Ingénierie de l'interaction multimodale en entréeApproche à composants ICARE

Jullien Bouchet

▶ **To cite this version:**

Jullien Bouchet. Ingénierie de l'interaction multimodale en entréeApproche à composants ICARE. Interface homme-machine [cs.HC]. Université Joseph-Fourier - Grenoble I, 2006. Français. NNT : . tel-00155227

HAL Id: tel-00155227

https://theses.hal.science/tel-00155227

Submitted on 16 Jun 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE
présentée par

Jullien Bouchet

pour obtenir le titre de
DOCTEUR de L'UNIVERSITÉ JOSEPH FOURIER - GRENOBLE I
(arrêtés ministériels du 5 juillet 1984 et du 30 mars 1992)
Spécialité : Informatique

INGÉNIERIE DE L'INTERACTION MULTIMODALE EN ENTRÉE
Approche à composants ICARE

date de la soutenance : 07 décembre 2006

Composition du Jury :

Président :	M. Jean Caelen
Directeur de thèse :	Mlle Laurence Nigay
Rapporteurs :	M. Philippe Palanque
	M. Jean Vanderdonckt
Examinateurs :	M. Yacine Bellik
	M. Didier Bazalgette

Thèse préparée au sein du laboratoire de Communication Langagière et Interaction Personne-Système
Fédération IMAG
Université Joseph Fourier - Grenoble I

REMERCIEMENTS

Je tiens à exprimer toute ma reconnaissance à Laurence Nigay, ma directrice de thèse, pour m'avoir fait découvrir ce monde passionnant qu'est la Recherche. Je la remercie pour sa confiance, son encadrement, ses nombreux conseils et son soutien constant tout au long de ma thèse.

Je remercie également les membres du jury : merci aux deux rapporteurs, Philippe Palanque et Jean Vanderdonckt, merci à Jean Caelen, président du jury et directeur de mon laboratoire de recherche pendant la majorité de la thèse, merci également à Didier Bazalgette et à Yacine Bellik.

Un grand merci à toute l'équipe du projet Intuition. Je pense à Thierry Ganille, Denis Philippon, Christian Nouvel et à tous les autres participants de THALES-Avionics (Le-Haillan). Je remercie également les membres du LIIHS et du LIMSI, notamment Cyril Rousseau, mon compagnon de thèse dans ce projet.

Je ne peux pas oublier toute l'équipe IIHM et son esprit familial. J'exprime ma profonde gratitude à Joelle Coutaz pour avoir créé cette équipe et cette ambiance. Merci aussi aux collègues et ami(e)s qui entretiennent ou qui ont entretenu cet esprit : Alex, Anne, Benoît, Béran, Céline, Chaouki, David, Fanf, Fred(s), Gaëlle, Gaëtan, Gilles, Guillaume, Isabelle, Jorge, JS, Julie, Julien(s), Kris, Lionel, Manu, Marcos, Nico, Olfa, Philippe, Renaud, Sophie, Sylvain, Sylvie, Tung, Vincent, Xavier, Yann et les oubliés (il y en a forcément). Mention spéciale pour ceux et celles qui ont partagé mon bureau (ou qui me l'ont piqué).
Merci également à Annie, Valérie et Nicole.

Je remercie vivement tous mes amis, les puerks, qui savent combien ils comptent pour moi. Je les remercie pour tout le bonheur qu'ils me procurent.

Merci beaucoup à toute ma famille pour leur soutien et leur amour. Un énorme merci à mes parents, dont je suis fier et éternellement reconnaissant pour avoir rempli ma vie de bonheur et de joie. Cette thèse est aussi la vôtre (je savais bien que vous aviez une idée en tête en investissant dans ce PC Amstrad 1640, il y a environ 16 ans).

Un paragraphe spécial pour Angélica, directrice artistique de ce mémoire : agradeço tudo o que tem feito por mim. Obrigado por sua ajuda e sua paciência. Esta tese é um pouco sua, eu espero poder ajudá-la como você me ajudou. Obrigado pelo seu amor, motor da minha motivação. Graças a você esta tese chega ao seu fim, devagar e sempre. Muito obrigado à sua família também.

Je termine par une pensée à mon meilleur ami, Gilles, parti trop vite, à qui je dédie mon mémoire de thèse.

TABLE DES MATIÈRES

TABLE DES FIGURES 11

TABLE DES TABLEAUX 15

INTRODUCTION 19

1 Systèmes multimodaux 20
1.1 Un peu d'histoire 20
1.2 Multiplication des modalités d'entrée 22
1.3 Variété des systèmes multimodaux 23
2 Difficultés pour la conception et le développement des systèmes multimodaux 25
2.1 Processus de développement difficile et couteux 25
2.2 Réutilisabilité de l'existant quasi-impossible 26
2.3 Absence d'outils adaptés 26
3 Objectifs et démarches de travail 28
3.1 Définition du modèle conceptuel : vers une théorie de la multimodalité 29
3.2 Opérationnalisation du modèle conceptuel dans un outil pour la multimodalité 29
3.3 Test de l'approche par la conception et le développement de plusieurs systèmes interactifs 30
4 Organisation du Mémoire 31

PARTIE 1 33
ESPACE DE CONCEPTION
VERS UNE THÉORIE DE LA MULTIMODALITÉ

CHAPITRE 1 37
PARADIGMES D'INTERACTION RÉCENTS
1 Interfaces geste/parole 39
2 Interfaces tangibles 40
3 Interfaces de réalité augmentée 42
4 Résumé du chapitre 1 44

CHAPITRE 2 45
MODÈLES DE LA MULTIMODALITÉ
1 Introduction : les systèmes interactifs multimodaux 46
2 Utilisateur et multimodalité 48
2.1 Théorie de l'action 48
2.2 Modes de communication 49
2.3 Etat de l'organisme et personnalité 50
2.4 Gestion des ressources 51
2.5 Fonctionnement cognitif de base 52
2.6 Compétences 53

3 Système informatique et multimodalité....55
3.1 Terminologie....55
3.2 Modèles de type "Acquisition – Traitement – Rendu"....56
3.3 Modèle des dispositifs physiques de [Buxton 1983]....58
3.4 Espace de Card, Mackinlay et Robertson [Card 1990]....58
3.5 Modèle des ressources d'interaction [Lachenal 2004]....59
3.6 Modèle d'interaction instrumentale [Beaudouin-Lafon 2000]....60
3.7 Taxonomie de la théorie de [Bernsen 1994] et ses extensions....61
3.8 Espace de [Foley 1984]....63
3.9 Espace de [Frohlich 1991]....63
3.10 Espace MSM [Nigay 1994]....64
4 Unification des modèles de psychologie cognitive et d'IHM....66
4.1 Processeur humain et principe "perception – traitement – action"....66
4.2 Théorie de l'action et modèle pipeline....67
4.3 Modèle ICS et couple <Dispositif, Langage d'interaction>....68
5 Résumé du chapitre 2....70

CHAPITRE 3....71
USAGES DES MODALITÉS D'INTERACTION
1 Résultats conceptuels....72
1.1 UOM : classification des systèmes multimodaux [Nigay 1994]....72
1.2 Propriétés CARE [Coutaz 1994] et extensions....76
1.3 Espace TYCOON [Martin 2002]....78
1.4 Synthèse....79
2 Résultats expérimentaux....80
2.1 Paramètres déterminant l'utilisabilité des modalités d'interaction....80
2.2 Qualité de l'interaction mettant en œuvre plusieurs modalités d'interaction....84
3 Résumé du chapitre 3....89

CHAPITRE 4....91
MODÈLE CONCEPTUEL DE LA MULTIMODALITÉ : UN PREMIER PAS VERS LA THEORIE
1 Méta-modèle....92
1.1 Cadre général....92
1.2 Possibilités d'interaction....93
1.3 Synthèse....95
2 Modalités d'interaction....97
2.1 Caractéristiques globales....97
2.2 Dispositifs....101
2.3 Langages d'interaction....105
2.4 Recommandations pour la mise en œuvre d'une modalité d'interaction....108

3 Combinaisons de modalités..... 111
3.1 Définition..... 111
3.2 Propriétés..... 111
3.3 Recommandations pour la mise en œuvre des combinaisons..... 113
4 Illustration du modèle conceptuel sur un exemple..... 116
4.1 Description du cas d'étude..... 116
4.2 Modalité GPS en coordonnées WGS84..... 116
4.3 Modalité de reconnaissance vocale..... 117
4.4 Combinaison complémentaire des deux modalités..... 118
5 Résumé du chapitre 4..... 120

PARTIE 2 123
OPERATIONNALISATION DE L'ESPACE CONCEPTUEL : L'OUTIL ICARE

CHAPITRE 5..... 127
OUTILS DE CONCEPTION LOGICIELLE ET DE DÉVELOPPEMENT
1 Outils et cycle de vie du logiciel..... 128
2 Grille d'analyse..... 130
2.1 Carte d'identité de l'outil..... 130
2.2 Critères d'IHM..... 131
2.3 Critères pour la multimodalité..... 131
2.4 Critères pour la fusion des données..... 132
3 Outils existants..... 133
3.1 Outil d'intégration multimodal du W3C..... 133
3.2 FAME..... 136
3.3 Approche multimodale à creusets..... 138
3.4 Outil multimodal à base de règles..... 139
3.5 IMBuilder et MEngine..... 141
3.6 Quickset..... 143
3.7 Embassi..... 145
3.8 Intuikit..... 147
3.9 Icon..... 148
3.10 ISL4WComp..... 151
3.11 PetShop..... 152
4 Résumé du chapitre 5..... 155

CHAPITRE 6..... 159
OUTIL ICARE, LES PRINCIPES DE L'APPROCHE À COMPOSANTS
1 Technologies à composants..... 160

1.1 Terminologie..... 160
1.2 Modèle à composants..... 162
2 Spécifications des composants de l'outil ICARE..... 165
2.1 Modalités d'interaction..... 165
2.2 Combinaisons de modalités d'interaction..... 173
3 Communication entre composants..... 181
4 Utilisation dans un système informatique et architecture logicielle..... 182
5 Résumé du chapitre 6..... 184

CHAPITRE 7..... 185
OUTIL ICARE, UN EXEMPLE D'IMPLÉMENTATION
1 Les technologies à composants existantes..... 186
1.1 Composants de Microsoft..... 186
1.2 Composants de Sun Microsystems..... 186
1.3 Composants de l'OMG..... 187
1.4 Composants de l'OSGi Alliance..... 188
2 JavaBeans..... 189
2.1 Pourquoi ce choix..... 189
2.2 Spécificités..... 189
3 Les composants de l'outil ICARE..... 191
3.1 Implémentation des composants Dispositifs et Langages d'interaction..... 191
3.2 Implémentation des composants de combinaison..... 191
3.3 Communication entre composants dans ICARE..... 192
4 Editeur graphique d'ICARE..... 193
4.1 Objectifs..... 193
4.2 Description..... 193
4.3 Démarche de conception..... 194
4.4 Génération automatique de code..... 195
4.5 Lien avec le reste du système informatique..... 195
5 Résumé du chapitre 7..... 198

CHAPITRE 8..... 199
OUTIL ICARE : SYNTHÈSE
1 Carte d'identité..... 200
2 Critères pour l'IHM..... 201
3 Critères généraux pour la multimodalité..... 202
4 Critères pour la fusion des données..... 203
5 Grille d'analyse..... 204
6 Résumé du chapitre 8..... 205

PARTIE 3 207
REALISATION DE
SYSTEMES MULTIMODAUX

CHAPITRE 9.....211
SYSTÈME INTERACTIF JOUET
1 Cahier des charges : présentation du système interactif..... 212
1.1 Tâche..... 212
1.2 Modalités d'interaction.....212
1.3 Combinaisons de modalités.....213
1.4 Interface graphique.....213
2 Spécification, conception et codage.....214
2.1 Spécifications des modalités d'interaction et des combinaisons.....214
2.2 Conception globale et détaillée.....218
3 Résumé du chapitre 9.....221

CHAPITRE 10.....223
SYSTÈMES INTERACTIFS DU PARADIGME GESTE/PAROLE
1 Système interactif YellowPages.....224
1.1 Description globale.....224
1.2 Conception : assemblage des composants..... 225
1.3 Synthèse..... 228
2 Simulateur ATC.....229
2.1 Description globale.....229
2.2 Conception : assemblage des composants pour la spécification des requêtes..... 230
2.3 Synthèse..... 231
3 Résumé du chapitre 10.....232

CHAPITRE 11.....233
SYSTÈMES INTERACTIFS DE RÉALITÉ AUGMENTÉE
1 MEMO.....234
1.1 Description globale.....234
1.2 Conception de l'interaction multimodale en entrée.....235
1.3 Synthèse..... 237
2 Simulateur d'avion de combat.....238
2.1 Description globale.....238
2.2 Conception de l'interaction multimodale en entrée pour les tâches de réalité augmentée..... 240
2.3 Synthèse..... 241
3 Résumé du chapitre 11.....242

CONCLUSION 245
1 Résumé de notre contribution 246
1.1 Modèle conceptuel de la multimodalité 246
1.2 Outil de conception et de développement de l'interaction multimodale : ICARE 246
1.3 Réalisations logicielles 247
2 Limites 249
2.1 Limitations conceptuelles 249
2.2 Limitations technologiques 251
3 Perspectives 252
3.1 Extensions du modèle conceptuel 252
3.2 Evaluations ergonomiques 253
3.3 Extensions de l'éditeur 257

BIBLIOGRAPHIE 257

LISTE DES PUBLICATIONS 268

ANNEXES 269

Table des figures

Introduction

Figure 0.1 : une première définition de “modalité d’interaction”....19
Figure 0.2 : l’ordinateur TX-0....20
Figure 0.3 : Ivan Sutherland et la première interface graphique SketchPad....20
Figure 0.4 : la première souris créée par Doug Engelbart....21
Figure 0.5 : le clavier à accord de Doug Engelbart....21
Figure 0.6 : un dispositif de contrôle au genou....21
Figure 0.7 : le système “put-that-there” de R. Bolt [Bolt 1980]....21
Figure 0.8 : réalisation d’un zoom avec les doigts [Han 2005]....22
Figure 0.9 : reconnaissance des mouvements de la souris pour lancer des sorts dans le jeu Black and White (Lionhead Studios)....22
Figure 0.10 : exemple de geste effectué à la souris pour revenir à la page précédente dans le navigateur Web Mozilla (extension Optimoz Mouse Gesture)....22
Figure 0.11 : exemple de Phicons : manipulation d’une carte en 2D du campus du MIT par la manipulation d’une maquette miniature de l’un des bâtiments [Ishii 1997]....22
Figure 0.12 : utilisation de CASPER lors de la phase de guidage [Dubois 2001]....23
Figure 0.13 : cockpit de l’A380, profusion de modalités d’interaction....23
Figure 0.14 : Tom Cruise manipulant des fichiers “à la main” et par reconnaissance vocale dans le film “Minority Report”....24
Figure 0.15 : console portable Nintendo DS avec écran tactile....24
Figure 0.16 : manette de jeu Nintendo Wii avec capteurs de mouvement....24
Figure 0.17 : équipement d’un utilisateur dans le système MARS [Feiner 1997]....24
Figure 0.18 : cockpits du Harrier (à gauche) et du Rafale (à droite)....25
Figure 0.19 : le modèle de Gaines [Gaines 1991] sur l’évolution du niveau d’apprentissage d’une technologie, adapté à l’interaction multimodale....28

Chapitre 1

Figure 1.1 : accroissement de l’intégration du monde physique dans les interfaces....37
Figure 1.2 : le système “put-that-there” d’après [Bolt 1980]....39
Figure 1.3 : les médiaBlocks d’après [Ullmer 1998]....40
Figure 1.4 : le pinceau de I/O brush d’après [Ryokai 2004]....40
Figure 1.5 : HUD d’un F16....42
Figure 1.6 : HUD d’une BMW....42
Figure 1.7 : lunettes semi-transparente Glasstron de Sony....42

Chapitre 2

Figure 2.1: le modèle SAI d’après [Rabardel 1995]....46

Figure 2.2 : système interactif = utilisateur + système informatique d'après [Nigay 1994]..... 46
Figure 2.3 : théorie de l'action de Norman d'après [Norman 1986]..... 48
Figure 2.4 : utilisation conjointe de deux modalités, d'après la gestion des ressources de [Leplat 2000]..... 51
Figure 2.5 : modèle du processeur humain d'après [Card 1983]..... 52
Figure 2.6 : le modèle ICS [Barnard 1987]..... 53
Figure 2.7 : niveaux physique et logique d'une modalité d'interaction d'après [Vernier 2001]..... 56
Figure 2.8 : cadre général : les 3 modules d'un système informatique..... 57
Figure 2.9 : le modèle Pipeline d'après [Nigay 1994]..... 57
Figure 2.10 : Hand with reflecting sphere, lithographie, M.C. Escher, 1935..... 66
Figure 2.11 : unification des modèles de base (le processeur humain de [Card 1983] et le modèle "perception-traitement-action")..... 66
Figure 2.12: le système informatique comme miroir de l'homme, fusion de la théorie de l'action de [Norman 1986] et du modèle Pipeline de [Nigay 1994]..... 67
Figure 2.13 : unification du modèle ICS et de la définition de la modalité comme un couple <Dispositif, langage d'interaction>..... 69

Chapitre 3

Figure 3.1 : usage des modalités d'interaction..... 71
Figure 3.2 : M²LD d'après [Nigay 1994]..... 72
Figure 3.3 : équipement d'un joueur d'ARQuake [Piekarski 2002]..... 73
Figure 3.4 : O²LD d'après [Nigay 1994]..... 73
Figure 3.5 : les différents niveaux d'abstraction de l'interaction entre un utilisateur et le noyau fonctionnel au sein du système informatique..... 74
Figure 3.6 : usage des Langages et des Dispositifs..... 75
Figure 3.7 : application des schémas de composition aux cinq aspects de composition [Vernier 2001]..... 77
Figure 3.8 : pourcentage des commandes que les utilisateurs ont exprimées par la complémentarité de deux modalités d'interaction en fonction du type de commande. Graphique issu de [Oviatt 1999]..... 81

Chapitre 4

Figure 4.1 : diagramme de classes UML du cadre général de la multimodalité, étape 1 : le cadre général..... 92
Figure 4.2 : diagramme de classes UML des possibilités d'interaction de la multimodalité..... 93
Figure 4.3 : diagramme de classes complet des concepts de la multimodalité..... 95
Figure 4.4 : adéquation entre le dispositif, le langage d'interaction et la tâche, rapport entre le niveau métaphorique et le degré d'intégration..... 98

Figure 4.5 : degré d'indirection d'une modalité d'interaction, rapport entre le niveau d'incarnation et la différence de temps entre l'action physique et la réponse de l'objet manipulé.....100
Figure 4.6: diagramme d'objets UML de la modalité GPS en coordonnées WGS84.....117
Figure 4.7: diagramme d'objets UML de la modalité reconnaissance vocale.....118
Figure 4.8 : diagramme d'objets UML de la combinaison des modalités GPS en coordonnées WGS84 et reconnaissance vocale.....119
Figure 4.9 : diagramme de classes UML des possibilités multimodales.....120

Chapitre 5

Figure 5.1 : les phases des cycles de vie en spirale [Boehm 1988] et en V [McDermid 1984].....128
Figure 5.2 : outil d'intégration multimodal W3C, d'après [W3C 2003].....134
Figure 5.3 : décomposition du composant "Entrée" (Outil du W3C), d'après [W3C 2003].....134
Figure 5.4 : architecture des applications suivant FAME, d'après [Duarte 2006].....136
Figure 5.5 : configuration d'entrée, issu de [Dragicevic 2004].....149
Figure 5.6 : un exemple de configuration d'entrée, issu de [Dragicevic 2004].....149
Figure 5.7 : éditeur PetShop.....153

Chapitre 6

Figure 6.1 : prise en compte du processus de développement [Favre 2003].....161
Figure 6.2 : interfaces d'un composant.....162
Figure 6.3 : assemblage de composants.....163
Figure 6.4 : diagramme de classes UML des possibilités multimodales.....165
Figure 6.5 : une modalité d'interaction d'entrée dans ICARE.....173
Figure 6.6 : un composant A assigné à un composant B dans l'outil ICARE.....173
Figure 6.7 : deux composants équivalents (A et B) pour un composant C dans l'outil ICARE.....173
Figure 6.8 : un composant de combinaison et ses deux ports permettant de combiner les données de deux composants.....174
Figure 6.9 : Complémentarité de deux langages d'interaction.....176
Figure 6.10 : Redondance de deux langages d'interaction.....178
Figure 6.11 : Redondance de deux langages d'interaction.....179
Figure 6.12 : Redondance/Equivalence de trois modalités d'interaction.....179
Figure 6.13 : exemple d'utilisation des composants connectés à un système informatique existant.....182
Figure 6.14 : (a) le modèle d'architecture logiciel ARCH, (b) les composants ICARE dans ARCH.....183

Chapitre 7

Figure 7.1 : l'éditeur graphique d'ICARE pour manipuler les composants... 194

Chapitre 9

Figure 9.1 : fenêtre principale du système MID..... 213
Figure 9.2 : modalité "utilisation du clavier" dans MID..... 214
Figure 9.3 : modalité "utilisation de la souris" dans MID..... 215
Figure 9.4 : modalité "reconnaissance vocale" dans MID..... 216
Figure 9.5 : équivalence des modalités d'interaction dans MID..... 217
Figure 9.6 : deux cas de redondance des modalités d'interaction dans MID..... 217
Figure 9.7 : architecture de MID..... 218
Figure 9.8 : premier assemblage de composants pour le système MID..... 218
Figure 9.9 : redondance partielle des modalités d'interaction pour le système MID..... 219

Chapitre 10

Figure 10.1 : interface graphique de YellowPages..... 224
Figure 10.2 : spécification des champs texte de YellowPages..... 226
Figure 10.3 : commande simple dans YellowPages..... 227
Figure 10.4 : commandes complexes dans YellowPages..... 228
Figure 10.5 : interface graphique du simulateur ATC..... 229
Figure 10.6 : assemblage de composants pour les requêtes du simulateur ATC..... 230

Chapitre 11

Figure 11.1 : vision de l'utilisateur avec le système interactif MEMO (image reconstituée)..... 234
Figure 11.2 : assemblage des composants ICARE de MEMO..... 235
Figure 11.3 : pilote dans le simulateur d'avion de combat temps-réel..... 238
Figure 11.4 : assemblage de composants ICARE pour les tâches de réalité augmentée..... 240

Conclusion

Figure 12.1 : le modèle de Gaines sur l'évolution du niveau d'apprentissage d'une technologie, adapté à l'interaction multimodale d'après [Gaines 1991]..... 245
Figure 12.2 : les composants ICARE dans l'architecture logicielle ARCH..... 249
Figure 12.3 : deux solutions d'assemblage de composants ICARE pour un système de

CHAPITRE 5

Tableau 5.1 : propriétés de la plate-forme W3C..... 135
Tableau 5.2 : propriétés de l'outil FAME.....138
Tableau 5.3 : propriétés de l'approche à creusets.....139
Tableau 5.4 : propriétés du moteur de fusion à base de règles..... 141
Tableau 5.5 : propriétés de l'outil IMBulder/MEngine..... 143
Tableau 5.6 : propriétés de l'outil Quickset..... 145
Tableau 5.7 : propriétés de l'outil Embassi.....147
Tableau 5.8 : propriétés de l'outil Intuikit.....148
Tableau 5.9 : propriétés de l'outil Icon..... 150
Tableau 5.10 : propriétés de l'outil ISL4WComp.....152
Tableau 5.11 : propriétés de l'outil PetShop.....154
Tableau 5.12 : synthèse des caractéristiques des outils (partie 1/2).....156
Tableau 5.13 : synthèse des caractéristiques des outils (partie 2/2).....157

CHAPITRE 6

Tableau 6.1 : attributs du composant Dispositif.....169
Tableau 6.2 : attributs des composants Systèmes représentationnels..... 172
Tableau 6.3 : attributs des composants de combinaison.....176
Tableau 6.4 : attributs des événements dans l'outil ICARE..... 181

CHAPITRE 8

Tableau 8.1 : propriétés de l'outil ICARE.....204

INTRODUCTION

INTRODUCTION

La multimodalité est un axe de recherche du domaine de l'Interaction Homme-Machine (voir l'encadré ci-dessous pour un rappel terminologique des termes "modalité" et "multimodalité"). Il a vu naître en 1996 sa première conférence internationale qui présente sa 8ème édition en 2006. Avec le nouveau défi de l'informatique ubiquitaire, appelée aussi informatique pervasive, de nouvelles perspectives pour la multimodalité sont proposées. Les moyens d'interaction sont de plus en plus nombreux, diversifiés et omniprésents. L'apparition et l'utilisation de nombreuses modalités d'interaction dans différents contextes environnementaux modifient les modèles et outils existants [Myers 2000]. Ce travail de thèse propose d'aborder le problème de la conception et du développement des systèmes multimodaux en entrée (de l'utilisateur vers le système informatique) prenant en compte les nouveaux paradigmes d'interaction.

Terminologie : les modalités d'interaction et la multimodalité

Devant la profusion des définitions autour des termes "modalité" et "multimodalité", une première définition est donnée afin de servir de base pour appréhender les premières sections de ce mémoire de thèse. Des définitions plus précises et détaillées seront données ensuite. Illustrées à la Figure 0.1, l'utilisateur interagit avec le système informatique pour réaliser des tâches. Les moyens d'action et de perception, appelés "modalités d'interaction", sont les médiateurs matériels et logiciels permettant à un utilisateur d'agir sur le système informatique ou de percevoir son état. Les modalités d'interaction composent le module interface du système informatique. Nous pouvons distinguer les modalités d'interaction en entrée (moyens d'action de l'utilisateur pour interagir avec le système informatique) des modalités d'interaction en sortie (moyens de perception de l'état du système informatique par l'utilisateur). La multimodalité correspond à la multiplicité des modalités d'interaction dans un même système informatique. Ces travaux de recherche se consacrent aux systèmes multimodaux en entrée.

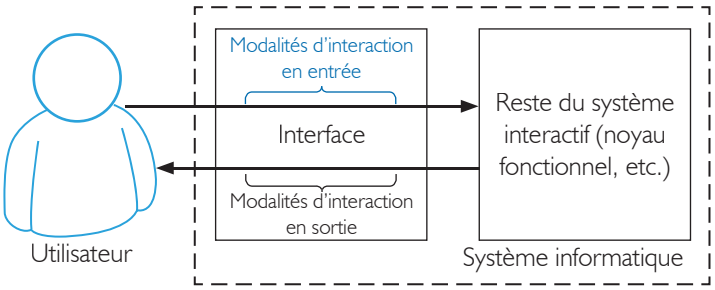


Figure 0.1 : une première définition de "modalité d'interaction".

Afin de mesurer la portée de ces travaux, le contexte de recherche, les objectifs et la structure de ce mémoire de thèse sont exposés dans les sections suivantes.

1.1 Un peu d'histoire

Depuis la naissance de l'informatique, les moyens d'action entre l'homme et le système informatique n'ont cessé d'évoluer. En 1957, l'ordinateur expérimental TX-0 (Figure 0.2) voit le jour au laboratoire Lincoln du MIT (Massachussetts Institue of Technology). Le TX-0 ouvre les portes de la multimodalité (multiplicité des modalités d'interaction) en introduisant pour la première fois deux dispositifs d'entrée : un clavier et un stylo optique.

C'est en 1963 que le premier système multimodal voit le jour avec la thèse d'Ivan Sutherland [Sutherland 1963], qui marque aussi les débuts de l'Interaction Homme-Machine (IHM). Il développe au laboratoire Lincoln du MIT la première interface graphique, SketchPad (Figure 0.3), qui établie la plupart des concepts fondamentaux de l'IHM d'aujourd'hui. SketchPad se base sur une évolution du TX-0, le TX-2, qui présente également un clavier et un stylo-optique permettant, entre autres, la désignation directe d'objets à l'écran ou la possibilité d'effectuer des actions de zoom.

Figure 0.2 : l'ordinateur TX-0.

Figure 0.3 : Ivan Sutherland et la première interface graphique SketchPad.

En 1964, Doug Engelbart invente la souris (Figure 0.4) pour désigner des objets à l'écran. En 1968, il met au point un clavier à accord (Figure 0.5) qui permet d'entrer des données en composant des accords avec les doigts d'une main et l'utilisation du clavier ou de la souris avec l'autre main. D'autres dispositifs sont inventés, comme le dispositif de contrôle au genou (Figure 0.6) mais comme le clavier à accord, ils ne connaîtront pas le succès de l'association clavier / souris pour la manipulation directe des objets graphiques.

Bien que nous constations que les systèmes informatiques intègrent plusieurs modalités d'interaction depuis 1957, les problématiques de recherche dédiées à la

Figure 0.4 : la première souris créée par Doug Engelbart.

Figure 0.5 : le clavier à accord de Doug Engelbart.

multimodalité sont assez récentes. C'est à partir de 1980 que la multimodalité est étudiée aux Etats-Unis. En particulier, Richard Bolt propose le paradigme du "put-that-there" ("mets ça là") dans le cas de la combinaison de deux modalités d'entrée naturelles : la parole et le geste [Bolt 1980]. Pour déplacer un objet, l'utilisateur pointe du doigt l'objet voulu en prononçant "put that" et désigne sa destination en prononçant le "there" (Figure 0.7).

En 1983, Nicholas Negroponte fonde le MediaLab au MIT, avec comme objectif l'étude de l'intégration des nouvelles technologies au sein d'une interface. En France, la prise de conscience a lieu plus tard, avec la création en 1990 d'un groupe d'étude au sein du GDR-PRC CHM (Groupement De Recherche – Programme de Recherches Concertées Communication Homme-Machine).

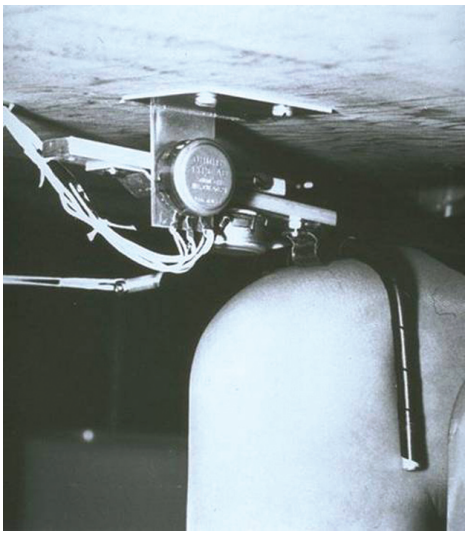


Figure 0.6 : un dispositif de contrôle au genou.

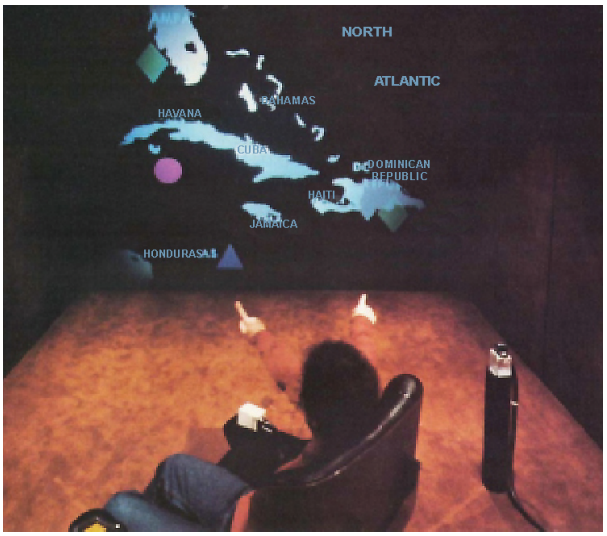


Figure 0.7 : le système "put-that-there" de R. Bolt [Bolt 1980].

1.2 Multiplication des modalités d'entrée

Depuis 1990, des avancées significatives ont été faites au niveau des modalités d'entrée et de leur mise en œuvre logicielle. Les modalités d'entrée se multiplient et deviennent de plus en plus robustes. Il existe aujourd'hui une pléthore de modalités dont une liste exhaustive serait difficilement réalisable. A titre d'exemples, les jeux vidéo ont démocratisé l'utilisation des joysticks et des manettes de jeu permettant le contrôle des actions de l'avatar. Les systèmes de reconnaissance vocale sont de plus en plus nombreux et efficaces. Par exemple, le logiciel Dragon NaturallySpeaking8 de ScanSoft permet de reconnaître jusqu'à 160 mots/minutes (contre 50 mots/minute en moyenne pour le clavier) et peut atteindre une précision de reconnaissance de 99%[1] [Dragon 2006]. Les modalités gestuelles se sont multipliées dans nos systèmes interactifs, que ce soit pour manipuler un affichage en 2 dimensions avec les doigts (Figure 0.8) [Han 2005] ou pour reconnaître des gestes réalisés avec la souris afin d'effectuer des actions simples (Figure 0.9 et Figure 0.10). D'autres modalités permettent la manipulation d'icônes au travers d'objets physiques comme les Phicons (Figure 0.11) [Ishii 1997].

[1] Données fournies par l'éditeur.

Figure 0.8 : réalisation d'un zoom avec les doigts [Han 2005].

Figure 0.9 : reconnaissance des mouvements de la souris pour lancer des sorts dans le jeu Black and White (Lionhead Studios).

Figure 0.10 : exemple de geste effectué à la souris pour revenir à la page précédente dans le navigateur Web Mozilla (extension Optimoz Mouse Gesture).

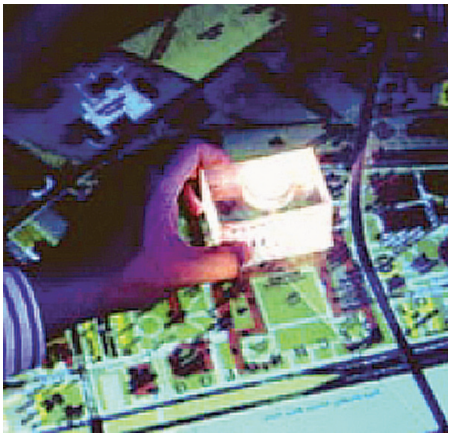


Figure 0.11 : exemple de Phicons : manipulation d'une carte en 2D du campus du MIT par la manipulation d'une maquette miniature de l'un des bâtiments [Ishii 1997].

1.3 Variété des systèmes multimodaux

Les progrès à la fois en termes de conception et de compréhension d'usages multimodaux (cf. Chapitre 1 et 2) ont favorisé la mise au point de systèmes multimodaux dans de nombreux domaines.

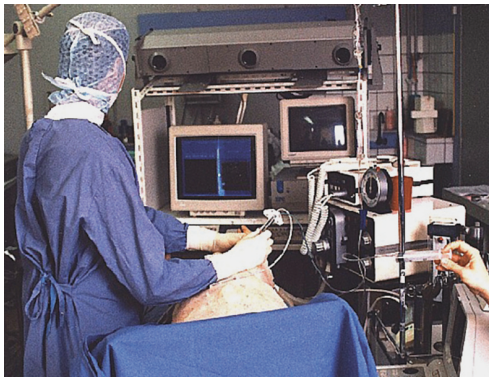


Figure 0.12 : utilisation de CASPER lors de la phase de guidage [Dubois 2001].

Le **domaine médical** s'intéresse de plus en plus aux systèmes multimodaux, qui semblent offrir des possibilités de sécurité et d'efficacité pour certaines chirurgies jugées trop invasives. Afin de garantir la qualité des gestes chirurgicaux et de limiter ainsi les conséquences post-opératoires, les chirurgiens sont maintenant aidés par des systèmes de guidage précis et sûrs. Par exemple, le système CASPER (Computer ASsisted PERicardial puncture) présenté dans [Dubois 2001] assiste le chirurgien lors de la réalisation d'une ponction péricardique en repérant la position 3D et la trajectoire de l'aiguille de ponction en fonction de la trajectoire prévue.

Plusieurs systèmes multimodaux existent aussi dans le **domaine militaire**. L'abondance des tâches et la recherche d'une efficacité totale dans leurs réalisations et leurs conséquences amènent une innovation constante des modalités d'interaction. Tout d'abord, la conduite des systèmes embarqués ayant une mission de défense (tanks, avions, drones, équipement du militaire, etc.) repose sur des interfaces multimodales riches. Par exemple, l'équipement proposé par Sagem pour le fantassin du futur (projet FELIN) inclut de nombreux dispositifs comme un système GPS, un PDA ou un viseur de casque. D'autre part, des systèmes multimodaux basés sur des modalités naturelles (geste et parole) pour les postes de commandements sont aussi largement étudiés [Cohen 1997].

Le **domaine aéronautique civil** utilise de plus en plus de modalités. Avec les joysticks pour le contrôle de l'avion, les manettes des gaz et les multiples boutons, de nouvelles modalités comme l'utilisation des surfaces tactiles, des dispositifs de pointage ou des systèmes de reconnaissance vocale font leur apparition. Par exemple, l'A380 utilise un dispositif de pointage de style trackball dans son cockpit et permet aux pilotes de manipuler directement les objets graphiques à l'écran (Figure 0.13).

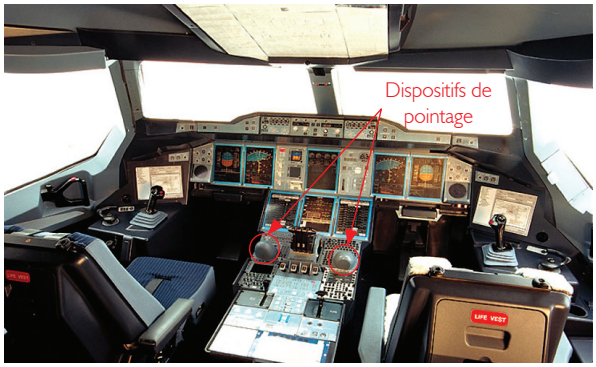


Figure 0.13 : cockpit de l'A380, profusion de modalités d'interaction.

Figure 0.14 : Tom Cruise manipulant des fichiers “à la main” et par reconnaissance vocale dans le film “Minority Report”.

Le **grand public** n’est pas en reste comme en témoigne la diffusion du film “Minority Report”, dans lequel Tom Cruise manipule des fichiers vidéo en combinant le geste et la parole (Figure 0.14). La plupart des téléphones portables sont équipés, en plus du clavier, d’un système de reconnaissance vocale permettant par exemple de lancer directement un appel en prononçant simplement le nom du correspondant. Les consoles de jeux couplent écran tactile et boutons classiques (Figure 0.15) ou proposent des manettes de jeux qui intègrent des capteurs de mouvement permettant, par exemple, de les utiliser comme des raquettes de tennis ou encore des clubs de golf (Figure 0.16). La multimodalité est incontournable dans les systèmes interactifs adaptés à tous, y compris aux personnes handicapées [Kirste 2001]. Des systèmes multimodaux pour le contrôle des dispositifs équipant les voitures tels que le GPS, le poste radio ou encore le téléphone sont aussi de plus en plus courants et déjà disponibles dans les véhicules de luxe [Pieraccini 2003]. Enfin, la recherche dans le domaine grand public est aussi très active. Des prototypes de recherche de réalité augmentée proposent différentes modalités combinées pour connaître l’orientation et la position de l’utilisateur et pour lui permettre de réaliser des commandes en situation mobile (Figure 0.17) [Feiner 1997] [Renevier 2004a].

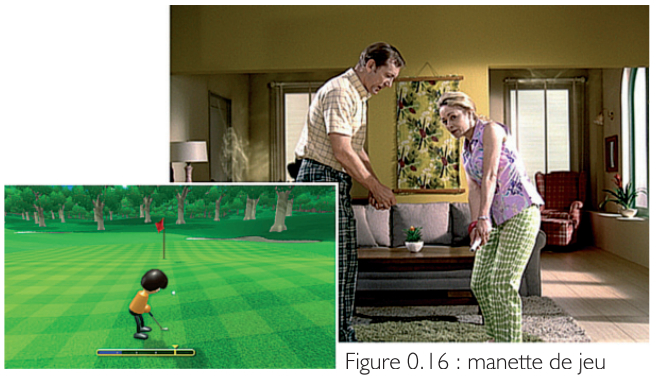


Figure 0.16 : manette de jeu Nintendo Wii avec capteurs de mouvement.

Figure 0.15 : console portable Nintendo DS avec écran tactile.

Figure 0.17 : équipement d’un utilisateur dans le système MARS [Feiner 1997].

2 Difficultés pour la conception et le développement des systèmes multimodaux

Tandis que de nombreux systèmes multimodaux sont aujourd'hui disponibles, leur conception ergonomique et logicielle reste néanmoins difficile. Alors que la mise en œuvre des systèmes interactifs avec un style d'interaction WIMP (Windows, Icon, Menu, Pointing devices) par le contrôle clavier ou souris est assez bien maîtrisée, la conception et le développement des systèmes interactifs incorporant d'autres modalités sont difficiles (voir l'encadré ci-dessous pour un rappel terminologique concernant les styles d'interaction WIMP et post-WIMP). Les systèmes multimodaux sont réalisés au cas par cas. La grande variabilité des modalités rendent les développements encore plus difficiles et coûteux. Les sections suivantes détaillent les principales difficultés rencontrées dans le processus de développement d'un système interactif multimodal.

Terminologie : les styles d'interaction WIMP et post-WIMP [Van Dam 1997]

WIMP pour Windows, Icon, Menu, Pointing devices est le nom donné au style d'interaction graphique qui utilise le système de fenêtrage, d'icônes et de menus manipulables à l'aide d'un pointeur. Le style d'interaction WIMP a été développé chez Xerox PARC en 1973 et a été popularisé par Macintosh en 1984. Copié ensuite par Windows, Unix et autres, il est aujourd'hui largement répandu et intégré dans les boîtes à outil de développement.

Le style d'interaction post-WIMP prend en compte les techniques d'interaction récentes qui utilisent par exemple des modalités telles que la reconnaissance vocale ou gestuelle. De nombreux travaux de recherche se focalisent aujourd'hui sur les interactions post-WIMP.

2.1 Processus de développement difficile et couteux

La multiplication des modalités d'interaction existantes accroît significativement les possibilités d'interaction pour un utilisateur, lui permettant de réaliser des tâches qu'il ne pouvait pas faire auparavant. A titre d'exemple, prenons le nombre de contrôles temps-réels sur le manche et la manette des gaz des avions militaires : alors qu'en 1970, il en existe 5 sur le joystick central du Harrier, le Rafale en intègre 33 aujourd'hui, répartis sur deux joysticks placés de chaque côté du pilote (Figure 0.18). Les systèmes multimodaux permettent ainsi d'effectuer un plus grand nombre de tâches.

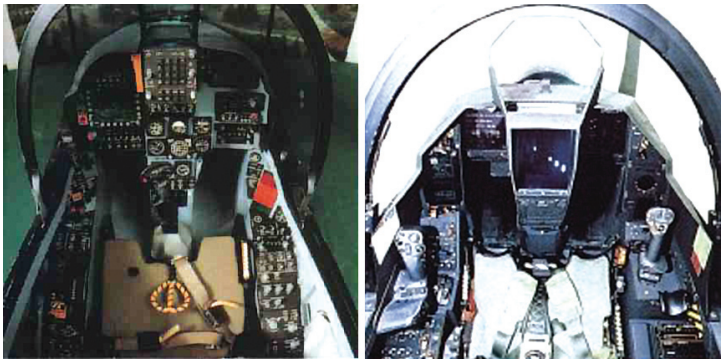


Figure 0.18 : cockpits du Harrier (à gauche) et du Rafale (à droite).

De plus, la grande variabilité des modalités d'interaction demande aujourd'hui un développement spécifique pour chaque système multimodal. En effet, alors que le développement des systèmes interactifs classiques utilisant la souris et le clavier est aujourd'hui maîtrisé, le domaine de l'ingénierie du logiciel doit proposer des solutions adaptées à la diversité des modalités d'interaction pouvant être développées.

Les systèmes multimodaux intégrant une variété de modalités d'interaction et les méthodes d'ingénierie n'étant pas arrivées à maturité, le processus de développement est aujourd'hui particulier à chaque réalisation. Pour chaque système interactif multimodal, le cycle de développement est long et le niveau d'expertise demandé est élevé et rare, entraînant une augmentation irrémédiable des coûts par rapport au développement de systèmes interactifs classiques.

2.2 Réutilisabilité de l'existant quasi-impossible

L'évolution des systèmes multimodaux, comme par exemple l'ajout d'une nouvelle modalité à un système existant, semble relever de l'impossible. En effet, la longue absence de modèles d'ingénierie a amené les entreprises développant de telles interfaces à réaliser des développements ad hoc, souvent sans documentation, comptant uniquement sur la présence de leurs ingénieurs qui avaient participé au développement. Quelques années après, les connaissances de l'architecture logicielle du système est bien souvent perdue avec le départ de ces ingénieurs et l'évolution du système multimodal est limitée voire impossible. De plus, la plupart des avancées conceptuelles et matérielles concernant les modalités remettent en cause les produits existants. Par exemple, lorsqu'il y a une évolution des techniques de programmation, la réutilisation de l'existant est très difficile à maîtriser et n'est souvent pas compatible avec les nouvelles modalités à mettre en œuvre. Au vue de ces constats, les entreprises effectuent de nouveaux développements complets.

2.3 Absence d'outils adaptés

Malgré les avancées significatives des travaux conceptuels concernant les modalités d'interaction ou leur combinaison (comme par exemple, dans le système "put-that-there" [Bolt 1980] qui combine geste et parole), les méthodes de conception et les outils de réalisation d'interfaces multimodales ne sont pas adaptés. Alors que de nombreux outils sont disponibles dans le domaine de l'IHM, les outils pour la multimodalité sont rares et peu pertinents pour le besoin actuel, notamment industriel. En effet, les outils existants s'adressent à des problèmes techniques

spécifiques tels que le mécanisme de combinaison de différentes modalités d'interaction [Bellik 1992] [Bourguet 2003] [Nigay 1994] ou ils sont dédiés à des modalités d'interaction spécifiques comme la parole et le geste [Cohen 1997] [Elting 2003]. Il n'existe pas aujourd'hui d'outil dédié aux systèmes multimodaux permettant de généraliser leur conception et leur développement en prenant en compte la diversité des cycles de développement logiciel industriels utilisés.

Les objectifs de ce travail de recherche ont été définis dans ce contexte. Ils sont présentés dans la section suivante.

Comme indiqué par le modèle de Gaines sur l'évolution du niveau d'apprentissage d'une technologie [Gaines 1991], le domaine de l'interaction multimodale commence aujourd'hui une nouvelle étape. Comme le montre la Figure 0.19, après la phase initiale de percée technologique, marquée notamment par les travaux de [Bolt 1980], l'interaction multimodale se positionne maintenant dans la phase de réplication. Comme nous venons de le voir, les systèmes multimodaux se multiplient. Néanmoins, leur conception et leur développement sont encore longs, difficiles et coûteux. Dans ce cadre, notre objectif vise à accélérer le processus de réplication afin de capitaliser les expériences produites, en définissant un modèle conceptuel comme un premier pas vers une théorie globale de la multimodalité. De plus, nous souhaitons aller encore plus loin et opérationnaliser notre modèle conceptuel par la définition d'un véritable outil de conception et de développement de systèmes multimodaux.

Pour répondre à ces objectifs, notre démarche de travail s'articule en trois phases : définition du modèle conceptuel, opérationnalisation et test.

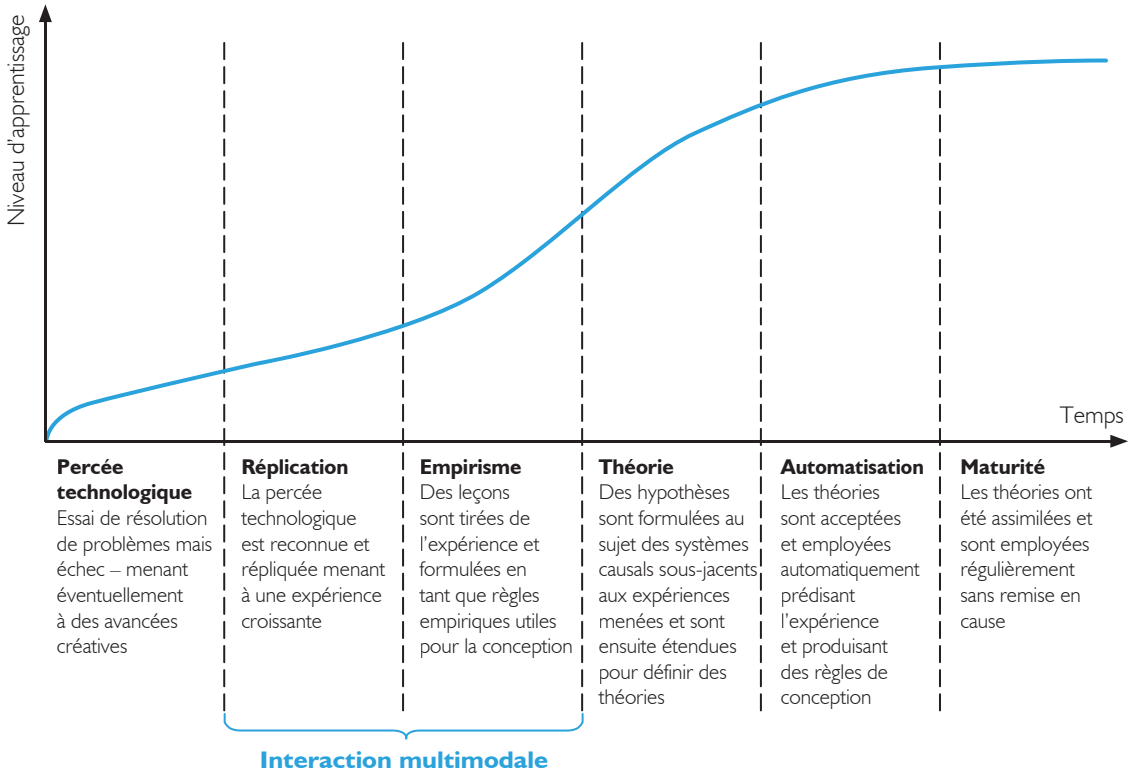


Figure 0.19 : le modèle de Gaines [Gaines 1991] sur l'évolution du niveau d'apprentissage d'une technologie, adapté à l'interaction multimodale.

3.1 Définition du modèle conceptuel : vers une théorie de la multimodalité

L'établissement d'un modèle conceptuel comme base d'une théorie sur la multimodalité est l'objectif principal de ce travail de recherche. Alors que nous disposons de plusieurs résultats sur le comportement humain et matériel, sur les techniques d'interaction et sur les usages des systèmes multimodaux, la réalisation de cette théorie de la multimodalité (cf. Figure 0.19) passe par la définition d'un modèle conceptuel unificateur des différents travaux existants. Cet objectif constitue la contribution majeure de ce travail de recherche et passe par l'étude des paradigmes d'interaction (WIMP et post-WIMP), des différents modèles ou espaces conceptuels de la multimodalité (côtés humain et technologique) et des usages multimodaux.

3.2 Opérationnalisation du modèle conceptuel dans un outil pour la multimodalité

Le second objectif est d'opérationnaliser le modèle conceptuel de la multimodalité dans un outil pour la conception et la réalisation d'interfaces multimodales. L'outil doit permettre de simplifier et d'accélérer le processus de création des interfaces multimodales. Les caractéristiques et le contenu de l'outil sont détaillés dans les deux sections suivantes.

Caractéristiques de l'outil

Les traits saillants de l'outil à concevoir sont :

- la généralité de l'outil d'intégration des modalités : l'objectif est ici de concevoir et de développer un outil permettant de combiner plusieurs modalités d'entrée ;
- le couplage des résultats de l'outil à des systèmes informatiques existants : l'interaction multimodale produite par l'outil doit pouvoir se connecter à un système informatique existant, comme montré à la Figure 0.1 ;
- la prise en compte des problèmes d'évolutivité et de réutilisabilité de l'interaction multimodale produite par l'outil est nécessaire ;
- l'adéquation avec les divers cycles de conception utilisés et notamment les cycles itératifs permettant le prototypage rapide, qui sont de plus en plus utilisés car ils permettent la définition d'interfaces utilisateurs de haute-qualité [Nielsen 1993] en réduisant le temps de développement [Myers 2000] ;

- la simplicité d'utilisation : les concepts manipulés doivent être simples et doivent pouvoir être manipulés par des utilisateurs non informaticiens ;
- un pouvoir d'expression élevé : la simplicité d'utilisation ne doit en aucun cas limiter le pouvoir d'expression de l'outil. Les diverses formes de multimodalité doivent pouvoir être traitées dans l'outil.

Contenu de l'outil

L'outil doit contenir :

- un modèle conceptuel cohérent des modalités et de leur combinaison ;
- un moteur de fusion général et paramétrable permettant de combiner les modalités d'interaction ;
- un ensemble de modalités d'entrée paramétrables et réutilisables ;
- un format d'échange entre les modalités de la plate-forme et les systèmes informatiques existantes ;
- une technique de développement récente, favorisant le suivi et la maintenance du développement par l'ensemble des acteurs du cycle de développement logiciel.

3.3 Test de l'approche par la conception et le développement de plusieurs systèmes interactifs

Le troisième et dernier objectif est le test du modèle conceptuel par l'utilisation de l'outil, pour concevoir et développer plusieurs systèmes multimodaux. L'outil doit permettre de réaliser plusieurs systèmes interactifs répondant à divers paradigmes d'interaction et différentes formes de combinaison inter-modalités. Des tests en situations d'usages réels par des acteurs industriels sont nécessaires pour valider complètement l'approche adoptée.

4 Organisation du Mémoire

L'organisation du mémoire reflète nos trois objectifs. Elle présente donc trois parties : espace conceptuel, opérationnalisation et réalisation logicielle.

La **Partie 1** définit l'espace conceptuel de ce travail de recherche. Il comprend quatre chapitres :

- Le chapitre 1 définit les nouveaux paradigmes d'interaction qui doivent être pris en compte.
- Le chapitre 2 fait le point sur les modèles des sciences cognitives et du domaine de l'IHM liés à la multimodalité.
- Le chapitre 3 présente les différents travaux sur les usages de la multimodalité.
- Le chapitre 4 propose un modèle conceptuel sur la multimodalité comme synthèse et unification des travaux présentés aux chapitres 1, 2 et 3.

La **Partie 2** expose l'opérationnalisation du modèle conceptuel qui est repris dans un outil de conception et de développement pour les systèmes multimodaux. Réalisée avec des composants logiciels, nous présentons cette opérationnalisation en quatre chapitres :

- Le chapitre 5 étudie les outils existants pour la multimodalité. Une grille d'analyse est établie et permet de comparer les différentes approches.
- Le chapitre 6 présente l'outil développé, appelé ICARE, en exposant ses principes généraux basés sur une approche à composants. Nous verrons que cet outil définit plusieurs types de composants qui peuvent être assemblés et qui constituent au final des mécanismes de fusion adaptés aux besoins du système interactif.
- Le chapitre 7 montre l'implémentation de l'outil ICARE dans une technologie à composants particulière (JavaBeans). Les différences entre technologies à composants sont abordées et le choix de JavaBeans est justifié. Enfin, l'environnement graphique de manipulation des composants d'ICARE est présenté.
- Le chapitre 8 est une synthèse des contributions de cette partie. L'outil ICARE est situé dans l'espace de classification des outils proposé au chapitre 5.

La **Partie 3** est dédiée aux réalisations logicielles avec l'outil ICARE. Pour chaque système interactif, différentes combinaisons entre modalités ont été réalisées et sont présentées. Cette partie comprend quatre chapitres :

- Le chapitre 9 présente le premier système interactif réalisé avec l'outil ICARE. Il s'agit d'un système d'identification d'individus selon plusieurs modalités d'interaction, appelé MID (Multimodal IDentification).

- Le chapitre 10 expose deux systèmes interactifs combinant geste et parole, comme dans le système “mets ça là” de [Bolt 1980]. Le premier d’entre eux concerne un système de recherche d’adresse appelée YellowPages. Le second, plus conséquent, est un prototype pour le contrôle de trafic aérien en vol.
- Le chapitre 11 développe deux systèmes interactifs proposant un paradigme d’interaction récent, la réalité augmentée. Le système MEMO est d’abord proposé. Il s’agit d’un logiciel grand public de réalité augmentée mobile où l’utilisateur manipule des post-its géo-localisés. Ensuite, l’interaction conçue et développée pour un simulateur d’avion de combat présentant des modalités d’interaction innovantes est décrite. Ce dernier système interactif, qui est le plus important en terme de développement, a été réalisé en collaboration avec THALES-Avionics propriétaire du simulateur.

En conclusion, les points contributifs sont soulignés par la synthèse des différents travaux de recherche présentés dans ce mémoire. Puis, les limites de l’approche adoptée sont rappelées. Enfin, de nombreuses perspectives à ce travail sont exposées.

Plusieurs annexes sont présentées à la fin du mémoire. L’annexe 1 regroupe deux exemples de programmation en Java : le premier concerne un composant Dispositif, le second, un composant Langage d’interaction. L’annexe 2 montre le code source des trois mécanismes de fusion génériques. L’annexe 3 expose l’implémentation et l’extension de notre modèle conceptuel pour des tests d’usages sur téléphone mobile. Enfin, l’annexe 4 développe une extension de notre travail par le couplage de notre outil ICARE avec l’outil Lutess, pour la réalisation de tests formels.

Partie 1

ESPACE DE CONCEPTION

VERS UNE THÉORIE DE LA MULTIMODALITÉ

Partie 1

ESPACE DE CONCEPTION
VERS UNE THÉORIE DE LA MULTIMODALITÉ

L'objectif de cette première partie est d'élaborer un espace conceptuel de l'interaction multimodale. Cet espace unifie les différents résultats sur le comportement humain et matériel, sur les techniques d'interaction et sur les usages des systèmes multimodaux, dans un modèle conceptuel servant de base pour une théorie de la multimodalité.

Les points clés de l'interaction multimodale résident dans la disponibilité de différents dispositifs, qui peuvent être utilisés conjointement, permettant à l'utilisateur de communiquer ses intentions au système informatique. Par exemple, l'application "put-that-there" de [Bolt 1980] illustre parfaitement une forme de combinaison complémentaire de deux modalités d'interaction, car l'utilisateur se sert conjointement de la parole et du geste pour le déplacement d'un objet graphique. En effet, l'utilisateur pointe du doigt l'objet voulu en prononçant "put that" et désigne sa destination en prononçant "there". Il est à noter que nous étudierons par la suite d'autres formes de combinaison. En conséquence, notre espace conceptuel doit fédérer les concepts multimodaux existants, fournissant un cadre unificateur pour la conception et le développement logiciels de n'importe quelle forme d'interaction multimodale.

La définition d'un canevas fédérateur répond à plusieurs buts, notamment à celui d'obtenir un système interactif final proposant tous les avantages des interfaces multimodales. La multimodalité offre aux différents types d'utilisateurs des alternatives pour l'interaction avec le système informatique. Cette flexibilité est exigée particulièrement dans des contextes variables d'utilisation [Zouinar 2003]. En effet, cette dernière étude souligne que la multimodalité est "*un moyen de régulation des variables contextuelles (contraintes environnementales, contraintes sociales, etc.) par l'adoption de la modalité la plus appropriée*". De plus, cette souplesse, définie par les différentes alternatives des modalités d'interaction, permet l'adaptation de l'interaction aux différences individuelles, comme par exemple lors d'un handicap permanent ou temporaire, ou encore aux tâches plus ou moins complexes [Oviatt 2000]. Enfin, comme montré dans [Cohen 2000], la multimodalité améliore, lors d'une conception et d'un développement adaptés, l'efficacité de l'interaction par rapport à l'utilisation d'une interface classique.

Sommaire >> >> >>

PARTIE 1
ESPACE DE CONCEPTION
VERS UNE THÉORIE DE LA MULTIMODALITÉ

Notre contribution, présentée au chapitre 4 de cette partie, doit permettre de capturer et de décrire les apports de la multimodalité. Pour ce faire, le chapitre 1 développe les nouveaux paradigmes d'interaction à prendre en compte. Les différents modèles existants sur la multimodalité sont ensuite étudiés au chapitre 2. Enfin, le chapitre 3 expose les travaux sur les usages des modalités d'interaction. L'analyse des travaux exposés dans ces trois premiers chapitres permet d'identifier les concepts fondamentaux et pertinents, utiles à l'élaboration de notre espace conceptuel (chapitre 4).

P. 91
P. 37
P. 45
P. 71

CHAPITRE 1 37
PARADIGMES D'INTERACTION RÉCENTS
1 Interfaces geste/parole 39
2 Interfaces tangibles 40
3 Interfaces de réalité augmentée 42
4 Résumé du chapitre 1 44

CHAPITRE 2 45
MODÈLES DE LA MULTIMODALITÉ
1 Introduction : les systèmes interactifs multimodaux 46
2 Utilisateur et multimodalité 48
3 Système informatique et multimodalité 55
4 Unification des modèles de psychologie cognitive et d'IHM 66
5 Résumé du chapitre 2 70

CHAPITRE 3 71
USAGES DES MODALITÉS D'INTERACTION
1 Résultats conceptuels 72
2 Résultats expérimentaux 80
3 Résumé du chapitre 3 89

CHAPITRE 4 91
MODÈLE CONCEPTUEL DE LA MULTIMODALITÉ :
UN PREMIER PAS VERS LA THEORIE
1 Méta-modèle 92
2 Modalités d'interaction 97
3 Combinaisons de modalités 111
4 Illustration du modèle conceptuel sur un exemple 116
5 Résumé du chapitre 4 120

CHAPITRE 1
PARADIGMES D'INTERACTION RÉCENTS

Depuis 1984 et leur apparition sur les Macintosh, les interfaces WIMP (interfaces classiques utilisant le système de fenêtrage, d'icônes et de menus manipulables à l'aide d'un pointeur, cf. encadré page 25) sont largement répandues et représentent la majorité des interfaces utilisées aujourd'hui. Le principe de ces interfaces se base sur la manipulation graphique des objets d'intérêts (appelée aussi manipulation directe [Shneiderman 1983]). Cependant, le futur est annoncé aux interfaces dites post-WIMP, qui reposent sur de nouveaux paradigmes d'interaction (voir l'encadré ci-dessous pour un rappel terminologique de paradigme d'interaction).

Terminologie : les paradigmes d'interaction

De façon générale [Larousse 2000], un paradigme est un modèle théorique de pensée qui oriente la recherche et la réflexion scientifiques. Pour le domaine de la psychologie, c'est une procédure méthodologique qui constitue un modèle de référence.

Pour [Dragicevic 2004], un paradigme d'interaction est un ensemble cohérent de techniques d'interaction qui coopèrent de façon étroite, ou qui reposent sur les mêmes principes techniques ou conceptuels.

Poussé par l'émergence de l'informatique ubiquitaire, dont le principe est que l'ensemble des ressources informatiques soit distribué dans une multitude d'objets nous entourant au quotidien [Weiser 1993], de nouveaux paradigmes d'interaction apparaissent afin que les utilisateurs tirent tous les bénéfices de ce nouveau contexte d'interaction. Avec les progrès des systèmes de localisation, de la miniaturisation des dispositifs, de la qualité des réseaux sans fils [Kangas 1999], le nombre de modalités d'interaction ne cesse d'augmenter. Ce chapitre décrit les paradigmes d'interaction récents qui font apparaître de nouveaux requis pour la multimodalité, notamment la mise en œuvre d'interfaces qui intègrent de plus en plus des éléments du monde physique dans ce qui était, jusqu'il y a peu, uniquement du numérique. La Figure 1.1 propose de classer les styles d'interfaces selon leur degré d'implication dans le monde physique : nous retrouvons, des moins impliquées au plus impliquées, les interfaces WIMP, les interfaces qui intègrent les modalités d'interaction naturelles

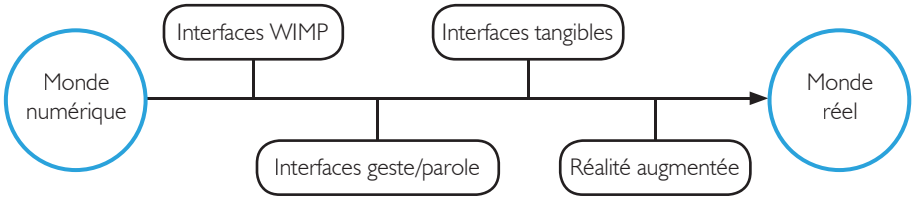


Figure 1.1 : accroissement de l'intégration du monde physique dans les interfaces.

du geste et de la parole, les interfaces tangibles et enfin les interfaces dites de réalité augmentée. Les interfaces WIMP incluent l'utilisation de la souris et du clavier qui se situe dans le monde physique. A ces deux modalités d'interaction, les interfaces geste/parole en ajoutent une nouvelle, la parole, elle aussi comprise dans le monde physique. Les interfaces tangibles s'impliquent encore plus dans le monde physique en définissant une multitude d'objets physiques couplés à des informations numériques. Enfin, nous considérons la réalité augmentée comme le paradigme qui est le plus engagé, car les tâches principales de ces interfaces se situent uniquement dans le monde physique.

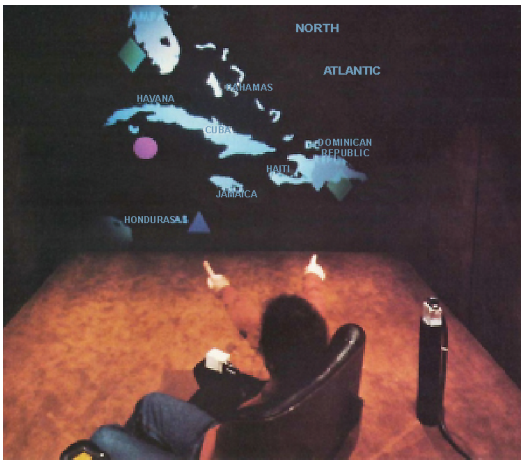
Le plan de ce chapitre est organisé selon cette figure. Les interfaces WIMP ne sont pas abordées ; seuls les paradigmes plus récents le sont, en commençant par les interfaces impliquant les modalités d'interaction naturelles du geste et de la parole.

Les interfaces de ce paradigme proposent de combiner les modalités d'interaction de reconnaissance vocale et gestuelle.

La première interface combinant ces deux modalités d'interaction est attribuée à l'application "put-that-there" ("mets ça là") de Richard Bolt [Bolt 1980]. Considéré aussi comme la première application multimodale, ce paradigme d'interaction est encore aujourd'hui considéré comme la référence de l'interaction multimodale. Comme le montre la Figure 1.2, pour déplacer un objet, l'utilisateur pointe du doigt l'objet voulu en prononçant "put that" et désigne sa destination en prononçant "there". La combinaison du geste et de la parole s'articule donc autour de la commande prononcée "put" ("mets") et des déictiques "that" ("ça") et "there" ("là"). A l'instar de [Cohen 1997] [Oviatt 2000] [Kaster 2003], beaucoup d'autres projets mettent en place et étudient ce paradigme. Il a donné lieu à de nombreuses études, comme celles des mécanismes de fusion des données provenant de modalités d'interaction différentes.

Figure 1.2 : le système "put-that-there" d'après [Bolt 1980].

Plusieurs applications utilisent ce paradigme avec des modalités d'interaction pseudo-naturelles. Ce paradigme est par exemple repris dans un simulateur de poste de commandement militaire informatisé et dans un système de recherche de données médicales [Cohen 1997]. Pour illustration, une langue pseudo-naturelle représente la sélection d'un sous-ensemble de mots-clés de la langue naturelle pour un système de reconnaissance vocale. Par exemple, des commandes simples permettent de contrôler l'environnement Windows (prononcez "start" et le menu démarrer s'ouvre). En fait, les langages d'interaction sont simplifiés à quelques mots-clés ou à des gestes simples dans le cas d'une interaction gestuelle. L'interaction est plus concise et ainsi plus adaptée à la tâche d'interaction [Baudel 1995].



Depuis son existence il y a 26 ans, beaucoup de travaux de recherche (voire la majorité des travaux sur la multimodalité) s'articulent autour du paradigme combinant la parole et le geste. Bien qu'il convient de ne pas réduire la multimodalité à la combinaison de la parole et du geste, ce paradigme a eu l'intérêt de mettre en évidence de nombreux concepts, en particulier sur la fusion des données provenant de modalités d'interaction différentes. Les modalités d'interaction peuvent être complètement naturelles ou constituées d'un sous ensemble (appelé pseudo-naturelles). La parole et le geste, sous une forme simplifiée (modalités pseudo-naturelles), présentent de meilleurs résultats (interaction simple, rapide et efficace) que l'utilisation des modalités naturelles.

D'après [Ishii 1997], les interfaces tangibles "*augmentent le monde physique en le couplant à des informations digitales (numériques) provenant de l'environnement ou des objets physiques de tous les jours*". Dans d'autres termes (et vu du sens opposé), les utilisateurs manipulent des objets physiques qui ont une influence dans le monde numérique. Dans la littérature, nous retrouvons plusieurs termes désignant les interfaces tangibles : interfaces incarnées ("embodied user interfaces" [Fishkin 2000]), interfaces saisissables ("graspable user interfaces" [Fitzmaurice 1995]), ou encore interfaces de manipulation ("manipulative user interfaces" [Harrison 1998]).

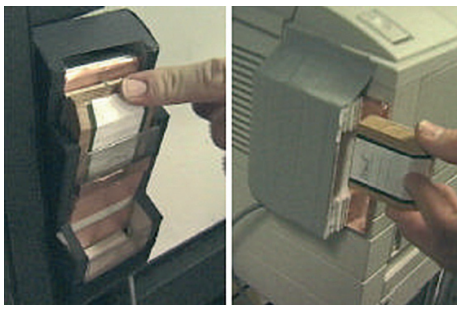


Figure 1.3 : les médiaBlocks d'après [Ullmer 1998].

Suite aux travaux de [Fitzmaurice 1995] et [Ishii 1997], de nombreuses applications sont développées et la plupart d'entre elles mettent en œuvre des objets physiques manipulables appelés aussi "phicons" (Physical Icons [Ullmer 1997]). Comme exemple d'applications, [Ullmer 1998] propose des MediaBlocks (phicons en forme de cube) permettant de contenir, transporter et contrôler des média en ligne. A la Figure 1.3, les Mediablocks sont des pièces en bois possédant un identificateur associé à une séquence d'un média en ligne. Parmi les diverses fonctionnalités proposées, il est par exemple possible de réaliser des copier/coller entre des sources de médias (ex : caméscope) et des dispositifs d'affichage (ex : vidéoprojecteur).

Plus récemment, [Ryokai 2004] présente un objet manipulable autre qu'un cube, un pinceau. Comme le montre la Figure 1.4, le pinceau est équipé d'une caméra avec des capteurs de contact. Le pinceau ainsi équipé permet de récupérer une couleur, une texture ou une animation (vidéo) capturée dans le monde physique pour s'en servir comme encre et l'appliquer sur une surface d'affichage numérique.

Les interfaces tangibles dites incarnées (embodied user interface) regroupent toutes les interfaces tangibles qui augmentent les stations de travail classiques par des métaphores du monde physique. [Fishkin 2000] propose plusieurs interactions possibles, comme la possibilité de tourner les pages lors d'une activité de lecture sur une tablette tactile, de

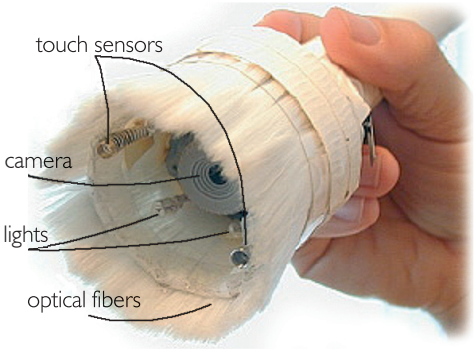


Figure 1.4 : le pinceau de I/O brush d'après [Ryokai 2004].

parcourir une liste de contacts par la simple inclinaison de son ordinateur de poche ou encore d'annoter des documents numériques par des commentaires écrits sur les ordinateurs possédant une surface tactile.

Une taxonomie des interfaces tangibles est décrite dans [Fishkin 2004]. Deux dimensions caractérisent les modalités d'interaction : le niveau d'incarnation et le niveau métaphorique. Le niveau d'incarnation d'une interface tangible se caractérise par la distance entre le focus d'entrée et celui de sortie. Ses valeurs peuvent être "total" ("full"), "rapproché" ("nearby"), "ambiant" ("environmental") ou "éloigné" ("distant"). Par exemple, le niveau d'incarnation du pinceau I/O brush [Ryokai 2004] est "rapproché" car le focus d'entrée se situe sur le pinceau, alors que le focus de sortie se trouve sur la surface peinte (c'est uniquement ici que l'encre numérique est visible). Le niveau métaphorique caractérise le niveau d'analogie des effets de l'interaction par rapport aux effets des actions similaires dans le monde physique. Il peut prendre les valeurs "aucun" ("none"), "nom" ou "verbe", "nom et verbe" ou "total" ("full"). La valeur "nom" se réfère à la forme d'un objet alors que la valeur "verbe" se rapporte aux actions avec un objet physique. Pour reprendre l'exemple du pinceau de [Ryokai 2004], le niveau métaphorique est "total" car l'utilisateur utilise son pinceau comme un pinceau classique.

Les interfaces tangibles représentent un paradigme de l'informatique ubiquitaire. En effet, le système interactif est en partie, voire complètement, intégré dans des objets communs, familiers des utilisateurs et facilitant son utilisation. Les interfaces tangibles sont donc caractérisées par une difficulté d'apprentissage faible, favorisant ainsi le développement d'interface intuitive. Les interfaces tangibles donnent ainsi naissance à de nombreuses modalités d'interaction, tout objet physique pouvant être support à l'interaction.

Le niveau d'incarnation et le niveau métaphorique définis dans [Fishkin 2004] représentent une piste intéressante pour la caractérisation des modalités d'interaction dans l'élaboration de notre modèle conceptuel.

Toujours dans le courant de l'informatique ubiquitaire, la réalité augmentée consiste à augmenter le monde physique par l'ajout d'informations numériques. Par rapport aux interfaces tangibles, la tâche principale des interfaces de réalité augmentée se situe dans le monde physique, les informations numériques sont fournies pour aider l'utilisateur à réaliser cette tâche. Les interfaces reposant sur ce paradigme ont pour défi la fusion harmonieuse des mondes physique et numérique [Renevier 2004b] : "*cette fusion consiste à unifier le monde numérique avec les activités des utilisateurs qui doivent se réaliser dans leur environnement physique*". Alors que l'enrichissement de l'environnement sonore est assez aisé avec les dispositifs de rendu existants (haut-parleurs, écouteurs, etc.), l'augmentation de la vision a donné lieu à de nouveaux dispositifs d'affichage permettant de visualiser le monde physique tout en superposant des images numériques. Ainsi, l'armée dispose depuis plusieurs années de HUD (Head Up Display ou affichage tête haute) qui équipe la majorité des avions de combat et qui permet d'afficher toutes les informations de vol nécessaires par transparence au dessus de la vision de l'environnement. Comme le montre la Figure 1.5, les mondes numérique et physique sont complètement superposés.

Figure 1.5 : HUD d'un F16.

Figure 1.6 : HUD d'une BMW.

Pour le grand public, les produits disponibles sont plus récents. Les voitures haut de gamme embarquent de plus en plus des HUD comme sur la voiture BMW présentée à la Figure 1.6, où les informations de vitesse ou encore de distance de freinage sont présentées sur le pare-brise.

Figure 1.7 : lunettes semi-transparente Glasstron de Sony.

Il existe aussi une solution pour les piétons, qui peuvent aujourd'hui s'équiper de lunettes semi-transparentes appelées HMD (Head Mounted Display) comme présentées à la Figure 1.7.

Le premier système de réalité augmentée est attribué à [Wellner 1993] et sa DigitalDesk (table digitale). L'intérêt de cette table est de manipuler des feuilles de papier et des crayons standards couplés aux fonctionnalités classiques du

monde numérique. L'utilisateur peut par exemple faire du copier/coller ou encore redimensionner ses dessins.

Plus récemment, le système MARS (Mobile Augmented Reality Systems) [Feiner1997] propose un système novateur pour la navigation. Cette application permet à des piétons munis de lunettes de réalité augmentée (Figure 1.7) de se renseigner sur les bâtiments du campus de la Columbia University. Des informations historiques ou simplement descriptives sur les bâtiments de ce campus sont disponibles et s'affichent dans les lunettes, en surimpression du bâtiment concerné.

Des travaux caractérisant les interfaces de réalité augmentée, nous retenons une caractéristique qui peut être appliquée à toutes les modalités d'interaction : il s'agit du lieu d'interaction identifié dans [Dubois 2001]. En entrée, le lieu d'interaction (appelé aussi lieu d'action) représente l'endroit géographique où l'utilisateur doit porter son attention pour pouvoir émettre des données à destination du système informatique. Bien entendu, le lieu d'interaction dépend de la position de l'utilisateur et sa valeur est donc dynamique. [Trevisan 2004] propose de caractériser le lieu d'interaction par la distance qui le sépare du sujet. Quatre zones sont ainsi définies : la zone centrale de 0 à 45 cm, la zone personnelle de 46 à 120 cm, la zone sociale de 121 à 360 cm et la zone publique au-delà de 360 cm. Par exemple, le lieu du DigitalDesk de [Wellner 1993] est la zone centrale.

Les interfaces de réalité augmentée fusionnent harmonieusement les mondes physique et numérique en les superposant. Ce paradigme cadre parfaitement avec les contraintes de mobilité car il permet d'effectuer une tâche dans le monde physique, tout en profitant avantageusement des outils informatiques. C'est le cas par exemple lors de la conduite d'un véhicule.

Pour la multimodalité, la réalité augmentée permet de mettre en œuvre des modalités d'interaction très intuitives, naturelles, concises et précises pour un utilisateur, car les objets numériques sont directement mis en situation dans le monde physique et ainsi rendus plus appréhendables. De plus, nous retenons que le lieu d'interaction identifié originellement pour les modalités d'interaction des interfaces de réalité augmentée dans [Dubois 2001] doit être pris en compte dans notre modèle conceptuel car il s'applique aussi à toutes les autres modalités d'interaction.

4 Résumé du chapitre 1

Poussé par le mouvement de l'informatique ubiquitaire (omniprésence de l'informatique dans les environnements), des nouveaux paradigmes d'interaction sont apparus. Ces paradigmes accroissent considérablement l'espace des possibilités des modalités d'interaction naturelles, intuitives et performantes. En effet, en intégrant des techniques d'interaction avec des objets communs ou dans un contexte naturel, la difficulté d'apprentissage pour l'utilisation des nouvelles modalités est considérablement réduite et l'interaction avec l'ordinateur moins complexe, donc plus efficace. Les interfaces combinant parole et geste incarnent la principale forme de communication entre humains et rendent l'interaction avec un ordinateur à portée de tous. Les interfaces tangibles proposent de manipuler des objets communs masquant en partie l'aspect numérique du dispositif et facilitant ainsi l'interaction avec le système informatique. Enfin, la réalité augmentée permet de faciliter les activités du monde physique en disposant de toutes les capacités (calculs, précision, etc.) des moyens d'interaction numériques.

En ce qui concerne la création de nouvelles modalités d'interaction, les interfaces tangibles et de réalité augmentée ouvrent un vaste champ de possibilités. Les interfaces tangibles définissent à elles seules des possibilités quasi-illimitées car tout objet physique peut être support à l'interaction. Pour la réalité augmentée, des modalités d'interaction spécifiques doivent être employées répondant à ce nouveau contexte. Par exemple, des modalités d'interaction renseignant sur la direction du regard et la localisation de l'utilisateur deviennent primordiales car la combinaison des données qu'elles transmettent permet de fusionner convenablement les mondes physique et numérique.

En résumé, les nouvelles formes de modalités d'interaction qui répondent à ces paradigmes constituent un important vivier en vue de définir des systèmes multimodaux novateurs. Dans notre objectif d'élaboration d'un modèle conceptuel de la multimodalité, les caractéristiques identifiées pour chacun des paradigmes présentés semblent fondamentales et leur intégration explicite doit être étudiée.

Alors que nous venons de présenter les paradigmes d'interaction récents que notre modèle conceptuel doit incorporer, le chapitre suivant expose les bases nécessaires pour atteindre cet objectif. Il traite des différents modèles existants sur la multimodalité.

CHAPITRE 2
MODÈLES DE LA MULTIMODALITÉ

D'après [Coutaz 2000] : "*L'Interaction Homme-Machine est un domaine d'étude des phénomènes cognitifs, matériels, logiciels et sociaux mis en jeu dans l'accomplissement de tâches avec un système informatique*". L'objectif de ce thème de recherche est de réaliser des systèmes interactifs en adéquation avec les besoins des utilisateurs (utilité) et en adéquation avec leurs capacités (utilisabilité). Cette dernière caractéristique doit permettre aux utilisateurs de réaliser leurs tâches avec un maximum de confort, d'efficacité, de sécurité et de qualité. Ainsi, l'IHM intègre plusieurs domaines de compétences tels que l'informatique, la psychologie cognitive, l'ergonomie et la sociologie. Avant de focaliser sur les usages de la multimodalité, traités dans le chapitre suivant, ce chapitre définit les principaux concepts des interfaces multimodales. La multimodalité y est étudiée selon deux points de vue complémentaires, l'un centré sur l'utilisateur, l'autre sur le système. Même si nos travaux se focalisent sur la multimodalité en entrée (de l'homme vers l'ordinateur), les modèles présentés englobent parfois les aspects entrée et sortie (voir l'encadré ci-dessous pour un rappel terminologique concernant les modèles). A chaque fin de section, nous recentrons les apports de ces modèles sur notre objet d'étude, la multimodalité en entrée. Nous concluons ce chapitre en visant l'unification des deux points de vue.

Terminologie : les modèles

D'après [Larousse 2000], un modèle est ce qui est donné pour servir de référence, de type. Dans d'autres termes, "*un objet A* est un modèle d'un objet A s'il permet de répondre aux questions que l'on se pose sur A*" [Minsky 1968].

1 Introduction : les systèmes interactifs multimodaux

L'utilisation d'un système informatique répond au modèle de Situation d'Activité Instrumentée (SAI) introduit en ergonomie dans [Rabardel 1995]. Cette situation compte trois entités liées (Figure 2.1) : le sujet, l'instrument et l'artefact. Alors que l'artefact correspond à l'objet matériel ou symbolique fabriqué, l'instrument représente le système technique permettant à l'utilisateur (le sujet) d'agir sur cet artefact. Dans notre étude, l'instrument correspond ainsi au système informatique utilisé par le sujet. Ce cadre théorique permet de localiser notre contribution. La conception et le développement d'une IHM multimodale fait partie de la définition de l'instrument. La création d'une IHM multimodale adéquate doit être la résultante d'une analyse de la relation sujet/instrument.

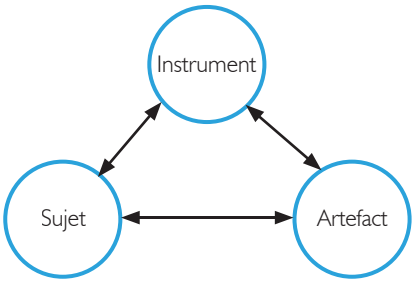


Figure 2.1 : le modèle SAI d'après [Rabardel 1995].

Notre étude se situe au niveau de l'interaction entre le sujet et l'instrument qui, dans notre cas, se résume au couple utilisateur/système informatique. Ce couple, que nous appelons "système interactif", doit être affiné. La notion d'utilisateur doit être prise au sens large car notre analyse doit aussi bien prendre en compte les différentes caractéristiques de chaque utilisateur (capacités physiques, mentales, etc.) que les informations dynamiques qui le lient à son environnement (sa position géographique, l'orientation de son regard, le type de lieux dans lequel elle/il évolue, etc.). En ce qui concerne le système informatique, nous le décomposons en deux modules indépendants : l'interface et le noyau fonctionnel. La Figure 2.2 rappelle le rôle de la partie Interface dans un système interactif [Nigay 1994].

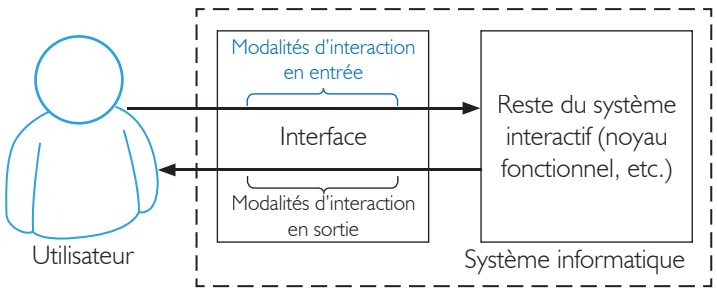


Figure 2.2 : système interactif = utilisateur + système informatique d'après [Nigay 1994].

L'interface correspond à la partie perceptible de l'iceberg. Elle permet à l'utilisateur d'interagir avec le reste du système informatique et notamment avec le noyau fonctionnel propre au domaine applicatif. Ce niveau de décomposition nous suffit pour dessiner les contours de l'interface multimodale devant être élaborée et nécessitant l'intégration des relations avec l'utilisateur et le noyau fonctionnel. La relation entre l'interface multimodale et le noyau fonctionnel est assez bien maîtrisée et de nombreuses solutions sont disponibles, comme celles proposées par [Nigay 1994] et [Bastide 2005]. L'enjeu du domaine de l'interaction multimodale est aujourd'hui de prendre en compte les dimensions humaines et technologiques pour une conception et un développement logiciel de bonne qualité.

Plusieurs modèles dédiés à la multimodalité existent. Ils sont ici regroupés en deux catégories : les modèles centrés utilisateur et ceux centrés système informatique. La section suivante expose les principaux concepts de la multimodalité du point de vue humain.

Le domaine de la psychologie cognitive a permis d'apporter à l'IHM de nombreux modèles permettant de prédire et d'expliquer le comportement humain. Ces modèles constituent de véritables outils pour prendre en compte toute la dimension humaine dans la conception des systèmes interactifs. Nous en présentons plusieurs qui posent le cadre de notre étude anthropocentrée de la multimodalité.

Le psychologue/ergonome Jacques Leplat [Leplat 2000] définit un utilisateur comme un système de traitement qui "*pour répondre aux buts et exigences de la tâche, met en œuvre des moyens, instruments et ressources qui sont ceux de l'homme en général (comme la vision, la mémoire) et ceux qui ont été acquis au cours de la vie (professionnelle et autre)*". Il propose d'aborder l'étude de l'utilisateur selon plusieurs points de vue, dont nous nous inspirons, pour illustrer dans les sections suivantes les multiples mécanismes actionnels, perceptuels et cognitifs mis en œuvre lorsqu'un utilisateur interagit avec un système informatique via une interface multimodale. Nous présentons d'abord la théorie de l'action [Norman 1986] comme un cadre unificateur des autres modèles. En effet, en nous reposant sur la théorie de l'action, nous présentons les modèles centrés utilisateur selon leur niveau d'abstraction. Nous commençons par présenter les modèles traitant des éléments actionnels et perceptuels de l'humain pour terminer avec les modèles sur le fonctionnement cognitif humain.

2.1 THÉORIE DE L'ACTION

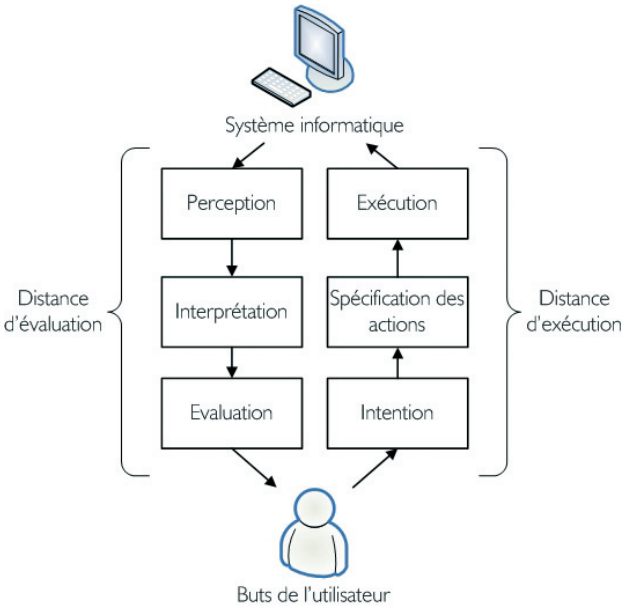


Figure 2.3 : théorie de l'action de Norman d'après [Norman 1986].

La théorie de l'action [Norman 1986] fournit un cadre fédérateur pour les modèles centrés sur l'utilisateur. La théorie décompose l'accomplissement d'une tâche par l'utilisateur en sept étapes. Pour réaliser son but, l'utilisateur fait un effort pour mettre en correspondance la représentation mentale qu'il a de sa tâche et la représentation physique de cette tâche pour le système informatique. Cet effort est appelé "distance d'exécution". Ensuite, le système réagit et l'utilisateur doit à nouveau faire un effort cognitif correspondant au processus inverse du processus d'exécution. Ce dernier est appelé "distance d'évaluation". Comme le montre la Figure 2.3, la distance d'exécution et la distance d'évaluation se décomposent en trois étapes.

La théorie de l'action est générale et fournit un cadre unificateur pour les autres modèles. Pour la multimodalité, elle met en évidence l'étape de choix des modalités en entrée (étape Exécution) à partir d'une représentation mentale des actions à mener, et celle de perception liée aux modalités employées en sortie par le système. A titre illustratif de la multimodalité, considérons une tâche de suppression d'un fichier sous Windows avec un système multimodal. L'utilisateur spécifie mentalement les actions à effectuer, il doit notamment sélectionner le fichier désiré puis spécifier sa suppression et enfin confirmer son effacement définitif. Nous considérons que le système multimodal permet à l'utilisateur deux formes d'exécution différentes. La première exécution, tout à fait classique, reprend la manipulation directe des icônes par la souris (interface WIMP) : l'utilisateur sélectionne l'icône du fichier désiré (clic gauche de la souris), réalise un glisser-déposer de ce fichier vers l'icône de la corbeille puis supprime le contenu de la corbeille par la sélection de cette fonctionnalité dans la liste présentée suite à un clic droit sur l'icône de la corbeille. Enfin, il confirme la suppression par un clic gauche sur le bouton correspondant de la boîte de dialogue. La deuxième exécution possible propose un paradigme geste/parole de type "put-that-there" [Bolt 1980] présenté au chapitre précédent. L'utilisateur désigne par le geste l'icône du fichier désiré (il le pointe avec son doigt), et prononce simultanément le mot-clé "supprimer". Une boîte de dialogue apparaît demandant la confirmation de cette suppression que l'utilisateur réalise en prononçant un "oui". Par cet exemple, nous montrons que pour une intention similaire, la multimodalité peut offrir plusieurs formes d'exécution impliquant des distances d'exécution et d'évaluation différentes.

2.2 Modes de communication

Pour caractériser l'interaction entre l'utilisateur et le système informatique, [Bellik 1995] introduit la notion de mode de communication. Cette notion se rapporte aux organes utilisés par un utilisateur lors de l'interaction avec un système informatique. Elle est l'équivalent des sens humains de l'interaction en sortie (vision, ouïe, odorat, toucher, gout) pour l'interaction en entrée. [Bellik 1995] distingue deux modes : le mode gestuel et le mode oral.

- Le mode gestuel correspond aux différents organes qui permettent, par mise en action de tout ou de certaines parties du corps (mains, bras, tête, visage, etc.) de transmettre des informations à un système. C'est de loin, le mode qui est le plus largement utilisé aujourd'hui pour interagir avec un ordinateur.
- Le mode oral correspond aux différents organes (poumons, cordes vocales, langue, lèvres, etc.) qui interviennent dans la production de sons (parole, bruits, sifflements etc.).

Au vue des dernières avancées concernant le domaine des interfaces cerveau-ordinateur, appelé aussi BCI (Brain Computer Interface), nous pouvons ajouter un autre mode de communication, le mode mental, où la mesure de l'activité cérébrale d'un utilisateur permet de déduire ses intentions à l'ordinateur. L'analyse des ondes cérébrales du cerveau est un nouveau moyen de communication mis en œuvre dans diverses applications, comme par exemple pour les jeux vidéo [Krepi 2004], pour la navigation internet [Tomori 2003] ou pour faciliter l'interaction et le contrôle de son environnement lorsque l'on présente un handicap [Adams 2003].

Les modes de communication permettent de caractériser l'interaction entre l'homme et le système informatique du point de vue des organes mis en jeu. Nous en distinguons trois : le mode gestuel, oral et mental.

2.3 Etat de l'organisme et personnalité

L'interaction est aussi influencée par l'état de l'organisme et la personnalité que nous pouvons caractériser par un ensemble de facteurs, variables temporellement, regroupés en trois catégories [Tyndiuk 2003] :

- les facteurs biologiques, comme l'âge, le sexe, la taille, les capacités physiques, etc.
- les facteurs socioculturels regroupant les aspects comme l'ethnicité, le niveau d'études, le niveau de formation professionnelle, etc.
- les facteurs idiosyncrasiques qui correspondent au caractère individuel, tempérament personnel [Robert 1998] et qui regroupent le niveau de stress, la motivation, etc.

L'état de l'organisme et la personnalité ont une répercussion directe sur l'usage des modalités pour interagir avec un système informatique par un utilisateur. Par exemple pour les facteurs biologiques, les téléphones portables présentant des touches très petites sont quasiment inutilisables par les personnes très âgées. Pour les facteurs socioculturels, un interlocuteur de nationalité russe, habitué à l'écriture cyrillique, est bien moins performant dans l'utilisation d'un clavier azerty. Enfin, pour les facteurs idiosyncrasiques, un niveau de stress élevé modifie l'intonation de la voix de l'utilisateur et peut rendre un système de reconnaissance vocale inutilisable.

2.4 Gestion des ressources

L'utilisation d'un ordinateur via les différentes modalités disponibles implique, chez l'utilisateur, des ressources. Les ressources peuvent être physiques ou mentales. Elles dépendent des facteurs biologiques, socioculturels et idiosyncrasiques présentés dans la section précédente.

Chez un utilisateur, les ressources entraînent un coût physique et cognitif [Leplat 2000]. Notre intérêt se porte ici sur la gestion de ces ressources, en d'autres mots sur la gestion de la charge mentale et physique de l'utilisateur. La Figure 2.4, issue de [Leplat 2000] et particularisée à la multimodalité, montre l'utilisation conjointe de deux modalités. Les axes représentent les niveaux de performance pour l'utilisation des modalités X et Y. Le point A représente le cas où les deux modalités sont utilisées avec une performance maximale. La courbe pleine représente le cas où les deux modalités sont utilisées avec une performance réduite. Un point de la courbe (par exemple le point B) représente le niveau de performance obtenu pour l'utilisation conjointe des deux modalités. L'objectif est ici de trouver le meilleur niveau de performance, appelé "compromis cognitif" dans [Amalberti 1996], permettant à un utilisateur de réaliser son but avec une charge acceptable. Ce point B doit se rapprocher le plus possible des axes en pointillés et du point A.

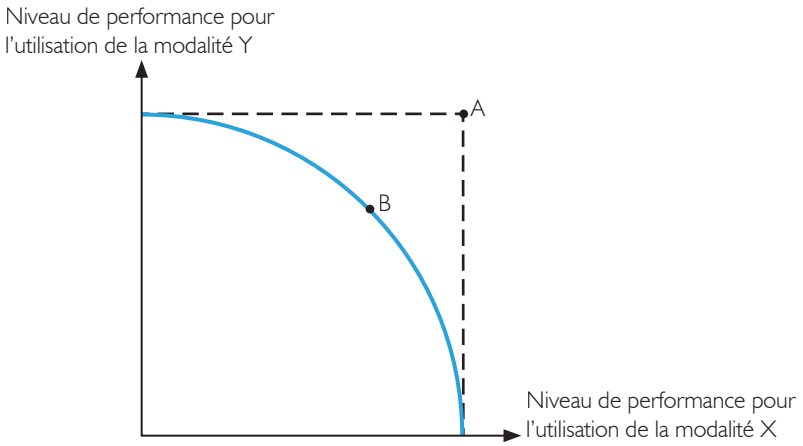


Figure 2.4 : utilisation conjointe de deux modalités, d'après la gestion des ressources de [Leplat 2000].

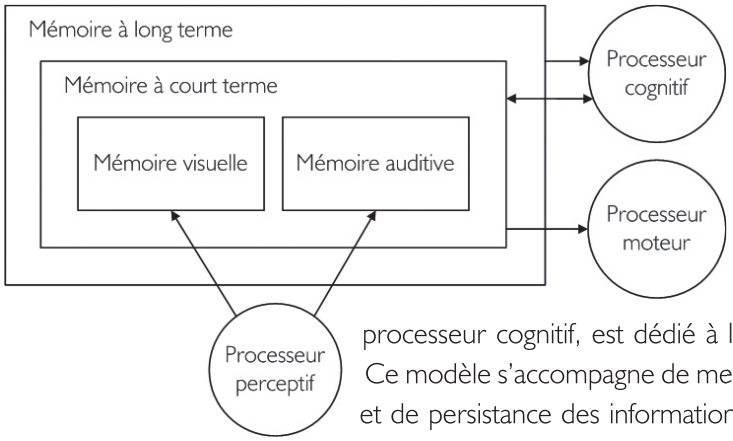
La gestion des ressources semble être un point essentiel que l'étude de la multimodalité doit prendre en compte. Elle permet d'introduire la notion de compromis cognitif [Amalberti 1996] lors de l'utilisation conjointe de plusieurs modalités. L'interface présentant plusieurs modalités, l'objectif de conception consiste alors à identifier le meilleur compromis cognitif pour l'utilisateur lors de l'utilisation simultanée de modalités afin qu'elle/il réalise son but efficacement.

2.5 Fonctionnement cognitif de base

Le terme cognition est défini dans [Montmollin 1995] comme "*l'ensemble des activités et des processus par lesquels un organisme acquiert de l'information, la traite, la conserve et l'exploite*". Différents modèles ont été établis. Nous présentons les travaux qui nous permettent d'appréhender simplement le fonctionnement cognitif de l'utilisateur lors de l'utilisation d'un système multimodal.

Le modèle du processeur humain [Card 1983] offre, dans une terminologie informaticienne, un cadre fédérateur pour les différentes connaissances en psychologie. Ce modèle, présenté à la Figure 2.5, représente l'utilisateur comme un système de traitement de l'information composé de trois sous-systèmes interdépendants : le système sensoriel, le système moteur et le système cognitif. Chaque sous-système comprend un processeur et une mémoire. Le système sensoriel permet de traiter un phénomène sensible (ou stimulus). Le système moteur est responsable des mouvements. Le système cognitif, composé de la mémoire à court terme, de la mémoire à long terme et d'un processeur cognitif, est dédié à la prise de décision et à la planification. Ce modèle s'accompagne de mesures des différents temps de traitement et de persistance des informations dans les sous-systèmes. Par exemple pour le processeur moteur, le mouvement qui correspond à la manipulation d'un dispositif physique se décompose en une suite de micro mouvements dont chacun prend en moyenne 70 ms.

Figure 2.5 : modèle du processeur humain d'après [Card 1983].



Sur le même principe, le modèle ICS (Interacting Cognitive Subsystems) [Barnard 1987] affine le modèle du processeur humain en présentant une structure modulaire plus détaillée. Présenté à la Figure 2.6, ICS distingue en effet les trois processeurs (perceptif, cognitif, moteur) de [Card 1983] et les décompose en neuf sous-systèmes travaillant de manière parallèle.

Le processeur perceptif est décomposé en trois sous-systèmes liés aux sens perceptifs humains : acoustique, visuel et état physique (état du corps : équilibre, etc.).

Le processeur moteur est décomposé en deux sous-systèmes : le sous-système articulatoire permet de bouger les muscles pour la production de son et le sous-système mouvement s'occupe de tous les autres gestes.

Le processeur cognitif est décomposé en quatre sous-systèmes : le sous-système propositionnel traite des connaissances a priori sur le monde (identité des objets, relations entre objets, etc.) ; le sous-système implicationnel maintient des connaissances a posteriori sur le monde (sentiments, impressions, etc.) ; le sous-système morphono-lexical interprète les sons provenant du sous-système acoustique ; enfin le sous-système objet permet d'interpréter les images fournies par le sous-système visuel.

Le principe d'application est d'identifier les sous-systèmes impliqués en suivant les informations transitant dans le modèle ICS. Par exemple, la suppression d'une phrase dans un éditeur de texte met en jeu plusieurs sous-systèmes. Tout d'abord, la scène est perçue dans le sous-système visuel. Le sous-système objet permet sa représentation spatiale. Ensuite, le sous-système morphono-lexical réalise une description structurelle de la phrase qui acquière un sens (description sémantique) dans le sous-système propositionnel. Enfin, le sous-système implicationnel permet la composition de schémas de résolution qui décide ici de l'élimination de la phrase. Enfin, la phrase est effectivement supprimée par le sous-système mouvement qui coordonne les mouvements du corps.

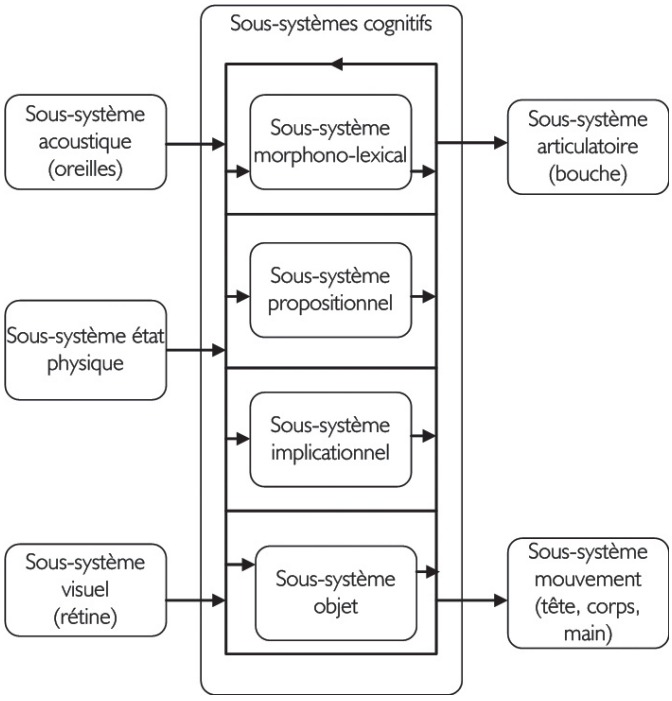


Figure 2.6 : le modèle ICS [Barnard 1987].

Pour la multimodalité, les modèles du processeur humain et ICS permettent de mettre en évidence les différents traitements (étapes) et contraintes liés à l'usage des modalités d'interaction par un humain. Ils fournissent un cadre théorique pour la mesure des temps de traitement moyens pour chaque processeur ou sous-systèmes lors de l'utilisation d'une modalité d'interaction et mettent en évidence les dépendances entre les différents modules perceptifs, cognitifs et moteurs.

2.6 Compétences

Les compétences sont définies dans [Montmollin 1986] comme "*les ensembles stabilisés de savoirs et de savoir-faire, de conduites-types, de procédures-standards,*

de types de raisonnement, que l'on peut mettre en œuvre sans apprentissage nouveau". D'après [Leplat 2000] inspiré de [Rogalski 1995], les compétences regroupent l'ensemble des connaissances concernant le domaine de travail, le mode d'exécution, l'usage des aides et l'expérience. Dans ce sens, le modèle SRK (Skill, Rule, Knowledge) de Rasmussen [Rasmussen 1986] décompose les comportements humains selon trois niveaux de contrôle :

- les comportements basés sur les réflexes (skills) ;
- les comportements basés sur les règles (rules) ;
- les comportements basés sur les connaissances (knowledge).

Pour illustrer les différences de ces comportements, nous considérons un scénario où un utilisateur doit aller d'un point A vers un point B. Si l'utilisateur fait plusieurs fois par jour le chemin de A vers B, il parcourt la route sans réfléchir au chemin qu'il prend. Son comportement est basé sur les réflexes et il ne demande pas d'effort cognitif de sa part. Dans un autre cas de figure, si l'utilisateur a réalisé seulement deux fois le trajet de A vers B, selon deux routes différentes, alors il va suivre l'une des deux routes qu'il a déjà empruntée. Son comportement est basé sur les règles. Enfin, si l'utilisateur n'a jamais fait le trajet de A vers B, il doit planifier son chemin en se basant uniquement sur les connaissances du terrain qu'il détient (comme une carte par exemple). Son comportement est basé sur les connaissances et cela lui demande un effort cognitif important.

Ainsi, Rasmussen fournit un cadre pour la modélisation de l'utilisateur en identifiant trois classes d'utilisateurs : les experts (comportements basés sur les réflexes), les intermédiaires (comportements basés sur les règles) et les novices (comportements basés sur les connaissances).

Les niveaux d'expertise de l'utilisateur sont pertinents pour expliquer les choix des modalités d'interaction et les formes de multimodalité que l'utilisateur exploite selon son niveau de compétence. La prise en compte des compétences de l'utilisateur pour la conception et le développement d'interfaces multimodales pertinentes pour un utilisateur est incontournable. [Oviatt 2004] démontre notamment que plus la tâche est complexe, plus l'utilisateur combine les modalités d'interaction.

3 Système informatique et multimodalité

Au contraire des sciences humaines, qui ont un point de vue centré utilisateur, cette section est centrée sur la technologie. L'absence de consensus parmi la communauté scientifique IHM sur la définition des termes "modalité" et "multimodalité" nous incite à lever l'ambiguïté de leurs définitions. Ensuite, de façon complémentaire aux modèles des sciences cognitives présentés précédemment, nous étudions plusieurs modèles de l'interaction homme-machine décrivant les traitements informatiques concernant la multimodalité. Les modèles sont organisés selon leur niveau d'abstraction, du plus concret au plus abstrait.

3.1 Terminologie

3.1.1 Modalité

Les modalités d'interaction en entrée sont les moyens d'action entre l'utilisateur et le reste du système informatique. Alors qu'il n'existe pas de consensus sur la définition de ce terme, notre choix a été dirigé par l'étude précédente des modèles centrés humain qui faisait apparaître deux niveaux d'abstraction : le niveau actionnel et le niveau cognitif. Plus précisément, nous adoptons, pour nos travaux, la définition de [Nigay 1996] où une modalité d'interaction est définie par un couple <d, L> où d désigne un dispositif physique et L, un langage d'interaction.

- Un *dispositif physique* est un élément du système qui acquière des informations. C'est le plus bas niveau d'abstraction de la modalité. Des exemples de dispositifs incluent un clavier, une souris, un microphone, un GPS ou encore un magnétomètre.
- Un *langage d'interaction* définit un ensemble d'expressions bien formées et significatives (par exemple, un assemblage conventionnel de symboles). La génération d'un symbole, ou d'un ensemble de symboles, résulte d'actions sur les dispositifs physiques d'entrée. De plus haut niveau d'abstraction que le dispositif, il représente le niveau logique de la modalité. Des exemples de langages d'interaction incluent le langage pseudo-naturel, la manipulation directe ou encore la localisation.

Une modalité d'entrée comme la parole peut être décrite par le couple <microphone, langage pseudo-naturel>, où le langage pseudo-naturel est défini par une grammaire spécifique. De façon similaire, la modalité de localisation d'un utilisateur peut être décrite par le couple <GPS, localisation dans un référentiel>. Comme montré à la Figure 2.7, cette définition de modalité caractérise les échanges entre le système et l'utilisateur, en mettant en relation deux niveaux d'abstraction : le niveau physique (dispositif) et le niveau logique (langage d'interaction).

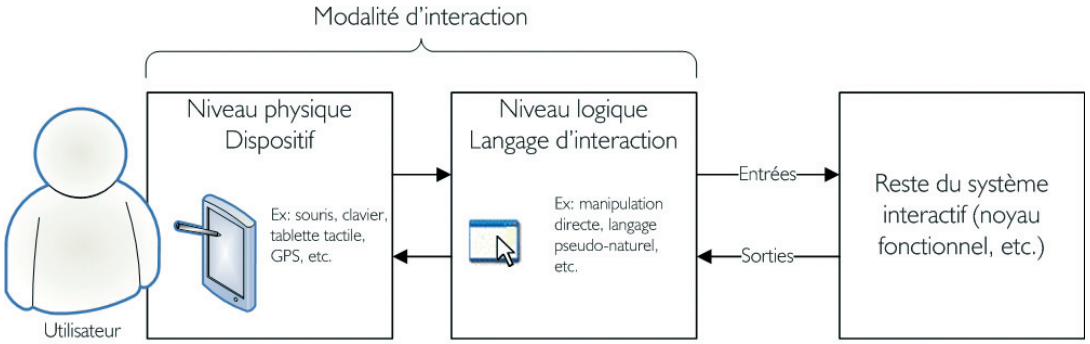


Figure 2.7 : niveaux physique et logique d'une modalité d'interaction d'après [Vernier 2001].

3.1.2 Multimodalité

En reprenant la définition de la modalité donnée au paragraphe précédent, la multimodalité reflète le caractère de multiplicité (préfixe multi) des modalités pour un même système interactif. Le nombre de modalités permet de faire la distinction entre la multiplicité et l'unicité. Un système est multimodal s'il dispose d'au moins deux modalités pour un sens donné (entrée ou sortie). Ainsi, le système peut être qualifié de multimodal en entrée si au moins deux modalités d'entrée sont disponibles.

Depuis les travaux de [Bolt 1980] sur le paradigme "put-that-there", plusieurs modèles conceptuels sur la multimodalité ont été établis. Les sections suivantes présentent les principaux modèles.

3.2 Modèles de type "Acquisition – Traitement – Rendu"

3.2.1 Cadre général

Comme pour le processeur humain de [Card 1983] (cf. section 2.5 de chapitre), le système informatique peut être décomposé selon trois modules (cf. Figure 2.8 de ce chapitre) : le module acquisition des données de l'utilisateur, le module de traitement de ces données et le module de rendu des résultats à l'utilisateur. Cette décomposition en trois facettes du système informatique permet une séparation des préoccupations que nous retrouvons aussi bien au niveau des domaines de recherche que des développements logiciels, comme par exemple dans [Bastide 2005], où la conception et le développement d'un système sont motivés par cette séparation modulaire. Comme autre exemple, le modèle d'architecture logicielle MVC (Modèle, Vue, Contrôleur) [Goldberg 1984] propose lui aussi de décomposer les systèmes interactifs en une hiérarchie d'agents selon trois

facettes : le modèle, la vue, le contrôleur. La facette modèle est le noyau fonctionnel de l'application. Il permet de traiter les données en réaction aux événements déclenchés par l'utilisateur et captés par la facette contrôleur. Enfin, la facette vue s'occupe du rendu perceptible des données de la facette modèle.

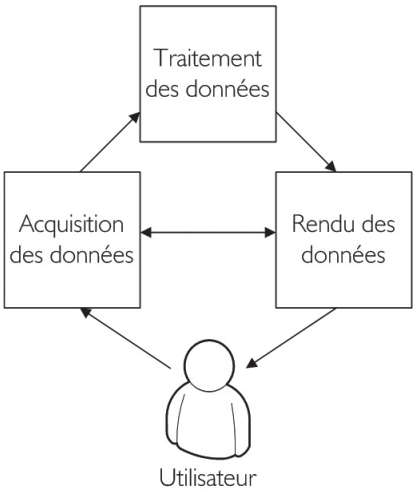


Figure 2.8 : cadre général : les 3 modules d'un système informatique.

3.2.2 Modèle Pipeline

[Nigay 1994] propose le modèle Pipeline qui affine la vue générale présentée précédemment. Il concilie l'aspect entrée/sortie des données et la prise en considération de l'utilisateur et du système informatique. En effet, la Figure 2.9 illustre les différentes transformations des données lors de l'utilisation d'une interface. Par exemple, l'utilisateur traduit une intention en une ou plusieurs actions physiques, au travers d'une ou plusieurs modalités. Ces actions sont interprétées et le résultat forme une ou plusieurs unités informationnelles. A partir de ces unités informationnelles, une ou plusieurs actions système sont identifiées puis exécutées par le noyau fonctionnel. En retour, le noyau fonctionnel spécifie la ou les unités informationnelles représentant les effets de l'action. Ces unités informationnelles sont interprétées et forment une suite d'actions physiques, qui sont effectivement rendues perceptibles par l'utilisateur grâce aux dispositifs physiques de sortie.

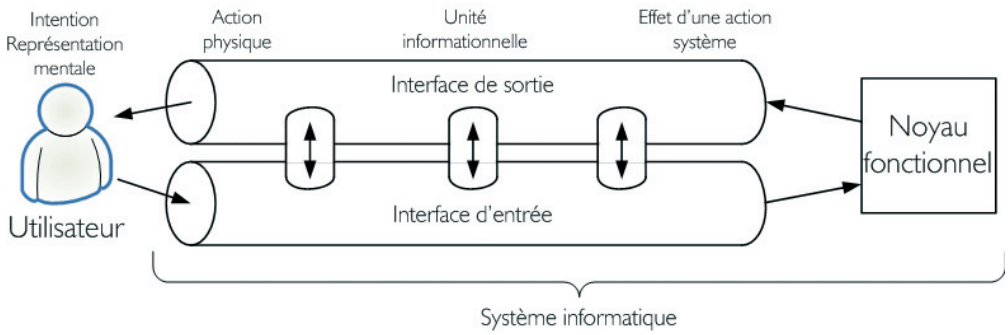


Figure 2.9 : le modèle Pipeline d'après [Nigay 1994].

La décomposition Pipeline permet de localiser précisément nos travaux. Les modalités d'entrée concernent le module d'acquisition des données. Le modèle Pipeline affine ce module en proposant différentes étapes d'abstraction des événements déclenchés par l'utilisateur. Tous les modèles présentés dans les sections suivantes ont trait à l'acquisition des événements de l'utilisateur. Parmi ces modèles, certains adoptent un point de vue générale sur l'interaction en considérant les entrées et les sorties.

3.3 Modèle des dispositifs physiques de [Buxton 1983]

Cet espace caractérise les dispositifs d'entrée selon trois axes. Dans cette taxinomie, le dispositif est défini comme un transducteur.

Le premier axe rassemble les propriétés que le dispositif est capable de capter comme la position, la pression et le mouvement. Le second axe définit le nombre de dimensions captées pour chaque propriété. Par exemple, pour un joystick, la position est déterminée par trois valeurs prises dans un système de coordonnées. Le troisième axe distingue le type direct ou indirect de la capture. Par exemple, pour un écran tactile, la mesure est directe alors que pour la souris, l'utilisateur manipule le dispositif physique qui a une représentation à l'écran. Dans ce cas, le type est indirect. Les deux premiers axes de cette taxonomie définissent les degrés de liberté du transducteur dans l'espace. Ils expriment la capacité du dispositif à traduire les actions de l'utilisateur en une forme exploitable par le système informatique. La richesse du dispositif est alors fonction du nombre de propriétés physiques captées et pour chaque propriété, le nombre de dimensions spatiales. Le type direct ou indirect de la capture traduit la présence ou non d'un instrument intermédiaire pour interagir avec l'objet d'intérêt. Cependant, cet espace ne concerne que les dispositifs actionnables par le geste. Les dispositifs qui ne répondent pas à ces conditions, comme par exemple un microphone, ne sont pas représentables avec cette taxinomie.

Le modèle de [Buxton 1983] met en évidence plusieurs caractéristiques générales aux dispositifs manipulables : le nombre de dimensions captées et le type direct ou indirect de cette capture.

3.4 Espace de Card, Mackinlay et Robertson [Card 1990]

Cet espace concerne aussi les dispositifs physiques et complète l'approche de Buxton. La taxinomie permet d'exprimer la variabilité des niveaux d'abstraction pour un même dispositif physique, en considérant l'assemblage de dispositifs élémentaires en unités de contrôle plus complexes. Dans cet espace, les dispositifs sont définis par un sextuplet <M, In, Out, R, S, W> où M représente un opérateur de manipulation appliqué par l'utilisateur, In représente le domaine des valeurs d'entrée possibles de M, Out représente le domaine des valeurs de sortie possibles du dispositif, R définit la correspondance entre les domaines In et Out, S

dénote l'état actuel du système (In ou Out), W désigne le fonctionnement externe et interne du dispositif physique. Par exemple, pour un bouton rotatif permettant de contrôler la vitesse d'une voiture radiocommandée :

BoutonRotatif =
< M = Ry (rotation autour de l'axe y)
In = [0,180]
Out = {0, 10, 20, 30, 40}
R = [0,45] -> 0,]45,90]->10, etc.
S = In ou Out
W = si (bouton relaché) alors In = 0 >

Des opérateurs expriment la possibilité d'assembler des dispositifs. La composition de deux dispositifs est le produit cartésien des domaines de sortie des dispositifs. Par exemple, la position d'une souris est le produit cartésien des deux dispositifs élémentaires captant respectivement l'ordonnée et l'abscisse.

Nous retenons de cette taxonomie les caractéristiques des dispositifs et les opérateurs qui expriment leur assemblage.

3.5 Modèle des ressources d'interaction [Lachenal 2004]

Plus récemment, [Lachenal 2004] définit toutes entités manipulables comme des ressources d'interaction potentielles. En entrée (de l'utilisateur vers le système informatique), l'instrument est considéré comme une ressource d'interaction, cela peut être un dispositif classique comme la souris, ou un effecteur humain, comme le doigt. La notion de dynamicité est introduite : par exemple, une entité manipulable peut, à un instant t, être une ressource d'interaction et ne plus l'être à t+1. Plusieurs propriétés sont identifiées pour ces instruments :

- Les actions possibles : pointer, sélectionner, saisir du texte, déplacer, etc.
- La précision : mesure la précision avec laquelle l'acteur humain peut effectuer les actions possibles au moyen de l'instrument.
- La résolution : désigne la plus petite variation perceptible de la grandeur à mesurer dans des conditions de mesure données.
- La stabilité : Un instrument est stable si, en l'absence d'action de la part de l'utilisateur, la grandeur mesurée ne varie pas. La souris est un instrument stable alors que les instruments basés sur des systèmes de vision par ordinateur ou de localisation ne le sont pas. En effet, ces derniers sont en permanence soumis à des perturbations qui varient au cours du temps (luminosité, champ magnétique, etc.).

- L'adéquation : dénote la capacité d'un instrument à faciliter les actions de l'acteur humain.
- L'utilisabilité à distance : définit si cet instrument peut être utilisé pour agir sur une surface sans contact avec elle.
- La maniabilité : introduit les attributs de forme, de taille, de poids et de texture de l'entité physique qui ont un impact sur la maniabilité de l'instrument.
- La ré-assignabilité : renseigne sur le fait que l'instrument peut servir à une autre fonction que celle pour laquelle il a été conçu.

Le modèle des ressources d'interaction introduit de nouvelles caractéristiques à combiner avec celles des deux modèles précédents pour décrire les dispositifs.

3.6 Modèle d'interaction instrumentale [Beaudouin-Lafon 2000]

Le modèle de [Beaudouin-Lafon 2000] définit un espace de conception pour l'interaction graphique. Le système informatique incarne ici un agglomérat d'instruments permettant de manipuler les objets du domaine d'application (l'artefact). Comme pour notre définition de la modalité, l'instrument présente un niveau physique et un niveau logique. Par exemple, un instrument peut être une barre de défilement (appelé "instrument logique") manipulée par l'intermédiaire d'une souris (appelé "instrument physique"). Ce paradigme d'interaction met en évidence la forte dépendance entre les actions et la perception de ces actions par un utilisateur pour obtenir une interface de qualité. En effet, outre l'action sur l'objet du domaine qui doit être rendu à l'utilisateur, l'instrument présente une réaction qui permet à l'utilisateur de contrôler son action. Par exemple, les icônes s'illuminent lors du survol de la souris. D'autre part, trois propriétés essentielles sont définies pour les instruments : le degré d'indirection, le degré d'intégration et le degré d'incompatibilité.

- Le degré d'indirection correspond au décalage spatial et temporel entre l'instrument et l'objet manipulé. Par exemple, la fermeture d'une fenêtre par l'icône dédiée a un degré d'indirection plus faible que cette même fermeture réalisable par le menu de la fenêtre.
- Le degré d'intégration est le rapport entre le nombre de dimensions (ou degré de liberté) physiques et le nombre de dimensions logiques qui sont manipulées lors de l'utilisation de l'instrument. Pour une barre de défilement (1D) manipulée par une souris (2D), il est de 1/2.

- Le degré de compatibilité traduit la similarité entre les manipulations effectuées sur l'instrument et la réaction de l'objet manipulé. Par exemple, la barre de défilement possède un degré de compatibilité plus faible que le glisser-déposer. En effet, lorsque la barre de défilement descend, le document monte. Par contre, lors d'un glisser-déposer, l'objet manipulé suit les mouvements de la souris.

Le modèle d'interaction instrumentale se distingue des modèles précédents car il introduit un niveau supplémentaire au dispositif physique. En effet, l'instrument logique, qui augmentent le niveau d'abstraction des données fournit par le dispositif, incarne le langage d'interaction de notre définition de la modalité d'interaction. Nous retenons ainsi que les trois propriétés présentées sont essentielles pour définir le couplage des deux niveaux d'abstraction d'une modalité d'interaction.

3.7 Taxonomie de la théorie de [Bernsen 1994] et ses extensions

La théorie de Bernsen propose un espace permettant de classer ce qu'il appelle les "modalités représentationnelles", comme un texte, un graphe ou une icône. Bien qu'étudiée pour les modalités de sortie, cette taxonomie est applicable pour les modalités d'entrée. Nous comprenons qu'une "modalité représentationnelle" est un signifiant qui véhicule des concepts du système informatique, appelés les signifiés. Par rapport à notre terminologie, les "modalités représentationnelles" représentent le langage d'interaction dans notre définition d'une modalité (couple dispositif / langage d'interaction). Les "modalités représentationnelles" s'appuient sur des formes élémentaires, qui peuvent être combinées en représentations plus complexes. Ces "modalités représentationnelles" sont caractérisées par le doublet <média, profil>, où le média désigne le support d'expression en relation avec les capacités sensorielles et le profil définit un ensemble de propriétés. Le média prend sa valeur dans l'ensemble {graphique, son, toucher} et chacune correspond à une capacité sensorielle {visuelle, auditive, tactile}. Le profil regroupe un ensemble de propriétés :

- statique ou dynamique (s'il y a une dimension temporelle dans la représentation, celle-ci est dynamique) ;
- linguistique ou non-linguistique (une représentation linguistique est un langage au sens d'un système structuré de signes remplissant une fonction de communication comme le langage écrit ou le langage gestuel) ;
- analogique ou non-analogique (une "modalité représentationnelle" est analogique si elle entretient un rapport de ressemblance avec la réalité) ;

- arbitraire ou non-arbitraire (une représentation est arbitraire si elle fonctionne en dehors d'un système conventionnel).

Bien que définie pour les sorties, nous étendons cette théorie aux modalités d'interaction en entrée. Les valeurs du média sont adaptées non plus en fonction des sens humains, mais en fonction des modes de communication liés au dispositif (cf. section 2.2 de ce chapitre) : parole, geste et mental. Le profil reste inchangé et s'applique aux langages d'interaction. Par exemple, pour la modalité d'interaction de manipulation directe, le dispositif de la souris a pour mode de communication le geste et le profil du langage d'interaction associé est {dynamique, non linguistique, analogique, non arbitraire}.

En extension, [Nigay 1994] propose d'affiner la dimension statique/dynamique avec les notions introduites par [Sellen 1992] qui définissent trois critères pour les retours d'informations dynamiques : éphémère/non-éphémère, évitable/non-évitable et prolongé/non-prolongé. Par exemple, un bip émis à l'ouverture d'une fenêtre est éphémère, inévitable (en conditions normales) et non-prolongé (une seule émission). [Vernier 2001] étend cette taxonomie en proposant de caractériser la représentation par une dimension spatiale. Par exemple, les représentations d'une modalité sonore peuvent être mono, bi ou tridimensionnelles. De plus, il propose pour la dimension linguistique de caractériser la complexité et la précision du langage : la langue naturelle écrite est par exemple plus complexe que celle effectivement reconnue par l'ordinateur, une image floue d'un plan est considérée comme peu précise (vague). Nous étendons également ces extensions pour les modalités d'interaction en entrée. De l'extension de [Nigay 1994], seul le critère éphémère/non-éphémère est applicable pour les entrées car il peut définir si les informations sur les actions de l'utilisateur sont fournies de manières continues (non-éphémère) ou discrétisées (éphémère) et ceux pour chaque niveau (dispositif et langage d'interaction). L'extension de [Vernier 2001] permet de caractériser le langage d'interaction par une dimension spatial et par un niveau de complexité. Par exemple, une interface WIMP est considérée comme bidimensionnelle avec un niveau de complexité intermédiaire.

En résumé, la taxinomie de Bernsen et ses extensions sont intéressantes car elles intègrent les capacités cognitives humaines comme caractéristiques des modalités d'interaction. Bien que toutes ces caractéristiques aient été mises en évidence pour l'interaction en sortie, nous les avons étendues pour les entrées. Le média se réfère donc au mode de communication du dispositif (parole, geste ou mental). Le profil peut être réutilisé tel quel pour les langages d'interaction en entrée. Des propriétés de Sellen adaptées aux entrées, nous retenons le critère continu/discret de l'envoi des données fournies par les dispositifs et langages d'interaction d'entrée. Enfin, nous considérons que les langages d'interaction en entrée se caractérisent par une dimension spatiale et un niveau de complexité.

3.8 Espace de [Foley 1984]

A l'opposé des deux premières taxinomies, cet espace considère la tâche que l'utilisateur effectue. En effet, pour classer un dispositif, Foley propose de le mettre en relation avec les tâches graphiques qu'il permet d'accomplir. Ces tâches sont au nombre de trois : sélection, position et orientation, qui peuvent, selon les dispositifs, être accomplies de manière directe (la mesure est directe, sans dispositif physique intermédiaire) ou indirecte (mesure indirecte avec un dispositif physique intermédiaire). Par exemple, une tablette tactile peut être directement utilisée pour la tâche de sélection d'une interaction WIMP.

L'espace de Foley relève pour la première fois l'importance de la relation entre le dispositif et la tâche pour laquelle il est utilisé. L'inconvénient de cet espace est qu'il concerne seulement le domaine graphique et comme pour les précédents, les dispositifs non actionnables par l'utilisateur (sans commande explicite) ne sont pas classifiables.

3.9 Espace de [Frohlich 1991]

Cet espace prend comme point de départ le modèle du Processeur Humain de [Card 1983] (cf. section 2.5 de ce chapitre) pour définir un espace de conception. L'espace distingue les entrées et les sorties qui s'unissent cependant autour de cinq notions essentielles : le mode, le canal de communication, le canal de l'interface, le média et le style.

Le mode définit les deux métaphores d'interaction qui sont le langage et l'action. Le canal de communication représente le sens humain (sortie) ou le mode de communication (entrée) mis en jeu. Le canal de l'interface est en correspondance directe avec le canal de communication ; il désigne la capacité sensorielle de l'interface permettant "d'écouter" le canal de communication de l'utilisateur. Par exemple, au canal de communication qu'est la parole, le canal de l'interface correspondant est l'audio. Le média désigne le langage d'interaction pour l'échange d'informations. Enfin, le style correspond aux techniques d'interaction comme les champs à remplir ou les icônes.

Nous retenons de cet espace que le dispositif est caractérisé par un mode et un canal. De plus, différentes formes de langages d'interaction sont définies en fonction du média et du style.

3.10 Espace MSM [Nigay 1994]

L'espace MSN, comprenant six axes, regroupe au sein d'un canevas intégrateur nombre de caractéristiques identifiées dans les espaces précédents. Les deux premiers axes sont le nombre et le sens (entrée ou sortie) des canaux permis pour un système donné ; ils ont trait à la notion de canal de communication. Les autres dimensions caractérisent les fonctions d'interprétation et de restitution du système : niveaux d'abstraction, contexte, fusion et fission des informations échangées, granularité du parallélisme.

- Le canal de communication d'entrée (inversement de sortie) regroupe des dispositifs physiques d'entrée ou capteurs (inversement de sortie ou effecteurs) capables de recevoir (inversement d'émettre) des informations de types donnés sous le contrôle d'une capacité computationnelle ou processus.
- La notion de niveau d'abstraction exprime le degré de transformation subie par les informations reçues ou émises sur les canaux. Elle couvre également l'éventail des représentations que gère le système depuis les informations brutes (les signaux) jusqu'aux représentations symboliques (le sens). Tout canal peut être caractérisé par son niveau d'abstraction. Pour un système interactif donné, certains canaux peuvent être très performants en abstraction, d'autres pas.
- La capacité d'une fonction à abstraire ou à matérialiser peut dépendre de variables contextuelles. Le contexte comprend un ensemble de paramètres d'état, utilisés par les processus internes pour contrôler l'interprétation ou la restitution. Deux contextes d'interprétation sont pertinents pour les systèmes actuels : les commandes et les concepts du domaine. Les informations liées aux commandes font l'objet d'une interprétation à un haut niveau d'abstraction, tandis que les informations ayant trait aux concepts du domaine, sont souvent laissées inchangées.
- La fusion est la combinaison de plusieurs unités d'information pour former de nouvelles unités. La fission correspond au processus inverse. L'une et l'autre traduisent deux activités importantes des processus d'interprétation et de restitution. En interprétation, la fusion peut intervenir à un bas niveau d'abstraction entre des informations pouvant provenir de plusieurs canaux d'entrée. Elle peut aussi s'effectuer à un plus haut niveau d'abstraction pour des informations issues de différents contextes. Par exemple, le paradigme du "mets ça là" de [Bolt 1980] (cf. chapitre 1), nécessite la fusion de l'événement parole reçu via le canal du microphone avec les évènements

déictiques fournis par la modalité gestuelle. La fission en interprétation traduit le besoin de décomposer une information issue d'un canal ou d'un contexte pour franchir un niveau d'abstraction. Par exemple, l'acte de parole "dessine un cercle dans une nouvelle fenêtre" fait référence à deux domaines de discours : les figures géométriques ("dessine un cercle") et l'interface homme-machine ("nouvelle fenêtre"). Cette phrase, dont le sens a pu être identifié le long d'un canal unique, doit être décomposée en deux primitives de haut niveau du système : "créer fenêtre" et "créer cercle" dans la nouvelle fenêtre.

- Le parallélisme se manifeste à trois niveaux de granularité : action physique, tâche élémentaire, grappe de tâches. Au niveau physique, le parallélisme en entrée autorise l'utilisateur à agir simultanément sur plusieurs dispositifs d'entrée. Si ces dispositifs sont répartis sur différents canaux, alors plusieurs canaux d'entrée sont sollicités en parallèle comme dans l'exemple du "mets ça là" [Bolt 1980]. Notons que la fusion et la fission à bas niveau d'abstraction dépendent de l'existence du parallélisme au niveau physique. Au niveau tâche élémentaire, le parallélisme en entrée autorise l'utilisateur à construire plusieurs commandes de manière concurrente. Une grappe de tâches est un ensemble de tâches élémentaires ; le parallélisme exprime ici les possibilités d'entrelacement entre des espaces de travail.

Le modèle MSM est la clef de voûte de notre étude. Il met en évidence les concepts et mécanismes propres à la multimodalité selon six axes. Les notions de canal de communication, niveaux d'abstraction, parallélisme, prise en compte du contexte, fusion/fission des données multimodales résument les axes principaux que notre approche doit couvrir.

4 Unification des modèles de psychologie cognitive et d'IHM

Les modèles de l'humain et de l'ordinateur dans le cadre de l'IHM sont complémentaires. Si l'on compare le système informatique à la sphère d'Escher présenté à la Figure 2.10, il peut être vu comme l'image de l'homme, et réciproquement. En étudiant tous ces modèles, nous avons observé beaucoup de similitudes. Afin de comprendre et de maîtriser l'ensemble du processus lors de l'utilisation d'une modalité d'interaction par un utilisateur, ce chapitre contribue à unifier certains modèles de psychologie cognitive (cf. section 2 de ce chapitre) à ceux du domaine de l'IHM (cf. section 3 de ce chapitre). Nous les avons organisés selon leur niveau de description, en partant des modèles les plus généraux pour finir avec les modèles plus précis.

Figure 2.10 : Hand with reflecting sphere, lithographie, M.C. Escher, 1935.

4.1 Processeur humain et principe "perception – traitement – action"

Dans la section 2.1 de ce chapitre, l'humain est vu comme un ordinateur multi-processeurs [Card 1983] selon le principe commun "perception – traitement – action" (cf. section 3.2 de chapitre). La Figure 2.11 montre l'unification de ces deux modèles présentés respectivement à la Figure 2.5 et à la Figure 2.8. Ainsi, le

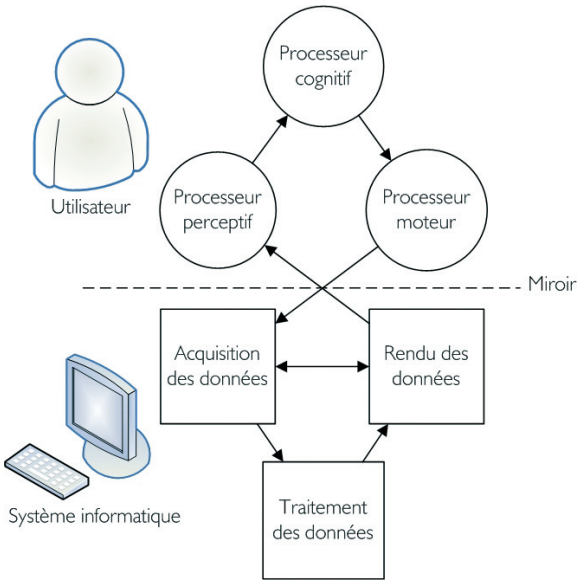


Figure 2.11 : unification des modèles de base (le processeur humain de [Card 1983] et le modèle "perception-traitement-action").

processeur perceptif humain correspond au module acquisition des données du système informatique, le processeur cognitif au module de traitement des données et le processeur moteur à celui du rendu des données. Ce schéma fait apparaître les dépendances existantes entre l'utilisateur et le système informatique : lorsque le système informatique rend perceptibles des données, elles sont interprétées par le processeur perceptif de l'utilisateur ; pour l'interaction en entrée, c'est le processeur moteur de l'utilisateur qui est à l'origine des données acquises par le système informatique.

L'unification de ces deux modèles met en évidence les similitudes des approches en psychologie cognitive et en IHM. Nous retrouvons les dépendances entre chaque module, notamment lors de l'utilisation d'une modalité d'interaction d'entrée qui relie directement les traitements du processeur moteur humain au module d'acquisition des données du système informatique.

4.2 Théorie de l'action et modèle pipeline

Si nous approfondissons cette démarche, nous remarquons que le modèle Pipeline de [Nigay 1994] et celui de la théorie de l'action de [Norman 1986] sont aussi le reflet l'un de l'autre. La Figure 2.12 présente les deux approches combinées. Le modèle Pipeline présenté précédemment (cf. section 3.2 de ce chapitre) est ici adapté en présentant les différentes étapes de transformation des données. Nous remarquons que les étapes sont les mêmes que celles identifiées par [Norman 1986] (cf. section

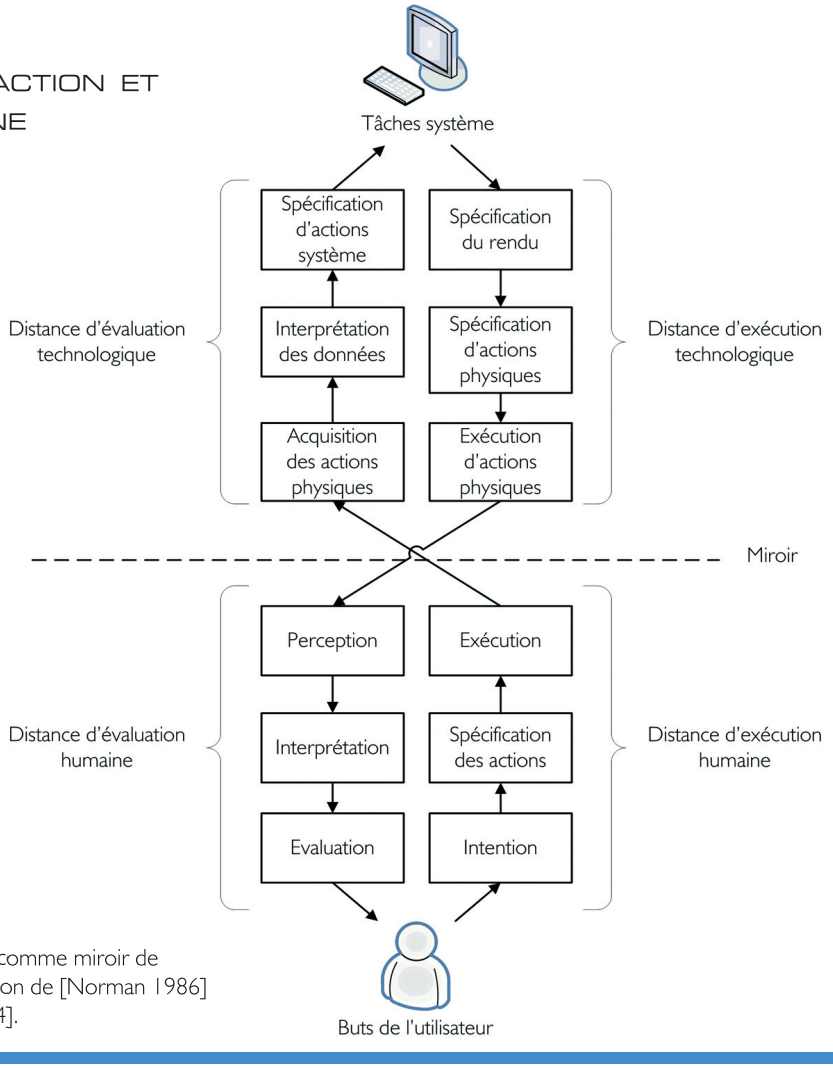


Figure 2.12: le système informatique comme miroir de l'homme, fusion de la théorie de l'action de [Norman 1986] et du modèle Pipeline de [Nigay 1994].

2.1 de ce chapitre) mais avec l'ordinateur comme centre d'étude. Ainsi les distances d'exécution et d'évaluation comprennent les mêmes étapes, que ce soit du côté humain ou technologique.

La mise en relation des points de vue humain et technologique permet une meilleure compréhension des phénomènes liés à la multimodalité. La fusion de ces deux modèles permet de voir que notre objectif pour la conception et le développement d'interfaces multimodales en entrée se joue à la fois sur les étapes de distance d'exécution humaine et de distance d'évaluation technologique.

4.3 Modèle ICS et couple <Dispositif, Langage d'interaction>

Enfin, nous étudions la relation entretenue entre le modèle ICS centré humain (cf. section 2.5 de ce chapitre) et la définition d'une modalité d'interaction adoptée dans nos travaux (cf. section 3.1 de ce chapitre).

A la Figure 2.13, nous mettons en relation le modèle ICS avec la partie système, où nous distinguons les dispositifs et les langages d'interaction. Nous retrouvons plusieurs similitudes entre les deux facettes utilisateur et système. En effet, les modules d'acquisition des données aussi bien humain (sous-systèmes acoustique, état physique et visuel) que technologique (dispositifs d'entrée tels le microphone, la caméra, le clavier ou la souris) sont similaires. Nous pouvons aussi rapprocher les modules de restitution humains (sous-systèmes articulatoire et mouvement) avec ceux du système informatique (par exemple l'écran ou les haut-parleurs). De même, les sous-systèmes cognitifs humains peuvent être comparés aux langages d'interaction et au reste du système informatique (contrôleur de dialogue, noyau fonctionnel, etc.).

Plus détaillé que les modèles présentés aux Figure 2.11 et Figure 2.12, il est intéressant de voir ici que la décomposition des deux acteurs, utilisateur et système informatique, sont très similaires. En effet, les différents niveaux d'abstraction sont retrouvés dans les deux modèles. De plus, nous observons les dépendances entre chaque module et particulièrement celles entre l'utilisateur et le système informatique. Du point de vue de l'interaction en entrée, les dispositifs d'entrée sont en relation directe avec les sous-systèmes articulatoire et mouvement d'ICS.

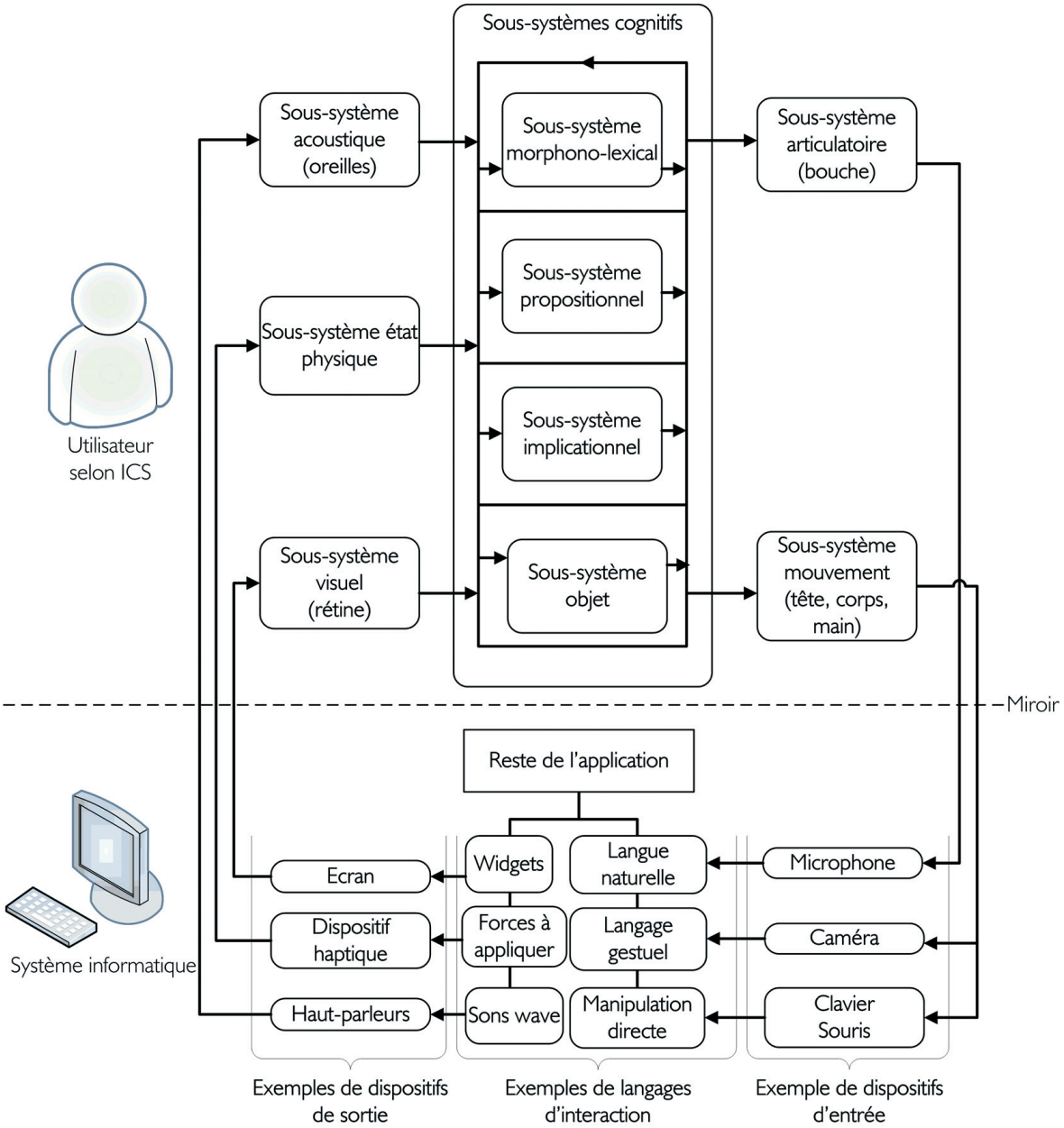


Figure 2.13 : unification du modèle ICS et de la définition de la modalité comme un couple <Dispositif, langage d'interaction>.

5 Résumé du chapitre 2

Ce chapitre pose le cadre conceptuel de notre étude en définissant les différentes notions et les mécanismes intervenant dans la multimodalité. Nous avons identifié nombre de caractéristiques des interfaces multimodales en entrée, que nous devons assimiler à notre proposition. Nous avons vu que ces caractéristiques touchent aussi bien l'aspect humain, avec les divers modèles cognitifs présentés, que l'aspect technologique et ses modèles issus du domaine de l'IHM.

L'unification des modèles de psychologie cognitive avec ceux d'IHM est pertinente car elle souligne que les modélisations sont similaires et permettent une meilleure assimilation des liaisons existantes entre l'utilisateur et le système informatique. La mise en relation du fonctionnement humain avec celui du système informatique permet, en effet, d'intégrer explicitement les caractéristiques de l'utilisateur dans les modules logiciels correspondant aux modalités d'interaction, rendant ainsi leur mécanisme complètement adapté aux possibilités d'usage.

L'ensemble des concepts étant établis au travers des différents modèles, aussi bien centrés humain que technologique, le chapitre suivant caractérise les usages des modalités d'interaction.

La Figure 3.1 pose le cadre d'étude de l'usage des modalités d'interaction. L'utilisateur communique ses intentions au système informatique. Cette information est véhiculée par la sélection d'une ou plusieurs modalités d'interaction. Le choix des modalités d'interaction est effectué en fonction des informations à transmettre, en fonction du contexte (à prendre ici au sens général : le contexte est l'ensemble des circonstances dans lesquelles se produit un fait [Robert 1998]) et parmi l'espace des possibilités des modalités d'interaction et de leur combinaison. L'espace des possibilités est soit statique, défini lors de la conception du système, soit variable par la découverte dynamique des modalités d'interaction. Pour les entrées, le choix des modalités d'interaction est effectué par l'utilisateur. De l'usage des modalités d'interaction, le système informatique capture, analyse et traite une expression multimodale en entrée.

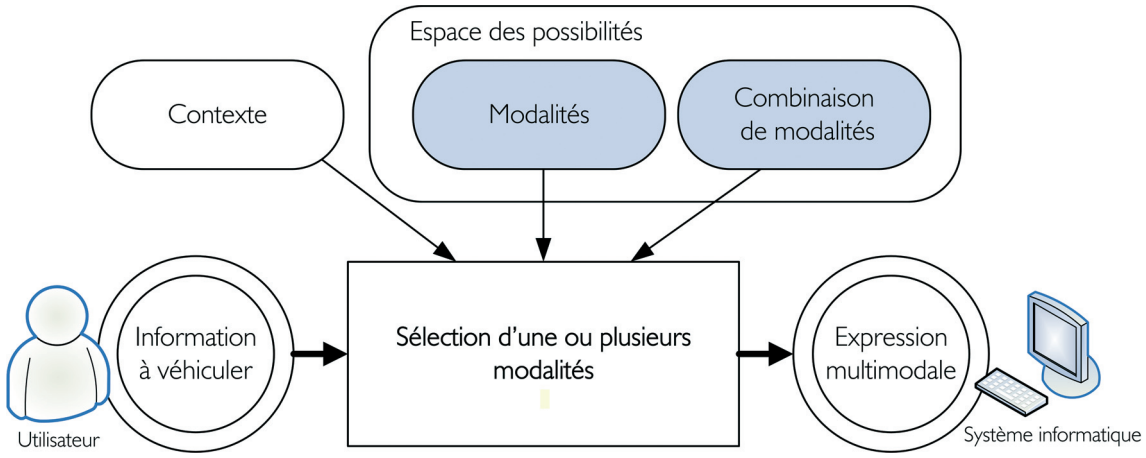


Figure 3.1 : usage des modalités d'interaction.

Dans ce cadre d'étude, ce chapitre dresse l'état de l'art concernant les usages des modalités d'interaction. La section 1 développe les principaux résultats conceptuels qui caractérisent ces usages. La section 2 expose les résultats expérimentaux sur les différents paramètres à prendre en compte, pour proposer des modalités d'interaction et des formes de combinaisons appropriées à l'utilisateur, à la tâche et à son contexte.

Les sections suivantes proposent trois modèles caractérisant l'espace des possibilités identifié sur la Figure 3.1. Le premier, UOM [Nigay 1994], couvre plusieurs aspects de la multimodalité. Il permet de classer les systèmes multimodaux selon la multiplicité des dispositifs et des langages d'interaction au sein du système étudié, selon l'équivalence de ces moyens en termes d'optionalités d'usage et encore selon la manière dont ils peuvent être utilisés (aspect temporel et possibilité d'utilisation combinée). Les deux suivants, CARE [Coutaz 1994] et TYCOON [Martin 2002], identifient des relations entre les modalités d'interaction en caractérisant précisément les diverses formes de combinaison possibles.

1.1 UOM : CLASSIFICATION DES SYSTÈMES MULTIMODAUX [NIGAY 1994]

UOM (Usage, Option, Multiplicité) se base sur les définitions de modalité et multimodalité. Il reprend les concepts du modèle MSM du chapitre 2. Pour rappel, une modalité d'interaction est l'association de deux éléments de niveaux d'abstraction différents : un dispositif DP (niveau physique) et un langage d'interaction LI (niveau logique). UOM permet la classification des systèmes informatiques multimodaux et se décompose en trois volets, selon que l'on considère la Multiplicité, le caractère Optionnel ou l'Usage des langages d'interaction et des dispositifs physiques du système interactif.

1.1.1 Multiplicité

La multiplicité des langages d'interaction et des dispositifs physiques est caractérisée par le volet M²LD d'UOM. Bien que ce modèle distingue l'interface d'entrée de celle de sortie, nous présentons uniquement la première facette. Quatre cas sont identifiés et présentés à la Figure 3.2. La multiplicité des dispositifs et des langages d'interaction est considérée : la valeur "non" sur un axe signifie que le système présente une seule entité pour cette dimension (unicité du dispositif ou du langage d'interaction). La valeur "oui" correspond au cas de la multiplicité des entités (multiplicité des dispositifs ou des langages d'interaction).

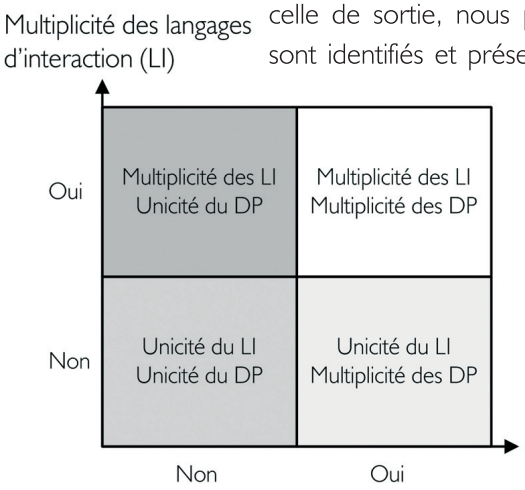


Figure 3.2 : M²LD d'après [Nigay 1994].

Pour illustration, considérons le système multimodal de réalité augmentée ARQuake [Piekarski 2002]. Il s'agit de l'adaptation du jeu Quake

où l'objectif est de tirer sur des monstres qui se présentent ici en superposition du monde réel. ARQuake présente une multiplicité des dispositifs et des langages d'interaction. En effet, comme le montre la Figure 3.3, le joueur est équipé d'un pistolet, d'un dispositif de localisation GPS (dans son dos) et d'un dispositif d'orientation (sur sa tête). Pour chacun de ces dispositifs, un langage d'interaction lui est associé. Nous retrouvons ainsi un système de coordonnées pour la localisation, un système de coordonnées pour l'orientation de sa tête et un langage de commande simple informant des appuis sur la gâchette du pistolet.

Figure 3.3 : équipement d'un joueur d'ARQuake [Piekarski 2002].

1.1.2 Optionalité

Le caractère optionnel des langages d'interaction et des dispositifs physiques est décrit par le volet O^2LD. Quatre cas sont identifiés et présentés à la Figure 3.3. L'axe vertical représente le choix du langage d'interaction pour une tâche donnée et l'axe horizontal représente l'existence de choix du dispositif physique pour un langage d'interaction donné. Les valeurs "oui" et "non" indiquent l'existence ou l'absence de choix, c'est-à-dire le caractère optionnel ou obligatoire du choix à l'instant t.

Reprenons l'exemple utilisé pour la multiplicité ARQuake [Piekarski 2002] : la multiplicité des dispositifs et des langages d'interaction n'offre ici aucun choix à l'utilisateur car chaque dispositif est associé à un langage d'interaction particulier et chaque langage d'interaction existe pour la réalisation d'une tâche particulière (définition de la position de l'utilisateur, de l'orientation de sa tête ou tir). Ainsi, ARQuake correspond au cas "obligation des dispositifs et des langages d'interaction".

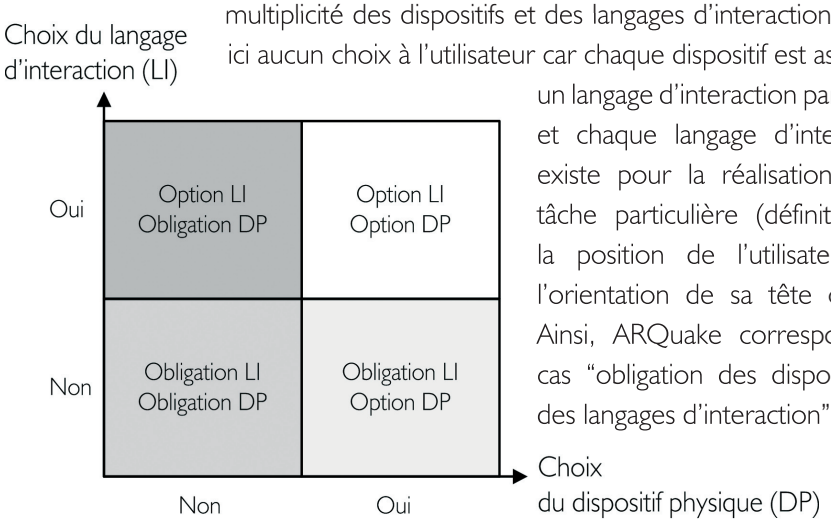


Figure 3.4 : O^2LD d'après [Nigay 1994].

1.1.3 Usage

Le résultat conceptuel sur la facette usage d'UOM se base sur les notions du modèle Pipeline [Nigay 1994] présenté au chapitre précédent. Pour rappel et comme le présente la Figure 3.5, les intentions de l'utilisateur se traduisent par une suite d'actions physiques qui sont ensuite interprétées, formant des unités informationnelles. Ces unités informationnelles définissent une suite d'actions système, dont les effets sont traités par le noyau fonctionnel. Par exemple, l'utilisateur a pour intention d'insérer une nouvelle ligne dans un texte. Les actions physiques correspondantes sont de positionner le curseur à la fin de la ligne 2 et d'appuyer sur la touche retour chariot. L'unité informationnelle qui résulte de ces deux actions physiques se présente de la sorte : "insérer un retour chariot à la fin de la ligne 2". Les effets des actions systèmes qui sont provoquées sont la création d'une nouvelle ligne dans le texte et la modification du point d'insertion.

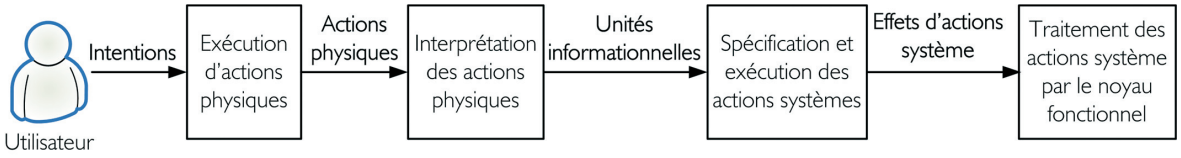


Figure 3.5 : les différents niveaux d'abstraction de l'interaction entre un utilisateur et le noyau fonctionnel au sein du système informatique.

L'usage des dispositifs ou des langages d'interaction (appelé facette ULD) se base donc sur ces notions et peut être classé au sein de la classification de la Figure 3.6.

L'usage :

- *exclusif* implique des dispositifs ou des langages d'interaction utilisés de façon séquentielle et l'absence de combinaison d'information. Par exemple, les téléphones portables qui proposent des systèmes de reconnaissance vocale en plus du clavier classique proposent un usage exclusif.
- *concurrent* inclut une utilisation parallèle des dispositifs ou des langages d'interaction en l'absence de combinaison d'information. Comme exemple, nous pouvons citer la plupart des interfaces graphiques qui proposent un usage concurrent de la souris et du clavier.
- *alterné* implique des dispositifs ou des langages d'interaction utilisés de façon séquentielle avec une combinaison des informations. Par illustration, le système multimodal MMI2 [Wilson 1991], utilisé pour la conception de réseau informatique, offre un usage alterné des dispositifs et des langages d'interaction. Par exemple, l'utilisateur sélectionne l'icône d'un ordinateur

sur le réseau puis saisit la phrase en langage naturel : "What is the cost of this machine ?". Ne pouvant le faire de façon simultanée, il s'agit ici d'un usage alterné.

- *synergique* comprend des dispositifs ou des langages d'interaction utilisés parallèlement avec une combinaison des informations. C'est le cas, par exemple, du système multimodal "put-that-there" de [Bolt 1980] et de l'ARQuake [Piekarski 2002], utilisé pour illustrer les deux facettes d'UOM précédentes et dont les actions avec le pistolet doivent être combinées avec la position et l'orientation du joueur, afin de déterminer quel est l'endroit visé par le tir du joueur.

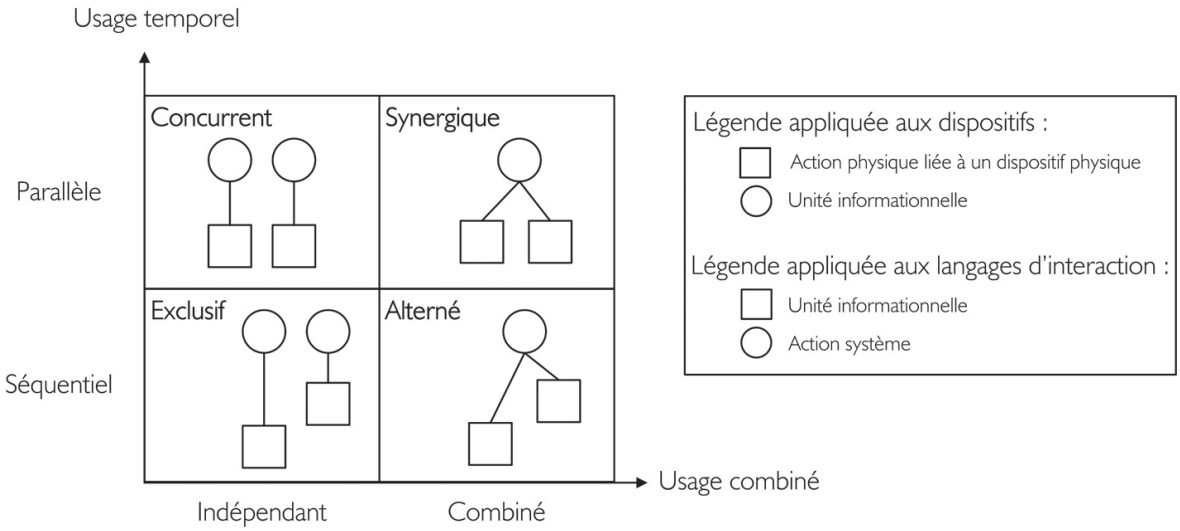


Figure 3.6 : usage des Langages et des Dispositifs.

La classification UOM offre, au travers de ces trois volets M^2LD, O^2LD et ULD, une caractérisation fine de la multimodalité en considérant le dispositif et le langage d'interaction qui composent une modalité d'interaction. Nous verrons dans les sections suivantes que des aspects d'UOM sont en relation directe avec l'utilisabilité (ergonomie). Par exemple, la souplesse (propriété d'ergonomie) peut se mesurer en termes des choix offerts à l'utilisateur (O^2LD).

La classification UOM [Nigay 1994] souligne par sa facette ULD les différents types d'usage des dispositifs et des langages d'interaction. Les facettes M^2LD et O^2LD caractérisent les choix offerts à l'utilisateur. L'ensemble de ces concepts doit apparaître dans notre modèle conceptuel et dans son opérationnalisation.

1.2 Propriétés CARE [Coutaz 1994] et extensions

Comme nous venons de le voir avec la facette usage d'UOM, les dispositifs et les langages d'interaction entretiennent des relations permettant de caractériser les systèmes multimodaux. [Coutaz 1994] définit un espace conceptuel permettant de préciser la nature de ces relations par la définition des propriétés CARE : Complémentarité, Assignation, Redondance et Equivalence.

- La *complémentarité* entre les dispositifs ou les langages d'interaction exprime qu'il faille utiliser tous les dispositifs ou tous les langages d'interaction pour obtenir une commande complète. Cela signifie qu'aucun des dispositifs ou des langages d'interaction ne suffise à lui seul. L'utilisation complémentaire de dispositifs ou de langages d'interaction peut être parallèle ou séquentielle. Comme exemple de complémentarité, le système de type "put-that-there" de [Bolt 1980], présenté au chapitre 1, est une référence : l'utilisateur spécifie une commande vocale de déplacement en complémentarité de l'objet graphique à déplacer et de sa destination spécifiée par le geste.
- L'*assignation* exprime l'obligation d'utiliser un dispositif pour un langage d'interaction donnée ou langage d'interaction particulier pour une tâche. Cette propriété exprime donc l'absence de choix. Elle dénote un manque évident de souplesse qui à défaut augmente le caractère préemptif du système multimodal. Comme exemple d'assignation, l'unique moyen d'allumer un ordinateur est d'utiliser le bouton prévu à cet effet.
- La *redondance* dénote l'utilisation séquentielle ou parallèle de plusieurs modalités d'interaction équivalentes. En entrée, la redondance d'information en provenance de l'utilisateur implique la prise en compte d'une seule des modalités d'interaction par le système, l'autre pouvant éventuellement contribuer à fiabiliser l'expression obtenue. Par exemple, le montage redondant d'un microphone et d'une caméra qui observe le mouvement des lèvres d'un utilisateur permet d'augmenter la robustesse d'un système de reconnaissance vocale.
- L'*équivalence* d'un ensemble de modalités d'interaction est vérifiée si chaque dispositif ou langage d'interaction permet d'atteindre le même but en produisant les mêmes données. Plusieurs dispositifs sont équivalents pour un langage d'interaction donné et plusieurs langages d'interaction sont équivalents pour une tâche donnée. Comme exemple d'équivalence, il est possible de fermer une fenêtre sous Windows soit par l'utilisation de l'icône de fermeture avec la souris soit par l'utilisation du raccourci clavier "Alt+F4". L'équivalence permet donc plus de souplesse ou même de

robustesse si par exemple, dans un milieu bruyant, le clavier peut être substitué au microphone.

[Vernier 2001] affine les propriétés CARE avec la mise en relation de cinq schémas de composition avec cinq aspects de composition, qui sont le temps, l'espace, l'articulation des dispositifs, la syntaxe des langages représentationnels et la sémantique de l'information véhiculée. La Figure 3.7 illustre cette mise en correspondance. Issus des travaux d'Allen [Allen 1983] sur l'aspect temporel, les cinq schémas de composition illustrent la composition de modalités éloignées (1ère colonne de la Figure 3.7), la composition de modalités avec un point de contact (2ème colonne de la Figure 3.7), la composition de modalités avec une intersection non vide (3ème colonne de la Figure 3.7), la composition de modalités dont l'une est plus étendue et englobe l'autre (4ème colonne de la Figure 3.7) et la composition de modalités de même étendue (5ème colonne de la Figure 3.7).

Cet espace de classification approfondit les deux propriétés CARE concernant la composition : la Complémentarité et la Redondance. Ainsi, nous notons que pour l'aspect sémantique, les deux premières colonnes sont relatives à la complémentarité, alors que les deux dernières correspondent aux différents cas de redondance. La colonne du milieu définit la relation qu'il peut exister entre ces deux propriétés.

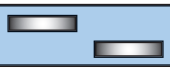
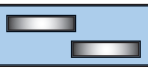
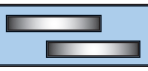
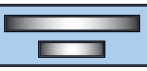
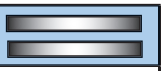
		Schémas de composition				
	Composition					
Aspects de composition	Temporelle	Anachronique	Séquentielle	Concomitante	Coïncidente	Parallèle / Simultanée
	Spatiale	Disjointe	Adjacente	Intersectée	Imbriquée	Recouvrance
	Articulatoire	Indépendance	Fissionnée	Fissionnée + Dupliquée	Partiellement Dupliquée	Dupliquée
	Syntaxique	Différente	Complétion	Divergence	Extension	Jumelage
	Sémantique	Concurrente	Complémentaire	Complémentaire + Redondante	Partiellement Redondante	Totalement Redondante

Figure 3.7 : application des schémas de composition aux cinq aspects de composition [Vernier 2001].

Les propriétés CARE fournissent un cadre simple pour définir les relations entre dispositifs, langages d'interaction et tâches. L'équivalence et l'assignation caractérisent l'absence ou non de choix. La complémentarité et la redondance qualifient l'aspect combinatoire. L'extension proposée précise différentes possibilités de combinaison en fonction des moments d'utilisation des dispositifs, des langages d'interaction et des origines de la composition de leurs données. Notre contribution doit s'appuyer sur les différents cas de combinaisons identifiés dans cette section.

1.3 Espace TYCOON [Martin 2002]

L'espace Tycoon a été spécialement développé pour l'analyse de l'activité des utilisateurs, en termes de coopération entre les modalités d'interaction utilisées. L'usage des modalités d'interaction se traduit par la mise en œuvre d'agents coopérants et communiquant les intentions de l'utilisateur pour l'interaction en entrée. Six types de primitives de coopération entre les agents sont définis : l'équivalence, le transfert, la spécialisation, la redondance, la complémentarité et la concurrence. Elles caractérisent les échanges d'information entre agents pour réaliser un but commun :

- L'*équivalence* entre deux modalités d'interaction implique que la nature des données qu'elles produisent est semblable.
- Le *transfert* entre deux agents définit que les sorties d'un agent sont réutilisées comme entrées par l'autre agent.
- La *spécialisation* signifie que pour un ensemble d'agents, il n'existe qu'un seul agent produisant un type de données particulier.
- La *redondance* indique que les données produites par les agents sont totalement ou partiellement similaires à un instant donné.
- La *complémentarité* entre plusieurs modalités d'interaction implique la fusion de données différentes (non-équivalentes) provenant de chaque modalité d'interaction.
- La *concurrence* décrit que plusieurs agents produisent des sorties au même moment.

Par rapport aux deux études présentées précédemment, Tycoon met au même niveau les aspects temporels (concurrence), les capacités de communication entre agents (transfert), l'absence de choix (spécialisation) et les combinaisons possibles entre modalités d'interaction (équivalence, redondance, complémentarité). Cette étude souligne à nouveau l'importance de ces éléments pour notre étude de la multimodalité.

1.4 Synthèse

Les résultats conceptuels retenus fournissent un cadre précis, nous permettant de caractériser les divers usages des modalités d'interaction au sein d'un système multimodal donné. Dans cet objectif, l'ensemble des concepts définis par ces modèles sont essentiels pour l'élaboration de notre modèle conceptuel et pour son opérationnalisation.

La classification UOM [Nigay 1994] offre les éléments généraux permettant à notre modèle conceptuel de couvrir les différents types de systèmes multimodaux. Sa facette ULD souligne les différents types d'usage des dispositifs et des langages d'interaction alors que les facettes M^2LD et O^2LD caractérisent les choix offerts à l'utilisateur en terme de possibilité et d'optionalité.

Les propriétés CARE et l'espace TYCOON approfondissent les relations d'usage données par la facette ULD d'UOM. Tout d'abord, l'usage combiné est défini par les propriétés de complémentarité et de redondance. L'existence de choix est incarnée par la propriété d'équivalence, alors que l'absence de choix est définie par l'assignation (CARE) ou la spécialisation (TYCOON). De plus, les différents aspects de composition (extensions de CARE par [Vernier 2001]) définissent les origines de cette composition (temporelle, spatiale, articulatoire, syntaxique ou sémantique). D'autre part, l'usage temporel est précisé par la propriété de concurrence (TYCOON) et par la définition précise des différents schémas de combinaisons temporelles (extensions de CARE [Vernier 2001] par les propriétés de [Allen 1983]).

Ces résultats conceptuels présentent les notions essentielles sur lesquelles notre modèle conceptuel doit se baser pour couvrir et définir les différentes formes d'usage des modalités d'interaction. Complémentaire à ces résultats conceptuels, la section suivante est consacrée aux résultats expérimentaux et regroupe les bases d'un véritable guide pour la conception de systèmes multimodaux adaptés.

L'espace des modalités d'interaction présenté à la Figure 3.1, que l'utilisateur a la possibilité d'utiliser doit être adapté à ses caractéristiques, à la tâche et au contexte. Alors que la section précédente définit les résultats conceptuels de l'usage des modalités d'interaction permettant de caractériser ces possibilités, cette section énonce des résultats expérimentaux évaluant l'adéquation des modalités d'interaction et des formes de multimodalité en fonction de l'utilisateur, de la tâche et du contexte. Nous en distinguons deux catégories : celles offrant des bases pour la compréhension des paramètres qui déterminent l'utilisation ou non des modalités d'interaction par un utilisateur, et celles qui présentent des recommandations pour la conception d'interfaces multimodales de qualité.

2.1 PARAMÈTRES DÉTERMINANT L'UTILISABILITÉ DES MODALITÉS D'INTERACTION

Cette section montre que le choix des modalités d'interaction dépend de plusieurs paramètres à prendre en compte, comme le mode d'implémentation, les tâches à réaliser, les spécificités des utilisateurs ou encore le contexte.

Il convient de noter que la majorité des résultats exposés dans ces sections ont été identifiés pour des modalités d'interaction naturelles (voir le rappel terminologique ci-dessous concernant les modalités d'interaction naturelles).

Terminologie : Les modalités d'interaction naturelles
Les modalités d'interaction naturelles concernent les moyens d'interaction qui sont utilisés spontanément chez un humain pour communiquer avec un tiers (humain ou non). Nous pouvons citer le langage naturel ou la désignation par le geste comme exemples typiques de modalités d'interaction naturelles. Depuis leur utilisation dans le système multimodal de référence de [Bolt 1980], la majorité des recherches dans le domaine multimodal se limitent à l'étude des modalités d'interaction naturelles et notamment de la reconnaissance vocale.

2.1.1 Mode d'implémentation

[Zouinar 2003] montre l'importance du mode d'implémentation concernant l'usage des modalités d'interaction. Il semble évident que les conclusions des expérimentations sur les systèmes multimodaux résultent de l'implémentation de chacune des modalités d'interaction. Une faible efficacité ou fréquence d'utilisation d'une modalité d'interaction peut révéler une inadéquation de sa réalisation technique. Par exemple, un système de reconnaissance vocale qui a un taux de reconnaissance variable ou trop faible est automatiquement substitué par une autre modalité équivalente lorsque l'optionalité est possible [Robbe 2000].

Lorsque l'utilisateur a le choix entre plusieurs modalités d'interaction, la manière dont est implémentée chaque modalité d'interaction et leurs éventuelles combinaisons est un critère déterminant de ce choix. L'utilisateur préfère utiliser la modalité d'interaction la plus fiable et la plus efficace, rendant ainsi l'interaction plus robuste.

2.1.2 Tâches à accomplir

Comme le montre la Figure 3.1, le choix des modalités d'interaction dépend de la tâche à accomplir. Ce principe est confirmé explicitement dans [Zouinar 2003] et [Oviatt 2004], qui notent que les modalités d'interaction choisies par un utilisateur changent en fonction de la tâche.

Plus précisément, [Oviatt 1999] montre que selon le type de tâches à réaliser (sélection, localisation spatiale, etc.), l'utilisateur a des préférences interactionnelles. Ces résultats sont issus d'une expérimentation réalisée avec les utilisateurs d'un système intégrant la parole et le geste (modalités d'interaction naturelles) pour le contrôle d'un poste de commandement militaire appelé Quickset [Cohen 1997]. Dans QuickSet, l'utilisation complémentaire de deux modalités d'interaction (au sens de CARE et TYCOON présentés respectivement aux sections 1.2 et 1.3 de ce chapitre) est utilisée dans 20% du temps d'une session de travail. Le reste du temps, les utilisateurs passent d'une modalité d'interaction à l'autre sans raison apparente et restent unimodaux. Comme le montre la Figure 3.8, les commandes

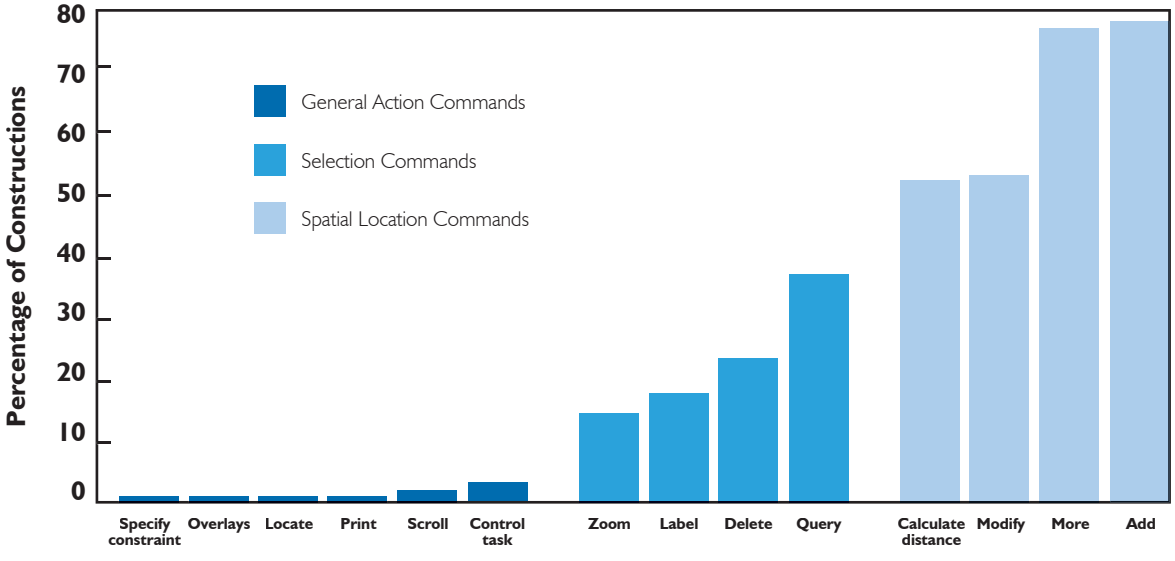


Figure 3.8 : pourcentage des commandes que les utilisateurs ont exprimées par la complémentarité de deux modalités d'interaction en fonction du type de commande. Graphique issu de [Oviatt 1999].

spatiales (à droite) sont plus fréquemment réalisées par la complémentarité des deux modalités d'interaction parole et geste que les autres commandes.

[Robbe 2000] fait également ce constat et précise que même si le choix des modalités d'interaction effectué par l'utilisateur a un coût cognitif plus important (notamment lors de l'utilisation de modalités d'interaction complémentaires), c'est l'efficacité perçu par l'utilisateur qui détermine son choix. De plus, les utilisateurs n'ont aucune difficulté à mémoriser les meilleurs types d'interaction en fonction des tâches [Jöst 2005].

Le choix des modalités d'interaction dépend donc de la tâche que doit accomplir l'utilisateur. L'efficacité qu'elle/il perçoit lors de l'utilisation d'une forme d'interaction pour réaliser sa tâche détermine son choix, au détriment d'une éventuelle augmentation de sa charge cognitive.

2.1.3 Spécificités individuelles

Comme le souligne [Carbonell 2003], l'utilisation des modalités d'interaction relève un fort caractère individuel. En effet, dans les expériences de [Zouinar 2003], la spécialisation d'une modalité d'interaction à la réalisation d'une ou plusieurs commandes n'est pas observable à un niveau global. C'est l'analyse des données spécifiques pour chaque sujet qui permet le constat d'une spécialisation. [Oviatt 1999] montre chez de nombreux sujets une tendance à des usages préférentiels de modalités. Ces tendances sont :

- *ponctuelles* lorsqu'elles n'apparaissent qu'au cours d'une seule session d'interaction ;
- *récurrentes* quand elles apparaissent dans d'autres sessions ;
- *individualisées* lorsqu'une même modalité d'interaction est utilisée pour réaliser une seule et même commande ;
- *plurielles* quand une même modalité d'interaction est utilisée pour réaliser plusieurs commandes.

L'utilisation de deux modalités d'interaction complémentaires pour une tâche spécifique présente des différences d'ordonnancement temporel selon les individus : certains utilisateurs les utilisent séquentiellement (l'un après l'autre) et d'autres simultanément.

De plus, la variabilité de l'état de l'organisme pour un individu influe sur l'efficacité des modalités d'interaction. Par exemple, l'étude de [Kumar 2004] montre que les taux de reconnaissance vocale et gestuelle sont relatifs à la forme physique de l'utilisateur.

L'usage des modalités d'interaction dépend de chaque individu. Les spécificités de chaque utilisateur (cf. chapitre 2), par rapport à leurs fonctionnements cognitifs, à leurs compétences ou encore à l'état de leurs organismes (âge, maladies, handicap temporaire ou permanent, etc.) entraînent des différences d'usage lorsque plusieurs modalités d'interaction sont équivalentes fonctionnellement. La mise en place de plusieurs options en termes de modalités d'interaction répond à cette spécificité individuelle [Oviatt 2000].

2.1.4 Contexte

Comme le montre la Figure 3.1 (page 71), l'utilisation d'une modalité d'interaction se révèle être dépendante du contexte. Nous considérons ici le sens général du contexte qui représente l'ensemble des circonstances dans lesquelles se produit un fait [Robert 1998]. Certaines modalités d'interaction sont parfois inadéquates au contexte. Il est ainsi très aisé d'imaginer que l'utilisation d'un système de reconnaissance vocale, dans un environnement extrêmement bruyant, soit peu performante [Heckmann 2002], ou encore que la brillance du soleil gêne l'interaction avec une interface graphique et que, dans ce cas, l'utilisation de la parole soit favorisée [Jöst 2005]. De même, l'utilisation d'un système de reconnaissance de geste est irréalisable lorsque l'utilisateur doit utiliser ses bras pour réaliser d'autres tâches en parallèle, comme par exemple la conduite d'un véhicule. A contrario, certaines modalités d'interaction sont plus pertinentes que d'autres en fonction du contexte. Par exemple, les systèmes de reconnaissance vocale ou gestuelle, par leurs aspects compact et portable, sont plus adaptés à une situation de mobilité que l'utilisation du clavier et de la souris [Oviatt 2000], comme nous l'observons pour les nouveaux baladeurs portables iPod®, dont l'interface de reconnaissance de geste est particulièrement adaptée aux utilisateurs qui courent.

D'autre part, [Jöst 2005] met en évidence que certaines modalités d'interaction comme la parole sont de moins en moins utilisée lorsque l'utilisateur est privé d'intimité. Dans ces expérimentations sur un système de navigation et d'information pour les piétons, les utilisateurs interagissent dans plusieurs lieux d'interaction (bars, zone piétonne, tramway, etc.). L'utilisation de la parole décroît de plus en

plus selon le degré d'intimité de l'utilisateur avec les personnes qui l'entourent. Lorsqu'elle/il est seul(e), l'utilisateur utilise fréquemment la parole. Cette fréquence décroît lorsqu'elle/il est entouré(e) de sa famille. Cet effet est accentué quand les personnes qui l'entourent sont ses amis, et devient épisodique lorsque ce sont des étrangers qui sont proches d'elle/lui.

De plus, les variations temporaires de la situation d'interaction peuvent modifier le choix des modalités d'interaction. Par exemple, un utilisateur qui porte un enfant peut temporairement être dans l'incapacité d'utiliser la tablette tactile d'une borne d'information publique, alors que l'utilisation de la parole est toujours possible. Ainsi, un système qui offre ces deux modalités d'interaction, de façon équivalente, permet à l'utilisateur d'adapter son choix en fonction des circonstances environnementales changeantes [Oviatt 2000].

Le contexte variable est un paramètre déterminant qui favorise une modalité d'interaction au détriment de telle autre. Comme pour l'aspect variable des spécificités d'un utilisateur, la variabilité du contexte modifie l'usage des modalités d'interaction par les utilisateurs.

2.2 Qualité de l'interaction mettant en œuvre plusieurs modalités d'interaction

Cette section identifie les apports majeurs de l'existence de plusieurs modalités d'interaction dans un même système interactif. Les travaux étudiés mettent en évidence les préférences d'usage des utilisateurs et les bénéfices apportés par les différentes modalités d'interaction présentant des relations d'équivalence, de redondance ou de complémentarité.

Rappelons que l'usage de modalités d'interaction redondantes ou complémentaires n'implique par forcément des signaux simultanés [Oviatt 1999]. En effet, le geste précède souvent la parole (99% des cas sont séquentiels), même lorsque les deux modalités d'interaction dénotent des informations synchrones comme les déictiques. Il n'y a finalement que 25% des énoncés qui sont véritablement simultanés (parallèle). Ainsi, l'utilisation simultanée de deux modalités d'interaction n'implique pas des signaux simultanés.

Plusieurs apports des combinaisons entre modalités d'interaction ont été observés lors d'expérimentations sur les usages. Nous présentons les résultats principaux constituant des règles de conception d'un système interactif multimodal performant et adapté aux besoins des utilisateurs.

2.2.1 Equivalence des modalités d'interaction pour une souplesse de l'interaction

Comme annoncé précédemment lors de l'étude des paramètres influençant le choix des modalités d'interaction, les systèmes multimodaux, qui proposent une équivalence entre les modalités d'interaction, offrent une souplesse aux utilisateurs. En effet, cette souplesse permet de répondre aux préférences des utilisateurs selon leurs spécificités individuelles, aux variations du contexte d'interaction ou encore de la tâche à accomplir. La variabilité des capacités humaines et du contexte d'interaction favorise la mise en œuvre d'une équivalence de modalités d'interaction, pour laisser la possibilité aux utilisateurs d'utiliser à tout moment les modalités d'interaction les plus adéquates.

Cependant, [Oviatt 1999] souligne que le pouvoir d'expression de diverses modalités d'interaction est différent. Même si parfois deux modalités d'interaction se rapprochent et sont équivalentes (au sens de CARE et TYCOON présentés respectivement aux sections 1.2 et 1.3 de ce chapitre), elles différent par leurs propriétés : précision, latence, etc. De plus, certaines modalités d'interaction sont plus inconscientes ou passives que d'autres : la direction du regard par exemple est plus inconsciente que la désignation par le doigt. Ces aspects ne doivent pas être négligés lors de la conception du système car leur prise en compte peut permettre une meilleure efficacité (rapidité, précision, etc.) du système multimodal et une baisse de l'effort cognitif de l'utilisateur.

L'équivalence de plusieurs modalités d'interaction (au sens de CARE et TYCOON présentés respectivement aux sections 1.2 et 1.3 de chapitre) au sein du même système interactif apporte des avantages indéniables pour s'adapter à un large public et à des conditions d'utilisation variables (principalement pour les situations mobiles). La souplesse de l'interaction augmente l'utilisabilité du système interactif.

2.2.2 Redondance de l'interaction pour une meilleure fiabilité

[Oviatt 1999] montre que l'équivalence de plusieurs modalités d'interaction ne favorise pas l'utilisation redondante de ces moyens d'interaction par les utilisateurs. Cela ne va pas dans un sens d'économie du point de vue de l'utilisateur et entraîne une augmentation du coût cognitif (l'utilisateur doit utiliser plusieurs modalités au lieu d'une seule). Même si l'utilisateur interagit de façon redondante, cet usage est rare. Même dans le cas d'échec puis d'essais de correction, il a été observé, expérimentalement, que l'usage redondant n'augmente pas de façon significative.

La redondance est néanmoins utilisée dans une recherche de fiabilité. Obliger l'utilisateur à spécifier un énoncé par au minimum deux modalités d'interaction équivalentes assure de ses intentions et évite la propagation de commandes non intentionnelles. Par exemple, l'utilisation de la reconnaissance vocale entraîne parfois des erreurs d'énoncés qui impliquent des commandes erronées. L'utilisation redondante de la reconnaissance de la parole et d'une autre modalité d'interaction renforce la fiabilité de l'interaction globale. La redondance est ainsi complètement adaptée aux tâches critiques. Nous pouvons imaginer, par exemple, que l'arrêt d'un serveur de données sensibles soit réalisé, de manière redondante, par une confirmation vocale simultanément à la confirmation sur l'interface graphique, évitant ainsi tout arrêt involontaire.

Même si l'usage est observé épisodiquement, l'utilisation redondante de deux modalités d'interaction n'est pas utilisée intuitivement par les utilisateurs. La redondance est pertinente pour assurer la fiabilité de l'énoncé capturé par le système informatique.

2.2.3 Complémentarité des modalités d'interaction pour faciliter la réalisation de tâches complexes

Plusieurs études telles [Jöst 2005] ou [Oviatt 2004] soulignent que la richesse sémantique de l'action favorise la complémentarité entre plusieurs modalités d'interaction. Elles montrent que cette complémentarité est de plus en plus utilisée quand la tâche est de plus en plus complexe.

Plus précisément, l'étude de [Oviatt 2004] porte sur un système informatique de coordination des secours lors d'inondation d'une ville et met en œuvre deux modalités d'interaction naturelles, pouvant être utilisées de manière complémentaire, équivalente ou redondante : il s'agit de la reconnaissance de la parole et de l'utilisation d'un stylet comme pointeur (geste de désignation). Les tâches proposées à l'utilisateur sont classées dans quatre catégories (bas, moyen, haut, très haut) en fonction du nombre de paramètres de la commande. Les résultats montrent que 61,8% des commandes sont réalisées par l'utilisation complémentaire des deux modalités d'interaction. De plus, les commandes multimodales pour les tâches de complexité faible (niveau bas) représentent 59,2% du total, 65,5 % pour les tâches de complexité moyenne (niveau moyen), 68,2% pour les tâches de haute complexité (niveau haut) et 75% pour les tâches de très haute complexité (niveau très haut). Cet accroissement reflète l'effort

des utilisateurs pour gérer les limitations de leur mémoire de travail, engendrées par l'augmentation de la complexité des tâches, en distribuant l'information à communiquer sur plusieurs modalités d'interaction.

L'expérimentation de [Oviatt 2004] souligne que plus la tâche est complexe, plus la multimodalité est employée. Il semble ici que, pour l'utilisateur, le coût cognitif de l'utilisation conjointe de deux modalités d'interaction soit négligé au profit du niveau de performance du résultat obtenu lorsque le nombre de paramètres de la tâche est élevé.

2.2.4 Complémentarité des modalités d'interaction pour une recherche de l'efficacité

Un système qui propose une complémentarité entre plusieurs modalités d'interaction est souvent considéré comme plus efficace qu'un système monomodal. En effet, les expérimentations de [Cohen 2000] et [Jöst 2005] concluent que l'utilisation complémentaire de plusieurs modalités d'interaction est plus efficace en termes de rapidité que l'utilisation d'une interface classique. Bien qu'utilisé pour les tâches complexes, les langages d'interaction utilisés de façon complémentaire sont syntaxiquement moins complexes : les énoncés sont plus compacts, moins ambigües et ainsi plus efficaces.

Cependant, cette efficacité n'est pas garantie quand une modalité d'interaction est particulièrement performante par rapport à une combinaison de plusieurs modalités d'interaction. L'étude [Oviatt 1999] montre qu'une commande multimodale est souvent plus longue à exprimer qu'une commande monomodale, car il y a un coût cognitif dû à la multimodalité (la multimodalité produit un débit de parole plus saccadé et des hésitations plus fréquentes). Par exemple, [Kaster 2003] propose plusieurs choix d'interaction en entrée pour réaliser différentes tâches sur des images présentées dans une mosaïque (sélection, défilement des images, etc.) : soit l'utilisateur utilise indépendamment la souris ou l'écran tactile, soit il utilise chacune des deux modalités d'interaction combinée à la parole (selon le paradigme du "put-that-there" de [Bolt 1980] présenté au chapitre 1). Dans l'une des expérimentations, dont le but était de rechercher une image précise dans la mosaïque, l'utilisation de l'écran tactile est la modalité d'interaction qui s'est avérée la plus rapide et les deux cas d'utilisation combinée correspondent aux temps les plus longs. Si nous réduisons l'efficacité à la rapidité de traitement, en

ignorant par exemple la précision, les modalités d'interaction combinées sont dans ce cas moins efficaces que l'utilisation indépendante des modalités d'interaction. L'utilisation dans un contexte de travail réel, notamment avec un entrelacement de tâches, aurait probablement donné d'autres résultats.

L'utilisation complémentaire de plusieurs modalités d'interaction permet, dans la plupart des cas, d'améliorer l'efficacité de l'interaction par rapport à l'utilisation d'une interface classique, notamment lorsqu'il s'agit de réaliser des tâches complexes. Néanmoins, il arrive que pour certaines tâches simples, l'utilisation d'une seule modalité d'interaction soit bien plus efficace.

3 Résumé du chapitre 3

Dans ce chapitre, consacré aux multiples usages des modalités d'interaction, les résultats conceptuels ont d'abord été exposés, suivis des résultats expérimentaux.

Les résultats conceptuels constituent des cadres de référence simples, directement applicables. Les systèmes multimodaux peuvent être caractérisés avec le modèle UOM, permettant de préciser l'unicité ou la multiplicité des dispositifs et des langages d'interaction, leur optionalité d'utilisation et les différentes formes d'usage (en fonction du temps et de la nécessité de combiner les modalités d'interaction). Particulièrement, CARE et TYCOON caractérisent les relations possibles entre les modalités d'interaction, en définissant notamment les deux formes de combinaison possibles avec les propriétés de complémentarité et de redondance.

Complémentaires aux cadres conceptuels, les résultats expérimentaux guident les concepteurs pour la mise en œuvre des modalités d'interaction et de leurs relations possibles dans le système interactif à concevoir. Dans les expérimentations considérées, les utilisateurs étaient libres d'utiliser les modalités d'interaction comme ils le souhaitaient. Il a été observé ainsi que le choix dépend de plusieurs paramètres tels la tâche à accomplir, leur spécificité individuelle, le contexte ou encore le mode d'implémentation de la modalité d'interaction. Concernant les relations entre les modalités d'interaction, les usages montrent les bénéfices apportés par l'utilisation de l'équivalence, de la redondance ou de la complémentarité de plusieurs modalités d'interaction. L'équivalence apporte de la souplesse dans l'interaction car elle permet de s'adapter à un large public et à des contextes variés. La redondance, parfois utilisée de façon naturelle par un utilisateur, est employée plutôt pour une recherche de la fiabilité. Enfin, la complémentarité apporte une réponse à la complexité des tâches et peut s'avérer être plus efficace que d'autres moyens d'interaction.

Ainsi, la multimodalité modifie les usages des utilisateurs en améliorant la souplesse et la robustesse de l'interaction générale. Néanmoins et comme nous l'avons vu, les modèles actuels ne prennent pas en compte toutes les caractéristiques de la multimodalité et seule leur unification permet de répondre à cet objectif. Ce but est l'objet du chapitre suivant, qui expose un modèle conceptuel rassemblant les divers concepts énoncés dans les chapitres précédents.

CHAPITRE 4
Modèle conceptuel de la multimodalité : Un premier pas vers la theorie

Ce chapitre conclut la première partie de ce mémoire sur les espaces de conception. Il constitue une contribution importante au domaine de la multimodalité, définissant un modèle conceptuel qui ouvre à la mise en place d'une théorie générale sur la multimodalité (voir l'encadré ci-dessous pour un rappel terminologique de théorie). Au vue de la grande variabilité des concepts, soulignée dans les chapitres précédents, due notamment à la relative jeunesse du domaine, l'élaboration d'une théorie globale et unificatrice est un travail complexe, sur le long terme, afin de couvrir des cas de multimodalité non encore identifiés, ni observés à ce jour. Notre objectif est néanmoins d'organiser les principaux concepts intervenant pour décrire et expliciter les phénomènes multimodaux, dans les interfaces multimodales existantes et émergentes, avec notamment la prise en compte des nouveaux paradigmes d'interaction telle la réalité augmentée.

Terminologie : Théorie

Le mot théorie vient du mot grec *theorein*, qui signifie contempler, observer, examiner. Dans le langage courant, une théorie est une idée ou une connaissance spéculative, souvent basée sur l'observation ou l'expérience, donnant une représentation idéale, éloignée des applications. (…) En sciences, une théorie est un modèle ou un cadre de travail pour la compréhension. [Wikipedia 2006]

Plus officiellement, une théorie est un ensemble organisé d'idées, de concepts abstraits prenant pour objet un domaine particulier qu'elle décrit et explicite [Robert 1998].

La démarche de travail pour l'établissement de notre modèle conceptuel se fonde sur l'analyse des différents travaux présentés dans les chapitres précédents. Nous commençons par définir les concepts de manière générale, puis nous les approfondissons tout au long de ce chapitre. Ainsi, une première section organise les principaux concepts au sein d'un méta-modèle de l'interaction multimodale. Ensuite, ces concepts fondamentaux sont décrits précisément par l'identification et la classification de propriétés. Enfin, nous illustrons notre modèle conceptuel sur un exemple.

L'élaboration d'un méta-modèle de l'interaction multimodale est un défi intéressant car il implique de mettre clairement et simplement en évidence les concepts fondamentaux intervenant dans une telle interaction. Cette section propose, en deux étapes, de positionner les notions essentielles identifiées dans les chapitres précédents, en les modélisant par un diagramme de classes UML [UML 2006]. Le diagramme de classes UML facilite la description des types d'entités (classe) et leur relation dans une activité multimodale. De cette manière, le cadre général de l'interaction multimodale est d'abord modélisé, suivi par la modélisation des possibilités d'interaction. Il est à noter que l'ensemble des caractéristiques qui décrit chacune des classes n'est pas abordé dans cette section, étant présenté en détail dans les sections suivantes de ce chapitre.

1.1 CADRE GÉNÉRAL

Le cadre de base de l'interaction homme-machine est modélisé par le diagramme de classes de la Figure 4.1. Comme identifié dans les travaux sur les situations d'activité instrumentée [Rabardel 1995], le but de l'utilisateur est d'agir sur un ou plusieurs objets numériques ou réels représentés ici par la classe "objet du domaine". Pour agir sur un ou plusieurs objets du domaine, l'utilisateur effectue une ou plusieurs tâches. Une tâche est un but à atteindre dans des conditions déterminées [Leplat 1997]. L'association Utilisateur-Tâche révèle un contexte particulier qui comprend, entre autre, la situation environnementale. Le contexte est un domaine de recherche complexe et très actif que nous n'essayerons pas de cerner dans ce mémoire. Pour information, plusieurs travaux sur le contexte, dont [Rey 2005], sont disponibles dans la littérature.

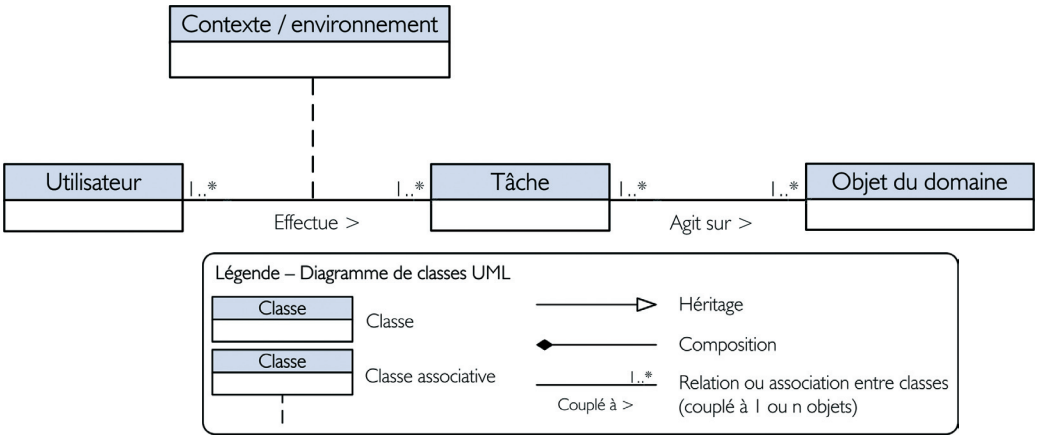


Figure 4.1 : diagramme de classes UML du cadre général de la multimodalité, étape 1 : le cadre général.

1.2 Possibilités d'interaction

L'intérêt des travaux sur la multimodalité réside dans les possibilités d'interaction qui sont offertes à l'utilisateur pour réaliser sa tâche. C'est ce point précis qui intéresse nos recherches et que nous modélisons. Les possibilités d'interaction couvrent aussi bien l'ensemble des modalités d'interaction que leurs possibles combinaisons. La Figure 4.2 illustre les différentes classes nécessaires et suffisantes pour représenter l'ensemble des possibilités d'interaction multimodale. Nous détaillons d'abord la notion de modalité d'interaction, avant de focaliser sur la description des combinaisons entre modalités d'interaction.

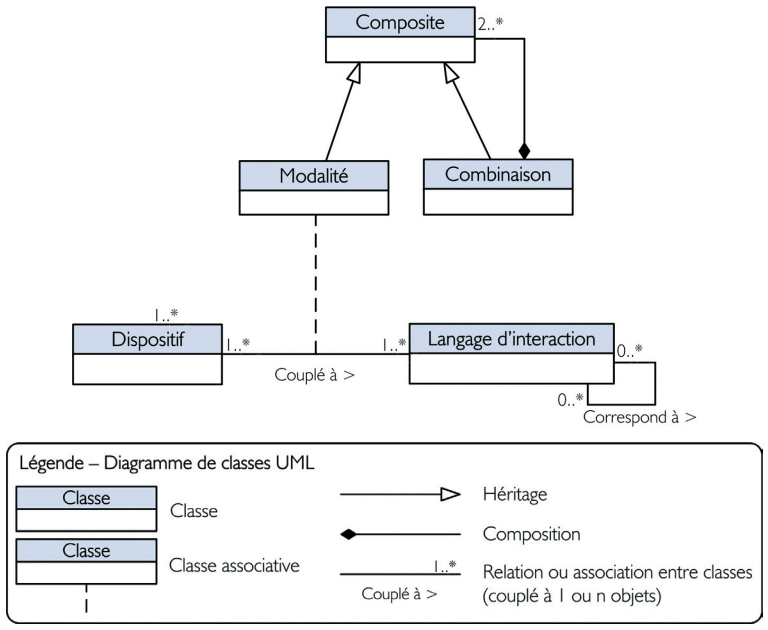


Figure 4.2 : diagramme de classes UML des possibilités d'interaction de la multimodalité.

1.2.1 Modalité : deux niveaux d'abstraction

L'association d'un dispositif et d'un langage d'interaction, permet d'obtenir une modalité d'interaction. La définition que nous utilisons ici est celle présentée par [Nigay 1996], où une modalité est un couple entre un dispositif physique et un langage d'interaction. En accord avec [Beaudouin-Lafon 2000], ces deux niveaux d'abstraction, le dispositif comme bas niveaux d'abstraction et le langage d'interaction comme plus haut niveau d'abstraction, sont complètement pertinents dans la démarche d'un développement logiciel, car ils font apparaître les diverses transformations des données que l'ordinateur doit effectuer pour répondre aux actions physiques de l'utilisateur. De plus, cette décomposition coïncide avec les

étapes mentales et physiques que l'utilisateur réalise pour spécifier ses intentions au système informatique (cf. Figure 2.13 du chapitre 2).

En complément, les différentes possibilités d'optionalité et de multiplicité entre dispositifs physiques et langages d'interaction identifiées dans UOM [Nigay 1994] (cf. section 1.1 du chapitre 3) sont modélisées. Nous précisons que plusieurs langages d'interaction peuvent correspondre à un dispositif (cardinalité 1..*) ; c'est le cas de la souris qui sert à la manipulation directe dans un environnement WIMP classique, ou qui peut être utilisée dans un langage de reconnaissance gestuel en analysant ses déplacements. Symétriquement, un langage d'interaction peut correspondre à plusieurs dispositifs, comme la manipulation directe d'un curseur souris, qui peut être déplacé par la souris ou par un pavé tactile. Il est remarquable que le méta-modèle présenté propose une possible association entre langages d'interaction, traduisant l'éventualité d'utiliser un langage d'interaction compatible comme transition vers un autre langage d'interaction. Par exemple, le langage d'interaction de la langue naturelle française peut servir d'intermédiaire pour un utilisateur français, afin de spécifier une commande en langue naturelle japonaise. Nous aurions ainsi deux langages d'interaction, le français comme langage d'interaction intermédiaire et le japonais comme langage d'interaction final. Cet exemple correspond à la propriété de transfert définit dans l'espace TYCOON (cf. section 1.3 du chapitre 3).

1.2.2 Modalités et combinaisons

Les possibilités d'interaction s'élargissent en multimodalité par la combinaison potentielle des modalités d'interaction présentes. Cet ensemble de possibilités est représenté par le patron UML composite de [Gamma 1996]. Ainsi, la classe composite représente l'ensemble des possibilités : la classe composite est, par la relation d'héritage, soit une modalité simple (couple dispositif/langage d'interaction), soit une combinaison entre plusieurs instances des classes modalité ou combinaison. Nous avons décidé de représenter explicitement, par une classe, les différentes combinaisons possibles entre les modalités, au lieu d'une simple relation, car les combinaisons représentent un concept fondamental que nous caractériserons dans les sections suivantes. Il est à noter que plusieurs couples de modalités combinées peuvent aussi se combiner entre elles, d'où la pertinence de ce patron UML et de la cardinalité 2..* pour le lien de composition. La classe Composite représente ainsi, aussi bien une modalité simple que plusieurs modalités combinées.

1.3 Synthèse

La mise en relation des deux diagrammes précédents permet d'appréhender l'interaction multimodale dans sa globalité. Pour réaliser ses tâches, l'utilisateur dispose d'une ou plusieurs modalités d'interaction potentiellement combinées entre elles. La Figure 4.3 donne le méta-modèle complet de l'utilisation d'un système interactif multimodal par l'ajout, premièrement, d'un lien entre la classe Utilisateur et la classe Dispositif, et deuxièmement, d'un lien entre la classe Composite (modalité d'interaction ou combinaison de modalités d'interaction) et la classe Tâche.

Une contrainte doit cependant être établie pour la compréhension correcte de ce schéma. Pour que l'application soit utilisable, il faut en effet préciser qu'il existe toujours une modalité ou combinaison de modalités (classe Composite) à la

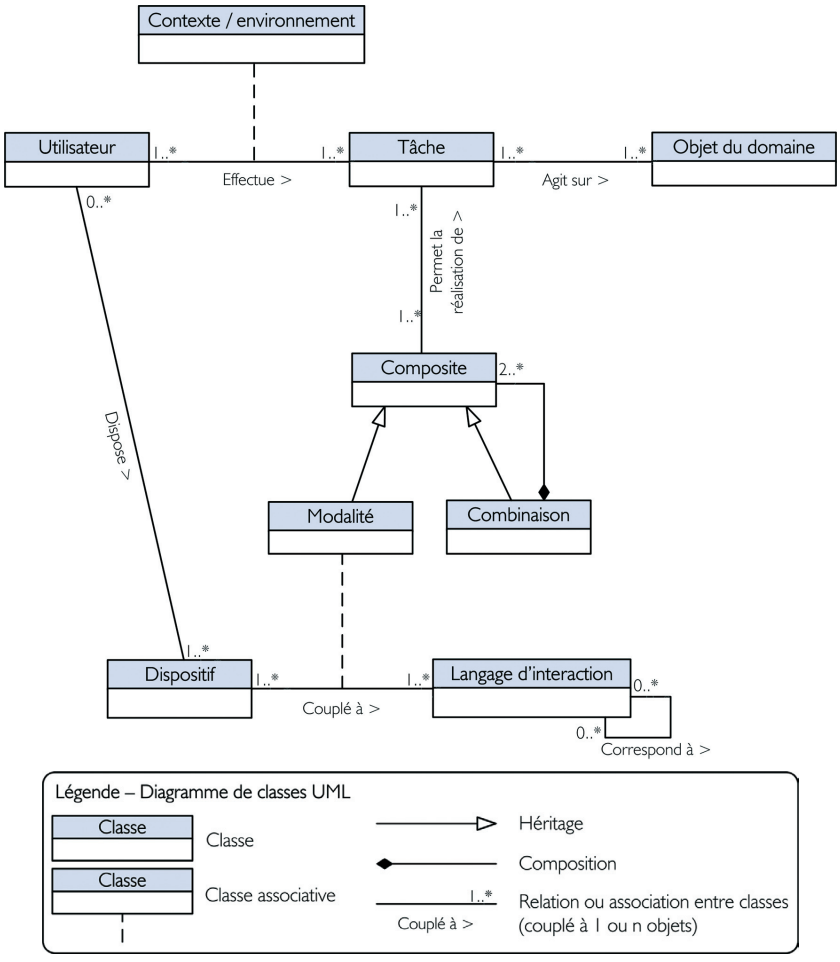


Figure 4.3 : diagramme de classes complet des concepts de la multimodalité.

disposition de l'utilisateur pour réaliser les tâches de l'application. Pour effectuer une tâche, l'utilisateur utilise obligatoirement une ou plusieurs modalités d'interaction.

Ce diagramme de classes UML de l'interaction multimodale résume, par la mise en relation de neuf classes, les concepts fondamentaux identifiés dans les travaux des chapitres précédents. Pour qu'il soit utilisable dans une phase de conception et de développement, ce diagramme doit être complété par la description précise de chacune des classes concernant les possibilités d'interaction. Cette description des classes fait l'objet des sections suivantes.

2 Modalités d'interaction

Une modalité d'interaction est l'association d'un dispositif et d'un langage d'interaction. Cette section propose une description approfondie de ces deux facettes, permettant de diriger et conforter les choix des acteurs lors des phases de conception et de développement de l'interaction multimodale. Avant de détailler chacune de ces deux classes, nous revenons sur les caractéristiques générales aux modalités d'interaction.

2.1 Caractéristiques globales

La synthèse des travaux sur la multimodalité révèle des caractéristiques communes aux modalités d'interaction. Cette section rassemble et décrit les propriétés globales des couples <dispositif, langage d'interaction>.

Trois propriétés caractérisant les modalités d'interaction ressortent de l'analyse des travaux présentés dans les chapitres précédents. Il s'agit de l'adéquation interne et externe d'une modalité d'interaction, du degré d'indirection entre la modalité d'interaction et l'objet manipulé, et enfin du type d'interaction qu'effectue un utilisateur lorsqu'il transmet des informations au système informatique. Ces trois caractéristiques sont maintenant approfondies.

2.1.1 Adéquation entre le dispositif, le langage d'interaction et la tâche

Défini par [Lachenal 2004] pour les instruments, la propriété d'adéquation pour une modalité d'interaction désigne sa capacité à faciliter les actions de l'utilisateur. Il s'agit de définir l'adéquation d'une modalité d'interaction avec le fonctionnement cognitif et physique d'un utilisateur.

L'adéquation se décline selon deux niveaux orthogonaux : le premier correspond à l'adéquation interne entre le dispositif et le langage d'interaction, le second définit l'adéquation externe du couple <dispositif, langage d'interaction> avec la tâche à réaliser. L'adéquation peut être définie par le rapport entre le degré d'intégration (premier niveau) et le degré de compatibilité (second niveau) de [Beaudouin-Lafon 2000]. Le degré d'intégration est le rapport entre le nombre de dimensions du dispositif physique et du langage d'interaction. Par exemple, il est égal à 0,5 pour une barre de défilement 1D manipulée par une souris 2D. Le degré de compatibilité correspond aux similarités entre les manipulations effectuées sur l'instrument et sa réaction sur l'objet manipulé. Sa métrique est subjective : très faible, faible,

moyenne, forte, très forte. Par exemple, la barre de défilement possède un degré de compatibilité plus faible que le glisser-déposer. En effet, lorsque la barre de défilement descend, le document monte. Par contre, lors d'un glisser-déposer, l'objet manipulé suit les mouvements de la souris. Cette dernière caractéristique est complètement adaptée aux modalités d'interaction dans une interface classique WIMP, mais elle n'est pas généralisable à toutes les modalités d'interaction. Pour notre étude, le niveau métaphorique identifié pour les interfaces tangibles par [Fishkin2004] est plus pertinent et peut remplacer le degré de compatibilité. En effet, le niveau métaphorique caractérise l'analogie des effets de l'interaction par rapport aux effets des actions similaires dans le monde réel. Les valeurs possibles sont "aucun", "nom" ou "verbe", "nom et verbe", "total". La valeur "nom" se réfère à la forme, alors que la valeur "verbe" se rapporte à la notion de mouvement. Comme exemple, imaginons l'adéquation d'une souris comme pointeur dans les interfaces WIMP, le degré d'intégration est égal à 1 (dimension du dispositif = 2D / dimension du langage d'interaction = 2D) et le niveau métaphorique est "total" car l'analogie avec le monde réel est complètement vérifiée. Au contraire, si nous mesurons l'adéquation d'un joystick 3D qui, quand on l'incline vers le haut, permet d'éteindre l'ordinateur, le degré d'intégration est 3 (dimension du dispositif = 3D / dimension du langage d'interaction = 1D) et le niveau métaphorique est "aucun" car il n'y a pas d'analogie avec le monde réel.

La Figure 4.4 résume l'adéquation entre le dispositif, le langage d'interaction et la tâche, en montrant les deux axes qui la caractérisent. Une modalité d'interaction optimale a un niveau métaphorique élevé et un degré d'intégration égal à 1.

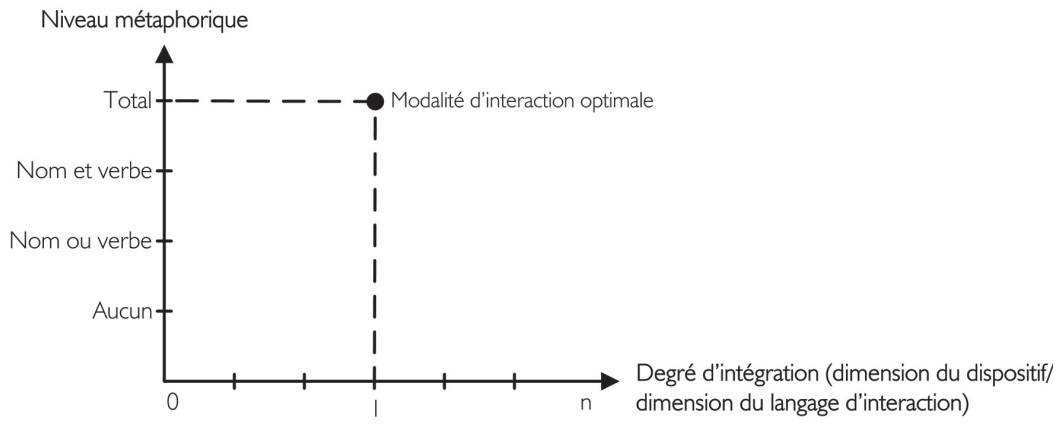


Figure 4.4 : adéquation entre le dispositif, le langage d'interaction et la tâche, rapport entre le niveau métaphorique et le degré d'intégration.

2.1.2 Degré d'indirection

Identifié et caractérisé par [Beaudouin-Lafon 2000] pour les instruments, le degré d'indirection correspond à la relation spatiale et temporelle entre la modalité d'interaction et l'objet manipulé. Cette propriété relève l'importance du cheminement pour la réalisation d'une tâche. Elle renseigne le niveau de performance physique/cognitive (utilisateur) et computationnelle (ordinateur) requis pour la réalisation de cette tâche. En effet, un degré d'indirection élevé est préjudiciable pour l'utilisateur et l'ordinateur, car le coût lié au traitement de l'action est supérieur à celui d'une modalité d'interaction de degré d'indirection plus faible. Comme exemple, la fermeture d'une fenêtre, par l'icône dédiée, a un degré d'indirection plus faible que cette même fermeture par le menu.

Dans son cadre d'étude sur l'interaction instrumentale, [Beaudouin-Lafon 2000] propose de mesurer le degré d'indirection, comme un rapport entre un indice spatial et un indice temporel. L'indice spatial correspond à la distance qui sépare la représentation de l'instrument et l'objet de la tâche (dans le cas d'une interaction WIMP). L'indice temporel représente la différence de temps entre l'action physique et la réponse de l'objet manipulé.

Définir le degré d'indirection pour chacune des modalités d'interaction offre un critère de choix pour des modalités équivalentes. Par exemple, pour deux modalités d'interaction équivalentes, c'est celle qui présente le degré d'indirection le plus faible qui doit être favorisée et utilisée. Cependant, notre métrique ne peut être la même que celle proposée par [Beaudouin-Lafon 2000] car elle ne peut s'appliquer à toutes les modalités d'interaction. L'indice spatial doit être adapté et le niveau d'incarnation proposé pour les interfaces tangibles par [Fishklin 2004] semble mieux correspondre au cas de la multimodalité. Le niveau d'incarnation représente la distance entre le focus d'entrée et celui de sortie. Quatre valeurs sont possibles : "total", "rapproché", "ambiant", "éloigné". Pour illustration, considérons l'exemple du pinceau I/O brush [Ryokai 2004] présenté au chapitre I de ce mémoire, l'indice temporel est proche de 300 ms (estimation au vue des vidéos de présentation) et le niveau d'incarnation est rapproché car le focus d'entrée se situe sur le pinceau alors que le focus de sortie se trouve sur la surface peinte. Au contraire, l'utilisation d'un système de reconnaissance vocale pour manipuler un ordinateur qui se trouve à plusieurs mètres de l'utilisateur présente un indice temporel proche de quelques secondes et un niveau d'incarnation éloigné.

La Figure 4.5 montre les deux axes qui caractérisent le degré d'indirection d'une modalité d'interaction. Une modalité d'interaction optimale répond par un niveau d'incarnation totale et un indice temporel égal à 0, correspondant au temps-réel.

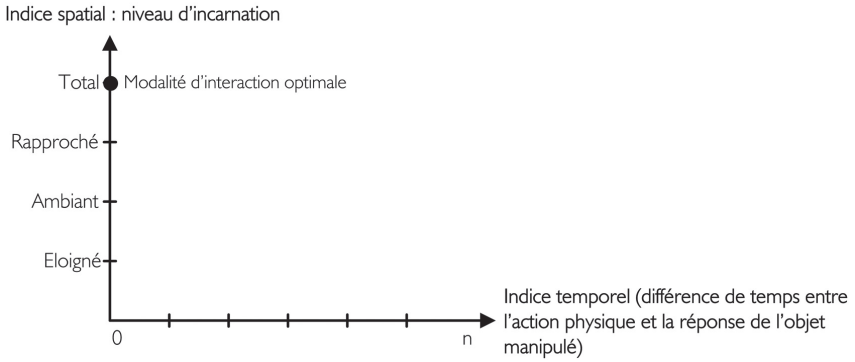


Figure 4.5 : degré d'indirection d'une modalité d'interaction, rapport entre le niveau d'incarnation et la différence de temps entre l'action physique et la réponse de l'objet manipulé.

2.1.3 Type d'interaction

[Renevier 2004b] distingue deux types d'usage pour les systèmes de réalité augmentée. Soit l'utilisateur spécifie explicitement l'augmentation (le type d'interaction est actif), soit l'augmentation est déterminé automatiquement sans intervention de l'utilisateur (le type d'interaction est passif). Cette distinction est complètement adéquate aux modalités d'interaction et présente nombre d'intérêts pour le choix ou la combinaison des modalités d'interaction dans un système multimodal. Ainsi, deux catégories de modalités sont identifiées : les modalités d'entrée actives et passives.

Modalité active

Une modalité d'entrée est qualifiée d'active quand l'utilisateur doit réaliser une action explicite avec un dispositif en vue de spécifier une commande au système. Par exemple, nous pouvons citer la parole <microphone, langage pseudo-naturel> comme une modalité active. De même, dans les interfaces manipulables, la modalité <ordinateur de poche, langage gestuel> constitue une modalité active, l'utilisateur incline par exemple l'ordinateur de poche pour faire défiler une liste. Lors de l'utilisation d'une modalité d'interaction active, la charge mentale et physique de l'utilisateur n'est pas négligeable ; la distance d'exécution humaine peut même s'avérer importante lorsque plusieurs modalités actives sont utilisées simultanément. Notre objectif consiste à la minimiser en identifiant le meilleur compromis cognitif introduit comme le point B à la Figure 2.4. La distance d'évaluation technologique est importante car il faut bien assimiler les intentions de l'utilisateur. Cette distance se complexifie en cas d'utilisation combinée de plusieurs modalités actives.

Modalité passive

Une modalité d'interaction est caractérisée de passive quand le dispositif qui la constitue ne requiert pas l'attention et d'action explicite de l'utilisateur. Par exemple, la capture par un GPS de la localisation d'un utilisateur correspond à la modalité passive décrite par le couple <GPS, localisation en données GPS>. De nombreuses interfaces exploitent des modalités d'entrée passives qui semblent rendre l'interaction plus robuste et efficace. La distance d'exécution humaine est ici considérée comme nulle. En effet, l'utilisateur ne dédie aucune charge mentale, ni physique, pour que des données soit transmises au système informatique via cette modalité passive. Lorsque le système informatique combine des données issues de deux modalités passives pour obtenir une unité informationnelle pertinente, le compromis cognitif ne s'applique pas et l'utilisation de ces modalités est considérée comme une gestion optimale des ressources utilisateur, correspondant au point A de la Figure 2.4. Dans le même sens, une modalité passive couplée à une modalité active simplifie la gestion des ressources de l'utilisateur. Le compromis cognitif ne concerne alors que la modalité active et il se situe sur la ligne pointillée de la Figure 2.4. Au contraire, la distance d'évaluation technologique (correspondant aux étapes de traitement du système informatique, cf. Figure 2.12) reste importante car l'interprétation de ces données par le système informatique implique leur fusion.

Les caractéristiques globales des modalités d'interaction permettent de les classer en différents types. Dans le cas de modalités d'interaction équivalentes, ces propriétés sont de véritables critères de choix pour concevoir et développer une interaction multimodale utile, performante et adaptée aux utilisateurs. L'approfondissement par la caractérisation des deux éléments constituant une modalité d'interaction (couple <dispositif, langage d'interaction>) doit guider plus avant les choix lors des phases de conception et de développement logiciel. Les deux sections suivantes proposent de décrire un dispositif, puis un langage d'interaction.

2.2 Dispositifs

2.2.1 Définition

Un dispositif représente la réalité physique d'entrée ou de sortie du système. Il s'agit du niveau d'abstraction le plus bas d'une modalité d'interaction. Pour notre étude, les dispositifs d'entrée regroupent les entités qui acquièrent des informations, de la part de l'utilisateur, pour les communiquer au système informatique. Le clavier, la souris, un GPS ou encore un microphone constituent des exemples de dispositifs d'entrée.

2.2.2 Propriétés

Cette section rassemble l'ensemble des propriétés qui caractérisent les dispositifs d'une modalité d'interaction et qui fournissent les éléments essentiels pour effectuer le meilleur choix possible. Trois catégories de propriétés sont identifiées en fonction de l'objectif de leur définition. La première catégorie définit les propriétés intrinsèques des dispositifs, telles que leurs dimensions ou leur poids. La deuxième catégorie regroupe les propriétés nécessaires pour déterminer si le dispositif est réellement adapté aux besoins et aptitudes de l'utilisateur cible en situation. Enfin, nous retrouvons les propriétés pertinentes et utiles lors de l'utilisation du dispositif. Alors que cette section décrit l'ensemble des propriétés, la section 2.4 propose des recommandations les exploitant.

Propriétés intrinsèques du dispositif

Ces propriétés représentent les spécificités propres au dispositif, sans tenir compte de son utilisation.

Nom : cette propriété essentielle permet d'identifier le dispositif sous la forme d'un texte.

Version : cette propriété est une description textuelle de la version courante du dispositif. Elle permet de renseigner les développeurs du statut actuel du dispositif, des mises à jour éventuelles, etc.

Dimensions : ces propriétés renseignent de la forme et de la taille du dispositif. Le domaine de valeur dépend du dispositif. Cependant, la plupart des dispositifs peuvent être décrits par leur largeur / hauteur / profondeur, données en centimètres.

Poids : cette propriété définit la masse du dispositif. Il doit être spécifié en kilogrammes.

Autonomie : avec la mobilité, l'utilisation d'un système informatique se réalise, de plus en plus, dans des contextes variés. L'autonomie d'un dispositif devient un critère déterminant de choix. Les valeurs peuvent être mesurées en heures. Une valeur infinie indique que le dispositif n'a pas besoin d'alimentation électrique, comme un microphone par exemple. Une valeur inconnue indique que le dispositif est autoalimenté par un autre dispositif, comme pour une souris optique avec fil par exemple.

Nombre de dimensions : cette propriété représente le nombre de degrés de liberté

du dispositif, comme identifié dans l'espace de [Buxton 83]. Par exemple, le joystick a trois degrés de liberté. En comparant le nombre de dimensions du dispositif avec celui du langage d'interaction auquel il est couplé, l'adéquation interne d'une modalité d'interaction peut être définie (cf. section 2.1.1 de ce chapitre).

Précision : cette propriété mesure l'exactitude des données propagées par les actions de l'utilisateur. Le domaine de valeur dépend du dispositif en question. Par exemple, la précision d'un GPS est au mètre près.

Résolution : cette propriété désigne la plus petite variation perceptible de la grandeur à mesurer, dans des conditions de mesures données [Lachenal 2004]. Le domaine de valeur dépend du dispositif en question.

Stabilité : identifiée dans [Lachenal 2004], cette propriété définit si, en l'absence d'action de la part de l'utilisateur, la grandeur mesurée ne varie pas. Le domaine de valeur est "vrai" ou "faux". Par exemple, la souris est stable, le GPS ne l'est pas.

Domaine des valeurs de sortie : comme pour l'espace de [Card 1990], le domaine des valeurs de sortie définit la nature des données transmises par le dispositif. Son domaine de valeur dépend du dispositif. Par exemple, un capteur d'orientation 3D peut fournir trois angles en degré dans l'intervalle [0…360[. Cette propriété est à mettre en correspondance avec le domaine des valeurs d'entrée des langages d'interaction, afin de savoir si le dispositif peut être couplé avec un langage d'interaction particulier.

Temps de traitement moyen : le temps de traitement moyen est une estampille de temps, en millisecondes, qui représente le temps moyen que met le dispositif pour traiter une donnée.

Fréquence des données : cette propriété renseigne sur la fréquence de transmission des données par le dispositif. Deux valeurs sont possibles : valeur "continue" ou "discrète". Par exemple, un GPS fournie la position d'un utilisateur toutes les n millisecondes (valeur continue), alors qu'un clavier transmet uniquement des données lorsqu'un utilisateur actionne les touches (valeur discrète).

Propriétés relatives à l'utilisation humaine

Ces propriétés permettent aux concepteurs, aussi bien informaticiens que non informaticiens, de choisir et de mettre en œuvre des dispositifs adaptés à l'utilisateur cible et à sa situation d'interaction, comme présenté précédemment à la Figure 4.3.

Portable : en relation avec les propriétés de dimensions et de poids, cette propriété définit si le dispositif est portable par l'utilisateur ou non. Les valeurs sont soit "vrai" soit "faux".

Niveau d'expertise requis : afin de savoir si le dispositif peut être utilisé par un utilisateur, nous caractérisons le dispositif par un niveau d'expertise requis pour son utilisation. Cette propriété permet d'indiquer, lors de la conception de la difficulté pour un utilisateur d'utiliser le dispositif. Trois valeurs sont possibles et suivent les niveaux d'expertise identifiés par [Rasmussen 1986] : "expert", "intermédiaire" et "novice". La spécification de la valeur de cette propriété doit incorporer le temps d'apprentissage demandé pour utiliser de façon optimale le dispositif. Par exemple, le niveau d'expertise d'une souris est novice alors que le joystick demande un niveau d'expertise intermédiaire.

Mode de communication : identifiée par [Bellik 1995], cette propriété se rapporte aux organes utilisés par un utilisateur lorsqu'il communique avec le système informatique. Trois valeurs sont possibles : "gestuel" (mains, bras, tête, visage, etc.), "oral" (cordes vocales, langues, etc.) ou "mental" (cerveau). La majorité des usages de dispositifs se fait par le geste, comme avec la souris, le clavier ou encore un joystick. L'utilisation de dispositifs comme le microphone se caractérise par le mode oral. Enfin, le mode mental est disponible aujourd'hui sur des dispositifs qui analysent l'activité du cerveau [Krepi 2004].

Lieu d'interaction nominal : définit par [Dubois 2001], le lieu d'interaction est le lieu où l'utilisateur doit porter son attention pour pouvoir fournir des données au système. Cette caractéristique étant dynamique, le lieu d'interaction nominal renseigne l'endroit où l'utilisateur utilise le dispositif dans des conditions normales. Son domaine de valeur peut être défini par quatre zones selon sa distance par rapport à l'utilisateur [Trevisan 2004] : la "zone centrale" de 0 à 45 cm, la "zone personnelle" de 46 à 120 cm, la "zone sociale" de 121 à 360 cm et la "zone publique" au-delà de 360 cm. Par exemple, le lieu d'interaction nominal d'une souris est la zone centrale.

Propriétés lors de l'utilisation

Il s'agit ici des propriétés permettant de connaître l'état d'exécution du dispositif.

Etat de marche : cette propriété permet de connaître l'état courant de fonctionnement du dispositif. Deux valeurs sont possibles : il peut être en marche (valeur "vrai") ou à l'arrêt (valeur "faux").

Etat de panne : il est parfois possible de détecter un dysfonctionnement d'un dispositif. Le dispositif continue de fonctionner mais avec des erreurs potentielles. Deux valeurs sont possibles selon s'il y a panne (valeur "vrai") ou non (valeur "faux").

Facteur de confiance des données : chacune des données fournies par le dispositif peut être caractérisée par un facteur de confiance définissant la pertinence et la crédibilité de l'information générée. Le facteur de confiance est noté de 0 à 100, 0 représente le facteur le plus bas et 100 le facteur le plus élevé.

Cette liste de propriétés forme une base commune pour tous les dispositifs. Alors que les propriétés intrinsèques et relatives à l'utilisation humaine permettent d'orienter les choix pendant ces phases, les propriétés lors de l'utilisation doivent aider au traitement des données. Cependant, le modèle peut évoluer et intégrer des propriétés supplémentaires. De plus, des caractéristiques particulières à des sous-ensembles de dispositifs peuvent être identifiées. Par exemple, pour un appareil photo numérique, le niveau de zoom optique est un critère déterminant de choix et pourrait être ajouté.

2.3 Langages d'interaction

2.3.1 Définition

Un langage d'interaction définit un ensemble d'expressions bien formées et significatives pour le reste du système interactif. Il correspond au niveau logique d'une modalité d'interaction [Nigay 1996], répresentant ainsi un niveau d'abstraction des données plus élevé que celui fourni par le dispositif. Cette élévation du niveau d'abstraction correspond à une structure de données compréhensible par le reste du système informatique. Par exemple, les angles en degré, qui informent de la position de la tête d'un utilisateur, constituent un langage d'interaction sous la forme d'un triplet de trois valeurs comprises entre –180 et 180 degrés. Il est à noter que le terme "système représentationnel" est parfois préféré car il peut y avoir ambiguïté entre le langage d'interaction et son caractère linguistique [Bernsen 1994].

2.3.2 Propriétés

Comme pour les dispositifs, les propriétés qui caractérisent les langages d'interaction sont classées en trois catégories. Nous retrouvons les propriétés intrinsèques au langage d'interaction, celles relatives à l'utilisation humaine et celles utiles lors de l'utilisation.

Propriétés intrinsèques du langage d'interaction

Nom : cette propriété essentielle permet d'identifier le langage d'interaction sous la forme d'un texte.

Nombre de dimensions spatiales : identifié dans [Vernier 2001], le langage d'interaction est représentable par une ou plusieurs dimensions spatiales. Par exemple, la position d'un utilisateur en coordonnées WGS84 (World Geodetic System 1984 : système de référence utilisé par les GPS pour les positions au voisinage de la Terre) compte 3 dimensions. Le domaine de valeur est de 0 à l'infini. En comparant le nombre de dimensions du langage d'interaction avec celui du dispositif auquel il est couplé, l'adéquation d'une modalité d'interaction peut être définie (cf. section 2.1.1 de ce chapitre).

Caractère analogique : identifiée par [Bernsen 1994], cette propriété repose sur la notion de ressemblance avec la réalité. Deux valeurs sont possibles : "vrai" ou "faux". Le système représentationnel est analogique quand il repose sur une métaphore du monde réel. Par exemple, un langage d'interaction qui transmet les mots reconnus vocalement par une chaîne de caractères équivalente est analogique. A l'opposé, s'il transforme ses mots en un code chiffré, il est non-analogique. Cette propriété participe à la spécification du niveau métaphorique global de la modalité (cf. section 2.1.1 de ce chapitre). En effet, le caractère analogique du langage d'interaction contribue à définir la notion de nom et de verbe du niveau métaphorique (notion de forme et de mouvement), si les actions système qu'il transmet se rapprochent de la spécification de ces mêmes actions lors d'une spécification dans le monde réel. Il est alors possible de déterminer si cette notion de forme et de mouvement, en adéquation avec l'utilisation du dispositif, est prise en compte.

Caractère arbitraire : identifiée par [Bernsen 1994], cette propriété reflète la nature du langage d'interaction choisi, en fonction de ce que l'on souhaite représenter. Deux valeurs sont possibles "vrai" ou "faux". Le langage d'interaction est non arbitraire quand il n'y a pas de correspondance sociale ou cognitive entre le signifiant et le signifié. Un langage d'interaction non-arbitraire consiste à produire, par exemple, comme pour le GPS, un système de coordonnées pour la localisation sur Terre se basant sur la latitude, la longitude et la hauteur par rapport à la surface de la Terre. Au contraire, l'utilisation des touches clavier dans les jeux vidéo est arbitraire, comme lorsque l'on presse la barre d'espace pour faire freiner une voiture.

Caractère linguistique : identifiée par [Bernsen 1994], cette propriété définit si le

langage d'interaction se rapporte à des entités distinctives de sens (phonèmes) permettant de définir des unités signifiantes (morphèmes et lexèmes). En d'autres mots, il s'agit de déterminer si le langage d'interaction est défini par une phonologie, une morphologie et une syntaxe. Les valeurs possibles sont "vrai" ou "faux". Par exemple, toutes les langues naturelles ou pseudo-naturelles portent le caractère linguistique. Comme exemple contraire, un système de coordonnées en trois dimensions n'a pas le caractère linguistique.

Dimension temporelle : identifiée par [Bernsen 1994], cette propriété indique si le langage d'interaction ne peut s'exprimer que par l'écoulement du temps. Deux valeurs sont possibles : "statique" ou "dynamique". Par exemple, la dimension temporelle de coordonnées de position 3D est statique. Au contraire, la dimension temporelle de la reconnaissance de geste est dynamique.

Domaine des valeurs d'entrée : comme pour l'espace de [Card 1990], le domaine des valeurs d'entrée définit la nature des données acceptées et traitables par le langage d'interaction. Son domaine de valeur dépend du langage d'interaction. Par exemple, un langage d'interaction qui transforme les signaux de déplacement x et y d'une souris prend en compte deux entiers. Cette propriété est à mettre en correspondance avec le domaine des valeurs de sortie des dispositifs, afin de savoir si le langage d'interaction peut être couplé avec un dispositif particulier.

Domaine des valeurs de sortie : comme pour l'espace de [Card 1990], le domaine des valeurs de sortie définit la nature des données transmises par le langage d'interaction. Son domaine de valeur dépend du langage d'interaction.

Temps de traitement moyen : le temps de traitement moyen est une estampille de temps en millisecondes qui représente le temps moyen que met le langage d'interaction pour traiter une donnée en provenance d'un dispositif, ou d'un autre langage d'interaction.

Fréquence des données : cette propriété renseigne sur la fréquence de transmission des données par le langage d'interaction. Deux valeurs sont possibles : "continue" ou "discrète". Par exemple, la manipulation directe dans les interfaces WIMP transmet uniquement des données lorsqu'un utilisateur réalise des actions (valeur discrète).

Propriétés relatives à l'utilisation humaine

Niveau d'expertise requis : afin de savoir si le langage d'interaction peut être compréhensible et utilisable par un utilisateur, nous le caractérisons par

un niveau d'expertise requis pour son utilisation. Il définit la complexité du langage d'interaction. Les trois valeurs sont définies et suivent les niveaux d'expertise identifiés par [Rasmussen 1986] : "expert", "intermédiaire" et "novice". Par exemple, le niveau d'expertise pour la manipulation directe est novice. La spécification de la valeur de cette propriété doit incorporer le temps d'apprentissage demandé pour utiliser de façon optimale le langage d'interaction.

Propriétés lors de l'utilisation

Il s'agit ici des propriétés permettant de connaître l'état d'exécution du langage d'interaction.

Etat de marche : cette propriété permet de connaître l'état courant de fonctionnement du langage d'interaction. Deux valeurs sont possibles, il peut être en marche (valeur "vrai") ou à l'arrêt (valeur "faux").

Etat de panne : il est parfois possible de détecter un dysfonctionnement du langage d'interaction. Le langage d'interaction continue de fonctionner mais avec des erreurs potentielles. Deux valeurs sont possibles, selon s'il y a panne (valeur "vrai") ou non (valeur "faux").

Facteur de confiance des données : Chacune des données fournies par le langage d'interaction peut être caractérisée par un facteur de confiance définissant la pertinence et la crédibilité de l'information générée. Le facteur de confiance est noté de 0 à 100, 0 représente le facteur le plus bas et 100 le facteur le plus élevé.

Cette liste de propriétés forme une base commune pour les langages d'interaction. Cependant, le modèle peut évoluer et intégrer des propriétés supplémentaires.

2.4 Recommandations pour la mise en œuvre d'une modalité d'interaction

Cette section développe des recommandations basées sur les propriétés présentées dans les sections précédentes, pour définir des modalités d'interaction efficaces au sein d'une interaction multimodale. Elles définissent un guide pour choisir des modalités d'interaction pertinentes, en fonction de l'utilisateur, des tâches à réaliser et du contexte.

2.4.1 Dispositif adapté à l'utilisateur

Afin de s'assurer que le dispositif est adapté à l'utilisateur cible, il convient de comparer ces propriétés aux capacités physiques et cognitives de l'utilisateur.

Les propriétés de portabilité (en relation avec les dimensions et poids) et celle du mode de communication doivent être confrontées avec les facteurs biologiques de l'utilisateur. Cette confrontation permet de détecter d'éventuelles incohérences, comme par exemple la présence d'un handicap moteur de l'utilisateur, qui amènerait une incapacité de l'utilisateur à utiliser des dispositifs avec un mode de communication gestuel. D'autre part, le lieu d'interaction nominal doit aussi correspondre aux particularités de l'utilisateur. Les dispositifs de zone centrale (proche de l'utilisateur) sont par exemple à favoriser pour les utilisateurs à mobilité réduite.

Enfin, le niveau d'expertise requis du dispositif est en relation directe avec les facteurs socioculturels humains. Ainsi, le niveau de formation et l'expérience de l'utilisateur définissant si l'utilisateur a les capacités cognitives pour l'utilisation optimale du dispositif.

2.4.2 Dispositif adapté à la tâche et au contexte

Pour que l'utilisation du dispositif soit efficace, il doit aussi être adapté à la tâche et à son contexte. Outre la fonction qu'il peut réaliser, l'autonomie, la précision, la résolution, la stabilité, l'utilisabilité à distance et le temps de traitement moyen du dispositif forment l'ensemble des caractéristiques qui dénotent de sa qualité fonctionnelle sur les données qu'il transmet. Par exemple, si la tâche à réaliser se situe dans un contexte mobile, une autonomie importante est à favoriser.

2.4.3 Langage d'interaction adapté à l'utilisateur

Comme pour les dispositifs, le niveau d'expertise demandé pour que l'utilisateur comprenne pleinement le langage d'interaction doit être mis en relation avec ses facteurs socioculturels. A ce titre, les langages d'interaction analogiques et non-arbitraires doivent être favorisés, puisqu'ils garantissent des repères à des éléments connus de l'utilisateur dans le monde réel, ainsi qu'un apprentissage plus facile. Ce type de langage d'interaction contribue à l'adéquation générale de la modalité d'interaction abordée dans la section suivante.

2.4.4 Haut niveau d'adéquation entre le dispositif, le langage d'interaction et la tâche

Comme nous l'avons défini à la section 2.1.1 de ce chapitre, l'adéquation entre le dispositif, le langage d'interaction et la tâche se détermine par le niveau métaphorique et le degré d'intégration. Comme le montre la Figure 4.4, une

modalité d'interaction cohérente et adaptée a un niveau métaphorique élevé et un degré d'intégration égal à 1. Ainsi, il faut favoriser les modalités d'interaction qui s'approchent le plus de ce niveau d'adéquation, en privilégiant le niveau métaphorique avant le degré d'intégration.

2.4.5 Haut degré d'indirection

Le degré d'indirection, défini à la section 2.1.2, doit aussi être le meilleur possible. Il se détermine par un niveau d'incarnation et un indice temporel. Comme le montre la Figure 4.5, une modalité d'interaction qui présente un degré d'indirection idéal revêt un niveau d'incarnation totale et un indice temporel égal à 0, correspondant au temps-réel. Les modalités d'interaction qui s'approchent le plus de ce degré d'indirection sont à favoriser. L'indice temporel peut cependant être plus ou moins important selon le type d'application finale. Par exemple, lorsque le temps-réel n'est pas nécessaire, l'indice temporel peut éventuellement être élevé.

3 Combinaisons de modalités

Alors que nous venons de poser les bases de notre espace conceptuel pour les modalités d'interaction, l'enjeu de la multimodalité se joue aussi sur les combinaisons des modalités d'interaction. Cette section caractérise les différentes formes de combinaison.

3.1 Définition

La combinaison représente l'union organisée de plusieurs modalités d'interaction pour la réalisation d'une tâche par un utilisateur. La combinaison s'oppose à l'utilisation indépendante de modalités d'interaction pour une tâche donnée qui constitue les cas des usages exclusif et concurrent des modalités d'interaction dans la classification UOM de [Nigay 1994] présentée au chapitre 3.

Comme l'explicite la Figure 4.3, les combinaisons peuvent s'effectuer pour plusieurs modalités d'interaction et à plusieurs niveaux (par exemple, une combinaison peut exister entre une modalité et une combinaison de modalités déjà existante). Les sections suivantes proposent un ensemble de propriétés pour caractériser les diverses combinaisons possibles.

3.2 Propriétés

Deux catégories de propriétés sont observables en fonction de leur utilité. Alors que les propriétés intrinsèques définissent l'usage combiné, les propriétés lors de l'utilisation définissent le comportement de l'implémentation de la combinaison.

Propriétés intrinsèques de la combinaison

Nature de la combinaison : cette propriété constitue la caractéristique essentielle d'une combinaison. Basée sur les espaces de conception de [Coutaz 1994] et [Martin 2002], une combinaison peut être de quatre natures différentes : complémentaire, équivalente, redondante ou redondante/équivalente. La **complémentarité** entre plusieurs modalités d'interaction signifie que l'utilisation de toutes ces modalités est nécessaire pour la réalisation de la tâche. Les données des modalités d'interaction complémentaires doivent être ajoutées les unes aux autres pour obtenir une commande complète et interprétable. L'**équivalence** de plusieurs modalités d'interaction signifie que toutes ces modalités d'interaction ont la même fonction et qu'elles permettent de réaliser les mêmes tâches. L'équivalence de modalités d'interaction laisse un choix à l'utilisateur pour réaliser ses actions. La **redondance** de modalités d'interaction implique que ces modalités d'interaction soient équivalentes

fonctionnellement et signifie que l'utilisateur spécifie plusieurs fois la tâche à réaliser selon plusieurs modalités d'interaction. L'utilisateur est donc forcé d'utiliser toutes les modalités d'interaction redondantes. La **redondance/ équivalence** correspond à un compromis entre "redondance" et "équivalence". Elle est utilisée pour laisser à l'utilisateur la possibilité d'utiliser n'importe quelles modalités d'interaction indépendamment, tout en traitant l'usage redondant. Pour chacune des natures, un mécanisme spécifique pour fusionner les données provenant des différentes modalités d'interaction doit être défini et développé.

Temporalité : identifiée dans la classification UOM de [Nigay 1994], cette propriété détermine l'usage temporel par l'utilisateur des modalités combinées. Quatre valeurs sont possibles : "aucune", "parallèle", "séquentielle" ou "parallèle et séquentielle". L'utilisation combinée de plusieurs modalités parallèlement est appelé usage synergique, alors que leur utilisation séquentielle est qualifié d'usage alterné. La valeur "aucune" est utilisée pour les combinaisons à nature d'équivalence. Les valeurs de cette propriété peuvent être affinées par les différents schémas de composition de [Allen 1983], réutilisés dans l'extension de CARE proposée par [Vernier 2001] et présentée au chapitre 3 de ce mémoire.

Ordre : cette propriété représente le besoin d'une succession à caractère temporel de l'usage des modalités d'interaction. La spécification d'un ordre définit la chronologique qui doit être effectuée par l'utilisateur pour l'emploi des modalités d'interaction combinées. L'ordre est indéfini pour une nature d'équivalence.

Propriétés lors de l'utilisation

Fenêtre temporelle : cette propriété définit en millisecondes l'intervalle de temps pendant lequel les informations des modalités d'interaction peuvent être combinées. La fenêtre temporelle constitue la caractéristique principale du mécanisme de fusion des données provenant des différentes modalités d'interaction à combiner. Une fenêtre temporelle peut être infinie. Pour l'équivalence, la fenêtre temporelle est indéfinie. Avec la propriété de temporalité définie précédemment, la fenêtre temporelle permet de spécifier précisément les possibilité de combinaison selon les différents schémas de composition temporels [Allen 1983] [Vernier 2001] présentés au chapitre 3 de ce mémoire.

Stratégie : identifiée dans [IHM 1992], cette propriété définit deux stratégies concernant la fusion d'informations : la stratégie "précoce" ou la stratégie "différée". Le choix de la stratégie permet d'accélérer le traitement de l'interaction (stratégie précoce) ou d'assurer la pertinence des informations avant de les propager (stratégie différée). La stratégie est indéfinie pour l'équivalence de plusieurs modalités d'interaction.

Domaine des valeurs d'entrée : le domaine des valeurs d'entrée définit la nature des données acceptées et traitables par la combinaison. Son domaine de valeur dépend des modalités d'interaction qu'elle doit combiner.

Domaine des valeurs de sortie : le domaine des valeurs de sortie définit la nature des données transmises par la combinaison des modalités d'interaction.

Temps de traitement moyen : le temps de traitement moyen est une estampille de temps en millisecondes qui représente le temps moyen que met la fusion des données multimodales selon la nature de la combinaison.

Fréquence des données : cette propriété renseigne sur la fréquence de transmission des données après la fusion multimodale. Deux valeurs sont possibles : "continue" ou "discrète".

Etat de marche : cette propriété permet de connaître l'état courant de fonctionnement de la combinaison. Deux valeurs sont possibles, elle peut être en marche (valeur "vrai") ou à l'arrêt (valeur "faux").

Etat de panne : il est parfois possible de détecter un dysfonctionnement de la combinaison. La combinaison continue de fonctionner mais avec des erreurs potentielles. Deux valeurs sont possibles, selon s'il y a panne (valeur "vrai") ou non (valeur "faux").

Facteur de confiance des données : Chacune des données fournies suite à la combinaison peut être caractérisée par un facteur de confiance définissant la pertinence et la crédibilité de l'information générée . Le facteur de confiance est noté de 0 à 100, 0 représente le facteur le plus bas et 100 le facteur le plus élevé.

3.3 Recommandations pour la mise en œuvre des combinaisons

Comme pour les modalités d'interaction, cette section liste des recommandations concernant la combinaison de modalités d'interaction. Ces recommandations

constituent les prémisses d'un guide de conception pour la mise en place d'une interaction multimodale utile et utilisable.

3.3.1 Privilèges des modalités passives

L'utilisation de modalités d'interaction passives minimise l'effort de la part de l'utilisateur. En effet, aucune action explicite n'est nécessaire pour que la modalité d'interaction transmette des données au reste du système informatique. Leur combinaison avec d'autres modalités d'interaction a l'avantage de ne pas entamer les ressources physiques et cognitives dont dispose l'utilisateur. L'emploi de modalités d'interaction passives combinées à des modalités actives est donc à favoriser lorsque cela est possible.

3.3.2 Spécification de la fenêtre temporelle

La détection du meilleur moment pour fusionner des données provenant de plusieurs modalités d'interaction combinées représente l'aspect essentiel du mécanisme de fusion multimodale. La fenêtre temporelle qui autorise les combinaisons doit donc être complètement adaptée aux temps de traitement moyens des dispositifs et des langages d'interaction qui constituent une modalité. Ainsi, la fenêtre temporelle devrait être, au minimum, égal au temps de traitement moyen de la modalité d'interaction la plus lente parmi celles qui produisent des données en continu, correspondant au cas où les données sont envoyées à des fréquences régulières.

3.3.3 Contraintes sur les lieux d'interaction

Lors d'une combinaison qui implique une complémentarité ou une redondance, le concepteur de l'interaction doit accorder une importance particulière aux lieux d'interaction nominaux des dispositifs impliqués. En effet, il est possible que l'utilisation de dispositifs éloignés nuise à l'utilisation simultanée de ces divers dispositifs. Il est ainsi recommandé de combiner des dispositifs qui se trouvent dans la zone centrale au plus près de l'utilisateur ou dans des zones compatibles.

3.3.4 Contraintes sur les modes de communication

Le mode de communication de chaque modalité d'interaction à combiner doit aussi être étudié attentivement, car leur utilisation peut se révéler impossible. En effet, pour des usages de nature complémentaire ou redondante avec une temporalité parallèle, les modalités d'interaction actives présentant le même mode

de communication sont inutilisables, à quelques exceptions près concernant le mode gestuel, qui peut correspondre à l'utilisation de membres différents pour manipuler les dispositifs.

3.3.5 Contraintes sur les niveaux d'expertise requis

Les niveaux d'expertise requis de chaque modalité d'interaction à combiner doivent être adaptés. Par exemple, l'utilisation de plusieurs modalités d'interaction demandant un niveau élevé expert peut entraîner une surcharge cognitive pour l'utilisateur, influençant ainsi l'efficacité globale de la combinaison des modalités d'interaction. Il est néanmoins difficile de définir des recommandations précises et il semble que seules des expérimentations peuvent permettre d'évaluer si l'adéquation des niveaux d'expertise de chaque modalité d'interaction à combiner est appropriée.

4 Illustration du modèle conceptuel sur un exemple

Cette section présente une illustration du modèle conceptuel, présenté précédemment, par un exemple concret de deux modalités combinées.

4.1 Description du cas d'étude

Ce cas d'étude a pour objectif d'illustrer notre modèle conceptuel sur un exemple concret de combinaisons entre deux modalités d'interaction. L'application concerne un utilisateur mobile qui visite la ville de Paris et qui peut effectuer un certain nombre d'actions par rapport à l'endroit où il se situe. La tâche qui va nous servir d'illustration est la suivante : l'utilisateur peut sauvegarder informatiquement la position des lieux qui présentent un intérêt.

La position des points d'intérêts doit être sauvegardée sous la forme de trois coordonnées, suivant le système géodésique WGS84 (système de référence permettant d'exprimer les positions au voisinage de la Terre). Pour cela, l'utilisateur dispose d'un GPS Tomtom® et d'un système de reconnaissance vocale ViaVoice® d'IBM, lui permettant, lorsqu'il prononce la commande "sauvegarder le lieu" d'enregistrer les coordonnées fournies par le GPS. Il s'agit donc ici d'utiliser de façon complémentaire les deux modalités d'interaction, afin de savoir pour quel endroit l'utilisateur réalise sa commande.

Nous caractérisons les deux modalités d'interaction et leurs combinaisons avec les propriétés de notre espace conceptuel.

4.2 Modalité GPS en coordonnées WGS84

La première modalité mise en place concerne le GPS Tomtom® qui fournit des coordonnées en trois dimensions selon la norme WGS 1984. Il s'agit d'une modalité d'interaction passive qui fournit les coordinnées de façon continue, sans intervention de l'utilisateur. La Figure 4.6 montre le diagramme d'objets de cette modalité selon le standard UML [UML 2006] et précise chacune des caractéristiques des objets du modèle. Il est remarquable que cette modalité représente une adéquation interne et externe optimale, avec un niveau métaphorique total (l'utilisateur se déplace normalement dans la ville) et un degré d'intégration égal à 1 (dimension du dispositif = 3D / dimension du langage d'interaction = 3D). Le degré d'indirection de cette modalité est convenable avec un niveau d'incarnation rapproché (l'utilisateur visualise la position calculée par le GPS sur le l'écran du Tomtom) et un indice temporel égal à 260 millisecondes (il s'agit du temps de traitement moyen par l'addition du temps de traitement moyen des données par

le dispositif –20 ms– avec celui du langage d'interaction –40ms– et le temps de traitement de cette action par le noyau fonctionnel et son retour perceptuel défini arbitrairement à 200ms). Ce degré d'indirection pourrait être amélioré en utilisant des GPS plus performants, mais plus coûteux.

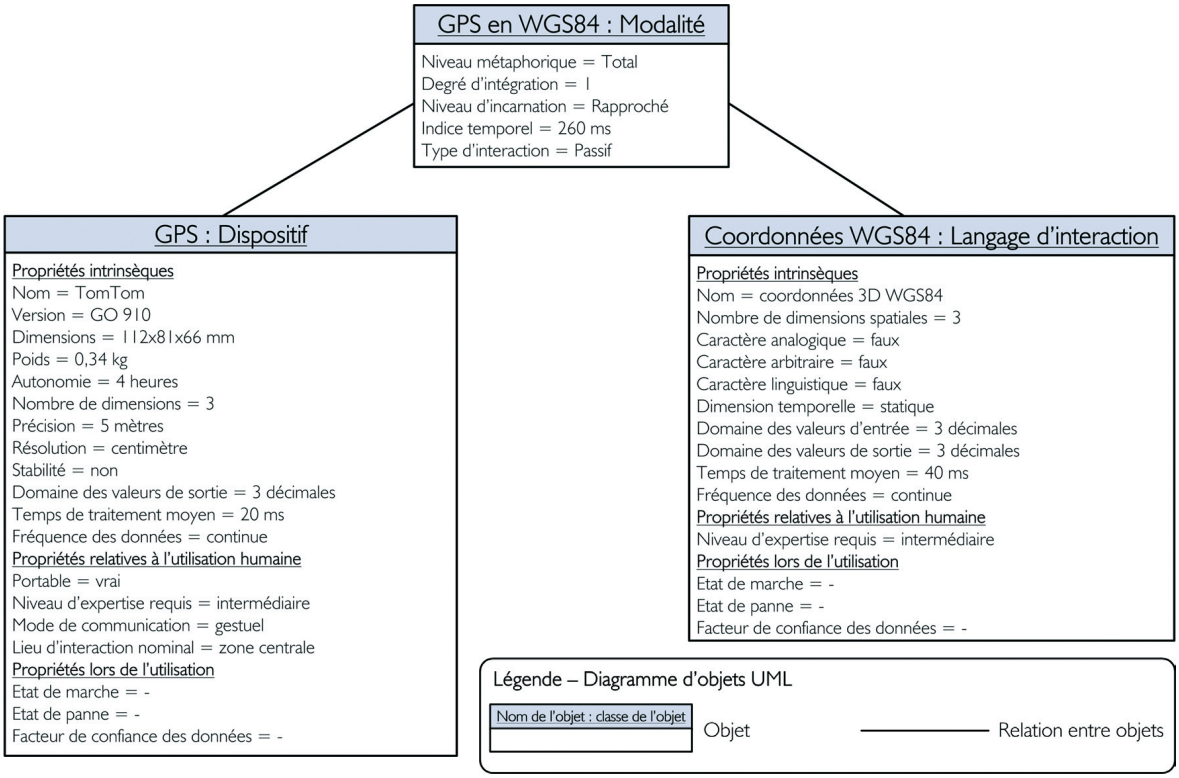


Figure 4.6: diagramme d'objets UML de la modalité GPS en coordonnées WGS84.

4.3 Modalité de reconnaissance vocale

La deuxième modalité de ce cas d'étude concerne la reconnaissance vocale d'un langage pseudo-naturel basé sur le Français. La Figure 4.7 montre le diagramme d'objets de cette modalité selon le standard UML [UML 2006] et précise chacune des caractéristiques des objets du modèle. Au contraire de la première, cette modalité est active car l'utilisateur doit explicitement prononcer des mots du vocabulaire pour spécifier des commandes. L'adéquation de cette modalité d'interaction est bonne en présentant un niveau métaphorique total (l'utilisateur a seulement besoin de parler à haute voix comme lorsqu'elle/il communique avec une autre personne) et un degré d'intégration égal à 1 (dimension du dispositif = 1D / dimension du langage d'interaction =1D). Le niveau d'indirection est très

bon car le niveau d'incarnation est total (le langage d'interaction retransmet les mots tels qu'ils ont été prononcés par l'utilisateur et nous considérons que le retour d'information affiche les mots reconnus) et l'indice temporel est égal à 300 millisecondes (il s'agit du temps de traitement moyen par l'addition du temps de traitement moyen des données par le dispositif –60 ms– avec celui du langage d'interaction –40ms– et avec le temps de traitement de cette action par le noyau fonctionnel et son retour perceptuel défini arbitrairement à 200ms). Le langage d'interaction propage ici seulement les mots de vocabulaire qui sont utiles au système informatique, ainsi la fréquence des données est discrète.

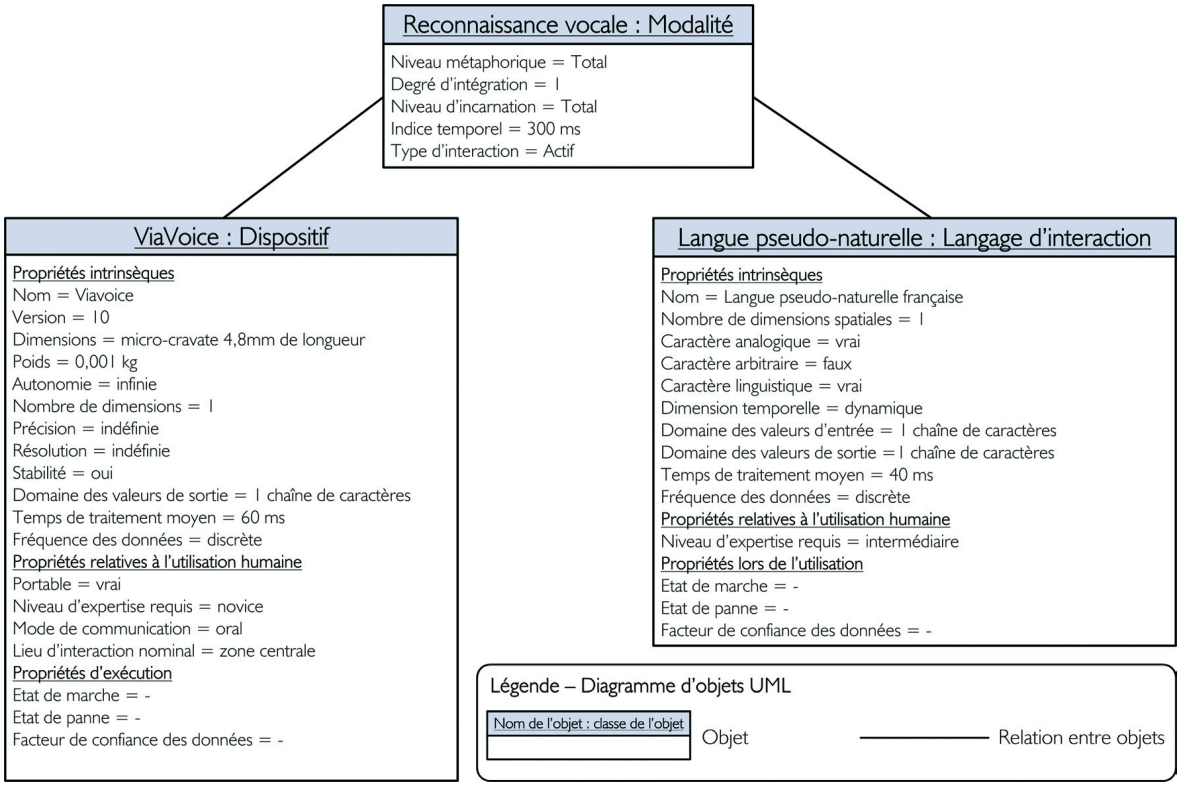


Figure 4.7: diagramme d'objets UML de la modalité reconnaissance vocale.

4.4 Combinaison complémentaire des deux modalités

Les deux modalités d'interaction étant définies, nous modélisons leur combinaison. En effet, la commande prononcée par l'utilisateur et reconnue par la modalité de reconnaissance vocale est à associer à la position exacte de l'utilisateur fournie par la seconde modalité d'interaction. Ces données, issues des deux modalités d'interaction, sont donc fusionnées et fournissent une commande complète et interprétable par le reste du système informatique.

La Figure 4.8 montre la caractérisation de la combinaison. Comme la totalité des données provenant des modalités d'interaction doivent être exploitées et qu'elles ne sont pas équivalentes en termes de fonctionnalités, la nature de la combinaison est la complémentarité. La temporalité ne revêt que peu d'importance, car la modalité d'interaction passive est continue et la fusion peut se réaliser de manière parallèle ou séquentielle. Cependant, pour une fusion séquentielle, l'écart entre les émissions des données multimodales ne doit excéder un laps de temps trop long. Ainsi, la fenêtre temporelle a été fixée à 60 millisecondes (correspondant à l'addition des temps de traitement moyens des données par le GPS et le langage d'interaction associé, fournissant des données en continu). L'ordre d'émission des données provenant des deux modalités d'interaction n'est pas déterminant, sa valeur est fixée à faux. Enfin la stratégie est précoce : dès que les données sont émises dans la même fenêtre temporelle, les données fusionnées sont envoyées au reste du système informatique.

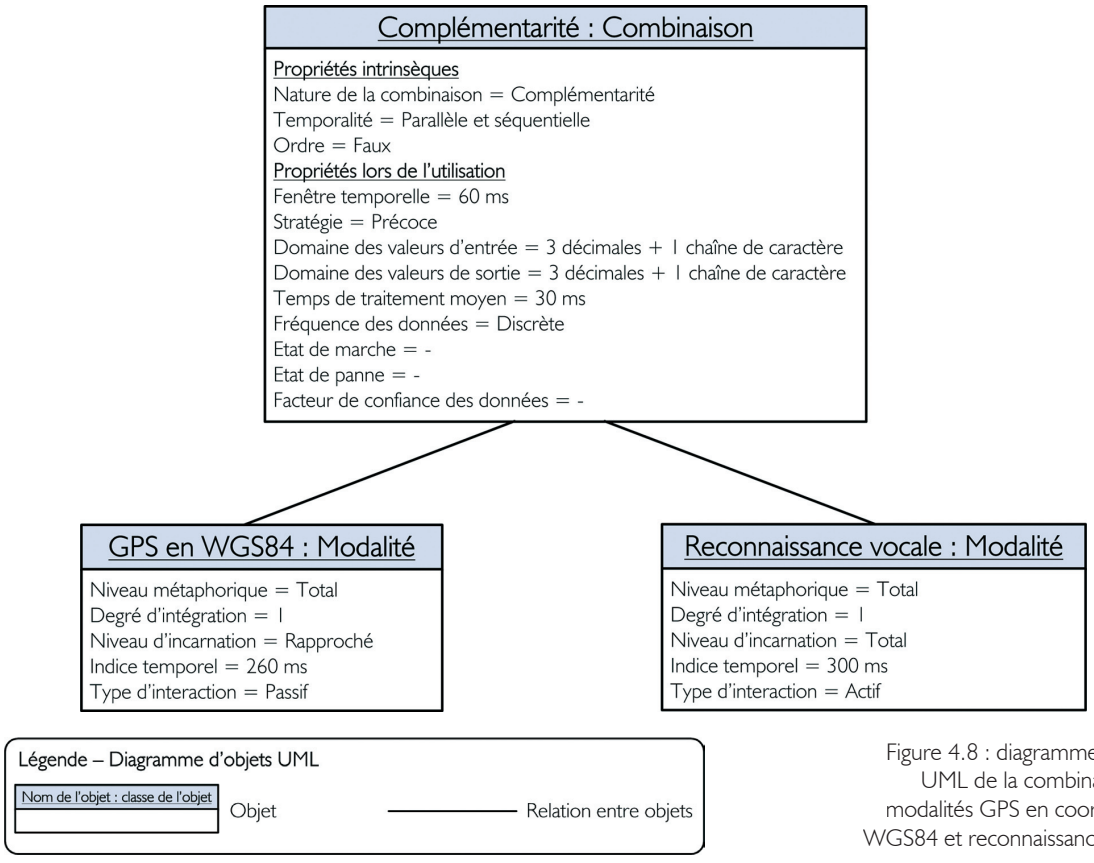


Figure 4.8 : diagramme d'objets UML de la combinaison des modalités GPS en coordonnées WGS84 et reconnaissance vocale.

5 Résumé du chapitre 4

Ce chapitre propose un modèle conceptuel de l'interaction multimodale, premier résultat vers une théorie de la multimodalité qui permet de raisonner sur les concepts clefs de ce type d'interaction. Il présente, par la mise en relation des études sur la multimodalité, un méta-modèle simple (diagramme de classes UML présenté à la Figure 4.3) pour spécifier plusieurs types d'interaction multimodale. Alors que le méta-modèle général organise toutes les entités actrices lors d'une

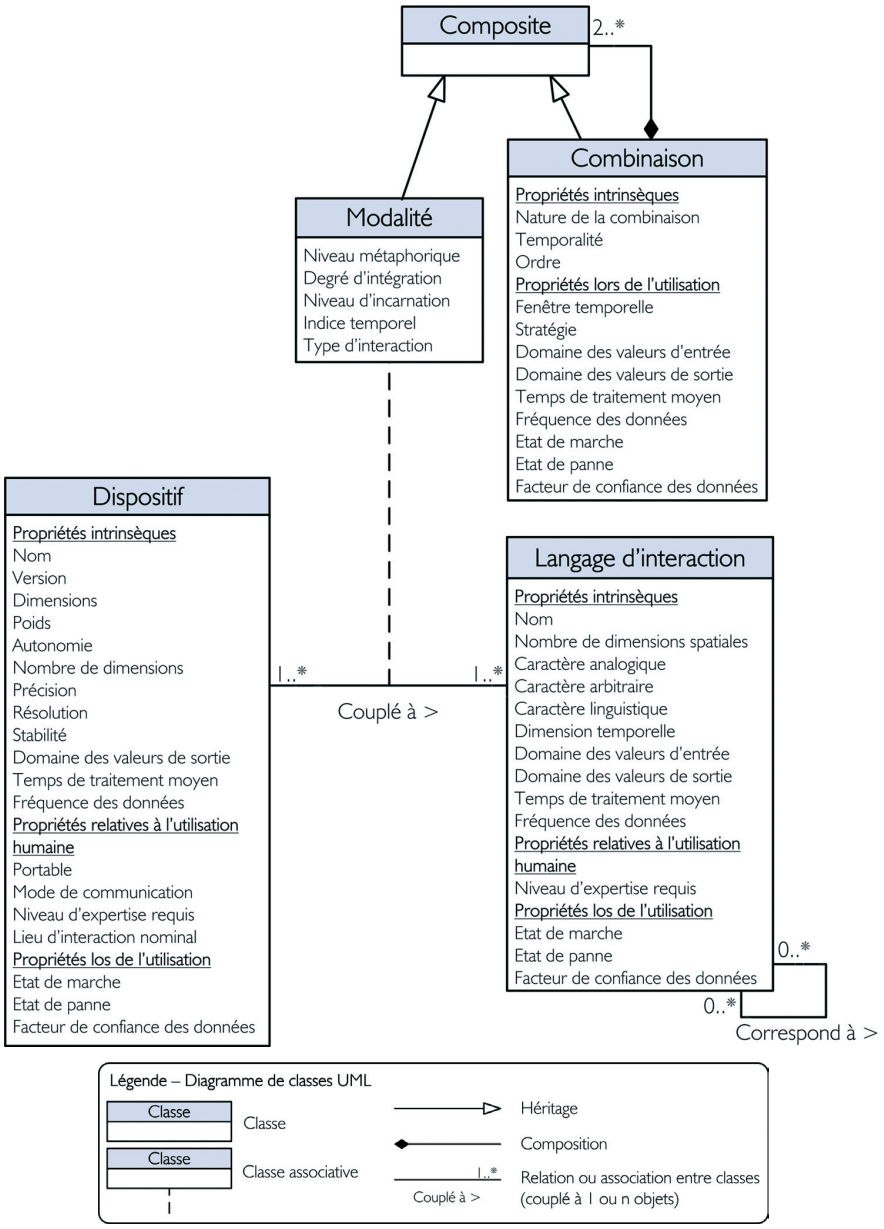


Figure 4.9 : diagramme de classes UML des possibilités multimodales.

interaction multimodale, les possibilités d'interaction incarnées par les modalités d'interaction et leurs combinaisons possibles sont approfondies par la définition d'un ensemble de propriétés propres à chaque classe. La Figure 4.9 présente le diagramme des classes UML concernant les possibilités d'interaction en y ajoutant les propriétés définies pour chacune des classes. Enfin, un exemple concret de combinaison entre deux modalités d'interaction illustrant notre modèle conceptuel est présenté par des diagrammes d'objets UML (Figure 4.6, Figure 4.7 et Figure 4.8).

Ce modèle conceptuel permet de définir l'ensemble des possibilités d'interaction multimodale. En effet, si nous reprenons la classification UOM de [Nigay 1994] des interfaces multimodales, notre méta-modèle couvre toutes les configurations possibles, aussi bien au niveau de la multiplicité des dispositifs et des langages d'interaction, de leurs optionalités ou encore de leurs usages (exclusif, concurrent, alternée ou synergique). La caractérisation précise des dispositifs, des langages d'interaction et de leurs combinaisons possibles offre de véritables critères de choix pour les concepteurs d'interaction multimodale.

En résumé, le modèle conceptuel avancé permet de spécifier précisément plusieurs types d'interaction multimodale. Le manque d'outil de conception et de développement pour les systèmes multimodaux nous a incité à opérationnaliser notre modèle au sein d'un outil informatique de conception et de développement. Cet outil, utilisable aussi bien pour le prototypage rapide que pour la réalisation de l'application finale, doit permettre d'accélérer le processus de création de systèmes multimodaux, de capitaliser les expériences produites, et permettre ainsi de définir de nombreuses règles de conception. L'outil à concevoir devient alors un moyen pour enrichir notre modèle, dans l'objectif de définir une théorie de la multimodalité. La partie suivante de ce mémoire est dédiée à l'analyse des outils existants et à la définition d'un nouvel outil basé sur notre modèle conceptuel.

PARTIE 2

OPERATIONNALISATION DE L'ESPACE CONCEPTUEL : L'OUTIL ICARE

Partie 2

OPERATIONNALISATION DE L'ESPACE CONCEPTUEL : L'OUTIL ICARE

Cette deuxième partie est consacrée à l'opérationnalisation de notre modèle conceptuel. Nous présentons un outil à base de composants logiciels appelé ICARE (Interaction – Complementarité, Assignation, Redondance et Equivalence) [Bouchet 2004b] en référence aux propriétés de CARE de [Coutaz 1994] (cf. section 1.2 du chapitre 3).

Afin d'obtenir la pleine mesure de notre outil, nous commençons, dans le chapitre 5 de cette partie, par décrire et comparer plusieurs outils existants. Onze outils sont présentés et comparés selon une grille d'analyse. Nous soulignons la variété des objectifs des outils existants. De cette comparaison, nous retenons que les outils existants sont soit trop généraux, soit trop spécifiques à un problème particulier de la multimodalité.

P. 127

De ce constat, l'outil ICARE constitue une solution innovante basée sur une approche à composants logiciels. Par assemblage de composants, ICARE permet de définir les modalités d'interaction et leurs usages. Il intègre explicitement et pour la première fois les différents types de combinaison (propriétés CARE [Coutaz 1994]) sous la forme de composants génériques, aux modalités d'interaction et aux domaines d'application. L'outil ICARE est présenté dans les trois derniers chapitres de cette partie. Le chapitre 6 pose les bases de notre modèle de composants pour concevoir et développer une interaction multimodale adaptée. Il décrit précisément les différents types de composants ICARE. Le chapitre 7 expose une solution d'implémentation de notre modèle à composants avec la technologie JavaBeans. Nous expliquons les raisons de ce choix et nous décrivons, ensuite, un éditeur graphique que nous avons conçu et développé pour permettre la conception et le développement d'interaction multimodale par assemblage graphique de composants. Enfin, le chapitre 8 situe la solution conceptuelle et implémentationnelle de l'outil ICARE au sein de notre grille d'analyse définie au chapitre 5, nous permettant ainsi de le comparer aux autres outils étudiés.

P. 185

P. 159

P. 199

Sommaire >> >> >>

PARTIE 2
OPERATIONNALISATION DE L'ESPACE CONCEPTUEL : L'OUTIL ICARE

CHAPITRE 5 127
OUTILS DE CONCEPTION LOGICIELLE ET DE DÉVELOPPEMENT

1 Outils et cycle de vie du logiciel 128
2 Grille d'analyse 130
3 Outils existants 133
4 Résumé du chapitre 5 155

CHAPITRE 6 159
OUTIL ICARE, LES PRINCIPES DE L'APPROCHE À COMPOSANTS

1 Technologies à composants 160
2 Spécifications des composants de l'outil ICARE 165
3 Communication entre composants 181
4 Utilisation dans un système informatique et architecture logicielle 182
5 Résumé du chapitre 6 184

CHAPITRE 7 185
OUTIL ICARE, UN EXEMPLE D'IMPLÉMENTATION

1 Les technologies à composants existantes 186
2 JavaBeans 189
3 Les composants de l'outil ICARE 191
4 Editeur graphique d'ICARE 193
5 Résumé du chapitre 7 198

CHAPITRE 8 199
OUTIL ICARE : SYNTHÈSE

1 Carte d'identité 200
2 Critères pour l'IHM 201
3 Critères généraux pour la multimodalité 202
4 Critères pour la fusion des données 203
5 Grille d'analyse 204
6 Résumé du chapitre 8 205

CHAPITRE 5
Outils de conception logicielle et de développement

Les chapitres de la Partie I ont été consacrés à l'établissement d'un espace conceptuel de la multimodalité. Nous abordons maintenant les différentes possibilités d'opérationnalisation de cet espace conceptuel. Nous constatons que la multimodalité est dotée de trop peu d'outils de conception logicielle et de développement. Par le terme outil, nous entendons les solutions conceptuelles et implémentationnelles qui aident les acteurs à effectuer les étapes du cycle de développement logiciel. Ces outils peuvent être éventuellement accompagnés d'un éditeur graphique, mais cette caractéristique n'est pas considérée comme un requis. Ce chapitre présente et compare des outils de conception logicielle et de développement existants selon une grille d'analyse.

Le chapitre est constitué comme suit : avant de présenter les outils existants pour la multimodalité, nous revenons sur leurs positionnements dans un cycle de développement d'un système interactif. Nous établissons un ensemble de critères qui définit notre grille d'analyse, pour ensuite étudier plusieurs outils existants.

La complexité de la conception et du développement des systèmes multimodaux, due à la variété des modalités d'interaction existantes et aux multiples possibilités de combinaison, nécessite de faciliter les phases de conception et de développement avec des outils adéquats. Pour rappel, le développement d'un logiciel passe par une succession d'étapes dont les résultats sont méthodiquement vérifiés avant de passer à l'étape suivante :

1. Analyse des besoins
2. Spécification
3. Conception globale (architecturale) et détaillée
4. Codage
5. Tests
6. Intégration
7. Recette et validation
8. Livraison
9. Maintenance du système

Plusieurs modèles de cycle de vie existent et ordonnent ces étapes. Deux grandes classes de cycle existent : les cycles séquentiels ou itératifs. Les cycles séquentiels comme le modèle en ligne, en cascade ou en V sont adaptés à de longs projets où les besoins sont très précis et maîtrisés. Les cycles itératifs comme le modèle incrémental, en spirale ou avec prototype proposent un développement plus souple. Ils permettent de maîtriser les risques d'un produit final exploratoire avec des besoins mal cernés. Plusieurs versions du produit peuvent être fournies,

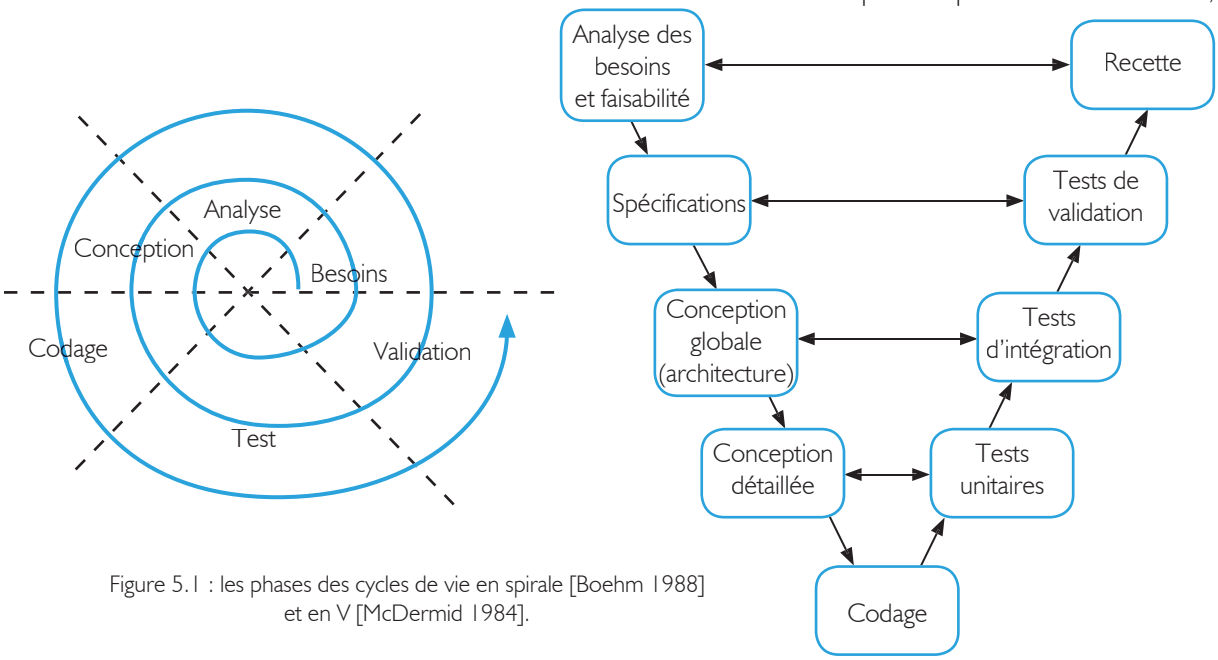


Figure 5.1 : les phases des cycles de vie en spirale [Boehm 1988] et en V [McDermid 1984].

satisfaisant les démarches commerciales classiques des produits grand public ou encore l'urgence d'être présent sur le marché. A titre d'exemple, la Figure 5.1 illustre l'enchainement des étapes des cycles de vie en spirale [Boehm 1988] et en V [McDermid 1984]. Le cycle en V demeure actuellement le cycle de vie le plus connu et certainement le plus utilisé, en raison de sa mise en correspondance des phases descendantes (conception et développement) et des phases montantes de tests, afin de viser une adéquation entre le produit demandé et celui livré. Néanmoins, les cycles de vie itératifs, comme celui en spirale, sont plus adaptés à l'Interaction Homme-Machine car l'utilisateur peut être le centre de chaque itération, offrant un produit adapté à ses besoins. Néanmoins, l'objectif de cette section n'est pas de comparer les différents cycles de vie, mais d'en extraire les étapes principales afin de définir un cadre fédérateur pour situer ensuite nos outils.

Idéalement, chaque phase du cycle de développement devrait être instrumentée par un seul outil ou plusieurs qui coopèrent. De plus, et comme il est souligné dans [Nigay 2006], les fonctionnalités principales d'un outil traitant les différentes phases seraient :

- de spécifier les modules logiciels à mettre en œuvre ;
- d'organiser les connexions entre ces modules ;
- de raisonner sur ces modules et leurs connexions ;
- de vérifier des propriétés logicielles et ergonomiques ;
- de modifier, tester et maintenir ces modules logiciels.

Dans notre revue des outils existants, nous verrons qu'il n'existe que des solutions partielles. Pour comparer les outils existants entre eux, nous établissons d'abord une grille d'analyse, exposée dans la section suivante.

2 Grille d'analyse

Notre grille d'analyse regroupe un ensemble de critères sur lesquels nous nous basons pour comparer plusieurs outils de conception logicielle et de développement de l'interaction multimodale. Les critères que nous avons choisis répondent à plusieurs préoccupations. Nous les organisons en quatre ensembles :

- La carte d'identité de l'outil permet d'appréhender la portée de l'outil et considérer aussi la capacité d'expression et les requis pour la prise en main ;
- Les critères liés à l'interaction homme-machine concernent l'utilisateur final du système interactif (et non de l'outil), ainsi que le contexte d'usage ;
- Les critères consacrés à la combinaison de modalités d'interaction ;
- Les critères spécifiques aux mécanismes de fusion des données multimodales dont la plupart sont issus des ateliers sur l'ingénierie des Interfaces Homme-Machine [IHM 1992].

2.1 Carte d'identité de l'outil

Couverture du cycle de développement : il s'agit de spécifier pour quelles étapes du cycle de développement logiciel l'outil est adapté (cf. section 1 de ce chapitre).

Support de prototypage : définit par [Myers 2000], ce critère renseigne sur la capacité de l'outil à prendre en compte le prototypage rapide (appelé aussi RAD – Rapid Application Development) permettant des développements itératifs utilisés lorsque, par exemple, les besoins sont mal cernés.

Cible utilisateur : il s'agit de spécifier à qui s'adresse l'outil. Les cibles correspondent aux acteurs du cycle de vie — concepteurs, développeurs, ergonomes, testeurs, etc.

Difficulté d'apprentissage : définit par [Myers 2000], ce critère mesure la difficulté pour apprendre à utiliser correctement l'outil. Les valeurs peuvent être "très faible", "faible", "moyen", "élevé", "très élevé".

Pouvoir d'expression : définit par [Myers 2000], ce critère concerne la capacité de l'outil à traiter toutes les possibilités d'interaction multimodale que nous avons cernées au sein de notre modèle conceptuel de la Partie 1. Les valeurs peuvent être "très faible", "faible", "moyen", "élevé", "très élevé".

Dépendance d'une technologie de développement : il s'agit de spécifier ici la dépendance éventuelle avec une technologie de développement telle la technologie procédurale, objet, à composants ou modèles.

Dépendance d'un langage de programmation : ce critère renseigne si l'outil est dépendant d'un langage de programmation comme Java, C++, etc.

Représentation des données : le format des données manipulées par l'outil peut être "unique" (quel que soit le niveau d'abstraction) ou "multiple" (adapté au niveau d'abstraction).

Gestion du code source : ce critère définit la manière dont est géré le code source dans l'outil. Trois valeurs sont possibles : "manuelle" (le code source est à la charge de l'utilisateur et nécessite un effort de programmation), "automatique" (la gestion du code source est masquée par l'outil) ou "semi-automatique" (la gestion du code source est en partie prise en charge par l'outil mais une partie peut ou doit être insérée par l'utilisateur).

Prévisibilité : définit par [Myers 2000], ce critère décrit si l'outil utilise des techniques d'automatisation dont le résultat est parfois imprévisible par les programmeurs.

2.2 Critères d'IHM

Prise en compte des capacités de l'utilisateur : ce critère renseigne sur la capacité de l'outil à prendre en compte les capacités physiques et cognitives des utilisateurs.

Prise en compte du contexte : ce critère concerne la capacité de l'outil à traiter des informations sur le contexte d'usage qui ont une incidence sur l'interaction, comme le niveau sonore, le contexte sociale, etc.

2.3 Critères pour la multimodalité

Généricité par rapport aux modalités d'interaction : ce critère définit si toutes les modalités d'interaction d'entrée peuvent être prises en compte par l'outil.

Généricité des mécanismes de fusion : ce critère concerne la capacité des mécanismes à fusionner les données de toutes les modalités d'interaction en entrée.

Nature des combinaisons possibles : il s'agit de définir si l'outil propose explicitement des solutions de combinaisons entre les modalités d'interaction, comme définies au chapitre 4 (complémentarité, équivalence, redondance ou redondance/équivalence).

Temporalité : ce critère caractérise l'usage temporel des modalités d'interaction qui peut être défini par l'outil, au sens d'UOM présenté au chapitre 3 [Nigay 1994]. Il peut être "séquentiel", "parallèle", les deux ou aucun des deux.

2.4 Critères pour la fusion des données

Si l'outil inclut un ou plusieurs mécanismes de fusion, nous les caractérisons par les critères suivants :

Stratégie d'intégration : il s'agit de décrire la manière dont sont combinées les modalités d'interaction [IHM 1992]. La stratégie peut être "précoce" ou "différée". Une stratégie précoce implique que les données, issues des diverses modalités d'interaction à combiner, sont traitées directement lorsqu'elles sont émises. Une stratégie différée permet de vérifier la pertinence de l'information avant sa propagation. A l'inverse de la stratégie différée, la stratégie précoce peut amener à défaire une fusion.

Proximité temporelle : ce critère détermine si les événements issus des modalités d'interaction sont mis en correspondance en fonction de leur proximité temporelle. Plusieurs paramètres permettent de définir cette proximité temporelle comme une fenêtre de détection et une limite d'attente d'événements multimodaux. Les travaux d'Allen [Allen 1983] et de Vernier [Vernier 2001] définissent plusieurs cas de modalités proches temporellement.

Complémentarité logique : ce critère définit si les événements structurellement complémentaires sont fusionnés, même s'ils sont temporellement distants [IHM 1992].

Incompatibilité des modalités à combiner : il s'agit de déterminer si le processus évite d'intégrer les modalités d'interaction qui ne peuvent pas être utilisées conjointement [IHM 1992].

Ordonnancement des données multimodales : ce critère renseigne si l'utilisation de plusieurs modalités d'interaction, dont les données sont à combiner, respecte une chronologie définie.

3 Outils existants

Malgré la maturité des résultats conceptuels et la multiplicité des modalités d'interaction de plus en plus robustes, les outils actuels de conception logicielle et de développement ne sont pas adaptés [Myers 2000]. La plupart des outils permettent de résoudre des problèmes techniques précis, mais il existe peu d'outils offrant des solutions complètes. Cette section présente une étude des différents outils pour la multimodalité. Chaque outil est présenté de façon globale puis détaillé en fonction de notre grille d'analyse. Pour remplir cette grille, nous nous sommes basés sur les publications concernant les outils présentés. La définition des critères est ainsi soumise à notre bonne interprétation des travaux consultés, notamment lorsque le critère étudié est peu explicité dans le document de référence.

Les outils présentés offrent des solutions à des problèmes distincts du domaine de la multimodalité. Les outils sont organisés selon leur carte d'identité et en particulier selon leur couverture des phases du cycle de vie. Nous commençons par décrire les outils qui proposent des architectures conceptuelles pour interfaces multimodales sans contrainte sur le mode d'implémentation. Il s'agit des outils du W3C [W3C 2003] et FAME [Duarte 2006]. Nous présentons ensuite des outils de conception logicielle pour les mécanismes de combinaisons des modalités d'interaction : il s'agit de l'approche à creusets [Nigay 1994] et à base de règles [Bellik 1992]. Puis, nous analysons trois outils dédiés à l'interaction multimodale naturelle. Ces outils concernent les phases de conception logicielle et de codage, et traitent de la fusion d'événements provenant de modalités naturelles : IMBuilder/MEngine [Bourguet 2003], Quickset [Cohen 1997] et Embassi [Elting 2003]. Enfin, nous présentons des outils qui proposent des solutions complètes, aussi bien conceptuelles qu'implémentationnelles, avec Intuikit [Intuikit 2006], Icon [Dragicevic 2004], ISL4WComp [Cheung 2006] et PetShop [Bastide 1998] [Schyn 2005].

3.1 Outil d'intégration multimodal du W3C

Le groupe de recherche du W3C a défini un outil multimodal pour les systèmes informatiques du web. Non encore finalisée, la solution consiste en un modèle conceptuel permettant de définir, mettre en œuvre et combiner des modalités d'interaction d'entrée et de sortie [W3C 2003]. Les modalités d'interaction d'entrée regroupent la parole, l'utilisation de touche clavier ou la reconnaissance de geste (manipulation de la souris, etc.), que nous retrouvons actuellement sur Internet.

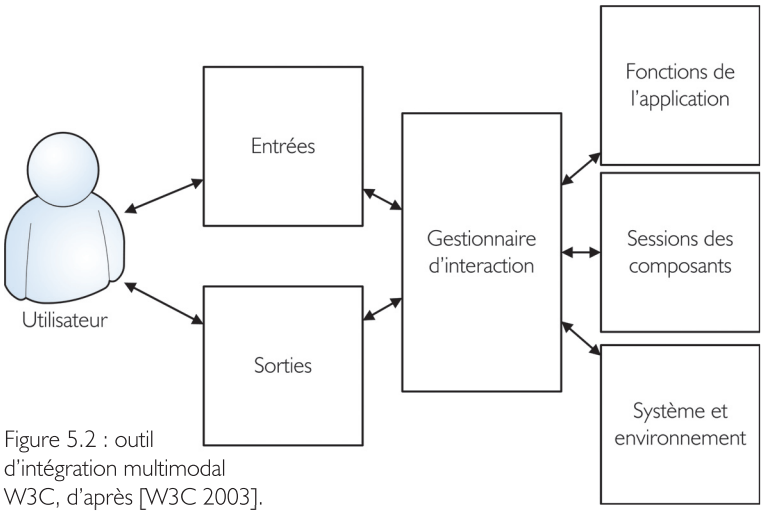


Figure 5.2 : outil d'intégration multimodal W3C, d'après [W3C 2003].

Comme présenté à la Figure 5.2, l'outil propose des composants Entrées, Sorties, Gestionnaire d'interaction et trois autres composants relatifs à l'application (Fonctions de l'application, Sessions des composants, Système et environnement). Comme illustré par la Figure 5.3, les modalités d'interaction sont traitées dans les composants entrées et sorties : les informations des dispositifs d'entrée (resp. sortie) sont captées (resp. organisées) puis interprétées (resp. rendus). Si les modalités sont combinées, les composants entrées et sorties disposent d'un sous-composant appelé composant d'intégration pour l'entrée et composant de génération pour la sortie. La stratégie d'intégration n'est pas spécifiée et dépend de l'application développée. Le lien avec le reste de l'application se fait dans le gestionnaire d'interaction. La fusion des deux énoncés s'effectue dans le sous-composant d'intégration (Entrées), qui transmet l'énoncé combiné au gestionnaire d'interaction, qui exécute à son tour la requête et émet les informations pour la sortie. Le format d'échange de données entre composants est multiple et dépend principalement des entités communicantes. Néanmoins, ce sont toujours des langages à balises qui sont employés.

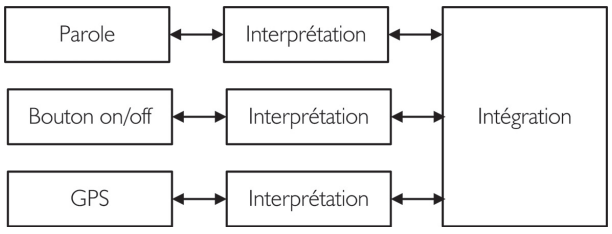


Figure 5.3 : décomposition du composant "Entrée" (Outil du W3C), d'après [W3C 2003].

Définissant les spécifications de chaque composant et le format des données échangées, utiles aux phases de conception globale et détaillée, l'outil est destiné aux concepteurs du logiciel (informaticiens). La difficulté d'apprentissage est moyenne car il faut que les concepteurs assimilent toutes les spécifications des composants et les différents formats d'échange entre composants. Le pouvoir d'expression est élevé car n'importe quel système multimodal peut être spécifié. L'outil ne dépend ni d'une technologie de développement particulière ni d'un langage de développement particulier, mais se base sur une spécification à base de langages à balises de type XML. Le modèle proposé spécifie n'importe quelle modalité d'interaction avec une importance particulière accordée à celle déjà utilisable sur Internet : parole, utilisation des touches d'un clavier, reconnaissance de gestes. L'outil ne propose pas de solution concernant le codage. Néanmoins et même si

les mécanismes des composants d'intégration (fusion des modalités d'interaction) n'ont pas de spécification précise, ils doivent répondre à certains requis comme la prise en charge des diverses formes de combinaisons possibles (Complémentarité, Redondance, Equivalence) et la gestion d'événements parallèles et séquentiels. Pour les autres critères de fusion, il n'existe pas de spécification. En ce qui concerne les critères pour l'IHM, le modèle de l'utilisateur n'est pas pris en compte, alors que le modèle de l'environnement est directement intégré à l'approche sous la forme d'un composant (Système et environnement). Le Tableau 5.1 résume les caractéristiques de l'outil, selon notre grille d'analyse.

Carte d'identité	
Couverture du cycle de développement	Conception globale et détaillée
Support de prototypage	Non
Cible utilisateur	Concepteurs informaticiens
Difficulté d'apprentissage	Moyenne
Pouvoir d'expression	Elevé
Dépendance d'une technologie de développement	Non
Dépendance d'un langage de programmation	Non
Représentation des informations	Multiple – langages à balises
Gestion du code source	Non définie
Prévisibilité	Non applicable
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Non
Prise en compte de l'environnement extérieur	Oui
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Oui
Généricité des mécanismes de fusion	Non définie
Nature des combinaisons possibles	Complémentarité, Redondance, Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Non définie
Proximité temporelle	Non définie
Complémentarité logique	Non définie
Incompatibilité des modalités à combiner	Non définie
Ordonnancement des données multimodales	Non défini

Tableau 5.1 : propriétés de la plate-forme W3C.

3.2 FAME

Présenté dans [Duarte 2006], l'outil FAME (Framework for Adaptive Multimodal Environments) propose une solution conceptuelle pour le développement des applications multimodales adaptatives. Utilisable par des concepteurs informaticiens ou non informaticiens, l'outil concerne les étapes de conception globale et détaillée du cycle de vie du logiciel. FAME propose une architecture conçue pour s'adapter aux actions utilisateurs, aux événements systèmes et aux modifications de l'environnement. Comme le montre la Figure 5.4, le système multimodal adaptatif selon l'architecture FAME est constitué de deux niveaux : interne et externe. Le niveau interne, appelé aussi module d'adaptation, comprend tous les modèles utiles au mécanisme d'adaptation. Les modèles de l'utilisateur, de l'environnement et de la plate-forme matérielle cible, se traduisant par un ensemble de propriétés, sont explicites dans l'outil. Le niveau externe correspond à la couche multimodale. Pour les entrées, les actions utilisateurs sont prises en compte et éventuellement fusionnées dans le composant de fusion multimodale adaptatif. Le mécanisme de ce dernier composant repose sur des règles d'adaptation et sur une matrice comportementale qui enregistre les données relatives à l'interaction et qui les mets en correspondance avec le modèle de l'utilisateur et de la plate-forme. En plus de cette architecture, FAME définit un guide de conception définissant six étapes à suivre pour concevoir et développer ce type d'application. Chaque étape aide à la spécification d'un ou plusieurs modules de l'architecture présentée à la Figure 5.4.

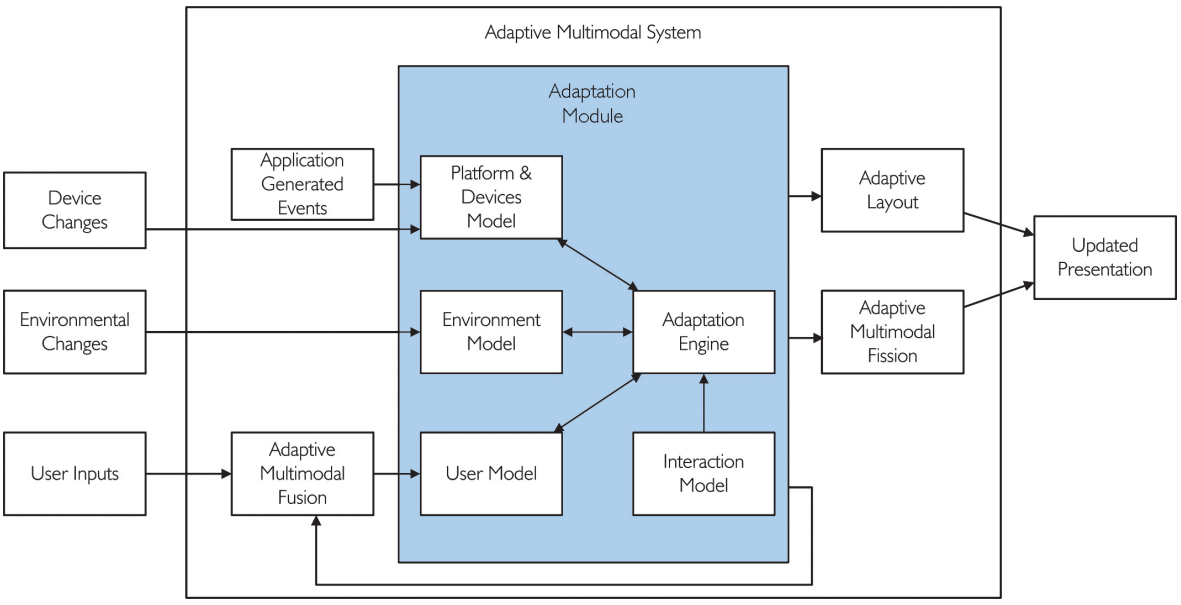


Figure 5.4 : architecture des applications suivant FAME, d'après [Duarte 2006].

FAME traite spécifiquement du problème de l'adaptation des modalités d'interaction aux actions utilisateurs, aux événements systèmes et aux modifications de l'environnement. L'architecture est générale aux applications multimodales. Par rapport aux autres approches, cet outil propose de mettre en œuvre les modèles de l'utilisateur et de l'environnement afin de proposer à l'utilisateur une interface de sortie qui leur est adaptée. La difficulté d'apprentissage de cet outil est faible car le guide de conception permet de spécifier rapidement et simplement les différents modules de l'architecture. Cependant, l'accent n'est pas mis sur la combinaison de modalités. Même si un module est identifié dans l'architecture, cet outil ne propose pas de solution précise pour les mécanismes de combinaison des modalités d'interaction. N'importe quelle combinaison peut alors être définie comme l'une des dimensions de la matrice comportementale, mais aucun guide n'est proposé pour identifier les différentes formes de combinaisons possibles. Le pouvoir d'expression en est ainsi élevé mais non dirigé. Enfin, le codage du système n'est pas abordé dans cet outil. Le Tableau 5.2 rassemble les caractéristiques de l'outil FAME.

Carte d'identité	
Couverture du cycle de développement	Conception globale et détaillée
Support de prototypage	Non
Cible utilisateur	Concepteurs informaticiens et non informaticiens
Difficulté d'apprentissage	Faible
Pouvoir d'expression	Elevé
Dépendance d'une technologie de développement	Non
Dépendance d'un langage de programmation	Non
Représentation des informations	Unique (matrice comportementale)
Gestion du code source	Non définie
Prévisibilité	Non applicable
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Oui
Prise en compte de l'environnement extérieur	Oui
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Oui
Généricité des mécanismes de fusion	Non définie
Nature des combinaisons possibles	Non définie
Temporalité	Non définie

Critères pour la fusion des données	
Stratégie d'intégration	Non définie
Proximité temporelle	Non définie
Complémentarité logique	Non définie
Incompatibilité des modalités à combiner	Non définie
Ordonnancement des données multimodales	Non défini

Tableau 5.2 : propriétés de l'outil FAME.

3.3 Approche multimodale à creusets

L'approche multimodal à creusets [Nigay 1994] consiste en un algorithme pour la fusion des données issues des modalités d'interaction. Ces données sont représentées dans un formalisme unique : le creuset. Un creuset est une matrice correspondant aux champs des requêtes de l'application. Lorsque plusieurs modalités d'interaction sont combinées, les creusets stockant les données des différentes modalités sont fusionnés. La fusion de deux creusets est un creuset. Trois types de fusion sont définis : la micro fusion (parallélisme), la macro fusion (proximité temporelle) et la fusion contextuelle (basée sur le contexte courant de l'interaction). Afin d'illustrer l'approche, l'application MATIS (Multimodal Airline Travel Information System) a été développée. C'est un système d'information multimodal sur les transports aériens permettant des commandes dictées, écrites ou spécifiées par manipulation directe. L'utilisateur exprime des requêtes au système pour planifier son voyage et MATIS lui fournit des informations sur les vols entre deux villes. Pour spécifier une requête, l'utilisateur dispose de trois modalités d'interaction. Elle/il peut utiliser la souris (manipulation directe), le clavier (en écrivant la requête en langue naturelle ou en remplissant un formulaire) ou la parole (langue naturelle). Enfin, les trois modalités précédemment citées peuvent être utilisées de façon combinée.

Utilisable par des programmeurs, cet outil propose un mécanisme générique de combinaison par rapport aux modalités d'interaction. La difficulté d'apprentissage est moyenne car il faut assimiler les concepts sous-jacents à l'approche comme la représentation en creusets. Le pouvoir d'expression est élevé car la plupart des systèmes multimodaux peuvent être implémentés selon cette approche. Cependant, ni le modèle de l'utilisateur, ni le modèle de l'environnement n'est pris en compte. La gestion du code source est manuelle et même si l'implémentation de l'application MATIS a été réalisée avec le langage Objective-C, l'outil ne préconise ni technologies, ni langages de programmation spécifiques. En ce qui

concerne la fusion des données, le mécanisme les combine dans un nouveau creuset, envoyé immédiatement au module suivant. Il s'agit donc d'une stratégie d'intégration précoce. Le Tableau 5.3 rassemble les propriétés de l'approche par creusets.

Carte d'identité	
Couverture du cycle de développement	Conception globale et détaillée
Support de prototypage	Non
Cible utilisateur	Programmeurs
Difficulté d'apprentissage	Moyenne
Pouvoir d'expression	Elevé
Dépendance d'une technologie de développement	Non
Dépendance d'un langage de programmation	Non – système réalisé en Objective-C
Représentation des informations	Unique – le creuset
Gestion du code source	Manuelle
Prévisibilité	Non applicable
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Non
Prise en compte de l'environnement extérieur	Non
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Oui
Généricité des mécanismes de fusion	Oui
Nature des combinaisons possibles	Complémentarité, Redondance, Equivalence, Redondance/Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Précoce
Proximité temporelle	Oui
Complémentarité logique	Oui
Incompatibilité des modalités à combiner	Oui
Ordonnancement des données multimodales	Non

Tableau 5.3 : propriétés de l'approche à creusets.

3.4 Outil multimodal à base de règles

Comme pour l'outil précédent, l'outil présenté dans [Bellik 1992] concerne la combinaison des modalités d'interaction. Les informations à fusionner (issues de données spécifiées selon plusieurs modalités d'interaction) font référence à

l'une des trois opérations suivantes : référence à une commande à exécuter, à un argument et à une entrée de donnée. L'intégration se fait soit par fusion locale (une donnée est associée à l'un des arguments de la commande), soit par fusion globale (production d'énoncés en rassemblant dans une structure commune toutes les données nécessaires à l'exécution d'une commande). La fusion locale a lieu entre deux informations qui ont une complémentarité logique, des types compatibles et qui sont proches temporellement. La fusion globale obéit à un certain nombre de règles. Comme exemple de règles, la structure commune doit contenir une seule information qui fait référence à une commande. Une autre règle détermine que si une donnée n'a pas fait l'objet d'une fusion locale, elle est associée, après vérification de certaines conditions, au premier argument de la commande en cours. Dernier exemple de règle, une information qui n'a pas fait l'objet d'une fusion locale et qui n'a pu être associée à un argument est ignorée. Au final, la fusion globale est achevée lorsque tous les arguments de la commande en cours ont été associés à des données issues des modalités d'interaction.

Cette approche a été illustrée par une application graphique, LIMSI-DRAW, qui permet de créer et de manipuler des formes géométriques élémentaires à travers un ensemble de commandes multimodales. Trois périphériques en entrée, correspondant à trois modalités d'interaction en entrée, sont disponibles : un système de reconnaissance vocale, un écran tactile et une souris. L'expression des énoncés des commandes de LIMSI-DRAW peut être faite selon une seule modalité d'interaction ou selon la composition de deux ou trois modalités d'interaction.

Comme pour l'approche à creusets, cet outil à base de règles pour la conception globale et détaillée est validé par une expérience pratique. La difficulté d'apprentissage est moyenne car il suffit d'assimiler les concepts sous-jacents à l'approche comme les différentes règles et les différents types de fusion. Le pouvoir d'expression de cet outil est élevé car la plupart des systèmes multimodaux peuvent être spécifiés et l'approche est générique par rapport aux modalités d'interaction. L'outil ne dépend pas d'une technologie ou d'un langage de programmation particulier. Le code source qui suit cette spécification doit être écrit par l'utilisateur sans guide de conception. Les capacités de l'utilisateur et les données liées à l'environnement ne sont pas explicites. Les mécanismes de fusion sont génériques par rapport aux modalités d'interaction et permettent de traiter les cas d'utilisation complémentaires, équivalents ou redondants (parallèle ou séquentiel). La fusion des données est toujours effectuée de façon différée. En effet, elle a lieu à haut niveau d'abstraction dans le contrôleur de dialogue (module gérant l'enchaînement des tâches) lorsque que toutes les actions de l'utilisateur ont été spécifiées. Le Tableau 5.4 synthétise les propriétés de cet outil.

Carte d'identité	
Couverture du cycle de développement	Conception globale et détaillée
Support de prototypage	Non
Cible utilisateur	Concepteurs informaticiens
Difficulté d'apprentissage	Moyenne
Pouvoir d'expression	Elevé
Dépendance d'une technologie de développement	Non
Dépendance d'un langage de programmation	Non
Représentation des informations	Multiple
Gestion du code source	Manuelle
Prévisibilité	Non applicable
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Non
Prise en compte de l'environnement extérieur	Non
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Oui
Généricité des mécanismes de fusion	Oui
Nature des combinaisons possibles	Complémentarité, Redondance, Equivalence, Redondance/Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Différée
Proximité temporelle	Oui
Complémentarité logique	Oui
Incompatibilité des modalités à combiner	Oui
Ordonnancement des données multimodales	Non

Tableau 5.4 : propriétés du moteur de fusion à base de règles.

3.5 IMBuilder et MEngine

Destiné aux systèmes interactifs combinant la parole et le geste, cet outil permet de concevoir et exécuter rapidement plusieurs versions d'une interface multimodale [Bourguet 2003]. Il est composé de deux modules IMBuilder et MEngine. IMBuilder est un outil graphique permettant de spécifier l'usage de la parole et du geste, sous la forme d'une collection de machines à états (automates finis), sauvegardées dans une

structure XML. Les machines à états représentent les combinaisons de modalités d'interaction possibles. Par exemple, la commande "déplacer un objet" peut être décrite par la séquence : bouton de la souris appuyé sur un objet graphique, souris déplacée, bouton de la souris relâché. Alternativement, la même commande peut être définie par une séquence différente telle que : bouton de la souris appuyé sur un objet graphique, commande orale "déplace" prononcée, bouton de la souris appuyé sur une position à l'écran. Plusieurs spécifications peuvent être définies afin de tester les différentes versions de l'interaction multimodale. Ces spécifications sont ensuite exécutées dans l'environnement MEngine qui met en œuvre les processus de reconnaissance vocale et gestuelle. Programmé en Java, MEngine utilise actuellement le système de reconnaissance vocale ViaVoice® d'IBM et la boîte à outil Satin pour la reconnaissance gestuelle de la souris [Hong 2000]. En utilisant les fichiers XML générés par IMBuilder, MEngine lance les systèmes de reconnaissance vocale et gestuelle après avoir reconstruit la collection de machines à états. Les actions de l'utilisateur sont interprétées, traitées et exécutées directement par MEngine selon les machines à états définies.

Cet outil propose une solution pour les phases de conception détaillée et de codage. Il permet le prototypage rapide en autorisant plusieurs spécifications de l'interaction multimodale. Les spécifications peuvent être exécutées rapidement grâce à la génération automatique de structure XML. L'automatisation de la programmation destine cet outil à des concepteurs qui ne sont pas forcément informaticiens. La difficulté d'apprentissage est moyenne car il convient d'assimiler les règles des machines à états finis pour définir les usages possibles des deux modalités d'interaction. Cependant, le pouvoir d'expression de l'outil est faible car seulement deux modalités d'interaction sont prises en charge : la reconnaissance de geste 2D (effectué à la souris) et la reconnaissance vocale de mots-clés. Les mécanismes de fusion sont définis dans les machines à états et permettent de spécifier uniquement des combinaisons séquentielles, complémentaires, équivalentes ou redondantes. Les machines à états utilisent par définition une stratégie précoce, où l'ordre d'utilisation des différentes modalités d'interaction est spécifié. La fusion des données se base non pas sur la proximité temporelle, mais sur la complémentarité logique des données à combiner. La représentation des données est multiple et dépend des systèmes utilisés pour les reconnaissances gestuelle et vocale. Le Tableau 5.5 résume les caractéristiques de cet outil selon notre grille d'analyse.

Carte d'identité	
Couverture du cycle de développement	Conception globale, Codage
Support de prototypage	Oui
Cible utilisateur	Concepteurs informaticiens et non informaticiens
Difficulté d'apprentissage	Moyenne
Pouvoir d'expression	Faible
Dépendance d'une technologie de développement	Objet
Dépendance d'un langage de programmation	Java
Représentation des informations	Multiple
Gestion du code source	Automatique
Prévisibilité	Oui
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Non
Prise en compte de l'environnement extérieur	Non
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Non – parole et geste
Généricité des mécanismes de fusion	Non
Nature des combinaisons possibles	Complémentarité, équivalence, redondance
Temporalité	Séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Précoce
Proximité temporelle	Non
Complémentarité logique	Oui
Incompatibilité des modalités à combiner	Oui
Ordonnancement des données multimodales	Oui

Tableau 5.5 : propriétés de l'outil IMBulder/MEngine.

3.6 Quickset

Quickset [Cohen 1997] est un outil qui propose une solution architecturale et implémentationnelle conçue pour répondre aux problèmes de la fusion spécifique entre le geste et la parole. L'outil a été utilisé pour réaliser un simulateur d'un poste de commandement militaire informatisé et pour un système de recherche de données médicales. Quickset repose sur une architecture multi-agents OAA (Open Agent Architecture). Chaque agent peut être exécuté sur différentes configurations de matériel et implémenté dans différents langages (Prolog, C++, Java, Basic, etc.). Plusieurs agents sont définis et incarnent les différentes parties de l'application interactive (modalités d'interaction d'entrée et de sortie, noyau fonctionnel, etc.).

Tous les agents sont mis en correspondance et communiquent par l'intermédiaire d'un module central, appelé le Facilitator, qui est chargé de l'enchaînement des tâches. Chaque modalité d'interaction en entrée est incarnée par un agent. Nous retrouvons, par exemple, l'agent reconnaissance de geste et l'agent reconnaissance de la parole. Concernant la fusion des deux modalités d'interaction naturelles, un agent, appelé agent d'intégration multimodale, reçoit via le Facilitator les données des deux modalités d'interaction (dans un format particulier pour chacune des modalités) et se charge d'identifier la meilleure interprétation de leur usage : soit il s'agit d'un usage exclusif, soit d'un usage combiné. La compatibilité temporelle et sémantique entre les données multimodales est analysée afin de comprendre l'intention de l'utilisateur.

Quickset fournit une solution pour les phases de conception et de codage. Sa difficulté d'apprentissage est élevée car il demande des connaissances poussées en programmation afin de programmer le mécanisme de chaque agent. Malgré une architecture à agents génériques, qui peut prendre en compte n'importe quelles modalités d'interaction, Quickset est spécialisé dans la gestion des deux modalités d'interaction naturelles (la parole et le geste), ce qui limite son pouvoir d'expression. Néanmoins, tous les usages possibles de ces deux modalités sont étudiés. Le moteur de fusion est donc dépendant de ces deux modalités d'interaction, mais il permet de couvrir tous les cas de combinaison en vérifiant l'aspect temporel et sémantique des données produites par l'utilisateur. Le codage doit être réalisé manuellement, destinant cet outil aux informaticiens. Le Tableau 5.6 rassemble les caractéristiques de Quickset.

Carte d'identité	
Couverture du cycle de développement	Conception globale et détaillée, codage
Support de prototypage	Non
Cible utilisateur	Concepteurs informaticiens, programmeurs
Difficulté d'apprentissage	Elevée
Pouvoir d'expression	Moyen
Dépendance d'une technologie de développement	Agents
Dépendance d'un langage de programmation	Non
Représentation des informations	Multiple
Gestion du code source	Manuelle
Prévisibilité	Non applicable
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Non
Prise en compte de l'environnement extérieur	Non

Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Non – parole, geste
Généricité des mécanismes de fusion	Non
Nature des combinaisons possibles	Complémentarité, Redondance, Equivalence, Redondance/Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Précoce et différée
Proximité temporelle	Oui
Complémentarité logique	Oui
Incompatibilité des modalités à combiner	Oui
Ordonnancement des données multimodales	Non

Tableau 5.6 : propriétés de l'outil Quickset.

3.7 Embassi

Comme IMBuilder/MEngine et Quickset, Embassi propose une solution architecturale et implémentationnelle pour les modalités d'interactions naturelles [Elting 2003]. Traitant des entrées comme des sorties, nous nous intéressons uniquement au premier aspect. L'architecture d'Embassi définit différents agents pour représenter une application complète. Une modalité d'interaction est définie par un couple de deux agents. Le premier agent incarne le dispositif et un second agent analyse les données du dispositif pour leur donner une sémantique (il peut être assimilé au langage d'interaction). Les combinaisons possibles entre les modalités d'interaction sont gérées par un agent, appelé PMI (Polymodal Input Module) qui fusionne uniquement les données des modalités d'interaction de la parole, du geste et de la manipulation directe d'interface graphique classique et ce en fonction de l'application choisie. Le mécanisme de combinaison qui n'est donc pas générique permet toutefois de déterminer si l'utilisateur utilise les modalités d'interaction de façon indépendante ou combinée. Pour affiner le mécanisme de fusion, le PMI est en relation avec un gestionnaire de contexte qui enregistre des données relatives, entre autres, à l'utilisateur et à l'environnement. Il est à noter que seule la complémentarité de type "mets ça là" est traitée. Pour l'implémentation, les agents doivent être développés en fonction de l'application. Un exemple d'application a été réalisé en Java et Prolog. Les agents communiquent par des messages décrits en XML.

Destinés aux concepteurs informaticiens et aux développeurs, cet outil couvre les phases de conception et de codage. Malgré une architecture générale qui peut s'appliquer à n'importe quelle application, l'implémentation est très dépendante de l'application cible. La difficulté d'apprentissage est élevée car les utilisateurs doivent avoir une bonne connaissance de la programmation pour programmer manuellement les mécanismes des agents et leurs connexions. Le pouvoir d'expression est moyen car le choix des modalités d'interaction et leurs combinaisons possibles sont limités. Actuellement, il est possible de télécharger les agents d'Embassi pour une application domotique, afin de contrôler sa télévision, son magnétoscope, etc. Tous les agents sont spécifiques à cette application : les modalités d'interaction comprennent un système de reconnaissance vocale, de désignation par laser (geste) et de manipulation directe d'interface WIMP. Le PMI est aussi dépendant de l'application et des modalités d'interaction mises en place. Il ne propose actuellement qu'un emploi équivalent ou complémentaire des modalités d'interaction utilisées parallèlement ou séquentiellement. Les mécanismes de fusion suivent une stratégie précoce en envoyant les premières combinaisons de données qui ont été détectées par proximité temporelle ou par leur complémentarité logique. Le Tableau 5.7 rassemble les caractéristiques de l'outil.

Carte d'identité	
Couverture du cycle de développement	Conception globale et détaillée, codage
Support de prototypage	Non
Cible utilisateur	Concepteurs informaticiens, programmeurs
Difficulté d'apprentissage	Elevée
Pouvoir d'expression	Moyen
Dépendance d'une technologie de développement	Agents
Dépendance d'un langage de programmation	Java, Prolog
Représentation des informations	Multiple – XML
Gestion du code source	Manuelle
Prévisibilité	Non applicable
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Oui
Prise en compte de l'environnement extérieur	Oui
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Non – parole, geste, WIMP
Généricité des mécanismes de fusion	Non

Critères pour la multimodalité	
Nature des combinaisons possibles	Complémentarité, Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Précoce
Proximité temporelle	Oui
Complémentarité logique	Oui
Incompatibilité des modalités à combiner	Oui
Ordonnancement des données multimodales	Non

Tableau 5.7 : propriétés de l'outil Embassi.

3.8 Intuikit

L'outil Intuikit [Intuikit 2006] est un produit commercial proposé par l'entreprise IntuiLab. Il propose une solution pour la conception et le développement d'interfaces multimodales et mobiles couvrant les cycles de développement classiques ou itératifs. Cet outil se compose de trois composants : la toolkit, les éditeurs et l'environnement d'exécution. La toolkit fournit un ensemble de composants logiciels offrant des bibliothèques d'objets pour développer les interfaces et les applications. Les éditeurs permettent de définir les modalités d'entrée et de sortie, leurs comportements et la fusion multimodale si nécessaire. L'environnement d'exécution a plusieurs cibles telles les PC, PDA et PC tactiles.

Intuikit fournit une boîte à outil idéale pour les projets en équipe du designer aux développeurs en passant par les concepteurs informatiques. Cependant, sa difficulté d'apprentissage est élevée car son utilisation demande des connaissances poussées dans les langages de programmation ou dans les éditeurs de conception graphique. Le pouvoir d'expression est toutefois très élevé car la puissance des langages de programmation permet de concevoir et développer n'importe quel système interactif. Contrairement aux outils précédemment présentés et demandant aussi une programmation manuelle, le pouvoir d'expression d'Intuikit est plus élevé . En effet, le choix des modalités d'interaction et des formes de combinaison n'est pas limité, comme nous le retrouvons dans les autres outils spécialisés dans les modalités d'interaction naturelles. A ce jour, le moteur de fusion multimodale (complémentarité, équivalence) permet la combinaison séquentielle ou simultanée des modalités d'interaction. Le Tableau 5.8 présente toutes les informations disponibles.

Carte d'identité	
Couverture du cycle de développement	Conception détaillée, codage
Support de prototypage	Oui
Cible utilisateur	Designer, concepteurs informaticiens, programmeurs
Difficulté d'apprentissage	Elevée
Pouvoir d'expression	Très élevé
Dépendance d'une technologie de développement	Non définie
Dépendance d'un langage de programmation	Perl, C++
Représentation des informations	Multiple
Gestion du code source	Manuelle
Prévisibilité	Non applicable
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Non
Prise en compte de l'environnement extérieur	Non
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Non
Généricité des mécanismes de fusion	Non
Nature des combinaisons possibles	Complémentarité, Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Non communiquée
Proximité temporelle	Non communiquée
Complémentarité logique	Non communiquée
Incompatibilité des modalités à combiner	Non communiquée
Ordonnancement des données multimodales	Non communiqué

Tableau 5.8 : propriétés de l'outil Intuikit.

3.9 Icon

Icon (Input Configurator) est un outil visuel de prototypage rapide pour la mise en œuvre de multiples dispositifs dans une application [Dragicevic 2004]. Il est basé sur un modèle conceptuel qui permet de spécifier ce qui est appelé une configuration d'entrée. Montrée à la Figure 5.5, une configuration d'entrée définie une interaction multimodale basée sur plusieurs dispositifs. Plusieurs briques de base sont retrouvées : les dispositifs système (modules qui décrivent pour la plupart des dispositifs d'entrée physiques), les dispositifs utilitaires (modules qui peuvent être vus comme des adaptateurs ou des langages d'interaction) et les dispositifs d'application

(qui incarnent les objets du domaine). L'outil propose une véritable bibliothèque de dispositifs prêts à l'emploi et un moteur d'exécution. Il est ainsi possible de construire et tester rapidement différentes configurations d'entrée. Le langage de programmation adopté pour les différentes couches de dispositifs et le moteur d'exécution est le Java.

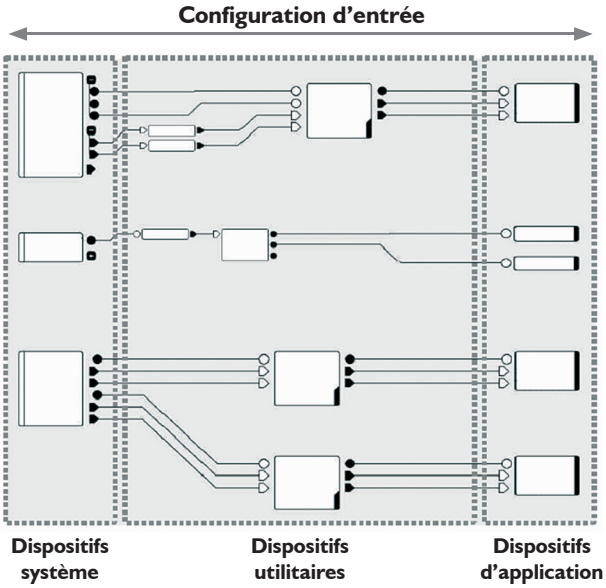


Figure 5.5 : configuration d'entrée, issu de [Dragicevic 2004].

L'outil permet de connecter rapidement les différentes briques de base par la mise en correspondance de leurs ports. Avec son éditeur graphique, il est adapté aux cycles de développement rapide, permettant de réaliser des prototypes. Par exemple, Icon permet de réaliser l'assemblage présentée à la Figure 5.6, qui met en œuvre un dispositif système incarné par la tablette tactile WACOM, deux dispositifs utilitaires (swPick et quikwrite), qui transforment les données de la tablette WACOM, afin de contrôler le widget JText (dispositif d'application) de l'api Swing de Java. Le moteur d'exécution peut alors prendre en compte cette configuration et exécuter directement le code correspondant.

Icon permet donc de concevoir rapidement et simplement des systèmes multimodaux. La gestion automatique du code source (interprétation du modèle sous-jacent) facilite son apprentissage et il est seulement nécessaire d'assimiler les concepts d'assemblage. L'approche de construction visuelle et le moteur d'exécution permet d'élargir la cible des utilisateurs de l'outil, rassemblant informaticiens et

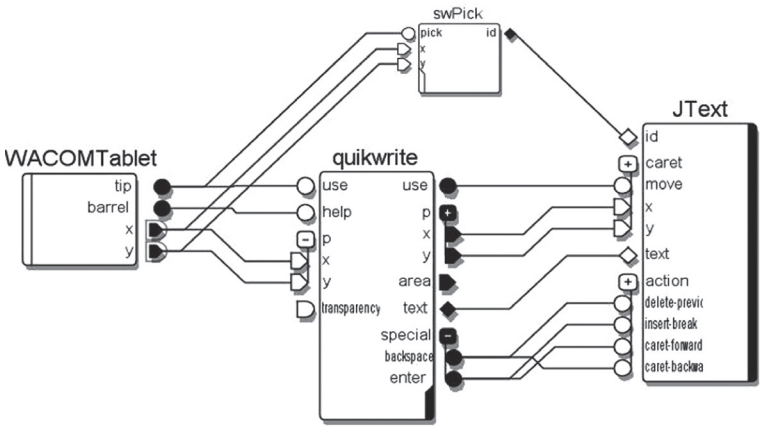


Figure 5.6 : un exemple de configuration d'entrée, issu de [Dragicevic 2004].

non informaticiens. Avec une collection de composants importante et adaptée, le pouvoir d'expression de cet outil est élevé car il permet de construire un grand nombre de système multimodaux. Cependant, la combinaison des modalités d'interaction n'est pas abordée explicitement dans cette approche. Ainsi, il existe des dispositifs utilitaires non génériques permettant de réaliser les différents types de combinaison, en fonction des sorties des dispositifs systèmes. L'aspect temporel n'est néanmoins pas pris en compte et limite la création de mécanismes complexes de combinaisons. Le Tableau 5.9 rassemble les caractéristiques de l'outil.

Carte d'identité	
Couverture du cycle de développement	Conception globale et détaillée, codage
Support de prototypage	Oui
Cible utilisateur	Concepteurs informaticiens et non informaticiens, programmeurs
Difficulté d'apprentissage	Faible
Pouvoir d'expression	Elevé
Dépendance d'une technologie de développement	Objets
Dépendance d'un langage de programmation	Java
Représentation des informations	Multiple
Gestion du code source	Automatique
Prévisibilité	Oui
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Non
Prise en compte de l'environnement extérieur	Non
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Oui
Généricité des mécanismes de fusion	Non
Nature des combinaisons possibles	Complémentarité, Redondance, Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Précoce
Proximité temporelle	Non
Complémentarité logique	Oui
Incompatibilité des modalités à combiner	Oui
Ordonnancement des données multimodales	Non

Tableau 5.9 : propriétés de l'outil Icon.

3.10 ISL4WComp

ISL4WComp [Cheung 2006] est un outil de développement rapide conçu pour les applications vestimentaires (équipement informatique pour utilisateur mobile), contextuelles ou ubiquitaires, faisant intervenir de nombreux dispositifs. L'outil inclut un éditeur graphique de programmation WComp couplé à un langage de spécification de l'interaction appelé ISL4 (Interaction Specification Langage), basé sur les composants logiciels. WComp permet de définir, en C#, les différents composants de l'application (dispositifs, etc.) et fournit un environnement visuel pour assembler les composants. Il incorpore aussi un moteur d'exécution (découverte et adaptation des composants). L'originalité de cet outil réside dans la définition en ISL4 de patrons d'interaction décrivant le comportement des composants. ISL4 est un langage de haut niveau ressemblant au Java ou au C#. Les patrons d'interaction définissent des règles, comme des règles de combinaisons, qui peuvent être réutilisés dans plusieurs systèmes interactifs. Il est à noter que l'aspect temporel n'est pas abordé.

Pour résumer, cet outil répond au problème de mise en place de plusieurs dispositifs dans un système interactif et le résout en adoptant une approche basée sur les composants logiciels. Les phases du cycle de développement concernées sont la conception et le codage. L'éditeur graphique permet de configurer rapidement des nouveaux assemblages de composants, favorisant les démarches de prototypages rapides. Malgré la présence d'un éditeur graphique, la difficulté d'apprentissage est élevée car les utilisateurs de l'outil doivent avoir des connaissances en programmation et doivent assimiler les notions du langage ISL4. Le pouvoir d'expression est très élevé car le langage ISL4 permet de spécifier n'importe quel système multimodal. Toutes les formes de combinaisons peuvent être effectuées par la mise en œuvre de patrons d'interaction qui définissent des règles, permettant ou non la réalisation d'une tâche en fonction de l'état des différents composants. Cependant, l'absence d'estampillage temporel des données limite les mécanismes de combinaisons élaborés. Le Tableau 5.10 rassemble les caractéristiques de cet outil selon notre grille d'analyse.

Carte d'identité	
Couverture du cycle de développement	Conception globale et détaillée, codage
Support de prototypage	Oui
Cible utilisateur	Concepteurs informaticiens, programmeurs
Difficulté d'apprentissage	Elevée
Pouvoir d'expression	Très élevé

Carte d'identité	
Dépendance d'une technologie de développement	Composants
Dépendance d'un langage de programmation	ISL4, C#
Représentation des informations	Multiple
Gestion du code source	Semi-automatique (définition manuelle des patrons de conception en ISL4)
Prévisibilité	Oui
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Non
Prise en compte de l'environnement extérieur	Oui
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Oui
Généricité des mécanismes de fusion	Non
Nature des combinaisons possibles	Complémentarité, Equivalence, Redondance, Redondance/Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Précoce
Proximité temporelle	Non
Complémentarité logique	Oui
Incompatibilité des modalités à combiner	Oui
Ordonnancement des données multimodales	Non défini

Tableau 5.10 : propriétés de l'outil ISL4WComp.

3.11 PetShop

PetShop est un outil graphique d'aide à la conception et au développement des logiciels. Il fournit une notation formelle pour spécifier les comportements de composants intervenant dans les systèmes distribués. Basé sur les réseaux de Pétri, cette notation décrit le comportement aussi bien des parties non interactives qu'interactives d'une application. En ce qui concerne la partie interactive, la description se réalise à partir d'ICO (Objets Coopératifs Interactifs) qui sont des classes d'objets définies à partir de réseaux de Pétri [Bastide 1998]. Alors qu'ils ont été conçus pour les systèmes interactifs WIMP, les ICO ont été étendus par [Schyn 2005] pour la multimodalité, en intégrant un aspect temporel (utile à la combinaison des modalités) et un mécanisme de communication événementielle. Les modalités

d'interaction et leurs combinaisons sont donc incarnées par la description de plusieurs classes ICO étendues. La Figure 5.7 montre l'éditeur graphique avec la spécification des places et des transitions d'un réseau de Pétri dans le formalisme ICO. Suite à la description complète de l'application, cette dernière peut être directement exécutée à partir de l'éditeur. Le développeur peut suivre l'état de cette exécution grâce à la visualisation des jetons qui parcourent le réseau de Pétri.

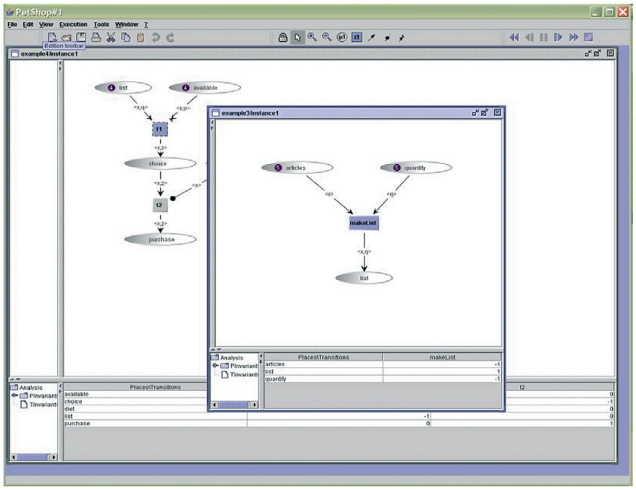


Figure 5.7 : éditeur PetShop.

PetShop fournit une solution pour les étapes de conception, codage et de tests du cycle de développement logiciel. L'éditeur graphique permet de simplifier et accélérer la réalisation de ces étapes. En effet, par la visualisation de l'exécution des réseaux de Pétri, il est possible de vérifier le comportement du système interactif défini à la façon d'un débuggeur (mode pas à pas), de le corriger et de reprendre son exécution. Le pouvoir d'expression de PetShop est très élevé car les réseaux de Pétri peuvent être considérés comme un langage de programmation et n'importe quel système multimodal peut être réalisé. Cet outil permet de couvrir l'ensemble des modalités d'interaction. Cependant, la combinaison des modalités n'est pas explicitement définie et il faut donc créer une classe ICO correspond à la combinaison recherchée. Il est ainsi possible de définir n'importe quels mécanismes de combinaison mais ils seront particuliers aux modalités d'interaction à combiner. Malgré ce manque de généricité, l'extension temporelle qui a été apportée à ICO facilite la mise en place d'algorithmes de combinaison. La gestion du code source est semi-automatique car même si le code, correspondant au réseau de Pétri général, est complètement géré automatiquement par l'outil (interprétation du modèle sous-jacent), l'utilisateur a parfois le besoin d'écrire le comportement d'une place en langage Java. Dans ce cas, la difficulté d'apprentissage est élevée car il faut assimiler le fonctionnement des réseaux de Pétri et le langage de programmation Java. Le Tableau 5.11 rassemble les caractéristiques de cet outil selon notre grille d'analyse.

Carte d'identité	
Couverture du cycle de développement	Conception globale et détaillée, codage, tests
Support de prototypage	Oui
Cible utilisateur	Concepteurs informaticiens, programmeurs, testeurs
Difficulté d'apprentissage	Elevée
Pouvoir d'expression	Très élevé
Dépendance d'une technologie de développement	Objets, ICO
Dépendance d'un langage de programmation	Java
Représentation des informations	Multiple
Gestion du code source	Semi-automatique
Prévisibilité	Oui
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Non
Prise en compte de l'environnement extérieur	Non
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Oui
Généricité des mécanismes de fusion	Non
Nature des combinaisons possibles	Complémentarité, Equivalence, Redondance, Redondance/Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Non définie
Proximité temporelle	Oui
Complémentarité logique	Oui
Incompatibilité des modalités à combiner	Oui
Ordonnancement des données multimodales	Non défini

Tableau 5.11 : propriétés de l'outil PetShop.

4 Résumé du chapitre 5

Les outils étudiés dans ce chapitre se caractérisent par une grande diversité des objectifs. Les Tableau 5.12 et Tableau 5.13 confrontent les onze outils que nous avons analysés. Tous proposent des solutions d'intégration différentes et répondent à certaines particularités de la multimodalité, mais aucun d'eux ne permet de satisfaire la totalité des critères définis.

Face à l'explosion du nombre de modalités d'interaction, il semble primordiale que l'outil recherché permette d'incorporer n'importe quelles modalités d'interaction, à l'instar de FAME, l'approche à creusets, l'outil à base de règles, Icon, ISL4WComp ou PetShop. A ce propos, les outils tels que l'approche à creusets et l'outil à base de règles, fournissant des mécanismes de fusion génériques par rapport aux modalités d'interaction répondent totalement aux perspectives d'évolution des systèmes multimodaux. L'outil doit aussi permettre de réaliser tous les usages possibles et combinaisons de la multimodalité comme dans FAME, l'approche à creusets, Quickset ou PetShop. De plus, il doit concerner le maximum de phases du cycle de développement comme le fait PetShop. La prise en compte des capacités de l'utilisateur et de l'environnement est aussi fondamentale, comme dans les outils W3C, FAME, Embassi et ISL4WComp qui ont complètement intégré ces aspects. La gestion automatique du code source et la facilité d'apprentissage, proposée par exemple dans les outils IMBuilder/MEngine et Icon, élargissent également le nombre d'utilisateurs en prenant en compte des personnes aussi bien informaticiennes que non informaticiennes. Cependant, cette facilité de prise en main ne doit pas limiter le pouvoir d'expression de l'outil, qui doit rester très élevé comme dans Intuikit, PetShop et ISL4WComp. Enfin, au vue des développements itératifs incontournables pour un système interactif, les outils tels qu'IMBuilder/MEngine, Intuikit, PetShop, Icon et ISL4WComp sont complètement adaptés au domaine de l'IHM.

Pour conclure, il n'existe pas d'outil prenant en compte l'intégralité des critères que nous avons définis pour concevoir et développer rapidement, simplement et efficacement n'importe quelle interaction multimodale. Malgré la présence de quelques outils répondant à des points particuliers, concevoir, développer et maintenir un système multimodal restent aujourd'hui des tâches difficiles et longues. Dans ce contexte, notre démarche de travail a alors consisté à définir un nouvel outil qui traite les différents critères de notre grille d'analyse et qui repose sur notre espace conceptuel du chapitre 1. L'outil conçu et développé fait l'objet des trois chapitres suivants.

Caractéristiques / Outils	W3C	FAME	Approche à creusets	Outil à base de règles	IMBuilder/ MEngine
Carte d'identité	Carte d'identité	Carte d'identité	Carte d'identité	Carte d'identité	Carte d'identité
Couverture du cycle de développement	Conception globale et détaillée	Conception globale et détaillée	Conception globale et détaillée	Conception globale et détaillée	Conception globale, Codage
Support de prototypage	Non	Non	Non	Non	Oui
Cible utilisateur	Concepteurs informaticiens	Concepteurs informaticiens et non informaticiens	Programmeurs	Concepteurs informaticiens	Concepteurs informaticiens et non informaticiens
Difficulté d'apprentissage	Moyenne	Faible	Moyenne	Moyenne	Moyenne
Pouvoir d'expression	Elevé	Elevé	Elevé	Elevé	Faible
Dépendance d'une technologie de développement	Non	Non	Non	Non	Objet
Dépendance d'un langage de développement	Non	Non	Non – système réalisé en Objective-C	Non	Java
Représentation des informations	Multiple – langages à balises	Unique - matrice comportementale	Unique – le creuset	Multiple	Multiple
Gestion du code	Non définie	Non définie	Manuelle	Manuelle	Automatique
Prévisibilité	Non applicable	Non applicable	Non applicable	Non applicable	Oui
Critères d'IHM					
Prise en compte des capacités de l'utilisateur	Non	Oui	Non	Non	Non
Prise en compte de l'environnement extérieur	Oui	Oui	Non	Non	Non
Critères pour la multimodalité					
Généricité par rapport aux modalités d'interaction	Oui	Oui	Oui	Oui	Non – parole et geste
Généricité des mécanismes de fusion	Non définie	Non définie	Oui	Oui	Non
Nature des combinaisons possibles	Complémentarité, Redondance, Equivalence	Non définie	Complémentarité, Redondance, Equivalence, Redondance/ Equivalence	Complémentarité, Redondance, Equivalence, Redondance/ Equivalence	Complémentarité, équivalence, redondance
Temporalité	Parallèle et séquentielle	Non définie	Parallèle et séquentielle	Parallèle et séquentielle	Séquentielle
Critères pour la fusion des données					
Stratégie d'intégration	Non définie	Non définie	Précoce	Différée	Précoce
Proximité temporelle	Non définie	Non définie	Oui	Oui	Non
Complémentarité logique	Non définie	Non définie	Oui	Oui	Oui
Incompatibilité des modalités à combiner	Non définie	Non définie	Oui	Oui	Oui
Ordonnancement des données multimodales	Non défini	Non défini	Non	Non	Oui

Tableau 5.12 : synthèse des caractéristiques des outils (partie 1/2).

Caractéristiques / Outils	Quickset	Embassi	Intuikit	Icon	ISL4WComp	PetShop
Carte d'identité	Carte d'identité	Carte d'identité	Carte d'identité	Carte d'identité	Carte d'identité	Carte d'identité
Couverture du cycle de développement	Conception globale et détaillée, codage	Conception globale et détaillée, codage	Conception détaillée, codage	Conception globale et détaillée, codage	Conception globale et détaillée, codage	Conception globale et détaillée, codage, tests
Support de prototypage	Non	Non	Oui	Oui	Oui	Oui
Cible utilisateur	Concepteurs, informaticiens, programmeurs	Concepteurs, informaticiens, programmeurs	Designers, concepteurs informaticiens, programmeurs	Concepteurs informaticiens et non informaticiens, programmeurs	Concepteurs informaticiens, programmeurs	Concepteurs informaticiens, programmeurs, testeurs
Difficulté d'apprentissage	Elevée	Elevée	Elevée	Faible	Elevée	Elevée
Pouvoir d'expression	Moyen	Moyen	Très élevé	Elevé	Très élevé	Très élevé
Dépendance d'une technologie de développement	Agents	Agents	Non définie	Objets	Composants	Objets, ICO
Dépendance d'un langage de développement	Non	Java, Prolog	Perl, C++	Java	ISL4, C#	Java
Représentation des informations	Multiple	Multiple – XML	Multiple	Multiple	Multiple	Multiple
Gestion du code	Manuelle	Manuelle	Manuelle	Automatique	Semi-automatique (déf. manuelle des patrons de conception en ISL4)	Semi-automatique
Prévisibilité	Non applicable	Non applicable	Non applicable	Oui	Oui	Oui
Critères d'IHM						
Prise en compte des capacités de l'utilisateur	Non	Oui	Non	Non	Non	Non
Prise en compte de l'environnement extérieur	Non	Oui	Non	Non	Oui	Non
Critères pour la multimodalité						
Généricité par rapport aux modalités d'interaction	Non – parole, geste	Non – parole, geste, WIMP	Non	Oui	Oui	Oui
Généricité des mécanismes de fusion	Non	Non	Non	Non	Non	Non
Nature des combinaisons possibles	Complémentarité, Redondance, Equivalence, Redondance/ Equivalence	Complémentarité, Equivalence	Complémentarité, Equivalence	Complémentarité, Redondance, Equivalence	Complémentarité, Equivalence, Redondance, Redondance/ Equivalence	Complémentarité, Equivalence, Redondance, Redondance/ Equivalence
Temporalité	Parallèle et séquentielle	Parallèle et séquentielle	Parallèle et séquentielle	Parallèle et séquentielle	Parallèle et séquentielle	Parallèle et séquentielle
Critères pour la fusion des données						
Stratégie d'intégration	Précoce et différée	Précoce	Non communiquée	Précoce	Précoce	Non définie
Proximité temporelle	Oui	Oui	Non communiquée	Non	Non	Oui
Complémentarité logique	Oui	Oui	Non communiquée	Oui	Oui	Oui
Incompatibilité des modalités à combiner	Oui	Oui	Non communiquée	Oui	Oui	Oui
Ordonnancement des données multimodales	Non	Non	Non communiqué	Non	Non défini	Non défini

Tableau 5.13 : synthèse des caractéristiques des outils (partie 2/2).

CHAPITRE 6
Outil ICARE, les principes de l'approche à composants

Face au constat du chapitre 5, notre objectif est la définition d'un outil pour la conception et le développement d'interfaces multimodales qui répondent aux requis identifiés.

ICARE, pour Interaction Complémentarité Assignation Redondance Equivalence, est un outil de conception et de développement pour les interfaces multimodales qui, en intégrant explicitement les concepts définis par notre modèle conceptuel au chapitre 4, fournit les fonctionnalités manquantes des outils étudiés au chapitre 5. Pour cela, ICARE offre plusieurs types de composants qui, une fois assemblés, définissent les modalités d'interaction et leurs différents usages. Actuellement, ICARE propose une méthodologie et l'outillage couvrant les phases de conception et de codage. Il est à noter que la couverture des phases de tests est aujourd'hui à l'étude (cf. annexe 3 [Serrano 2006] et annexe 4 [Jourde 2006] [Madani 2005]). Nous présentons ICARE selon deux niveaux d'abstraction, car l'outil se compose d'un modèle conceptuel à composants indépendant du mode d'implémentation. Nous considérons que l'implémentation peut être effectuée avec n'importe quelle technologie à composants. Le modèle conceptuel d'ICARE fait l'objet de ce chapitre. Nous proposons une implémentation dans le chapitre suivant afin d'opérationnaliser notre modèle et tester complètement notre approche.

Nous présentons, dans ce chapitre, la spécification des composants d'ICARE permettant de définir un système multimodal. En accord avec le méta-modèle présenté au chapitre 4, ICARE propose deux catégories de composants : les composants élémentaires, qui correspondent aux modalités d'interaction, et les composants de composition, qui représentent les unités de fusion constituant le cœur de notre mécanisme de fusion. Ce chapitre décrit précisément les différents types de composants que l'on retrouve dans ICARE et définit les règles relatives aux combinaisons multimodales. Ensuite, un schéma général d'utilisation des composants pour un système informatique est donné et nous étudions comment l'architecture de composition s'intègre dans l'architecture globale du système cible. Nous débutons ce chapitre par un rappel sur les technologies à composants.

1 Technologies à composants

Depuis le début des années 90 et notamment avec l'introduction des composants COM dans la programmation Windows avec O.L.E [Williams 1994], les industriels du logiciel ont fait émerger une nouvelle approche de développement logiciel : le développement de systèmes informatiques à base de composants. Depuis longtemps, les fabricants de matériel électrique, électronique et mécanique utilisent des composants pour leurs produits. Ces composants sont connectés entre eux et leur assemblage permet d'obtenir un produit final complet. Le terme et l'approche ont été exploités en Informatique et toute grande entreprise, comme Sun ou Microsoft, recherchent, développent et proposent des technologies pour mettre en œuvre des composants logiciels. Comme les autres industries telles que celles de l'électronique ou de l'automobile, cette approche permet une véritable industrialisation et commercialisation du développement logiciel en intégrant les différents acteurs du cycle de vie.

Depuis 2000, les études académiques sur les composants logiciels se sont multipliées et ont influencé les entreprises du logiciel. Depuis, le monde informatique (académique et industriel) a multiplié le nombre de modèles basés sur les composants logiciels. Ainsi, il n'existe pas à ce jour de consensus sur la programmation basée sur les composants : différentes définitions sont proposées selon le domaine et le métier exercé. La section suivante propose une revue terminologique qui précise notamment les intérêts qui nous ont poussés à adopter une telle approche.

1.1 Terminologie

Nous considérons un composant comme "une unité réutilisable de composition et de déploiement accessible à travers des interfaces" [Szyperski 1998]. Mais d'autres concepts sont mis en jeu et plusieurs termes sont employés lorsque l'on parle de composants. Nous retrouvons dans [Bass 2000] une définition générale des termes "composant", "technologie à composants", "développement basé sur le composant" et "ingénierie du logiciel basée sur le composant" :

- "*Un **composant** logiciel est une implémentation de fonctionnalités. Le composant est réutilisable dans différentes applications et il est accessible via une interface de développement logiciel. Il peut, mais ce n'est pas une nécessité, être vendu comme un produit commercial. Un composant logiciel est généralement implémenté par et pour une technologie particulière*".[1]

[1] "*A software component is an implementation, in software, of some functionality. It is reused as-is in different applications, and accessed via an application-programming interface. It may, but need not be, sold as a commercial product. A software component is generally implemented by and for a particular component technology.*" [Bass 2000]

- "*Une **technologie à composants** logiciels inclut un logiciel qui fournit un environnement d'exécution des composants logiciels (appelé aussi plate-forme à composants) et d'autres outils pour la conception, l'assemblage ou le déploiement des composants ou des applications construites avec des composants*".[2]
- "*Le **développement à base de composants** implique des étapes de conception et d'implémentation des composants logiciels, ainsi que d'assemblage des systèmes construits avec des composants logiciels et de déploiement des systèmes assemblés dans les environnements cibles*".[3]
- " *L'**ingénierie du logiciel à base de composants** inclut les pratiques nécessaires au développement à base de composants de façon systématique afin de construire des systèmes qui ont des propriétés prévisibles*".[4]

De ces définitions, nous retenons notamment que le cycle de vie d'un composant est véritablement pris en compte avec une distinction très nette entre la conception, la construction, l'assemblage et le déploiement des composants. Comme le montre la Figure 6.1, différents acteurs (concepteur, programmeur, assembleur, etc.) interviennent pour la mise en place d'une application. Chaque acteur n'a

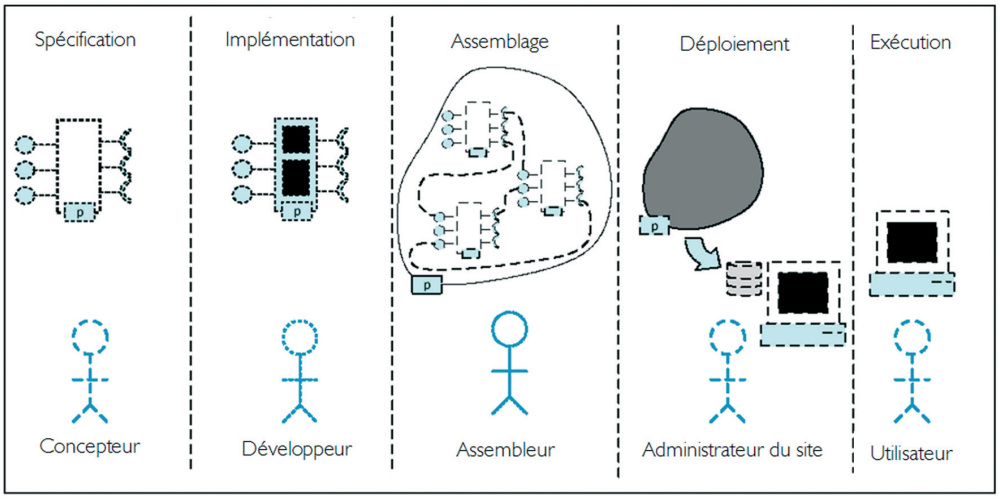


Figure 6.1 : prise en compte du processus de développement [Favre 2003].

[2] "*Software component technology includes the software that provides a runtime environment for software components (sometimes called a component framework) as well as any other tools useful in designing, building, combining, or deploying components or applications built from components.*" [Bass 2000]

[3] "*Component-based development (CBD) involves the technical steps for designing and implementing software components, assembling systems from pre-built software components, and deploying assembled systems into their target environments.*" [Bass 2000]

[4] "*Component-based software engineering (CBSE) involves the practices needed to perform CBD in a repeatable way to build systems that have predictable properties.*" [Bass 2000]

néanmoins pas besoin de connaître le travail de l'autre. Par exemple, l'assembleur n'a pas besoin de connaître comment a été programmé un composant.

Les différentes approches existantes fournissent un processus de développement, un modèle de composant, un modèle d'implémentation, un ensemble d'outils (pour l'assemblage, le test, etc.), un ensemble d'infrastructures (pour l'exécution, l'assemblage, les tests, etc.) et une documentation.

1.2 Modèle à composants

Malgré la diversité des modèles utilisés pour les technologies à composants, un début de normalisation a vu le jour au sein de l'OMG avec la spécification Corba3 [Corba 2002], utilisée pour les CCM (Corba Component Model). Cette norme correspond globalement à la description des diverses technologies à composants existantes et fournit ainsi un cadre général à la compréhension globale de notre approche à composants. Dans ce modèle, un composant est "*une unité logicielle qui encapsule une partie donnée et une partie logique de fonctionnement et qui définit des connexions (interfaces) pour la communication. Le composant est conçu pour être réutilisé dans le développement d'applications avec ou sans personnalisation de ce composant*"[5]. Pour interagir avec un composant, quatre ports ont été identifiés : les facettes, les réceptacles, les sources d'événements et les puits d'événements.

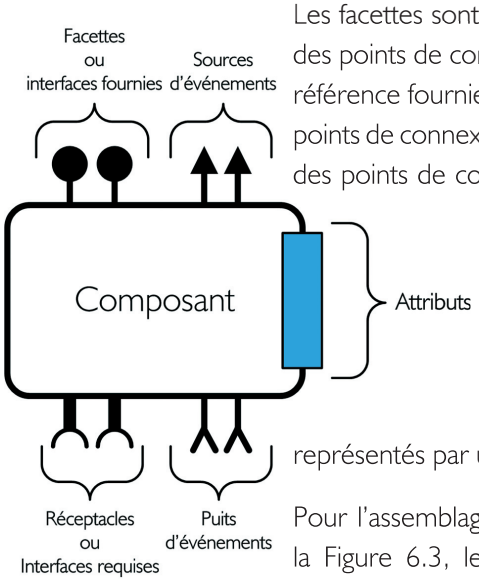


Figure 6.2 : interfaces d'un composant.

Les facettes sont les interfaces fournies par le composant. Les réceptacles sont des points de connexion qui décrivent la capacité d'un composant à utiliser une référence fournie par un autre composant. Les sources d'événements sont des points de connexion qui émettent des événements. Les puits d'événements sont des points de connexion qui reçoivent des événements. Le concept d'attribut est aussi utilisé pour désigner les primitives permettant de configurer un composant. Les implémentations des facettes (interfaces fournies) sont encapsulées dans le composant. La structure interne du composant est opaque (boîte noire). La Figure 6.2 illustre la vue externe d'un composant. Par la suite et pour des raisons de simplification, les composants sont représentés par un simple rectangle dans les figures suivantes de ce mémoire.

Pour l'assemblage, le concept de connecteur est abordé. Comme le montre la Figure 6.3, les composants sont assemblés à l'aide de connecteurs. Les

[5] "*A self-contained unit of software code consisting of its own data and logic, with well-defined connections or interfaces exposed for communication. It is designed for repeated use in developing applications, either with or without customization*". [Corba 2002]

connecteurs peuvent être simples (pipe, appel de procédure, etc.) ou plus complexes (protocole client-serveur, lien à une base de données, etc.) [Barbier 2002].

Les composants doivent être configurables afin de permettre leur paramétrage au moment de l'assemblage. De plus, il peut être utile que les composants soient capables, durant leur exécution, de s'adapter dynamiquement aux modifications de leur environnement.

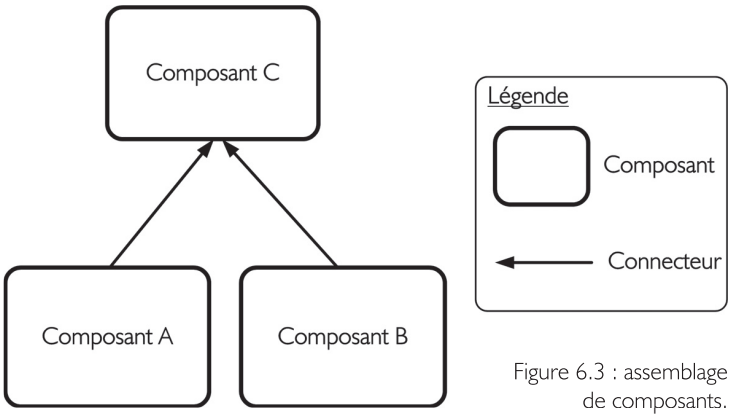


Figure 6.3 : assemblage de composants.

1.2.1 Intérêts

- Les composants sont vus comme des boîtes noires connectées entre elles. L'interface du composant (vue externe) est séparée de son implémentation (vue interne). Ainsi, une boîte noire peut facilement être remplacée par une autre, à condition qu'elles présentent des interfaces semblables.
- Cette approche permet de construire des systèmes informatiques par composition de briques de base configurables (composants élémentaires).
- La réutilisation des composants permet de réduire les coûts et la durée du développement (appelé aussi Rapid Application Development ou RAD). Cela permet aussi d'avoir une meilleure réactivité pour la fabrication d'un nouveau logiciel.
- L'indépendance des composants permet de faciliter leur déploiement et leur maintenance.
- Le processus de développement et les acteurs qui interviennent sont explicitement pris en compte. En effet, les phases de conception, d'implémentation, d'assemblage, de déploiement et d'utilisation sont clairement identifiées dans la définition de la technologie à composants.

1.2.2 Inconvénients

- Pour utiliser des composants hétérogènes, il est nécessaire d'utiliser une infrastructure de déploiement et d'assemblage plus complexe que celles existantes aujourd'hui.
- L'absence de standard multiplie le nombre de technologies à composants alors que les concepts généraux sont semblables. Les composants sont

des solutions techniques proposées par les industriels du logiciel pour répondre à des problèmes ponctuels. Les concepts n'ont pas été figés au départ, engendrant une profusion de technologies.

- L'absence de modèles d'architecture dans la programmation basée à composants est un handicap sérieux par rapport aux autres techniques de programmation. Cependant, des modèles conceptuels sont en cours d'établissement.

De notre point de vue, les avantages que présentent les technologies à composants répondent en partie aux requis demandés pour la conception d'un outil adapté à la conception et au développement d'interactions multimodales. Les quelques inconvénients présentés ne représentent pas de véritables obstacles et certains points peuvent être contournés, notamment le problème d'hétérogénéité qui fait l'objet d'un projet européen en cours [OpenInterface 2006].

Ayant motivé notre choix d'une approche à composants, nous détaillons maintenant les composants de l'outil ICARE.

2 Spécifications des composants de l'outil ICARE

L'outil ICARE reprend précisément tous les concepts identifiés dans notre modèle conceptuel de la multimodalité, présenté au chapitre 4. Il n'est donc pas étonnant que certaines informations paraissent redondantes. Pour rappel, la Figure 6.4 présente les différents éléments constituant notre modèle conceptuel. Ces éléments définissent les briques de base qui interviennent dans une interaction multimodale. Il s'agit des dispositifs, des langages d'interaction et ses diverses formes de combinaison. La technologie à composants adoptée présente l'intérêt de transformer directement ces briques de base en composant logiciel. De plus, chaque caractéristique identifiée pour ces éléments est incarnée par un attribut du composant correspondant.

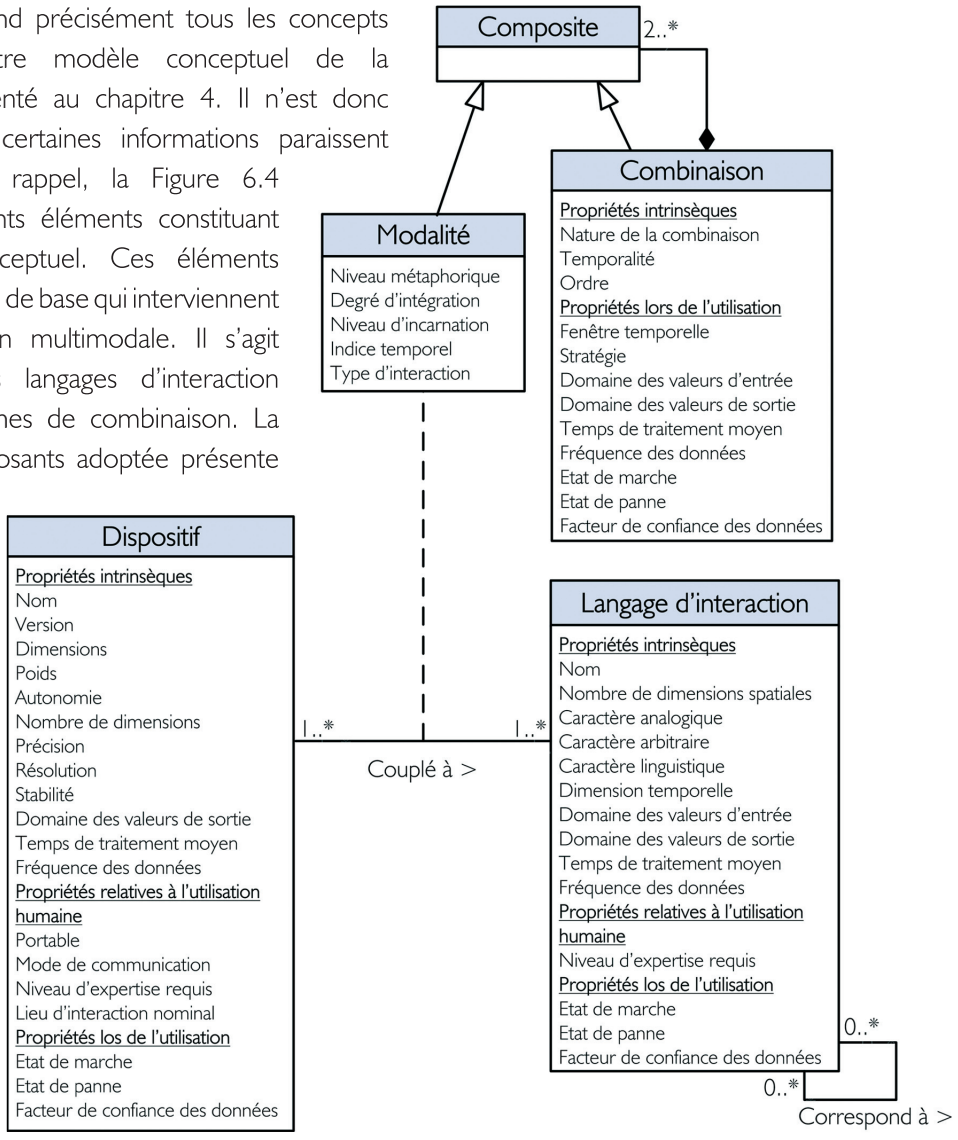


Figure 6.4 : diagramme de classes UML des possibilités multimodales.

2.1 Modalités d'interaction

Comme nous l'avons exposé dans notre modèle conceptuel définit au chapitre 4, l'association d'un Dispositif et d'un Langage d'interaction permet d'obtenir une

modalité d'interaction. Cette décomposition met en évidence les diverses transformations des données que l'ordinateur doit effectuer, suite aux actions physiques de l'utilisateur, et coïncide avec les étapes que l'utilisateur réalise pour spécifier ses intentions au système informatique (cf. Figure 2.13 du chapitre 2).

En adéquation avec notre modèle conceptuel, les modalités d'interaction sont définies par l'association de deux types de composants élémentaires : les composants Dispositifs et les composants Langages d'interaction. Cette section présente en détail ces deux types de composants et leurs attributs génériques. Avec ces attributs, tout dispositif ou langage d'interaction peut être incarné par un composant logiciel réutilisable dans les systèmes interactifs conçus et développés avec l'outil ICARE. Pour chaque type, l'ensemble des caractéristiques de notre modèle conceptuel est retrouvé. Pour simplifier la lecture en évitant des reports au chapitre 4, nous rappelons à chaque fois la définition de la caractéristique. De plus, de nouvelles propriétés de génie logiciel sont ajoutées.

2.1.1 Composant Dispositif

Définition

En accord avec la définition de notre modèle conceptuel sur la multimodalité, un composant Dispositif représente la réalité physique d'entrée du système interactif. Le composant Dispositif peut être un pilote (driver) ou être une couche supplémentaire fournissant les propriétés du dispositif et les méthodes qui permettent de le piloter. Ce composant est celui qui est en contact direct avec l'utilisateur. Par exemple, des composants Dispositifs peuvent contrôler le clavier, la souris ou encore un GPS.

Attributs

Le Tableau 6.1 rassemble l'ensemble des attributs des composants Dispositifs. Bien que la majorité de ces attributs correspondent aux caractéristiques définies dans le modèle conceptuel, il en apparaît de nouveaux. Il s'agit des attributs utiles au génie logiciel afin de pouvoir maintenir et suivre l'évolution des composants. Ainsi, nous identifions les composants par un nom (différent du nom du dispositif qu'il incarne), un numéro de version (différent de la version du dispositif qu'il incarne) et le nom de son créateur. De plus, nous ajoutons un historique qui sauvegarde les données traitées et transmises par le dispositif.

Pour chaque attribut et lorsque cela est possible, nous donnons son nom, sa description, son type et son domaine de valeur.

Nom de l'attribut	Description	Type	Domaine de valeur
Nom du composant	Cet attribut permet d'identifier le composant. Il doit être représentatif du dispositif qu'il représente (par exemple pour la souris, le nom peut être "D_souris").	Chaîne de caractères	D_dispositifX
Version du composant	Il s'agit du numéro de version du composant. Il est composé de deux chiffres dont le premier représente les versions stables. Par exemple, 1.0 représente une première version stable du composant.	Chaîne de caractères	entier.entier
Nom du créateur	Il s'agit du nom du créateur permettant de retrouver l'auteur de ce composant.	Chaîne de caractères	Nomducreateur
Nom du dispositif	Cet attribut essentiel permet d'identifier sans ambigüité le dispositif sous la forme d'un texte.	Chaîne de caractères	Nomdudispositif
Version du dispositif	Cet attribut est une description textuelle de la version courante du dispositif. Elle permet de renseigner du statut actuel du dispositif physique, des mises à jour, etc.	Chaîne de caractères	Versiondudispositif
Dimensions	Cet attribut renseigne de la forme et de la taille du dispositif. Le domaine de valeur dépend du dispositif. Cependant, la plupart des dispositifs peuvent être décrits par leur largeur / hauteur / profondeur, données en centimètres.	—	—
Poids	Cet attribut définit la masse du dispositif. Il doit être spécifié en kilogrammes.	Nombre rationnel	[0, ∞[
Autonomie	Cet attribut détermine le temps d'utilisation du dispositif. Les valeurs peuvent être mesurées en heures. Une valeur infinie indique que le dispositif n'a pas besoin d'alimentation électrique comme un microphone.	Nombre rationnel	[0, ∞[
Nombre de dimensions	Cet attribut représente le nombre de degrés de liberté du dispositif comme identifié dans l'espace de [Buxton 83]. Par exemple, le joystick a trois degrés de liberté.	Entier	[0, ∞[
Précision	Cet attribut mesure l'exactitude de l'information propagée par les actions de l'utilisateur. Le domaine de valeur dépend ici du dispositif en question. Par exemple, la précision d'un GPS est au mètre près.	—	—
Résolution	Cet attribut désigne la plus petite variation perceptible de la grandeur à mesurer dans des conditions de mesures données [Lachenal 2004]. Le domaine de valeur dépend du dispositif en question.	—	—

Nom de l'attribut	Description	Type	Domaine de valeur
Stabilité	Cet attribut définit si, en l'absence d'action de la part de l'utilisateur, la grandeur mesurée ne varie pas [Lachenal 2004]. Par exemple, la souris est stable, le GPS ne l'est pas.	Booléen	{vrai, faux}
Domaine des valeurs de sortie	Le domaine des valeurs de sortie définit la nature des données transmises par le dispositif. Son domaine de valeur dépend du dispositif. Par exemple, un capteur d'orientation 3D peut fournir trois angles en degré dans l'intervalle [0…360[. Cette propriété est à mettre en correspondance avec le domaine des valeurs d'entrée des langages d'interaction afin de savoir si le dispositif peut être couplé avec un langage d'interaction particulier.	—	—
Temps de traitement moyen du dispositif	Cet attribut est une estampille de temps en millisecondes qui représente le temps moyen que met le dispositif pour traiter une donnée.	Entier	[0, ∞[
Fréquence des données	Cet attribut renseigne sur la fréquence de transmission des données par le dispositif. Par exemple, un GPS fournit la position d'un utilisateur toutes les n millisecondes (valeur continue) alors qu'un clavier transmet uniquement des données lorsqu'un utilisateur actionne les touches (valeur discrète).	Chaîne de caractères	{continue, discrète}
Portable	En relation avec les attributs de dimensions et de poids, cette propriété définie si le dispositif est portable par l'utilisateur ou non.	Booléen	{vrai, faux}
Mode de communication	Cet attribut se rapporte aux organes utilisés par un utilisateur lorsqu'il communique avec le système informatique [Bellik 1995]. Trois valeurs sont possibles : le mode "gestuel" (mains, bras, tête, visage, etc.), "oral" (cordes vocales, langues, etc.) ou "mental" (cerveau). La majorité des usages de dispositifs se fait par le geste, comme avec la souris, le clavier ou encore un joystick. L'utilisation de dispositifs comme le microphone se caractérise par le mode oral. Enfin, le mode mental est disponible aujourd'hui grâce à des dispositifs qui analysent l'activité du cerveau.	Chaîne de caractères	{gestuel, oral, mental}
Niveau d'expertise requis	Cet attribut permet d'indiquer la difficulté pour un utilisateur d'utiliser le dispositif. Trois valeurs sont possibles et suivent les niveaux d'expertise identifiés par [Rasmussen 1986] : "expert", "intermédiaire" et "novice". La spécification de la valeur de cette propriété doit prendre en compte le temps d'apprentissage demandé pour utiliser de façon optimale le dispositif. Par exemple, le niveau d'expertise d'une souris est novice alors que le joystick demande un niveau d'expertise intermédiaire.	Chaîne de caractères	{expert, intermédiaire, novice}

Nom de l'attribut	Description	Type	Domaine de valeur
Lieu d'interaction nominal	Le lieu d'interaction est le lieu géographique où l'utilisateur doit porter son attention pour pouvoir émettre des données [Dubois 01]. Cette caractéristique étant dynamique, le lieu d'interaction nominal renseigne l'endroit où l'utilisateur utilise le dispositif dans des conditions normales. Son domaine de valeur peut être défini par quatre zones selon sa distance par rapport à l'utilisateur [Trevisan 2004] : la "zone centrale" de 0 à 45 cm, la "zone personnelle" de 46 à 120 cm, la "zone sociale" de 121 à 360 cm et la "zone publique" au-delà de 360 cm. Par exemple, le lieu d'interaction nominal d'une souris est la zone centrale.	Chaîne de caractères	{central, personnel, social, public}
Etat de marche	Cet attribut permet de connaître l'état courant de fonctionnement du dispositif. Etant maintenant lié au fonctionnement du composant, cet attribut concerne alors l'état de marche du composant. Deux valeurs sont possibles, il peut être en marche (valeur "vrai") ou à l'arrêt (valeur "faux").	Booléen	{vrai, faux}
Etat de panne	Il est parfois possible de détecter un dysfonctionnement d'un dispositif. Le dispositif continue de fonctionner mais avec des erreurs potentielles. Etant maintenant lié au fonctionnement du composant, cet attribut concerne alors l'état de panne du composant. Deux valeurs sont possibles selon s'il y a panne (valeur "vrai") ou non (valeur "faux").	Booléen	{vrai, faux}
Facteurs de confiance des données	Chacune des données fournies par le dispositif peut être caractérisée par un facteur de confiance définissant la pertinence et la crédibilité de l'information générée par le dispositif. Le facteur de confiance est dynamique et noté de 0 à 100, 0 représentant le facteur le plus bas et 100 le facteur le plus élevé.	Tableau d'entiers	[0, 100]
Historique	Il s'agit de sauvegarder l'ensemble des données traitées et transmises par le composant.	Liste	—

Tableau 6.1 : attributs du composant Dispositif.

Nous rappelons que cette liste forme une base commune pour les composants Dispositifs, qui peut évoluer pour intégrer des attributs supplémentaires dépendant, par exemple, des contraintes algorithmiques propres à chaque dispositif.

2.1.2 Composant Langage d'interaction

Définition

Ce composant incarne le langage d'interaction de notre modèle conceptuel. Il transforme les données provenant des composants Dispositifs en une forme

idéale et compréhensible par le reste du système informatique. Par exemple, les angles en degré qui informent de la position de la tête d'un utilisateur constituent un langage d'interaction sous la forme d'un triplet de trois valeurs comprises entre –180 et 180 degrés.

Attributs

Le Tableau 6.2 rassemble les attributs des composants Langages d'interaction. La plupart d'entre eux correspondent aux caractéristiques définies dans notre modèle conceptuel (cf. chapitre 4). Néanmoins et comme pour les composants Dispositifs, de nouveaux attributs sont identifiés. Il s'agit tout d'abord d'attributs utiles au génie logiciel afin de pouvoir maintenir correctement et suivre l'évolution des composants. Ainsi, nous identifions les composants par un nom (différent du nom du langage d'interaction qu'il incarne), un numéro de version et le nom de son créateur. Comme pour le composant Dispositif, un historique sauvegarde l'ensemble des données traitées et transmises par le composant Langage d'interaction.

Nom de l'attribut	Description	Type	Domaine de valeur
Nom du composant	Cet attribut permet d'identifier le composant. Il doit être représentatif du langage d'interaction qu'il représente.	Chaîne de caractères	L_langageX
Version du composant	Il s'agit du numéro de version du composant. Il est composé de deux chiffres dont le premier représente les versions stables. Par exemple, 1.0 représente une première version stable du composant.	Chaîne de caractères	entier.entier
Nom du créateur	Il s'agit du nom du créateur permettant de retrouver l'auteur de ce composant.	Chaîne de caractères	Nomducreateur
Nom du langage d'interaction	Cet attribut essentiel permet d'identifier sans ambigüité le langage d'interaction sous la forme d'un texte.	Chaîne de caractères	Nomdulangage
Nombre de dimensions spatiales	Le langage d'interaction est représentable par une ou plusieurs dimensions spatiales [Vernier 2001]. Par exemple, la position d'un utilisateur en coordonnées WGS84 (World Geodetic System 1984 : système de référence utilisé par les GPS pour les positions au voisinage de la Terre) compte 3 dimensions. Le domaine de valeur est de 0 à l'infini. En comparant le nombre de dimensions du langage d'interaction avec celui du dispositif auquel il est couplé, l'adéquation avec le dispositif peut être définie (cf. chapitre 4).	Entier	[0, ∞[

Nom de l'attribut	Description	Type	Domaine de valeur
Caractère analogique	Cet attribut repose sur la notion de ressemblance avec la réalité [Bernsen 1994]. Le langage d'interaction est analogique quand il repose sur une métaphore du monde réel. Comme exemple, un langage d'interaction qui transmet les mots reconnus vocalement par une chaîne de caractères équivalente est analogique. A l'opposé, s'il transforme ses mots en un code chiffré, il est non-analogique. Cette propriété aide à la spécification du niveau métaphorique global de la modalité (cf. chapitre 4). En effet, le caractère analogique du langage d'interaction contribue à définir la notion de nom et de verbe du niveau métaphorique (notion de forme et de mouvement) si les actions système qu'il transmet se rapprochent de la spécification de ces mêmes actions lors d'une spécification dans le monde réel. Il est alors possible de déterminer si cette notion de forme et de mouvement, en adéquation avec l'utilisation du dispositif, est prise en compte.	Booléen	{vrai, faux}
Caractère arbitraire	Cet attribut reflète la nature du langage d'interaction choisi en fonction de ce que l'on souhaite représenter [Bernsen 1994]. Le langage d'interaction est non arbitraire quand il n'y a pas de correspondance sociale ou cognitive entre le signifiant et le signifié. Un langage d'interaction non-arbitraire consiste à produire, par exemple, comme pour le GPS, un système de coordonnées pour la localisation sur terre se basant sur la latitude, la longitude et la hauteur par rapport à la surface de la terre. Au contraire, l'utilisation des touches clavier dans les jeux vidéo est arbitraire, comme lorsque l'on presse la barre d'espace pour faire freiner une voiture.	Booléen	{vrai, faux}
Caractère linguistique	Identifié par [Bernsen 1994], cet attribut définit si le langage d'interaction se rapporte à des entités distinctives de sens (phonèmes) permettant de définir des unités signifiantes (morphèmes et lexèmes). Par d'autres mots, il s'agit de déterminer si le langage d'interaction est défini par une phonologie, une morphologie et une syntaxe. Par exemple, toutes les langues naturelles ou pseudo-naturel porte le caractère linguistique Comme exemple contraire, un système de coordonnées en trois dimensions n'a pas de caractère linguistique.	Booléen	{vrai, faux}
Dimension temporelle	Cet attribut indique si le langage d'interaction ne peut s'exprimer que par l'écoulement du temps [Bernsen 1994]. Par exemple, la dimension temporelle de coordonnées de position 3D est statique. Au contraire, la dimension temporelle de la reconnaissance de geste est dynamique.	Chaîne de caractères	{statique, dynamique}

Nom de l'attribut	Description	Type	Domaine de valeur
Domaine des valeurs d'entrée	Le domaine des valeurs d'entrée définit la nature des données acceptées et traitables par le langage d'interaction. Son domaine de valeur dépend du langage d'interaction. Par exemple, un langage d'interaction qui transforme les signaux de déplacement x et y d'une souris prend en compte deux entiers. Cette propriété est à mettre en correspondance avec le domaine des valeurs de sortie des dispositifs afin de savoir si le langage d'interaction peut être couplé avec un dispositif particulier.	—	—
Domaine des valeurs de sortie	Le domaine des valeurs de sortie définit la nature des données transmises par le langage d'interaction. Son domaine de valeur dépend du langage d'interaction.	—	—
Temps de traitement moyen	Le temps de traitement moyen est une estampille de temps en millisecondes qui représente le temps moyen que met le langage d'interaction pour traiter une donnée en provenance d'un dispositif.	Entier	[0, ∞[
Fréquence des données	Cet attribut renseigne sur la fréquence de transmission des données par le langage d'interaction. Deux valeurs sont possibles : "continue" ou "discrète". Par exemple, la manipulation directe dans les interfaces WIMP transmet uniquement des données lorsqu'un utilisateur réalise des actions (valeur discrète).	Chaîne de caractères	{continue, discrète}
Niveau d'expertise requis	Il définit la complexité du langage d'interaction. Les trois valeurs sont définies et suivent les niveaux d'expertise identifiés par [Rasmussen 1986] : "expert", "intermédiaire" et "novice". Par exemple, le niveau d'expertise pour la manipulation directe est novice. La spécification de la valeur de cette propriété doit prendre en compte le temps d'apprentissage demandé pour utiliser de façon optimale le langage d'interaction.	Chaîne de caractères	{expert, intermédiaire, novice}
Etat de marche	Cet attribut permet de connaître l'état courant de fonctionnement du composant. Deux valeurs sont possibles, il peut être en marche (valeur "vrai") ou à l'arrêt (valeur "faux").	Booléen	{vrai, faux}
Etat de panne	Il est parfois possible de détecter un dysfonctionnement du composant. Deux valeurs sont possibles selon s'il y a panne (valeur "vrai") ou non (valeur "faux").	Booléen	{vrai, faux}
Facteurs de confiance des données	Chacune des données produites par le langage d'interaction peut être caractérisée par un facteur de confiance définissant la pertinence et la crédibilité de l'information générée. Le facteur de confiance est dynamique et noté de 0 à 100, 0 représentant le facteur le plus bas et 100 le facteur le plus élevé.	Tableau d'entiers	[0, 100]
Historique	Il s'agit de sauvegarder l'ensemble des données traitées et transmises par le composant.	Liste	—

Tableau 6.2 : attributs des composants Systèmes représentationnels.

Nous rappelons que cette liste forme une base commune pour les composants Langages d'interaction, qui peut évoluer pou intégrer des attributs supplémentaires.

2.1.3 Mise en œuvre d'une modalité d'interaction d'entrée

D'après notre modèle conceptuel, une modalité est un couple (Dispositif, Langage d'interaction) qui se traduit dans ICARE par un simple assemblage d'un composant Dispositif avec un composant Langage d'interaction. La Figure 6.5 montre un exemple d'assemblage entre un composant Dispositif GPS et un composant Langage d'interaction produisant des données selon le système WGS84.

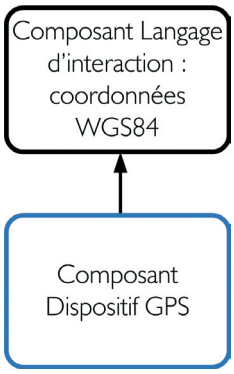


Figure 6.5 : une modalité d'interaction d'entrée dans ICARE.

2.2 Combinaisons de modalités d'interaction

Alors que chaque modalité d'interaction peut être utilisée indépendamment d'un système multimodal, lorsque plusieurs modalités d'interaction sont disponibles dans le même système interactif, leur usage combiné devient possible. Les combinaisons de modalités d'interaction ouvrent alors un vaste champ de possibilités pour la conception du système interactif que nous étudions au regard des propriétés CARE [Coutaz 1994], également identifié dans l'espace TYCOON [Martin 2002]. Nous avons identifié quatre natures différentes pour les combinaisons : la complémentarité, la redondance, l'équivalence et la redondance/équivalence. Il est remarquable que l'assignation de CARE n'est pas été utilisée, à juste titre, car l'assignation ne traduit pas une forme de combinaison mais l'absence de choix des modalités d'interaction, au contraire de l'équivalence qui exprime la possibilité de choix. L'assignation et l'équivalence ne s'incarnent pas comme composants dans ICARE car aucun traitement algorithmique particulier n'est exigé pour exprimer ces deux propriétés. Elles peuvent cependant être explicites dans un assemblage de composants. Comme le montre la Figure 6.6, l'assignation est représentée par un simple lien entre deux composants. Un composant A est relié à un composant B impliquant que A est assigné à B.

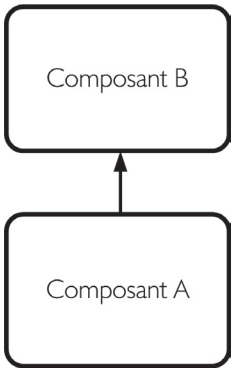


Figure 6.6 : un composant A assigné à un composant B dans l'outil ICARE.

Pour l'équivalence, la Figure 6.7 montre cette relation de choix quand plusieurs composants (2 à n composants) sont liés au même composant.

Nous proposons donc seulement trois composants différents pour la combinaison : le composant Complémentaire, le composant Redondance et le composant Redondance/Equivalence.

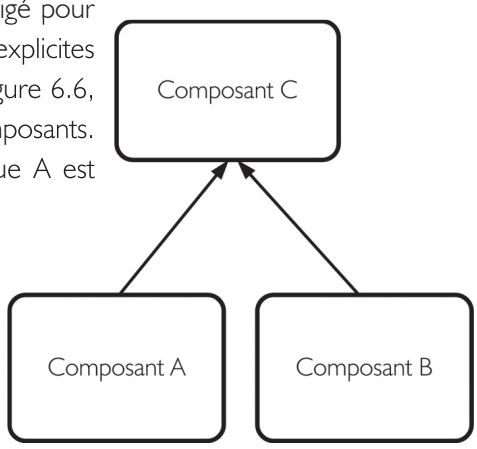


Figure 6.7 : deux composants équivalents (A et B) pour un composant C dans l'outil ICARE.

Bien qu'ils présentent les mêmes caractéristiques (fenêtre temporelle, ordre, etc.) identifiées dans une seule classe (cf. Figure 6.4), cette décomposition en trois composants distincts permet toutefois de proposer les trois natures de combinaison correspondantes de façon claire et précise, répondant ainsi à l'un de nos objectifs concernant la simplification d'utilisation par des concepteurs, aussi bien informaticiens que non informaticiens, et par des développeurs.

Dans ce contexte, nous commençons par rappeler les caractéristiques communes à chacune des combinaisons qui se traduisent en attributs des composants. Ensuite, les trois composants de combinaison sont étudiés et illustrés par des exemples d'utilisation, en commençant par la complémentarité, puis la redondance et enfin la redondance/équivalence.

Figure 6.8 : un composant de combinaison et ses deux ports permettant de combiner les données de deux composants.

2.2.1 Attributs communs des composants de combinaison

Les trois composants de combinaison (Complémentarité, Redondance, Redondance/Equivalence) sont complètement indépendants des modalités d'interaction et des données qu'ils combinent. Ils décrivent les mécanismes de fusion de données fournies par 2 à n composants [Bouchet 2004c]. La Figure 6.8 montre un composant de combinaison et ses deux ports, auxquels doivent être reliés les composants qui produisent les données à combiner.

Ces mécanismes de combinaison sont définis par des attributs que nous présentons dans le Tableau 6.3. Il s'agit principalement des caractéristiques que nous avons identifiées dans notre modèle conceptuel, permettant de couvrir tous les cas de combinaisons en fonction du moment de l'émission des données multimodales. Une fenêtre temporelle gère ainsi les données reçues. Cette fenêtre définit la proximité temporelle (+/-Δt) des données à combiner. Par exemple, deux données d1 et d2 sont combinées si leur estampille de temps, respectivement t1 et t2 sont proches dans le temps :

Proximité temporelle (t1, t2) est satisfait si : t2 Δ [t1-Δt, t1+Δt].

De plus, il est aussi possible, en définissant l'attribut Ordre à vrai, de ne combiner les données qui ont été spécifiées dans un ordre spécifique, suite aux actions de l'utilisateur.

Enfin, et comme pour les composants Dispositifs et Langages d'interaction, trois attributs sont ajoutés, permettant le suivi logiciel : le nom du composant, sa version et le nom de son créateur.

Nom de l'attribut	Description	Type	Domaine de valeur
Nom du composant	Cet attribut permet d'identifier le composant. Il doit être représentatif de la combinaison qu'il représente.	Chaîne de caractères	Naturedela- - combinaison
Version du composant	Il s'agit du numéro de version du composant. Il est composé de deux chiffres dont le premier représente les versions stables. Par exemple, 1.0 représente une première version stable du composant.	Chaîne de caractères	entier.entier
Nom du créateur	Il s'agit du nom du créateur permettant de retrouver l'auteur de ce composant.	Chaîne de caractères	Nomducreateur
Temporalité	Cet attribut détermine l'usage temporel qui doit être effectué par l'utilisateur sur l'ensemble des modalités combinées. Quatre valeurs sont possibles : "aucune", "parallèle", "séquentiel" ou "parallèle et séquentiel" [Nigay 1994]. L'utilisation combinée de plusieurs modalités en parallèle est appelé usage synergique alors que leur utilisation séquentielle est qualifié d'usage alterné.	Chaîne de caractères	{aucune, parallèle, séquentiel, parallèle et séquentiel}
DeltaT (Fenêtre temporelle)	Cet attribut permet de définir en millisecondes l'intervalle de temps pendant lequel les informations des modalités d'interaction peuvent être combinées. La fenêtre temporelle constitue la caractéristique principale du mécanisme de fusion des données provenant des différentes modalités d'interaction à combiner. Si la valeur correspond à l'infini, il n'y a pas de contrainte temporelle pour la fusion.	Entier	[0, ∞[
Ordre	Cet attribut définit le besoin d'une succession temporelle précise de l'usage de chacune des modalités d'interaction. La spécification d'un ordre définit la chronologique qui doit être effectuée par l'utilisateur pour l'emploi des modalités d'interaction combinées.	Booléen	{vrai, faux}
Stratégie	Cet attribut définit deux stratégies concernant la fusion d'informations : la stratégie "précoce" ou la stratégie "différée" [IHM 1992]. Le choix de la stratégie permet d'accélérer le traitement de l'interaction (stratégie précoce) ou d'assurer la pertinence des informations avant de les propager (stratégie différée).	Chaîne de caractères	{précoce, différée}
Domaine des valeurs d'entrée	Le domaine des valeurs d'entrée définit la nature des données acceptées et traitables par la combinaison. Son domaine de valeur dépend des modalités d'interaction qu'elle doit combiner.	—	—
Domaine des valeurs de sortie	Le domaine des valeurs de sortie définit la nature des données transmises par la combinaison des modalités d'interaction.	—	—
Temps de traitement moyen	Le temps de traitement moyen est une estampille de temps en millisecondes qui représente le temps moyen que met la fusion des données multimodales selon la nature de la combinaison.	Entier	[0, ∞[

Nom de l'attribut	Description	Type	Domaine de valeur
Fréquence des données	Cet attribut renseigne sur la fréquence de transmission des données. Deux valeurs sont possibles : "continue" ou "discrète".	Chaîne de caractères	{continue, discrète}
Etat de marche	Cet attribut permet de connaître l'état courant de fonctionnement du composant. Deux valeurs sont possibles, il peut être en marche (valeur "vrai") ou à l'arrêt (valeur "faux ").	Booléen	{vrai, faux}
Etat de panne	Il est parfois possible de détecter un dysfonctionnement du composant. Deux valeurs sont possibles selon s'il y a panne (valeur "vrai") ou non (valeur "faux").	Booléen	{vrai, faux}
Facteurs de confiance des données	Chacune des données produites par la combinaison peut être caractérisée par un facteur de confiance définissant la pertinence et la crédibilité de l'information générée. Le facteur de confiance est dynamique et noté de 0 à 100, 0 représentant le facteur le plus bas et 100 le facteur le plus élevé.	Tableau d'entiers	[0, 100]
Historique	Il s'agit de sauvegarder l'ensemble des données traitées et transmises par le composant.	Liste	—

Tableau 6.3 : attributs des composants de combinaison.

2.2.2 Composant Complémentarité

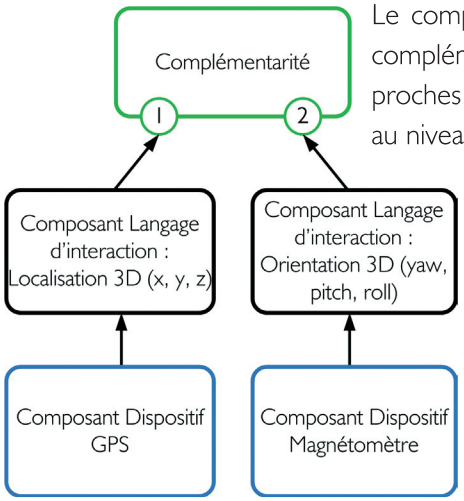


Figure 6.9 : Complémentarité de deux langages d'interaction.

Le composant Complémentarité correspond à la propriété CARE de complémentarité. Il fusionne les données complémentaires qui sont proches dans le temps. Par exemple, considérons une complémentarité au niveau logique entre plusieurs langages d'interaction, pour une tâche donnée : n langages d'interaction sont complémentaires pour la construction d'une expression correspondant à une tâche pouvant être partitionnée en n sous-expressions, où chaque expression est obtenue à partir d'un seul des n langages d'interaction. Pour illustration, la Figure 6.9 montre deux composants Dispositifs équipant un utilisateur, un magnétomètre et un GPS. Le magnétomètre est lié à un composant Langage d'interaction donnant trois angles en radians (yaw, pitch, roll). Le GPS est lié à un composant Langage d'interaction de localisation en trois dimensions comprenant trois coordonnées (x, y, z). Les données de ces deux composants Langages d'interaction sont complémentaires car leur composition permet de connaître l'endroit exact où l'utilisateur regarde.

Avec les attributs définis dans la section précédente, plusieurs algorithmes peuvent être élaborés et permettent de fusionner les données complémentaires. Nous en proposons un qui se base sur la proximité temporelle des données à combiner peuvant être émises parallèlement ou séquentiellement. Nous avons décidé de ne traiter que la stratégie précoce. Une structure commune, appelée structure de fusion, est utilisée pour fusionner les données reçues des composants connectés. Elle contient un emplacement par numéro de port dans lequel sont sauvegardées les données produites par le composant connecté au port correspondant. Si nous reprenons l'exemple de la Figure 6.9, cette structure de fusion est constituée de deux parties (deux ports), l'une qui contient la localisation [x, y, z] et l'autre qui contient l'orientation [yaw, pitch, roll]. La fusion des données réelles est effectuée si les données appartiennent à la même fenêtre temporelle. Une structure de fusion est considérée comme complète quand tous les composants connectés aux différents ports ont transmis des données dans la même fenêtre temporelle. Dans ce cadre, notre mécanisme de combinaison suit un ensemble de règles :

Règle 1 : Quand de nouvelles données proviennent d'un composant, une nouvelle structure de fusion est créée contenant ces nouvelles données, dans l'emplacement correspondant au port sur lequel elles sont transmises.

Règle 2 : Les anciennes structures de fusion sont complétées par les nouvelles données lorsque l'emplacement correspondant au numéro de port est libre et que les données sont dans la même fenêtre temporelle. Si l'attribut Ordre est activé, une vérification est effectuée sur l'estampille de temps des nouvelles données afin d'assurer que l'arrivée des données dans les structures de fusion respecte la chronologie demandée.

Règle 3 : Quand une structure de fusion est complète, elle est envoyée vers le composant suivant. Quand plusieurs structures de fusion sont complètes, le composant propage la structure qui contient les données les plus proches temporellement.

Règle 4 : Quand une structure de fusion est envoyée au composant suivant, toutes les données qu'elle contient sont supprimées des structures de fusion encore présentes dans le composant. Les structures de fusion vides sont supprimées et la règle 2 peut à nouveau être appliquée.

Règle 5 : Les données des structures de fusion incomplètes peuvent être supprimées quand leur durée de vie est expirée. Nous avons choisi arbitrairement de supprimer toutes les données qui présentent une estampille de temps inférieure au temps courant moins deux fois ΔT (fenêtre temporelle).

Ainsi, nous proposons un composant simple pour la complémentarité qui couvre, comme nous le verrons dans la Partie 3 de ce mémoire, la plupart des cas. Il est tout à fait envisageable de définir d'autres mécanismes plus complexes, par exemple un composant Complémentarité qui exploite aussi les facteurs de confiance.

2.2.3 Composant Redondance

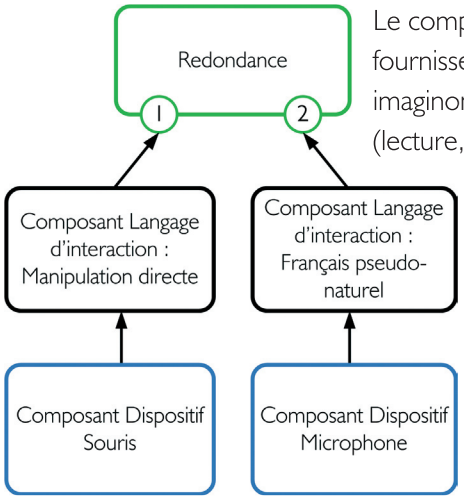


Figure 6.10 : Redondance de deux langages d'interaction.

Le composant Redondance correspond au cas où plusieurs composants fournissent des données redondantes proches dans le temps. Par exemple, imaginons un lecteur de vidéo sur ordinateur qui doit être contrôlé (lecture, stop, etc.) par l'utilisation de la souris (en utilisant les boutons de l'interface graphique du lecteur), en redondance de commandes prononcées vocalement par l'utilisateur ("lecture", "stop", etc.). L'assemblage de composants est présenté à la Figure 6.10. Les deux modalités d'interaction (Souris, Manipulation directe) et (Microphone, Français pseudo-naturel) sont chacune reliée à un port du composant Redondance qui va détecter, selon la fenêtre temporelle définie, si les deux modalités d'interaction ont bien été utilisée pour spécifier la même commande.

Comme pour la gestion de la complémentarité, plusieurs algorithmes peuvent être définis. Celui que nous avons spécifié est proche de celui décrit pour le composant Complémentarité, excepté que les données stockées dans des emplacements différents (ports différents) sont comparées pour déterminer les cas de redondance et ainsi produire une nouvelle fusion de données. Les structures de fusion ne contiennent que des données redondantes. Les cinq règles du composant Complémentarité sont conservées pour l'algorithme du composant Redondance, excepté la structure de fusion qui n'est pas propagée en l'état. Seul un jeu des données redondantes est envoyé au composant suivant. Il s'agit de celui qui présente les meilleurs facteurs de confiance.

2.2.4 Composant Redondance/Equivalence

Le composant Redondance/Equivalence correspond aux deux propriétés de CARE [Coutaz 1994], également identifiée dans [Martin 2002]. Il correspond au composant Redondance dans lequel la redondance peut être optionnelle. Si nous reprenons l'exemple donné précédemment pour le composant Redondance, où l'utilisateur manipule un lecteur vidéo sur ordinateur par la voix ou par l'utilisation de la souris, l'usage redondant pourrait être trouvé trop contraignant. En conservant les mêmes modalités mais en changeant le composant Redondance par un composant Redondance/Equivalence, l'utilisation du lecteur est plus flexible. Par

exemple, pour arrêter la lecture en cours, l'utilisateur peut soit utiliser la souris et appuyer sur le bouton stop, soit utiliser sa voix et prononcer le mot "stop", soit, enfin, appuyer sur le bouton stop et prononcer le mot "stop" dans un intervalle de temps proche. Au final et quelque soit l'usage, une seule commande stop est envoyé au reste du système informatique du lecteur vidéo.

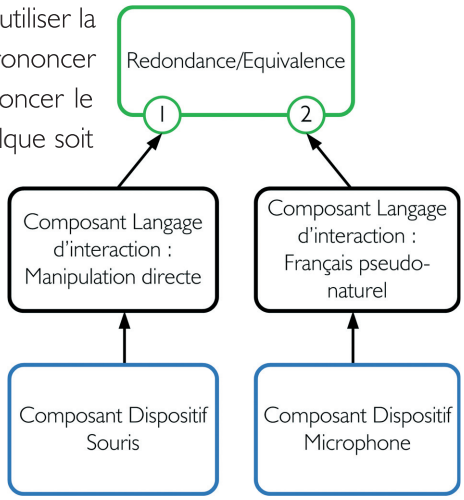


Figure 6.11 : Redondance de deux langages d'interaction.

Comme le montre la Figure 6.12, ce composant n'est pas obligatoire, il simplifie seulement la spécification du diagramme des composants en s'assurant que les commandes de l'utilisateur ne soitent pas transmises plusieurs fois.

Le mécanisme du composant Redondance/Equivalence que nous avons défini est plus complexe que celui du composant Redondance car il intègre deux stratégies différentes : précoce et différée. Alors que nous ne les avions pas prises en compte pour les autres composants de combinaison, la stratégie se révèle être une importante caractéristique de ce composant car, en l'utilisant, les concepteurs laissent à l'utilisateur diverses possibilités d'usage dont le choix est difficilement prédictible. Alors que pour les composants de complémentarité et de redondance, le choix de la stratégie différée aurait surtout permis de pallier à des ambigüités liées aux performances du système, le composant Redondance/Equivalence présente une incertitude sur l'usage équivalent ou redondant des modalités d'interaction par l'utilisateur pouvant être levée en stratégie différée. Ainsi, nous proposons un mécanisme soit efficace, avec la stratégie précoce, soit sûr avec la stratégie différée. En adoptant une stratégie

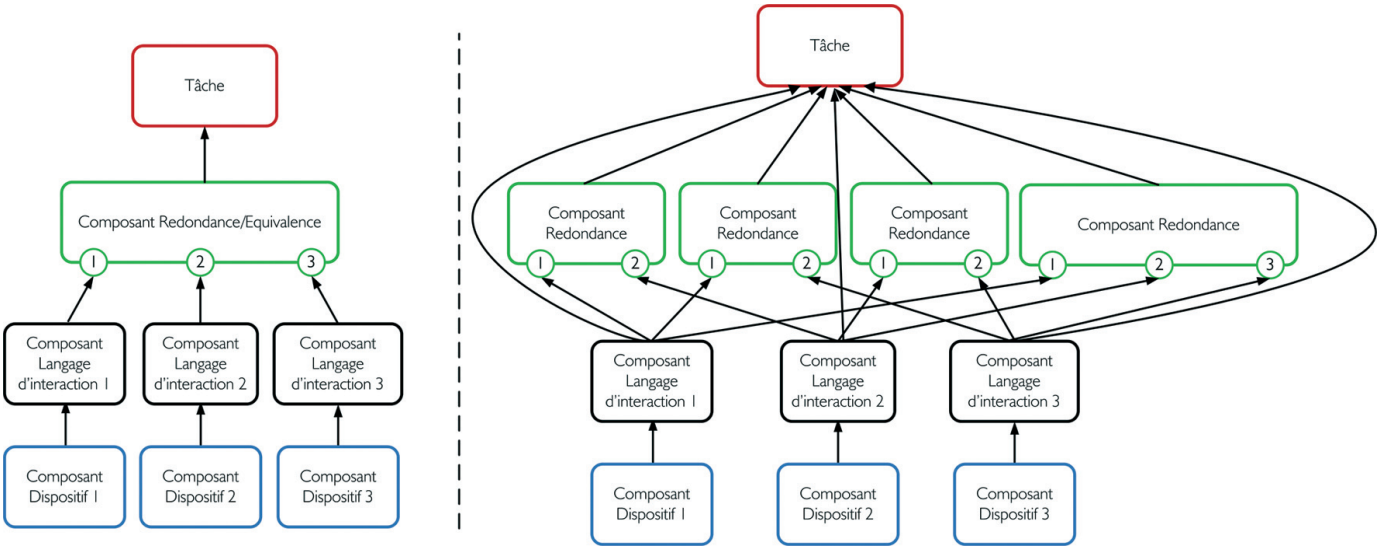


Figure 6.12 : Redondance/Equivalence de trois modalités d'interaction.

précoce, le composant n'attend pas pour propager les données aux composants suivants. Cependant, il démarre une fenêtre temporelle qui permet de détecter les données redondantes qui peuvent être reçues plus tard. Cette approche a l'avantage de fournir un retour immédiat à l'utilisateur. L'inconvénient est que les données propagées sont les premières reçues par le composant et n'ont peut être pas les facteurs de confiance les plus élevés. Au contraire, le mécanisme de la stratégie différée attend la fin de la fenêtre temporelle avant de propager les données. L'avantage de cette stratégie est de toujours garantir que les données propagées ont le plus haut facteur de confiance, comme pour le composant Redondance. Une autre solution consisterait à calculer un nouveau facteur de confiance plus élevé lors des cas de redondance.

Les différents composants de l'outil ICARE étant exposés, nous décrivons leur mode de communication.

3 Communication entre composants

Les technologies à composants s'articulent autour d'une communication événementielle permettant une totale indépendance entre les composants. Ainsi, notre travail a consisté à définir une structure événementielle cohérente. Pour cela, les mécanismes génériques des composants de combinaison ont grandement orienté nos choix. Ne connaissant pas les structures de données qu'ils auront à combiner, nous avons choisi de définir une structure générique à n'importe quelles données et exploitable à n'importe quel niveau d'abstraction. Nous avons donc choisi de définir tous les événements entre les différents composants par une structure unique. Cette structure, appelée ICAREEvent, se caractérise par des attributs facilitant le traitement des événements par des mécanismes présentant un aspect temporel fort. Le Tableau 6.4 présente l'ensemble des attributs des ICAREEvent.

Nom de l'attribut	Description	Type	Domaine de valeur
Données	Il s'agit des données produites par un composant.	Liste	—
Facteurs de confiance	Chaque donnée fournie par un composant est associé à un facteur de confiance. Le facteur de confiance permet d'évaluer la pertinence et la crédibilité de l'information générée et peut être utilisé dans divers buts comme le filtrage de données non sûres, etc. Le facteur de confiance est dynamique et noté de 0 à 100, 0 représente le facteur le plus bas et 100 le facteur le plus élevé.	Tableau d'entiers	[0, 100]
Estampilles de temps courant des données	Chaque donnée de l'événement est associée à un temps correspondant à son dernier traitement par un composant. Le temps doit être calculé en fonction du temps système généralement défini en millisecondes.	Tableau d'entiers	[0, ∞[
Estampilles du temps initial des données	Chaque donnée de l'événement est associée à un temps correspondant à sa première propagation par un composant. Le temps doit être calculé en fonction du temps système généralement défini en millisecondes.	Tableau d'entiers	[0, ∞[
Moyenne des temps courants	Il s'agit de la moyenne des estampilles de temps courants de chaque donnée de l'événement.	Entier	[0, ∞[
Moyenne des temps initiaux	Il s'agit de la moyenne des estampilles de temps initiaux de chaque donnée de l'événement.	Entier	[0, ∞[

Tableau 6.4 : attributs des événements dans l'outil ICARE.

4 Utilisation dans un système informatique et architecture logicielle

L'outil ICARE présente plusieurs composants pour concevoir et développer le système multimodal en entrée : les composants Dispositifs, les composants Langages d'interaction et les composants de combinaison (Complémentarité, Redondance, Redondance/Equivalence). En fonction des tâches à réaliser, l'assemblage de ces différents composants définit les modalités d'interaction que l'utilisateur a à disposition et la manière dont il peut s'en servir. Pour chaque tâche, un assemblage de composants est défini. Alors que nous verrons des exemples concrets dans la Partie 3 de ce mémoire, la Figure 6.13 montre un assemblage de composants pour un système informatique présentant deux tâches d'interaction d'entrée. Nous observons que les composants formant les modalités d'interaction et leurs combinaisons sont assemblées en fonction des deux tâches du système. Les tâches peuvent se concrétiser sous diverses formes, comme par exemple une simple méthode prenant en paramètre les données d'un ICAREEvent. De plus, les interfaces d'entrée et de sortie partagent un lien très étroit qui demande parfois, lorsque le niveau d'indirection est très faible (cf. section du chapitre 4), une communication directe entre elles. Il est ainsi possible que l'interface de sortie et les composants de l'outil ICARE communiquent directement sans qu'une tâche soit en jeu. Il est à noter que même si aucun requis concernant l'interface de sortie n'est demandé, des travaux sur l'adaptation des composants d'ICARE pour les interfaces multimodales de sortie sont à l'étude dans [Mansoux 2006].

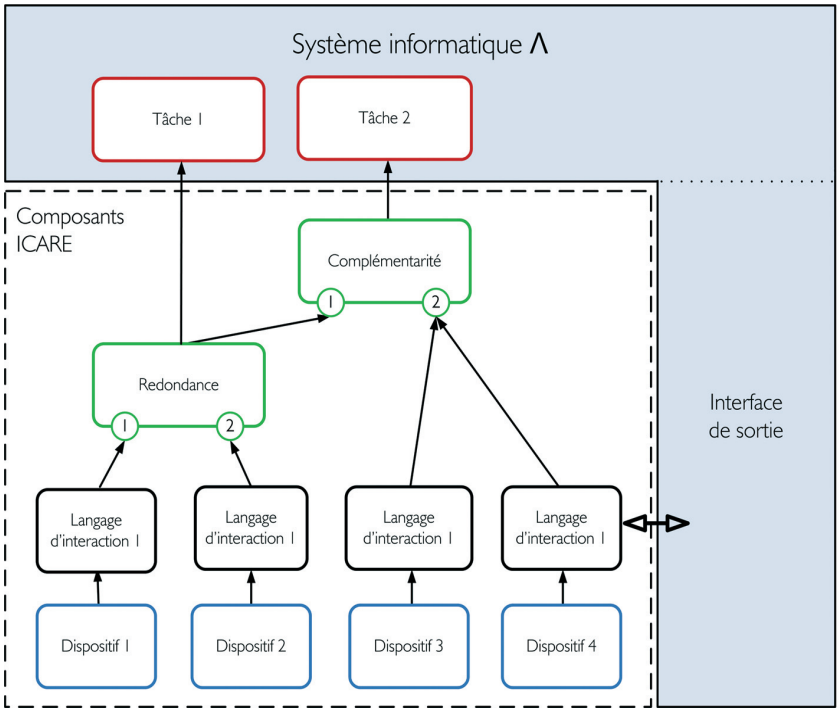


Figure 6.13 : exemple d'utilisation des composants connectés à un système informatique existant.

D'un point de vue architectural, les composants ICARE permettent de distinguer clairement la partie interaction du reste du système informatique, comme dans la plupart des modèles conceptuels d'architecture logicielle. Par exemple, la Figure 6.14 situe les composants ICARE dans une architecture de type ARCH [Bass 1992], dont le pilier de droite rassemble toute la partie interactive avec

l'utilisateur. L'enchaînement des tâches est ici géré par le contrôleur de dialogue situé au centre. Le lien entre l'assemblage et le contrôleur de dialogue se situe donc autour des tâches. Enfin, le pilier de gauche représente les modules propres aux concepts manipulés dans le système informatique.

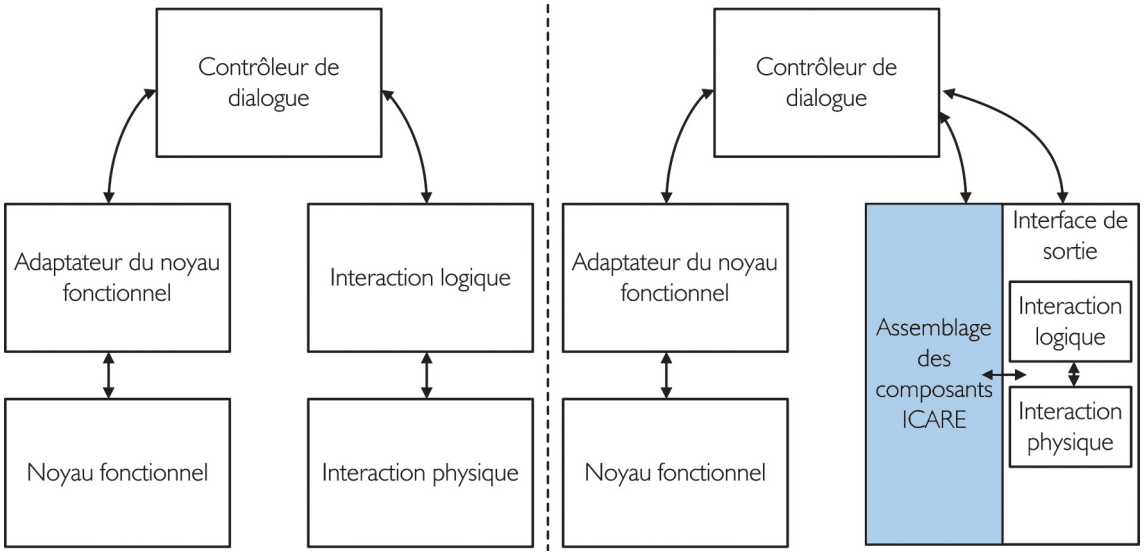


Figure 6.14 : (a) le modèle d'architecture logiciel ARCH, (b) les composants ICARE dans ARCH.

5 Résumé du chapitre 6

L'outil ICARE utilise les avantages des technologies à composants comme solution pour la conception et le développement rapide de systèmes interaction multimodaux. Il fournit plusieurs types de composants permettant de définir les modalités d'interaction et leurs combinaisons au sein du système interactif [Bouchet 2004c]. L'assemblage de ces composants constitue l'interaction en entrée du système multimodal.

L'association d'un composant Dispositif et d'un composant Langage d'interaction forme une modalité d'interaction possible de combiner avec d'autres modalités d'interaction, grâce aux trois composants de combinaison (Complémentarité, Redondance, Redondance/Equivalence). Les composants ICARE sont caractérisés par des attributs génériques aux dispositifs et aux langages d'interaction, permettant d'implémenter de nouveaux composants incarnant les dispositifs et langages d'interaction existants et de préparer leur possible combinaison. Les trois composants génériques de combinaison sont aussi caractérisés par des attributs et nous avons détaillé un fonctionnement simple pour la Complémentarité et la Redondance. Les composants de combinaison sont génériques et permettent de combiner toute modalité d'interaction. De plus, la communication entre composants de l'outil ICARE est permise grâce à une structure unique d'événement exploitable à n'importe quel niveau d'abstraction. Enfin, l'assemblage des composants de l'outil ICARE créé s'intègre parfaitement aux architectures logicielles, distinguant la partie interactive du reste du système informatique.

Dans ce chapitre, nous avons proposé un modèle conceptuel à composant pour l'outil ICARE. Nous en présentons une implémentation au chapitre suivant.

CHAPITRE 7
OUTIL ICARE, UN EXEMPLE D'IMPLÉMENTATION

Présenté dans le chapitre précédent, l'outil ICARE définit un modèle à composants spécialement dédié à la multimodalité. Avec trois composants de combinaison génériques, l'outil propose pour la première fois une solution simple et rapide pour créer les diverses formes de combinaison entre les modalités d'interaction, correspondant aux possibilités d'usage de l'utilisateur. Pour tester notre outil, l'implémentation de la spécification des composants ICARE doit être réalisée. Ce chapitre expose la solution d'implémentation que nous avons mise en œuvre afin de tester notre outil, par la réalisation de plusieurs systèmes interactifs multimodaux.

Bien que le modèle conceptuel de l'outil ICARE soit programmable avec n'importe quelle technologie à composants existante, il nous a fallu faire un choix. Pour ce faire, nous avons étudié les diverses technologies dont nous résumons les principaux traits dans la première section. Parmi elles, la technologie JavaBeans a retenu notre attention et nous développons ensuite les raisons de ce choix. Puis nous exposons nos travaux concernant l'implémentation effective de notre modèle à composants ICARE en de composants utilisables et exécutables. Enfin, nous décrivons l'éditeur graphique permettant la manipulation facile et rapide des composants ICARE en vue d'une conception et d'un développement itératif. Pour rappel, le cadre bleu ci-après résume la définition que nous avons adoptée pour les composants logiciels.

Rappel : Les composants logiciels

Dans notre approche, un composant logiciel est identifié comme un module logiciel autonome qui peut être composé et déployé. La composition peut être faite de façon statique (les composants sont assemblés en une seule étape ; il est impossible d'ajouter ou d'enlever un composant pendant l'exécution) ou de façon dynamique (les composants peuvent se connecter ou se déconnecter de l'application pendant son exécution). Comme présenté au chapitre 6, un composant est une entité encapsulée qui possède une ou plusieurs interfaces fournies (accès aux attributs, méthodes publiques) et qui a besoin d'une ou de plusieurs interfaces (interfaces requises permettant son exécution). Un composant peut être configuré (modification de ses attributs). Les composants communiquent entre eux par des événements (communication événement – réaction). Un même composant peut être réutilisé dans plusieurs systèmes informatiques.

1 Les technologies à composants existantes

Cette section présente succinctement les technologies à base de composants qui sont le plus utilisées et les plus connues dans le développement de systèmes informatiques. Toutes sont distribuées par des organismes tels que Microsoft, Sun, l'OMG (Object Management Group) ou l'OSGi Alliance. La diversité des technologies à composants réside dans le domaine d'application visé.

1.1 Composants de Microsoft

Plusieurs technologies à composants Microsoft exsitent.

Les composants Microsoft COM, DCOM et COM+ [Microsoft 2002] sont utilisés uniquement dans un environnement Windows. Dès 1990, les premiers composants COM de Microsoft sont utilisés pour la réalisation de document composite en intégrant des objets d'une application dans une autre application (comme par exemple l'intégration d'un tableau Excel dans un document Word). Ils améliorent ainsi l'indépendance entre les unités logicielles. La technologie DCOM est une extension de COM pour la distribution des composants sur des machines distantes et la technologie COM+ étend celle de DCOM avec un support pour la persistance et la transaction des données.

Les composants de la plate-forme .NET de Microsoft [Microsoft 2003] sont quant à eux utilisés pour les applications Internet (Web services) et les applications Client-Serveur. Leur objectif consiste à faciliter le développement de ces applications en laissant la possibilité de programmer les composants avec divers langages tels que C#, Visual Basic ou encore C++.

1.2 Composants de Sun Microsystems

Deux technologies à composants sont proposées par Sun Microsystems : les *JavaBeans* et les *Entreprise JavaBeans* (EJB).

Les JavaBeans [Beans 1997] sont destinés au développement des interfaces graphiques classiques des utilisateurs (Graphical User Interface ou GUI). Plusieurs outils de développement graphique les prennent en charge. Dans le développement de GUI en Java, la plupart des environnements de développement proposent de construire l'interface graphique de façon visuelle avec des composants AWT et

SWING. Cette manipulation est possible car les composants SWING et AWT adhèrent aux spécifications JavaBeans. Néanmoins, des composants non graphiques peuvent aussi être des JavaBeans. L'intérêt principal de JavaBeans réside dans cette manipulation visuelle au travers des environnements de développement. Cette caractéristique est plus prononcée que dans les autres technologies à composants car elle est prise en compte explicitement dans la spécification des JavaBeans : "*un JavaBean est un composant logiciel réutilisable qui peut être manipulé visuellement dans un outil de construction*"[1] [Beans 1997].

En ce qui concerne les EJB [EJB 2002], l'objectif est différent : ils sont dédiés au développement d'applications distribuées et pour Internet. Les EJB intègrent différents services non fonctionnels utiles dans un environnement distribué, comme les concepts de transaction, de persistance ou de sécurité.

1.3 Composants de l'OMG

L'OMG propose elle aussi une technologie à composants : CCM [Corba 2002]. Cette technologie est indépendante de la plate-forme cible et du langage de programmation. Les composants CCM sont, comme les EJB et les composants .Net, destinés aux applications distribuées et notamment celles d'Internet. Chaque composant est décrit sous la forme d'une interface écrite en langage IDL (Interface Description Language) permettant une correspondance avec les différents langages de programmation. En effet, des pré-compilateurs dédiés permettent de générer automatiquement le squelette de l'interface IDL dans un langage donné, en produisant aussi le code qui assure l'appel de fonctions distantes et le traitement des résultats. Ce code porte le nom de stub, côté client, et de skeleton, côté serveur. Un module dont l'interface est spécifiée en IDL peut ainsi être programmé en C++, tandis que des modules Java qui l'utilisent effectuent des appels sur une interface Java générée à partir du même IDL. Au final, l'architecture CORBA assure l'acheminement des appels entre les processus.

[1] "*A JavaBeans is a reusable software component that can be manipulated visually in a builder tool.*" [Beans 97]

1.4 Composants de l'OSGi Alliance

La plate-forme OSGi (Open Services Gateway initiative) [OSGi 2003] présente une technologie à composants destinée à faire communiquer entre eux des dispositifs distants. Les composants sont plutôt destinés aux petits dispositifs (PDA, téléphones portables, etc.). Par rapport à d'autres technologies, l'assemblage des composants peut être dynamique, lors de l'exécution du programme. Basé sur le langage Java, OSGi fournit les fonctionnalités pour la création, le déploiement et l'activation des composants.

Les technologies à composants sont multiples et répondent à des objectifs divers. Il convient de noter que toutes les technologies présentés ici sont écrites avec un langage orienté objet (Java, C++, etc.). A la différence de la programmation orientée objet, la programmation basée composant définit les concepts de composition (assemblage) et de déploiement.

2 JavaBeans

Parmi toutes les technologies existantes, notre choix d'implémentation des composants ICARE s'est tourné vers la technologie JavaBeans de Sun Microsystems [Beans 1997]. Cette section décrit les raisons de ce choix puis développe les caractéristiques spécifiques de cette technologie. Pour rappel, "*un JavaBean est un composant logiciel réutilisable qui peut être manipulé visuellement dans un outil de construction*" [Beans 1997].

2.1 Pourquoi ce choix

Notre choix est motivé par deux raisons répondant aux objectifs définis au début de ce mémoire :

1. Le langage Java présente de nombreux avantages. Il a l'intérêt d'être multiplate-forme nous assurant la compatibilité de l'interaction multimodale créée avec tous les systèmes informatiques existants. Sa simplicité, son haut niveau d'abstraction et sa large diffusion permettent de faciliter la création de nouveaux composants Dispositifs et Langages d'interaction par un grand nombre de développeurs. Enfin, il fournit les passerelles vers de nombreux autres langages nous permettant, par exemple, de développer la surcouche du composant Dispositif en suivant la technologie JavaBeans alors que l'algorithme de calcul spécifique au dispositif est réalisé dans un autre langage (C++, Mathlab, etc.).
2. La manipulation visuelle des JavaBeans constitue une approche véritablement intéressante car le travail de l'assembleur de composants est simplifié. L'outil de manipulation graphique permet d'améliorer le temps de production d'un système informatique. En présentant une API (Application Programming Interface) spécialement dédiée à cet aspect, cette technologie nous a permis de développer facilement un éditeur graphique pour l'assemblage des composants ICARE (présenté dans les sections suivantes de ce chapitre).

2.2 Spécificités

Les JavaBeans sont des composants logiciels écrits en langage Java qui ne doivent pas être confondus avec les Entreprise JavaBeans (EJB). Un composant JavaBeans respecte certaines conventions sur le nommage des méthodes, la construction et le comportement. Le respect de ces conventions rend possible l'utilisation, la réutilisation, le remplacement et la connexion de JavaBeans par des outils de développement. Le composant doit ainsi :

- être “serialisable” pour pouvoir sauver et restaurer l’état des composants ;
- posséder un constructeur sans argument ;
- posséder des méthodes d’accès et de modifications de ses attributs ;
- suivre une convention de nommage pour les méthodes.

Par exemple, le fragment de code source ci-dessous montre le code d’un composant avec la convention de nommage pour les méthodes d’accès et de modification des attributs.

```
// MonComposant.java

public class MonComposant implements java.io.Serializable
   {
      private String unAttribut;

      // Constructeur par défaut (Ne prenant pas d'argument).
      public MonComposant() {}

      // La méthode d'accès à l'attribut commence obligatoirement par
      // « get » suivi du nom de l'attribut
      public String getUnAttribut()
      {
         return (this.unAttribut);
      }

      // La méthode modifiant l'attribut commence obligatoirement par
      // « set » suivi du nom de l'attribut
      public void setUnAttribut(String nouvelleValeur)
      {
         this.unAttribut = nouvelleValeur;
      }
      …
   }
```

Le code des composants JavaBeans peut être enrichi par d’autres informations, comme une icône qui représente le composant dans les éditeurs graphiques. Après compilation, les composants sont distribués dans des archives Java (fichier.jar).

3 Les composants de l'outil ICARE

Cette section expose l'implémentation, avec la technologie JavaBeans, des différents composants ICARE disponibles.

3.1 Implémentation des composants Dispositifs et Langages d'interaction

Pour créer ces composants, nous avons défini deux classes "mères" dont chaque composant Dispositif ou Langage d'interaction peut hériter. Ces classes mères définissent les différents attributs que nous avons identifiés au chapitre 6 pour ces deux types de composants.

De plus, nous précisons, pour chaque composant Dispositif ou Langage d'interaction, la nature des données (type et description textuelle) qui sont transmises et peuvent être reçues, nous permettant ainsi d'assurer la compatibilité lors de la connexion de plusieurs composants.

Nous fournissons plusieurs composants Dispositifs et Langages d'interaction, que nous avons utilisés dans les systèmes interactifs présentés dans la Partie 3 de ce mémoire. L'annexe 1 montre un exemple de code source pour un composant Dispositif et pour un composant Langage d'interaction.

3.2 Implémentation des composants de combinaison

L'implémentation des trois composants de combinaison (Complémentarité, Redondance, Redondance/Equivalence) s'est réalisée en suivant strictement les règles spécifiées au chapitre 6. L'annexe 2 montre le code source des trois mécanismes de fusion.

Pour valider complètement les différents mécanismes de fusion, chacun d'eux a été testé rigoureusement. Nous avons utilisé une architecture de test composée de l'outil Tobias, permettant de générer un large nombre de cas de test à partir d'un scénario, de JUnit pour exécuter ces cas de test et des assertions JML (Java Modelling Language) pour produire automatiquement le résultat des cas de test. Présentés dans [Dupuy-Chessa 2005], le code des composants de combinaison a été augmenté par des assertions JML décrivant les propriétés des algorithmes. Ensuite, les cas de test générés par Tobias ont été exécutés, avec JUnit, sur les composants de combinaison augmentés des assertions JML. Les premiers tests des composants de combinaison d'ICARE ont présenté plusieurs erreurs. Cette activité de test nous a permis d'obtenir des mécanismes robustes.

3.3 Communication entre composants dans ICARE

La communication entre les composants est faite par génération d'événements et appels de méthodes, correspondant au modèle d'événements Java. L'événement généré, selon la spécification d'un ICAREEvent donnée au chapitre 6, véhicule les données à traiter. Lorsque deux composants sont connectés, le composant récepteur s'abonne aux événements générés par le composant émetteur selon les mécanismes JavaBeans prévus.

D'autre part, chaque composant ICARE implémente une queue d'événements FIFO (First In First Out) assurant le traitement des événements dans leur ordre d'émission. Un processus léger (thread) est créé pour chaque traitement de l'événement jusqu'à la production d'un nouvel événement. Cette gestion asynchrone des événements permet de ne pas bloquer les processus de traitement des composants émetteurs.

4 Editeur graphique d'ICARE

Profitant des atouts de la technologie JavaBeans fournissant tous les éléments pour manipuler graphiquement les composants, nous avons développé un éditeur graphique spécialement dédié pour l'outil ICARE et la multimodalité. Cette section expose les motivations de la création d'un tel éditeur avant de le décrire précisément.

4.1 Objectifs

La création d'un éditeur graphique spécifique à la multimodalité est motivé par les objectifs de simplification et d'accélération du processus de conception et de développement. Pour ce faire, il nous a semblé déterminant que cet éditeur :

- facilite la conception et la re-conception de l'interaction multimodale, s'adaptant ainsi aux conceptions itératives centrées sur utilisateur,
- soit manipulable par des utilisateurs informaticiens et non-informaticiens,
- génère automatiquement le code correspondant à la conception.

La section suivante décrit l'éditeur que nous avons développé.

4.2 Description

Notre éditeur permet l'assemblage des composants Dispositifs, Langages d'interaction et de combinaison développés avec la technologie JavaBeans.

La Figure 7.1 présente l'interface graphique de l'éditeur. Il contient une palette de composants (zone A), une zone d'édition (zone B) et un panneau pour configurer les attributs des composants (zone C).

Zone A : la palette de composants est organisée selon les trois types de composants : combinaison, dispositifs et langages d'interaction permettant de visualiser directement les possibilités offertes aux concepteurs.

Zone B : la zone d'édition permet aux concepteurs d'assembler les composants ICARE, définissant ainsi les modalités d'interaction disponibles pour l'utilisateur et les usages qu'il peut en faire en fonction des tâches à réaliser.

Zone C : le panneau de configuration permet de visualiser et modifier, le cas échéant, les attributs afin de configurer/modifier le comportement du composant. Par exemple, dans ce panneau, la fenêtre temporelle des composants de combinaison peut être modifiée.

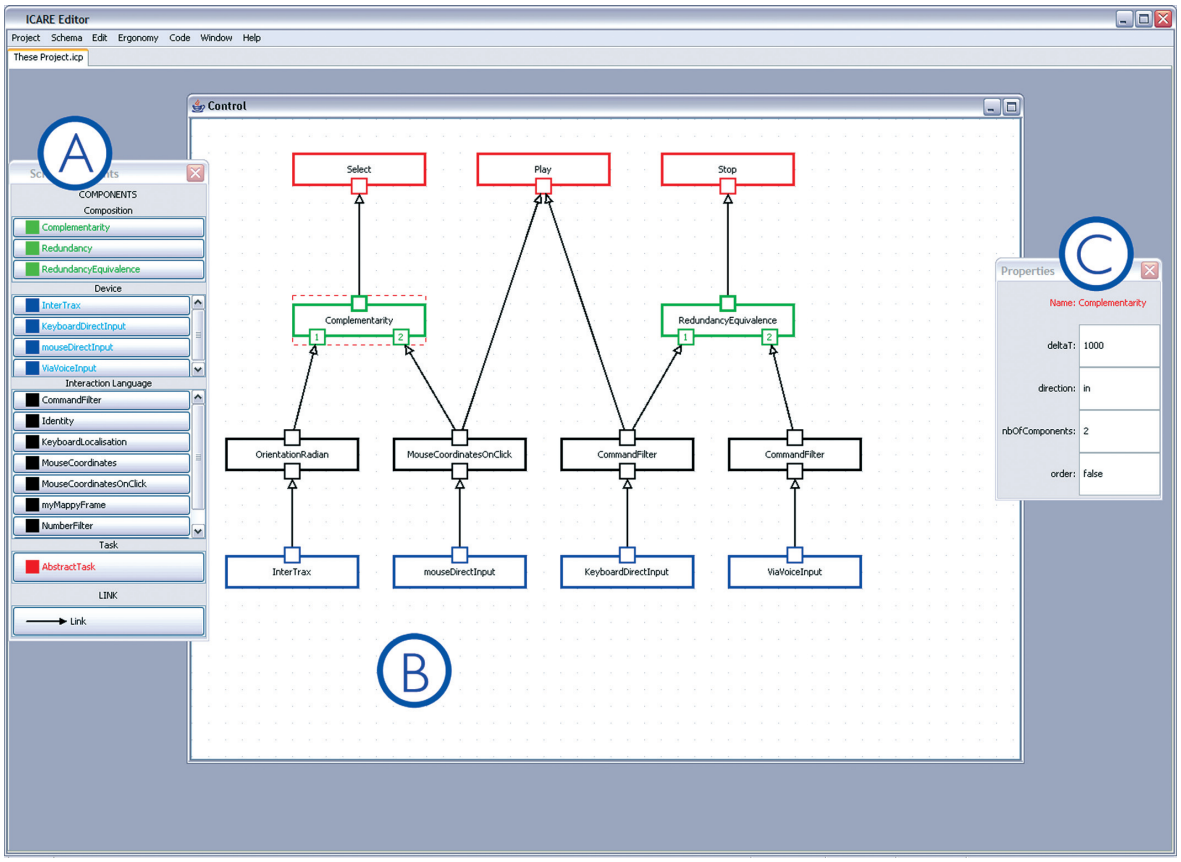


Figure 7.1 : l'éditeur graphique d'ICARE pour manipuler les composants.

4.3 Démarche de conception

Alors que les différents composants ont été développés préalablement et placés dans un répertoire approprié, la palette des composants les présente aux concepteurs, utilisateurs de l'éditeur.

Le concepteur sélectionne les modalités d'interaction (composants Dispositifs et Langages d'interaction) et spécifie leurs combinaisons (composants Complémentarité, Redondance, Redondance/Equivalence) en assemblant graphiquement les composants logiciels dans la zone d'édition, sans connaître les détails du code de chaque composant. Il paramètre ensuite les attributs aux valeurs souhaitées, comme par exemple la durée de la fenêtre temporelle ou la stratégie de combinaison (précoce ou différée). Une fois la conception graphique du système multimodal réalisé pour les tâches données, la génération automatique du code correspondant à l'assemblage et aux interfaces avec le reste du système informatique est effectuée.

Une fois la première intégration avec le reste du système informatique effectuée, l'assemblage des composants peut être modifié facilement en remplaçant, par exemple, une forme de combinaison par une autre ou en modifiant l'un des attributs. En quelques minutes, la nouvelle configuration, générée automatiquement, peut être testée au sein du système informatique complet, permettant de faire de nombreux tests avec les utilisateurs finaux du système multimodal. L'outil ICARE est donc un outil qui favorise une conception itérative centrée sur l'utilisateur.

4.4 Génération automatique de code

Préparant l'intégration avec le reste du système informatique, la génération automatique du code, en Java, correspondant à l'assemblage des composants, est effectuée lors de la sauvegarde. Pour chaque projet, un répertoire est créé et contient tous les fichiers nécessaires pour exécuter l'interaction multimodale conçue sur l'ordinateur cible. Le répertoire comprend ainsi les différents composants utilisés (archives Java), le fichier de configuration qui contient l'assemblage des composants, le fichier permettant de démarrer, arrêter et accéder aux composants, et enfin le fichier de liaison avec le reste du système informatique. Ce dernier fichier doit être implémenté manuellement avec le reste de l'application.

4.5 Lien avec le reste du système informatique

Le lien entre l'assemblage des composants ICARE et le reste du système informatique est réalisé au travers des tâches de l'utilisateur. En effet, chaque assemblage de composants s'effectue par rapport à une tâche de l'utilisateur. Les données qui proviennent des modalités d'interaction, éventuellement combinées, renseignent de toutes les informations dont à besoin le système informatique pour traiter la tâche visée. Le fichier de liaison entre l'assemblage de composants ICARE et le reste du système informatique est généré automatiquement. Il doit donc être complété manuellement. Le fragment de code source ci-après montre un exemple du fichier généré automatiquement à partir d'assemblages de composants correspondant à deux tâches de l'utilisateur.

```
//Interface entre l'assemblage des composants ICARE et le reste du
//système informatique

public class IcareInterface  implements java.io.Serializable
{

   // A modifier si le type de dialog est connu
   private Object dialog = null;

   public IcareInterface()
   {
   }

   public void setDialog (Object theDialog)
   {
      //Référence sur l'objet qui traite directement les données
      //multimodales (ex : contrôleur de dialogue)
      //A COMPLETER
   }

   public void task1(ICARE.ICAREEvent event)
   {
      //event contient un vecteur contenant toutes les données
      //fournies par les différentes modalités d'interaction
      //utilisées pour cette tâche
      //A COMPLETER
   }

   public void task2(ICARE.ICAREEvent event)
   {
      //event contient un vecteur contenant toutes les données
      //fournies par les différentes modalités d'interaction
      //utilisées pour cette tâche
      //A COMPLETER
   }
}
```

Le même fragment de code, présenté sur la page suivante et complété manuellement, relie effectivement les deux parties (interaction multimodale en entrée et reste du système informatique) pour obtenir un système interactif utilisable. Dans cet exemple, nous utilisons un simple appel de méthode. D'autres solutions sont possibles. Par exemple, si le reste du système informatique est écrit dans le langage C++, nous pourrions ici programmer une ouverture de socket UDP ou encore réaliser un lien avec la technologie JNI (Java Native Interface) prévue pour relier Java au C++.

```
//Interface entre l'assemblage des composants ICARE et le reste du
//système informatique

public class IcareInterface  implements java.io.Serializable
{

   //dialog est du type DialogController
   private DialogController dialog = null;

   public IcareInterface()
   {
   }

   public void setDialog (Object theDialog)
   {
      //Référence sur l'objet qui traite directement les données
      //multimodales (ex : contrôleur de dialogue)
      //A COMPLETER

      dialog = (DialogController)theDialog;
   }

   public void task1(ICARE.ICAREEvent event)
   {
      //event contient un vecteur contenant toutes les données
      //fournies par les différentes modalités d'interaction
      //utilisées pour cette tâche
      //A COMPLETER

      dialog.execute_tache1(event.getVector());
   }

   public void task2(ICARE.ICAREEvent event)
   {
      //event contient un vecteur contenant toutes les données
      //fournies par les différentes modalités d'interaction
      //utilisées pour cette tâche
      //A COMPLETER

      dialog.execute_tache2(event.getVector());
   }
}
```

5 Résumé du chapitre 7

Pour développer des systèmes multimodaux, nous avons proposé une implémentation de notre modèle à composants en JavaBeans. Parmi la diversité des technologies à composants existantes, la technologie JavaBeans rassemble tous les atouts du langage Java (opérabilité, facilité d'usage ou encore large diffusion parmi les programmeurs) pour l'implémentation des composants ICARE. De plus, la technologie JavaBeans offre tous les mécanismes permettant de manipuler visuellement les composants.

Ainsi plusieurs composants Dispositifs et Langages d'interaction sont aujourd'hui disponibles et sont utilisés dans plusieurs systèmes informatiques, que nous présentons dans la Partie 3 de ce mémoire. Les trois composants de combinaison (Complémentarité, Redondance, Redondance/Equivalence) fournissent des mécanismes génériques simples pour combiner plusieurs modalités d'interaction.

Finalement, un éditeur graphique de manipulation des composants ICARE complète notre outil global en simplifiant et accélérant grandement les démarches de conception et de développement. La manipulation graphique (glisser/déposer) des composants logiciels permet de créer ou modifier très simplement les modalités d'interaction et leurs combinaisons en fonction des tâches à effectuer. Le haut niveau d'abstraction des concepts manipulés (Dispositif, Langage d'interaction, Complémentarité, Redondance, Redondance/Equivalence) est approprié aux utilisateurs aussi bien informaticiens que non informaticiens. Enfin, la génération automatique de code correspondant à l'assemblage accélère la phase de codage du cycle de développement logiciel dans le cadre d'une conception itérative centrée sur l'utilisateur.

Nous concluons la présentation de l'outil ICARE en le situant, au chapitre suivant, dans notre grille d'analyse utilisée pour étudier les outils existants.

CHAPITRE 8
OUTIL ICARE : SYNTHÈSE

Ce chapitre résume les apports de notre outil ICARE en fonction de son modèle à composants présenté au chapitre 6 et de notre solution d'implémentation par des composants JavaBeans présentée au chapitre 7. Pour ce faire, nous positionnons l'outil ICARE en spécifiant les critères qui le caractérisent dans la grille d'analyse utilisée au chapitre 5 pour étudier les outils existants.

Ce chapitre contient quatre premières sections correspondant à la structure de notre grille d'analyse. La carte d'identité de l'outil est d'abord présentée en détail. Nous développons ensuite les critères généraux à l'IHM. Puis, nous exposons ceux spécifiques à la multimodalité. Enfin, nous décrivons les critères dédiés aux mécanismes de fusion. Les deux dernières sections présentent la grille d'analyse complète et la synthèse de ce chapitre.

1 Carte d'identité

En considérant le modèle conceptuel et notre solution d'implémentation à l'aide des composants JavaBeans, l'outil ICARE participe aux phases de spécification, conception (globale et détaillée), de codage et prochainement à celle de tests. Lors de la phase de spécification, le modèle conceptuel que suit ICARE (cf. chapitre 4) permet de définir les modalités d'interaction (Dispositifs, Langage d'interaction) et les combinaisons à mettre en œuvre. Ensuite, l'outil ICARE participe à la phase de conception globale par l'assemblage des différents composants dans l'éditeur graphique correspondant aux spécifications définies précédemment. La définition précise des différentes caractéristiques de chaque composant offre une aide pour la conception détaillée. La génération automatique de code accélère considérablement la phase de codage. Enfin, des travaux de recherche pour la couverture des phases de tests sont aujourd'hui à l'étude. Deux approches complémentaires sont adoptées. La première consiste au couplage de l'outil ICARE avec l'outil ACIDU qui permet de capturer les actions des utilisateurs sur téléphone mobile (tests d'usage) en traçant les données qui circulent dans les composants d'ICARE [Serrano 2006] (cf. annexe 3). La seconde se concrétise par le couplage de l'outil ICARE avec l'outil Lutess, une plate-forme de test pour les logiciels réactifs synchrones [Madani 2005] [Jourde 2006] (cf. annexe 4). L'outil ICARE contribue ainsi à de nombreuses phases du cycle de développement logiciel, comme l'outil PetShop présenté au chapitre 5.

La manipulation graphique des composants et la possibilité de changer rapidement de composants (modalités d'interaction ou combinaisons) couplées à la génération automatique de code représente une solution totalement adaptée au prototypage rapide car plusieurs formes d'interaction multimodale peuvent être simplement et rapidement créées, modifiées et exécutées.

Le haut niveau d'abstraction des composants logiciels et l'absence de programmation lors de la construction de l'assemblage des composants destinent cet outil à des concepteurs informaticiens ou non informaticiens (ergonomes, psychologues, etc.). Par contre, le développement préalable de chaque composant se fait manuellement et demande des connaissances en programmation Java. De même, la connexion du code généré automatiquement et correspondant aux assemblages de composants ICARE avec le reste du système informatique se fait aussi manuellement. Cependant, le peu de concepts à manipuler (Dispositifs, Langage d'interaction, Combinaisons) facilite l'apprentissage global de l'outil.

Son pouvoir d'expression est très élevé car de nombreux systèmes interactifs multimodaux peuvent être conçus et développés avec l'outil ICARE. Les nombreux systèmes multimodaux développés avec ICARE et présentés dans la Partie 3 de ce mémoire témoignent de ce pouvoir d'expression.

2 Critères pour l'IHM

Comme pour les outils FAME et Embassi (cf. chapitre 5), l'outil ICARE répond favorablement aux critères de prise en compte des capacités de l'utilisateur et des données issues de l'environnement extérieur.

En intégrant directement des caractéristiques propres aux utilisateurs dans les composants Dispositifs et Langages d'interaction, il est possible de déterminer si les modalités d'interaction employées sont adaptées aux capacités physiques et cognitives des utilisateurs. Comme nous l'avons précisé au chapitre 4, la portabilité, le mode de communication et le lieu d'interaction nominal des dispositifs doivent être confrontés avec les facteurs biologiques de l'utilisateur. Ensuite, les niveaux d'expertise requis pour l'utilisation des dispositifs et des langages d'interaction sont en relation directe avec les facteurs socioculturels humains. Ainsi, le niveau de formation et l'expérience de l'utilisateur permet de définir si l'utilisateur a les capacités cognitives pour leur utilisation optimale. De plus, les caractères analogique et non-arbitraire des langages d'interaction garantissent des repères à la réalité physique pour l'utilisateur et augmentent ainsi sa capacité à interpréter les informations. La confrontation des caractéristiques des dispositifs et des langages d'interaction avec celles de l'utilisateur permet de détecter d'éventuelles incohérences ergonomiques.

La gestion des données de l'environnement extérieur est rendue possible dans l'outil ICARE par la création de modalités d'interaction passives pouvant être combinées selon les mêmes principes que pour les autres modalités d'interaction. En effet, il est possible de créer des composants Dispositifs et Langages d'interaction captant par exemple le niveau sonore d'une pièce dans laquelle se trouve l'utilisateur et de combiner cette modalité d'interaction de manière complémentaire avec une modalité de reconnaissance vocale afin de déterminer, par exemple, la nature visuelle ou auditive du retour d'information de la tâche réalisée en fonction du niveau de bruit.

L'outil ICARE permet de manipuler toutes modalités d'interaction en entrée.

Les mécanismes de fusion sont génériques. Ils sont définis dans les trois composants Complémentarité, Redondance et Redondance/Equivalence. Ils combinent les données issues de deux à n modalités d'interaction utilisées parallèlement ou séquentiellement, répondant ainsi à toutes les formes d'usage identifiées au chapitre 3, et ce en indépendance totale avec le domaine d'application.

4 Critères pour la fusion des données

Les mécanismes de fusion des données issues des combinaisons de plusieurs modalités d'interaction sont entièrement paramétrables, dans l'outil ICARE, en modifiant les attributs des composants Complémentarité, Redondance et Redondance/Equivalence. Il est le seul des outils étudiés qui satisfait l'ensemble de nos critères.

La fusion des données se réalise principalement sur leur proximité temporelle (définition d'une fenêtre temporelle), qui peut toutefois être désactivée lorsque des données provenant d'actions utilisateurs sont à fusionner indépendamment des aspects temporels. Ensuite, les composants Complémentarité et Redondance proposent une stratégie d'intégration précoce permettant d'obtenir d'une bonne performance sans ambigüités sur les actions de l'utilisateur (vérifié lors des tests des composants de combinaison, cf. chapitre 7). Le composant Redondance/Equivalence propose les deux stratégies d'intégration (précoce et différée) car cette combinaison ouvre une incertitude sur l'usage équivalent ou redondant des modalités d'interaction. En stratégie différée, cette incertitude peut facilement être levée en attendant un laps de temps supplémentaire, avant la propagation des nouvelles données produites par le composant.

De plus, la vérification du type de données à combiner dans les mécanismes de fusion ne permet pas de combiner des modalités d'interaction incompatibles. Par exemple, il n'est pas possible de vérifier la redondance des informations sans que les modalités d'interaction ne produisent des données équivalentes.

Enfin, nos mécanismes de fusion peuvent être paramétrés pour ne fusionner que les données des modalités d'interaction qui ont été spécifiées par l'utilisateur, selon une certaine chronologie. Grâce à la vérification des estampilles de temps définies lors de l'utilisation du dispositif par l'utilisateur, et qui circulent dans la structure événementielle, l'ordonnancement des données qui est observé correspond bien à l'ordonnancement des actions produites par l'utilisateur.

Le Tableau 8.1 résume les critères de l'outil ICARE présentés précédemment.

Carte d'identité	
Couverture du cycle de développement	Spécifications, conception globale et détaillée, codage, tests (en cours)
Support de prototypage	Oui
Cible utilisateur	Concepteurs informaticiens et non informaticiens, programmeurs
Difficulté d'apprentissage	Moyenne
Pouvoir d'expression	Très élevé
Dépendance d'une technologie de développement	Composants
Dépendance d'un langage de programmation	Java
Représentation des informations	Unique – ICAREEvent
Gestion du code source	Semi-automatique
Prévisibilité	Oui
Critères d'IHM	
Prise en compte des capacités de l'utilisateur	Oui
Prise en compte de l'environnement extérieur	Oui
Critères pour la multimodalité	
Généricité par rapport aux modalités d'interaction	Oui
Généricité des mécanismes de fusion	Oui
Nature des combinaisons possibles	Complémentarité, Redondance, Equivalence, Redondance/Equivalence
Temporalité	Parallèle et séquentielle
Critères pour la fusion des données	
Stratégie d'intégration	Précoce
Proximité temporelle	Oui
Complémentarité logique	Oui
Incompatibilité des modalités à combiner	Oui
Ordonnancement des données multimodales	Oui

Tableau 8.1 : propriétés de l'outil ICARE.

6 Résumé du chapitre 8

L'outil ICARE répond aux objectifs visés. Il permet la conception et la réalisation logicielle rapide de systèmes multimodaux tout en permettant tous types de multimodalité. Le caractère générique des composants Complémentarité, Redondance, Redondance/Equivalence représente un réel avantage car ces composants permettent d'exprimer toutes les formes de combinaisons possibles entre les modalités d'interaction. Cet aspect qui facilite grandement les choix de conception, distingue complètement cet outil des autres solutions proposées pour la multimodalité.

Ce chapitre clôt la description de notre outil à composants reflétant notre contribution principale. Nous illustrons l'apport de notre outil ICARE dans la partie suivante de ce rapport. Cette dernière est consacrée aux systèmes multimodaux développés avec ICARE. Ces systèmes sont variés en termes de modalités d'interaction et formes de multimodalité.

Partie 3

REALISATION DE SYSTEMES MULTIMODAUX

PARTIE 3

REALISATION DE SYSTEMES MULTIMODAUX

Cette troisième partie du mémoire présente les différents systèmes interactifs développés avec l'outil ICARE. Cinq systèmes multimodaux sont développés et présentés dans les trois chapitres suivants.

Le chapitre 9 présente le premier système interactif réalisé avec des composants de l'outil ICARE. Il s'agit d'un système multimodal pour l'identification des utilisateurs appelé MID (Multimodal IDentification). Ce système montre la faisabilité de notre approche pour les phases de spécifications, de conception et de codage du cycle de développement logiciel. MID offre un cas simple de multimodalité par équivalence. L'équivalence des différentes modalités d'interaction offre plusieurs évolutions possibles en intégrant par exemple des cas de redondance. Une évolution du système est étudiée, montrant la facilité et la rapidité de re-conception offerte par l'outil ICARE.

P. 211

Le chapitre 10 rassemble deux systèmes interactifs combinant le geste et la parole. Il s'agit d'un système permettant de situer l'adresse d'une personne sur un plan, appelé YellowPages, et d'un simulateur de trafic aérien en vol (ATC – Air Trafic Control). Ces deux systèmes montrent diverses possibilités de combinaison entre des modalités d'interaction gestuelle et orale. Ils présentent aussi des solutions différentes en termes d'assemblage de composants, afin de prendre en charge des interactions fortement couplées entre l'interface d'entrée et de sortie, comme lors de l'utilisation de la souris dans les interfaces WIMP (modalité gestuelle).

P. 223

Enfin le chapitre 11 regroupe deux systèmes interactifs mettant en œuvre le paradigme de réalité augmentée. Nous focalisons notre présentation aux moyens qu'offre l'outil ICARE pour permettre la fusion harmonieuse des mondes numérique et physique. Nous exposons ainsi le système MEMO, qui met en œuvre des post-its géo-localisés, et un simulateur d'avion de combat (SAC).

P. 233

Sommaire >> >> >>

PARTIE 3
REALISATION DE SYSTEMES MULTIMODAUX

CHAPITRE 9 211
SYSTÈME INTERACTIF JOUET
1 Cahier des charges : présentation du système interactif 212
2 Spécification, conception et codage 214
3 Résumé du chapitre 9 221

CHAPITRE 10 223
SYSTÈMES INTERACTIFS DU PARADIGME GESTE/PAROLE
1 Système interactif YellowPages 224
2 Simulateur ATC 229
3 Résumé du chapitre 10 232

CHAPITRE 11 233
SYSTÈMES INTERACTIFS DE RÉALITÉ AUGMENTÉE
1 MEMO 234
2 Simulateur d'avion de combat 238
3 Résumé du chapitre 11 242

CHAPITRE 9
SYSTÈME INTERACTIF JOUET

Ce chapitre présente un premier système interactif réalisé avec des composants de l'outil ICARE. Ce cas d'étude démontre la faisabilité de notre approche par un exemple simple de combinaisons multimodales de plusieurs modalités d'interaction équivalentes pour une seule tâche. L'équivalence des différentes modalités d'interaction offre plusieurs évolutions possibles, en intégrant par exemple des cas de redondance entre ces modalités d'interaction. Au sein de ce système interactif, une évolution est étudiée, montrant la facilité et la rapidité de re-conception offerte par l'outil ICARE. Il est à noter ici que nous ne nous sommes pas préoccupés de l'utilisabilité des modalités d'interaction mises en œuvre.

Ce chapitre décrit le système interactif MID (Multimodal IDentification) avant de présenter sa spécification, sa conception et son codage avec l'outil ICARE.

1 Cahier des charges : présentation du système interactif

Cette première section décrit notre système interactif jouet. Il s'agit d'un système multimodal d'identification appelé MID (Multimodal IDentification) intégrant plusieurs modalités d'interaction basées sur des dispositifs classiques. L'utilisation de ce système se déroule sur un poste de travail classique par des utilisateurs ne présentant aucun handicap.

1.1 Tâche

Comme il s'agit du premier système multimodal développé avec notre approche, nous limitons ce cas d'étude à une seule tâche d'interaction. Des systèmes interactifs plus complexes sont présentés dans les deux chapitres suivants.

La tâche d'interaction à réaliser est de spécifier un mot de passe permettant d'identifier un utilisateur sans ambigüité. Par rapport aux systèmes de connexion existants, le mot de passe fait aussi office d'identifiant.

1.2 Modalités d'interaction

Pour réaliser cette tâche, trois modalités d'interaction équivalentes sont proposées aux utilisateurs.

1. **L'utilisation du clavier** : il ne s'agit pas ici de rentrer uniquement une chaîne de caractères mais de jouer sur la combinaison et l'état des touches (pressée, relâchée). Par exemple, la combinaison peut être la suivante : "a_pressé, z_préssé, c_pressé, c_relâché, a_relâché, d_pressé, d_relâché, z_relâché, entrée_pressé, entrée_relâché", l'utilisateur maintient ainsi la touche a, puis la touche z pendant qu'il tape sur la touche c, puis relâche la touche a (la touche z étant toujours maintenue), tape la touche d et relâche enfin la touche z. Le mot de passe se finit toujours par une pression et un relâchement de la touche entrée.

2. L'**utilisation des boutons de la souris** : cette modalité d'interaction est prévue pour une souris cinq boutons. De la même façon que pour le clavier, les cinq boutons de la souris servent à spécifier une combinaison en fonction de leur état (pressé, relâché) pour spécifier un mot de passe. Par exemple, la combinaison peut être "bouton1_pressé, bouton4_pressé, bouton1_relâché, bouton3_pressé, bouton3_relaché, bouton4_relaché, bouton5_pressé, bouton5_relâché". Le mot de passe se finit toujours par la pression et le relâchement du cinquième bouton.

3. La **reconnaissance vocale** : il s'agit de définir une phrase clé utilisée comme mot de passe comme par exemple : "Ma voix est mon mot de passe".

1.3 Combinaisons de modalités

Dans un premier temps, les modalités d'interaction sont utilisées de manière équivalente. Par la suite, une évolution est proposée afin de tester des cas de redondance entre d'une part, l'utilisation du clavier en redondance de la parole et d'autre part, l'utilisation de la souris en redondance de la parole. L'utilisateur a le choix (équivalence) entre ces deux possibilités de redondance.

1.4 Interface graphique

La Figure 9.1 montre l'interface graphique du système interactif MID. L'utilisateur doit spécifier son mot de passe devant cet écran qui indique, une fois le mot de passe entré, la reconnaissance ou non de l'utilisateur par un message textuel et vocal.

Figure 9.1 : fenêtre principale du système MID.

Dans cette section, nous décrivons l'utilisation de notre approche à composants ICARE pour les phases de spécifications, de conception et de codage.

2.1 SPÉCIFICATIONS DES MODALITÉS D'INTERACTION ET DES COMBINAISONS

Comme défini dans le cahier des charges, le système interactif MID met en œuvre trois modalités d'interaction pour spécifier l'identité de l'utilisateur au système interactif : l'utilisation du clavier, l'utilisation des boutons de la souris et la reconnaissance vocale. Ces trois modalités sont d'abord utilisées de manière équivalente par l'utilisateur puis combinées de manière redondante. Deux cas de redondance sont spécifiés : redondance entre l'utilisation du clavier et la reconnaissance vocale et redondance entre l'utilisation des boutons de la souris et la reconnaissance vocale. Nous utilisons le modèle conceptuel défini au chapitre 4 de ce mémoire pour spécifier ces trois modalités d'interaction et les différentes formes de combinaison que nous voulons obtenir.

2.1.1 Modalités d'interaction

La première modalité d'interaction est l'utilisation du clavier. Comme le montre la Figure 9.2, cette modalité d'interaction résulte du couple dispositif "clavier" et du langage d'interaction "mot de passe clavier". Il s'agit d'une modalité d'interaction active qui demande un acte gestuel de la part de l'utilisateur. L'adéquation

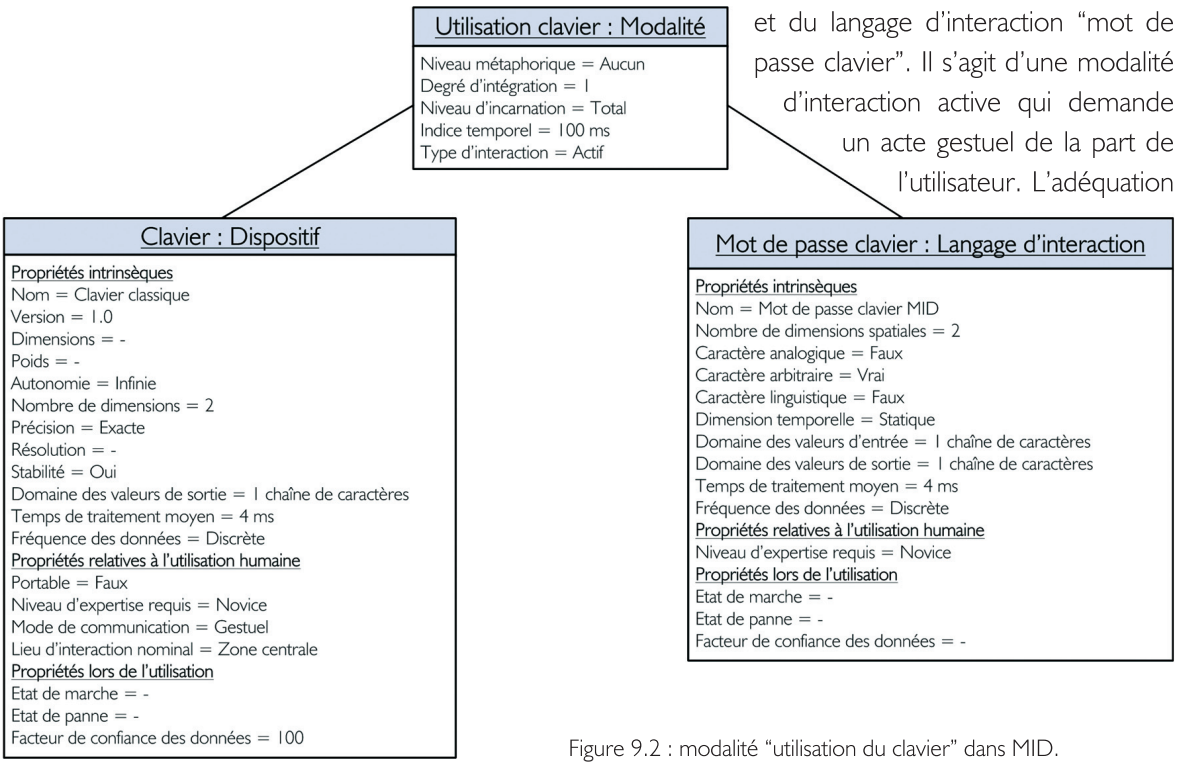


Figure 9.2 : modalité "utilisation du clavier" dans MID.

entre le dispositif, le langage d'interaction et la tâche, définie par le niveau métaphorique et le degré d'intégration n'est pas optimale. En effet, la ressemblance avec la réalité n'existe pas (niveau métaphorique = aucun) car il s'agit d'un moyen complètement nouveau pour la spécification d'un mot de passe, qui n'utilise aucune référence à des moyens déjà existants dans le monde physique. Par contre, le degré d'intégration est idéal (égal à 1). En ce qui concerne le degré d'indirection, il est très bon : le niveau d'incarnation est total car les effets de l'utilisation du clavier se répercute directement sur le dispositif ; l'indice temporel est infime (100 ms).

La seconde modalité d'interaction est l'utilisation des boutons de la souris. La Figure 9.3 montre le couple dispositif "souris" et langage d'interaction "mot de passe souris", ainsi que les caractéristiques définies pour chacun des niveaux. Comme pour l'utilisation du clavier, il s'agit d'une modalité d'interaction active, demandant un acte gestuel de la part de l'utilisateur. Nous retrouvons également les mêmes caractéristiques que pour l'utilisation du clavier, en ce qui concerne son adéquation et son degré d'indirection.

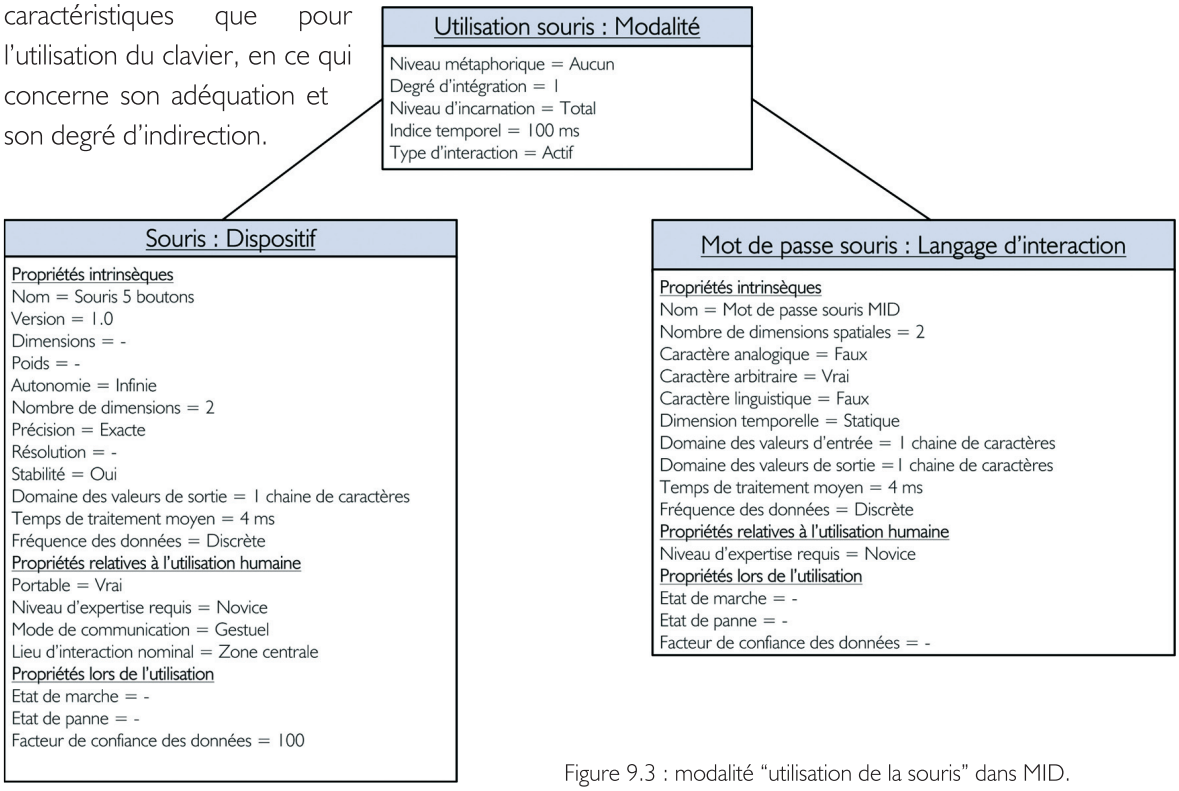


Figure 9.3 : modalité "utilisation de la souris" dans MID.

La troisième et dernière modalité d'interaction est la reconnaissance vocale. La Figure 9.4 montre les caractéristiques du couple dispositif "Viavoice" et "mot de passe vocal". Il est remarquable ici que le dispositif n'est pas le microphone mais directement la technologie de reconnaissance vocale. En effet, le système

Viavoice, qui reconnaît les paroles de l'utilisateur, ne nous autorise pas à contrôler le microphone. Nous considérons ainsi que le dispositif employé est la technologie de reconnaissance elle-même, car elle représente la plus petite entité manipulable dans le système d'exploitation de l'ordinateur. Il s'agit d'une modalité d'interaction active qui demande une communication orale de la part de l'utilisateur. L'adéquation entre le dispositif, le langage d'interaction et la tâche est ici bien meilleure qu'avec les deux modalités d'interaction précédentes, car l'analogie avec l'utilisation de la voix pour communiquer dans le monde physique est totale. En effet, le niveau métaphorique est total et nous rappelle d'ailleurs le plus connu des contes des milles et une nuits, Ali Baba et les quarante voleurs, avec son célèbre mot de passe "Sésame, ouvre toi !" ouvrant la cachette des trésors. Le degré d'intégration est aussi optimal (égal à 1). Par contre, le degré d'indirection est un peu moins bon que les deux précédents car, même si le niveau d'incarnation est total (le mot de passe est entendue directement par l'utilisateur), l'indice temporel est moins bon (400 ms) dû aux traitements plus complexes, et donc plus long, pour l'analyse des mots par le système de reconnaissance vocale.

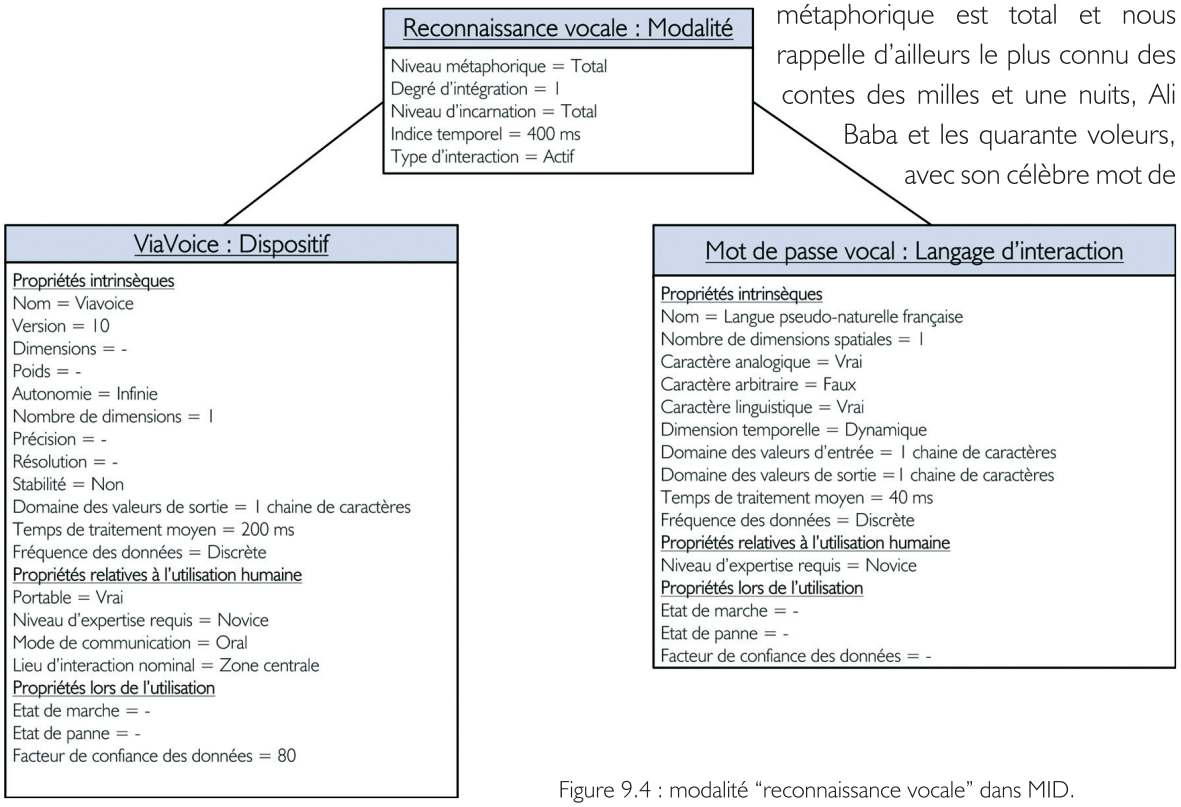


Figure 9.4 : modalité "reconnaissance vocale" dans MID.

2.1.2 Combinaisons des modalités d'interaction

Dans un premier temps, le cahier des charges définit une équivalence entre

les trois modalités d'interaction. La Figure 9.5 spécifie cette relation. Le temps de traitement moyen est nul car aucun mécanisme de fusion ne doit être employé pour gérer l'équivalence.

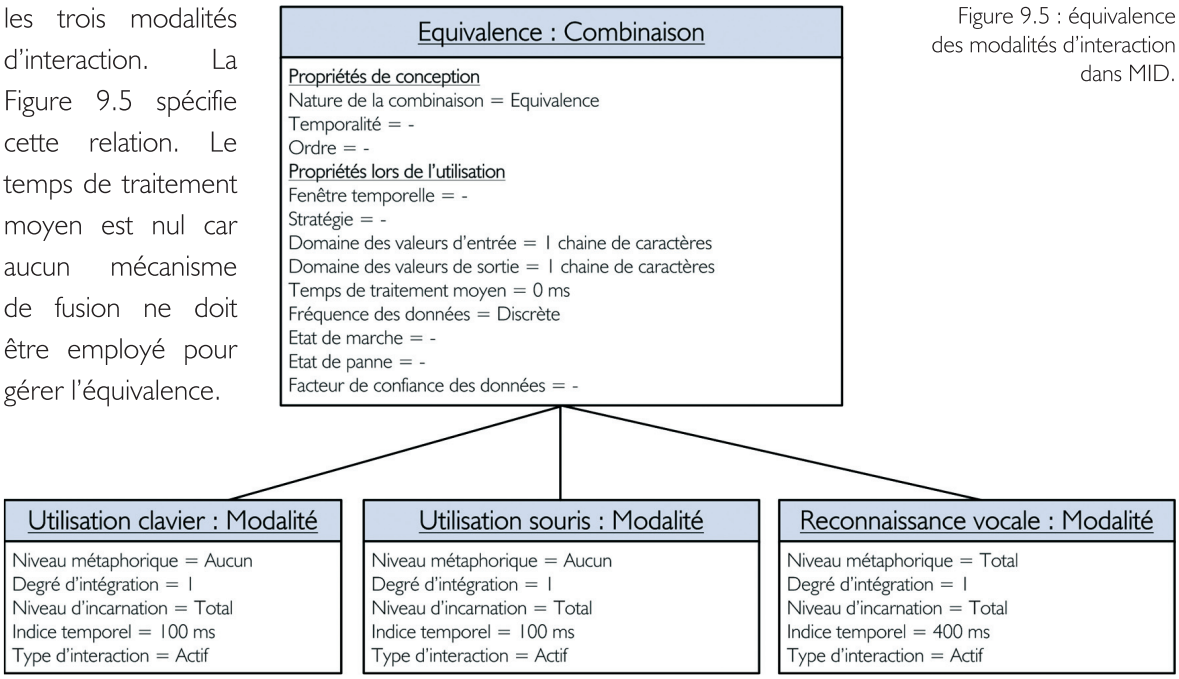


Figure 9.5 : équivalence des modalités d'interaction dans MID.

Dans un second temps, nous proposons une évolution de l'interaction multimodale. Il s'agit de proposer une équivalence entre deux cas de redondance. La Figure 9.6 montre la spécification de ces deux cas de redondance. Nous définissons les deux

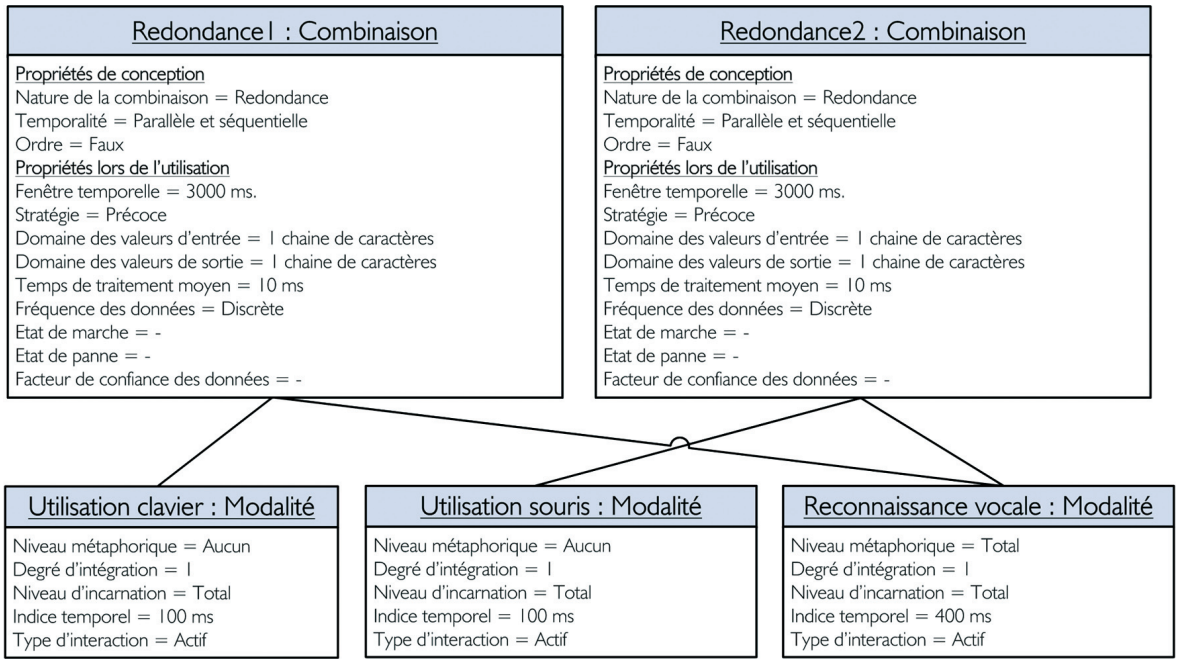


Figure 9.6 : deux cas de redondance des modalités d'interaction dans MID.

formes de redondance par des caractéristiques identiques. L'utilisateur final a trois secondes pour utiliser les deux modalités d'interaction de manière parallèle ou séquentielle. De plus, aucun ordre chronologique n'est demandé pour l'utilisation des deux modalités d'interaction redondantes.

2.2 Conception globale et détaillée

2.2.1 Architecture logicielle

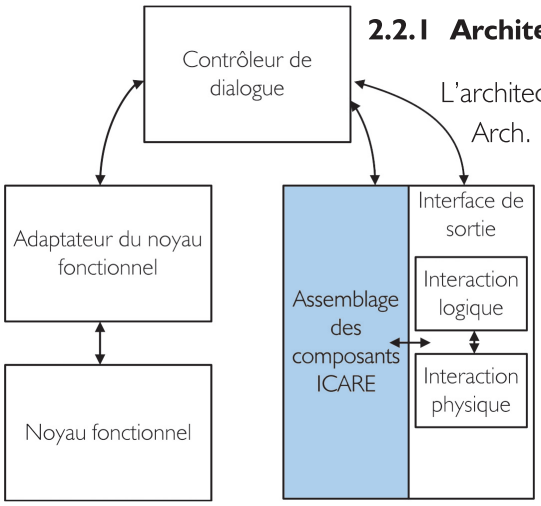


Figure 9.7 : architecture de MID.

L'architecture logicielle employée est une architecture de type Arch. La Figure 9.7 rappelle cette architecture et positionne l'assemblage des composants ICARE. Au vue de la simplicité du système qui ne contient qu'une seule tâche (identification de l'utilisateur), le contrôleur de dialogue, l'adaptateur du noyau fonctionnel et le noyau fonctionnel sont concis. L'interface de sortie gère principalement l'affichage (capture d'écran présentée à la Figure 9.1) et le rendu sonore de la reconnaissance ou non de l'utilisateur. La tâche d'identification est une simple méthode qui prend comme paramètre une chaîne de caractère représentant l'identification de l'utilisateur dont le profil complet (nom, prénom, etc.) est stocké dans le noyau fonctionnel.

2.2.2 Premier assemblage de composants

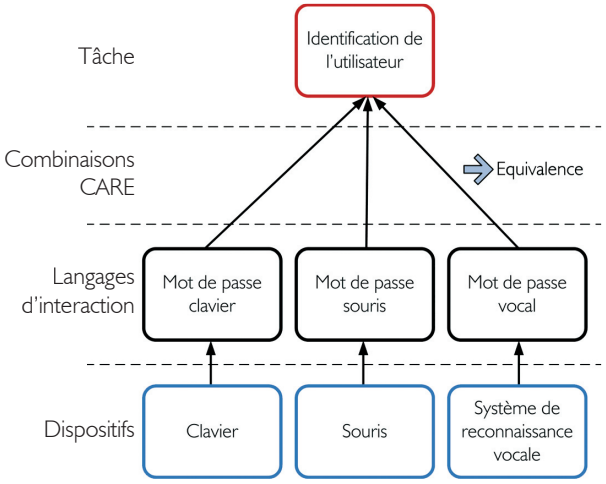


Figure 9.8 : premier assemblage de composants pour le système MID.

Suite à la phase de spécification, l'outil ICARE est utilisé pour assembler les différents composants correspondants. Ces composants logiciels ont été développés préalablement et intégrés à l'éditeur graphique d'ICARE par un simple ajout de l'archive Java correspondante dans un répertoire donné. La Figure 9.8 montre le premier schéma d'assemblage créé dans la zone d'édition de l'éditeur. Nous retrouvons, de bas en haut, les différents niveaux d'abstraction de l'interaction multimodale en entrée : composants Dispositifs, composants Langages d'interaction, combinaisons et représentation de la tâche.

Dans ce premier assemblage, l'équivalence entre les trois modalités d'interaction est représentée par les trois flèches qui convergent vers la même tâche (identification

de l'utilisateur). Les langages d'interaction transforment les données des différents dispositifs en une chaîne de caractères représentant le numéro d'identification de l'utilisateur. Par exemple, le composant mot de passe clavier récupère la suite des touches clavier utilisées et la transforme en une autre chaîne "utilisateur_1". Les autres composants langages d'interaction procèdent de la même manière. Les caractéristiques des composants identifiées à la section 2.1 de ce chapitre sont définies grâce au panneau de configuration de l'éditeur graphique (cf. chapitre 7).

2.2.3 Evolution, un nouvel assemblage

Afin de satisfaire l'évolution identifiée dans le cahier des charges, une nouvelle configuration de l'assemblage des composants est effectuée. La Figure 9.9 montre ce nouvel assemblage, où sont introduits deux composants de combinaison Redondance entre l'utilisation du clavier et la reconnaissance, puis entre l'utilisation de la souris et la reconnaissance vocale. Ces deux combinaisons sont équivalentes pour la tâche d'identification de l'utilisateur, laissant ainsi le choix à l'utilisateur d'utiliser l'une ou l'autre des redondances. Dans l'éditeur graphique, les composants Redondance sont mis en place par glisser-déposer et liés aux autres composants. Ensuite, ils sont paramétrés comme définis lors des spécifications. Cette évolution montre que la modification des formes de combinaison entre les modalités d'interaction demande un effort minimal. Dans le cas où cette nouvelle configuration ne convienne pas, il est envisageable de modifier à nouveau, et avec la même simplicité, l'assemblage des composants. Une redondance entre les trois modalités d'interaction peut être envisagée ; l'utilisation du composant Redondance/Equivalence peut aussi être choisie.

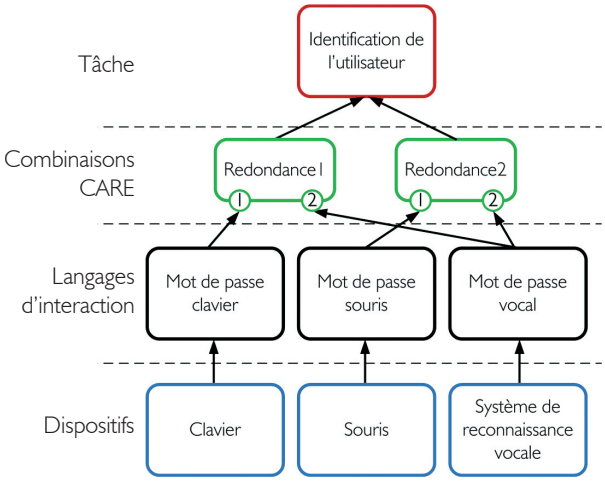


Figure 9.9 : redondance partielle des modalités d'interaction pour le système MID.

2.2.4 Codage

Nous retrouvons trois phases pour le codage :

- La première phase consiste à programmer manuellement les premiers composants Dispositifs et Langages d'interaction en se basant sur les classes mères de chaque catégorie (cf. chapitre 6). Comme présentés dans les chapitres suivants, ces composants sont réutilisés dans les autres systèmes interactifs.

- La seconde phase est la génération automatique du code correspondant à l'assemblage des composants à partir de l'éditeur. Un répertoire est créé et contient tous les fichiers nécessaires pour exécuter l'interaction multimodale conçue sur l'ordinateur cible. Le répertoire comprend ainsi les différents composants utilisés (archives Java), le fichier de configuration qui contient l'assemblage des composants grâce au mécanisme de "sérialisation" Java, le fichier permettant de démarrer, arrêter et accéder aux composants, et enfin le fichier de liaison avec le reste du système informatique.

- La troisième phase est la programmation manuelle reliant notre assemblage avec la méthode représentant la tâche dans le contrôleur de dialogue. Le langage Java est utilisé pour implémenter ce module. Il s'agit donc d'un simple appel de méthode où les données, résultant des langages d'interaction dans le cas d'équivalence ou des composants Redondance, sont passées en paramètre (cf. section 4.5 du chapitre 7). Cette troisième phase n'est effectuée qu'une seule fois, à moins de l'ajout ou du retrait d'une tâche. Il n'est donc pas nécessaire de toucher manuellement au code lors de la re-conception de l'interaction multimodale pour les tâches qui ont déjà été intégrées une première fois.

3 Résumé du chapitre 9

Le système interactif présenté dans ce chapitre est la première interaction multimodale réalisée avec notre outil. Il nous a permis de conforter notre démarche basée sur les composants logiciels en montrant, effectivement, qu'il est très aisé de mettre en œuvre plusieurs modalités d'interaction et de modifier les formes de combinaison. De plus, notre base de composants Dispositifs et Langages d'interaction compte ses premières unités.

En outre, ce cas concret montre l'apport de notre approche sur les phases de spécifications, de conception et de codage du cycle de développement logiciel. Le passage de l'équivalence entre les trois modalités d'interaction vers la redondance des modalités d'interaction illustre la capacité de notre outil à s'adapter aux démarches de prototypage rapide par la modification simple et rapide des formes de multimodalité.

Ce système interactif a été spécifié et conçu pour tester notre approche sur un cas simple, complètement détaillé dans ce chapitre. Les quatre autres systèmes développés avec ICARE, présentés dans les deux chapitres suivants, constituent des systèmes beaucoup plus conséquents en taille de code source, avec des tâches, des modalités d'interaction et des formes de multimodalité variées.

CHAPITRE 10
Systèmes interactifs du paradigme geste/parole

Ce chapitre présente deux systèmes interactifs intégrant des modalités d'interaction gestuelles et orales. Il s'agit d'un système permettant de situer l'adresse d'une personne sur un plan, appelé YellowPages, et d'un simulateur de trafic aérien en vol (ATC – Air Trafic Control). Nous les décrivons globalement avant de les détailler avec notre outil ICARE. Ces deux systèmes proposent des solutions différentes en termes d'assemblage de composants afin de répondre à des besoins semblables comme, notamment, la prise en charge de l'utilisation de la souris dans les interfaces WIMP (modalité gestuelle).

1 Système interactif YellowPages

Le système YellowPages sert d'étude de cas au projet VERBATIM, qui a pour sujet la vérification formelle et l'automatisation du test des interfaces multimodales. Le concept de ce projet repose sur le couplage de notre outil ICARE avec l'outil Lutess, une plate-forme pour le test de logiciels réactifs synchrones [Jourde 2006] (cf. annexe 4). Cette association permet de valider l'assemblage des composants ICARE en fonction des spécifications du système interactif cible.

Notre objectif est ici d'illustrer, par un exemple concret, comment le paradigme geste/parole est pris en compte au moment de la conception par les assemblages de composants ICARE. Nous présentons une description globale du système interactif avant de détailler les assemblages de composants mis en place.

1.1 Description globale

YellowPages permet de situer l'adresse d'une personne par sa visualisation sur un plan. La Figure 10.1 montre l'interface graphique du système interactif.

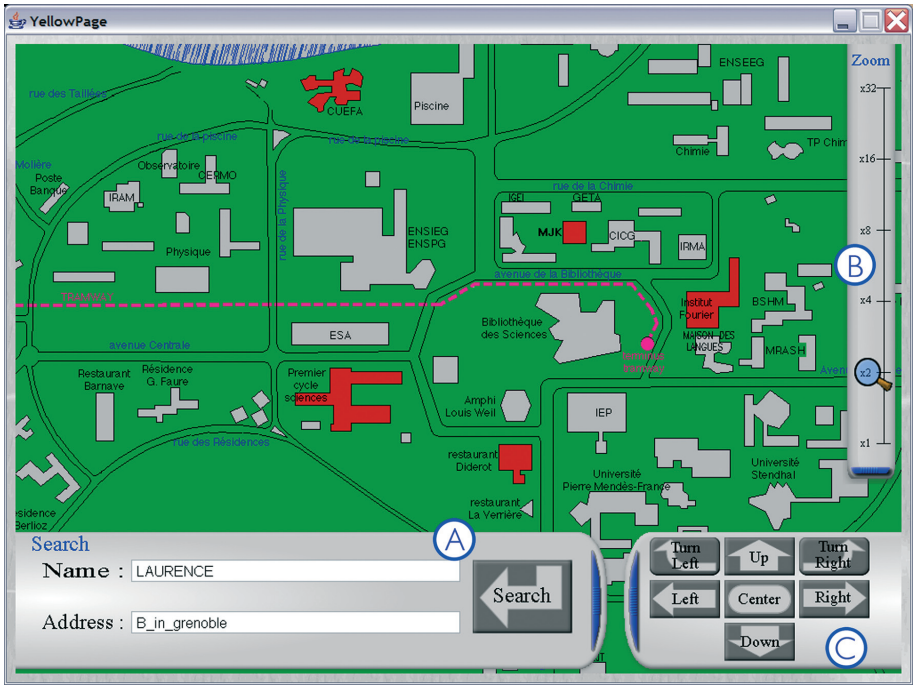


Figure 10.1 : interface graphique de YellowPages.

La zone A présente les champs nom et adresse que doit spécifier l'utilisateur. Pour ce faire, il peut utiliser de manière équivalente un système de reconnaissance vocale ou la combinaison classique du clavier et de la souris.

Après validation et la vérification de l'existence du nom et de l'adresse dans la base de données, le plan correspondant à l'adresse s'affiche dans la zone B. L'utilisateur peut alors manipuler ce plan par des commandes simples (zoom+, zoom–, déplacement haut/bas/gauche/droite) ou par des commandes plus complexes combinant geste et parole (zoom+ ici et zoom– ici). Les commandes simples sont effectuées de manière équivalente ou redondante, soit par l'utilisation de touches spécifiques au clavier (par exemple "page up" pour zoom+), par un mot prononcé oralement (par exemple "zoom") ou par l'utilisation de la souris sur des icônes graphiques dédiées (zone C). Les commandes complexes résultent d'une utilisation complémentaire du geste (pointeur de la souris qui spécifie le point central du zoom et correspondant au déictique "ici") et de la commande "zoom+ ici" ou "zoom- ici" spécifiée soit au clavier soit par la parole.

1.2 Conception : assemblage des composants

Nous présentons maintenant l'assemblage des composants dans l'éditeur ICARE. Nous avons regroupé les tâches qui demandent des données similaires (nombre et type identiques), ce qui permet de simplifier considérablement le schéma d'assemblage. Nous retrouvons ainsi trois groupes de tâches :

1. les tâches de spécification des champs texte (nom ou adresse) qui demandent le la spécification du champ à remplir puis la saisie de la valeur souhaitée.
2. les tâches des commandes simples comme le lancement de la recherche, le zoom ou encore le déplacement haut/bas/gauche/droite de la carte.
3. les tâches des commandes complexes (zoom+ ici et zoom– ici) qui demandent la fusion de la commande (zoom+ ici ou zoom– ici) et de la spécification de la position correspondant au déictique "ici".

1.2.1 Spécifications des champs texte

La Figure 10.2 présente l'assemblage des composants pour les tâches de spécification des champs texte. Il s'agit ici d'effectuer une fusion entre la spécification du champ à remplir et la saisie de la valeur. Pour ces deux phases, l'utilisateur peut utiliser soit la parole soit les modalités d'interaction classiques des interfaces WIMP (clavier et souris). Cet exemple montre que notre approche peut cohabiter avec les approches déjà existantes. Nous avons décidé de réutiliser les mécanismes intégrés à l'API Java pour la gestion de la souris et du clavier. La partie dessinée en pointillée illustre que les composants Dispositifs souris et clavier ne sont pas mis en œuvre ici car le module de sortie s'est directement abonné aux

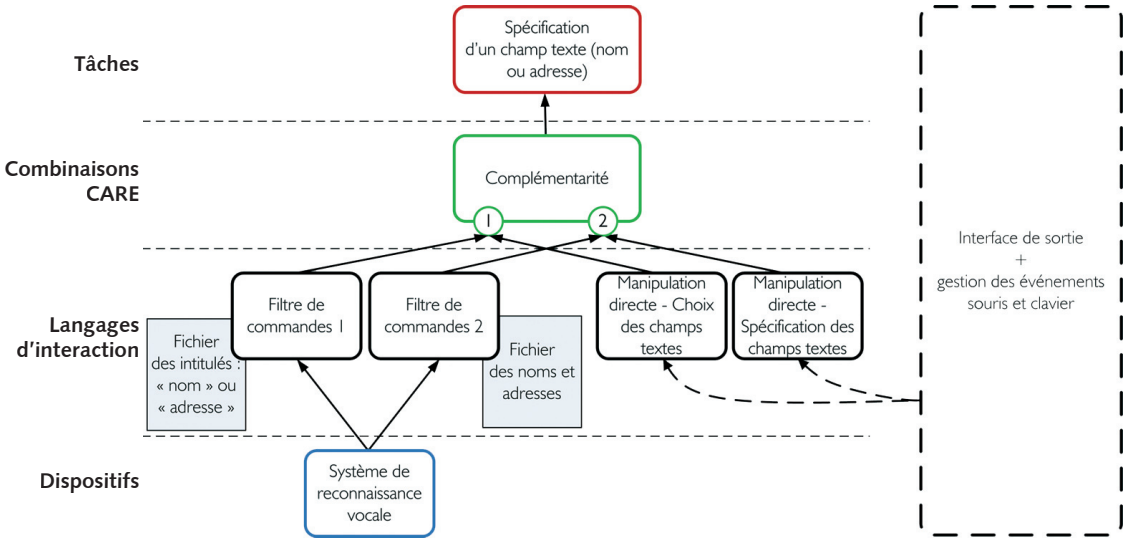


Figure 10.2 : spécification des champs texte de YellowPages.

événements souris et clavier gérés par Java, comme lors d'un développement classique d'interfaces WIMP. Nous le couplons ensuite avec deux composants Langages d'interaction d'ICARE. En effet, le module de sortie appelle directement les composants Langages d'interaction "Manipulation directe – Choix des champs textes" et "Manipulation directe – Spécification des champs textes".

D'autre part, la parole peut être utilisée pour déterminer le champ sélectionné par la modalité d'interaction (système de reconnaissance vocale, filtre de commandes 1) ou pour spécifier la valeur par la modalité d'interaction (système de reconnaissance vocale, filtre de commandes 2). Les filtres de commandes sont des composants qui prennent en compte un fichier dans lequel nous spécifions les chaînes de caractères qui sont correctes. Le filtre de commande 1 s'occupe donc de laisser passer les commandes identifiant le champ à sélectionner ("nom" ou "adresse") alors que le filtre de commande 2 traite les noms et les adresses valides.

L'utilisateur a donc quatre possibilités pour la spécification d'un champ texte de type <choix du champ, spécification de la valeur>. Il a le choix entre : <parole, parole>, <parole, WIMP>, <WIMP, parole> ou enfin <WIMP, WIMP>. La fenêtre temporelle du composant Complémentarité est infinie, permettant par exemple à l'utilisateur de cliquer dans le champ "nom" et d'attendre un temps assez long pour spécifier la valeur par la voix ou le clavier. De plus, nous obligeons l'utilisateur à choisir d'abord le champ avant d'en spécifier la valeur (attribut Ordre du composant Complémentarité défini à vrai).

1.2.2 Commandes simples

La Figure 10.3 présente l'assemblage des composants pour les tâches de commandes simples (zoom, déplacement de la carte, lancement de la recherche). Nous utilisons ici le composant Redondance/Equivalence entre l'utilisation du clavier, de la parole ou des moyens classique des interfaces WIMP.

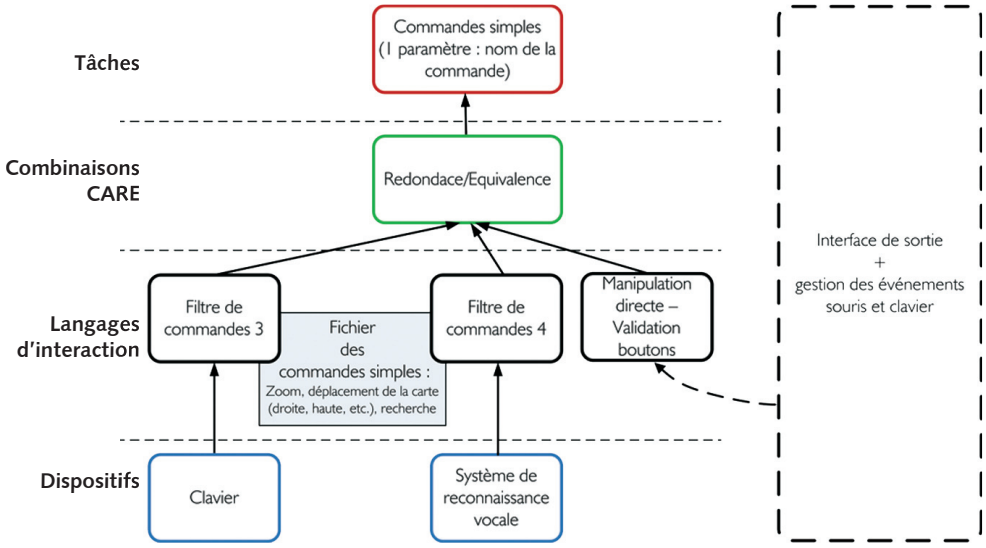


Figure 10.3 : commande simple dans YellowPages.

Nous remarquons que nous avons ajouté l'utilisation du clavier comme une modalité d'interaction classique de notre approche. En effet, certaines touches du clavier sont directement associées aux commandes. Par exemple, la touche "page down" permet de faire la commande zoom–. De la même manière toutes les commandes simples peuvent être déclenchées par une touche du clavier définie arbitrairement. Nous ne sommes plus ici dans le cas d'une interface WIMP classique, où il existe des standards pour l'usage du clavier, comme par exemple l'utilisation de la touche "tabulation" pour changer de champs. Par contre, ce dernier cas est traité, comme pour la manipulation des icônes incarnant les commandes simples par la souris, avec une liaison directe entre le module de sortie et le langage d'interaction "Manipulation directe – validation boutons".

Enfin, l'utilisateur peut aussi utiliser la reconnaissance vocale en prononçant les mots clés correspondant aux commandes simples (zoom+, zoom–, haut, etc.).

Le choix du composant Redondance/Equivalence permet à l'utilisateur d'utiliser le moyen qui lui convient le mieux. La fenêtre temporelle détectant les cas de redondance est définie à 3 secondes. Dans le cas où il effectue une redondance

entre deux modalités d'interaction, une seule commande est envoyée au reste du système. Ainsi, si l'utilisateur appuie par exemple sur "page up" pour effectuer un zoom+ et qu'il prononce au même moment les mots "zoom plus", la carte n'est zoomée qu'une seule fois. Si nous avions uniquement laissé l'équivalence entre les trois modalités d'interaction, le zoom de la carte aurait été réalisé deux fois de suite.

1.2.3 Commandes complexes

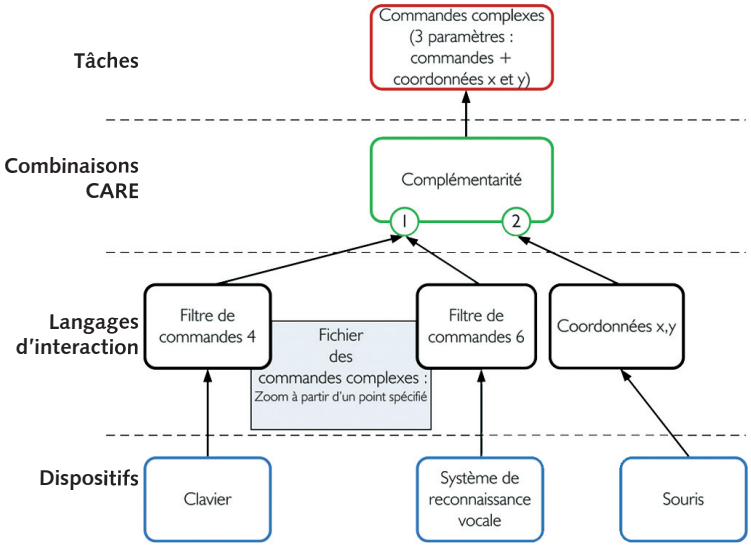


Figure 10.4 : commandes complexes dans YellowPages.

La Figure 10.4 montre l'assemblage des composants pour les tâches complexes ("zoom + ici" et "zoom – ici") offrant une combinaison entre le geste (pointeur de la souris sur l'endroit central du zoom) et la commande (zoom+ ici ou zoom– ici) réalisé soit au clavier, soit par la parole (Equivalence). Dans ce schéma d'assemblage des composants, nous n'utilisons plus les techniques des interfaces WIMP. La modalité d'interaction (souris, coordonnées x et y) transmet la position du point central du zoom. Les modalités d'interaction (clavier, filtre de commandes 4) et (système de reconnaissance vocale, filtre de commandes 6) transmettent les commandes "zoom+ ici" ou "zoom– ici" et sont fusionnés dans le composant Complémentarité avec le geste de la souris. La fenêtre temporelle est définie à 1 seconde et le geste peut aussi bien être spécifié avant ou après la commande (attribut Ordre à faux).

1.3 Synthèse

Le système YellowPages présente différentes combinaisons entre les modalités d'interaction du geste et de la parole. Il montre aussi une solution concernant le couplage de l'interface d'entrée avec celle de sortie, lorsqu'elles sont fortement liées comme dans les interfaces WIMP, en soulignant la compatibilité de l'outil ICARE avec les boîtes à outils existants.

Le système interactif suivant illustre aussi les interfaces geste/parole. Contrairement au système YellowPages, la gestion de la souris dans le cadre d'une interface WIMP est décrite ici avec des composants ICARE.

2 Simulateur ATC

Ce système interactif est réalisé dans le cadre du projet INTUITION (PEA – Projet d'Etude Amont de la DGA). Il s'agit d'un partenariat entre l'industriel THALES-Avionics et les laboratoires de recherche en IHM du LIIHS (Toulouse) et du LIMSI (Orsay). Dans le cadre de ce projet, un processus de conception pour les interfaces multimodales entrée et sortie est présenté dans [Bastide 2005]. Le simulateur de trafic de contrôle aérien (ATC) permet de valider l'approche globale du projet.

2.1 Description globale

Le projet INTUITION décrit de façon détaillée une architecture générique pour toutes les interfaces multimodales. La solution proposée se base sur l'architecture de référence ARCH et regroupe notre outil ICARE pour l'interaction multimodale en entrée, l'outil Most [Rousseau 2005] pour l'interaction multimodale en sortie et l'outil PetShop [Bastide 1998] pour la réalisation du contrôleur de dialogue et de l'adaptateur du noyau fonctionnel.

Le simulateur ATC suit donc cette architecture et propose un simulateur innovant, proposant une visualisation en trois dimensions de l'espace aérien. La Figure 10.5 montre l'interface graphique du simulateur. Le secteur de vol et deux avions le traversant sont représentés. Plusieurs modalités d'interaction en entrée sont utilisées pour contrôler la vue en trois dimensions. Nous proposons un capteur d'orientation permettant de modifier l'angle de vision ou l'utilisation du clavier pour déplacer la vue (droite, gauche, haut, bas) et effectuer des zooms. Dans cette section, nous ne détaillons pas cet aspect mais nous nous concentrons sur les tâches répondant à une utilisation du geste et de la parole. Ces tâches concernent les requêtes qui sont envoyées par le contrôleur aérien aux pilotes des avions rentrant dans leur secteur de vol. Les requêtes sont multiples. Elles peuvent correspondre à une demande de changement d'altitude, de cap ou de vitesse. Elles sont toutes constituées de la même manière et comprennent trois paramètres: le numéro d'avion concerné, le paramètre à changer (altitude, cap, vitesse) et la nouvelle valeur

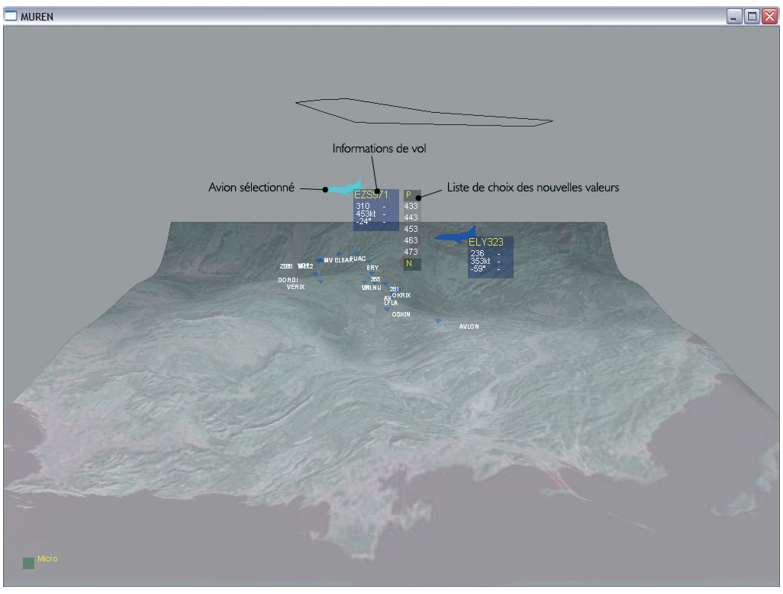


Figure 10.5 : interface graphique du simulateur ATC.

souhaitée. Une requête est constituée comme suit : <numéro d'avion, paramètre de vol, nouvelle valeur>. Par exemple, le contrôleur demande à l'avion identifié par "EZS971" de changer son altitude à 300 pieds. Le contrôleur sélectionne l'avion par un clic gauche de la souris sur l'avion puis il a le choix entre l'utilisation de la souris ou la parole (Equivalence) pour spécifier le paramètre de vol et la nouvelle valeur. Pour le choix du paramètre à modifier, le contrôleur peut cliquer sur le paramètre correspondant dans l'interface graphique et qui s'affiche à côté de l'avion sélectionné ou il peut prononcer la commande, comme par exemple "climb to" pour changer l'altitude de l'avion. Pour la nouvelle valeur souhaitée, il peut cliquer sur la valeur qui s'affiche dans la liste déroulant à côté des informations de vol ou il peut la prononcer. Les trois paramètres des requêtes peuvent donc être spécifiés de la manière suivante : <souris, souris, souris> ou <souris, parole, parole> <souris, souris, parole> ou <souris, parole, souris>.

2.2 Conception : assemblage des composants pour la spécification des requêtes

Comme dans le système YellowPages, nous avons regroupé les tâches correspondant aux différentes requêtes (changement d'altitude, de cap ou de vitesse) car elles présentent le même nombre et le même type de paramètres. Nous définissons ainsi qu'un seul assemblage de composants pour l'ensemble de ces requêtes, présenté à la Figure 10.6.

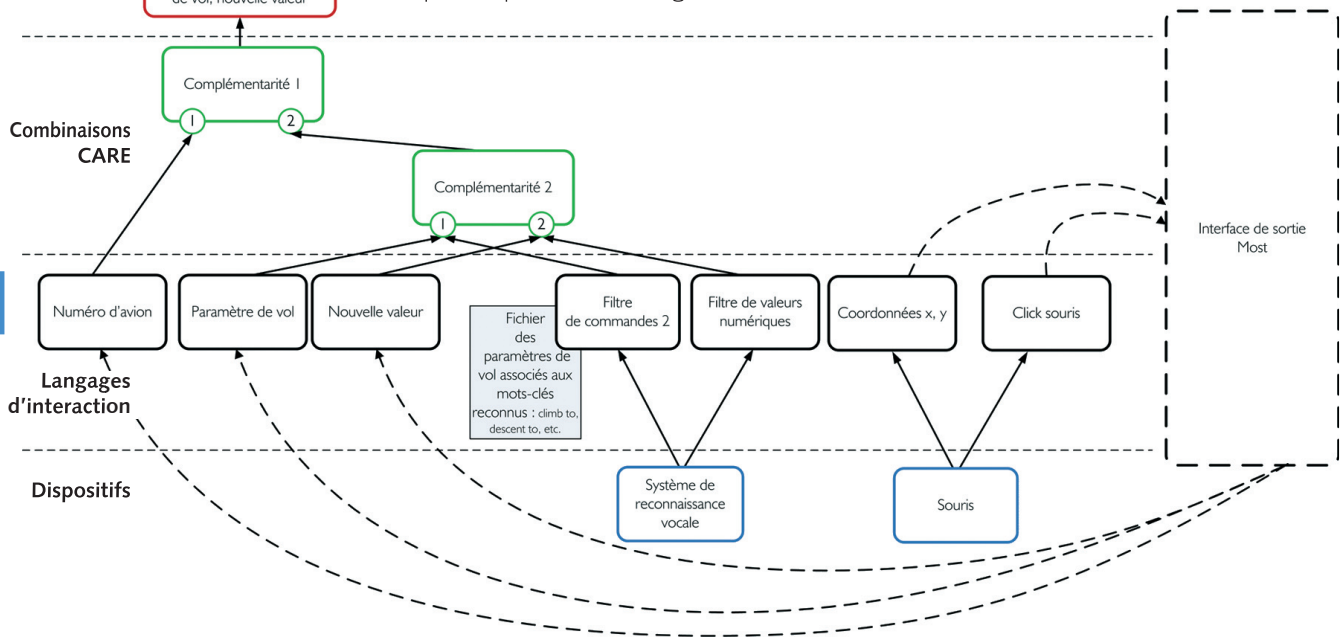


Figure 10.6 : assemblage de composants pour les requêtes du simulateur ATC.

Les requêtes résultent donc de la fusion du numéro de l'avion sélectionné, du paramètre de vol à modifier et de la nouvelle valeur souhaitée. Cette complémentarité est illustrée par les deux composants Complémentarité (à noter qu'il est possible de n'en utiliser qu'un seul avec trois ports au lieu de deux). Le port 1 du composant Complémentarité 1 permet de récupérer le numéro de l'avion, alors que le paramètre à modifier et la nouvelle valeur sont liés au port 2, après avoir été fusionnés dans le composant Complémentarité 2. Le paramètre de vol renseigne le port 1 du composant Complémentarité 2 soit par l'utilisation de la souris dans l'interface graphique, soit par la modalité d'interaction équivalente de la parole (système de reconnaissance vocale, Filtre de commandes 2). La valeur souhaitée arrive sur le port 2 après avoir été spécifiée par la souris ou par la parole (Système de reconnaissance vocale, Filtre de valeurs numériques).

Pour ce système, nous n'avons pas utilisé les mécanismes déjà existants pour la gestion des événements souris. Nous avons ainsi mis en œuvre deux modalités d'interaction pour la gestion de la souris. Elles utilisent le même dispositif (la souris) mais deux langages d'interaction sont définis : le premier traite les coordonnées de la souris par rapport à l'écran et le second est dédié aux pressions sur les boutons de la souris. Ces deux composants Langages d'interaction appellent l'interface de sortie réalisée avec l'outil Most. Ce dernier renvoie aux composants Langages d'interaction le numéro d'avion, le paramètre de vol ou la nouvelle valeur sélectionnée, avec la souris, sur l'interface graphique.

2.3 Synthèse

Le simulateur ATC présente une complémentarité entre le geste et la parole. Il présente aussi une autre solution que celle proposée dans le système YellowPages pour traiter des interactions entrée et sortie fortement couplées, comme c'est le cas dans une interaction WIMP.

3 Résumé du chapitre 10

Les points remarquables des deux systèmes YellowPages et simulateur ATC intégrant une interaction de type geste/parole, sont les suivants :

- Tout d'abord, les tâches qui demandent des données de même nature (nombre de données et types identiques) sont regroupées et simplifient le schéma d'assemblage des composants. Par exemple dans YellowPages, il est préférable d'avoir un regroupement des tâches simples que d'avoir à définir le même assemblage de composants pour la tâche "zoom+" puis pour la tâche "zoom–", puis pour "déplacement haut", etc. Il en est de même pour toutes les requêtes du simulateur ATC.
- D'autre part, nous présentons qu'il est possible de prendre en compte les interfaces WIMP présentant un fort couplage entre l'interface d'entrée et celle de sortie. Pour ce faire, nous mettons en place un pont entre les composants ICARE et les modules de l'interface de sortie, comme lors de l'utilisation de la souris dans le simulateur ATC. Nous montrons également que l'outil ICARE est compatible avec des outils existants, comme c'est le cas pour la gestion des événements souris et clavier des interfaces WIMP Java utilisées dans le système YellowPages.
- Enfin, plus spécifiquement aux interfaces geste/parole, nous avons montré que différentes combinaisons étaient possibles entre ces deux modalités d'interaction. Ces modalités d'interaction peuvent être utilisées de manière équivalente, redondante ou encore complémentaire.

CHAPITRE 11
Systèmes interactifs de réalité augmentée

Ce chapitre présente deux systèmes interactifs où la multimodalité est utilisée pour permettre la fusion harmonieuse des mondes numérique et physique, véhiculée par le paradigme de la réalité augmentée. Il s'agit du système MEMO, qui met en œuvre des post-its géo-localisés à l'instar des travaux présentés dans [Persson 2001], et d'un simulateur d'avion de combat.

11 MEMO

Le système interactif MEMO est un système de réalité augmentée que nous avons développé en interne à notre laboratoire, afin de valider notre outil ICARE pour la réalité augmentée [Bouchet 2004a]. Il met en œuvre des modalités d'interaction passives pour faire correspondre les mondes numérique et physique. Nous le décrivons globalement avant de détailler l'assemblage des composants réalisé avec notre outil ICARE.

11.1 Description globale

Le système interactif MEMO permet à un utilisateur d'annoter des lieux physiques avec des mémos numériques. Il peut ensuite les manipuler en se déplaçant dans le monde physique. Les mémos peuvent être lus, supprimés et déplacés par d'autres utilisateurs. Dans ce prototype, les mémos sont déjà créés et l'utilisateur ne peut pas en ajouter. La plate-forme matérielle est constituée d'un ordinateur portable, de lunettes de réalité augmentée semi-transparentes, d'une souris, d'un capteur d'orientation et d'un microphone.

La Figure 11.1 montre la vision d'un utilisateur. Ce dernier perçoit le monde physique au travers des lunettes de réalité augmentée semi-transparentes. Dans cet exemple, elle/il se trouve devant l'un des bâtiments du campus universitaire de Grenoble et visualise deux mémos numériques. Le mémo situé au centre, sous son curseur de visé, vient d'être déplacé par l'utilisateur, car comme nous l'indique le retour d'information, le mémo vient d'être posé à un nouvel emplacement. Il est à noter que l'utilisateur manipule uniquement les mémos qui se trouvent dans son viseur désigné par la croix centrale.

Figure 11.1 : vision de l'utilisateur avec le système interactif MEMO (image reconstituée).

1.2 Conception de l'interaction multimodale en entrée

La Figure 11.2 montre l'assemblage des composants ICARE pour cette interaction multimodale en entrée. Deux tâches (ou groupe de tâches) sont identifiées :

- La mise à jour de la visualisation 3D (T1) : cette tâche doit connaître l'orientation et la localisation de l'utilisateur pour que le système affiche, dans les lunettes semi-transparentes les mémos visibles en superposition du monde physique et en fonction de la position et de l'orientation de l'utilisateur mobile.
- La manipulation d'un mémo (prendre, déposer, supprimer) (T2) : nous avons regroupé ces trois tâches élémentaires pour ne spécifier qu'un seul assemblage de composants.

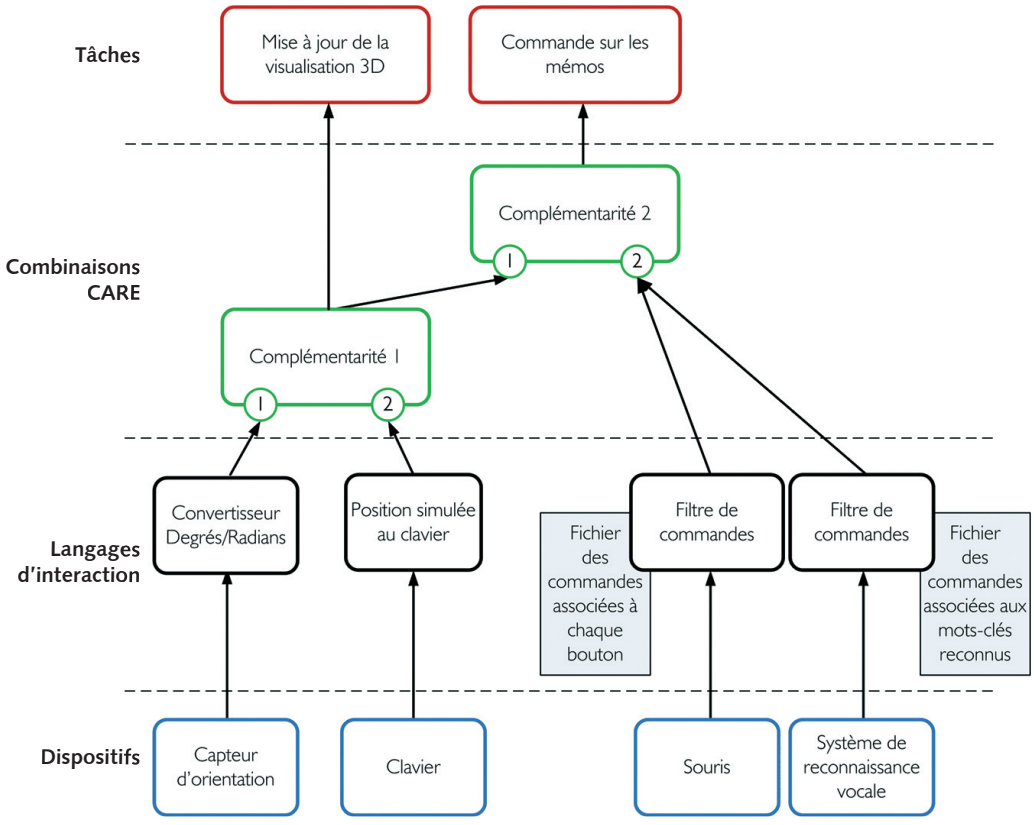


Figure 11.2 : assemblage des composants ICARE de MEMO.

1.2.1 Mise à jour de la visualisation 3D en superposition du monde réelle

Deux modalités d'interaction passives et complémentaires, informant de l'orientation et de la localisation de l'utilisateur, sont employées ici pour mettre en correspondance la scène numérique 3D et le monde physique que l'utilisateur perçoit au travers de ces lunettes semi-transparentes. Nous les visualisons sur le côté de gauche de la Figure 11.2.

La modalité d'interaction "orientation" est représentée par le couple de composants (capteur d'orientation, convertisseur degrés/radians). Le dispositif que nous utilisons ici est un magnétomètre qui est fixé sur la tête de l'utilisateur. Il fournit trois angles en degré. Le langage d'interaction convertit ces angles en degré par des angles en radians correspondant au format employé par les algorithmes de traitement de la scène 3D que nous utilisons.

La modalité d'interaction "localisation" est simulée par la technique de magicien d'Oz. Il s'agit en réalité d'un compère qui utilise les flèches d'un clavier pour informer au système informatique que l'utilisateur se déplace. Cette modalité d'interaction se traduit par le couplage des composants (Clavier, Position simulée au clavier).

Les données de ces deux modalités d'interaction sont fusionnées pour connaître précisément où l'utilisateur regarde. Elles sont reliées au composant Complémentarité 1 dont la fenêtre temporelle a été définie à 30 ms correspondant au temps de traitement de la modalité "orientation", transmettant des données en continu.

1.2.2 Manipulation d'un mémo

Deux modalités d'interaction actives et équivalentes sont utilisées pour la manipulation des mémos. En effet, les commandes peuvent être spécifiées en utilisant des mots clés à prononcer ou par l'utilisation des boutons de la souris (chaque bouton est associé à une commande : bouton gauche pour prendre, bouton du milieu pour déposer et bouton droit pour supprimer). Nous obtenons donc une première modalité d'interaction (souris, filtre de commandes) qui transforme les événements de la souris en commande (prendre, déposer, supprimer) et une deuxième modalité d'interaction correspondant à l'usage de la parole par le couple (Système de reconnaissance vocale, Filtre de commandes).

Pour manipuler un mémo, l'utilisateur doit le viser grâce au curseur en forme de croix présenté dans les lunettes de semi-transparentes (cf. Figure 11.1). Pour

savoir exactement quel mémo est visé par la commande de l'utilisateur, nous la fusionnons avec l'orientation et la localisation de l'utilisateur à l'aide du composant Complémentarité 2, qui définit une fenêtre temporelle de 100ms.

Pour une meilleure compréhension, nous décrivons ce qu'il se passe dans l'assemblage des composants lors de la suppression d'un mémo. Toutes les 3 millisecondes, la modalité passive "orientation" fournit trois réels correspondant à l'orientation 3D en radian (cap, tangage, roulis). Avec la même fréquence, la modalité passive "localisation" fournit aussi trois réels mais correspondant cette fois-ci à la position 3D de l'utilisateur (x, y, z). Le composant de Complémentarité 1 réalise la fusion de six données qui sont envoyées au composant suivant. Il s'agit ici du composant Complémentarité 2. Si l'estampille de temps de l'événement correspondant à la commande <supprimer>, reçue de l'une des deux modalités actives équivalentes (manipulation des boutons de la souris ou parole), appartient à la même fenêtre temporelle que les six réels des deux modalités passives, alors les deux événements sont combinés. Le composant Complémentarité 2 envoie la commande complète <supprimer, six réels> au contrôleur de dialogue du système qui détermine par appel au Noyau fonctionnel quel mémo doit être supprimé.

1.3 Synthèse

Le système interactif MEMO est un exemple de réalité augmentée qui met en évidence une solution réutilisable permettant de superposer les mondes numérique et physique. Afin de déterminer précisément ce que voit l'utilisateur, la solution qui est proposée ici est de combiner, de façon complémentaire, deux modalités d'interaction passives nous informant de la position et de l'orientation de l'utilisateur.

Le système interactif suivant appartient également au domaine de la réalité augmentée et montre une solution similaire à celle présentée par le système MEMO, avec des composants de modalité d'interaction différents.

2 Simulateur d'avion de combat

Le dernier système interactif multimodal que nous présentons est de loin le plus conséquent en nombre de lignes de code et de tâches. Il s'agit d'un simulateur d'avion de combat réalisé pour le compte du projet INTUITION (DGA, THALES-Avionics, LIIHS, LIMSI) [Bouchet 2004c] validant, comme pour le simulateur ATC présenté au chapitre 10, le processus de développement et l'outillage prévus spécifiquement pour la conception et le développement d'interfaces multimodales [Bastide 2005]. A cette occasion, des ingénieurs de chez Thales-Avionics ont utilisé l'outil ICARE pour spécifier l'interaction multimodale du simulateur, nous permettant de tester notre approche avec des concepteurs/développeurs autres que ceux de notre laboratoire. Nous décrivons globalement le simulateur avant de détailler précisément les assemblages de composants réalisés avec notre outil ICARE.

2.1 Description globale

Le Simulateur d'avion de combat (SAC) est un système temps-réel utilisé chez THALES-Avionics pour étudier les techniques d'interaction futures pouvant être embarquées dans le cockpit. La Figure 11.3 montre un pilote testant le SAC. Il est assis dans un cockpit d'avion reconstitué et visualise le paysage à l'aide de trois grands écrans positionnés dans son champs de vision. Les tâches qu'il doit réaliser incluent :

- Piloter l'avion : le pilote suit une trajectoire prédéfinie qu'il adapte en fonction de divers paramètres comme les conditions météorologiques, géographiques et les caractéristiques de l'avion.
- Naviguer : le pilot crée ou modifie le plan de vol en entrant par exemple des nouveaux points de route.

Figure 11.3 : pilote dans le simulateur d'avion de combat temps-réel.

- Gérer l'avion : le pilote doit constamment vérifier les systèmes de l'avion (fuel, systèmes hydrauliques, systèmes électriques, etc.).
- Gérer la mission : par exemple, le pilote doit se protéger des attaques.
- Communiquer avec les contrôleurs du trafic aérien et les autres pilotes de la patrouille.

Alors que la liste des tâches est générale, des sous-tâches plus concrètes peuvent être identifiées. Par exemple, la tâche "positionner une marque au sol" peut être une sous-tâche de la tâche générale "gérer la mission". En identifiant les tâches élémentaires (ou sous-tâches), nous considérons une variété de modalités d'interaction en entrée et de formes de combinaison pouvant être développées en utilisant notre outil ICARE. Pour ce faire, un ensemble de modalités d'interaction en entrée basées sur différents dispositifs d'entrée sont disponibles :

- Le HOTAS (Hands On Throttle And Stick) représente les deux joysticks (un pour chaque main) qui permettent de piloter l'avion et effectuer certaines commandes, comme par exemple "positionner une marque au sol".
- Le viseur de casque permet au pilote de sélectionner une cible dans le monde physique. Il se base sur plusieurs dispositifs renseignant l'orientation et la localisation de l'avion et du pilote.
- Le système de reconnaissance vocale pour réaliser certaines commandes comme "positionner une marque au sol".
- La surface tactile positionnée entre les jambes du pilote pour spécifier des commandes par manipulation directe, comme par exemple la mise en route d'un équipement.

En plus des multiples modalités d'interaction en entrée disponibles, plusieurs contextes relatifs aux missions militaires sont identifiés. Par exemple, une mission peut comprendre des phases de navigation relativement calmes, des phases de combat très stressantes, etc. Les interfaces multimodales, par la flexibilité qu'elles offrent, jouent un rôle central pour gérer ces contextes. En effet, le pilote peut changer de modalités d'interaction ou de forme de multimodalité en fonction du contexte courant, du niveau de stress ou des paramètres physiques (par exemple lorsqu'il est exposé à une forte accélération).

Pour chacune des sous-tâches, les ingénieurs de chez Thales-Avionics ont utilisé l'outil ICARE pour spécifier l'interaction multimodale du simulateur. Nous ne détaillons pas l'ensemble des assemblages de composants mais nous allons nous focaliser sur les tâches mettant en œuvre le paradigme de réalité augmentée. Nous décrivons ainsi l'assemblage de composants correspondant à deux tâches relatives au marquage d'une cible au sol. Les assemblages de composants des autres tâches sont décrits dans [Bouchet 2005].

2.2 Conception de l'interaction multimodale en entrée pour les tâches de réalité augmentée

La Figure 11.4 montre l'assemblage des composants ICARE correspondant aux tâches de réalité augmentée. Nous remarquons qu'il est très proche de celui présenté pour le système MEMO montré précédemment. Nous retrouvons deux tâches : celle permettant de mettre à jour le paysage dans lequel évolue le pilote et celle permettant le marquage d'un point au sol par le pilote pendant un vol. Aujourd'hui, il n'existe que cette commande mais d'autres similaires peuvent être ajoutées. Si cela était le cas, nous regrouperions les tâches demandant des données identiques comme lors des systèmes multimodaux précédents.

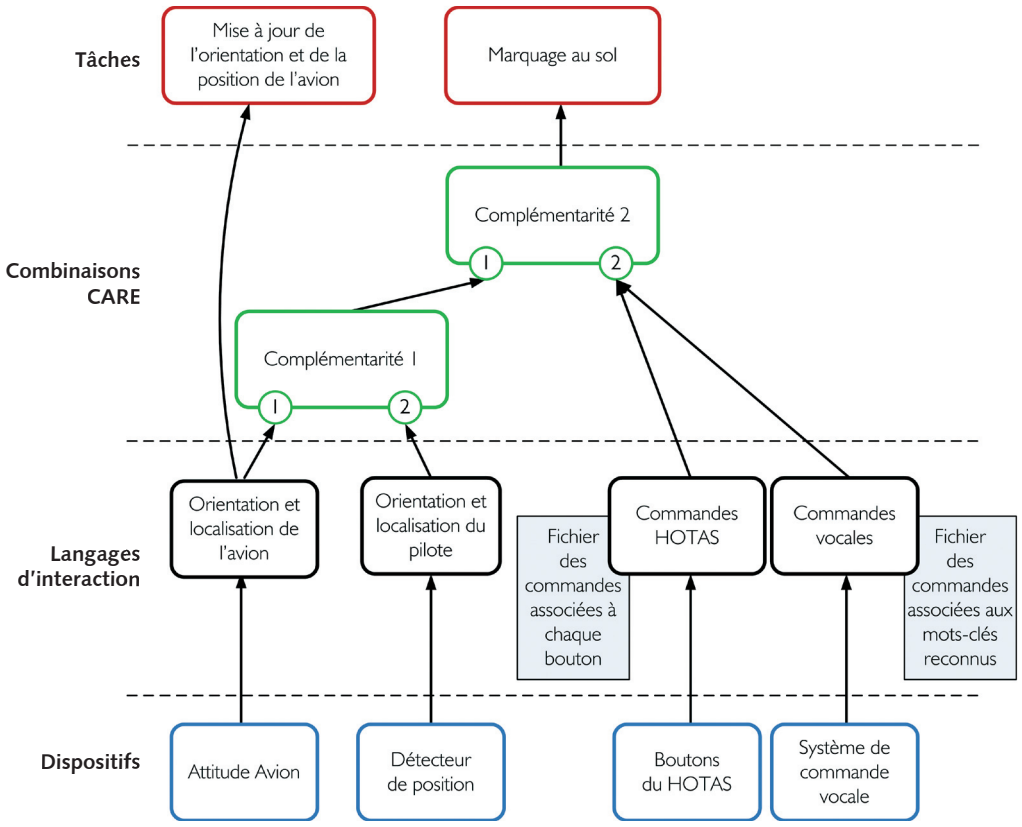


Figure 11.4 : assemblage de composants ICARE pour les tâches de réalité augmentée.

Quatre modalités d'interaction sont mises en œuvre :

- M1 = (Attitude Avion, Orientation et position de l'avion) est une modalité d'interaction passive qui transmet l'orientation et la position de l'avion dans le repère terrestre.

- M2 = (Détecteur de position, Orientation et position du pilote) est une modalité d'interaction passive représentant l'orientation et la position du pilote en relatif par rapport à l'avion.
- M3 = (HOTAS, Commandes HOTAS) est une modalité d'interaction active qui permet au pilote de spécifier une commande à effectuer en appuyant sur le bouton du joystick correspondant.
- M4 = (Système de reconnaissance vocale, Commandes vocales) est la modalité d'interaction active correspondant à l'usage de la parole. Des mots-clés sont définis. Pour positionner une marque au sol, le pilote doit prononcer le mot "mark".

M1 et M2 sont utilisés de manière complémentaire pour mettre à jour le paysage que visualise le pilote et pour détecter la cible sélectionnée par le pilote dans son viseur de casque : elles sont reliées par le composant Complémentarité 1. Contrairement au système MEMO, le SAC fait intervenir un dispositif fournissant directement l'association des données d'orientation et de localisation alors qu'il existait deux dispositifs dans MEMO, un pour l'orientation et un pour la localisation.

Ensuite, le pilote a le choix entre deux modalités d'interaction pour positionner la marque au sol. Il peut utiliser M3 ou M4 qui sont fonctionnellement équivalentes. Pour obtenir la commande complète, la commande "positionner une marque au sol" provenant de M3 ou M4 doit être combinée (composant Complémentarité 2) avec le point ciblé par le pilote à travers son viseur de casque (M1 et M2). La commande complète obtenue est ensuite transmise au contrôleur de dialogue.

2.3 Synthèse

Le système SAC montre une nouvelle fois que la prise en charge du paradigme de réalité augmentée est possible avec notre outil ICARE, permettant d'intégrer des modalités d'interaction passives nécessaires. Il montre une autre solution que celle proposée dans MEMO pour déterminer la direction du regard de l'utilisateur. Enfin, il permet de tester notre approche dans un environnement temps-réel.

3 Résumé du chapitre 11

Les deux systèmes interactifs présentés dans ce chapitre intègrent des modalités d'interaction passives pour réaliser la fusion des mondes numérique et physique (ou réalité augmentée). Ces deux systèmes se distinguent par leur nombre de lignes de code, bien plus important dans le SAC. MEMO a entièrement été développé dans notre laboratoire. Pour le SAC, seule la partie interactive en entrée est réalisée avec notre outil ICARE, testant ainsi, avec succès, la capacité de l'interaction multimodale créée pour se coupler avec les autres parties du système interactif développées selon d'autres approches.

Nous remarquons que les schémas d'assemblage des deux systèmes sont très proches. Les différentes modalités d'interaction sont toujours combinées de façon complémentaire, mais les modalités d'interaction elles-mêmes sont très différentes.

D'une part, la correspondance entre le monde numérique et le monde physique est assurée par la complémentarité de plusieurs modalités d'interaction passives, situant l'utilisateur par son orientation et sa position. Deux solutions différentes sont proposées. Dans MEMO, la modalité d'interaction d'orientation est directement couplée avec celle de localisation car elles se basent toutes les deux sur le même repère terrestre alors que dans SAC, nous utilisons deux autres modalités d'interaction. La première renseigne l'orientation et la position de l'avion par rapport au repère terrestre et la deuxième détermine l'orientation et la position du pilote définie dans le repère de l'avion. Afin de connaître précisément l'orientation et la position du pilote dans le repère terrestre, ces deux modalités d'interaction sont combinées de manière complémentaire.

D'autre part, les commandes portant sur un objet numérique positionné dans le monde physique doivent aussi être combinées avec l'orientation et la position de l'utilisateur pour déterminer précisément l'objet ciblé lorsque l'utilisateur spécifie sa commande.

Conclusion

Conclusion

Centré sur les systèmes multimodaux en entrée, notre travail contribue au domaine de l'ingénierie logicielle des interfaces utilisateur sous deux formes complémentaires : un modèle conceptuel de la multimodalité, qui inclut les points de vue système et utilisateur, et une opérationnalisation de ce modèle pour définir un outil de conception et de développement de l'interaction multimodale. En reprenant le modèle de Gaines, présenté en introduction et montré à la Figure 12.1, nous rappelons que notre contribution dans le domaine de l'interaction multimodale en entrée se situe donc au niveau des phases de réplication et d'empirisme, amenant ensuite à l'établissement de théories sur la multimodalité, après étude des systèmes causals issus des expériences. Notre objectif principal est ainsi d'accompagner le passage d'une innovation à la théorie, en fournissant des outils conceptuels pour les phases de réplication et d'empirisme : "*a breakthrough in one technology is triggered by a supporting technology as it moves from its research to its empirical stage*" [Gaines 1991].

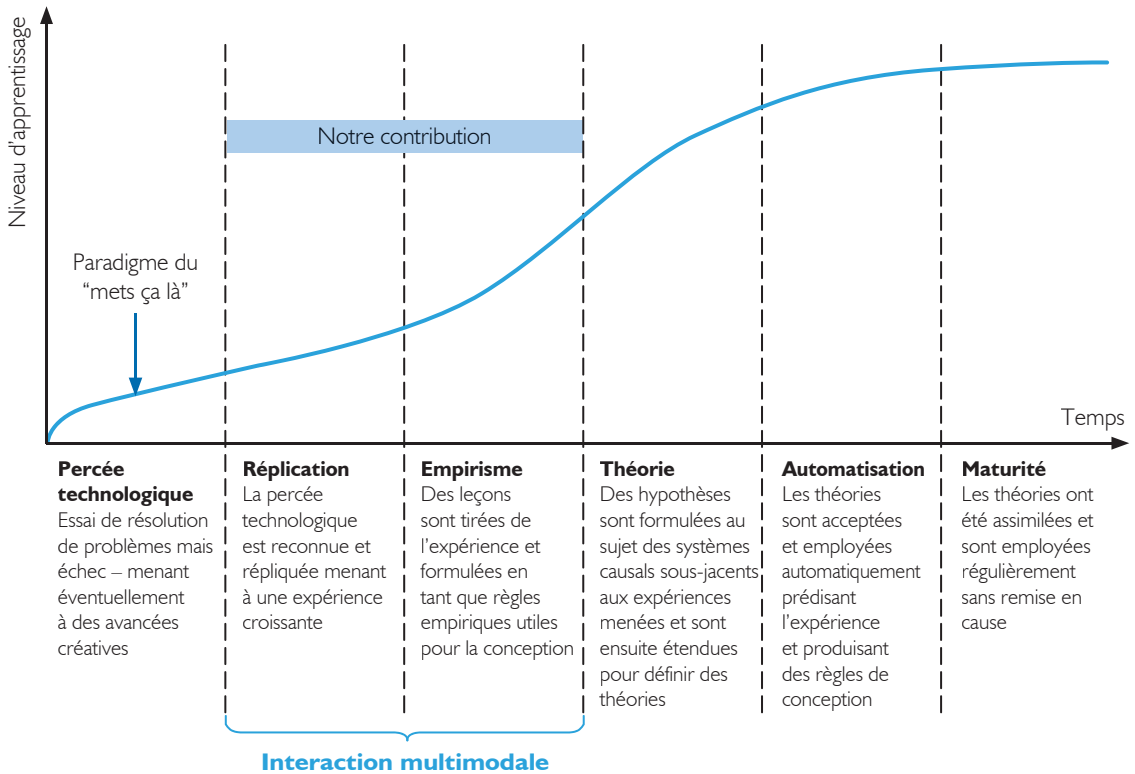


Figure 12.1 : le modèle de Gaines sur l'évolution du niveau d'apprentissage d'une technologie, adapté à l'interaction multimodale d'après [Gaines 1991].

En conclusion, nous résumons notre contribution, puis nous exposons un ensemble de limites à nos travaux et identifions des perspectives à court et long terme, devant faire suite à ces travaux.

1 Résumé de notre contribution

1.1 Modèle conceptuel de la multimodalité

Une première contribution réside dans le modèle conceptuel (chapitre 4) décrivant les modalités d'interaction et les différentes formes de combinaisons possibles entre ces modalités. Il permet la spécification claire et précise de l'interaction multimodale en entrée et cerne son espace de conception.

L'approche adoptée s'appuie sur une analyse conceptuelle des systèmes multimodaux et des espaces de conception existants. L'originalité de cette analyse tient à sa capacité à prendre en compte à la fois le système et l'utilisateur. En effet, en étudiant les travaux concernant la multimodalité, aussi bien au niveau des paradigmes récents d'interaction à prendre en compte (chapitre 1), des modèles centrés sur l'utilisateur ou sur le système informatique (chapitre 2), que des usages de la multimodalité (chapitre 3), nous établissons et caractérisons précisément, dans le chapitre 4, les principaux concepts de la multimodalité utiles à l'ingénierie du logiciel et organisés dans notre modèle conceptuel. Ces concepts englobent les deux niveaux d'abstraction de la modalité d'interaction (niveau Dispositif et niveau Langage d'interaction) identifiés dans [Nigay 1996] et les formes de combinaisons (Complémentarité, Redondance, Equivalence, Redondance/Equivalence) basées sur les propriétés CARE de [Coutaz 1994]. De plus, l'identification de nombreuses caractéristiques pour décrire les dispositifs, les langages d'interaction et les combinaisons offrent les éléments attendus par les concepteurs pour choisir les modalités d'interaction et mettre en œuvre les formes de multimodalité adaptées à l'utilisateur, au contexte et aux tâches. Basé sur ces caractéristiques, nous présentons les premiers éléments d'un guide de conception de la multimodalité.

1.2 Outil de conception et de développement de l'interaction multimodale : ICARE

En adoptant comme fondement notre espace conceptuel (chapitre 4), nous proposons dans la Partie 2 de ce mémoire, un outil de conception et de développement nommé ICARE, dont l'objectif est de favoriser les phases de réplication et d'empirisme de la Figure 12.1. En effet, l'outil ICARE permet de simplifier et d'accélérer le processus de conception et de développement de l'interaction multimodale.

L'outil ICARE est une opérationnalisation de notre modèle conceptuel et répond à l'absence d'outil adéquat à l'interaction multimodale en entrée, un constat mis en évidence par l'étude des outils existants au chapitre 5. Différents outils y sont étudiés, au regard d'une grille d'analyse commune, soulignant les apports et lacunes de chacun.

Pour définir l'outil ICARE, l'approche à composants logiciels a été adoptée. Nous justifions ce choix au chapitre 6. Ainsi, l'outil ICARE consiste en un modèle à composants logiciels qui reprend directement les concepts et les caractéristiques de notre modèle conceptuel défini dans le chapitre 4. Au chapitre 6, nous retrouvons des composants Dispositifs, des composants Langages d'interaction et des composants de combinaisons (Complémentarité, Redondance, Redondance/Equivalence), dont les attributs ne sont autres que les caractéristiques identifiées pour chacun de ces concepts (mode de communication, lieu d'interaction, etc.). L'assemblage des différents composants permet de concevoir l'interaction multimodale désirée en mettant en œuvre les modalités d'interaction et leurs combinaisons.

Pour mesurer la faisabilité et tester notre outil ICARE, nous exposons au chapitre 7 un exemple d'implémentation du modèle à composants ICARE, à l'aide de la technologie JavaBeans. Le choix de cette technologie présente de nombreux avantages, comme l'utilisation du langage de programmation Java (multiplate-forme, haut niveau d'abstraction, large diffusion parmi les programmeurs) et la disponibilité de tous les mécanismes, permettant de manipuler graphiquement les composants. Nous proposons ainsi un éditeur graphique qui admet la manipulation des composants ICARE. Cet éditeur simplifie et accélère les démarches de conception et de développement, dans le cadre d'une conception nécessairement itérative, puisque centrée utilisateur. En effet, l'éditeur autorise la manipulation graphique par glisser/déposer des composants logiciels, permettant de créer ou modifier très simplement les modalités d'interaction et leurs combinaisons, en fonction des tâches à effectuer. De plus, il fournit un système de génération automatique du code de l'assemblage des composants, accélérant considérablement la phase de codage. Nous soulignons que l'outil ICARE a été utilisé par des ingénieurs/développeurs autres que les membres de notre équipe de recherche, soulignant la maturité de l'outil développé.

1.3 Réalisations logicielles

Nous avons enfin examiné la validité des concepts développés au sein de l'outil ICARE, en utilisant l'outil développé, avec la technologie JavaBeans, pour concevoir et développer cinq systèmes multimodaux. Ces systèmes font l'objet de la Partie 3 de ce mémoire, incluant différentes modalités et correspondant à des formes de multimodalité et paradigmes d'interaction distincts. Le premier système, détaillé au chapitre 9, constitue un système simple "jouet" (MID-Multimodal IDentification) et peut être considéré comme un test de l'outil. Deux systèmes interactifs beaucoup

plus conséquents sont ensuite présentés dans le chapitre 10 : YellowPages et un simulateur ATC (Air Trafic Control). Ces deux systèmes mettent en œuvre le paradigme du "mets ça là" de [Bolt 1981]. Enfin, deux autres systèmes multimodaux de réalité augmentée sont détaillés dans le chapitre 11 et proposent des modalités passives combinées. Il s'agit des systèmes MEMO et SAC (Simulateur d'avion de combat). La conception et le développement du système SAC a permis l'utilisation de notre outil par des ingénieurs de THALES.

D'autres développements logiciels ont été basés sur l'outil ICARE, comme une implémentation en C# pour définir un outil de conception et de développement pour téléphones mobiles (annexe 3), ainsi que des extensions de la version de l'outil ICARE, selon la technologie JavaBeans, pour le test formel (annexe 4). Nous verrons que ces travaux entrepris pour étendre notre contribution font partie de nos perspectives. Avant de les présenter, un bilan analytique de nos travaux nous conduit à identifier des limites à notre contribution.

2 LIMITES

Nous identifions deux types de limitation à nos travaux : les limitations concernant les aspects conceptuels et celles liées aux aspects technologiques.

2.1 LIMITATIONS CONCEPTUELLES

2.1.1 Non couverture de toute la partie interface homme-machine de l'architecture logicielle

La limite principale de notre approche est illustrée par son positionnement dans une architecture logicielle de référence, comme l'architecture ARCH montrée à la Figure 12.2. Nous remarquons que l'assemblage des composants ICARE se situe en lieu et place des modules d'interaction physique et logique de l'interface d'entrée. Dans ce mémoire, nous ne proposons pas de solution sur les autres modules à mettre en œuvre : adaptateur du noyau fonctionnel, contrôleur de dialogue et interface de sortie (interaction logique et physique). De plus, il semble nécessaire de s'assurer de la validité de notre approche lorsque nous couplons l'assemblage des composants ICARE avec le résultat des autres outils, afin d'obtenir des IHM cohérentes, performantes et adaptées.

Cette limite, identifiée très tôt dans notre travail de recherche, donne naissance à des perspectives à court terme, présentées dans la section 3 de cette conclusion. De plus, nous avons déjà validé notre approche avec deux autres outils, qui permettent la réalisation des autres parties de l'IHM, lors du projet INTUITION (DGA, Thales-Avionics, LIIHS, LIMSI). Ce projet, dont nous avons présenté les deux systèmes multimodaux résultants dans les chapitres précédents (simulateur ATC et simulateur d'avion de combat), proposent une solution globale couvrant toute l'architecture logicielle de la partie IHM. Cette solution [Bastide 2005] regroupe donc notre outil ICARE pour l'interaction multimodale en entrée, l'outil Most [Rousseau 2005] pour l'interaction multimodal en sortie et l'outil PetShop [Bastide 1998] pour la réalisation du contrôleur de dialogue et de l'adaptateur du noyau fonctionnel.

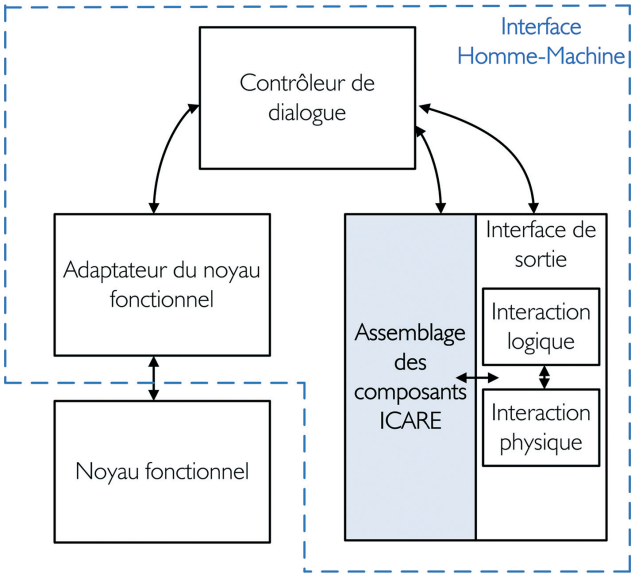


Figure 12.2 : les composants ICARE dans l'architecture logicielle ARCH.

De plus, la définition d'un modèle conceptuel ICARE en sortie [Mansoux 2006] fait l'objet de la thèse de B. Mansoux (en cours), dans laquelle l'approche ICARE pour l'interaction multimodale en sortie est appliquée au cas de la chirurgie augmentée.

2.1.2 Non prise en charge de combinaisons spécifiques

La généricité de nos composants de combinaison couvre de nombreux cas de composition de modalités d'interaction, comme nous l'avons illustrée avec les cinq systèmes multimodaux développés. Néanmoins, des cas de combinaisons spécifiques et non explicites dans le modèle peuvent exister, demandant à notre modèle d'être étendu. Il est très fréquent par exemple d'utiliser un déclencheur, comme un bouton, pour activer ou désactiver un système de reconnaissance vocale. La reconnaissance des mots prononcées par l'utilisateur n'est alors effectuée que lorsqu'elle/il appuie sur le bouton. Le schéma à gauche de la Figure 12.3 montre l'assemblage des composants ICARE permettant de réaliser ce cas. La modalité d'interaction <bouton, alternat> transmet l'autorisation ou non de prendre en compte les mots reconnus par la modalité d'interaction de reconnaissance vocale <système de reconnaissance vocale, langage pseudo-naturel>. Pour ce faire, le composant Complémentarité fusionne les données des deux modalités d'interaction et transmet le résultat au langage d'interaction "commandes vocales", qui filtre les données. La complémentarité transmet des vecteurs de données du type [autorisation, commande reconnue] ou [pas d'autorisation, commande reconnue]. Le composant Commandes vocales ne prend en compte que les messages de type [autorisation, commande reconnue] puis transmet seulement la commande reconnue aux composants suivants. Cet exemple montre un cas particulier d'utilisation de la composition (par complémentarité), qui ne correspond pas au cas classique du "mets ça là" de [Bolt 1981]. En effet,

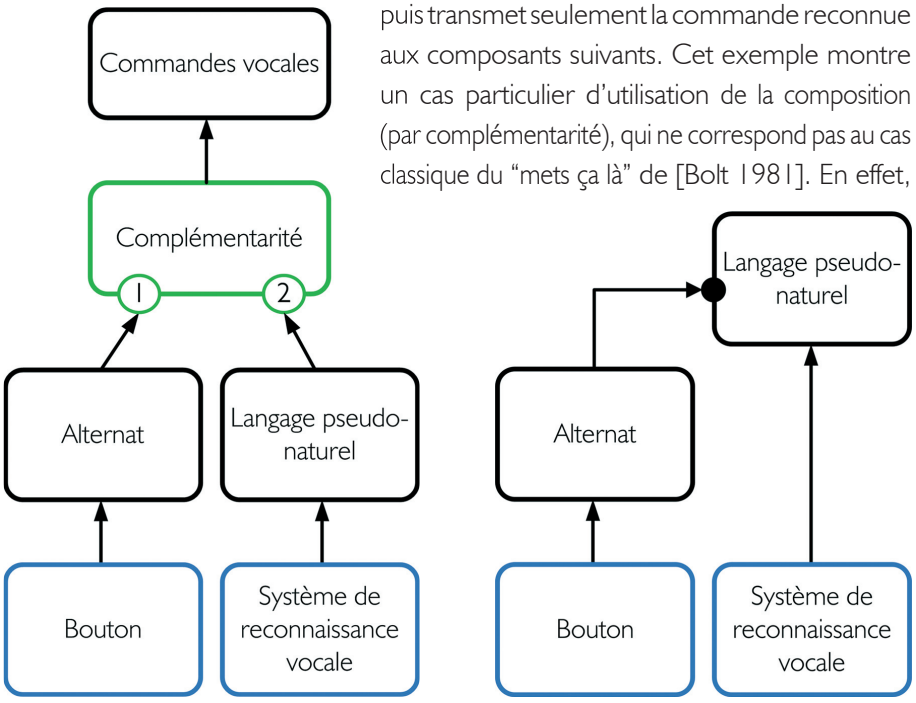


Figure 12.3 : deux solutions d'assemblage de composants ICARE pour un système de reconnaissance vocale avec alternat.

l'exemple met en évidence une relation particulière entre modalités qui certes, est exprimable dans notre modèle actuel, mais peut, par extension du modèle, définir un nouveau concept de connexion entre composants ICARE, simplifiant le schéma existant. Par exemple, une solution possible est, comme montrée sur le schéma à droite de la Figure 12.3, de laisser un port de contrôle au langage d'interaction "langage pseudo-naturel", permettant de mettre en marche ou d'arrêter l'envoi des mots reconnus et de connecter à ce port la modalité <Bouton, Alternat>.

En résumé, nous avons défini des composants de combinaison génériques dans notre modèle conceptuel. Bien que testés pour plusieurs formes de multimodalité, il semble nécessaire d'étendre le modèle conceptuel pour traiter des cas particuliers de relation entre modalités, comme l'exemple précédent. Enfin, notre modèle conceptuel inclut explicitement la notion de facteur de confiance, permettant leur mise en œuvre informatique, mais aucun calcul (si ce n'est très simple dans le cas de la redondance) n'est effectué sur ces facteurs de confiance, en particulier dans les composants de combinaison.

2.2 Limitations technologiques

2.2.1 Programmation manuelle des composants et de l'intégration avec les autres modules logiciels

L'éditeur de manipulation des composants, avec sa génération automatique de code, vise une utilisation par des concepteurs non informaticiens. Néanmoins, une phase préalable de programmation de chaque composant Dispositif et Langage d'interaction, ainsi qu'une phase d'intégration de l'interaction multimodale en entrée créée (code généré automatiquement suite à l'assemblage des composants) avec le reste du système informatique, doivent être réalisées manuellement. Ces phases sont incontournables mais peuvent et doivent être simplifiées en limitant les interventions des programmeurs.

2.2.2 Non vérification automatique des règles de conception

Plusieurs caractéristiques des composants ICARE permettent de vérifier des règles de conception dont nous avons exposées les prémisses dans le chapitre 4. Pour ces règles et pour les nouvelles règles qui seront éventuellement découvertes dans les travaux futurs, nous ne proposons pas, actuellement, leur spécification et leur vérification lors de l'assemblage des composants dans l'éditeur graphique d'ICARE.

En résumé, les limites que nous avons identifiées amènent plusieurs perspectives à notre travail de recherche. Nous les présentons dans la section suivante.

Plusieurs perspectives à notre travail de recherche sont identifiées. Répondant aux limites de notre approche, nous les organisons en trois catégories : celles qui concernent les extensions du modèle conceptuel, celles sur l'évaluation ergonomique de l'interaction multimodale développée et celles concernant l'éditeur graphique.

3.1 Extensions du modèle conceptuel

3.1.1 Prise en compte de l'interface de sortie et du contrôleur de dialogue

Notre approche sépare clairement l'interface d'entrée et de sortie. Cette distinction permet une plus grande modularité du logiciel final. Néanmoins, la coopération de ces deux interfaces est forte car il existe une boucle, appelée boucle de rétro-action [Bérard 1999], incarnant la dépendance continue, sur un intervalle de temps donné, entre les actions humaines et les réactions de l'ordinateur, et réciproquement. Il semble ainsi approprié d'avoir une approche homogène pour les spécifications, la conception et le développement des deux types d'interfaces. Pour cela, nous travaillons actuellement sur la mise en œuvre de notre approche à composants pour les interfaces de sortie, permettant à terme de maîtriser la boucle de rétro-action. Des résultats sont d'ores et déjà publiés [Mansoux 2006]. Nous anticipons une approche de travail basée sur le modèle Pipeline (cf. chapitre 2) permettant de définir, pour une modalité d'entrée des retours d'information selon des modalités de sortie à plusieurs niveaux d'abstraction.

Pour le même objectif d'homogénéité et pour couvrir toutes les parties logicielles de l'IHM, il est également pertinent de proposer, à plus long terme, une solution pour répondre à la conception et au développement du contrôleur de dialogue. Les pistes envisagées concernent l'intégration de notre outil avec des outils permettant de spécifier les tâches du système informatique à haut niveau d'abstraction, comme par exemple l'outil CTTE [Mori 2002].

Ces deux extensions permettent d'avoir une vue unifiée, au sein de l'outil ICARE, de toute l'interaction, permettant par exemple de vérifier la compatibilité de plusieurs schémas ICARE en entrée, au sein du même système interactif (relation entre tâches définie dans le Contrôleur de dialogue), ou de schémas ICARE en entrée et en sortie.

3.1.2 Nouvelles formes de combinaison

Comme nous le présentons dans la section précédente sur les limites de notre approche, de nouvelles formes de combinaison multimodale seront à intégrer

dans notre modèle conceptuel. Comme pour l'exemple de la reconnaissance vocale combinée avec un alternat qui déclenche ou non la reconnaissance, de nouvelles combinaisons particulières à certaines modalités d'interaction peuvent exister et amener à modifier le modèle conceptuel existant.

Notre modèle conceptuel explicite la notion de facteurs de confiance, mais ces derniers sont peu utilisés dans notre modèle (si ce n'est que de façon simple, pour la redondance). D'autres utilisations permettraient d'obtenir des combinaisons plus exactes, comme par exemple l'utilisation d'un filtre pour bloquer les données avec des facteurs de confiance faibles.

3.2 Evaluations ergonomiques

3.2.1 Tests ergonomiques

Pour prendre en compte les phases de tests ergonomiques, des études de couplage d'ICARE avec différents outils ont été initiées.

D'une part, l'outil ICARE (version pour téléphones mobiles en C#) coopère avec l'outil ACIDU, une sonde permettant d'enregistrer et d'analyser a posteriori les actions des utilisateurs sur téléphones mobiles [Serrano 2006]. Cette étude, exposée en annexe 3, participe aux tests d'usage en traçant les données qui circulent entre les composants d'ICARE. Chaque composant ICARE communique avec la sonde ACIDU et les données capturées sont ainsi au format ICARE (corpus multimodal ICARE).

D'autre part, une étude préliminaire a consisté à associer l'outil Lutess, une plate-forme de test formel pour les logiciels réactifs synchrone [Madani 2005] [Jourde 2006], à ICARE. Cette association permet de valider l'assemblage des composants ICARE en fonction des spécifications du système interactif cible. Comme expliqué en annexe 4, les tests générés par Lutess peuvent être à plusieurs niveaux d'abstraction : dispositif, langage, composition ou tâche. Ainsi, le niveau des tests est basé sur le modèle conceptuel ICARE.

3.2.2 Evaluation prédictive

Complémentaires aux tests expérimentaux, il semble pertinent que notre outil propose une évaluation prédictive sur la cohérence interne et globale des schémas d'assemblage de composants. Cette évaluation doit être réalisée avant la phase de codage, lors de l'utilisation de l'éditeur d'assemblage des composants. En effet, il est intéressant d'avertir les utilisateurs de l'éditeur lorsque les assemblages

de composants qu'ils conçoivent ne respectent pas les règles de conception et d'ergonomie identifiées préalablement, tout en leur laissant le choix de l'action appropriée.

3.2.3 Identification et spécification des règles de conception, capitalisation de l'expérience

Lors de l'établissement de notre modèle conceptuel au chapitre 4, nous exposons quelques recommandations pour la conception de systèmes multimodaux adaptés. Ces recommandations constituent les prémisses d'un véritable guide de conception. Avec la multiplication des systèmes multimodaux disponibles, l'une de nos perspectives à court terme est d'identifier et de spécifier de nombreuses règles de conception. Un premier travail est déjà réalisé et regroupe cinq patrons de conception dans [Godet-Bar 2006] à intégrer dans l'outil.

Ensuite, le développement de nouvelles interactions multimodales avec notre outil permettrait, tout en élargissant notre base de composants, de renforcer notre capital d'expériences en nous offrant le recul nécessaire pour identifier le plus grand nombre de patrons de conception d'interaction multimodale.

Nous anticipons que si une large communauté utilise notre outil, il sera à la fois un répertoire de modalités et de composition sous la forme de composants ICARE, mais aussi un outil qui permettrait aux utilisateurs de spécifier les retours de leurs expériences (un répertoire de règles de conception).

3.3 Extensions de l'éditeur

3.3.1 Nouvelle version de l'éditeur graphique

Parallèlement aux évolutions du modèle conceptuel présentées précédemment, il semble important que les prochains résultats soient pris en charge dans l'éditeur. En effet, les travaux sur l'interaction multimodale en sortie et sur le contrôleur de dialogue doivent être intégrés de manière cohérente dans l'éditeur, à l'instar de nos travaux sur l'interaction en entrée. Cette intégration doit être adaptée à une utilisation par des informaticiens et des non informaticiens. A ce titre, il peut être pertinent de modifier la vision des modalités d'interaction comme une seule entité (au lieu du couple <Dispositif, Langage d'interaction>), simplifiant l'utilisation de l'éditeur par des non informaticiens et offrant des techniques d'encapsulation au sein de l'éditeur. L'éditeur comprendrait alors deux vues des composants : l'une orientée conception et l'autre programmation.

Une autre version de l'éditeur doit être développée afin de manipuler des composants de tout type et pas seulement des composants JavaBeans. Nous avons touché du doigt ce problème, lorsque nous avons étudié la version ICARE sur téléphones mobiles avec des composants logiciels en C# (cf. annexe 3). Ce travail peut reposer sur la plate-forme à composants hétérogènes, définis dans le réseau d'excellence européen SIMILAR (FP6-2002-IST1-507609). Cette perspective est en cours d'étude dans le cadre du projet européen STREP OpenInterface (FP6-2005-IST5-035182) qui a démarré le 1er septembre 2006.

De plus, l'éditeur doit proposer un module vérifiant automatiquement les règles de conception, pendant que les concepteurs définissent et caractérisent l'interaction multimodale. Nous imaginons ce contrôle comme un système d'alerte, les prévenant d'un dysfonctionnement potentiel entre les modalités d'interaction combinées. Une solution doit être proposée, mais le choix de l'appliquer ou non est laissé finalement à l'expertise des concepteurs.

3.3.2 Evaluation ergonomique de l'éditeur graphique

Intégrant les perspectives identifiées, une évaluation ergonomique de l'éditeur étendu doit être menée. Il s'agit ici d'effectuer une étude d'utilisabilité, impliquant de nombreux concepteurs avec des connaissances variées en programmation. A ce jour, seuls les développeurs de THALES-Avionics, dans le cadre du développement du Simulateur d'avion de combat, ont utilisé notre outil. Une évaluation plus importante permettrait la validation de l'éditeur pour une cible d'utilisateurs plus large, intégrant notamment informaticiens et non informaticiens.

De plus, le SAC est le seul système interactif validant complètement notre approche, une démarche itérative de conception centrée sur l'utilisateur, impliquant six pilotes militaires. Les autres systèmes développés ont seulement démontré que notre approche simplifie et accélère la conception, le codage et la maintenance de l'interaction multimodale. La prochaine étape doit donc passer par la validation de notre approche dans un plus grand nombre de projets, reposant sur une démarche itérative de conception centrée sur les utilisateurs de l'interaction multimodale.

BIBLIOGRAPHIE

A

[Allen 1983] Allen, J., *Maintaining Knowledge about Temporal Intervals*. Communication of ACM 26 (11), Novembre 1983, p. 832-843.

[Adams 2003] Adams, L., Hunt, L., Moore, M., *The Aware System: Prototyping an Augmentative Communication Interface*. Proceedings of RESNA'2003, Atlanta, GA, Juin 2003.

[Amalberti 1996] Amalberti, R., *La conduite de systèmes à risques*. Paris, PUF, 1996.

B

[Barbier 2002] Barbier, F., Cauvet, C., Oussalah, M., Rieu, D., Bennasri, S., Souveyet, C., *Composants dans l'ingénierie des systèmes d'information : concepts clés et techniques d'utilisation*. Actes des 2ème assises nationales du GdR i3, Nancy, Décembre 2002.

[Barnard 1987] Barnard, P.J., Cognitive resources and the learning of human-computer dialogs. In J.M. Carroll (Ed.), *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*. Cambridge, Massachusetts, MIT Press, 1987, p. 112-158.

[Bass 1992] Bass, L. et al. *A Metamodel for Runtime Architecture of an Interactive System*. The UIMS Workshop Tool Developers, SIGCHI Bulletin, 24 (1), 1992, p. 32-37.

[Bass 2000] Bass, L., Buhman, C., Comella-Dorda, S., Long, F., Robert, J., Seacord, R., Wallnau, K., *Volume I : Market Assessment of Component-Based Software Engineering*. Carnegie Mellon University, Software Engineering Institute, Technical note CMU/SEI-2001-TN-007, Mai 2000.

[Bastide 1998] Bastide, R., Palanque, P., Le, D., Munoz, J., *Integrating rendering specifications into a formalism for the design of interactive systems*. Dsv-is'98, Abington, UK, June, 1998, Springer Verlag.

[Bastide 2005] Bastide, R., Nigay, L., Bazalgette, D., Bellik, Y., Nouvel, C., *The INTUITION Design Process: Structuring Military Multimodal Interactive Cockpits Design According to the MVC Design Pattern*. HCI International, 3rd International Conference on Universal Access in Human-Computer Interaction, Las Vegas, Nevada, USA, Juillet 2005.

[Baudel 1995] Baudel, T., *Morphologie de l'Interaction Humain-Ordinateur : Étude de Modèles d'Interaction Gestuelle*. Thèse à l'université de Paris-Sud, U.F.R. scientifique d'Orsay, Décembre 1995.

[Beans 1997] *JavaBeans 1.01 specification*, Sun Microsystems, Juillet 1997.

[Beaudouin-Lafon 2000] Beaudouin-Lafon, M., *Instrumental interaction : an interaction model for designing post-WIMP user interfaces*. Proceedings of the 2000 Conference on Human Factors in Computing System (CHI-00), , N. Y., Avril 2000, ACM Press, p. 446-453.

[Bellik 1992] Bellik, Y., Teil, D., *Multimodal Dialog Interface*. WWDU'92, Berlin, Septembre 1992.

[Bellik 1995] Bellik, Y., Ferrari, S., Néel, F., Teil, D., Pierre, E., Tachoires, V., *Interaction Multimodale : Concepts et Architecture*. L'Interface des Mondes Réels et Virtuels, Montpellier, Juin 1995.

[Bérard 1999] Bérard, F., *Vision par ordinateur pour l'interaction homme-machine fortement couplée*. Thèse de Doctorat de l'Université Joseph Fourier, Grenoble, France, 1999.

[Bernsen 1994] Bernsen, N., *Modality Theory in support of multimodal interface design*. Proceedings of Intelligent Multi-Media Multi-Modal Systems, 1994, p. 37-44.

[Bolt 1980] Bolt, R., *Put that there: Voice and gesture at the graphics interface*. Computer Graphics, 1980, p. 262-270.

[Boehm 1988] Boehm, B., *The Spiral Model of Software Development and Enhancement*. IEEE Computer, May 1988, p. 61-72.

[Bouchet 2004a] Bouchet, J., Nigay, L., *ICARE : A Component-Based Approach for the Design and Development of Multimodal Interfaces*. Late Breaking Results of Conference on Human Factors in Computing Systems ACM-CHI 2004, Vienna, Austria, ACM Publisher, April 2004.

[Bouchet 2004b] Bouchet, J., Nigay, L., Bazalgette, D., *ICARE : Approche à composants pour l'interaction multimodale*. Actes des Premières Journées Francophones : Mobilité et Ubiquité 2004, Nice, France, ACM Publisher, Juin 2004, p. 36-43.

[Bouchet 2004c] Bouchet, J., Nigay, L., Ganille, T., *ICARE Software Components for Rapidly Developing Multimodal Interfaces*. Proceedings of ICMI 2004, State College, PA, USA, ACM Publisher, October 2004, p. 251-258.

[Bouchet 2005] Bouchet, J., Nigay, L., Ganille, T., *The ICARE Component-Based Approach for Multimodal Input Interaction: Application to Real-Time Military Aircraft Cockpits*. HCI International, 3rd International Conference on Universal Access in Human-Computer Interaction, Las Vegas, Nevada, USA, Juillet 2005.

[Bourguet 2003] Bourguet, M.L. *Designing and Prototyping Multimodal Commands*. Proceedings of INTERACT'03, Zurich, September 2003, p. 717-720.

[Buxton 1983] Buxton, W., *Lexical and pragmatic considerations of input structures*. Computer Graphics, 17 (1), 1983, p. 31-37.

C

[Carbonell 2003] Carbonell, N., *Recommendations for the design of usable multimodal command languages*. Universal Access in the Information Society Journal, Special issue' Multimodality, a step towards universal access', 2(2), 2003, p. 143-159.

[Card 1983] Card, S., Moran, T. P. and Newell, A., *The Psychology of Human-Computer Interaction*. Erlbaum Associates, Hillsdale, New Jersey, 1983.

[Card 1990] Card, S., Mackinlay, J., Robertson, G., *The Design Space of Input Devices*. Proceedings of Computer Human Interaction'90, Seattle, 1990, p. 117-124.

[Cheung 2006] Cheung, D.; Tigli, J.-Y.; Lavirotte, S.; Riveill, M. *Wcomp: a Multi-Design Approach for Prototyping Applications using Heterogeneous Resources*. Rapid System Prototyping, Seventeenth IEEE International Workshop on 14-16, June 2006, p.119–125.

[Cohen 1997] Cohen, P. R., Johnston, M., McGee, D., Oviatt, S., Pittman, J., Smith, I., Chen, L., Clow and J., *QuickSet: multimodal interaction for distributed applications*, International Multimedia Conference. Proceedings of the fifth ACM international conference on Multimedia, Seattle, 1997, p. 31–40.

[Cohen 2000] Cohen, P. R., McGee, D. R. and Clow, J., *The efficiency of multimodal interaction for a map-based task*. Proceedings of the Applied Natural Language Processing Conference (ANLP'00), Seattle, WA, Morgan Kaufmann, April 2000, p. 331-338.

[Corba 2002] *Formal/02-06-65 : CORBA Components*, v3.0 full specification, OMG, Juin 2002.

[Coutaz 1994] Coutaz, J., Nigay, L., *Les propriétés "CARE" dans les interfaces multimodales*. Actes de la conférence IHM'94, Lille, 1994.

[Coutaz 2000] Coutaz, J., *Modélisation en Interaction Homme-Machine*. Support de cours du DEA ISC de l'Université Joseph Fourier, Grenoble, 2000.

D

[Demumieux 2005] Demumieux, R. and Losquin, P., *Gathering customer's real usage on mobile phones*. Proceedings of MobileHCI'05, Salzburg, Austria, Septembre 2005.

[Dey 2001] Dey, A. K., Salber, D., and Abowd, G. D., *A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications*. Anchor article of a special issue on context-aware computing in the Human-Computer Interaction (HCI) Journal, Volume 16 (2-4), 2001, p. 97-166.

[Dragicevic 2004] Dragicevic, P., *Un modèle d'interaction en entrée pour des systèmes interactifs multi-dispositifs hautement configurables*. Thèse de l'Université de l'École Nationale Supérieure des Techniques Industrielles et des Mines de Nantes, Nantes, 2004, 271 p.

[Dragon 2006] Site du système de reconnaissance vocale Dragon NaturallySpeaking 8 du constructeur Nuance, http://www.nuance.com/naturallyspeaking/

[Duarte 2006] Duarte, C., Carrico, L. *A conceptual framework for developing adaptive multimodal applications*. Proceedings of IUI 2006, ACM Press, p. 132-139.

[Dubois 2001] Dubois, E., *Chirurgie Augmentée, un Cas de Réalité Augmentée* ; Conception et Réalisation Centrées sur l'Utilisateur. Thèse de l'Université Joseph Fourier, Grenoble, 2001.

[Dupuy-Chessa 2005] Dupuy-Chessa, S., du Bousquet, L., Bouchet, J., Ledru, Y., *Test of the ICARE platform fusion mechanism*. LNCS, Springer, 12th International Workshop on Design, Specification and Verification of Interactive Systems, Juillet 2005.

E

[Elting 2003] Elting, C. et al. *Architecture and implementation of multimodal plug and play*. Proceedings of ICMI'03, 2003, p. 93-100.

[EJB 2002] Sun Microsystems, *Enterprise JavaBeans Specification*, Version 2.1, Août, 2002.

F

[Favre 2003] Favre, J.-M., *Organisational issues in CBSE*. Cours sur les applications à base de composants du DEA ISC de l'Université Joseph Fourier, Grenoble, 2003.

[Feiner 1997] Feiner, S., MacIntyre, B., Höllerer, T., Webster, A., *A Touring Machine : 3D Mobile Augmented Reality Systems for Exploring the Urban Environment*. ISWC'97, Cambridge, 1997, p. 74-75.

[Fishkin 2000] Fishkin, K.P., Gujar, A., Harrison, B. L., Moran, T. P., Want, R.. *Embodied User Interfaces for Really Direct Manipulation*. Communications of the ACM, 43, 9, 2000, p. 74-80.

[Fishkin 2004] Fishkin, K.P. *A Taxonomy for and Analysis of Tangible Interfaces*. Journal of Personal and Ubiquitous Computing, Springer-Verlag, 8, 5, 2004, p. 347-358.

[Fitzmaurice 1995] Fitzmaurice, G., Ishii, H., Buxton, W., *Bricks: laying the foundations for graspable user interfaces*. Proceedings of the CHI'95 conference on human factors in computing systems, Denver, Colorado, May 1995, p. 442–449.

[Foley 1984] Foley, J., Wallace, V., Chan, P., *The Human Factors of computer Graphics interaction techniques*. IEEE computer Graphics and Applications, 4(11), 1984 p. 13-48.

[Frohlich 1991] Frohlich, D., *The Design Space of Interfaces*. Multimedia Systems, Interaction and Applications, 1st Eurographics Workshop, Stockholm, Suède, Springer Verlag, 1991, p. 53-69.

G

[Gaildrat 1993] Gaildrat, V., Caubet, R., Rubio, R., *Conception d'un modeleur déclaratif de scènes tridimensionnelles pour la synthèse d'images*. MICAD'93, Paris, 1993.

[Gaines 1991] Gaines, B.R., *Modeling and Forecasting the Information Sciences*. Information Sciences 57-58,1991, p. 3-22.

[Gamma 1996] Gamma, E., Hlem, R., Johnson, R., Vilissides, J., *Design Patterns : Catalogue de modèles de conception réutilisables*. International Thomson Publishing France, Paris, 1996.

[Godet-Bar 2006] Godet-Bar, G., Dupuy-Chessa, S., Nigay, L., *Towards a System of Patterns for the Design of Multimodal Interfaces*. Chapter 3; Computer-Aided Design of User Interfaces, Proceedings of 6th International Conference on Computer-Aided Design of User Interfaces CADUI'2006, Bucharest, Information Systems Series, Springer-Verlag, Berlin, June 2006, p. 27-40.

[Goldberg 1984] Goldberg, A., *Smalltalk-80: The Interactive Programming Environment*. Addison-Wesley, 1984.

[Gray 2002] Gray, P. and Salber, D. *Modelling and Using Sensed Context Information in the Design of Interactive Applications*. Proceedings of EHCI 01, 2002, p. 92-111.

H

[Han 2005] Han, J. Y., *Low-Cost Multi-Touch Sensing through Frustrated Total Internal Relection*. Proceedings of UIST'05, Washington, USA, 2005, p. 115-118.

[Harrison 1998] Harrison B, Fishkin K, Want R, Gujar A, Mochon C, *Squeeze me, hold me, tilt me! An exploration of manipulative user interfaces*. Proceedings of the CHI'98 conference on human factors in computing systems, Los Angeles, California, April 1998, p. 17–24

[Heckmann 2002] Heckmann, M., Berthommier, F. & Kroschel, K., *Noise adaptive stream weighting in audio-visual speech recognition*. EURASIP JASP, 2002, 11, p. 1260-1273.

[Hong 2000] Hong, J. and Landay, J., *Satin : a toolkit for informal ink-based applications*. Proceedings of UIST'00, San Diego, California, 2000, p. 63-72.

I

[IHM 1992] *Compte rendu des ateliers*, IHM'92 Quatrièmes Journées sur l'Ingénierie des Interfaces Homme-Machine, 1992.

[Intuikit 2006] Site internet officiel, http://www.intuilab.com/presentation/201-intuikit.html

[Ishii 1997] Ishii H. and Ullmer B., *Tangible Bits: Towards Seamless Interfaces between People*, Bits and Atoms. Proceedings of CHI'97, Atlanta, Georgia, 1997, p. 234-241.

J

[Jöst 2005] Jöst; M., Häußler, J., Merdes, M., Malaka, R., *Multimodal Interaction for Pedestrians: An Evaluation Study*. Proceedings of IUI'05, San Diego, California, USA, Janvier 2005, p. 59-66.

[Jourde 2006] Jourde, F., Nigay, L., Parissis, I., *Formal test of interactive systems: ICARE-Lutess / Test formel de systèmes interactifs multimodaux : couplage ICARE – Lutess*. Proceedings of ICSSEA 2006, The 19th International Conference on Software & Systems Engineering and their Applications, Paris, France, December 2006.

K

[Kangas 1999] Kangas, K. and Röning, J. *Using Code Mobility to Create Ubiquitous and Active Augmented Reality in Mobile Computing*. Proceedings of Int'l Conf. Mobile Computing and Networking (Mobicom' 99), Seattle, 1999, p. 48-58.

[Kaster 2003] Kaster, T., Pfeiffer, M., Bauckhage, C., *Combining Speech and Haptics for Intuitive and Efficient Navigation through Image Databases*. Proceedings of the 5th international conference on Multimodal interfaces (ICMI'03), Vancouver, British Columbia, Canada, ACM Press, 2003, p. 180-187.

[Kirste 2001] Kirste, T., Herfet, T., Schnaider, M., *EMBASSI: multimodal assistance for universal access to infotainment and service infrastructures*. Proceedings of the 2001 EC/NSF workshop on Universal accessibility of ubiquitous computing: providing for the elderly, Alcácer do Sal, Portugal, 2001, p. 41-50.

[Krepki 2004] Krepki, R., Blankertz, B., Curio, G. and Müller, K.-R.., *The Berlin Brain-Computer Interface (BBCI): towards a new communication channel for online control in gaming applications*. Journal of Multimedia Tools and Applications, 2004. invited contribution.

[Kumar 2004] Kumar S., Cohen P., Coulston R., *Multimodal Interaction under Exerted Conditions in a Natural Field Setting*. Proceedings of the Sixth International Conference on Multimodal Interfaces (ICMI 2004), Pennsylvania, USA, 2004.

L

[Lachenal 2004] Lachenal, C., *Modèle et infrastructure logicielle pour l'interaction multi-instrument multisurface*, Thèse de l'Université Joseph Fourier, Grenoble, 2004, 200 p.

[Larousse 2000] Dictionnaire Larousse, 2000.

[Leplat 1997] Leplat, J., *Regards sur l'activité en situation de travail, contribution à la pshychologie ergonomique*. Le Travail Humain. Paris : PUF, 1997, 263 p.

[Leplat 2000] Leplat, J., *L'analyse psychologique de l'activité en ergonomie*. 1ere édition, Octarès éditions, Toulouse, 2000, 164 p.

M

[Mackay 1998] Mackay, W., *Augmented Reality: Linking real and virtual worlds*, A new Paradigm for interacting with computers. Proceedings of AVI'98, ACM conference on Advanced Visual Interfaces, ACM, NewYork, 1998, p. 1-9.

[Madani 2005] Madani, L., Oriat, C., Parissis, I., Bouchet, J., Nigay, L., *Synchronous Testing of Multimodal Systems: an Operational Profile-Based Approach*. Proceedings of ISSRE 2005, 16th IEEE International Symposium on Software Reliability Engineering, Chicago, Illinois, USA, IEEE, Novembre 2005.

[Mansoux 2006] Mansoux, B., Nigay, L., Troccaz, J., *Output Multimodal Interaction: The Case of Augmented Surgery*. Proceedings of HCI 2006, BCS Conference Series, Springer-Verlag and ACM Press, Spetembre 2006.

[Martin 2002] Martin, J.-C. *On the Use of the Multimodal Clues in Observed Human Behavior for the Modeling of Agent Cooperative Behavior*. Papers from the AAAI Workshop on "Autonomy, Delegation, and Control: From Inter-agent to Groups", Eighteenth National Conference on Artificial Intelligence (AAAI-2002, Technical report WS-02-03. AAAI Press. ISBN 1-577735-156-8. Edmonton, Alberta, Canada, p 39-43.

[McDermid 1984] McDermid, J., Ripkin, K., *Life Cycle Support in the ADA environment.* Cambridge University Press, 1984.

[Microsoft 2002] Microsoft, *MicrosoftCOM White Papers*, les composants Microsoft http ://www.microsoft.com/com/wpaper/default.asp#COMPapers, 2002.

[Microsoft 2003] Microsoft, *Plate-forme .NET*, http ://www.microsoft.com/net/, 2003.

[Minsky 1968] Minsky, M. L. Matter, *Mind and Models*. Semantic Information Processing, MIT Press, 1968.

[Montmollin 1986] de MontMollin, M., *L'intelligence de la tâche*. Eléments d'ergonomie cognitive, 2eme édition, Bern, Peter Lang, 1986.

[Montmollin 1995] de MontMollin M. et al., *Vocabulaire pour l'ergonomie*. 1ere édition, Octarès éditions, Toulouse, 1995, 255 p.

[Mori 2002] Mori, G., Paternò, F., Santoro, C., *CTTE: Support for Developing and Analyzing Task Models for Interactive System Design*. IEEE Trans. Software Eng. 28(8), 2002, p. 797-813.

[Myers 2000] Myers, B., Hudson, S. E., Pausch, R., *Past, Present, and Future of User Interface Software Tools*. ACM Transactions on Computer-Human Interaction, Vol. 7, No. 1, Mars 2000, p. 3–28.

N

[Nielsen 1993] Nielsen, J. 1993. *Usability Engineering*. Academic Press Prof., Inc., San Diego, CA.

[Nigay 1994] Nigay, L., *Conception et modélisation logicielles des systèmes interactifs*. Thèse de l'Université Joseph Fourier, Grenoble, 1994.

[Nigay 1996] Nigay, L., Coutaz, J., *Espaces conceptuels pour l'interaction multimédia et multimodale*. TSI, spécial Multimédia et Collecticiel, AFCET &Hermes Publ., Vol 15(9). 1996, p. 1195-1225.

[Nigay 2006] Nigay, L., Gray, P., *Interactive systems & new interface technologies: Engineering for Multimodal Human-Computer Interaction*. European Workshop on Interactive systems & new interface technologies, organized by the European Commission in order to prepare the Seventh Framework Programme (FP7 – 2007-2013) of the European Community for research, technological development and demonstration activities, European Commission, Luxembourg, January 2006.

[Norman 1986] Norman, D., *Cognitive Engineering, User Centered System Design*. New Perspectives on Computer Interaction, édité par Norman D., Draper S., Hillsdale, New Jersey : Lawrence Erlbaum Associates, 1986, p. 31-61.

O

[OpenInterface 2006] Site internet officiel du projet, http://www.openinterface.org

[OSGi 03] OSGi, *OSGi Service Platform*, Release 3, Mars, 2003

[Oviatt 1999] Oviatt, S., *Ten Myths of Multimodal Interaction*. Communications of the ACM, Vol. 42, No 11, 1999, p. 74-81.

[Oviatt 2000 Oviatt, S., Cohen, P. R., Wu, L., Vergo, J., Duncan, L., Suhm, B., Bers, J., Holzman, T., Winograd, T., Landay, J., Larson, J. and Ferro, D., *Designing the user interface for multimodal speech and gesture applications: state-of-the-art systems and research directions*. Human Computer Interaction, 15(4), 2000, p. 263-322.

[Oviatt 2004] Oviatt, S., Coulston, R. and Lundsford, R., *When Do We Interact Multimodally?* Cognitive Load and Multimodal Communication Patterns. Proceedings of IEEE International Conference on Multimodal Interfaces (ICMI'04), 2004, p. 129-136.

P

[Persson 01] Persson, P., Espinoza, F., Cacciatore, E., *GeoNote : Social Enhancement of Physical Space*. CHI 01, 2001.

[Piekarski 2002] Piekarski, W., Thomas, B., *ARQuake : The Outdoor Augmented Reality Gaming System*. Communications of the ACM, Janvier 2002.

[Pieraccini 2003] Pieraccini, R., Dayanidhi, K., Bloom, J., Dahan, J. G., Phillips, M., Boodman, B. R. and Prasad, K. V., *A multimodal conversational interface for a concept vehicle*. Proceedings of Eurospeech, Geneva, Switzerland, September 2003.

R

[Rabardel 1995] Rabardel, P., *Les hommes et les technologies, approche cognitive des instruments contemporains*. Colin, Paris, 1995, 239 p.

[Rasmussen 1986] Rasmussen, J., *Information processing and human-machine interaction, an approach to cognitive engineering*. Livre édité par Elsevier Science Publishing, 1986.

[Renevier 2004a] Renevier, P., Nigay, L., Bouchet, J., Pasqualetti, L., *Generic Interaction Techniques for Mobile Collaborative Mixed Systems*. Proceedings of the Fourth International Conference on Computer-Aided Design of User Interfaces CADUI'04, Funchal, Isle of Madeira, Kluwer Academics Publishers, Janvier 2004, p. 314-327.

[Renevier 2004b] Renevier, P., *Systèmes Mixtes Collaboratifs sur Supports Mobiles: Conception et Réalisation*. Thèse de l'Université Joseph Fourier, Grenoble, 2004, 219 p.

[Rey 2005] Rey, G., *Contexte en Interaction Homme-Machine : le contexteur*. Thèse de l'Université Joseph Fourier, Grenoble, 2005, 186 p.

[Robbe 2000] Robbe, S, Carbonell, N, Dauchy, P., *Expression constraints in multimodal human-computer interaction*. In H. Lieberman, (Ed.). Proceedings of 2000 International Conference on Intelligent User Interfaces (IUI'2000), New Orleans, New York (NY): ACM Press, Janvier 2000, p. 225-229.

[Robert 1998] Le Robert – Micro, dictionnaire de langue française, 1998.

[Rogalski 1995] Rogalski, J., *From real situations to training situations: conservation of functionalities*. Expertise and technology, Hillsdale, Lawrence Erlbaum Associates, 1995, p. 125-140.

[Rousseau 2005] Rousseau, C., Bellik, Y., Vernier, F. et Bazalgette, D., *Multimodal Output Simulation Platform for Real-Time Military Systems*. HCI International 2005, Las Vegas, Nevada USA, Juillet 2005.

[Ryokai 2004] Ryokai, K., Marti, S., Ishii, H., *I/O brush: drawing with everyday objects as ink*. Proceedings of the SIGCHI conference on Human factors in computing systems, Vienne, Autriche, 2004, p. 303-310.

S

[Schyn 2005] Schyn, A., *Une approche fondée sur les modèles pour l'ingénierie des systèmes interactifs multimodaux*. Thèse de l'université de Toulouse, juin 2005, 199 p.

[Sellen 1992] Sellen, A., Kurtenbach, G. P., Buxton, W., *The prevention of Mode errors Trough Sensory Feedback*. Human Computer Interaction, Lawrence Erlbaum, Vol. 7, No 2, 1992, p. 141-164.

[Serrano 2006] Serrano, M., Nigay, L., Demumieux, R., Descos, J., Losquin, P. *Multimodal Interaction on Mobile Phones: Development and Evaluation Using ACICARE*. Proceedings of MobileHCI 2006, The 8th International Conference on Human Computer Interaction with Mobile Devices and Services, Epoo, Finland, Septembre 2006.

[Shneiderman 1983] Shneiderman, B.,. *Direct manipulation : A step beyond programming languages*. IEEE Computer, 16(8), Août 1983, p. :57–69.

[Sutherland 1963] Sutherland, I. E , *Sketchpad, a man-machine graphical communication system*. Thèse, MIT Libraries, 1963.

[Szyperski 1998] Szyperski, C., *Component Software: Beyond Object-Oriented Programming*. Addison Wesley, 1998.

T

[Tomori 2003] Tomori, O. and Moore. M., *The Neurally-Controllable Internet Browser*. Proceedings of SIGCHI 03, Fort Lauderdale, FL, Avril 2003.

[Trevisan 2004] Trevisan, D., Gemo, M., Vanderdonckt, J., Macq, B., *Focus-based design of mixed reality systems*. ACM International Conference Proceedings Series, Vol. 86, Proceedings of the 3rd annual conference on Task models and diagrams, Prague, République tchéque, 2004, p. 59-66.

[Tyndiuk 2003] Tyndiuk, F., Thomas, G., Schlick, C., Claverie, B.. *Modèles et facteurs humains en IHM*. Application à la Réalité virtuelle, MFI'03 - Modèles Formels de l'Interaction - Actes des Secondes journées Francophones, Lille. France, Mai 2003.

U

[Ullmer 1997] Ullmer, B., Ishii, H., *The metaDESK: models and prototypes for tangible user interfaces*. Proceedings of the 10th annual ACM symposium on User interface software and technology, Banff, Alberta, Canada, 1997, p. 223-232.

[Ullmer 1998] Ullmer, B., Ishii, H., Glas, D., *MediaBlocks: physical containers, transports, and controls for online media*. Proceedings of the 25th annuel conference on Computer graphics and interactive techniques, ACM press, 1998, p. 379-386.

[UML 2006] Site officiel UML par l'Object Management Group, http://www.uml.org.

[UPnP 2006] Universal Plug and Play : Device and Service descriptions, forum UPnP, http://www.upnp.org.

V

[Van Dam 1997] Van Dam, A., *Post-WIMP User Interfaces*. Communications of the ACM, Vol. 40, No 2, Février 1997, p. 63-67.

[Vernier 2001] Vernier, F., *La multimodalité en sortie et son application à la visualisation de grandes quantités d'information*. Thèse de l'Université Joseph Fourier, Grenoble, 2001.

W

[W3C 2003] *W3C Multimodal Interaction Framework*, http://www.w3.org/TR/mmi-framework/, W3C Note 3, mai 2003.

[Weiser 1993] Weiser, M. *Some computer science issues in ubiquitous computing*. Communications of the ACM, 36, 7, 1993, p. 75-84.

[Wellner 1993] Wellner, P., *Interacting with paper on the DigitalDesk*. Communications of the ACM, Special issue on computer augmented environments: back to the real world, Vol. 36, Issue 7, 1993, p. 87-96.

[Wikipedia 2006] Site Wikipédia, l'encyclopédie libre en ligne http://fr.wikipedia.org.

[Williams 1994] Willliams, S., Kindel, C., *The Component Object Model : A Technical Overview*. MSDN Library Microsoft Corporation, Octobre, 1994.

[Wilson 1991] Wilson, M. D., Sedlock, D., Binot, J.-L., Falzon, P., *An Architecture for Multimodal Dialogue*. Proceedings of the second Venona Workshop on Multi-Modal Dialogue, M.M. Taylor, F. Neel & D.G. Bouwhuis Eds., 1991.

Z

[Zouinar 2003] Zouinar, M., Salembier, P., Pasqualetti, L., Nigay, L., Calvet, G., Kahn, J., Rey, G., *Multimodal Interaction on Mobile Artifacts*. Communicating with smart objects developing technology for usable pervasive computing systems, Kogan Page Science, 2003.

LISTE DES PUBLICATIONS

CONFÉRENCES

2004

Renevier, P., Nigay, L., Bouchet, J., Pasqualetti, L., *Generic Interaction Techniques for Mobile Collaborative Mixed Systems*. Proceedings of the Fourth International Conference on Computer-Aided Design of User Interfaces CADUI'04, Funchal, Isle of Madeira, Kluwer Academics Publishers, Janvier 2004, p. 314-327.

Bouchet, J., Nigay, L., *ICARE : A Component-Based Approach for the Design and Development of Multimodal Interfaces*. Late Breaking Results of Conference on Human Factors in Computing Systems ACM-CHI 2004, Vienna, Austria, ACM Publisher, April 2004.

Bouchet, J., Nigay, L., Bazalgette, D., *ICARE : Approche à composants pour l'interaction multimodale*. Actes des Premières Journées Francophones : Mobilité et Ubiquité 2004, Nice, France, ACM Publisher, Juin 2004, p. 36-43.

Bouchet, J., Nigay, L., Ganille, T., *ICARE Software Components for Rapidly Developing Multimodal Interfaces*. Proceedings of ICMI 2004, State College, PA, USA, ACM Publisher, October 2004, p. 251-258.

2005

Bouchet, J., Nigay, L., Ganille, T., *The ICARE Component-Based Approach for Multimodal Input Interaction: Application to Real-Time Military Aircraft Cockpits*. HCI International, 3rd International Conference on Universal Access in Human-Computer Interaction, Las Vegas, Nevada, USA, Juillet 2005.

Dupuy-Chessa, S., du Bousquet, L., Bouchet, J., Ledru, Y., *Test of the ICARE platform fusion mechanism*. LNCS, Springer, 12th International Workshop on Design, Specification and Verification of Interactive Systems, Juillet 2005.

Madani, L., Oriat, C., Parissis, I., Bouchet, J., Nigay, L., *Synchronous Testing of Multimodal Systems: an Operational Profile-Based Approach*. Proceedings of ISSRE 2005, 16th IEEE International Symposium on Software Reliability Engineering, Chicago, Illinois, USA, IEEE, Novembre 2005.

GROUPES DE TRAVAIL

2004

Renevier, P., Nigay, L., Bouchet, J., Pasqualetti, L., *Generic Interaction Techniques for Mobile Collaborative Mixed Reality Systems*. On-line Proceedings of International Workshop on Exploring the Design and Engineering of Mixed Reality Systems MIXER 2004, Funchal, Isle of Madeira, Janvier 2004.

Bouchet, J., Ingénierie ergonomique de l'Interaction Multimodale : Approche à composants, Annexe des actes d'IHM 2004, Namur, Belgique, Août 2004, p. 5-8.

2005

Bouchet, J., Mansoux, B., Nigay, L., *A Component-Based Approach : ICARE*. Workshop: The Future of User Interface Design Tools of ACM-CHI 2005, Portland, OR, USA, April 2005.

Nigay, L., Bouchet, J., Ganille, T., *Interaction Multimodale dans le Cockpit d'un Avion de Combat*. Actes de la 2ème journée du Groupe Régional Rhône-Alpes de l'AAAF, Grenoble, France, Juillet 2005. p. 23.

SOMMAIRE

SOMMAIRE

ANNEXE 1 270

CODE SOURCE D'UN COMPOSANT DISPOSITIF ET D'UN LANGAGE D'INTERACTION

1 Exemple de composant Dispositif 270
2 Exemple de composant Langage d'interaction 273

ANNEXE 2 277

CODES SOURCES DES MÉCANISMES DE FUSION

1 Mécanisme de fusion du composant Complémentarité 277
2 Mécanisme de fusion du composant Redondance 282
3 Mécanisme de fusion du composant Redondance/Equivalence 287

ANNEXE 3 289

ACICARE : IMPLÉMENTATION DU MODÈLE CONCEPTUEL ICARE EN C# POUR TÉLÉPHONES MOBILES, CORPUS MULTIMODAUX ICARE

1 ACICARE : couplage d'ACIDU et ICARE 289
2 ICARE sur supports mobiles : ICARE-Mobile 290
3 Corpus multimodaux au format ICARE 290

ANNEXE 4 292

TEST FORMEL DE SYSTÈMES INTERACTIFS MULTIMODAUX : COUPLAGE ICARE – LUTESS

1 Principe de validation avec Lutess d'un système multimodal développé avec la plate-forme ICARE 292
2 Couplage ICARE-Lutess 293
3 Extension de l'éditeur ICARE 294
4 Tests de YellowPages 294

ANNEXE 1
CODE SOURCE D'UN COMPOSANT DISPOSITIF ET D'UN LANGAGE D'INTERACTION

Cette annexe montre deux exemples de code correspondant à un composant Dispositif et à un composant Langage d'interaction.

1 EXEMPLE DE COMPOSANT DISPOSITIF

Le code source ci-dessous décrit la surcouche Java d'un capteur d'orientation (il s'agit du modèle Intertrax d'Intersense) fournissant trois angles en degré.

```
/**
 * Intertrax.java
 *
 * Description:     InterTrax is a Device component providing Orientation
Data in Degree (YAW, PITCH, ROLL)
 * @author       bouchet
 * @version      1.0
 */

package intertrax;

import java.util.Vector;
import java.io.Serializable;

public class InterTrax extends ICARE.ICAREDevice implements java.lang.
Runnable,Serializable {

   protected static final long serialVersionUID = 1;

   /*-----------DLL (Driver) LINKS-----------*/
   // Nous utilisons ici la librairie (JNI) qui fait le lien avec le
driver C++ du capteur
       // d'orientation
   static { System.loadLibrary(«j_intertrax»); }

   private static float Datastruct[] = new float[3];

      private static native void OpenTracker();
      private static native void CloseTracker();
      private static native void ResetAngles();
      private static native void GetData(float Attachment[]);

   /*---------PROPERTIES---------*/

   //delay in milliseconds
   private long delay;
```

```
      //state of thread
   private boolean run;

   //Orientation data of device
   private Vector data;

   //Event with vector of data
   private ICARE.ICAREEvent event;

   //support
   private intertrax.InterTraxSupport interTraxSpt = new intertrax.
InterTraxSupport();

   //thread
   private transient Thread myThread;


   /*---------METHODS---------*/

   /*---------constructor---------*/
   public InterTrax() {
      init();
   }

   /*-----------init-----------*/
   public void init()
   {
      myThread = new Thread(this);
      myThread.setPriority(Thread.MIN_PRIORITY);
      event = new ICARE.ICAREEvent(this);
      setDelay(10);
      data = new Vector();
   }

   //Delay
   public long getDelay() {
      return delay;
   }

   public void setDelay(long value) {
      delay = value;
   }

   public void setDelay(int value) {
      delay = value;
   }


   private void setRun(boolean value) {
      run = value;
   }

   //open tracker
   public void startDevice()
   {
      OpenTracker();
      setRun(true);
      setState(true);
      myThread.start();
```

```
   }

   //close tracker
   public void end()
   {
      setRun(false);
      setState(false);
      CloseTracker();
      System.out.println(«[ICARE] End of InterTrax»);
   }

   //reset angles
   public void resetAngles()
   {
      ResetAngles();
   }

   //data
   private Vector getData() {
      return data;
   }

   private void setData(float [] values) {
      data.clear();
      for (int i = 0; i<3; i++)
      {
         data.addElement(new Float(values[i]));
      }
   }

   /**
   * Method to run the thread
   **/
   public void run()
   {
      try
      {
         while (run)
         {
            GetData(Datastruct);
            setData(Datastruct);
            event.setVector(getData());
            event.setTime(System.currentTimeMillis());
            event.setInitialTime(System.currentTimeMillis());
            event.setConfidenceFactor(this.getConfidenceFactor());
            event.setNbOfValues(3);
            interTraxSpt.fireNewInterTraxData(event);
            Thread.sleep(delay);
         }
      }
      catch (Exception exc)
      {
         exc.printStackTrace();
      }
   }

   private void readObject(java.io.ObjectInputStream stream) throws java.
io.IOException
   {
```

```
      try
      {
         stream.defaultReadObject();
      }
      catch (Exception e)
      {
         System.out.println(«[ICARE] InterTrax «+e);
      }

      if (myThread == null)
      {
         myThread = new Thread(this);
      }
   }

   public synchronized void addInterTraxListener(intertrax.
InterTraxListener l) {
      interTraxSpt.addInterTraxListener(l);
   }
   public synchronized void removeInterTraxListener(intertrax.
InterTraxListener l) {
      interTraxSpt.removeInterTraxListener(l);
   }
}

/* @(#)Intertrax.java */
```

2 Exemple de composant Langage d'interaction

Le code source ci-dessous décrit un langage d'interaction qui convertit en radian trois données spécifiées en degré.

```
/**
 * OrientationRadian.java
 *
 * @author       bouchet
 * @version      1.0
 */

package OrientationRadian;

import java.io.Serializable;
import java.util.*;

public class OrientationRadian extends ICARE.ICAREInteractionLanguage
implements java.lang.Runnable, Serializable {

   protected static final long serialVersionUID = 1;

   /*-----------STATIC-----------*/
   //degree to radian
   public final static double CONVERT = 2*Math.PI/360.0 ;

   /*---------PROPERTIES---------*/
   //vector of data
   private Vector data;
```

```
   //confidence factor of data
   private int cfOfData;

   //initial time of Data
   private long initTimeOfData;
   private ICARE.ICAREEvent event;

   /*pour le thread*/
   //thread
   private transient Thread myThread;

   //run
   private boolean run;
   private boolean on;

   //delay
   private int delay;

   //support
   private OrientationRadianSupport orientationRadianSpt = new Orientatio
nRadianSupport();

   /*---------METHODS---------*/

   /*---------constructor---------*/
   public OrientationRadian() {
      init();
   }

   /*-----------init-----------*/
   public void init()
   {
      data = new Vector();

      data.addElement(new Double(0.0));
      data.addElement(new Double(0.0));
      data.addElement(new Double(0.0));

      myThread = new Thread(this);
      myThread.setPriority(Thread.MIN_PRIORITY);
      setRun(false);
      on = true;
      setDelay(1);
   }

   /*-----------convert data-----------*/
   public void setData(ICARE.ICAREEvent theEvent) {
      double tmpValue;

      Vector value = theEvent.getVector();
      cfOfData = theEvent.getConfidenceFactor();
      initTimeOfData = theEvent.getInitialTime();

      for (int i = 0; i<3; i++)
      {
         data.setElementAt(new Double(0.0),i);
      }

      for (int i = 0; i<3; i++)
```

```
      {
         tmpValue = ((Float)value.elementAt(i)).doubleValue()*CONVERT;
         data.setElementAt(new Double(tmpValue),i);
      }
   }

   public boolean getRun() {
      return run;
   }

   public void setRun(boolean value) {
      run = value;
   }

   public void turnOn(Boolean value)
   {
      on = value.booleanValue();
   }

   public int getDelay() {
      return delay;
   }

   public void setDelay(int value) {
      delay = value;
   }

   /**
   * Start the process
   */
   public void startLanguage()
   {
      if ( !this.getBreakdown() && !this.getState() )
      {
         setRun(true);
         setState(true);
         myThread.start();
      }
      else
      {
         System.err.println(«[ICARE] ***********************************»);
         System.err.println(«[ICARE] Error to start OrientationRadian IL «);
         System.err.println(«[ICARE] State : «+this.getState());
         System.err.println(«[ICARE] Breakdown : «+this.getBreakdown());
         System.err.println(«[ICARE] ***********************************»);
      }
   }

   /**
   * End
   */
   public void end()
   {
      setRun(false);
      this.setState(false);
      System.out.println(«[ICARE] End of Orientation Radian Interaction
Language»);
   }

   /**
```

```
   * Method to run the thread
   **/
   public void run()
   {
      try
      {
         while (run)
         {
            while(on)
            {
               event = new ICARE.ICAREEvent(this);
               event.setVector(data);
               event.setConfidenceFactor(cfOfData);
               event.setInitialTime(initTimeOfData);
               event.setTime(System.currentTimeMillis());
               event.setNbOfValues(data.size());
               orientationRadianSpt.fireNewOrientationRadian(event);
               Thread.sleep(delay);
            }
            if (!on)
            {
               Thread.sleep(delay);
            }
         }
      }
      catch (InterruptedException exc)
      {
         exc.printStackTrace();
      }
   }

   private void readObject(java.io.ObjectInputStream stream) throws java.
io.IOException
   {
      try
      {
         stream.defaultReadObject();
      }
      catch (Exception e)
      {
         System.out.println(«[ICARE] OrientationRadian «+e);
      }

      if (myThread == null)
      {
         myThread = new Thread(this);
      }
   }

   public synchronized void addOrientationRadianListener(OrientationRadia
nListener l) {
      orientationRadianSpt.addOrientationRadianListener(l);
   }

   public synchronized void removeOrientationRadianListener(OrientationRa
dianListener l) {
      orientationRadianSpt.removeOrientationRadianListener(l);
   }
}

/* @(#)OrientationRadian.java */
```

ANNEXE 2
CODES SOURCES DES MÉCANISMES DE FUSION

Cette annexe montre le code Java correspondant au mécanisme des trois composants de combinaison. Le code complet de chaque composant étant important, seule la méthode setData (ICARE.ICAREEvent theEvent) qui reçoit les différents événements et les combinent, le cas échéant, est présentée. Les données combinées sont ensuite mises dans une queue d'événements (myQueue) et sont propagées selon leur ordre de traitement.

1 MÉCANISME DE FUSION DU COMPOSANT COMPLÉMENTARITÉ

```
/**
* set data
* the port is specified using another method called «associate»
*/
   public synchronized void setData(ICARE.ICAREEvent theEvent)
   {
      if (this.getState())
      {
         if (!run)
         {
            //lock
            run = true;

            int nbPortTmp;
            ICARE.ICAREEvent theEventTmp = new ICARE.ICAREEvent(this);

            if (myWaitingQueue.size() >= 1)
            {
               nbPortTmp = ((Integer)(((Vector)(myWaitingQueue.
elementAt(myWaitingQueue.size()-1))).elementAt(0))).intValue();
               theEventTmp = (ICARE.ICAREEvent)(((Vector)(myWaitingQueue.
elementAt(myWaitingQueue.size()-1))).elementAt(1));
            }
            else
            {
               nbPortTmp = ((Integer)myHashtable.get(theEvent.
getSource())).intValue();
               theEventTmp = theEvent;
            }

            /***********
            ** STEP 1 **
            ************/
            // ******** remove old event  or all olders if infini
*********
            if (this.getDeltaT() != -1)
            {
               for (int i=0; i<myMeltingPot.length; i++)
               {
                  for (int j=0; j<((Vector)(myMeltingPot[i])).size(); j++)
                  {
                     ICARE.ICAREEvent eventTemp0 = (ICARE.ICAREEvent)(((Ve
```

```
ctor)(myMeltingPot[i])).elementAt(j));

                    if( eventTemp0.getInitialTime()+(2*this.getDeltaT())
< System.currentTimeMillis())
                    {
                       ((Vector)(myMeltingPot[i])).removeElementAt(j);
                    }
                 }

              }
           }
           else
           {
              //infini deltaT == -1 destroy all data in nbPort
              ((Vector)(myMeltingPot[nbPortTmp-1])).removeAllElements();

           }

           /***********
           ** STEP 2 **
           ************/

           // ******** build the melting pot ********
           if (theEventTmp.getVector() != null)
           {
              ((Vector)(myMeltingPot[nbPortTmp-1])).
addElement(theEventTmp);
           }

           /***********
           ** STEP 3 **
           ************/

           // ******** build myStructures **********
           boolean candidate = true;

           //verify if all ports are used
           for (int i=0; i<myMeltingPot.length; i++)
           {
              if (((Vector)(myMeltingPot[i])).size() <= 0 )
              {
                 candidate = false;
              }
           }


           //all ports are used
           if (candidate)
           {
              myStructures = new Vector();

              for (int i=0;i<((Vector)(myMeltingPot[0])).size();i++)
              {
                 Vector vtemp = new Vector();
                 vtemp.addElement( ((Vector)(myMeltingPot[0])).
elementAt(i) );

                 myStructures.addElement(vtemp);
```

```
            }

            int index = 1;
            if (index < myMeltingPot.length )
            {
               combination(index);
            }

         /***********
         ** STEP 4 **
         ************/

         // ******** choice the good event **********
            long minCalculus = (new Long(Long.MAX_VALUE)).longValue();
            Vector theGoodStructure = null;

            long theTimeStamp = 0;

            for (int i=0; i<myStructures.size(); i++)
            {
               //init var
               long timeStamp = (new Long(Long.MAX_VALUE)).longValue();
               long calculus = 0;

               //get the vector
               Vector vtemp2 = new Vector((Vector)(myStructures.
elementAt(i)));

               boolean flagorder = true;

               if (this.getOrder())
               {
                  long timeStampForOrder = (new Long(Long.MIN_VALUE)).
longValue();

                  for (int j=0; j<vtemp2.size(); j++)
                  {
                     if (timeStampForOrder <= ((ICARE.
ICAREEvent)(vtemp2.elementAt(j))).getInitialTime() )
                     {
                        timeStampForOrder = ((ICARE.ICAREEvent)(vtemp2.
elementAt(j))).getInitialTime();
                     }
                     else
                     {
                        flagorder = false;
                     }
                  }
               }

               if (flagorder)
               {
                  //calculate the minimum timestamp
                  for (int j=0; j<vtemp2.size(); j++)
                  {
                     if (timeStamp > ((ICARE.ICAREEvent)(vtemp2.
elementAt(j))).getInitialTime() )
                     {
```

```
                        timeStamp = ((ICARE.ICAREEvent)(vtemp2.
elementAt(j))).getInitialTime();
                     }
                  }

                  boolean samePeriod = true;

                  //calculate the difference
                  for (int j=0; j<vtemp2.size(); j++)
                  {
                     ICARE.ICAREEvent eventTemp = (ICARE.
ICAREEvent)(vtemp2.elementAt(j));
                     calculus = calculus + (eventTemp.getInitialTime()-
timeStamp);

                     if (this.getDeltaT() != -1)
                     {
                        if ( Math.abs(eventTemp.getInitialTime()-
timeStamp) > this.getDeltaT() )
                        {
                           samePeriod = false;
                        }
                     }
                  }


                  //choice the good structure
                  if ( minCalculus > calculus && samePeriod )
                  {
                     theGoodStructure = new Vector(vtemp2);
                     theTimeStamp = timeStamp;
                     minCalculus = calculus;
                  }
               }
               //else flagorder == false to do nothing
            }

         /************
         ** STEP 5 **
         ************/
         // ******** generate the event *******
            if (theGoodStructure != null)
            {
               myEvent = new ICARE.ICAREEvent(this);

               int cfTemp = 0;

               Vector vtemp3 = new Vector();

               for(int i=0; i<theGoodStructure.size();i++)
               {
                  ICARE.ICAREEvent eventTemp2 = (ICARE.ICAREEvent)(theG
oodStructure.elementAt(i));

                  // ** remove occurence of the good structure in
melting pot **
                  for (int j=0; j<((Vector)(myMeltingPot[i])).size();
j++)
```

```
                    {
                       ICARE.ICAREEvent eventTemp3 = (ICARE.ICAREEvent)((
(Vector)(myMeltingPot[i])).elementAt(j));
                       if (eventTemp2 == eventTemp3)
                       {
                          ((Vector)(myMeltingPot[i])).removeElementAt(j);
                       }
                    }

                    //calculate good confidence factor
                    cfTemp = cfTemp + eventTemp2.getConfidenceFactor();

                    Vector vtemp4 = new Vector(eventTemp2.getVector());
                    for (int j=0;j<vtemp4.size();j++)
                    {
                       vtemp3.addElement(vtemp4.elementAt(j));
                    }
                 }

                 cfTemp = cfTemp/(theGoodStructure.size());

                 myEvent.setVector(vtemp3);
                 myEvent.setNbOfValues(vtemp3.size());
                 myEvent.setInitialTime(theTimeStamp);
                 myEvent.setConfidenceFactor(cfTemp);
                 myEvent.setTime(System.currentTimeMillis());

                 synchronized(myQueue)
                 {

                    myQueue.addElement(myEvent);
                 }

                    this.threadWakeUp();
              }
           }
           else
           {
              //nothing
           }

           if (myWaitingQueue.size() >= 1)
           {
              myWaitingQueue.remove(myWaitingQueue.size()-1);
           }

           run = false;
        }
        else
        {
           Vector element = new Vector();
           element.addElement((Integer)myHashtable.get(theEvent.
getSource()));
           element.addElement(theEvent);
           myWaitingQueue.add(0,element);
        }
     }
     else
```

```
      {
         System.out.println(«[ICARE] setData method of Complementarity
not start»);
      }

   }

   /************ combination *****************/
   /**
   *  Build combination
   */
   private void combination(int index)
   {
      Vector structuresTemp = new Vector();

      for (int i=0; i<myStructures.size(); i++)
      {
         for (int j=0; j<((Vector)(myMeltingPot[index])).size(); j++)
         {
            Vector vtemp = new Vector( ((Vector)(myStructures.
elementAt(i))) );
            vtemp.addElement( ((Vector)(myMeltingPot[index])).
elementAt(j) );

            structuresTemp.addElement(vtemp);
         }
      }

      myStructures.clear();
      myStructures = new Vector(structuresTemp);

      if (index < myMeltingPot.length - 1)
      {
         combination(index + 1);
      }
   }

(…)
```

2 Mécanisme de fusion du composant Redondance

```
/**
* set data
*/
   public synchronized void setData(ICARE.ICAREEvent theEvent)
   {
      if (this.getState())
      {
         if (!run)
         {
            //lock
            run = true;

            int nbPortTmp;
            ICARE.ICAREEvent theEventTmp = new ICARE.ICAREEvent(this);

            if (myWaitingQueue.size() >= 1)
```

```
            {
               nbPortTmp = ((Integer)(((Vector)(myWaitingQueue.
elementAt(myWaitingQueue.size()-1))).elementAt(0))).intValue();
               theEventTmp = (ICARE.ICAREEvent)(((Vector)(myWaitingQueue.
elementAt(myWaitingQueue.size()-1))).elementAt(1));
            }
            else
            {
               nbPortTmp = ((Integer)myHashtable.get(theEvent.
getSource())).intValue();
               theEventTmp = theEvent;
            }

            /***********
            ** STEP 1 **
            ************/

            // ******** remove old event *********
            for (int i=0; i<myMeltingPot.length; i++)
            {
               for (int j=0; j<((Vector)(myMeltingPot[i])).size(); j++)
               {
                  ICARE.ICAREEvent eventTemp0 = (ICARE.ICAREEvent)(((Vecto
r)(myMeltingPot[i])).elementAt(j));

                  if( eventTemp0.getInitialTime()+(2*this.getDeltaT()) <
System.currentTimeMillis())
                  {
                     ((Vector)(myMeltingPot[i])).removeElementAt(j);
                  }
               }
            }

            /***********
            ** STEP 2 **
            ************/

            // ******** build the melting pot ********
            ((Vector)(myMeltingPot[nbPortTmp-1])).addElement(theEventTmp);

            /***********
            ** STEP 3 **
            ************/

            // ******** build myStructures **********
            boolean candidate = true;

            //verify if all ports are used
            for (int i=0; i<myMeltingPot.length; i++)
            {
               if ( ((Vector)(myMeltingPot[i])).size() <= 0 )
               {
                  candidate = false;
               }
            }

            //all ports are used
            if (candidate)
```

```
            {
               myStructures = new Vector();

               for (int i=0;i<((Vector)(myMeltingPot[0])).size();i++)
               {
                  Vector vtemp = new Vector();
                  vtemp.addElement( ((Vector)(myMeltingPot[0])).
elementAt(i) );

                  myStructures.addElement(vtemp);
               }

               int index = 1;
               if (index < myMeltingPot.length )
               {
                  combination(index);
               }

            /***********
            ** STEP 4 **
            ************/

            // ******** choice the good event **********
               long minCalculus = (new Long(Long.MAX_VALUE)).longValue();
               Vector theGoodStructure = null;

               for (int i=0; i<myStructures.size(); i++)
               {
                  //init var
                  long timeStamp = (new Long(Long.MAX_VALUE)).longValue();
                  long calculus = 0;

                  //get the vector
                  Vector vtemp2 = new Vector((Vector)(myStructures.
elementAt(i)));

                  //calculate the minimum timestamp
                  for (int j=0; j<vtemp2.size(); j++)
                  {
                     if (timeStamp > ((ICARE.ICAREEvent)(vtemp2.
elementAt(j))).getInitialTime() )
                     {
                        timeStamp = ((ICARE.ICAREEvent)(vtemp2.
elementAt(j))).getInitialTime();
                     }
                  }

                  boolean samePeriod = true;
                  boolean equivalent = true;

                  //calculate the difference
                  for (int j=0; j<vtemp2.size(); j++)
                  {
                     ICARE.ICAREEvent eventTemp = (ICARE.
ICAREEvent)(vtemp2.elementAt(j));
                     ICARE.ICAREEvent eventRedundant = (ICARE.
ICAREEvent)(vtemp2.elementAt(0));
                     calculus = calculus + (eventTemp.getInitialTime()-
```

```
timeStamp);

                    if (!eventRedundant.getVector().equals(eventTemp.
getVector()))
                    {
                       equivalent = false;
                    }

                    if ( Math.abs(eventTemp.getInitialTime()-timeStamp) >
this.getDeltaT() )
                    {
                       samePeriod = false;
                    }
                 }

                 //choice the good structure
                 if ( minCalculus > calculus && samePeriod && equivalent)
                 {
                    theGoodStructure = new Vector(vtemp2);
                    minCalculus = calculus;
                 }
              }

           /***********
           ** STEP 5 **
           ************/
           // ******** generate the event *******
              if (theGoodStructure != null)
              {
                 myEvent = new ICARE.ICAREEvent(this);

                 int cfTemp = -1;
                 long initTimeTemp = 0;
                 Vector vtemp4 = new Vector();

                 for(int i=0; i<theGoodStructure.size();i++)
                 {
                    ICARE.ICAREEvent eventTemp2 = (ICARE.ICAREEvent)(theG
oodStructure.elementAt(i));

                    // ** remove occurence of the good structure in
melting pot **
                    for (int j=0; j<((Vector)(myMeltingPot[i])).size();
j++)
                    {
                       ICARE.ICAREEvent eventTemp3 = (ICARE.ICAREEvent)((
(Vector)(myMeltingPot[i])).elementAt(j));
                       if (eventTemp2 == eventTemp3)
                       {
                          ((Vector)(myMeltingPot[i])).removeElementAt(j);
                       }
                    }

                    if (cfTemp<=eventTemp2.getConfidenceFactor())
                    {
                       cfTemp = eventTemp2.getConfidenceFactor();
                       vtemp4 = new Vector(eventTemp2.getVector());
                       initTimeTemp = eventTemp2.getInitialTime();
```

```
                    }
                }

                myEvent.setVector(vtemp4);
                myEvent.setNbOfValues(vtemp4.size());
                myEvent.setInitialTime(initTimeTemp);
                myEvent.setConfidenceFactor(cfTemp);
                myEvent.setTime(System.currentTimeMillis());

                synchronized(myQueue)
                {

                   myQueue.addElement(myEvent);
                }

                this.threadWakeUp();
             }
          }
          else
          {
             //nothing
          }

          if (myWaitingQueue.size() >= 1)
          {
             myWaitingQueue.remove(myWaitingQueue.size()-1);
          }

          run = false;
       }
       else
       {
          Vector element = new Vector();
          element.addElement((Integer)myHashtable.get(theEvent.
getSource()));
          element.addElement(theEvent);
          myWaitingQueue.add(0,element);
       }
    }
    else
    {
       System.out.println(«[ICARE] setData method of Redundancy not
start»);
    }

 }

 /************* combination ******************/
 /**
 *  Build combination
 */
 private void combination(int index)
 {
    Vector structuresTemp = new Vector();

    for (int i=0; i<myStructures.size(); i++)
    {
       for (int j=0; j<((Vector)(myMeltingPot[index])).size(); j++)
```

```
         {
            Vector vtemp = new Vector( ((Vector)(myStructures.
elementAt(i))) );
            vtemp.addElement( ((Vector)(myMeltingPot[index])).
elementAt(j) );

            structuresTemp.addElement(vtemp);
         }
      }

      myStructures.clear();
      myStructures = new Vector(structuresTemp);

      if (index < myMeltingPot.length - 1)
      {
         combination(index + 1);
      }
   }
```

3 Mécanisme de fusion du composant Redondance/Equivalence

```
/**
* Set data
*/
   public void setData(ICARE.ICAREEvent theEvent)
   {
      if (this.getState())
      {
         int exist = -1;
         exist = isExist(theEvent);

         if (strategy.equals(«eager»))
         {
            if (exist == -1)
            {
               //create a new structure
               Vector vTemp = new Vector();
               vTemp.addElement(new Long(theEvent.getInitialTime()));
               vTemp.addElement(theEvent);

               synchronized(structuresVector)
               {
                  structuresVector.addElement(vTemp);
               }

               synchronized(myQueue)
               {
                  myEvent = new ICARE.ICAREEvent(this);
                  myEvent.setVector(theEvent.getVector());
                  myEvent.setNbOfValues(theEvent.getNbOfValues());
                  myEvent.setInitialTime(theEvent.getInitialTime());
                  myEvent.setConfidenceFactor(theEvent.
getConfidenceFactor());
                  myEvent.setTime(System.currentTimeMillis());
                  myQueue.addElement(myEvent);
```

```
               }
            }
            //else nothing
         }
         else if (strategy.equals(«lazy»))
         {
            if (exist == -1) {
               //create a new structure
               Vector vTemp = new Vector();
               vTemp.addElement(new Long(theEvent.getInitialTime()));
               vTemp.addElement(theEvent);

               synchronized(structuresVector){
                  structuresVector.addElement(vTemp);
               }
            }
            else
            {
               if (theEvent.getConfidenceFactor() > ((ICARE.ICAREEv
ent)(((Vector)(structuresVector.elementAt(exist))).elementAt(1))).
getConfidenceFactor() )
               {
                  synchronized(structuresVector) {
                     ((Vector)(structuresVector.elementAt(exist))).setElem
entAt(theEvent,1);
                  }
               }
            }
         }
      }
      else {
         System.out.println(«[ICARE] setData method of
RedundancyEquivalence not start»);
      }
   }

/**
* the data exists
*/
   public int isExist(ICARE.ICAREEvent theEvent){
      synchronized(structuresVector){
         int e = -1;
         for (int i = 0; i<structuresVector.size();i++) {

            if ( theEvent.getVector().equals( ( (ICARE.ICAREEvent) ( ((Ve
ctor)(structuresVector.elementAt(i))).elementAt(1) ) ).getVector() ) ) {
               if (theEvent.getInitialTime() <= ((Long) ( ((Vector)(struct
uresVector.elementAt(i))).elementAt(0))).longValue() + this.getDeltaT()
                  && theEvent.getInitialTime() >= ((Long) ( ((Vector)(stru
cturesVector.elementAt(i))).elementAt(0))).longValue() - this.getDeltaT()
                  )
               e = i;
            }
         }
         return e;
      }
   }
```

ANNEXE 3
ACICARE : IMPLÉMENTATION DU MODÈLE CONCEPTUEL ICARE EN C# POUR TÉLÉPHONES MOBILES, CORPUS MULTIMODAUX ICARE

Tandis que le chapitre 7 présente notre implémentation du modèle conceptuel ICARE avec la technologie JavaBeans, une autre implémentation faite par M. Serrano [Serrano 2006] a été faite en C# afin d'explorer l'interaction multimodale sur téléphones mobiles. De plus, l'objectif de cette implémentation a été de coupler ICARE-Mobile avec la sonde ACIDU [Demumieux 2005] qui permet de capturer les données d'usage sur téléphones lors de tests expérimentaux en situation.

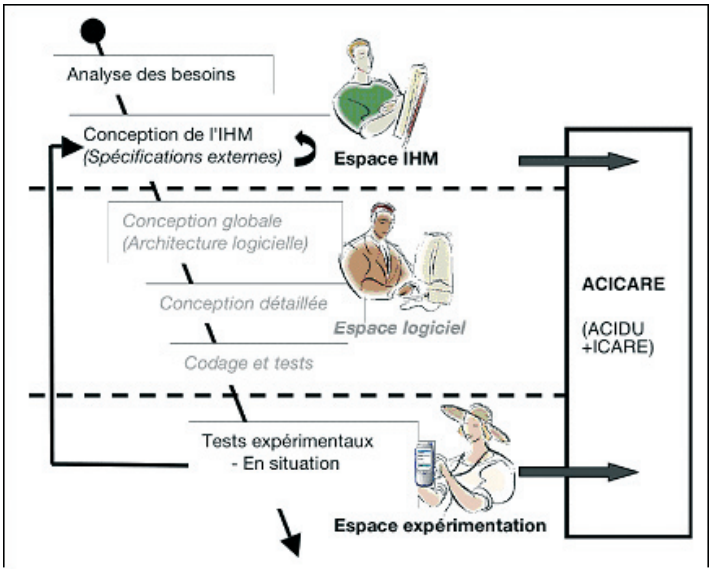
Aussi l'objectif de ce projet est double :

- ICARE-Mobile, une implémentation du modèle conceptuel ICARE en C# ;
- ACICARE, le couplage de la plate-forme ICARE-mobile avec un outil de capture.

1 ACICARE : COUPLAGE D'ACIDU ET ICARE

En couplant ACIDU et ICARE (ACICARE), les étapes d'un processus itératif de conception et de développement sont les suivantes :

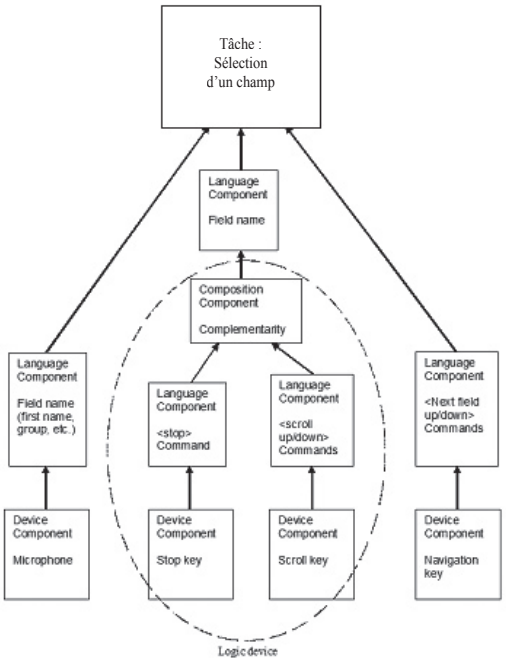
1. Analyse des besoins
2. Conception de l'IHM : schémas ICARE
3. Codage : génération automatique du code de l'interaction multimodale par ICARE
4. Tests expérimentaux : ACIDU collecte automatiquement les données d'usage via son couplage avec les composants exécutables ICARE.
5. Itération étape 2 (re-conception : modifications des schémas ICARE) : la re-conception peut se faire à différents niveaux d'abstraction correspondant à ceux d'ICARE – combinaison de modalités, Langage d'interaction ou Dispositif. Cela permet donc une grande variété dans la re-conception, comme le changement de formes de multimodalité ainsi que l'ajout ou la suppression de modalités. Les étapes de processus itératif avec l'outil ACICARE, issu du couplage d'ACIDU avec ICARE, sont résumées à la figure suivante :



2 ICARE sur supports mobiles : ICARE-Mobile

Le modèle conceptuel ICARE a été implémenté en C# et a été testé sur SPV c500 pour la gestion des contacts. Par exemple, les diagrammes ICARE pour les tâches de sauvegarde ou d'annulation d'un nouveau contact ont été conçus et développés. Ces diagrammes présentent un ensemble de modalités développées selon le modèle conceptuel ICARE et reposant sur des dispositifs distincts.

Plusieurs modalités (composants Dispositif et Langage d'interaction) ont été développées reposant sur la parole, le clavier ou des touches dédiées. Pour les composants de combinaison, les composants Complémentarité et Redondance ont été développés (le composant Redondance/Equivalence n'a pas été développé). Un exemple de complémentarité pour sélectionner un champ du formulaire de création d'un contact est montré ci-contre : le bouton sur la droite du téléphone sert à faire démarrer le défilement circulaire vers le haut ou le bas tandis qu'une touche du clavier permet de l'arrêter. Deux autres modalités non combinées sont équivalentes à cette modalité combinée par complémentarité pour sélectionner un champ.



La version ICARE sur téléphones mobiles, ICARE-Mobile, n'a pas donné lieu au développement d'un éditeur graphique comme dans la version avec la technologie JavaBeans (chapitre 7). Aussi avec ICARE-Mobile, l'assemblage des composants et la modification de cet assemblage (diagramme ICARE) se fait à la main.

Outre l'implémentation du modèle conceptuel ICARE sur supports mobiles, ce projet a donné lieu à la définition de format de capture de données d'usage selon le modèle conceptuel ICARE.

3 Corpus multimodaux au format ICARE

Pour la capture de données d'usage basée sur le couplage d'ICARE avec la sonde ACIDU, le modèle conceptuel ICARE est utilisé et chaque composant ICARE va capturer des données. Cette approche permet d'identifier quatre niveaux d'abstraction dans les données capturées : dispositif, langage, combinaison et tâche. Les données capturées expliciteront ces quatre niveaux d'abstraction : le format de capture de données est appelé Format-ICARE. Aucune information d'un niveau d'abstraction inférieure ne sera perdue, principe appliqué par exemple dans la boîte à outils de capture de contexte d'A. Dey [Dey 2001]. Ainsi lors de l'analyse des données capturées, des filtres par niveaux d'abstraction seront applicables.

Pour chaque composant ICARE mis en jeu, un triplet est mémorisé et contient le type de composant, la valeur en sortie du composant appelé événement ou valeur et la date ou estampille de l'information. Enfin pour chaque modalité pure ou combinée, la tâche ou sous-tâche concernée est mémorisée.

Le format est donc le suivant.

- pour le cas d'une modalité pure

  [(dispositif, événement, date),
  (langage, valeur, date),
  tâche/sous-tâche
  ]

- pour le cas d'une modalité combinée

  [(type de combinaison, valeur, date)
  ((dispositif, événement, date),
  (langage, valeur, date),
  tâche/sous-tâche) /* modalité 1 */
  …
  ((dispositif, événement, date),
  (langage, valeur, date),
  tâche/sous-tâche) /* modalité n */
  tâche/sous-tâche
  ]

Remarque : le format s'applique récursivement si les combinaisons de modalités concernent des modalités elles-mêmes combinées. De plus si des combinaisons du niveau dispositif sont appliquées, le format considère le dispositif comme combiné en indiquant le type de combinaison, la valeur après combinaison et l'estampille de l'événement combiné.

Grâce à ACICARE, des tests peuvent être menés dans un contexte réel d'usage. Par exemple, le fichier de capture (ou fichier de log) ci-dessous, établi à un niveau de capture ICARE-Langage, montre que l'utilisateur a commencé à saisir un contact (Prénom = Patrick, Nom de famille = Patrick) puis a activé le champ "groupe" et enfin a quitté le système sans sauvegarder. La trace montre ensuite que l'utilisateur a relancé le système et a recommencé à saisir le contact en spécifiant le prénom (Patrick).

```
************  Log ACICARE *************
** Niveau de capture: Langage *****

** Langage MicroContenu ** :
                (parole: patrick, temps 28/04/2006 12:58:17)

** Langage Microchamp   ** :
                (champ: last name, temps 28/04/2006 12:58:23)

** Langage MicroContenu ** :
                (parole: patrick, temps 28/04/2006 12:58:31)

** Langage Microchamp   ** :
                (champ: group, temps 28/04/2006 12:58:39)

** Langage CommandeMenuIL              ** :
                ( commande: exit, key: F2, temps 28/04/2006 12:58:58)


************  Log ACICARE *************
** Niveau de capture: Langage *****

** Langage MicroContenu ** :
                (parole: patrick, temps 28/04/2006 13:00:09)

** Langage CommandeHautBasIL    ** :
                (commande: Champ suivant vers le bas, clavier touche: Down, temps 28/04/2006 13:00:55)
```

Dans l'exemple ci-dessus, plusieurs commandes <Annuler> ont été spécifiées de façon involontaire. En effet, il est observable que l'utilisateur reprend l'enregistrement d'un contact après avoir spécifié la commande <Annuler>. La commande <Annuler> a donc été utilisée de façon involontaire, interrompant la tâche principale de l'utilisateur. Aussi il est possible de réduire le risque de spécifier la commande <Annuler> involontairement en forçant l'usage redondant de deux modalités pour spécifier la commande. Pour cela, il suffit d'ajouter le composant ICARE Redondance pour transformer deux modalités initialement équivalentes en modalités redondantes. Ainsi l'utilisateur doit utiliser la parole et la touche dédiée en même temps (dans une fenêtre temporelle spécifiée dans les paramètres du composant ICARE Redondance) pour quitter le système interactif.

ANNEXE 4
TEST FORMEL DE SYSTÈMES INTERACTIFS MULTIMODAUX : COUPLAGE ICARE – LUTESS

La complexification des systèmes interactifs multimodaux rend nécessaire leur développement et leur validation rigoureux. Pour cela, l'application de techniques de test de logiciels réactifs synchrones basées sur l'outil Lutess [Madani 2005] a été étudiée à la validation de systèmes interactifs multimodaux, en s'appuyant sur le constat suivant : un système interactif peut, sous certaines conditions, être considéré comme un système réactif synchrone. S'il est vrai que la rapidité de réaction de systèmes interactifs à un événement externe est moins cruciale que dans le cas des systèmes traditionnellement qualifiés de synchrones, on peut toutefois constater que leurs comportements sont constitués de cycles "action-réaction". Ainsi, des interactions multimodales correspondant à des comportements des utilisateurs ainsi que certaines propriétés du système et de son contexte peuvent s'exprimer dans un formalisme synchrone à des fins de tests.

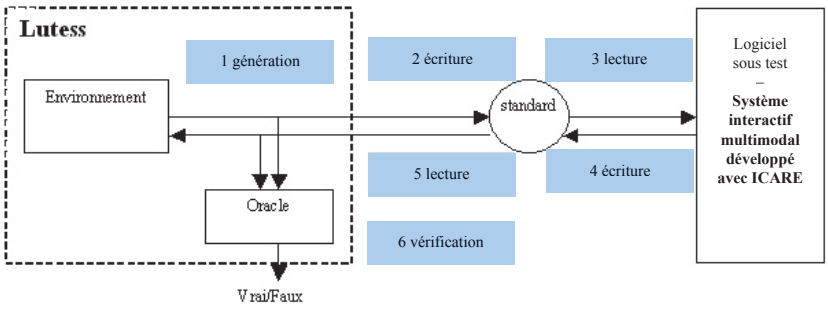
Dans ce contexte, l'étude de techniques de test de logiciels réactifs synchrones basées sur Lutess a été faite pour la validation de systèmes interactifs multimodaux développés selon le modèle conceptuel ICARE. Aussi le couplage de l'outil ICARE développé avec la technologie JavaBeans avec la plate-forme de test Lutess a été effectué par F. Jourde [Jourde 2006].

1 PRINCIPE DE VALIDATION AVEC LUTESS D'UN SYSTÈME MULTIMODAL DÉVELOPPÉ AVEC LA PLATE-FORME ICARE

L'approche repose sur l'hypothèse qu'un système interactif multimodal peut se comporter comme un système réactif synchrone, c'est-à-dire, que son comportement au cours du temps peut être représenté sous la forme d'une suite de couples <entrée, sortie>. Sous cette hypothèse, l'outil de vérification de logiciel réactif synchrone Lutess est utilisé.

Lutess construit automatiquement un générateur de données de test pour le programme sous test. Pour cela, il requiert en entrée une description de l'environnement écrite en langage Lustre ainsi que des directives de guidage. Le test est effectué par cycles successifs action-réaction. Un cycle se déroule comme suit :

1. Lutess génère un vecteur d'entrées.
2. Lutess l'envoie au logiciel sous test puis attend.
3. Le logiciel sous test lit le vecteur d'entrées.
4. Le logiciel sous test réagit en envoyant un vecteur de sorties à Lutess.
5. Lutess lit les sorties du logiciel sous test.
6. Lutess vérifie la propriété décrite dans l'oracle en prenant en compte les entrées-sorties de ce cycle.



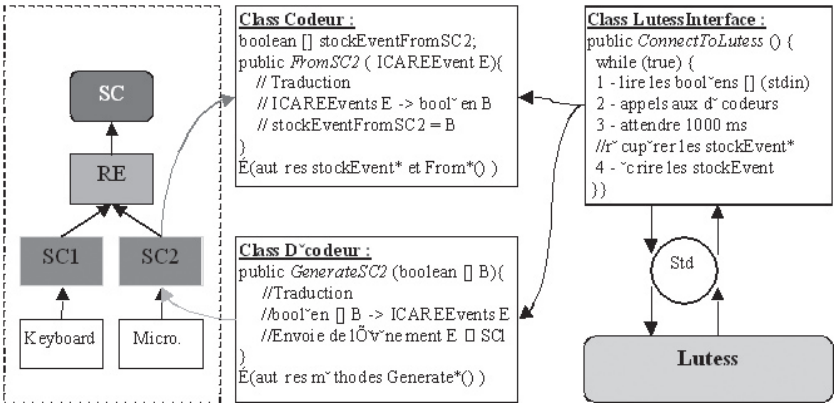
L'originalité de l'approche réside dans la vérification de propriétés ergonomiques de la multimodalité, comme les propriétés CARE. Pour cela, des tests de type boîte noire avec Lutess sont effectués sur des assemblages de composants ICARE.

2 Couplage ICARE-Lutess

Lutess communique avec l'extérieur par la lecture-écriture de booléens sur l'entrée-sortie standard. Ce type de communication implique le développement de deux mécanismes.

- Un mécanisme de communication réalise la lecture-écriture de vecteurs de booléens sur l'entrée-sortie standard du côté du logiciel sous test.
- Un mécanisme de traduction de la sémantique des événements permet à Lutess et au logiciel sous test de se comprendre. Ce mécanisme dépend des entrées-sorties choisies pour être simulées et observées sur le logiciel sous test. Ces entrées-sorties dépendent elles-mêmes des propriétés à vérifier.

La communication s'effectue au moyen d'événements ICAREEvents qui sont échangés au sein d'un schéma ICARE depuis les composants de type Dispositif (niveau inférieur d'un schéma ICARE) jusqu'aux tâches ou commandes (niveau supérieur d'un schéma ICARE). Chaque ICAREEvent porte une sémantique propre correspondant à un niveau d'abstraction dans la fonction qui abstrait les actions de l'utilisateur en tâche ou commande traitée par le système interactif. Lors des tests avec Lutess, un événement par l'instanciation d'un ICAREEvent et son envoi à un composant ICARE est simulé. Il est ainsi possible de simuler le comportement d'un composant C, en émettant des ICAREEvents à destination des composants cibles de C. Dans le schéma ci-dessus, cette communication, entre d'une part un schéma ICARE à gauche incluant deux modalités composées de façon Redondante/Equivalente pour une tâche SC et d'autre part Lutess à droite, est montrée.



L'instrumentation proposée pour la connexion ICARE-Lutess consiste à simuler des événements ICARE et donc à simuler le comportement de composants ICARE. Trois niveaux d'abstraction de simulation sont identifiés :

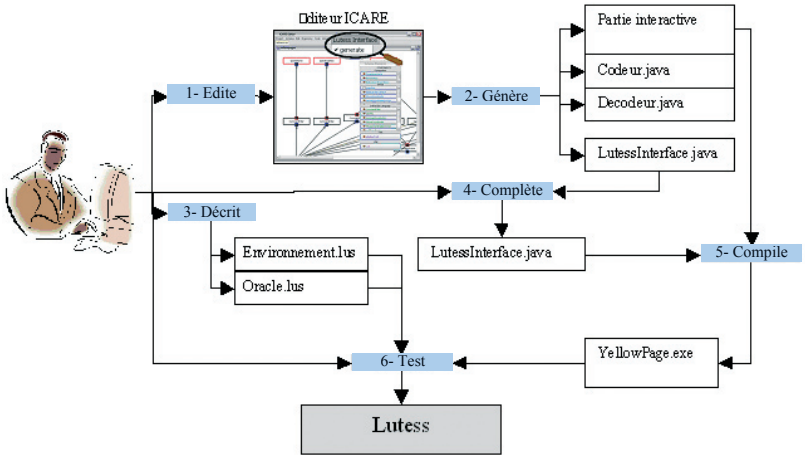
- Niveau dispositif : il s'agit de simuler un composant ICARE de type Dispositif par la génération d'événements ICAREEvents à destination du composant Langage d'Interaction associé.
- Niveau langage d'interaction : il s'agit de simuler un composant ICARE de type Langage d'Interaction. La simulation est réalisée en émettant des événements ICAREEvents à destination des composants cibles de type Combinaison ou Langage d'Interaction ou à destination du reste du système en simulant des tâches élémentaires abstraites ou commandes.

- Niveau combinaison : il s'agit de simuler un composant ICARE de Combinaison par l'émission d'événements à destination de ses cibles : des composants de type Combinaison ou Langage d'Interaction ou encore le reste du système s'il s'agit de tâches élémentaires abstraites.

Ainsi l'approche permet d'effectuer des tests en simulant des entrées de l'utilisateur du système interactif multimodal à différents niveaux d'abstraction. Il est donc possible de simuler des actions physiques de l'utilisateur (Dispositif) et des actions de plus haut niveau d'abstraction (Langage d'Interaction et Combinaison).

3 Extension de l'éditeur ICARE

Le couplage à Lutess a été intégré à l'éditeur ICARE (chapitre 7). L'utilisateur peut l'activer ou la désactiver par la boîte à cocher du menu de l'éditeur ICARE.



Une grande partie du code de connexion présenté est générée automatiquement (étape 2). Pour effectuer des tests, il convient néanmoins de décrire les fichiers environnement et oracle (étape 3), puis de compléter le contenu de la boucle "while" de la classe "LutessInterface" (étape 4). Ces activités manuelles de programmation dépendent des vérifications à effectuer et correspondent à l'écriture d'un total de moins de 100 lignes de code (JAVA/LUSTRE). Après compilation (étape 5), les tests peuvent alors être effectués (étape 6). Tandis que la vérification de nouvelles propriétés sur les mêmes entrées-sorties entraîne la modification de l'oracle seulement (étape 3), effectuer des tests avec de nouvelles entrées-sorties implique de refaire les étapes 3, 4 et 5.

4 Tests de YellowPages

L'approche et son instrumentation au sein de l'éditeur ICARE ont été testées sur le système YellowPages présentés au chapitre 10. Des tests ont été effectués sur l'usage redondant et complémentaire de modalités. En entrée, les composants ICARE Dispositifs sont simulés et en sortie les événements ICARE de plus haut niveau d'abstraction, le niveau des tâches élémentaires abstraites ou commandes sont observés.

Nous détaillons dans cette annexe le cas de l'usage complémentaire ; par l'exemple, l'utilisateur énonce la commande vocale "zoom in here" tout en sélectionnant un point sur la carte avec la souris. Les trois composants de type Dispositif sont simulés. Pour les composants de type Dispositif "Microphone" et "Keyboard", les entrées comme "zoom in/out here" sont simulés. L'environnement inclut un invariant qui interdit la production de plusieurs entrées vocales simultanément. Pour le composant Dispositif "Mouse",

des événements “coordonnées(x,y) sur la carte” sont simulés. Les coordonnées sont modélisées en un vecteur de booléens de 28 bits (14 bits pour x et 14 bits pour y). Le composant Dispositif “Mouse” émet classiquement un événement comprenant les coordonnées du pointeur toutes les 10 ms. Afin de reproduire un comportement proche, il a été décidé de générer un événement à chaque cycle. Ainsi, pour toutes commandes de zoom (clavier / voix) par exemple, une sortie doit être produite. Le nœud LUSTRE correspondant à l’oracle est le suivant :

```
node C1( entree:bool ; entreecoord:bool^28 ; sortie:bool;
sortiecoord:bool^28 ) returns (ok:bool);
let
ok =  not(entree) or
      (VectEqual(28, entreecoord, sortiecoord) and sortie);
tel
```

“VectEqual” est un noeud qui vérifie l’égalité de deux vecteurs. Cette propriété est vérifiée pour chacune des entrées correspondant à des commandes spécifiées de façon complémentaire telles que zoom avant et arrière, centrage sur un point etc. La propriété finale dans l’oracle est alors un «et» logique des C1.

La série de tests n’a pas révélé de viol de l’oracle. Pour simplifier la lecture des résultats ci-dessous, deux événements de type coordonnées sont générés : (x=0, y=0) et (x=0, y=1) notés respectivement Y0 et Y1. Le test démarre dans un état où la carte est affichée.

Cycle	Entrées	Sorties	Oracle
1	Y0	-	vrai
2	C-zoom_in, Y0	CO-zoom_in_point, Y0	vrai
3	C-zoom_out, Y0	CO-zoom_out_point, Y0	vrai
4	Y0	-	vrai
5	Y0	-	vrai
6	V-smaller_here,Y0	CO-zoom_out_point, Y0	vrai
7	Y1	-	vrai
8	Y1	-	vrai
9	Y1	-	vrai
10	V-bigger_point, Y1	CO-zoom_in_point, Y1	vrai

Au cycle 2, nous observons en entrée un événement Clavier “zoom in” et un événement de type coordonnées “Y0”. La fusion des événements correspondant à un usage complémentaire des deux modalités est réalisée et nous observons en sortie la commande complète “zoom in point Y0”. Une telle fusion des événements est aussi observable aux cycles 3, 6 et 10. A l’opposé, lors des cycles 1, 4, 5, 7, 8 et 9, nous observons en entrée qu’un seul événement de type coordonnées et donc pas de sortie puisqu’aucune fusion d’événements multimodaux n’est effectuée.

Résumé

Depuis les travaux fondateurs de R. Bolt « Mets ça là » combinant la voix et le geste, les modalités d'interaction se sont multipliées, diversifiées et améliorées. Les récents paradigmes d'interaction comme les interfaces tangibles incarnées ou la réalité augmentée, couplés aux progrès des systèmes de localisation, à la miniaturisation des dispositifs, à la qualité des réseaux sans fils, à l'amélioration de la reconnaissance de la parole ou de gestes ouvrent un vaste champ de possibilités d'interaction pour les systèmes multimodaux. Dans ce contexte, et bien que de nombreux systèmes multimodaux soient disponibles, leur développement et leur maintien restent encore des tâches difficiles, notamment par manque de réutilisabilité de l'existant. Ce travail de thèse aborde ce problème de conception et de développement pour la multimodalité en entrée (de l'utilisateur vers le système informatique). Nous décrivons un modèle conceptuel de la multimodalité qui organise dans un canevas unificateur les modalités et leurs formes de combinaison. Basé sur ce modèle, nous définissons une approche générique à composants logiciels, notée ICARE, facilitant et accélérant la conception, le développement et le maintien des interfaces multimodales. Nous démontrons l'apport de cette approche par l'outil ICARE qui est une opérationnalisation de notre approche à composants. Un éditeur graphique est fourni, simplifiant la phase d'assemblage des composants et générant automatiquement le code correspondant à l'interaction multimodale. Cinq systèmes multimodaux aux caractéristiques distinctes (systèmes de réalité augmentée, de virtualité augmentée et mobiles) ont été développés avec l'outil ICARE.

Mots-clés

Interaction Homme-Machine, modalités d'interaction, multimodalité, ingénierie logicielle, composants logiciels, outil de conception, réalité augmentée.

Abstract

The area of multimodal interaction has expanded rapidly since the seminal "Put that there" demonstrator of R. Bolt that combines speech and gesture. In parallel with the development of the Graphical User Interface technology, natural language processing and gesture recognition have made significant progress. In addition, recent interaction paradigms such as tangible and embodied user interfaces as well as augmented reality open a vast world of possibilities for interaction modalities. Significant achievements have been made in terms of both modalities and real multimodal systems. Although several multimodal systems have been built, their design and development still remains a difficult task particularly because of a lack of reusability. In this thesis we address this problem of design and development for input multimodal interfaces (from the user to the system). We describe a conceptual model of multimodality as a unified framework for modalities and combinations of modalities. Based on this conceptual model, we define a generic component-based approach called ICARE which allows the easy and rapid design, development and maintenance of multimodal interfaces. We have developed the ICARE tool to prove the usefulness of our component-based approach. The ICARE tool is a graphical platform that enables the designer/developer to graphically manipulate and assemble ICARE software components in order to specify the multimodal interaction. From this specification, the code of the multimodal interaction is automatically generated. Five multimodal systems with different characteristics (augmented virtuality/ reality, mobile systems) have been developed with the ICARE tool.

Keywords

Human-Computer interaction, interaction modalities, multimodality, software engineering, software components, software design tools, augmented reality.