** On regarde si l'expression en cours possède encore un membre non évalué :
    - Si oui, on se positionne sur ce membre qui devient le membre courant de l'expression en cours et on repasse à l'étape 3.
    - Si non, on passe à l'étape 6
- **Etape 6 :** On remonte d'un niveau dans la hiérarchie des expressions définie à travers la règle métier et l'expression en cours devient égale à l'expression sur laquelle on vient de se positionner.
    - Si on se trouve sur le premier niveau ; c'est-à-dire au niveau de l'expression associée à la règle métier alors on passe à l'étape 7.
    - Sinon, on se positionne sur le membre suivant de l'expression de niveau supérieur. S'il n'y a pas de membre suivant, on repassse à l'étape 6. Sinon, on évalue la valeur de l'évaluation de l'expression en cours comme étant égale à l'évaluation logique de la dernière valeur de l'expression en cours avec le résultat logique de l'évaluation qu'on vient d'effectuer ; le lien entre les deux éléments étant réalisés à l'aide de l'opérateur logique associé au membre précédent de l'expression en cours. On passe ensuite à l'étape 5.
- **Etape 7 :** Deux cas :
    - Si on ne se trouve pas sur le dernier élément de l'expression en cours alors on évalue la valeur de l'évaluation de l'expression en cours comme étant égale à l'évaluation logique de la dernière valeur de l'expression en cours avec le résultat logique de l'évaluation qu'on vient d'effectuer ; le lien entre les deux éléments étant réalisé à l'aide de l'opérateur logique associé au membre précédent de l'expression en cours. On passe ensuite à l'étape 5.
    - Si on se trouve sur le dernier élément de l'expression en cours (qui est l'expression associée à la règle métier), on retourne la valeur de l'évaluation de l'expression en cours comme valeur de l'évaluation de la règle. On passe ensuite à l'étape 8.
- **Etape 8 :** Si la règle retourne la valeur Faux et qu'elle possède un lien vers une autre règle alors la valeur de la règle devient égale à l'évaluation de la valeur de l'autre règle.

Le schéma global de l'algorithme associé au moteur d'évaluation de règles métier est donné à titre indicatif dans la figure A.2 ; l'étape 9 n'étant pas représentée.

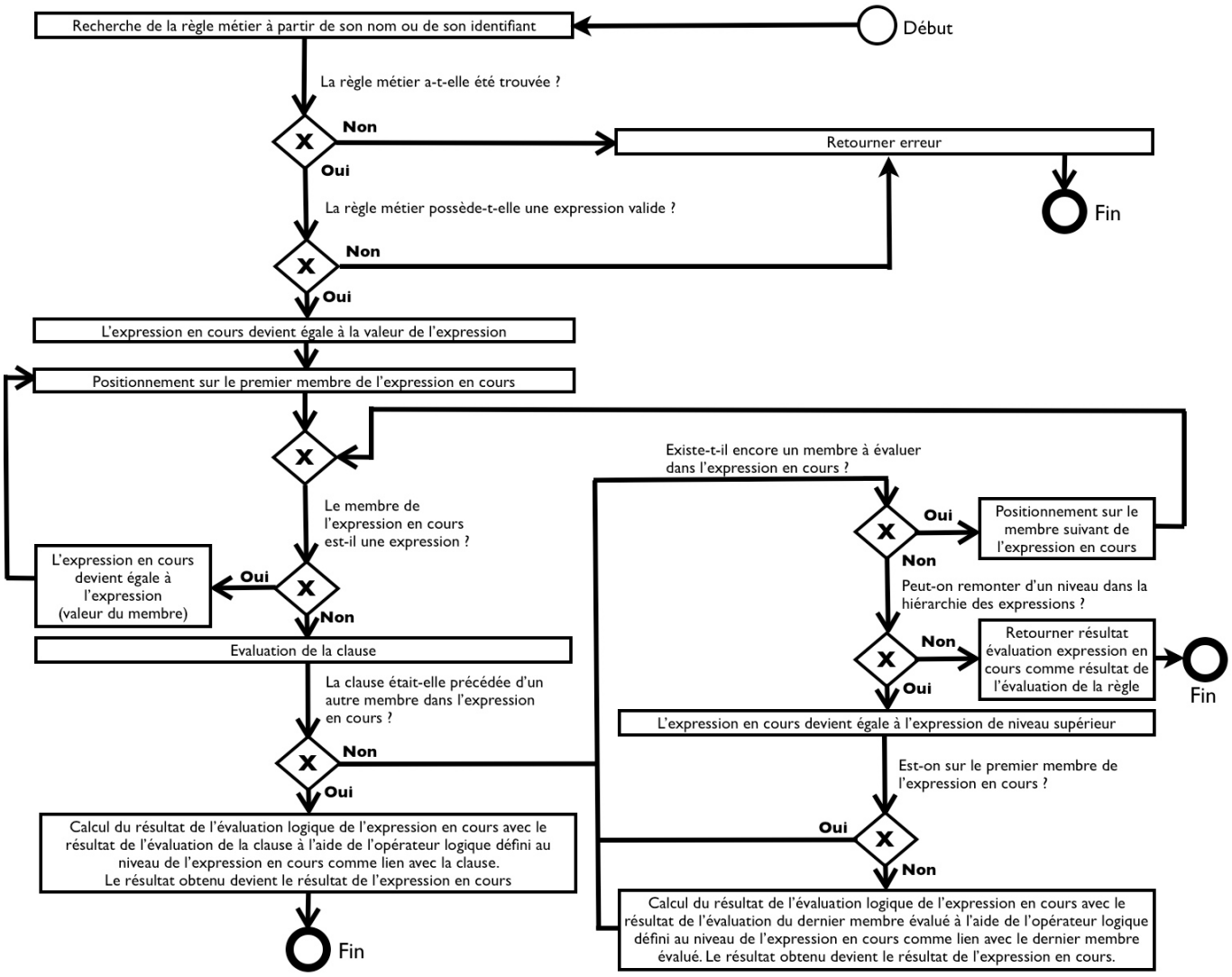


Figure A.2. Schéma global de l'algorithme du moteur d'évaluation des règles métier

A.4.1. Le moteur d'évaluation des domaines de validité

Pour rappel, un domaine de validité est une règle de validation particulière. A ce titre, elle ne peut retourner comme résultat de son évaluation que deux valeurs : Vrai ou Faux.

Le moteur d'évaluation des domaines de validité fonctionne sur le modèle suivant :

- **Etape 1 :** On commence par rechercher le contenu du domaine de validité à partir de son nom ou de son identifiant. Si on ne trouve pas de domaine de validité pour le nom ou l'identifiant fourni, on retourne la valeur Faux. Sinon, on passe à l'étape suivante.
- **Etape 2 :** Si le domaine de validité possède des restrictions géographiques, on les évalue sinon, on passe à l'étape suivante. Chaque restriction géographique est évaluée de manière indépendante et dès que l'évaluation d'une restriction géographique retourne la valeur Faux, on met fin à l'évaluation du domaine de validité et on retourne la valeur Faux comme résultat de cette évaluation. Si chaque restriction géographique a été évaluée comme étant Vrai, on passe à l'étape suivante.
- **Etape 3 :** Si le domaine de validité possède des restrictions temporelles, on les évalue sinon, on passe à l'étape suivante. Chaque restriction temporelle est évaluée de manière indépendante et dès que l'évaluation d'une restriction temporelle retourne la valeur Faux, on met fin à l'évaluation du domaine de validité et on retourne la valeur Faux comme résultat de cette évaluation. Si chaque restriction temporelle a été évaluée comme étant Vrai, on passe à l'étape suivante.
- **Etape 4 :** Si le domaine de validité possède des restrictions applicatives, on les évalue, sinon on retourne Vrai comme résultat de l'évaluation du domaine de validité. Chaque restriction applicative est évaluée de manière indépendante et dès que l'évaluation d'une restriction applicative retourne la valeur Faux, on met fin à l'évaluation du domaine de validité et on retourne la valeur Faux comme résultat de cette évaluation sinon, on retourne la valeur Vrai.

Le schéma global de l'algorithme associé au moteur d'évaluation des domaines de validité est donné dans la figure A.3.

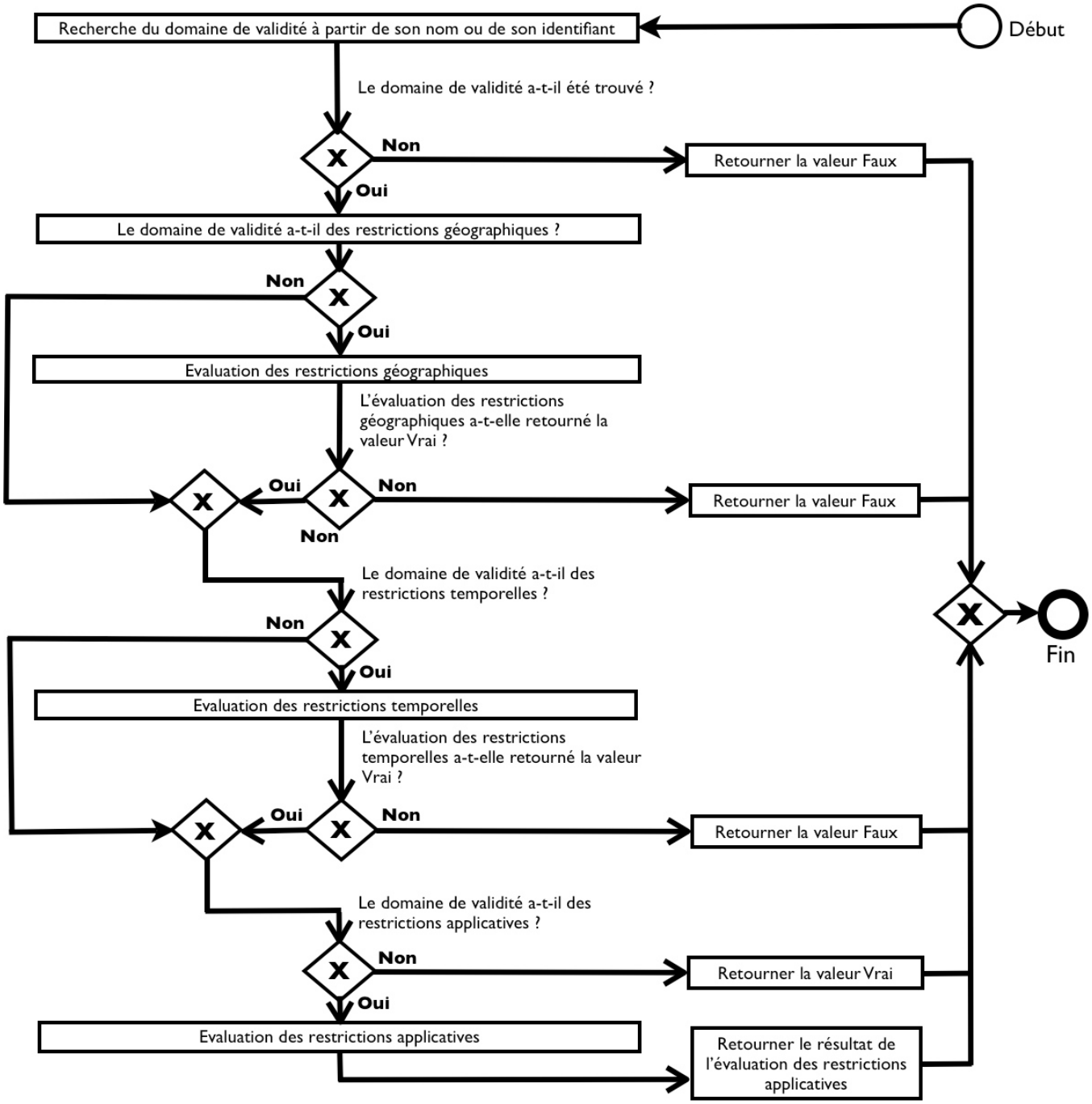


Figure A.3. Schéma global de l'algorithme du moteur d'évaluation des domaines de validité

Annexe B : Quelques éléments utilisés dans le cadre du formalisme de modélisation de PERCOMOM

B.1. Les propriétés associées à un événement (modèle EM3)

L'ensemble des propriétés associées à un événement est indiqué dans le tableau B.1.

Tableau B.1. Ensemble des propriétés associées à un événement (modèle EM3)

Propriété	Type	Description
id	Obligatoire	Identifiant de l'événement. Cet identifiant doit être un nombre entier qui doit être unique pour l'ensemble des événements.
nom	Obligatoire	Nom de l'événement. Ce nom doit être unique pour l'ensemble des événements. Il permet de retrouver un événement à partir de son nom.
domainevalidité	Optionnel	Permet d'indiquer le domaine de validité (géographique, temporel et applicatif) associé à l'application (cf. chapitre 3, §3.1.3.5 pour une description complète des domaines de validité). Par défaut, un événement est valable pour toutes les applications sans aucune restriction ni au niveau géographique ni au niveau temporel
dureedevie	Optionnel	Permet de définir la durée de vie de l'événement. Cette propriété peut prendre trois valeurs : • *PremierTraitement* : l'événement sera rendu inactif dès qu'il aura été traité dans une application et ceci quel que soit l'utilisateur et l'application qui va le traiter. • *DuréeLimitée* : l'événement sera rendu inactif après une durée en minutes déterminée par la valeur de la propriété *dureedevieenminutes.* • *Toujours* : l'événement reste actif tant qu'il n'est pas invalidé par une application (à travers une action (modèle BM1)). Par défaut, un événement est valide tant qu'il n'est pas invalidé par une action dans une application.
dureedevieenminutes	Optionnel	Permet de définir la durée de vie, en minutes, d'un événement dans le cas où il est du type *DuréeLimitée*. À la fin de cette durée, l'événement sera automatiquement rendu inactif.

B.2. Description des principaux éléments statiques d'interaction utilisés dans PERCOMOM au niveau des modèles statiques d'interaction (IM2)

La description des principaux éléments statiques d'interaction utilisés dans PERCOMOM est fournie dans le tableau B.2.

Tableau B.2. Description des principaux éléments statiques d'interaction dans PERCOMOM au niveau CIM (modèle IM2)

<table>
<tr><th>Élément d'interaction</th><th>Description</th></tr>
<tr><td>UIApplication</td><td>Un élément de type UIApplication caractérise, du point de vue des interactions, une application interactive.
Cet élément est présenté en détail dans le chapitre 3, §3.1.4.</td></tr>
<tr><td>UIGroup</td><td>Un élément de type UIGroup permet de regrouper des UIUnit ou des UIUnitSuite en groupes logiques du point de vue métier.
Cet élément est présenté en détail dans le chapitre 3, §3.1.4.</td></tr>
<tr><td>UIUnit</td><td>Un élément de type UIUnit permet de regrouper des éléments d'interactions en groupes logiques du point de vue des interactions ; l'ensemble des éléments d'interaction constitutifs de l'UIUnit ne pouvant être séparés les uns des autres du point de vue de la logique d'interaction.
Cet élément est présenté en détail dans le chapitre 3, §3.1.4.</td></tr>
<tr><td>UIUnitSuite</td><td>Un élément de type UIUnitSuite correspond à un ensemble d'UIUnit qui sont reliés les uns aux autres de manière logique et qui peuvent être utilisés au sein d'un UIGroup où un seul UIUnit contenu dans l'UIUnitSuite pourra être affiché à l'écran à un moment donné.</td></tr>
<tr><td>UISubUnit</td><td>Un UISubUnit regroupe un ensemble d'éléments de type UIField qui peuvent être utilisés dans plusieurs UIUnit. De manière pratique, Un UISubUnit correspond à ensemble d'éléments élémentaires d'interaction réutilisable.</td></tr>
<tr><td>UIField</td><td>Un UIField représente, de manière abstraite, un élément physique élémentaire d'interaction comme, par exemple, un champ de saisie, un bouton, un tableau ou encore un menu.</td></tr>
<tr><td>UIFieldData</td><td>Un UIFieldData représente, de manière abstraite, un élément physique d'interaction permettant d'afficher une information provenant d'un ou de plusieurs individus de l'ontologie de domaine (modèle OD) sur un écran.</td></tr>
<tr><td>UIFieldInOut</td><td>Un UIFieldInOut représente un élément physique d'interaction permettant la saisie et l'affichage d'une propriété d'un individu ou de plusieurs individus de l'ontologie de domaine (modèle OD); cette propriété pouvant retourner une valeur unique ou une liste de valeurs.</td></tr>
<tr><td>UIFieldStatic</td><td>Un UIFieldStatic représente un élément physique d'interaction permettant l'affichage d'une information statique non modifiable à l'écran comme, par exemple, un libellé placé juste devant un champ de saisie de type UIFiledInOut. Le contenu d'un UIFieldStatic ne provient pas de l'ontologie de domaine mais est fixé, par l'expert métier, au moment de la création du modèle statique d'interaction.</td></tr>
<tr><td>UIFieldTable</td><td>Un UIFieldTable représente un élément abstrait d'interaction permettant d'afficher des informations, provenant de l'ontologie de domaine, sous forme d'un tableau à l'écran.</td></tr>
<tr><td>UIFieldTableSimple</td><td>Un UIFieldTableSimple représente un élément d'interaction permettant d'afficher un tableau, issu d'une extraction d'informations à partir de l'ontologie de domaine, sous forme d'une liste. Par exemple, un UIFieldTable pourrait être utilisé pour afficher la liste des voyageurs disposant d'un abonnement de travail. Un exemple de ce type de tableau est donné ci-dessous :
<table>
<tr><th>Nom</th><th>Prénom</th><th>Age</th></tr>
<tr><td>Jean</td><td>Valjean</td><td>67</td></tr>
<tr><td>Isaac</td><td>Newton</td><td>44</td></tr>
<tr><td>Albert</td><td>Einstein</td><td>53</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Élément d'interaction</th><th>Description</th></tr>
<tr><td>UIFieldTableDoubleEntry</td><td>Un UIFieldTableDoubleEntry représente un élément d'interaction permettant d'afficher, sous forme d'un tableau à deux dimensions, une extraction d'informations à partir de l'ontologie de domaine. Ainsi, un UIFieldTableDoubleEntry pourrait être utilisé pour afficher suivant l'axe des X l'ensemble des rotations d'un bus sur une ligne déterminée, suivant l'axe des Y l'ensemble des arrêts de bus situés sur la ligne et à l'intersection de X et de Y l'heure de passage effective du bus à un arrêt donné pour une rotation donnée.
<table>
<tr><th></th><th>Rotation 1</th><th>Rotation 2</th><th>Rotation 3</th></tr>
<tr><td>Arrêt Balzac</td><td>08:45</td><td>-</td><td>08:59</td></tr>
<tr><td>Arrêt Hôtel de Ville</td><td>08:50</td><td>09:10</td><td>-</td></tr>
<tr><td>Arrêt Université</td><td>08:59</td><td>-</td><td>09:09</td></tr>
<tr><td>Arrêt Gare centrale</td><td>09:11</td><td>09:28</td><td>10:21</td></tr>
</table>
</td></tr>
<tr><td>UIFieldControl</td><td>Un UIFieldControl représente un élément abstrait d'interaction permettant d'effectuer des interactions avec l'application qui ne concernent ni l'affichage d'informations ni l'affichage de données multimédia.</td></tr>
<tr><td>UIFieldAction</td><td>Un UIFieldAction représente un élément d'interaction permettant de démarrer une action comme, par exemple, un bouton.</td></tr>
<tr><td>UIFieldNavigation</td><td>Un UIFieldNavigation repésente un élément d'interaction permettant de naviguer d'un UIGroup vers un autre UIGroup ou d'un UIUnit vers un autre UIUnit faisant partie du même UIUnitSuite.</td></tr>
<tr><td>UIFieldMenu</td><td>Un UIFieldMenu représente un élément d'interaction de type menu dans lequel l'utilisateur peut sélectionner un élément pour soit démarrer un nouveau processus métier, soit effectuer une action, soit naviguer vers un nouveau UIGroup ou un nouveau UIUnit.</td></tr>
<tr><td>UIFieldMultimedia</td><td>Un UIFieldMultimedia représente un élément abstrait d'interaction permettant de manipuler des éléments d'interaction de type multimédia. Dans le cadre de nos travaux, nous n'avons, pour l'instant, pas cherché à être exhaustif sur les types d'éléments multimédia pouvant être manipulé et ceci aussi bien au niveau de leur nature (musique, vidéo, etc.) qu'au niveau de leur forme à travers les types de fichiers pris en charge.</td></tr>
<tr><td>UIFieldGraphic</td><td>Un UIFieldGraphic représente un élément d'interaction de type graphique dans lequel les éléments affichés à l'écran proviennent soit de manipulations effectuées sur l'élément par l'utilisateur (création d'un dessin) soit d'une action effectuée au sein de l'application (création d'un graphe de valeurs numériques sous forme d'une courbe temporelle par exemple).</td></tr>
<tr><td>UIFieldPicture</td><td>Un UIFieldPicture représente un élément d'interaction permettant d'afficher et de manipuler une image de type JPG, GIF ou BMP.</td></tr>
<tr><td>UIFieldVideo</td><td>Un UIFieldVideo représente un élément d'interaction permettant d'afficher et de manipuler une vidéo de type AVI à l'écran.</td></tr>
<tr><td>UIFieldSound</td><td>Un UIFieldSound représente un élément d'interaction permettant de manipuler un élément sonore de type MP3 à l'écran.</td></tr>
<tr><td>UIFieldGeoMap</td><td>Un UIFieldGeoMap représente un élément d'interaction permettant d'afficher et de manipuler un élément d'interaction de type carte géographique (de type GoogleMap par exemple) à l'écran.</td></tr>
</table>

Annexe C : L'outil de création de modèles de processus métier (IM1) et des modèles statiques d'interaction (IM2) développé dans le cadre de PERCOMOM

B.1. Les propriétés associées à un événement (modèle EM3)

Dans le cadre de nos travaux de recherche, un outil de création de modèles de processus métier (IM1) a été développé ; il est complètement décrit dans (Firas, 2008). Cet outil a pour but :

- De permettre de créer, à l'aide d'un outil graphique, des modèles de processus métier (IM1) et des modèles statiques d'interaction (IM2).
- De permettre une validation de ces modèles lors de leur création.
- De permettre la génération d'un fichier de définition des processus métier au niveau PIM (dans le cas de l'outil, les fichiers générés sont au format BPEL.

Il est important de noter que l'outil n'est pour l'instant pas capable de faire la transformation des modèles statiques d'interaction de niveau CIM vers le niveau PIM.

Dans le cadre de cette annexe, nous ne présenterons pas cet outil en détail. Nous nous contenterons d'en présenter les principales fonctionnalités.

B.2. Quelques copies d'écran de l'outil de modélisation

C.2.1. L'écran d'accueil

Techniquement, l'outil a été développé comme un plugin pour la plateforme de développement Eclipse. Au lancement de notre application, une seule fenêtre s'ouvre contenant le plan de travail (*Workbench*). Il est basé sur un principe de fenêtre unique découpée en sous-fenêtres (les vues et les éditeurs) auxquelles s'ajoutent une barre de menu et une barre d'outils (cf. figure C.1). Dans la suite de cette annexe, nous reprendrons des éléments de la documentation disponible dans (Firas, 2008) pour présenter les différentes fonctionnalités de l'outil.

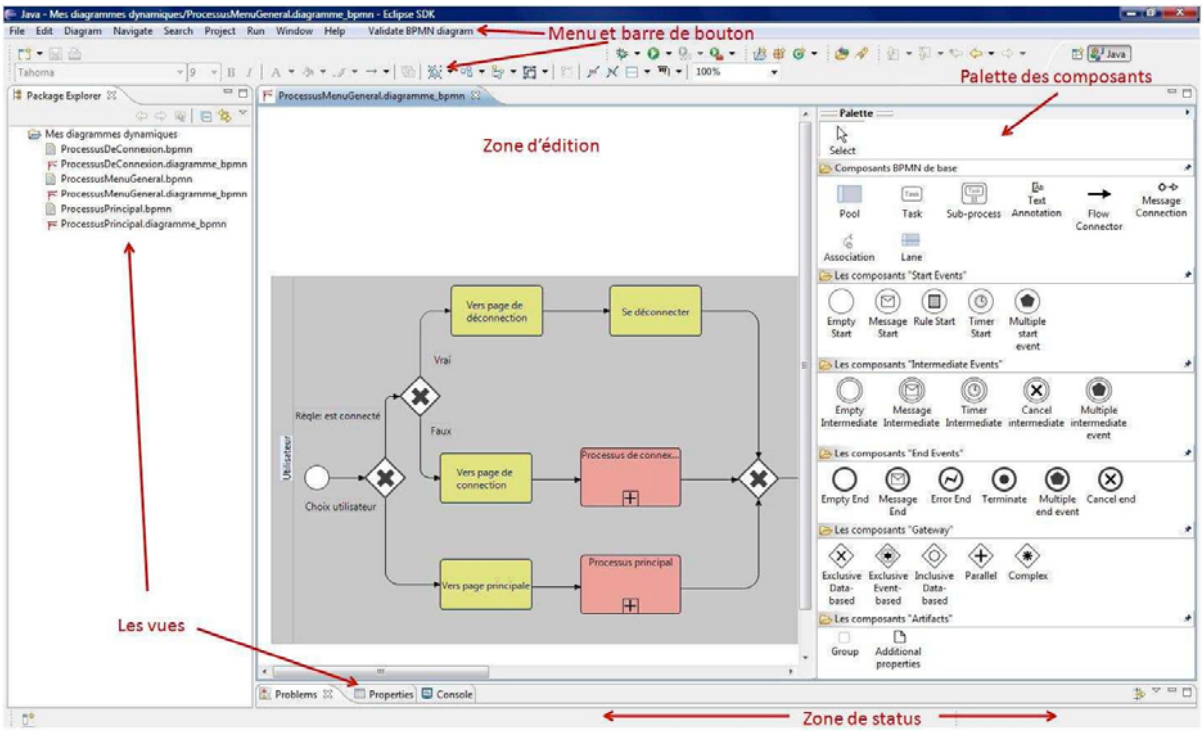


Figure C.1. Vue générale de l'outil graphique de création de modèles de processus métier (IM1) et des modèles statiques d'interaction (IM2)

Les vues

Elles supportent l'interaction entre l'utilisateur de l'application et le *workbench*. Elles fournissent une présentation de tout type d'information dans le *workbench* et permettent de naviguer dans les ressources. L'outil de modélisation dispose de 5 vues différentes :

- une vue pour l'exploration des différents diagrammes réalisés et modélisés et leurs affichage d'une manière hiérarchique et organisée ;
- une vue contenant la palette du dessin dans laquelle se trouve les différents outils et les différents composants nécessaires à la construction d'un modèle et à partir de laquelle il est possible de créer de modèles à l'aide d'opérations de type *drag-and-drop* ;
- une vue intitulée « problems» dans laquelle s'affiche le résultat de validation d'un modèle ;
- une vue nommée « properties » contenant toutes les informations et les attributs d'un composant appartenant à un modèle bien spécifique ;
- une vue appelée « console » et qui sert à afficher les différentes étapes de migration de BPMN vers BPEL et à indiquer les problèmes pouvant éventuellement survenir pendant cette phase de migration.

Les éditeurs

Ils représentent un type de vues spécialement conçu pour visualiser et éditer des données. Dans l'outil, le seul type d'éditeur utilisé est l'éditeur graphique. Il sert à l'affichage des modèles réalisés par l'expert métier. L'application permet à l'expert métier d'ouvrir plusieurs modèles au même temps.

Les barres de menu et d'outils

Elles ont le même rôle que dans la majorité des interfaces utilisateur graphiques. Elles contiennent des commandes usuelles permettant la manipulation des différents composants d'un modèle et la gestion de l'édition des fichiers générés. En plus des outils de mise en forme des composants et des polices utilisées, il existe aussi plusieurs options dans la barre des outils telles que les fonctionnalités d'arrangement automatique des éléments d'un diagramme dessiné ou bien la possibilité de regrouper

des éléments dans un seul élément (i.e. regrouper des composants BPMN sélectionnés dans un sous processus).

C.2.2. La création d'un nouveau projet de modélisation

La création d'un nouveau projet de modélisation est la première tâche qu'effectuera un expert métier dans le processus de modélisation d'une nouvelle application. L'outil dispose d'un *wizard* (petite application ayant pour but de faciliter l'utilisation de l'outil de modélisation) de création des nouveau modèles métier. Ainsi, en allant dans le menu :

"File >New >Examples>Nouveau diagramme BPMN"

ou bien

"File >New >Examples >Nouveau diagramme statique",

l'utilisateur accède, comme l'indique la figure C.2, à la liste des dossiers de l'espace de travail pour en choisir un dans lequel il créera le nouveau modèle de processus métier.

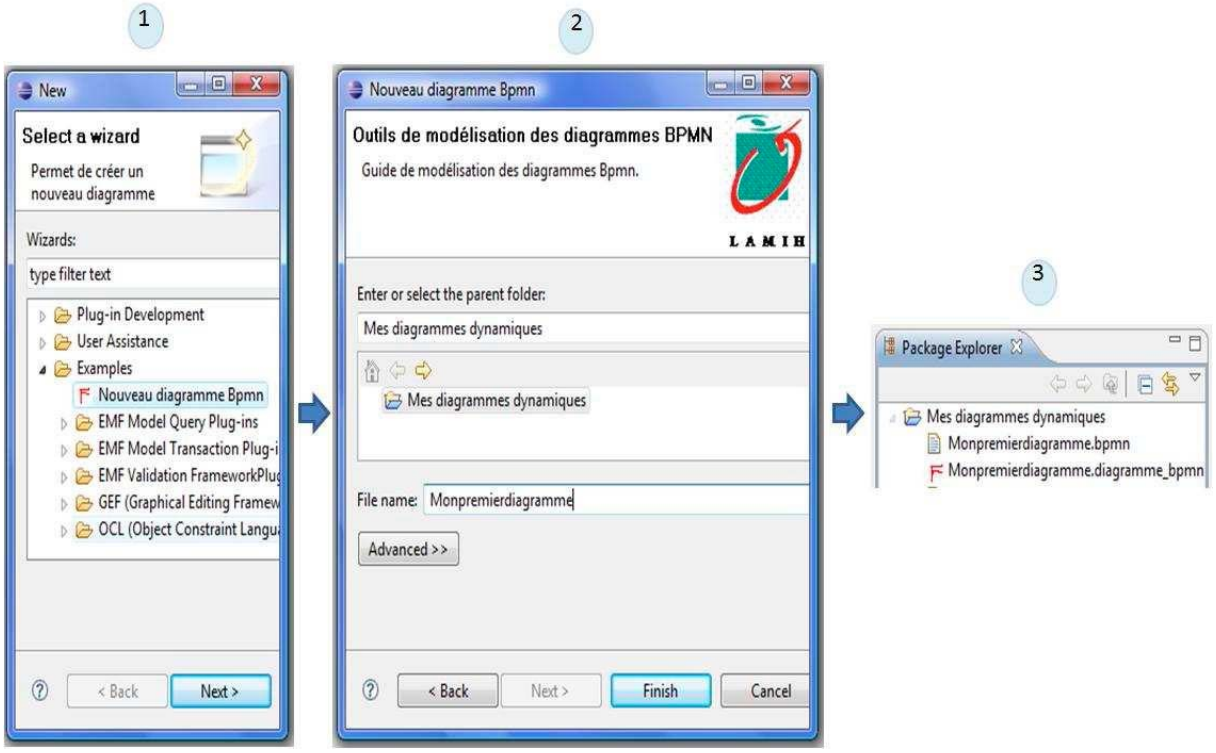


Figure C.2. Les trois étapes de création d'un nouveau modèle de processus métier (IM1) à l'aide du *wizard* créé dans l'outil graphique de création des modèles de processus métier (IM1) et des modèles statiques d'interaction (IM2)

L'utilisation ce *wizard* entraîne la création de deux fichiers XML pour chaque modèle créé :

- Pour le modèle de processus métier (IM1), les fichiers créés portent les extensions *.bpmn* et *.diagramme_bpmn.*
- Pour le modèle statique d'interaction (IM2), les fichiers créés portent les extensions *.statique* et *.diagramme_statique*

Les fichiers qui contiennent le mot *diagramme* représentent l'aspect graphique du modèle i.e. c'est dans ce type de fichier que nous stockons les informations spécifiques à la mise en forme des composants telles que leurs couleurs et leurs tailles. Les autres fichiers contiennent l'ensemble des informations permettant de décrire les relations entre les différents éléments d'un diagramme.

Le fichier .bpmn sera utilisé par la suite comme entrée pour l'outil de migration de BPMN vers BPEL (transformation de modèle du niveau PIM au niveau PSM) et sur lequel l'application va effectuer des modifications afin de produire du code BPEL.

C.2.3. La validation d'un diagramme

La validation d'un diagramme représente une des exigences majeures de la solution proposée. De ce fait, l'outil dispose de deux types de validation afin d'aider l'expert métier à concevoir des modèles conformes aux spécifications demandées.

Le premier type de validation est effectué en "temps réel" lors de la création des modèles. Cette validation permet de vérifier que l'expert métier, en train de réaliser le modèle, respecte les règles d'association, de connexions et d'appartenance entre les composants d'un diagramme. Dans le cas où une des règles ne serait pas respéectée, l'outil affiche un signe d'interdiction, comme l'indique la figure C.3 si la relation n'est pas permise entre deux composants au moment de la modélisation.

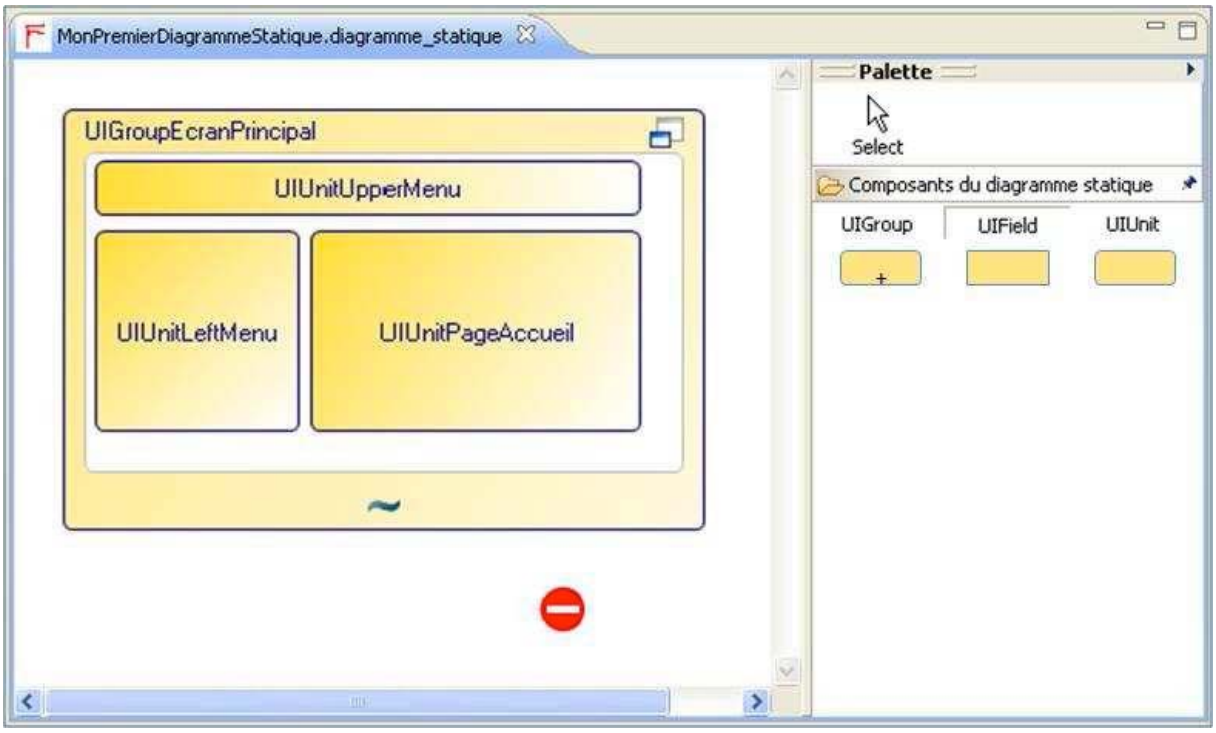


Figure C.3. Exemple de validation en "temps réel", pendant la création d'un modèle statique d'interaction (IM2), du respect des règles de modélisation

Le deuxième type de validation, que nous appellerons "validation formelle", peut être effectué après avoir terminé l'étape de modélisation. De manière pratique, l'outil propose la possibilité de valider les modèles métiers de façon plus précise par rapport à l'ensemble des règles associées au formalisme défini dans le cadre de PERCOMOM. Ainsi, lors de cette deuxième validation, les règles vérifiées sont les suivantes :

- Un évènement de début ne doit pas envoyer un message,
- Un évènement de fin ne doit pas recevoir un message,
- Un évènement intermédiaire ne doit pas envoyer un message,
- Un message ne peut pas être envoyé entre les éléments d'un même ensemble d'activité (cf. chapitre 3, §3.1.4.1),
- Un évènement de début ne doit pas être à la fin d'une séquence,

- Un évènement de fin ne doit pas être au début d'une séquence,
- Un branchement conditionnel doit avoir soit plus qu'une entrée soit plus qu'une sortie, et il n'est pas possible d'avoir les deux à la fois,
- Un branchement conditionnel ne doit pas posséder plus qu'une sortie de type "défaut".

Dans l'outil, une rubrique du menu principal intitulé « validate BPMN diagram » permet d'effectuer ce deuxième type de validation. S'il existe une erreur, un marqueur sous la forme d'une image apparaît sur le composant causant cette erreur. De plus, une description détaillée de l'erreur produite s'affiche sur la vue intitulée « Problems ». Un exemple illustrant ce type de validation est présenté au niveau de la figure C.4.

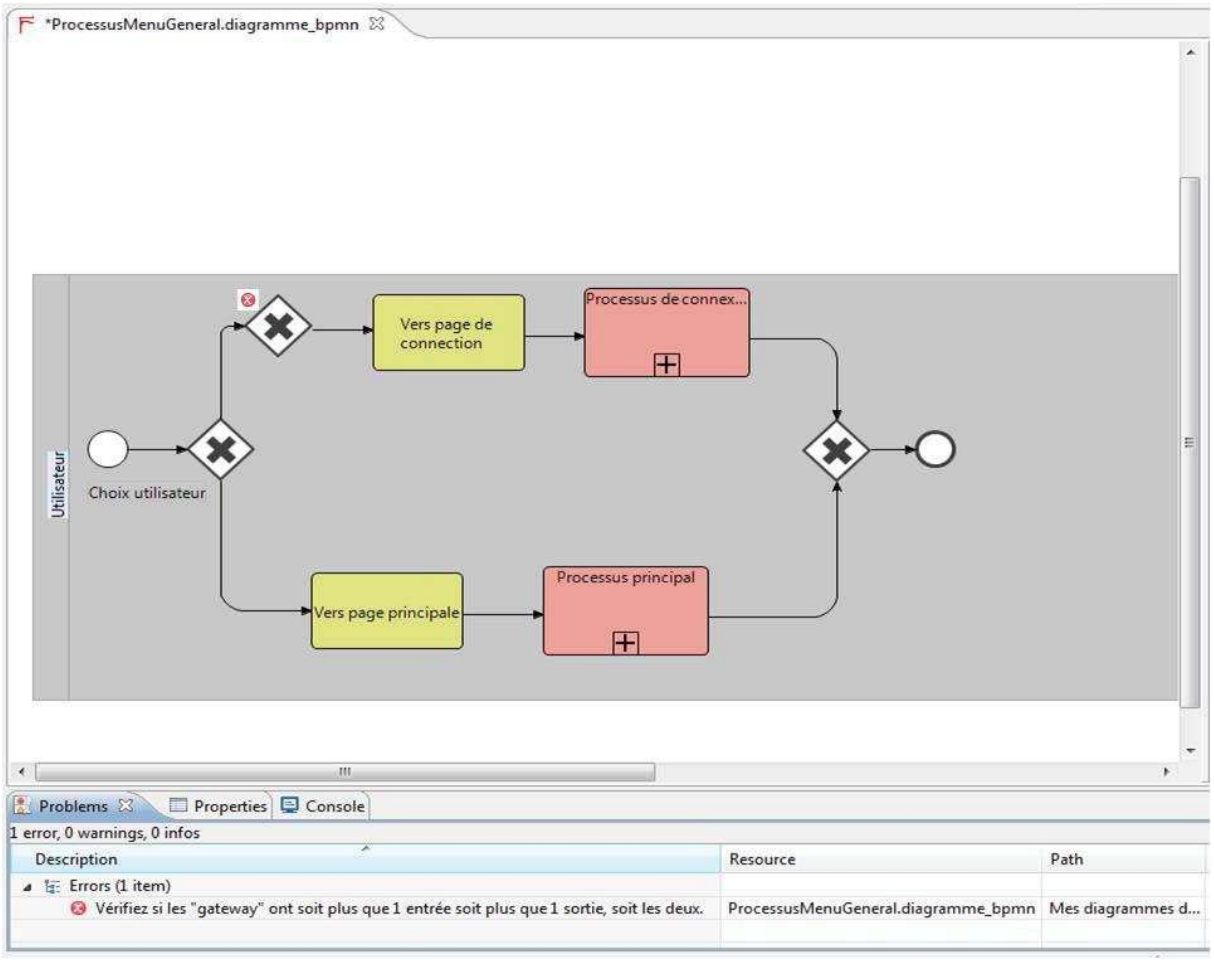


Figure C.4. Exemple de validation "formelle" d'un processus métier (IM1) dans l'outil graphique de création des processus métier (IM1) et des modèles statiques d'interactin (IM2)

C.2.3. L'association d'artefact aux différents éléments composant les modèles

Dans l'état actuel de son développement, l'outil ne prend pas encore en charge la gestion des artefacts aussi bien au niveau des modèles de processus métier (IM1) qu'au niveau des modèles statiques d'interaction (IM2). Afin d'apporter un premier niveau de réponse à ce problème, il est possible, dans l'outil, d'associer à chaque élément d'un modèle un fichier externe de type XML contenant l'ensemble des propriétés et valeurs de propriétés normalement définies au niveau de l'artefact. Par contre, l'outil ne vérifie pas la validité des contenus de ces fichiers XML par rapport au formalisme défini dans le cadre de PERCOMOM. La figure C.5 présente un exemple de ce type d'association.

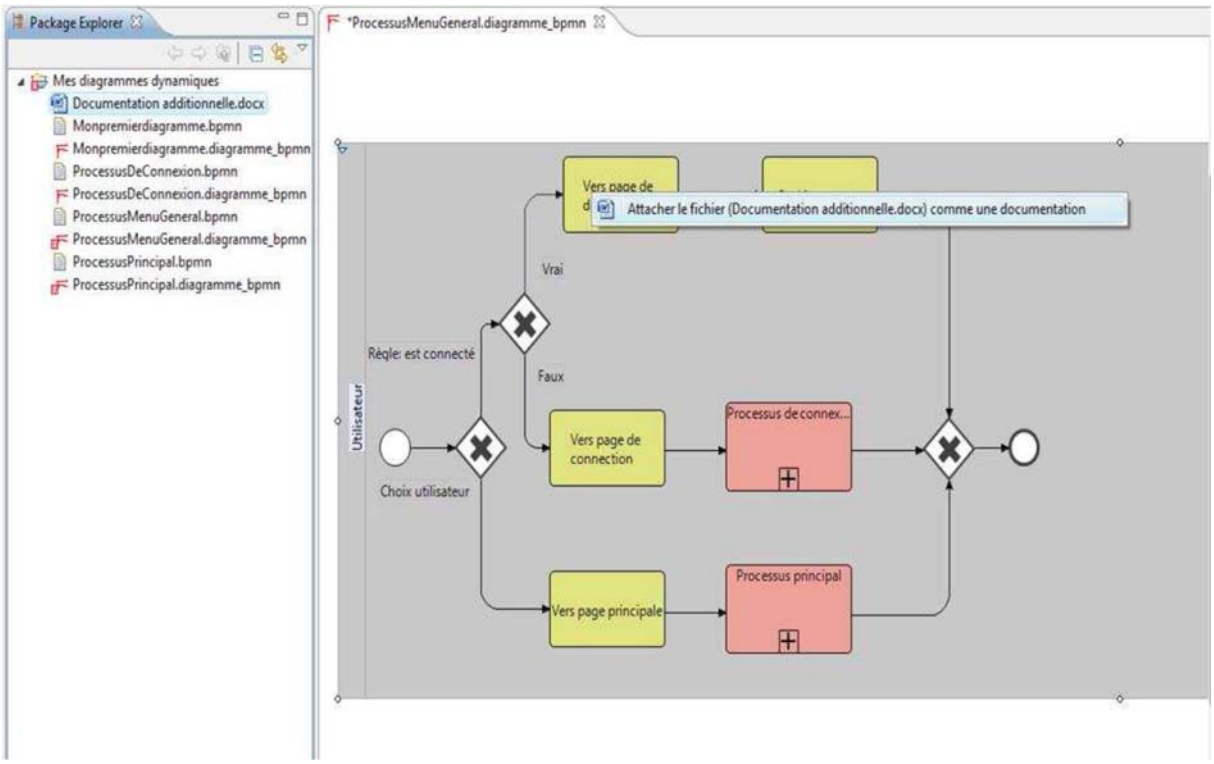


Figure C.5. Exemple d'association d'un fichier externe à un élément constitutif d'un modèle de processus métier (IM1) dans l'outil graphique de création des modèles de processus métier (IM1) et des modèles statiques d'interaction (IM2)

C.2.4. Transformation de modèles de processus métier (IM1) du niveau CIM au niveau PIM

Dans le cadre de la transformation des modèles de processus métier (IM1) du niveau CIM vers le niveau PIM, l'outil offre une fonctionnalité de migration du BPMN vers BPEL. Pour effectuer cette tâche, l'outil prend comme entrée le fichier d'extension .bpmn produit par le modélisateur et génére le fichier BPEL correpondant. A l'issue de cette transformation, s'il n'y a pas eu d'erreur, un fichier .bpel est généré.

Lors de la transformation de modèles, une vérification de la non existence des zones mortes (tâches ne pouvant jamais être exécutées) est effectuée.

Tous les résultats de la transformation de modèles, ainsi que les différentes étapes du processus de transformation de modèles, sont affichés, comme le montre la figure C.6, sur la vue de l'application intitulée « Console ».

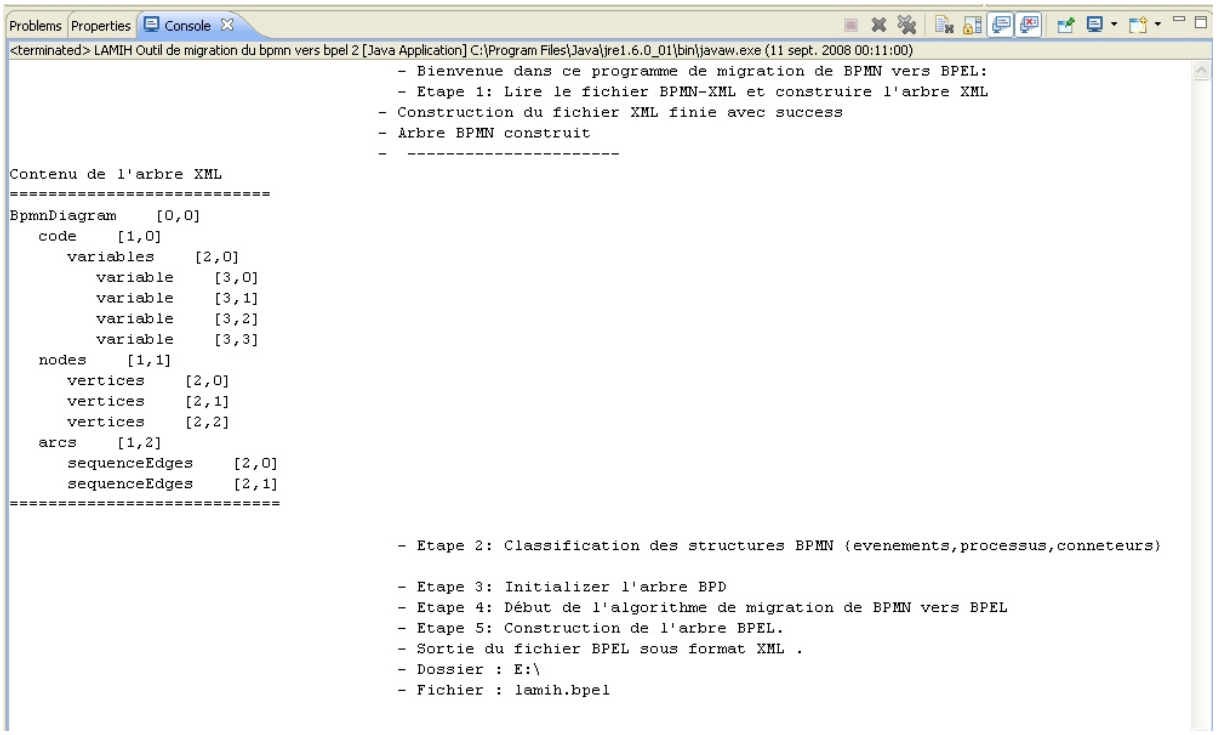


Figure C.6. Exemple de sortie de la "Console" lors de la transformation d'un modèle de processus métier (IM1) en un fichier BPEL (niveau PIM)

TITRE

PERCOMOM : une méthode de modélisation des applications interactives personnalisées appliquée à l'information voyageur dans le domaine des transports collectifs.

RESUME

Cette thèse propose une première version d'une méthode de modélisation des applications interactives personnalisées à travers une approche dirigée par les modèles. Cette méthode, appelée PERCOMOM (PERsonalization and COnceptual MOdeling Method), vise à permettre la création d'applications interactives de type WIMP (Windows, Icons, Mouse and Pointing device) par des experts métier ; chaque expert métier prenant en charge, à travers une ontologie de domaine et des modèles adaptés principalement basés sur la notation BPMN pour les modèles d'interaction, la modélisation d'une partie de l'application en fonction de son expertise propre.
PERCOMOM propose aussi, au niveau de la modélisation conceptuelle, des solutions pour prendre en compte les problématiques liées à la personnalisation des contenus dans les applications interactives ; que ce soit pour les problématiques liées au contexte, à l'utilisateur ou alors au type de données manipulées.
Enfin, à travers l'utilisation d'une architecture spécifique s'appuyant sur une approche de type MDA (Model Driven Architecture), PERCOMOM permet d'envisager une génération semi-automatique des applications à partir des modèles conceptuels et ceci pour des plateformes techniques différentes.
Notre contribution PERCOMOM a été appliquée pour le développement de systèmes d'information personnalisés dans le domaine de l'information voyageur.

Mots clés : méthodologie, Ingénierie Dirigée par les Modèles, personnalisation, transport

---

TITLE

PERCOMOM: a modeling method for personalized interactive application Application to traveler information in public transportation.

ABSTRACT

This thesis proposes a first version of a modelling method for personalized interactive applications through a model driven approach. This method, called PERCOMOM (PERsonalization and COnceptual MOdeling Method), aims to allow the creation of interactive applications of WIMP type (Windows, Icons, Mouse and Pointing device) by domain experts; each of them taking care, through the use of a domain ontology and specific models mainly based on the BMPN notation for the interactions models, of a part of the the application according to their expertise.
PERCOMOM also proposes, at the conceptual modelling level, solutions to take into account problems associated to the personalization of the contents in the interactive applications; whether it is for problems bound to the context, to the user or to the type of data used.
Finally, through the use of a specific architecture based on a MDA (Model Driven Architecture) approach, PERCOMOM allows to envisage a semi-automatic generation of applications from the abstract models and this for different technical platforms.
PERCOMOM was applied for the development of personalized information systems in the field of the traveler information.

Keywords : methodology, Model Driven Engineering, Personalization, transport