


UNIVERSITE JOSEPH FOURIER – GRENOBLE I
U.F.R EN INFORMATIQUE ET MATHEMATIQUES APPLIQUEES

DEA Informatique : Systèmes et Communication

Projet présenté par :
Jullien BOUCHET

Sous la responsabilité de :
Laurence NIGAY

APPROCHE À COMPOSANTS POUR LA CONCEPTION ET LE DEVELOPPEMENT D'INTERFACES MULTIMODALES

Présenté et soutenu le 23 juin 2003

Jury : M. Jacques BRIAT
Mme Joëlle COUTAZ
Mlle Laurence NIGAY
M. Christian NOUVEL

Laboratoire de Communication Langagière
et Interaction Personne-Système
Fédération IMAG
Unité Mixte de Recherche : CNRS – UJF - INPG

REMERCIEMENTS

Pour commencer, je tiens à remercier mes parents qui m'ont toujours soutenu dans mes choix professionnels et sans lesquels je ne serais pas là aujourd'hui ☺. Je remercie aussi Sandra, le reste de ma famille et mes amis pour leur soutien moral.

D'autre part, comment ne pas citer Laurence qui m'a fait confiance et qui m'a permis de découvrir un domaine passionnant de l'informatique. Je la remercie vivement de m'avoir proposé ce sujet et de m'avoir encadré pendant toute cette année.

Merci également à tous les membre de l'équipe IIHM pour leur aide morale et technique : je pense à Joëlle, Gaëlle, Sophie, François, Philippe, Olfa, Bérangère, Lionel, Julien, Benoît, Yann, Alexandre, Gaétan, Christophe, Nicolas, Chaouki, Sylvain et aux autres étudiants de passage.

Ce mémoire est dédié à toutes ses personnes qui permettent à ma vie d'être agréable ; encore une fois MERCI.

TABLE DES MATIERES

INTRODUCTION

Nos travaux ont trait à la conception et à la réalisation de systèmes interactifs mettant en œuvre plusieurs moyens d'interaction ou de communication entre l'utilisateur et le système informatique. Ces moyens sont appelés « modalités d'interaction » et constituent la partie Interface d'un système interactif. La Figure 1 rappelle le rôle de la partie Interface dans un système interactif. Elle est l'intermédiaire matériel et logiciel entre l'utilisateur et le noyau fonctionnel. Grâce à elle, l'utilisateur peut agir sur le système et percevoir des informations.

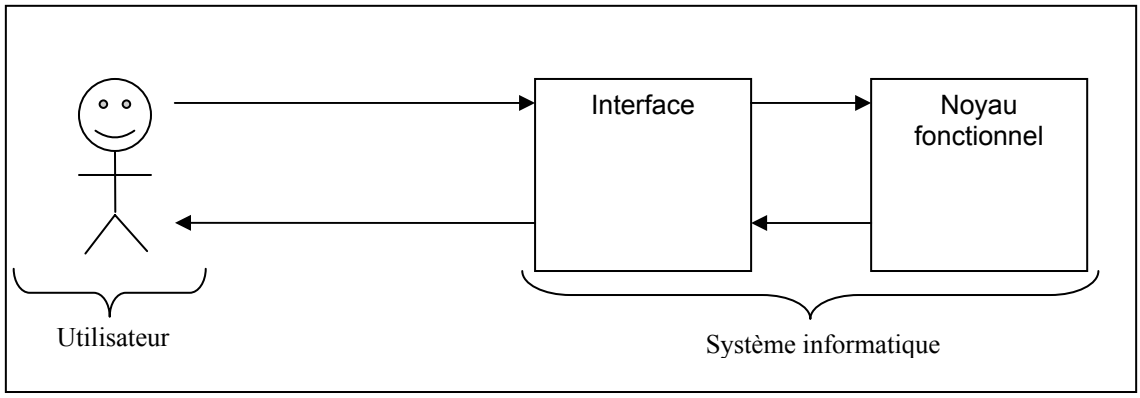


Figure 1 : Système interactif = utilisateur + système informatique

Au sein de la partie Interface, la multiplicité des moyens d'interaction, que nous appelons « multimodalité », prend son sens aussi bien dans la variété des canaux d'interaction mis à disposition de l'utilisateur que dans les diverses manières d'utiliser ces canaux. La multimodalité est un thème de recherche au sein du domaine de l'Interaction Homme-Machine (IHM) assez récent. Dès 1980, la multimodalité est étudiée aux États-Unis : en particulier Richard Bolt propose le paradigme du « mets ça là » dans le cas de l'intégration de deux modalités d'interaction, la parole et le geste [Bolt 80]. 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 Recherche Coordonnée Communication Homme-Machine). Le concept d'interaction multimodale est né.

Malgré des avancées conceptuelles significatives, que ce soit sur la caractérisation de l'usage des modalités ou la spécification de mécanismes de fusion de données issues de modalités distinctes, il n'existe pas, à l'heure actuelle, d'outil de réalisation logicielle permettant de concevoir des systèmes interactifs multimodaux utiles et utilisables. Ainsi, la réalisation logicielle de systèmes interactifs multimodaux continue aujourd'hui d'être une tâche difficile et conséquente.

1 Motivations

La multimodalité joue un rôle important dans les nouvelles applications. L'utilisateur et le système ont la possibilité de combiner plusieurs modalités en fonction de la tâche, des préférences ou des intentions communicationnelles. La multiplicité est un facteur de souplesse. Cependant, leur utilisation pose des problèmes d'interaction de plus en plus difficiles à résoudre car les dispositifs d'entrée/sortie se multiplient et de très nombreuses modalités d'interaction sont envisageables. Les combinaisons de ces modalités élargissent encore un espace de possibilités déjà très vaste. On retrouve les modalités :

- en entrée (de l'utilisateur vers le système) comme la parole, le suivi du regard, l'écran tactile, etc.
- en sortie (du système vers l'utilisateur) comme des graphismes sur un écran d'un assistant personnel (PDA) ou dans un casque de réalité augmentée, une projection sur un objet physique, une manette à retour d'effort, du son spatialisé, etc.

2 Objectifs

Face à cette multiplication des modalités et des systèmes interactifs qui en intègrent plusieurs, l'objectif de l'étude est de définir, concevoir et réaliser un environnement logiciel pour la conception et la réalisation d'interfaces multimodales.

Les deux traits saillants de l'outil de développement à concevoir sont :

- la généralité de la plate-forme d'intégration des modalités : l'objectif est ici de concevoir et développer une plate-forme permettant de combiner plusieurs modalités d'entrée et de sortie.
- le couplage de la plate-forme à des applications existantes : la plate-forme à concevoir et à développer doit pouvoir se connecter à une application existante.

Pour cela la plate-forme contient :

- un moteur de fusion et de fission de modalités d'interaction général et paramétrable,
- un ensemble de modalités d'interaction, d'entrée et de sortie,
- un format d'échange entre les modalités de la plate-forme et les applications existantes.

Notre plate-forme s'appuie également sur les récentes techniques de développement du logiciel. Une approche intéressante présentent de nombreux avantages en termes de coûts de production, de réutilisabilité, de maintenabilité et

d'évolutivité : c'est le développement d'application à base de composants. Cette technique est une adaptation informatique des composants que nous retrouvons en électronique. Les composants sont vus comme des unités logicielles autonomes qui se connectent à d'autres pour former des systèmes plus complexes. Cette approche risque de devenir le standard de ces quinze prochaines années dans le domaine du génie logiciel.

L'originalité de notre approche à composants réside dans l'intégration d'aspects ergonomiques dans le composant. L'ancrage entre l'ergonomie et le génie logiciel devient alors explicite.

3 Structure du mémoire

L'organisation du mémoire reflète notre démarche de travail : espace problème, espace solution et réalisation logicielle.

Le premier chapitre définit l'espace problème de nos travaux. Il comprend six paragraphes :

Le paragraphe 1 fait le point sur la terminologie que nous avons adoptée en définissant notamment les termes « modalité » et « multimodalité ».

Le paragraphe 2 présente le cadre de l'Interaction Homme-Machine en illustrant plusieurs modèles issus de la psychologie cognitive qui définissent des éléments utiles à la conception du comportement de l'utilisateur vis à vis de l'outil informatique.

Le paragraphe 3 expose les espaces taxinomiques dédiés aux interfaces. La diversité des objectifs des études menées a multiplié les espaces taxinomiques. Certains d'entre eux se focalisent uniquement sur un sens de l'interface, soit en entrée, soit en sortie, alors que d'autres essayent de généraliser leur approche en proposant des espaces valides aussi bien en entrée qu'en sortie.

Le paragraphe 4 traite des travaux ergonomiques de la composition des modalités. Nous listons un ensemble de propriétés ergonomiques qui qualifie les compositions de modalités au sein d'un système interactif.

Le paragraphe 5 expose plusieurs approches de fusion visant à composer les interactions issues de modalités différentes. Un espace de classification synthétise notre étude des moteurs de fusion.

Le paragraphe 6 dresse un état de l'art des technologies à composants. Les avantages et les inconvénients des technologies étudiées sont identifiés.

Le deuxième chapitre de ce mémoire définit l'espace solution. Les concepts de l'espace problème sont repris dans un outil de conception et de développement pour les systèmes multimodaux. Réalisé avec des composants logiciels, nous le présentons en détail en trois paragraphes :

Le paragraphe 1 présente le modèle à composants que nous avons choisi. Il ne s'agit pas d'une technologie particulière mais des concepts que nous voulons retrouver dans nos composants logiciels.

Le paragraphe 2 développe notre solution pour le développement des systèmes interactifs multimodaux intégrant des propriétés ergonomiques. Nous verrons que notre plate-forme définit plusieurs types de composants qui peuvent être assemblés et qui constituent au final un moteur de fusion adapté aux besoins de l'application.

Le paragraphe 3 est une synthèse de notre contribution. Nous y reprenons l'espace de classification des moteurs de fusion présenté dans la partie « Espace problème » afin de comparer notre solution avec les approches existantes.

Le troisième chapitre de ce mémoire est dédié à notre réalisation logicielle. Il comprend quatre paragraphes :

Le paragraphe 1 présente le cahier des charges d'une application multimodale développée avec notre plate-forme. Les fonctionnalités, les dispositifs et les contraintes logicielles sont expliqués.

Le paragraphe 2 définit les spécifications externes de l'application développée. Les tâches de l'application sont détaillées et la description de l'interface est présentée.

Le paragraphe 3 expose l'architecture logicielle de l'application, correspondant à la phase de conception logicielle. De plus, l'architecture du moteur de fusion reposant sur notre plate-forme logicielle à composants est détaillée.

Le paragraphe 4 est quant à lui dédié à la phase d'implémentation. Le passage de la phase de conception à la phase de codage de nos composants est expliqué.

En conclusion, nous soulignons les points contributifs de nos travaux et nous développons les perspectives de travaux futurs.

I. ESPACE PROBLEME

La multiplication des dispositifs d'interaction (PDA, écran tactile, etc.) et la prise de conscience de l'utilisateur au centre du processus de développement d'une application ont engendré de nouveaux besoins concernant la conception et la réalisation des systèmes interactifs.

Ce chapitre établit le cadre de nos travaux sur la multimodalité. Les concepts présentés dans l'espace problème servent de fondements à notre contribution présentée dans le chapitre II « Espace solution ». Nous retrouvons : une terminologie des termes « modalités » et « multimodalité », les études de psychologie cognitive sur les comportements de l'humain utilisant un système informatique, les espaces taxinomiques sur l'interaction Homme-Machine, les propriétés ergonomiques des systèmes multimodaux, les moteurs de fusion existants et un état de l'art des technologies à composants.

1 Terminologie : Modalité et Multimodalité

L'absence de consensus parmi la communauté scientifique sur la définition de ces deux termes nous incite à lever l'ambiguïté de leurs définitions dans ce mémoire. Deux manières d'appréhender le problème existent : soit ces notions sont abordées par rapport à l'individu, soit par rapport à la technologie. Bien que notre approche soit centrée sur l'utilisateur, notre objectif est de réaliser un outil de développement. Au contraire des sciences humaines qui ont un point de vue centré individu, notre domaine de réflexion est donc orienté ici vers la technologie.

1.1 Modalité

Pour nos travaux, nous reprenons la définition de Laurence Nigay [Nigay 96] : une modalité est définie comme un couple <d, l> où d désigne un dispositif physique et l, un langage d'interaction. Un langage d'interaction se définit par un vocabulaire d'éléments terminaux et une grammaire. Les éléments terminaux sont produits ou captés par les dispositifs d'entrée/sortie. Comme le montre la Figure 2, cette définition caractérise les échanges entre le système et l'utilisateur car elle identifie et met en relation deux niveaux d'abstraction dont le niveau physique (dispositif) et le niveau logique (langage d'interaction).

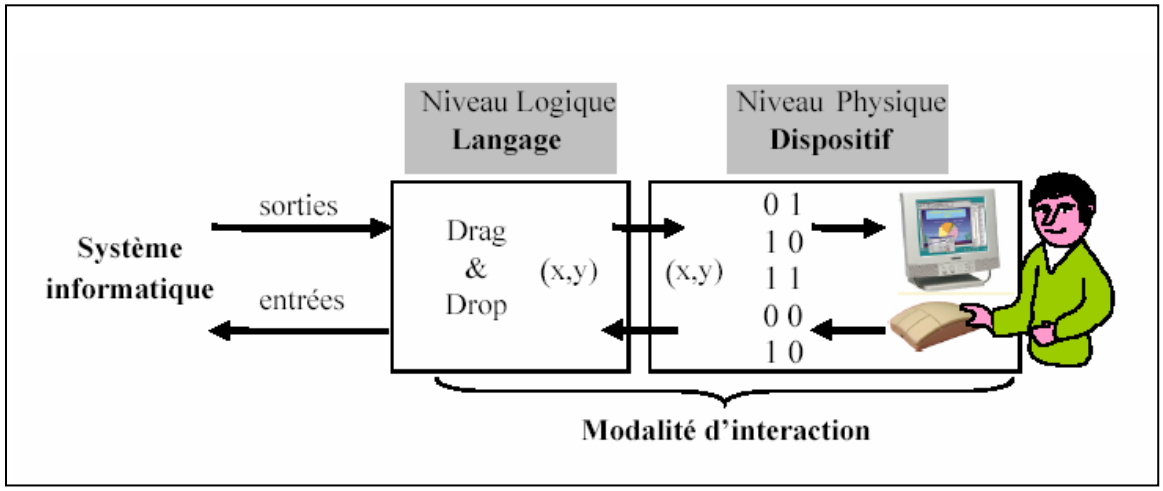


Figure 2 : Niveaux physique et logique d'une modalité d'interaction [Vernier 01]

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 de même sens (entrée ou sortie) permet de faire la distinction entre la multiplicité de 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 et de multimodal en sortie si au moins deux modalités de sortie existent.

Après avoir établi la terminologie adoptée dans ce mémoire, nous nous intéressons aux études de psychologie cognitive sur l'interaction Homme-Machine.

2 Cadre de l'interaction Homme-Machine

D'après [Coutaz 00] : « *L'Interaction Homme-Machine (IHM) 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, la sociologie, etc.

En particulier, 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 la conception des systèmes interactifs. Nous présentons trois modèles qui posent le cadre de notre étude centrée utilisateur de la multimodalité.

2.1 Modèle de Rasmussen [Rasmussen 86]

Le modèle de Rasmussen décompose les comportements humains selon trois niveaux de contrôle :

- les comportements basés sur les réflexes,
- les comportements basés sur les règles,
- les comportements basés sur les connaissances.

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 et cela lui demande un effort cognitif très court. 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 lourd.

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).

Ces niveaux d'expertise de l'utilisateur nous semblent pertinents pour expliquer les choix des modalités utilisés par l'utilisateur ainsi que les formes de modalités.

2.2 Modèle du Processeur Humain (MPH) [Card 83]

Ce modèle offre, dans une terminologie informaticienne, un cadre fédérateur pour les différentes connaissances en psychologie. Ce modèle voit 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 a permis de mesurer les différents temps de traitement et de persistance des informations dans les sous-systèmes.

Pour la multimodalité, il permet de mettre en évidence les différents traitements humains liés à des sens de perception.

2.3 Théorie de Norman [Norman 86]

En complément des modèles présentés précédemment, cette théorie décompose l'accomplissement d'une tâche par l'utilisateur en étapes. Pour réaliser son but, l'utilisateur fait un effort cognitif 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 3, la distance d'exécution et la distance d'évaluation se décomposent en trois étapes.

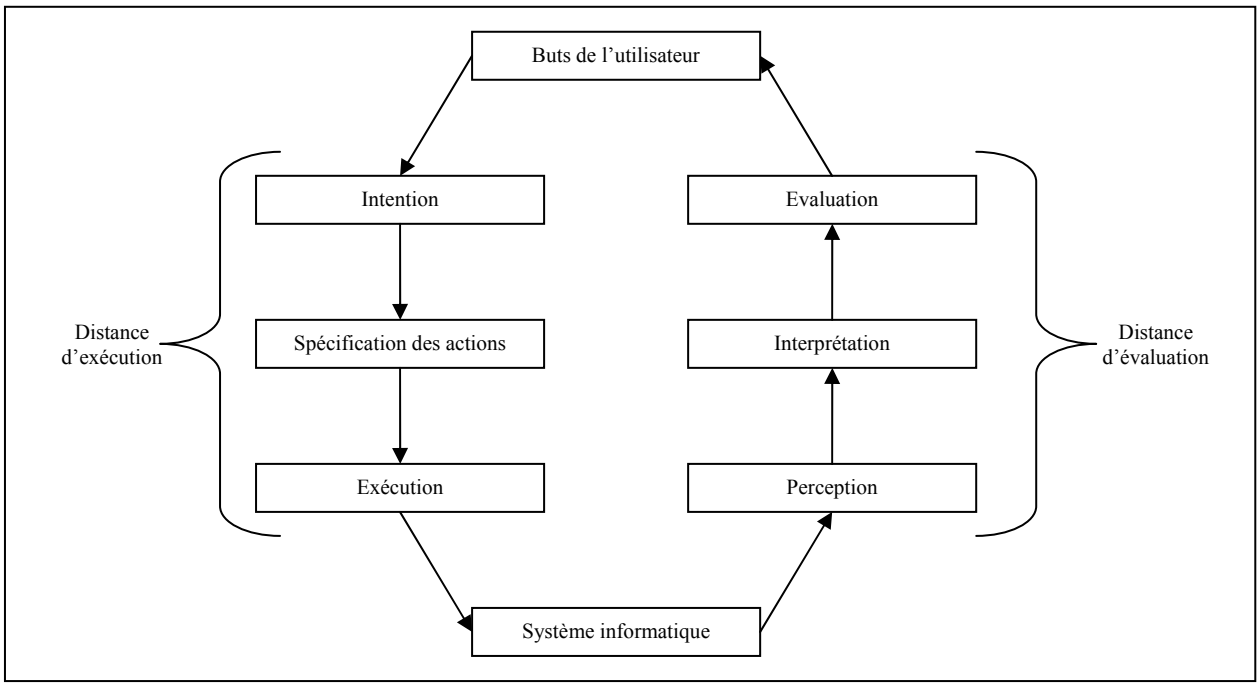


Figure 3 : Théorie de l'action de Norman

La théorie de l'action est très générale. 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 à la modalité employée en sortie par le système.

Ces modèles définissent des cadres généraux. Dans les sections suivantes, nous nous focalisons sur l'interaction les espaces taxinomiques de l'interaction Homme-Machine.

3 Espaces taxinomiques

Différents espaces taxinomiques pour les interfaces ont été définis. Cette multiplicité des espaces est en partie due à la distinction qui peut être faite entre l'interface d'entrée et l'interface de sortie. L'interface d'entrée est constituée des moyens communicationnels mis à disposition de l'utilisateur pour modifier l'état du système. L'interface de sortie correspond aux moyens communicationnels exploités par le système pour présenter son état à l'utilisateur. Cette distinction se retrouve au niveau de la théorie de Norman présentée au paragraphe 2, où la distance d'exécution correspond à l'interface d'entrée et la distance d'évaluation à l'interface de sortie. Ainsi, certains espaces dissocient explicitement ces deux facettes tandis que d'autres se focalisent sur une seule des facettes.

3.1 Interaction en entrée

Les espaces taxinomiques de ce paragraphe sont dédiés aux modalités d'entrée. Les deux premiers espaces présentés sont consacrés aux dispositifs physiques, tandis que le dernier est de plus haut niveau d'abstraction en considérant les tâches que l'utilisateur peut réaliser.

3.1.1 Espace de Buxton [Buxton 83]

Cet espace caractérise les dispositifs d'entrée selon trois axes. Dans cette taxinomie, le dispositif est vu 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 le degré d'engagement physique de l'utilisateur : le type direct demande un geste plus important de l'utilisateur que le type indirect. 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.

3.1.2 Espace de Card, Mackinlay et Robertson [Card 90]

Cet espace concerne aussi les dispositifs physiques et complète l'approche de Buxton. Leur 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 de la théorie expriment la possibilité d'assembler des dispositifs. La composition de deux dispositifs est le produit cartésien des domaines d'entrée des dispositifs. Par exemple, la position d'une souris est le produit cartésien des deux dispositifs élémentaires captant respectivement l'ordonnée et l'abscisse. Néanmoins, et comme pour l'espace de Buxton, cette taxinomie ne concerne que les dispositifs manipulés par l'utilisateur. Ici aussi, les dispositifs tels que les microphones ne sont pas pris en compte.

3.1.3 Espace de Foley [Foley 84]

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). La taxinomie de Foley est montrée à la Figure 4. 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.

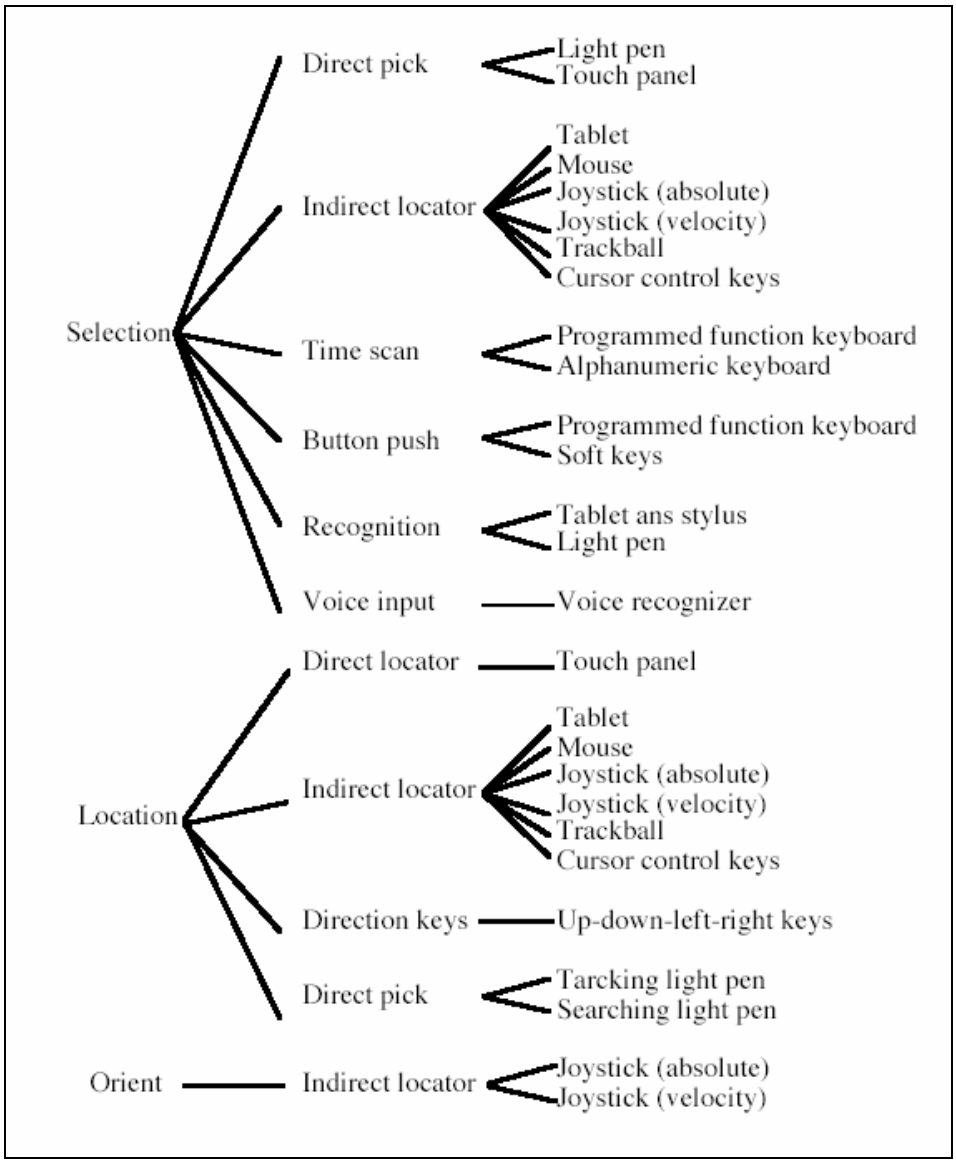


Figure 4 : Taxinomie de Foley pour les tâches graphiques [Foley 84]

Les espaces dédiés aux interfaces en entrée étant présentés, nous nous intéressons aux cadres taxinomiques des interfaces en sortie.

3.2 Interaction en sortie : la taxonomie de Bernsen [Bernsen 93]

Dans ce paragraphe, une taxonomie est présentée. Il s'agit de celle de Bernsen qui est la seule dédiée exclusivement aux modalités de sortie.

La taxonomie de Bernsen traite des interfaces de sortie en présentant un espace permettant de classer ce qu'il appelle les « modalités représentationnelles » comme un texte, un graphe ou une icône. Nous devons comprendre ici 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 » de Bernsen représente le langage d'interaction dans notre définition de la 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, tactiles}. Le profil regroupe un ensemble de propriétés qui définissent si la « modalité représentationnelle » est :

- 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).

Par exemple, la « modalité représentationnelle » de la langue naturelle écrite a pour média « graphique » et un profil {statique, linguistique, non analogique, non arbitraire}.

En combinant l'ensemble des caractéristiques définies et en éliminant les incohérences (par exemple, les médias son et toucher ne peuvent être statiques), nous obtenons 28 « modalités représentationnelles ».

[Nigay 94] propose d'affiner la dimension statique/dynamique avec les notions introduites par Sellen [Sellen 92] qui définissent trois critères pour les retours d'informations : 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 01] prolonge aussi 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 du langage : la langue naturelle écrite est par exemple plus complexe que la « modalité représentationnelle » des langages reconnus par ordinateur.

En résumé, Bernsen est le seul à avoir proposé un espace dédié exclusivement aux modalités d'interaction de sortie. Sa taxinomie est intéressante car elle intègre les capacités cognitives humaines.

3.3 Interaction en entrée et en sortie

Nous finissons notre revue des espaces taxinomiques par ceux qui considèrent les modalités d'entrée et de sortie.

3.3.1 Espace de Frohlich [Frohlich 91]

Cet espace prend comme point de départ le Modèle du Processeur Humain présenté au paragraphe 2.2 pour définir un espace de conception. L'analyse de Frohlich distingue les entrées et les sorties. Il y a donc deux espaces de conception correspondant présentés à la Figure 5 et la Figure 6. Cinq notions sont distinguées : 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 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 voix, le canal de l'interface correspondant est l'audio.

Le média désigne le système représentationnel pour l'échange d'informations. Enfin le style correspond aux techniques d'interaction comme les champs à remplir ou les icônes.

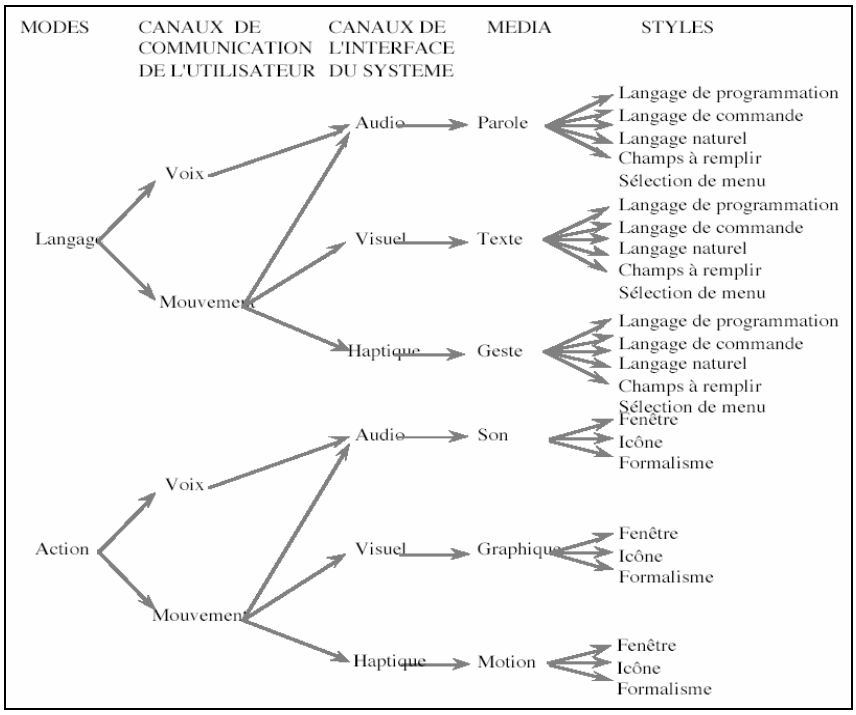


Figure 5 : Espace de conception des interfaces en entrée de Frohlich

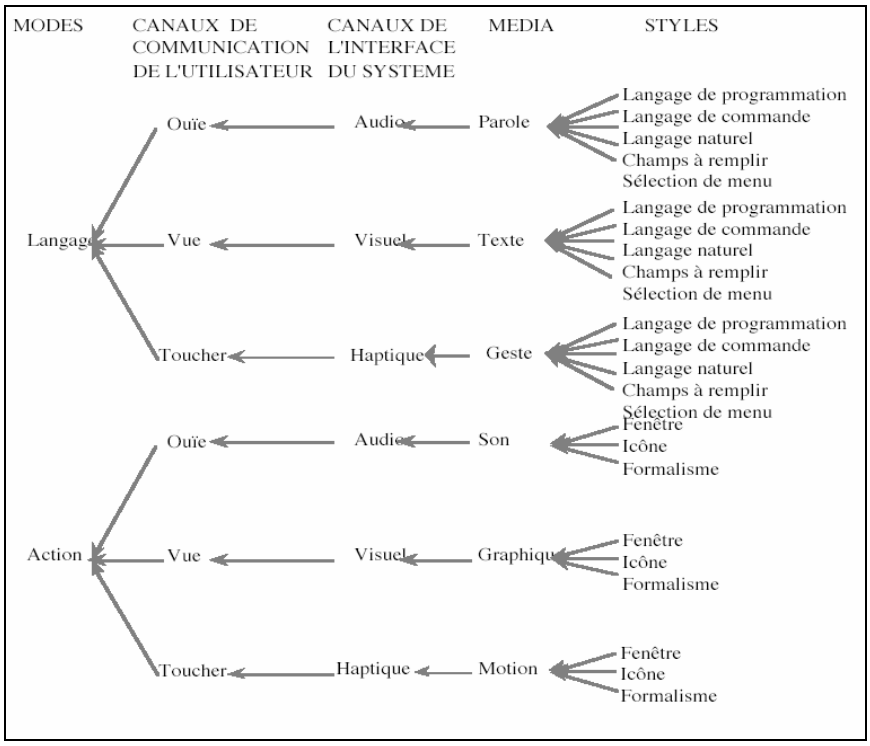


Figure 6 : Espace de conception des interfaces en sortie de Frohlich

En accord avec notre terminologie d'une modalité, le mode et les canaux représentent le dispositif alors que le média et le style sont des affinements du langage d'interaction. Cet espace met en évidence que le sens humain constitue une caractéristique des dispositifs.

3.3.2 Espace MSM [Nigay 94]

Cet espace est lui aussi un espace de conception pour les interfaces multimodales. Cependant, cette taxinomie est plus générale que celle de Frohlich. MSM propose six axes présentés à la Figure 7.

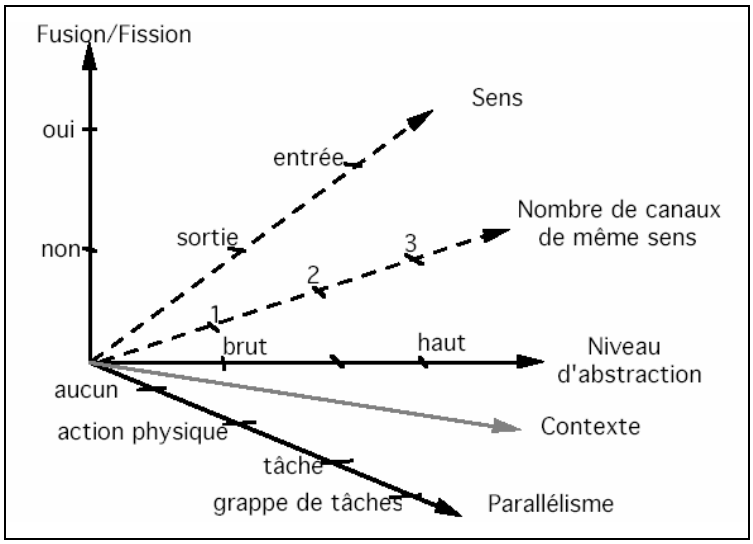


Figure 7 : Espace MSM [Nigay 94]

Les deux premières dimensions, le nombre et le sens des canaux permis pour un système donné, ont trait à la notion de canal de communication. Ces notions font l'objet du paragraphe a). 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. Ces dimensions sont présentées au paragraphe b).

a) Canal de communication

Un 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. Cette définition met l'accent, non pas sur l'aspect liaison, mais sur les intervenants de l'acte de communication : la source et le récipient. La définition s'applique aussi bien aux canaux humains qu'aux canaux digitaux. Par exemple, le canal visuel humain comprend plusieurs capteurs, les rétines, et un sous-système local visuel qui traduit les relations spatio-temporelles des photons en une représentation interne acceptable pour les sous-systèmes représentationnels internes.

b) Fonctions d'interprétation et de restitution

L'information acquise par les canaux d'entrée digitaux est transformée par les processus internes du système. Cette suite d'opérations modélise la fonction d'interprétation. Dans l'autre sens, l'information interne (c.-à-d. l'état interne du système) est transformée jusqu'à ce qu'elle soit prête à être traitée par les canaux de sortie du système. Cette suite d'opérations constitue la fonction de restitution. A chaque canal d'entrée, nous associons une fonction d'interprétation qui prolonge au cœur du système, les capacités de calcul du canal. De même, à chaque canal de sortie, nous associons une fonction de restitution. Interprétation et restitution font intervenir quatre ingrédients en intime relation : niveaux d'abstraction, contexte, fusion et fission, parallélisme.

Niveaux d'abstraction

La notion de niveau d'abstraction exprime le degré de transformations subies 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). Une fonction d'interprétation se caractérise par son pouvoir d'abstraction des évènements reçus du canal d'entrée auquel elle est associée. Une fonction de restitution se qualifie par son pouvoir de matérialisation depuis les représentations internes jusqu'aux représentations de bas niveau de son canal de sortie. En raison de la correspondance bi-univoque entre un canal et une fonction, 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.

Contexte

La capacité d'une fonction à abstraire ou à matérialiser peut dépendre de variables contextuelles. Nous conviendrons qu'un contexte comprend un ensemble de paramètres d'état utilisés par les processus internes pour contrôler l'interprétation ou la restitution. Il agit comme un filtre cognitif identifié dans le modèle du processeur humain (paragraphe 2.2). Dans les systèmes actuels, nous observons deux contextes d'interprétation pertinents : 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. Par exemple, dans les systèmes de traitement de texte, les saisies au clavier ou à la souris qui correspondent aux commandes sont comprises par le système qui déclenche à son tour une action, tandis que ces mêmes saisies à destination du contenu des documents ne sont pas interprétées. Ainsi, le pouvoir d'abstraction ou de matérialisation d'un système est propre à chaque canal, mais pour un canal donné, ce pouvoir dépend du contexte.

Fusion et fission

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 » [Bolt 80], nécessite la fusion de l'événement parole reçu via le canal du microphone avec les évènements souris émanant certainement d'un autre canal. Mieux connue est la fusion de clics souris répartis sur une palette et une zone de dessin pour dessiner une figure géométrique. Ces informations issues du même canal transitent selon des contextes différents pour être regroupés à un haut niveau d'abstraction.

En restitution, la fusion intervient également à plusieurs niveaux d'abstraction. Au niveau le plus haut, elle a lieu dans l'adaptation des informations du noyau fonctionnel aux besoins de l'interface adaptés à l'utilisateur et à sa tâche. Au niveau le plus bas, elle se manifeste par exemple sous forme d'incrustations (vidéo et graphique).

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.

La fission en restitution revêt plusieurs formes. La plus courante est la représentation multiple d'un même concept sur un canal donné. Par exemple, le concept de température est restitué sous forme d'un thermomètre gradué ou d'un réel. On dit que les deux représentations sont équivalentes. La représentation multiple peut aussi s'effectuer en coréférence sur des canaux distincts tel le message oral « attention à cette température » accompagné de l'affichage en rouge du thermomètre à surveiller. Dans ce cas, nous parlons de complémentarité.

Parallélisme

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 80]. En sortie, le parallélisme signifie que plusieurs primitives de sortie peuvent être produites simultanément sur un même canal (comme des animations graphiques) ou sur plusieurs canaux. Notons que la fusion et la fission à bas niveau d'abstraction dépendent de l'existence du parallélisme au niveau physique. On appelle tâche élémentaire, toute tâche que l'utilisateur peut accomplir avec le système via une et une seule commande. Au niveau tâche élémentaire, le parallélisme en entrée autorise l'utilisateur à construire plusieurs commandes de manière concurrente : en parallélisme vrai si le parallélisme est géré au niveau physique ou, en l'absence de parallélisme physique, de manière entrelacée. En sortie, une tâche élémentaire recouvre l'ensemble des primitives de sortie pour exprimer un changement d'état du système. Nous appelons grappe de tâches, un ensemble de tâches élémentaires permettant à l'utilisateur d'accomplir une tâche composée. Au niveau grappe de tâches, le parallélisme exprime les possibilités d'entrelacement entre des espaces de travail. Par exemple, pour un robot mobile de surveillance, les tâches de l'utilisateur peuvent être structurées en trois espaces : description des lieux géographiques, spécification de missions, et contrôle d'exécution de mission. Les relations entre ces trois espaces peuvent avoir une incidence sur l'utilisation temporelle des canaux de communication et donc sur les fonctions d'interprétation et de restitution du système.

3.3.3 UOM : classification des systèmes interactifs [Nigay 94]

UOM est une méthode de classification des systèmes interactifs qui se décompose en trois volets selon que l'on considère l'Usage, le caractère Optionnel ou la Multiplicité des langages d'interaction et des dispositifs physiques du système interactif.

La multiplicité des langages d'interaction et des dispositifs physiques est caractérisée avec le volet M^2LD présenté à la Figure 8. La distinction entre l'interface d'entrée et l'interface de sortie y est présente. La multiplicité des dispositifs et des langages est ensuite considéré : la valeur « non » sur un axe signifie que le système est « mono » pour cette dimension. La valeur « oui » correspond au cas « multi ».

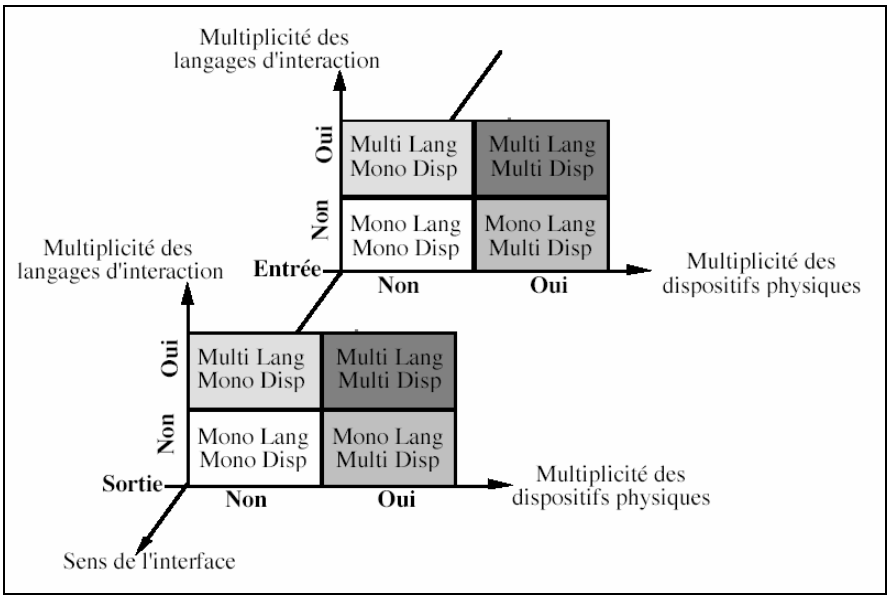


Figure 8 : M²LD

Le caractère optionnel des langages d'interaction et des dispositifs physiques est décrit par le volet O²LD présenté à la Figure 9. 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. L'axe vertical représente le choix du langage d'interaction pour une tâche donnée et l'axe horizontal représente le choix du dispositif physique pour un langage donné.

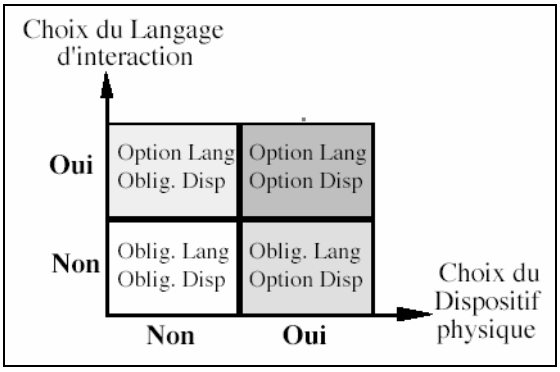


Figure 9 : O²LD

L'Usage des Langages d'interaction et des Dispositifs est identifié grâce au volet appelé ULD. Chaque langage d'interaction et chaque dispositif du système interactif peuvent être catégorisé dans le schéma visible de la Figure 10. Selon qu'ULD est appliqué à un dispositif ou à un langage d'interaction, la légende de la Figure 10 change. La Figure 11 présente la légende d'ULD appliquée au dispositif tandis que la Figure 12 celle d'ULD appliquée au langage.

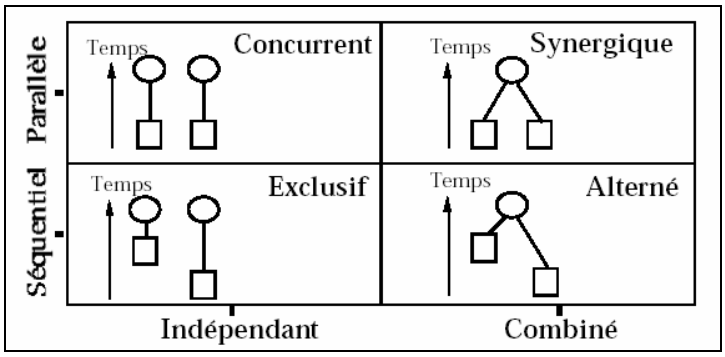


Figure 10 : Usage des Langages et des Dispositifs

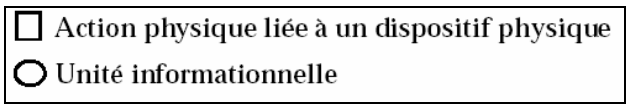


Figure 11 : Légende d'ULD appliquée aux dispositifs

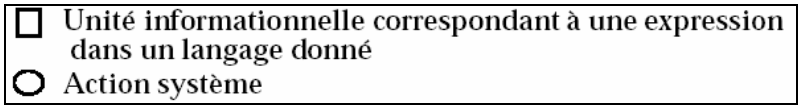


Figure 12 : Légende d'ULD appliquée aux langages

La méthode UOM offre au travers de ces trois volets M^2LD, O^2LD et ULD une métrique de la multimodalité. Nous verrons au paragraphe suivant que certains aspects d'UOM sont en relation avec l'utilisabilité (ergonomie). Par exemple, la souplesse (propriété d'ergonomie) peut se mesurer en termes des choix offerts à l'utilisateur (O^2LD).

Les espaces qui ont été présentés identifient des traits saillants d'une modalité et de la multimodalité que nous allons intégrer dans notre démarche de développement d'un système interactif multimodal. Les espaces de Buxton [Buxton 83], Card [Card 90] et Foley [Foley 84] permettent de définir les propriétés des dispositifs physiques. L'espace de Bernsen [Bernsen 93] propose des caractéristiques intéressantes pour les langages d'interaction. Les travaux de Frohlich [Frohlich 91] mettent en évidence différents niveaux d'abstraction retrouvés aussi dans MSM [Nigay 94]. De plus, MSM ajoute les notions d'interprétation, de restitution, de fusion, de fission, de contexte et de parallélisme qu'il faut aussi prendre en compte dans notre modèle à composants pour le développement de systèmes multimodaux. Enfin, la classification UOM [Nigay 94] montre avec sa facette ULD les différents types d'usages des dispositifs et des langages d'interaction à couvrir dans notre plate-forme.

Nous exploitons les taxinomies présentées pour identifier les concepts de base que notre outil doit intégrer, comme les niveaux d'abstraction, les types d'usage de la multimodalité. Comme nous l'avons annoncé en introduction, nous visons aussi un lien explicite entre l'utilisabilité que nous déclinons en termes de propriétés d'ergonomie et le développement. Pour cela, nous souhaitons intégrer dans l'outil de développement des propriétés ergonomiques que nous étudions dans le paragraphe suivant.

4 Ergonomie dans la composition des modalités

Dans le développement d'applications classiques, l'intégration et le respect des propriétés ergonomiques permettent d'assurer l'utilisabilité du logiciel final. Cette utilisabilité qui s'inscrit avec d'autres facteurs [McCall 77] définit la qualité générale du logiciel. Nous retrouvons, dans [Coutaz 00], une liste de propriétés ergonomiques destinées aux systèmes interactifs. Cette partie traite plus spécifiquement des propriétés relatives aux systèmes interactifs multimodaux.

4.1 Propriétés CARE [Coutaz 94]

Dans [Coutaz 94], un espace conceptuel ergonomique pour l'interaction multimodale est présenté : les propriétés CARE. Cet espace définit quatre propriétés correspondant aux relations qui existent entre dispositifs, langages d'interaction et tâches. C, A, R et E sont les initiales des quatre propriétés : Complémentarité, Assignation, Redondance, Equivalence.

- L'*assignation* exprime l'obligation, pour un état de départ et d'arrivée fixés, d'utiliser une modalité. Cette propriété exprime donc l'absence de choix pour réaliser une tâche, consistant à passer d'un état de départ à un état d'arrivée. Ceci n'est pas forcément un point négatif pour l'interaction. Tout d'abord, l'assignation d'une modalité en sortie à une tâche permet de garantir une stabilité de la forme des expressions de sortie. L'assignation dénote cependant un manque évident de souplesse mais peut néanmoins contribuer à la robustesse en sortie. L'exemple le plus fréquent est le message d'erreur. Ce dernier doit pouvoir être produit quel que soit l'état d'instabilité du système, comme par exemple la déficience du système de gestion du son ou du graphisme.

- L'*équivalence* de modalités se définit pour un état de départ et d'arrivée fixés : l'équivalence d'un ensemble de modalités est vérifiée si chaque modalité permet d'atteindre l'état d'arrivée depuis l'état de départ en une seule étape. Deux modalités peuvent être équivalentes pour un acte de communication ou une tâche.

- La *redondance* dénote l'utilisation séquentielle ou parallèle de plusieurs modalités équivalentes. En entrée, la redondance d'information en provenance de l'utilisateur implique la prise en compte d'une seule des modalités par le système, l'autre pouvant éventuellement contribuer à désambiguïser l'expression obtenue. En sortie, la redondance est un choix de conception liée aux deux propriétés d'ergonomie d'observabilité et d'insistance. En effet si le concepteur a choisi de montrer de différentes manières une même information, cela a pour conséquence de renforcer son observabilité et constitue aussi un moyen pour attirer l'attention de l'utilisateur (propriété d'insistance).

- La *complémentarité* entre les modalités d'un ensemble M exprime le fait que pour passer d'un état à un autre, il faille utiliser toutes les modalités de l'ensemble M. Cela signifie qu'aucune des modalités de l'ensemble M ne suffise à elle seule pour passer de l'état de départ à l'état d'arrivée. Toutefois, il n'est pas exclu

qu'une modalité non contenue dans M puisse permettre cette transition. Comme pour la redondance, l'utilisation complémentaire de modalités peut être parallèle ou séquentielle. En sortie, la complémentarité implique que l'utilisateur établisse les liens entre les informations présentées selon des modalités différentes : l'utilisateur doit combiner les informations perçues pour interpréter l'expression de sortie. Cela implique donc une fusion au niveau perceptuel ou cognitif.

4.2 Affinement des propriétés CARE [Vernier 01]

[Vernier 01] 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 13 illustre cette mise en correspondance. Tirés des travaux d'Allen [Allen 83] sur l'aspect temporel, les cinq schémas de composition illustre la composition de modalités éloignées (1ère colonne de la Figure 13), la composition de modalités avec un point de contact (2ème colonne de la Figure 13), la composition de modalités avec une intersection non vide (3ème colonne de la Figure 13), la composition de modalités dont l'une est plus étendue et englobe l'autre (4ème colonne de la Figure 13) et la composition de modalités de même étendue (5ème colonne de la Figure 13).

Cet espace de classification approfondit les deux propriétés CARE concernant la composition, la Complémentarité et la Redondance. Ainsi, nous remarquons que 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

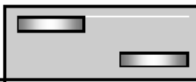
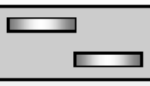
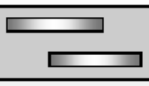
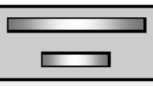
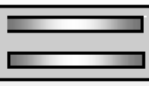
Aspects de composition \ Composition					
Temporelle	Anachronique	Séquentielle	Concomitante	Coïncidente	Parallèle / Simultanée
Spatiale	Disjointe	Adjacente	Intersectée	Imbriquée	Recouvrance
Articulatoire	Indépendance	Fissionnée	Fissionnée + Dupliquée	Partiellement Dupliquée	Dupliquée
Syntaxique	Différente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

Figure 13 : Application des schémas de composition aux cinq aspects de composition [Vernier 01]

Nous qualifions ces propriétés comme étant ergonomiques. Elles caractérisent l'usage des modalités. Dans le paragraphe suivant, nous adoptons le point de vue système et nous étudions les différentes approches logicielles correspondant aux usages de plusieurs modalités.

5 Moteurs de fusion

Les interfaces Homme-Machine multimodales autorisent l'utilisation conjointe de différentes modalités. Cette partie expose différents processus d'intégration de modalités au sein d'une architecture logicielle. Pour cet objectif, le paragraphe 5.1 présente notre grille d'analyse, étendue de celle de [IHM 92]. Ces caractéristiques sont ensuite illustrées par des modèles d'intégration existants, appelés aussi « Moteur de fusion » au paragraphe 5.2.

5.1 Grille d'analyse

Les travaux des ateliers sur l'ingénierie des interfaces Homme-Machine [IHM 92] ont permis d'identifier plusieurs propriétés pour les moteurs de fusion. De plus, d'autres besoins du génie logiciel (prise en compte des capacités de l'utilisateur, prise en compte de l'environnement, etc.) ont mis en évidence de nouvelles propriétés à prendre en compte. Nous en proposons une liste permettant de caractériser un moteur de fusion :

- Type du système multimodal [IHM 92] : il existe quatre types qui correspondent au volet ULD de la méthode UOM présentée au paragraphe 3.3.3. Il s'agit des systèmes de multimodalité alternée, exclusive, synergique ou concurrente. Il est possible que le système soit hybride et qu'il couvre plusieurs valeurs.

- Stratégie d'intégration [IHM 92] : la stratégie peut être précoce ou différée. Une stratégie précoce permet aux unités informationnelles issues des modalités d'être 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 intégration.

- Stratégie d'exploration [IHM 92] : le processus d'intégration élabore un espace de possibilités de relation entre les modalités. Cet espace est parcouru soit en largeur d'abord, soit en profondeur d'abord. La stratégie en largeur d'abord identifie les relations possibles entre les modalités à un niveau d'abstraction donné et choisit la meilleure avant de passer au niveau suivant. La stratégie en profondeur d'abord exploite la première relation et l'enrichit à travers les niveaux d'abstraction en fonction des relations structurelles et sémantiques.

- Proximité temporelle [IHM 92] : les événements issus des modalités sont mis en correspondance s'ils sont produits à des instants très proches. 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 83] et de Vernier [Vernier 01] définissent plusieurs cas de modalités proches temporellement.

- Complémentarité logique [IHM 92] : les événements structurellement complémentaires sont fusionnés même s'ils sont temporellement distants.

- Complétude d'une structure de données [IHM 92] : Le passage de l'information à un plus haut niveau d'abstraction est conditionné par la complétude de cette information.

- Incompatibilité des modalités [IHM 92] : le processus évite d'intégrer les modalités ne pouvant être utilisées conjointement.

- Représentation des informations [IHM 92] : le formalisme peut être unique (quel que soit le niveau d'abstraction) ou multiples (adapté au niveau d'abstraction).

- Couverture du cycle de vie de développement : le modèle du moteur de fusion peut préciser la réalisation des phases de spécification, de conception et d'implémentation.

- Prise en compte des propriétés ergonomiques : capacité du moteur de fusion à intégrer explicitement les propriétés ergonomiques des systèmes interactifs multimodaux.

- Intégration du modèle de l'utilisateur : capacité du moteur à maintenir des informations sur l'utilisateur et ses capacités. Par exemple, la situation de stress d'un militaire, en situation de combat, réduit ses capacités à interagir avec le système.

- Prise en compte de l'environnement : capacité du moteur à recevoir des informations sur l'environnement. Par exemple, l'utilisation d'une modalité de sortie sonore peut s'avérer inutile si l'environnement est bruyant.

- Généricité du moteur de fusion : capacité du moteur pour être intégré à n'importe quelle application.

5.2 Analyse des approches de fusion

La grille d'analyse du paragraphe 5.1 permet de comparer les différents moteurs de fusion. Dans ce paragraphe, nous présentons une plate-forme d'intégration générale et quatre autres systèmes d'intégration conceptuels utilisés dans des applications multimodales.

5.2.1 Plate-forme d'intégration multimodale du W3C [W3C 03]

Le W3C propose un modèle général pour les systèmes interactifs multimodaux du web. Ce modèle, appelé plate-forme d'intégration multimodale, n'est pas une architecture mais se situe à un niveau d'abstraction supérieur. Comme présenté à la Figure 14, la plate-forme est constituée des composants entrées, sorties, gestionnaire d'interaction et de trois autres composants relatifs à l'application (Fonctions de l'application, Sessions des composants, Système et environnement). Les modalités 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 ont besoin d'être 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 l'application se fait dans le gestionnaire d'interaction. Par exemple, une carte est visible à l'écran et l'utilisateur souhaite connaître le nom d'une place visible sur cette carte. L'utilisateur peut formuler sa requête en combinant la parole et l'utilisation de la souris de la façon suivante :

<Parole : Quel est le nom de cette place ?>
+ <Souris : désignation de la place>

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 selon les langages à balises utilisés avec D.O.M. (Document Object Model).

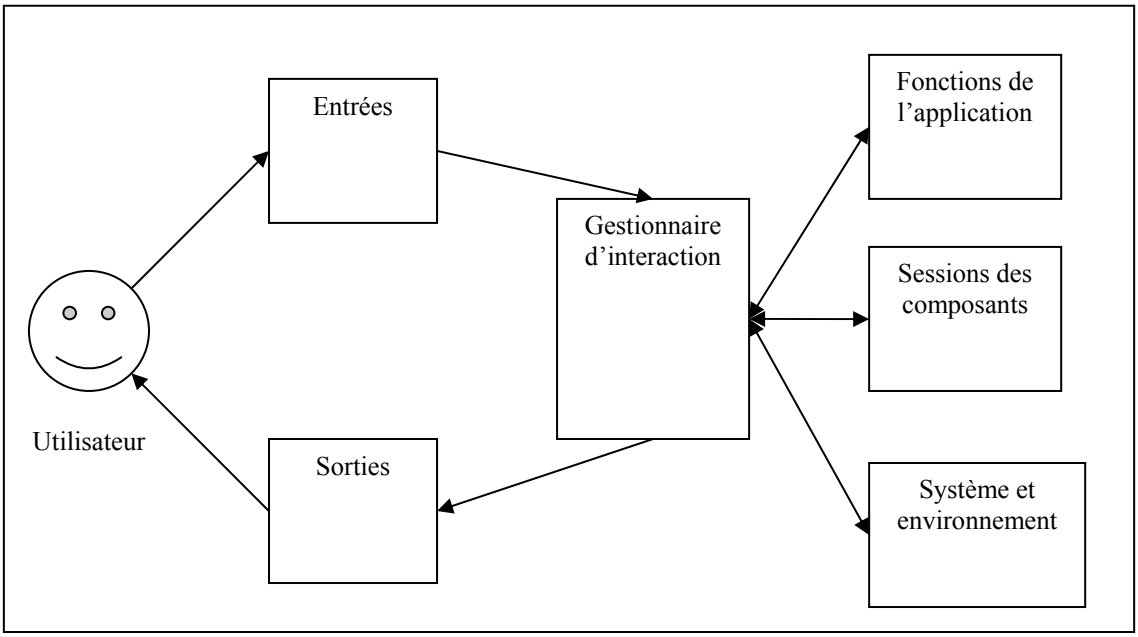


Figure 14 : Plate-forme d'intégration multimodale W3C

Le Tableau 1 illustre les propriétés de cette approche de fusion. Plusieurs choix (marqué « à définir ») sont laissés au concepteur comme la stratégie d'intégration, etc.

Type	Stratégie d'intégration	Stratégie d'exploration	Proximité temporelle	Complémentarité logique	Complétude de l'information	Incompatibilité des données
S,E,A,C	A définir	Profondeur	A définir	A définir	A définir	oui

Représentation des infos	Couverture du cycle de vie	Propriétés ergonomiques	Modèle utilisateur	Environnement	Généricité
multiple	C, S	non	non	non	oui

Légende Type : S = Synergique, E = Exclusive, A = Alternée, C = Concurrent

Cycle de vie : C = Conception, S = Spécification

Tableau 1 : Propriétés de la plate-forme W3C

Le modèle proposé est générique. Cependant, les composants d'intégration n'ont pas de spécifications précises et ils dépendent ainsi de l'application développée. Le choix de la stratégie d'intégration et les critères d'intégration (proximité temporelle, complémentarité logique, etc.) est laissé au développeur. Par contre les propriétés ergonomiques, le modèle de l'utilisateur et le modèle de l'environnement ne sont pas pris en compte. De plus, aucune technique d'implémentation n'est spécifiée.

5.2.2 Approche par creusets

Le moteur de fusion présenté ici a été exploité dans le cadre d'une application concrète nommée MATIS [Nigay 94]. L'application est d'abord présentée puis nous décrivons le fonctionnement du moteur de fusion sous-jacent.

a) Description de l'application MATIS

MATIS (Multimodal Airline Travel Information System) est un système d'information multimodal sur les transports aériens supportant 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 cela, l'utilisateur peut utiliser sa souris et spécifier sa requête par manipulation directe, il peut utiliser le clavier et spécifier sa requête en langue naturelle ou en remplissant un formulaire, il peut utiliser la parole et dicter sa requête ou il peut utiliser les trois modalités précédemment citées en les combinant. Par exemple, une requête peut être constituée comme suit :

<Parole : je voudrais connaître les vols de cette ville>
+ <Sélection avec la souris d'une ville>
+<Parole : arrivant à>
+<Saisie d'un horaire en utilisant le clavier>

b) Fonctionnement du moteur

Le moteur de fusion manipule des informations représentées dans un formalisme unique. Les données qui sont à intégrer sont représentées par des « creusets ». Les « creusets » sont des matrices correspondant aux champs des requêtes de l'application. La fusion de deux « creusets » est un creuset. De plus, trois types de fusion sont définis : la micro fusion (parallélisme), la macro fusion (proximité temporelle) et la fusion contextuelle (requête utilisant le contexte courant de l'interaction).

Le Tableau 2 rassemble les propriétés du moteur de fusion par creusets.

Type	Stratégie d'intégration	Stratégie d'exploration	Proximité temporelle	Complémentarité logique	Complétude de l'information	Incompatibilité des modalités
S,R	Précoce	Profondeur	oui	oui	oui	non

Représentation des infos	Couverture du cycle de vie	Propriétés ergonomiques	Modèle utilisateur	Environnement	Généricité
unique	C, S	non	non	non	oui

Légende Type : S = Synergique, C = Concurrent

Cycle de vie : C = Conception, S = Spécification

Tableau 2 : Propriétés du moteur de fusion par creusets

Ce moteur a l'avantage d'avoir été validé par une expérience pratique mais il ne prend pas en compte les propriétés ergonomiques, ni le modèle de l'utilisateur, ni le modèle de l'environnement. L'approche du moteur est générique et il peut être implémenté pour n'importe quelle application.

5.2.3 Moteur de fusion à base de règles

Le moteur de fusion présenté ici a été exploité dans le cadre d'une application concrète nommée LIMSI-DRAW [Bellik 92] [IHM 92]. Comme dans le paragraphe précédent, nous décrivons l'application avant détailler l'approche de fusion.

a) Description de l'application LIMSI-DRAW

LIMSI-DRAW est une application graphique qui permet de créer et de manipuler des formes géométriques élémentaires à travers un ensemble de commandes multimodales. Trois périphériques 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 avec la souris, avec l'écran tactile, en combinant souris et écran tactile, en combinant parole et souris, en combinant parole et écran tactile ou en combinant les trois modalités. Par exemple, une commande de création d'un rectangle peut être spécifiée comme suit :

<Parole : Rectangle>
+ <Clic souris : coin supérieur gauche>
+ <Contact écran tactile : coin inférieur droit>

b) Fonctionnement du moteur

Au sein du moteur de fusion, les informations à intégrer font référence à l'une des trois opérations suivantes : référence à une commande à exécuter, référence à un argument, et référence à une entrée de donnée. L'intégration se fait soit par fusion locale (une donnée est associée avec un argument), soit par fusion globale (production d'énoncés en combinant les données dans une structure commune). La fusion locale a lieu entre deux informations qui ont une complémentarité logique ou qui ont des types compatibles ou qui sont proches temporellement. La fusion globale

est établie si la structure commune contient une seule information qui fait référence à une commande. Dans ce cas, si une information de type « entrée de donnée » ne fait pas l'objet d'une fusion locale et si les types des informations sont compatibles alors les deux informations sont associées.

Le Tableau 3 synthétise les propriétés du moteur.

Type	Stratégie d'intégration	Stratégie d'exploration	Proximité temporelle	Complémentarité logique	Complétude de l'information	Incompatibilité des données
S,E,A	Différée	Largeur	oui	oui	oui	oui

Représentation des infos	Couverture du cycle de vie	Propriétés ergonomiques	Modèle utilisateur	Environnement	Généricité
multiple	C, S	non	non	non	oui

Légende Type : S = Synergique, E = Exclusive, A = Alternée

Cycle de vie : C = Conception, S = Spécification

Tableau 3 : Propriétés du moteur de fusion à base de règles

Comme pour l'approche par creusets, le moteur de fusion à base de règles est validé par une expérience pratique. Cependant, il ne prend pas en compte les propriétés ergonomiques, le modèle utilisateur et le modèle de l'environnement. Son implémentation dépend de l'application mais les représentations des données circulant au sein du moteur sont génériques. Par contre, rien n'est spécifié pour son implémentation.

5.2.4 Moteur de fusion d'événements

Comme pour les moteurs de fusion précédents, celui-ci a été exploité dans le cadre d'une application permettant la déclaration d'énoncés multimodaux dans un modeleur 3D [IHM 92] [Gaildrat 83]. Son fonctionnement est expliqué à la suite de la description de l'application.

a) Description de l'application

Cette application permet de décrire une scène en trois dimensions. L'utilisateur peut décrire la scène (placement des objets, descriptions des propriétés de textures, couleur, etc.) avec des commandes purement orales ou écrites ou multimodales combinant parole et geste de désignation. Par exemple, pour déplacer une bouteille sur une table, l'énoncé peut être du type :

<Parole : Mets la bouteille ici>
+ <Souris : désignation de la position souhaitée sur la table>

b) Fonctionnement du moteur

Pour interpréter une commande complète, trois unités de base sont utilisées : les actions, les arguments nécessaires à l'exécution de l'action et les

relations entre objets (par exemple, -posé-sur-, -collé-contre-,…). Dans l'exemple ci-dessus, l'action correspond à mettre, les arguments sont la bouteille et le lieu indiqué par la souris, la relation correspond à celle qui existe entre la bouteille et le lieu. La fusion repose sur des contraintes temporelles, des connaissances pragmatiques, des connaissances sur l'utilisation des médias et des connaissances sur l'état du dialogue.

Le Tableau 4 rassemble les propriétés de ce moteur de fusion.

Type	Stratégie d'intégration	Stratégie d'exploration	Proximité temporelle	Complémentarité logique	Complétude de l'information	Incompatibilité des données
S,E,A	Différée	-	oui	oui	oui	oui

Représentation des infos	Couverture du cycle de vie	Propriétés ergonomiques	Modèle utilisateur	Environnement	Généricité
multiple	C, S	Non	non	Non	non

Légende Type : S = Synergique, E = Exclusive, A = Alternée

Cycle de vie : C = Conception, S = Spécification

Tableau 4 : Propriétés du moteur de fusion d'événements

Ce moteur de fusion n'est pas générique car il est très dépendant de l'application (nécessité de connaître des informations sur les actions possibles et sur l'utilisation des modalités). La stratégie d'exploration n'est pas précisée. Comme pour les autres modèles, les propriétés ergonomiques, le modèle utilisateur, le modèle de l'environnement ne sont pas pris en compte. Enfin, le modèle ne couvre pas la phase d'implémentation.

5.2.5 Dialogue multimodal

Le moteur présenté dans [Bourguet 92] a aussi été exploité dans le cadre d'une application [IHM 92]. Son fonctionnement est expliqué à la suite de la présentation de l'application de dessin multimodale.

a) Description de l'application de dessin multimodale

Destinée à la réalisation de plans architecturaux, cette application permet de créer, d'assembler, de modifier et de positionner des objets à l'aide de la parole et de gestes de désignation. Les deux modalités peuvent être utilisées seules ou peuvent être combinées. Par exemple, pour dessiner une porte sur un mur déjà existant, l'énoncé peut être :

<Parole : dessine une porte sur ce mur>
+ <Geste : désignation du mur>

b) Fonctionnement du moteur

Ce moteur est décomposé en trois modules correspondant à des niveaux d'abstraction différents : le gestionnaire des événements (bas niveau), l'analyseur multimodal (niveau intermédiaire) et le contrôleur de dialogue (haut niveau). La fusion des modalités est réalisée dans le modèle de la tâche à un haut niveau d'abstraction. Il peut aussi être fait à bas niveau dans le gestionnaire des événements lorsque ceux-ci sont proches temporellement. Les informations prennent trois formes différentes pour chaque niveau d'abstraction.

Le Tableau 5 rassemble les propriétés de ce moteur de fusion.

Type	Stratégie d'intégration	Stratégie d'exploration	Proximité temporelle	Complémentarité logique	Complétude de l'information	Incompatibilité des données
S,A,E	Précoce	Largeur	oui	oui	non	Oui

Représentation des infos	Couverture du cycle de vie	Propriétés ergonomiques	Modèle utilisateur	Environnement	Généricité
multiple	C, S	non	non	non	non

Légende Type : S = Synergique, E = Exclusive, A = Alternée

Cycle de vie : C = Conception, S = Spécification

Tableau 5 : Propriétés du moteur de dialogue multimodal

Ce moteur n'est pas générique et seules les modalités de la parole et de la désignation sont étudiées. Par contre, il offre les outils nécessaires à la mise en œuvre d'un système de communication multimodal. Comme pour les autres approches, les propriétés ergonomiques, le modèle utilisateur et le modèle de l'environnement ne sont pas intégrés au moteur.

Les approches de fusions présentées dans ce paragraphe soulignent une grande diversité. Tous proposent des solutions d'intégration différentes. Le modèle du W3C permet d'avoir une approche générique, un de nos objectifs. L'approche par creusets et le moteur de dialogue multimodal identifie clairement les différents niveaux d'abstraction et les différentes représentations de l'information. Néanmoins, aucun de ces modèles ne prend en compte les propriétés ergonomiques, le modèle utilisateur et le modèle de l'environnement. De plus, aucun ne spécifie une technique d'implémentation.

Dans le paragraphe suivant, nous présentons les technologies à composants qui semblent devenir un standard de programmation et qui constitue une approche pour traiter les aspects d'implémentation absents de toutes les approches étudiées.

6 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 94], les industriels du logiciel ont, fait émerger une nouvelle approche de développement logiciel : le développement d'application 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 toutes les plus grandes entreprises 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 nouvelle 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. D'après une analyse du groupe Gartner [Phipps 00] : « En 2002, 70% des nouvelles applications, qui seront déployées, seront des applications construites à bases de composants (80% de probabilités) »[1]. Depuis, le monde informatique (académique et industriel) est en ébullition face à ce concept de composant et chacun essaye d'imposer son modèle. Ainsi, il n'existe pas à ce jour de consensus sur la programmation basée sur les composants : différentes définitions sont proposées selon le domaine dans lequel on se trouve et le métier que l'on exerce.

Dans cette partie, une revue terminologique sur les technologies à composants est présentée. Puis nous décrivons les différentes technologies employées. Enfin, une synthèse présente les avantages et les inconvénients de l'utilisation d'une telle technologie.

6.1 Terminologie

Nous considérons un composant comme : « une unité réutilisable de composition et de déploiement accessible à travers des interfaces » [Szyperski 98]. Mais d'autres concepts sont mis en jeu et plusieurs termes sont employés lorsque l'on parle de composants. On retrouve dans [Bass 00], 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,

[1] "*By the year 2002, 70 percent of all new applications will be deployed using component-based application building blocks (0.8 probability)" [Phipps 00]*

être vendu comme un produit commercial. Un composant logiciel est généralement implémenté par et pour une technologie particulière »[2].

- « 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 »[3].

- « 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 »[4].

- « 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 »[5].

On remarque au travers de la définition de ces termes et notamment de « technologie à composants » 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 15, différents acteurs (concepteur, programmeur, assembleur, etc.) interviennent pour la mise en place d'une application. Chacun est en relation, mais personne n'a 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 fournissent ainsi 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.

[2] *"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 00]*

[3] *"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 00]*

[4] *"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 00]*

[5] "*Component-based software engineering (CBSE) involves the practices needed to perform CBD in a repeatable way to build systems that have predictable properties." [Bass 00]*

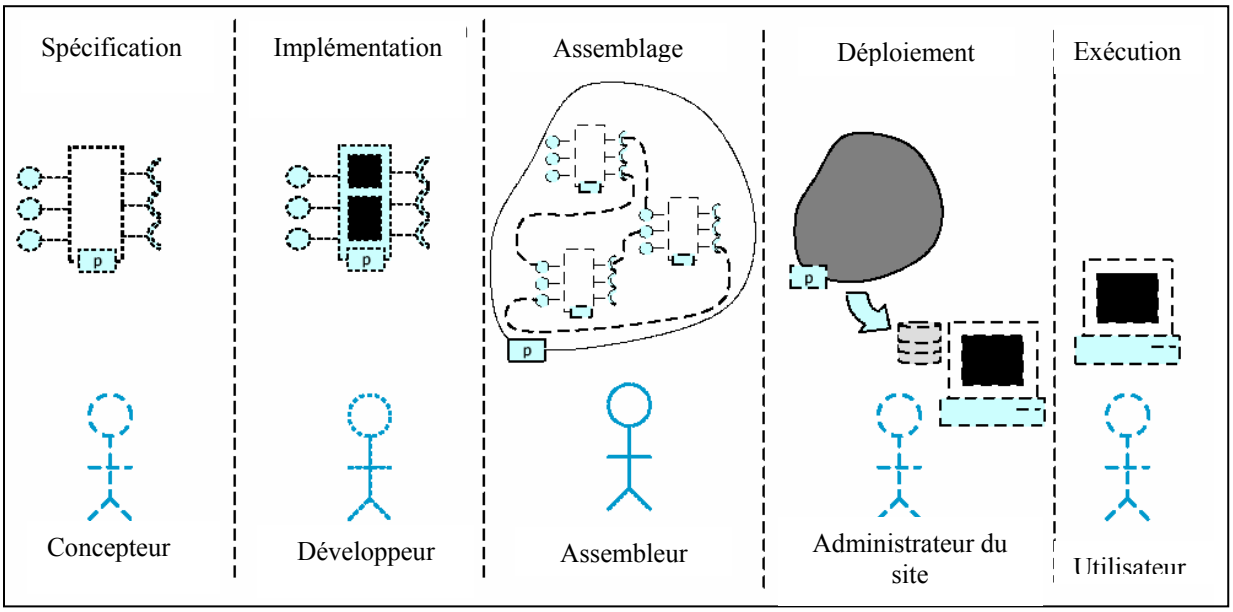


Figure 15 : Prise en compte du processus de développement [Favre 03]

La terminologie étant figée, nous étudions maintenant les technologies à composants existantes.

6.2 Technologies à composants

Cette partie présente les technologies qui sont le plus utilisées dans le développement d'applications. Toutes sont distribuées par des organismes tels que Microsoft, Sun ou l'OMG (Object Management Group). Chacune de ces technologies définit son propre modèle et est adaptée à un besoin particulier. Cependant, un modèle général a vu le jour au sein de l'OMG et fait l'objet du premier paragraphe. Ensuite, une revue des différentes technologies en fonction du domaine d'application est présentée.

6.2.1 Modèle de 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 02] utilisée pour les CCM (Corba Component Model). 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 »[6]. Pour interagir avec un composant, quatre ports ont été identifiés : les facettes, les réceptacles, les sources d'événements et les puits d'événements (*event sinks*). Les facettes sont les interfaces fournies par le composant. Les réceptacles sont des points de connexions qui décrivent la capacité d'un composant à utiliser une référence fournie par un autre

[6] "*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 02]*

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 les 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 16 illustre la vue externe d'un composant.

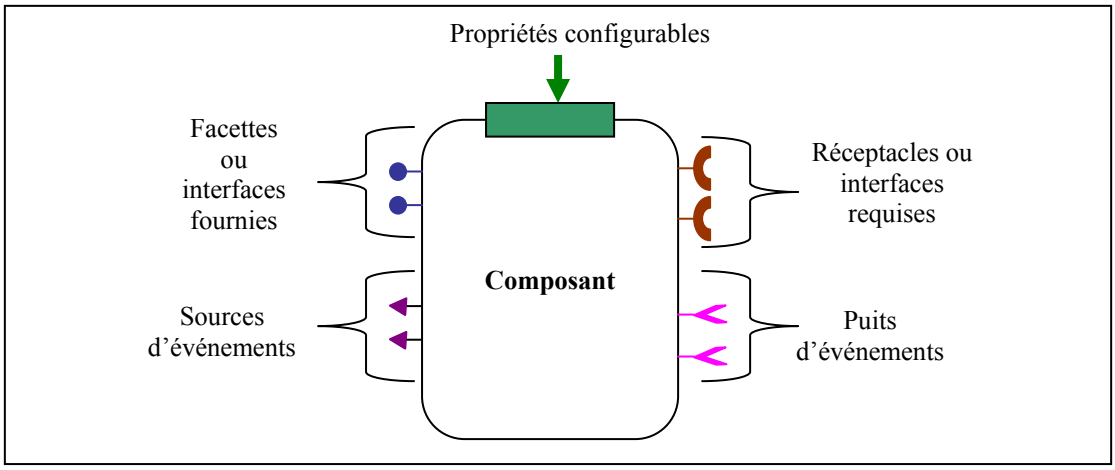


Figure 16 : Interfaces d'un composant

Pour l'assemblage, le concept de connecteur est abordé. Comme le montre la Figure 17, les composants sont assemblés à l'aide de connecteurs. Les connecteurs peuvent être simples (pipe, appel de procédure, etc.) ou plus complexes (protocole client-serveur, lien avec une base de données, etc.) [Barbier 02].

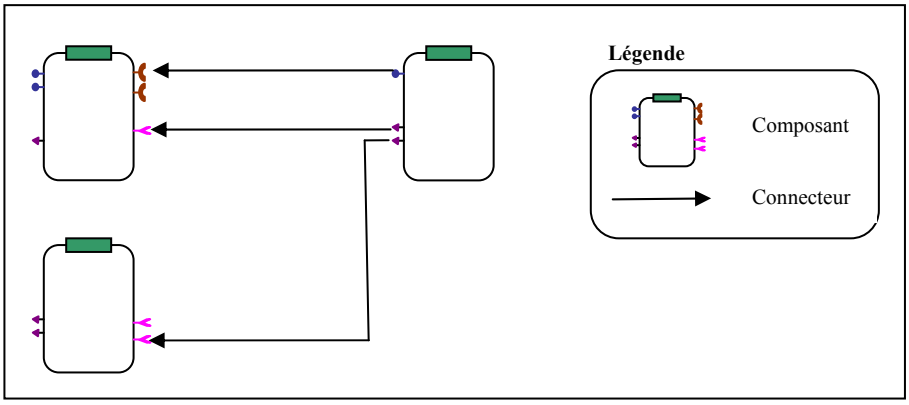


Figure 17 : Assemblage de composants

Les composants doivent être ouverts et adaptables. Les composants doivent être adaptables afin de permettre leur paramétrage au moment de l'assemblage. L'assemblage doit pouvoir être structurel mais aussi fonctionnel. De plus, il peut être utile que les composants soient capables, durant leur exécution de s'adapter dynamiquement aux modifications de leur environnement (fonctionnement d'un composant en « *mode dégradé* » par exemple).

6.2.2 Différents domaines d'application

La diversité des technologies à composants réside dans le domaine d'application visé. Ce paragraphe décrit brièvement les technologies qui sont les plus utilisées (COM/DCOM/COM+/JavaBeans/EJB/CCM/OSGI) tout en exposant leur contexte d'utilisation.

Les composants Microsoft COM, DCOM et COM+ [Microsoft 02] sont utilisés dans les domaines de Conception Assistée par Ordinateur (CAO) ou Computer-Aided Design (CAD) et de Fabrication Assistée par Ordinateur (FAO) ou Computer-Aided Manufacturing (CAM). Ces composants ne peuvent être utilisés que 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 et assurent l'interopérabilité entre les langages de programmation. 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 03] 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.

Chez Sun Microsystems, on retrouve deux technologies à composants : les JavaBeans et Les Entreprise JavaBeans (EJB). Les JavaBeans [Beans 97] 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 »[7] [Beans 97]. En ce qui concerne les EJB [EJB 02], 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.

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

L'OMG propose elle aussi sa technologie à composants avec CCM [Corba 02]. Le modèle, décrit au paragraphe 6.2.1, est assez général et pourrait s'appliquer à l'ensemble des technologies présentées. Les composants CCM sont, comme les EJB et les composants .Net, destinés aux applications distribuées et notamment celles d'Internet.

La plate-forme OSGi (Open Services Gateway initiative) [OSGi 03] 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 comme les JavaBeans, l'assemblage des composants est dynamique, à l'exécution.

6.3 Synthèse

Les technologies à composants sont multiples mais il convient de noter que toutes ces technologies sont écrites avec un langage orienté objet (Java, C++, etc.). A la différence de la programmation orientée objet, la programmation orientée composant définit les concepts de composition (assemblage) et de déploiement. De plus, l'héritage qui est le mécanisme principal de la programmation orientée objet viole dans certains cas le principe d'encapsulation et ne permet pas à l'objet d'être réutilisable ; ce n'est pas le cas dans une programmation orientée composants.

Pour conclure sur les technologies à composants, nous listons leurs principaux intérêts et inconvénients.

6.3.1 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 applications 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.
- Au travers, des technologies à composants, 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.

6.3.2 Inconvénients

- La nécessité d'une infrastructure spécifique (technologie à composants permettant de fournir des services techniques, de déployer et d'assembler les composants) est un désavantage pour l'utilisation de composants hétérogènes.

- L'absence de standard multiplie le nombre de technologies à composants alors que les concepts généraux sont semblables. En fait, 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 orientée composants est un handicap sérieux par rapport aux autres techniques de programmation. Cependant, des modèles conceptuels sont en cours d'établissement.

7 Conclusion

Dans ce chapitre, nous avons établi le cadre de nos travaux en introduisant les concepts utiles à notre étude. Nous retenons de ce chapitre la définition d'une modalité comme un couple (dispositif, langage d'interaction) mis en relation avec des modèles de psychologie cognitive. Des taxinomies présentées, nous extrayons un ensemble de caractéristiques d'une modalité mais aussi de l'usage de plusieurs modalités. Pour caractériser l'utilisabilité de la multimodalité, nous avons introduit une liste de propriétés d'ergonomie et pour réaliser une interaction multimodale, un ensemble de mécanismes génériques (moteurs de fusion) et des technologies à composants.

Dans le chapitre suivant, nous exposons notre espace solution : il s'agit d'une solution à la conception logicielle et au développement d'interfaces multimodales.

II. ESPACE SOLUTION : ICARE, MODELE A COMPOSANTS POUR L'INTERACTION MULTIMODALE

Comme exposé au chapitre précédent, l'approche à composants présente de nombreux atouts. L'étude et l'utilisation d'une technologie à composants pour le moteur de fusion de modalités permet de définir un modèle générique pour les modalités et les règles de composition afin d'obtenir des systèmes interactifs véritablement utiles et utilisables. L'idée directrice des recherches réside dans l'adaptation directe des travaux et des concepts sur la multimodalité et sur l'ergonomie des systèmes multimodaux dans un modèle à composants.

Notre espace solution présente une plate-forme de conception et de développement des systèmes interactifs multimodaux. Cette plate-forme, que nous avons intitulée ICARE (Interaction Complémentarité Assignation Redondance Equivalence), propose un ensemble de composants permettant de concevoir et développer une interaction multimodale en entrée comme en sortie. Cette partie décrit notre solution avec dans un premier temps, notre modèle à composants, et dans un second temps, l'ensemble des composants qui interviennent de notre plate-forme de développement. Nous concluons ce chapitre par une synthèse de nos contributions.

1 Modèle à composants choisi

Dans notre approche, nous reprenons la définition donnée par l'OMG qui semble recouvrir l'ensemble des concepts de la plupart des technologies. Ainsi, le composant 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 le montre la Figure 16, 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 propriétés). Les composants communiquent entre eux par des événements (communication événement – réaction).

D'autre part, l'aspect permettant de manipuler visuellement les JavaBeans constitue une approche véritablement intéressante car le travail de l'assembleur de composants est véritablement simplifié. L'outil de manipulation graphique permet d'améliorer le temps de production d'une application. L'assembleur peut appliquer l'architecture d'implémentation des composants logiciels conçue lors des phases amont et positionne ses composants en glisser-déposer (drag & drop).

Notre solution ne considère pas une technologie particulière mais doit s'appliquer à l'ensemble des technologies. Le modèle idéal serait défini par l'union des technologies existantes. Dans nos travaux, nous considérons seulement les caractéristiques communes des technologies à composants, définissant un composant comme une unité de composition et de déploiement.

2 Plate-forme ICARE

ICARE (Interaction Complémentarité Assignation Redondance Equivalence) est une plate-forme de conception et de développement qui fournit plusieurs types de composants correspondant aux concepts identifiés dans l'interaction multimodale. ICARE propose aussi une méthodologie de développement qui couvre les phases de conception et de codage. Pour la phase de codage, on considère que l'implémentation des composants peut être effectuée avec n'importe quelle technologie qui répond au modèle présenté au paragraphe 1 de l'espace solution.

Les composants ICARE sont de trois types. Nous retrouvons les composants élémentaires (ou de base) qui correspondent aux concepts élémentaires des systèmes interactifs multimodaux. Ensuite, nous avons les composants de composition qui représentent les unités de fusion et qui constituent le cœur de notre moteur de fusion. Enfin, nous retrouvons les composants d'assignation qui permettent de faire le lien avec les tâches de n'importe quelle application.

Cette partie décrit les différents types de composants que l'on retrouve dans ICARE et définit les règles relatives à la fusion.

2.1 Composants élémentaires

Les composants élémentaires d'ICARE représentent les concepts utiles à la conception et au développement du système interactif. Quatre types de composants élémentaires sont identifiés : les composants Dispositifs, les composants Systèmes représentationnels, les composants Utilisateurs et les composants Environnements.

Pour les modalités, nos travaux reprennent la définition du paragraphe 1.1 : une modalité est une association d'un ou plusieurs dispositifs avec un ou plusieurs systèmes représentationnels (langages d'interaction dans [Nigay 94]). Cette définition est intéressante car les deux niveaux d'abstraction, physique et logique, sont considérés. En adéquation avec cette définition d'une modalité, les composants élémentaires de cette approche sont les composants Dispositifs et Systèmes Représentationnels. Leur multiplicité dans une application permet d'obtenir des systèmes interactifs multimodaux en entrée et en sortie.

Le composant Utilisateur représente les caractéristiques de l'utilisateur. Elles correspondent à différents types d'informations statiques et dynamiques. Plusieurs composants Utilisateurs peuvent être définis dans le cas d'un collecticiel.

Le composant Environnement contient les caractéristiques de l'environnement pertinentes pour l'interaction et l'utilisation de l'application. Beaucoup de travaux essayent de modéliser l'environnement. Nous ne prétendons pas fournir un modèle générique de l'environnement mais nous rassemblons seulement les différentes caractéristiques identifiées.

Nous présentons en détail ces quatre types de composants. Pour chaque type, nous proposons une définition, un symbole et leurs propriétés. Ainsi, nous définissons un modèle de composant pour l'interaction multimodale avec différents types de composant. Enfin, nous expliquons comment chacun peut être intégré dans l'interaction.

2.1.1 Composant Dispositif

a) Définition

Un composant Dispositif représente la réalité physique d'entrée ou de sortie du système. Les composants Dispositifs peuvent être les pilotes eux-mêmes (drivers) 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, dans les applications classiques, les dispositifs d'entrée sont le clavier et la souris et les dispositifs de sortie sont l'écran et les haut-parleurs.

b) Symbole

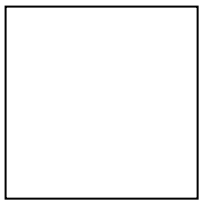


Figure 18 : Symbole du composant Dispositif

c) Propriétés

Le Tableau 6 rassemble l'ensemble des propriétés des composants Dispositifs.

Notre vision du problème qui est dirigé génie logiciel nous impose de caractériser les composants afin de pouvoir les maintenir et suivre leur évolution. Ainsi, nous identifions les composants par un nom, un numéro de version et le nom de son créateur.

De plus et comme nous avons pu le remarquer dans les espaces présentés au paragraphe 3 de l'espace problème, la distinction de l'interaction en entrée et en sortie est un paramètre important qui se traduit par une propriété du composant.

Deux propriétés caractérisent la fréquence et la préférence que l'utilisateur accorde au dispositif. Elles permettent de faire le choix entre plusieurs modalités. Soit le choix est laissé à l'utilisateur (propriété ergonomique d'adaptabilité) et la propriété « préférence d'utilisation » de chaque modalité est analysée par le système, soit le choix de la modalité est laissé au système (propriété ergonomique d'adaptativité) et c'est la propriété « fréquence d'utilisation » de chaque modalité qui est prise en compte par le système.

Spécifiquement aux dispositifs, d'autres propriétés sont prises en compte :

- [Nigay 94] identifie le niveau de conscience de l'utilisateur sur l'utilisation du dispositif et nous l'avons intégré dans notre composant. Quand le dispositif demande une commande explicite de l'utilisateur, il est qualifié d' « actif » comme par exemple l'utilisation d'une souris. Au contraire, un capteur d'orientation de la tête d'un utilisateur est un dispositif « passif ».

- L'espace de Buxton [Buxton 83] caractérise le nombre de degrés de liberté des dispositifs.

- L'espace de Frohlich [Frohlich 91] met en évidence le rôle du sens humain dans l'interaction. Cette propriété est pertinente pour les dispositifs de sortie car il s'agit des sens qui nous permettent de percevoir les phénomènes du monde. Dans ce cas, les cinq sens humains qui sont la « vision », l' « audition », le « toucher », le « goût » et l' « odorat » permettent de caractériser le dispositif.

- [Dubois 01] caractérise le lieu d'interaction d'un dispositif. Cela correspond au lieu géographique où l'utilisateur doit porter son attention pour pouvoir émettre ou recevoir des données.

Les autres propriétés permettent au système de proposer un service efficace pour la gestion du dispositif. Il y en a plusieurs : l'état du dispositif, le facteur de confiance des données générées, le temps de traitement, le temps maximum de traitement, sa capacité à être utilisé dans des conditions anormales et l'historique de chacune des propriétés.

Propriété	Description
nom	Le nom d'un composant Dispositif est constitué d'une chaîne de caractères commençant par « D_ » et il doit être représentatif du dispositif qu'il représente (par exemple pour la souris, le nom peut être « D_souris »).
version du composant	La version du composant est une chaîne de caractères commençant par « V » et suivi d'un numéro (par exemple, V1.0 représente une première version stable du composant).
nom du créateur	Le nom du créateur du composant est une chaîne de caractères qui permet de retrouver l'auteur de ce composant (Par exemple : Bouchet).
Direction	Le sens du dispositif est caractérisé par cette propriété. La direction « entrée » (resp. « sortie ») illustre le flux de données de l'utilisateur vers le système (resp. du système vers l'utilisateur). La direction « entrée_sortie » concerne les dispositifs permettant des flux utilisateur->système et système->utilisateur.
conscience de l'utilisateur	Les dispositifs d'entrée peuvent être classés dans deux catégories (« actif », « passif ») en fonction de la conscience de l'utilisateur sur l'utilisation du dispositif. Cette propriété est « sans objet » pour les dispositifs de sortie.
nombre de degrés de liberté	Le nombre de degrés de liberté pour les outils. Par exemple, le joystick à trois degrés de liberté.
sens humain	Pour les dispositifs de sortie, les cinq sens humains : la « vision », l' « audition », le « toucher », le « goût » et l' « odorat » peuvent être impliqués. Pour les dispositifs d'entrée, le sens humain est qualifié de « sans objet ». En fait, cette propriété représente l'ensemble des sens humains qui permettent de percevoir l'information du dispositif. Par exemple, une tablette en braille est perçue avec le toucher ou avec la vision ; il y a donc deux sens humains attachés à ce dispositif.
lieu	En entrée, le lieu représente l'endroit où l'utilisateur doit utiliser le dispositif. Dans ce cas, nous parlons du lieu d'action. Parfois, ce lieu est libre et l'utilisateur peut utiliser le dispositif sans contrainte. En sortie, le lieu représente l'endroit où le dispositif est perceptible par l'utilisateur. Dans ce cas, nous parlons de lieu de perception.
facteur de confiance	Les données sont caractérisées par un facteur de confiance définissant la pertinence et la crédibilité de l'information générée par le dispositif. Le facteur de confiance est noté de 0 à 100, 0 représente le facteur le plus bas et 100 le facteur le plus élevé.
état	Cette propriété permet de connaître l'état courant de fonctionnement du dispositif. Il peut être en marche (valeur « marche »), à l'arrêt (valeur « arrêt »), en attente (valeur « pause ») ou en dysfonctionnement (valeur « erreur »).

Propriété	Description
temps de traitement	Le temps de traitement est une estampille qui représente le moment où le dispositif traite une donnée. Chaque donnée fournit par le dispositif est estampillée par un temps de traitement.
Temps maximum de traitement	Cette propriété représente le temps maximum que le composant doit mettre pour traiter et délivrer ses données.
Capacité mode dégradé	Le dispositif peut avoir la capacité d'être utilisé dans une situation dégradée et le mécanisme de traitement évolue dynamiquement en fonction de la situation. S'il est capable de fonctionner autrement que dans son fonctionnement nominal, il peut alors traiter l'information différemment selon des situations que nous qualifions de dégradées. Par exemple, le clavier du nouveau PowerBook G4 d'Apple peut être éclairé. En fonctionnement dégradé (ici, une baisse de la luminosité), les symboles des touches s'éclairent en fonction du niveau de luminosité.
historique de chaque propriété	Pour chacune des propriétés, un historique de leurs valeurs est sauvegardé.
fréquence d'utilisation	Cette propriété traduit la fréquence d'utilisation du dispositif. Sa valeur peut être « jamais », « souvent » ou « toujours ».
préférence d'utilisation	Cette propriété décrit les préférences de l'utilisateur concernant l'emploi de ce dispositif. Sa valeur peut être « faible », « normale » ou « élevée ».

Tableau 6 : Propriétés du composant Dispositif

Cette liste forme une base commune pour les composants Dispositifs. Cependant, elle ne prétend pas être exhaustive et le modèle peut évoluer et intégrer des propriétés supplémentaires que nous n'avons pas encore identifiées. Les dispositifs correspondent au niveau d'abstraction le plus bas et nous utilisons le terme d' « articulatoire » pour identifier ce niveau, en référence à la théorie de l'action présentée au chapitre précédent.

2.1.2 Composant Système Représentationnel

a) Définition

Un composant Système Représentationnel est le langage d'interaction défini dans [Nigay 94]. Les termes « Système représentationnel » sont préférés car il peut y avoir ambiguïté entre le langage d'interaction et son caractère linguistique. Par exemple, les angles en degré qui informent de la position de la tête d'un utilisateur constituent un système représentationnel sous la forme d'un triplet de trois valeurs comprise entre -180 et 180 degrés.

b) Symbole

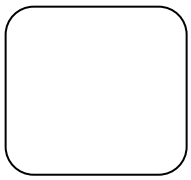


Figure 19 : Symbole du composant Système représentationnel

c) Propriétés

Le Tableau 7 rassemble les propriétés des composants Systèmes représentationnels.

Certaines sont communes avec celles du composant Dispositif comme le nom, la version, le nom de l'auteur, la direction, l'état, le temps de traitement, préférence d'utilisation etc. D'autres sont spécifiques aux systèmes représentationnels. L'étude de l'interaction en sortie de Bernsen [Bernsen 93] propose une classification des langages d'interaction. Les propriétés qu'il met en évidence (linguistique / analogique / arbitraire / dimension temporelle) et celles de [Vernier 01] (nombre de dimensions spatiales / complexité du langage) peuvent aussi s'appliquer aux systèmes représentationnels en entrée et constituent donc six propriétés propres aux composants Systèmes représentationnels d'ICARE.

Propriété	Description
nom	Le nom du composant commence par « SR_ » suivi d'une chaîne de caractères représentant le système représentationnel.
version du composant	Le numéro de version du composant (Exemple : V1.2).
nom du créateur	Le nom du créateur du composant.
direction	La valeur est « entrée », « sortie » ou « entrée_sortie ».
état	La valeur est « marche », « arrêt », « pause » ou « erreur ».
temps de traitement	Estampille qui représente le moment où le système représentationnel traite une donnée.
temps maximum de traitement	Cette propriété représente le temps maximum que le composant doit mettre pour traiter et délivrer ses données.
fréquence d'utilisation	La valeur est « jamais », « souvent » ou « toujours ».
préférence d'utilisation	La valeur est « faible », « normale » ou « élevée ».

Propriété	Description
capacité mode dégradé	Capacité du système représentationnel à fonctionner autrement (mode dégradé) que dans son fonctionnement nominal.
facteur de confiance	Pertinence et crédibilité des données générées par le système représentationnel. Noté de 0 à 100, 0 représente le facteur le plus bas et 100 le facteur le plus élevé.
historique de chaque propriété	Pour chacune des propriétés, un historique de leurs valeurs est sauvegardé.
nombre de dimensions spatiales	Le système représentationnel est représentable par une ou plusieurs dimensions spatiales. Par exemple, une graphique en 2 dimensions ou une scène en 3 dimensions.
dimension temporelle	Le caractère « statique » ou « dynamique » du système représentationnel repose sur la présence ou non d'une dimension temporelle dans la communication.
linguistique	Cette propriété définit si le système représentationnel a un caractère linguistique comme par exemple la langue française écrite (valeur « oui » ou « non »).
complexité du langage	Si le système représentationnel a un caractère linguistique, cette propriété définit sa complexité (valeur « facile », « normale » ou « difficile »). Par exemple, les langues naturelles ont comme complexité difficile alors que les langages pseudo-naturels ont une complexité normale. Sa valeur est « facile », « normale » ou « difficile ». La définition de cette propriété est laissée au développeur du système interactif. Cependant, une classification des systèmes représentationnels existants selon cette propriété pourra être fournie avec la plate-forme ICARE.
analogique	Cette propriété (valeur « oui » ou « non ») repose sur la notion de ressemblance avec la réalité. Le système représentationnel est analogique quand il repose sur une métaphore du monde réel. Par exemple, sur un plan de ville, une maison peut être représentée seulement par un rectangle (non analogique) ou par un rectangle (les murs) surmonter d'un triangle (le toit) qui est une métaphore du monde réel (analogique).
arbitraire	Cette propriété (valeur « oui » ou « non ») reflète la nature du système représentationnel choisie en fonction de ce que l'on souhaite représenter. Le système représentationnel est non arbitraire quand il n'y a pas de correspondance sociale ou cognitive entre le signifiant et le signifié. Un système représentationnel non-arbitraire consiste à utiliser, par exemple comme on le voit en France, le code de couleur rouge et bleu pour différencier le robinet d'eau chaude et d'eau froide. Dans une page web, l'utilisation des couleurs pour le texte des liens visités ou non encore visités repose sur un système représentationnel arbitraire.

Tableau 7 : Propriétés des composants Systèmes représentationnels

L'usage de systèmes représentationnels analogiques et non-arbitraires doit être favorisé dans la conception puisqu'ils garantissent que l'utilisateur est capable d'interpréter les informations.

Cette liste de propriétés forme une base commune pour les composants Systèmes représentationnels. Cependant, elle ne prétend pas être exhaustive et le

modèle peut évoluer et intégrer des propriétés supplémentaires. A ce niveau d'abstraction, nous parlons de niveau « syntaxique ».

2.1.3 Composant Utilisateur

a) Définition

Dans notre modèle, l'utilisateur correspond à un composant. Les propriétés de l'utilisateur sont multiples et peuvent être statiques (comme par exemple son sexe) ou dynamiques (comme son niveau de stress). Les propriétés de ce composant sont identifiées à partir du modèle utilisateur présenté dans [Coutaz 00] et des modèles présentés au chapitre précédent, paragraphe 2 [Rasmussen 86] [Card 83] [Norman 86]. Néanmoins, la modélisation de l'utilisateur constitue un vaste domaine de recherche en soi. Les propriétés que nous identifions ici constituent une esquisse du modèle de l'utilisateur et il convient d'affiner le composant.

b) Symbole

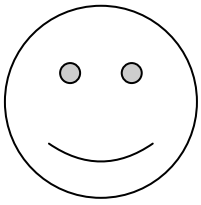


Figure 20 : Symbole du composant Utilisateur

c) Propriétés

Le Tableau 8 présente l'ensemble des propriétés du composant Utilisateur.

Comme pour les autres composants de notre plate-forme, nous devons connaître le nom du composant, sa version et le nom de son créateur.

De plus, toutes les informations générales sur une personne sont prises en compte. Nous identifions toutes les propriétés qui concernent l'identité (nom, prénom) et les coordonnées (adresse) de l'utilisateur. Nous le caractérisons aussi par ces données biométriques (taille, etc.) et sociales (déficiences sensitives, niveau de formation, etc.). Enfin, nous retrouvons les données dynamiques de l'utilisateur comme le niveau de stress ou le niveau de concentration.

Propriété	Description
nom du composant	Le nom du composant commence par « U_ » suivi d'une chaîne de caractères représentant l'utilisateur.
version du composant	Le numéro de version du composant (Exemple : V1.2).
nom du créateur	Le nom du créateur du composant.
nom	Le nom de l'utilisateur.
prénom	Le prénom de l'utilisateur.
adresse mail	L'adresse électronique de l'utilisateur.
adresse	L'adresse personnelle de l'utilisateur.
région	La région d'habitation de l'utilisateur.
pays	Le pays d'habitation de l'utilisateur.
nationalité	La nationalité de l'utilisateur.
taille	La taille en cm de l'utilisateur.
date de naissance	La date de naissance de l'utilisateur au format jj/mm/aaaa.
âge	L'âge de l'utilisateur.
sexe	Le sexe « masculin » ou « féminin » de l'utilisateur.
déficiences sensitives	Cette propriété représente l'ensemble des déficiences sensitives statiques de l'utilisateur. Cela peut être relatif au toucher, goût, odorat, vue et ouïe.
niveau de formation	Ce niveau définit le niveau de formation suivi par l'utilisateur.
niveau de compétence	Ce niveau définit le niveau de connaissances que l'utilisateur a de l'application. Cela correspond au modèle de [Rasmussen 86] où trois types d'utilisateur ont été identifié : « débutant », « occasionnel », « expert ». Il convient d'appliquer les trois niveaux au cas articulatoire et syntaxique, ainsi que sémantique, ce dernier niveau étant lié à l'application.
niveau de stress	Ce niveau représente le niveau de stress de l'utilisateur selon les valeurs « stressé », « moyennement stressé » et « pas stressé ».
niveau de concentration	Ce niveau représente le niveau de concentration de l'utilisateur selon les valeurs « concentré », « moyennement concentré » et « pas concentré ».
aptitudes des sens	Cette propriété fournit la capacité que l'utilisateur a pour se servir de chacun de ses sens. Pour chaque sens humain, l'aptitude est une évaluation (« ok » ou « utilisation impossible ») de l'utilisation du sens de perception.

Tableau 8 : Propriétés du composant Utilisateur

Comme nous l'avons annoncé dans la définition du composant Utilisateur, nous ne fournissons pas une liste exhaustive des propriétés d'un utilisateur. De plus, la tâche d'identification des caractéristiques dynamiques est difficile car beaucoup de données pourraient en faire partie comme par exemple la tension, le niveau de fatigue, etc.

2.1.4 Composant Environnement

a) Définition

La prise en compte de l'environnement dans une application devient de plus en plus importante. Avec la profusion des systèmes mobiles, de nouvelles formes d'interaction adaptatives à l'environnement sont observées : au cinéma par exemple, la sonnerie du téléphone laisse la place à une vibration. L'environnement fait partie du contexte d'utilisation au même titre que les caractéristiques de l'utilisateur. Ses caractéristiques influent sur l'utilisation et l'utilisabilité du système. Dans notre approche, l'environnement prend la forme d'un composant qui reprend certains points des travaux et des ontologies du contexte, le contexte étant ici réduit à l'environnement physique. Comme pour la modélisation de l'utilisateur, le domaine de recherche est vaste et il n'y a pas à ce jour de modélisation consensuelle. Plusieurs composants Environnements peuvent être définis au sein d'une même application : nous illustrons ce cas au paragraphe 2.1.5.

b) Symbole

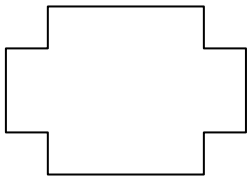


Figure 21 : Symbole du composant Environnement

c) Propriétés

Le Tableau 9 rassemble les propriétés du composant Environnement.

Nous retrouvons : les propriétés utiles au génie logiciel (nom du composant, version, nom de l'auteur) et les propriétés qui ont une influence sur l'interaction. Par exemple, quand le niveau de luminosité baisse, le clavier pourrait être retro-éclairé ; s'il y a un fond sonore élevé, le volume pourrait être augmenté ou arrêté.

L'environnement est aussi caractérisé par des informations qui décrivent plusieurs situations d'utilisation de l'application, classées selon leurs contraintes. Les situations définies sont propres, pour l'instant, à chaque application.

Propriété	Description
nom du composant	Le nom du composant commence par « E_ » suivi d'une chaîne de caractères représentant l'environnement.
version du composant	Le numéro de version du composant (Exemple : V1.2).
nom du créateur	Le nom du créateur du composant.
température extérieure	Cette propriété nous fournit la température en degré de l'environnement.
niveau de luminosité	Ce niveau en lumens nous informe sur la luminosité.
niveau sonore	Ce niveau nous informe sur l'intensité du fond sonore de l'environnement.
heure courante	Cette propriété nous donne l'heure courante en millisecondes depuis le 1er janvier 1970.
date courante	Cette propriété définit la date courante au format jj/mm/aaaa (j : jour ; m : mois ; a : année).
ensemble des situations	Cet ensemble définit une situation nominale et plusieurs situations dégradées (Ensemble s = [nominale, dégradée 1, dégradée 2,…]). Pour l'exemple d'une application sur un ordinateur de poche, la situation nominale est une utilisation de jour avec une température supérieur à 0 degré. Une situation dégradée de cette même application est une situation de nuit. Une dernière situation dégradée est un environnement dont la température est inférieure à 0 degré. Avec cette propriété, le composant Environnement communique la situation dans laquelle il se trouve aux composants Dispositifs et Systèmes représentationnels qui sont abonnés et si un traitement spécifique à la situation courante a été défini dans ces composants (capacité mode dégradé), ils s'adaptent et traitent d'une façon différente les données.

Tableau 9 : Propriétés du composant Environnement

2.1.5 Liens entre les composants élémentaires

Les composants élémentaires constituent la base de notre plate-forme. Ils peuvent être reliés pour échanger des informations.

a) Modalités simples

Une modalité est un couple <dispositif, système représentationnel>. Dans notre approche cela se traduit par l'existence de composants Dispositif et de composants Système représentationnel. Comme le montre la Figure 22 et la Figure 23, l'assemblage de ces deux types de composants permet de former des modalités simples appelées aussi modalités pures (à opposer aux modalités composées).

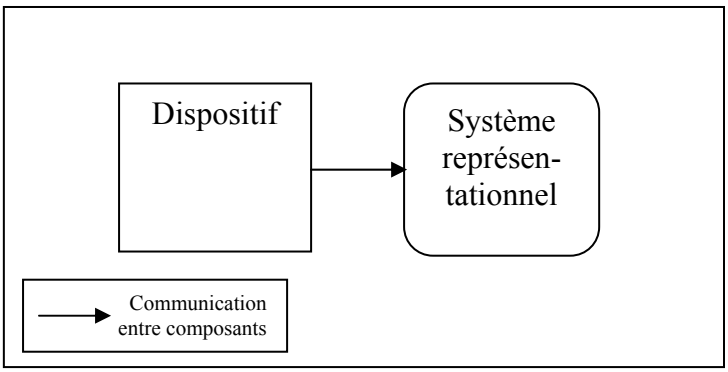


Figure 22 : Une modalité d'entrée simple

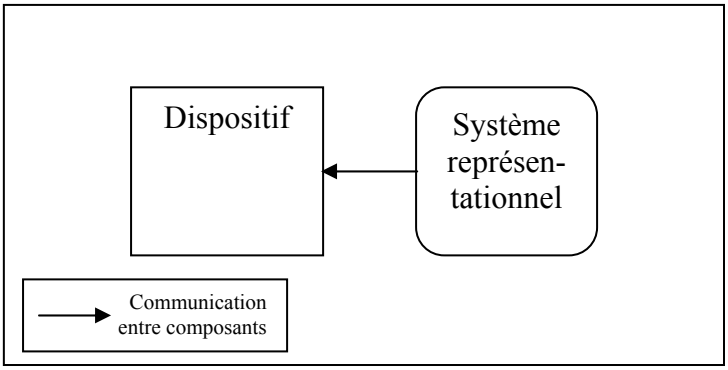


Figure 23 : Une modalité de sortie simple

En entrée (Figure 22), le composant dispositif fournit au composant système représentationnel une information d'interaction dite articulatoire. L'information articulatoire est transformée dans le composant Système représentationnel sous une forme interprétable par le système. Les données d'entrée sont alors représentées à un plus haut niveau d'abstraction. Ensuite, le composant Système représentationnel transmet au niveau d'abstraction supérieur (celui qui effectue le lien avec la tâche) une donnée qualifiée de syntaxique. En sortie (Figure 23), c'est le processus inverse ; le composant Système représentationnel reçoit une information syntaxique de l'application et la traite pour fournir au composant Dispositif une information articulatoire avec laquelle le composant pourra effectuer le rendu attendu sur le dispositif.

Des modalités plus complexes sont aussi envisageables. Elles résultent de la composition de modalités simples. Nous les appelons modalités composées. Plusieurs exemples sont détaillés dans le paragraphe 2.2.3. La modalité simple est un motif de conception important dans notre plate-forme et représente l'association la plus courante dans une application. La multiplicité de cette association au sein d'une architecture d'implémentation détermine le caractère multimodal en entrée ou en sortie de l'application construite.

De plus, l'intégration d'une partie des modèles de l'utilisateur et de l'environnement fournit de nouvelles données pour gérer efficacement les dispositifs physiques en tenant compte des capacités humaines et des conditions extérieures.

b) Intégration des utilisateurs

Une application est définie pour une ou plusieurs classes d'utilisateurs. Ses classes permettent de créer un système interactif adapté au type d'utilisateur identifié. Notre composant Utilisateur va plus loin en stockant les données propres à chaque individu. Les informations que fournit le composant Utilisateur permettent de s'assurer que les composants Dispositifs vont pouvoir être utilisés (manipulés) par l'utilisateur. Par exemple, si l'utilisateur est aveugle, les dispositifs, qui ont comme seul sens perceptible la vision, ne seront pas utilisés dans l'interaction. Les dispositifs associés à l'audition ou aux autres sens seront alors privilégiés.

Pour le cas des collecticiels, plusieurs composants Utilisateur peuvent être définis. Prenons le cas par exemple d'une application avec une souris et un clavier comme dispositifs d'entrée. L'application est prévue pour être utilisée par deux utilisateurs, l'un s'occupant de la souris pendant que l'autre manipule le clavier. Dans ce cas, deux composants Utilisateurs seront définis ; l'un est relié au composant Dispositif du clavier et l'autre a celui de la souris.

Enfin, les capacités d'un utilisateur varient en fonction de l'environnement extérieur. Par exemple, si le niveau sonore est très élevé (>100 db), sa capacité pour utiliser son sens auditif est réduite et l'utilisateur peut ne plus rien entendre. Ainsi, une communication est établie entre le composant Utilisateur et le composant Environnement pour ajuster certaines caractéristiques dynamiques de l'utilisateur.

c) Intégration des environnements

De nombreuses études en l'interaction Homme-Machine visent à définir et modéliser le contexte. Pour le développement des systèmes interactifs multimodaux, nous souhaitions que notre plate-forme permette d'intégrer l'environnement extérieur. C'est le rôle du composant Environnement. Les propriétés présentées au paragraphe 2.1.4 ne sont certes pas exhaustives mais peuvent être facilement étendues en fonction des besoins et des avancées des travaux sur le contexte.

Le composant Environnement communique avec les autres composants élémentaires. Comme décrit au paragraphe 2.1.4, il communique la situation courante (nominale, dégradée1, etc.) aux composants Dispositifs et Systèmes représentationnels qui sont abonnés. De plus et comme décrit au paragraphe précédent, il est en étroite relation avec le composant Utilisateur.

Plusieurs environnements peuvent être définis. Chacun d'entre eux peuvent représenter un sous-ensemble de l'environnement extérieur. Par exemple, deux composants Environnement peuvent être définis pour une application embarquée dans une voiture. L'environnement interne de l'habitacle et l'environnement externe à la voiture font l'objet de deux composants distincts. L'environnement interne communique principalement avec le composant Utilisateur alors que l'environnement externe permet d'identifier les situations normales ou de danger vis à vis de la

circulation. Il communique ces situations aux composants Dispositifs et Systèmes Représentationnels qui modifient leur manière de traiter l'information.

ICARE propose ainsi quatre types de composants élémentaires en adéquation avec les études du domaine. Chaque type de composant propose un ensemble de propriétés paramétrables permettant de développer des systèmes interactifs utilisables. Pour la complétude de notre modèle, il convient maintenant de se focaliser sur la composition de ces composants élémentaires. L'intégration (fusion) de plusieurs dispositifs pour un système représentationnel donné ou l'intégration de plusieurs systèmes représentationnels pour une tâche donnée est réalisée grâce à des composants génériques que nous présentons dans le paragraphe intitulé « composants de composition ».

2.2 Composants de composition

Notre objectif consiste à prendre en compte les propriétés ergonomiques dédiées à la multimodalité dans les composants de composition (ou de fusion). Ces propriétés que nous avons identifiées au paragraphe 4.1 de l'espace problème sont les quatre propriétés Complémentarité, Assignation, Redondance et Equivalence [Coutaz 94]. Les composants de composition sont le composants Complémentarité, Redondance et Equivalence. Ils traduisent les relations possibles entre plusieurs dispositifs pour un système représentationnel donné et entre plusieurs systèmes représentationnels pour une tâche donnée. L'assignation est aussi explicite dans notre modèle. Le composant Assignation n'est pas utilisé pour la composition mais pour faire le lien avec les applications. L'assignation fait l'objet du paragraphe 2.3 de l'espace solution.

Les composants de composition sont au nombre de deux. Nous retrouvons le composant Complémentarité qui représente la propriété complémentarité de CARE et le composant Redondance/Equivalence qui unifie au sein d'un seul composant les deux propriétés de CARE correspondantes.

Nous présentons en détail ces deux composants génériques. Pour chacun, nous proposons une définition, un symbole, leurs propriétés, des règles ergonomiques de composition et des métriques. Enfin, nous expliquons comment chacun peut être intégré dans le système interactif.

2.2.1 Composant Complémentarité

a) Définition

Le composant Complémentarité incarne la propriété CARE de complémentarité. Ce composant permet la composition de dispositifs ou de système représentationnels. Deux niveaux de complémentarité sont observés et correspondent à deux aspects de composition mis en évidence dans [Coutaz 94].

Le premier intervient au niveau physique et concerne la complémentarité dite articulatoire de plusieurs dispositifs pour un système représentationnel donné : n dispositifs sont complémentaires pour la construction d'une expression du système représentationnel pouvant être partitionnée en n sous-expressions où chaque sous-expression est obtenue à partir d'un seul des n dispositifs.

Le deuxième intervient au niveau logique et concerne la complémentarité dite syntaxique de plusieurs systèmes représentationnels pour une tâche donnée : n systèmes représentationnels sont complémentaires pour la construction d'une expression utile à une tâche pouvant être partitionnée en n sous-expressions où chaque expression est obtenue à partir d'un seul des n systèmes représentationnels.

b) Symbole

Pour faire le lien avec l'électronique, le composant complémentaire peut être interprété comme une porte ET. Ainsi, le symbole utilisé correspond à celui du ET en électronique.

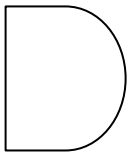


Figure 24 : Symbole du composant Complémentarité

c) Propriétés

Le Tableau 10 rassemble les propriétés du composant Complémentarité.

Comme pour les composants élémentaires, le nom du composant, sa version, le nom de son auteur, le sens de l'interaction, l'état du composant, les temps de traitement, les préférences d'utilisation, etc. sont pris en compte.

Le composant Complémentarité est connecté à plusieurs autres composants et les informations sur ces composants (leurs nombres, etc.) sont nécessaires pour fusionner leur données. De plus, deux propriétés (complétude et stratégie) ont été mises en évidence pour paramétrer la stratégie d'intégration.

Propriété	Description
nom du composant	Le nom du composant commence par « C_ » suivi d'une chaîne de caractères représentant la complémentarité.
version du composant	Le numéro de version du composant (Exemple : V1.2).
nom du créateur	Le nom du créateur du composant.
direction	Le sens de la complémentarité est caractérisé par cette propriété. La direction « entrée » (resp. « sortie ») illustre le flux de données de l'utilisateur vers le système (resp. du système vers l'utilisateur). La direction « entrée_sortie » concerne les dispositifs permettant des flux utilisateur->système et système->utilisateur.
état	Cette propriété permet de connaître l'état courant de fonctionnement du composant. Il peut être en marche (valeur « marche »), à l'arrêt (valeur « arrêt »), en attente (valeur « pause ») ou en dysfonctionnement (valeur « erreur »).
fréquence d'utilisation	Cette propriété traduit la fréquence d'utilisation du composant. Sa valeur peut être « jamais », « souvent » ou « toujours ».
préférence d'utilisation	Cette propriété décrit les préférences de l'utilisateur concernant l'emploi de ce composant. Sa valeur peut être « faible », « normale » ou « élevée ».
capacité mode dégradé	Capacité du composant à fonctionner autrement (mode dégradé) que dans son fonctionnement nominal.
facteur de confiance	Le facteur de confiance est noté de 0 à 100, 0 représente le facteur le plus bas et 100 le facteur le plus élevé. Le composant complémentaire calcule son facteur de confiance en fonction des facteurs de confiance véhiculé par les données qui lui sont transmises des composants complémentaires. Son facteur de confiance est égal à la moyenne des facteurs de confiance des composants complémentaires.
temps de traitement	Cette propriété correspond au moment où le composant Complémentarité fournit une donnée au composant suivant.
temps maximum de traitement	Cette propriété représente le temps maximum que le composant doit mettre pour traiter et délivrer ses données.
nombre de composants complémentaires	Cette propriété représente le nombre de composants qui sont connectés.
ensemble des composants complémentaires	Cet ensemble permet de stocker les composants qui sont reliés.
fenêtre temporelle	La fenêtre temporelle définit l'intervalle de temps pendant lequel les informations des composants complémentaires peuvent être combinées.
espérance de vie	L'espérance de vie représente le temps limite après lequel l'information, composée de n informations, n'est plus pertinente. Ainsi, l'information composite n'est pas diffusée et elle est détruite.

Propriété	Description
complétude	Comme nous l'avons vu au paragraphe 5 de l'espace problème, cette propriété (valeur « vrai » ou « faux ») définit l'état de la fusion des informations. La valeur est « vrai » si une expression complète a été obtenue en fusionnant les informations des composants complémentaires.
stratégie	Deux stratégies ont été sélectionnées concernant la fusion d'informations [IHM 92] : la stratégie précoce ou la stratégie différée. En étroite relation avec la propriété complétude, 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). En stratégie précoce, les informations complémentaires sont fusionnées en une seule information qui lorsqu'elle est complète (les n composants complémentaires ont transmis au moins une information dans une même fenêtre temporelle) est propagée. En stratégie différée, l'information à propager est constituée des informations complémentaires proches temporellement et qui ont le plus grand facteur de confiance et elle est diffusée seulement à la fin de la fenêtre temporelle.
historique des informations	Cet historique maintient les informations fournies par les composants qui sont complémentaires. Cela permet en stratégie différée de propager l'information provenant d'un des composants complémentaires qui a le plus haut facteur de confiance et non pas celle qui est arrivé en dernier par rapport à la fenêtre temporelle.
historique de chaque propriété	Pour chacune des propriétés, un historique de leurs valeurs est sauvegardé.

Tableau 10 : Propriétés du composant Complémentarité

d) Règles et métriques pour la complémentarité

Au niveau articulatoire (niveau physique contenant les dispositifs), les composants qui sont complémentaires suivent une règle :

> **Règle 1** : lors d'une complémentarité articulatoire (niveau physique), les lieux des dispositifs d'entrée doivent être proches pour que l'utilisateur puisse réaliser sa tâche (atteignabilité). Pour les dispositifs de sortie, la proximité permet d'assurer la perception des informations complémentaires et la propriété ergonomique d'observabilité est alors vérifiée.

D'autre part, une métrique est identifiée.

Métrique 1 : le facteur de confiance de l'information issue du composant Complémentarité est égal à la moyenne des facteurs de confiance des composants complémentaires. Par exemple, si le clavier et la souris sont des dispositifs complémentaires et si le clavier a 100 de facteur de confiance et la souris 50 alors le facteur de confiance du composant Complémentarité de ces deux dispositifs est de 75.

En résumé, le fonctionnement du composant Complémentarité contient les propriétés nécessaires pour caractériser la fusion d'informations complémentaires. Les autres propriétés de CARE sont aussi utilisées : les propriétés redondance et équivalence sont intégrées au sein d'un seul composant que nous présentons dans le paragraphe suivant.

2.2.2 Composant Redondance/Equivalence

a) Définition

Le composant Redondance/Equivalence incarne les propriétés CARE de redondance et d'équivalence. Comme pour le composant Complémentarité, ce composant permet la composition au niveau articulatoire (composition de dispositifs) et au niveau syntaxique (composition de systèmes représentationnels).

Au niveau articulatoire, le composant gère l'équivalence et/ou la redondance des informations des composants Dispositifs. Par exemple, dans une application où le curseur de souris peut être manipulé par deux dispositifs d'entrée (soit la souris, soit le pavé tactile), ces deux dispositifs sont considérés comme redondants et équivalent. S'il y a redondance, c'est ce composant de composition qui va choisir l'une des deux informations pour la propager. L'équivalence permet seulement de propager l'information même si l'un des deux dispositifs n'a pas été utilisé.

Au niveau syntaxique, le composant effectue le même processus que celui présenté précédemment mais les informations sont en provenance des composants Systèmes représentationnels.

b) Symbole

Pour faire le lien avec l'électronique, le composant redondance peut être interprété comme une porte OU. Ainsi, le symbole utilisé correspond à celui du OU en électronique.

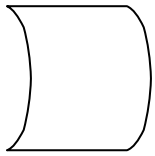


Figure 25 : Symbole du composant Redondance/Equivalence

c) Propriétés

Le Tableau 11 rassemble les propriétés du composant Redondance / Equivalence.

L'ensemble des propriétés du composant Complémentarité est retrouvé ici. Certaines sont traitées différemment comme le calcul du facteur de confiance ou la stratégie différée.

D'autre part, le paramètre « adaptation » permet de définir la stratégie de choix entre deux informations à facteurs de confiance égaux provenant deux modalités redondantes. Soit le composant choisit la modalité qui est le plus fréquemment utilisée et dans ce cas, la propriété ergonomique d'adaptativité est vérifiée ; soit il choisit la modalité la plus préférée de l'utilisateur et dans ce cas, c'est la propriété ergonomique d'adaptabilité qui est vérifiée.

Ce composant gère donc la redondance et l'équivalence. Cependant, il peut être paramétré pour couvrir qu'un seul des cas. Deux propriétés sont ainsi définies (« redondance » et « équivalence »).

Propriété	Description
nom du composant	Le nom du composant commence par « RE_ » suivi d'une chaîne de caractères représentant la redondance et/ou l'équivalence.
version du composant	Le numéro de version du composant (Exemple : V1.2).
nom du créateur	Le nom du créateur du composant.
redondance	Cette propriété détermine (valeur « vrai » ou « faux ») si la redondance des informations est prise en compte.
équivalence	Cette propriété détermine (valeur « vrai » ou « faux ») si l'équivalence des informations est prise en compte.
direction	Le sens de la redondance et/ou de l'équivalence est caractérisé par cette propriété. La direction « entrée » (resp. « sortie ») illustre le flux de données de l'utilisateur vers le système (resp. du système vers l'utilisateur). La direction « entrée_sortie » concerne les dispositifs permettant des flux utilisateur->système et système->utilisateur.
état	Cette propriété permet de connaître l'état courant de fonctionnement du composant. Il peut être en marche (valeur « marche »), à l'arrêt (valeur « arrêt »), en attente (valeur « pause ») ou en dysfonctionnement (valeur « erreur »).
nombre de composants redondants/équivalents	Cette propriété représente le nombre de composants qui sont connectés.
ensemble des composants redondants/équivalents	Cet ensemble permet de stocker les composants qui sont reliés.
fréquence d'utilisation	Cette propriété traduit la fréquence d'utilisation du composant. Sa valeur peut être « jamais », « souvent » ou « toujours ».
préférence d'utilisation	Cette propriété décrit les préférences de l'utilisateur concernant l'emploi de ce composant. Sa valeur peut être « faible », « normale » ou « élevée ».
capacité mode dégradé	Cette propriété représente la capacité du composant à fonctionner autrement (mode dégradé) que dans son fonctionnement nominal.
facteur de confiance	Le facteur de confiance est noté de 0 à 100, 0 représente le facteur le plus bas et 100 le facteur le plus élevé. Le composant Redondance/Equivalence calcule son facteur de confiance en fonction des facteurs de confiance véhiculé par les données transmises des composants qui lui sont connectés. Son facteur de confiance est égale au facteur de confiance du composant qui a fournit l'information qui va être traitée et propagée.
temps de traitement	Cette propriété correspond au moment où composant Redondance/Equivalence fournit une donnée au composant suivant.
temps maximum de traitement	cette propriété représente le temps maximum que le composant doit mettre pour traiter et délivrer ses données.

Propriété	Description
fenêtre temporelle	La fenêtre temporelle définit l'intervalle de temps pendant lequel les informations des composants connectés peuvent être prises en compte.
espérance de vie	L'espérance de vie représente la temps limite après lequel l'information traitée n'est plus pertinente. Ainsi, l'information n'est pas diffusée et elle est détruite.
complétude	Comme pour le composant Complémentarité, cette propriété (valeur « vrai » ou « faux ») définit l'état de l'information à propager. La valeur est « vrai » si une expression complète est obtenue.
stratégie	Nous retrouvons les deux stratégies possibles [IHM 92] : la stratégie précoce ou la stratégie différée. En étroite relation avec la propriété de complétude, 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). En stratégie précoce, la première information arrivée est propagée. En stratégie différée, l'information à propager est celle qui a le plus grand facteur de confiance ou celle déterminée par la propriété «adaptation ».
historique de chaque propriété	Pour chacune des propriétés, un historique de leurs valeurs est sauvegardé.
historique des informations	Cet historique maintient les informations fournies par les composants qui sont redondants et/ou équivalents. Cela permet en stratégie différée de propager l'information des composants qui a le plus haut facteur de confiance et non pas celle qui est arrivée en dernier par rapport à la fenêtre temporelle.
adaptation :	La valeur est une chaîne de caractères qui peut être « adaptabilité » ou « adaptativité ». L'adaptabilité signifie que le choix est laissé à l'utilisateur. Dans ce cas, les informations sont choisies d'après la propriété « préférence d'utilisation » des composants connectés. Par exemple si deux dispositifs sont redondants et que le premier a une préférence d'utilisation « faible » et le second a une préférence d'utilisation « forte », ce dernier détient l'information qui est prise en compte. L'adaptativité signifie que le choix est effectué par le système. La propriété « fréquence d'utilisation » est alors étudiée pour choisir de quel composant l'information est choisie. Le composant qui a une fréquence plus forte est favorisé.

Tableau 11 : Propriétés du composant Redondance/Equivalence

d) Règles et métriques pour le composant Redondance/Equivalence

Deux règles sont préconisées :

Règle 1 : Si n modalités sont redondantes et leur facteur de confiance est égal :

- Si l'adaptation est décidée par l'utilisateur (adaptabilité), la modalité choisie correspond à celle qui a la plus grande préférence d'utilisation.

- Si l'adaptation est décidée par le système (adaptativité), la modalité choisie correspond à la modalité la plus fréquemment utilisée.

Règle 2 : Si le composant fait partie de l'interaction de sortie alors les propriétés ergonomiques suivantes sont vérifiées : multiplicité de la représentation, observabilité et insistance.

Une seule métrique est définie :

Métrique 1 : Le facteur de confiance du composant Redondance/Equivalence est égal au facteur de confiance du composant utilisé pour la transmission de l'information.

Cette liste de règles et de métriques peut être étendue.

Les propriétés complémentarité, redondance et équivalence sont donc explicites, dans notre plate-forme, sous la forme de composants. Ces composants sont génériques. Ils peuvent recevoir des informations de n composants. Utilisables à deux niveaux d'abstraction que sont le niveau articulatoire et le niveau syntaxique, ils permettent de composer les dispositifs et les systèmes représentationnels.

2.2.3 Composition de composants élémentaires

Avec les deux composants de composition que nous avons définis, le composant Complémentarité et le composant Redondance/Equivalence, il est possible de réaliser des compositions pertinentes pour l'interaction multimodale et notamment construire des modalités dites composées.

a) Modalités composées

La composition est un élément du caractère multimodal d'une application et revêt différentes formes. En effet, la composition peut intervenir au niveau physique (dit articulatoire) et/ou au niveau logique (dit syntaxique).

Les Figure 26 et Figure 27 mettent en évidence la composition articulatoire qui correspond au niveau physique. Les dispositifs sont ici composés pour un système représentationnel donné. Par exemple, une souris est composée d'un dispositif permettant de connaître le déplacement de la souris suivant l'axe des abscisses et d'un autre dispositif permettant de connaître le déplacement de la souris suivant l'axe des ordonnées. En associant ces deux dispositifs, il est possible de connaître le déplacement de la souris dans un espace à deux dimensions et ainsi de positionner le curseur à l'écran. Les informations des deux dispositifs sont considérées ici comme complémentaires pour le système représentationnel de localisation en deux dimensions. La Figure 26 illustre cette complémentarité des deux dispositifs.

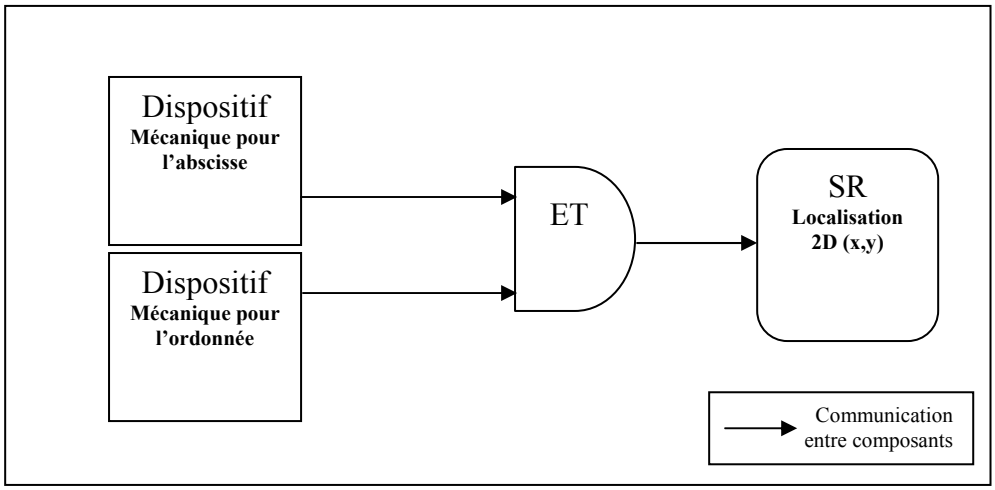


Figure 26 : Complémentarité articulatoire de deux dispositifs d'entrée

La Figure 27 illustre une redondance et une équivalence articulatoires entre deux dispositifs. Par rapport à l'exemple précédent, le niveau de granularité est plus élevé ; en effet, les dispositifs considérés sont une souris et un laser qui sont redondants et équivalents pour le système représentationnel de localisation en deux dimensions. Dans cet exemple, l'utilisateur a le choix d'utiliser l'un des deux dispositifs comme pointeur dans un espace en deux dimensions.

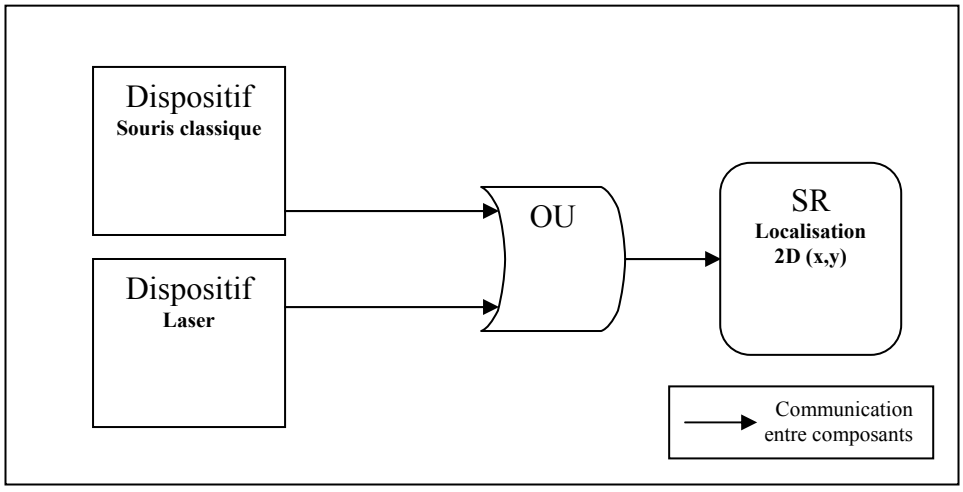


Figure 27 : Redondance et équivalence articulatoires de deux dispositifs d'entrée

Ces deux exemples traitent seulement des modalités d'entrée mais la composition peut aussi se faire pour les modalités de sortie. La communication se fait alors en sens inverse (les flèches sont inversées) comme à la Figure 23.

Au niveau logique, la composition est aussi possible. Qualifiée de syntaxique, cette composition est illustrée à la Figure 28 et à la Figure 30. Les systèmes représentationnels sont composés pour une tâche donnée. A la Figure 28, deux dispositifs équipant un utilisateur, un magnétomètre et un GPS, sont associés à un système représentationnel. Le magnétomètre est lié à un système de représentation donnant trois angles en radians (yaw, pitch, roll). Le GPS est lié à un système représentationnel de localisation en trois dimensions comprenant trois coordonnées (x, y, z). Ces deux systèmes représentationnels sont complémentaires syntaxiquement car leur composition va permettre au contrôleur de dialogue de connaître l'endroit où l'utilisateur regarde.

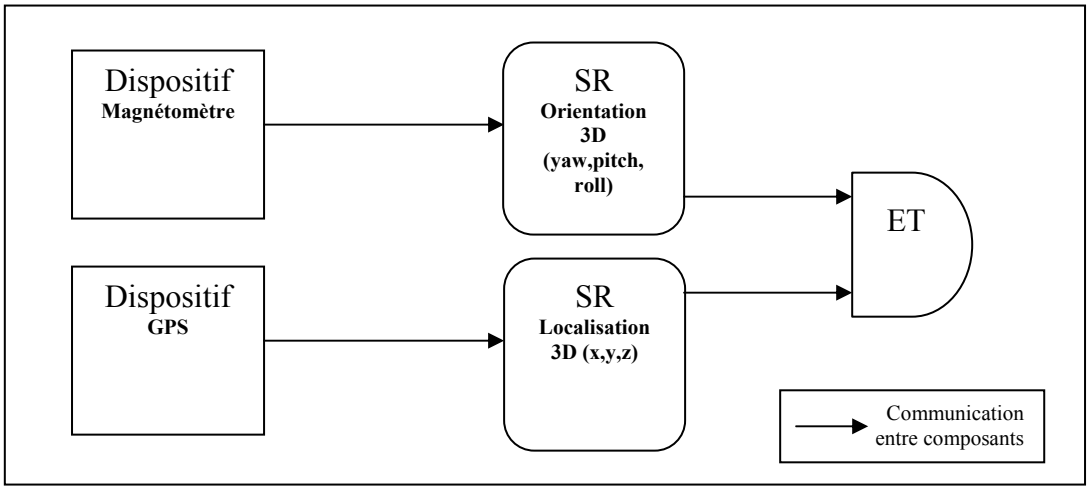


Figure 28 : Complémentarité syntaxique de deux systèmes représentationnels

L'exemple de la Figure 30 illustre la redondance et l'équivalence syntaxique entre deux modalités de sortie. La tâche du système consiste à poser une question à l'utilisateur lors d'un sous-dialogue de confirmation. Comme le montre la Figure 29, la première modalité est la visualisation sur l'écran d'une boîte de dialogue posant cette question. La seconde modalité est redondante à la première ; elle informe vocalement l'utilisateur de la nature de cette question.

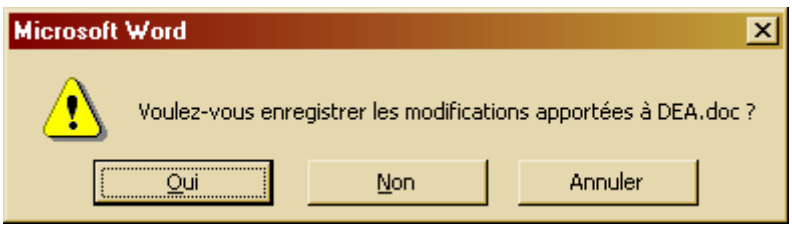


Figure 29 : Boîte de dialogue

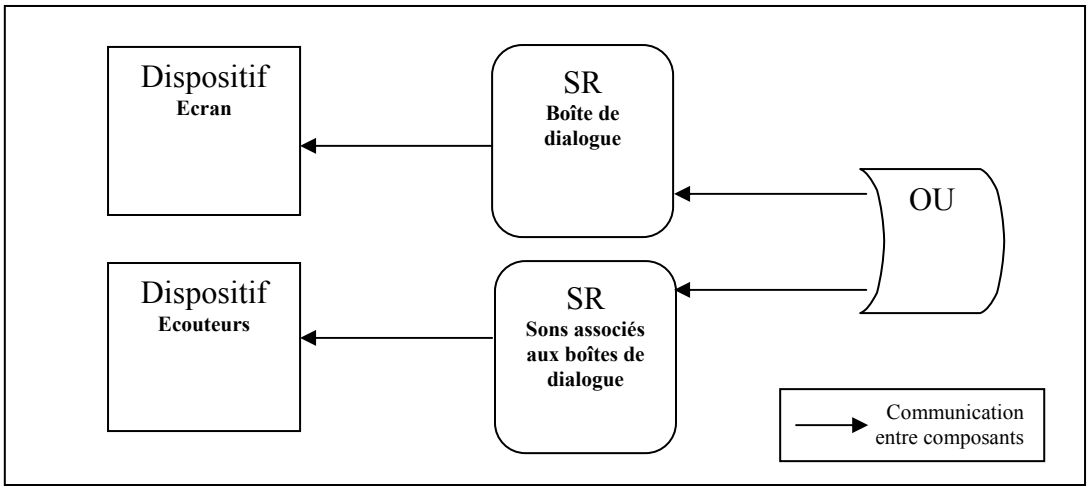


Figure 30 : Redondance et équivalence syntaxiques de deux systèmes représentationnels

Les exemples précédents reposent sur différentes modalités. De nombreuses modalités sont envisageables. Nos composants génériques de composition peuvent intégrer de deux à n composants et leur emploi permet de véhiculer les propriétés de complémentarité, de redondance et d'équivalence de CARE.

b) Perspectives pour la composition des environnements et des utilisateurs

Pour l'instant, les composants de composition ne sont valables que pour les composants Dispositifs et Systèmes représentationnels. Les composants Environnements et Utilisateurs ne peuvent être composés ainsi. Cependant, il est envisageable que deux composants Environnements soit complémentaires comme cela est le cas lorsque l'on considère un environnement représentant l'habitacle d'une voiture et son environnement externe. De plus, deux utilisateurs peuvent être redondants pour l'emploi d'un même dispositif. Nos composants de composition ne prennent pas en compte cet aspect et nous avons comme perspectives d'étudier l'impact de ces compositions sur l'interaction et sur notre plate-forme.

Nous avons fait apparaître deux niveaux de composition : le niveau articulatoire et le niveau syntaxique. Dans [Vernier 01], trois niveaux d'abstraction sont mis en évidence : les niveaux articulatoire, syntaxique et sémantique. Le niveau sémantique fait aussi partie de notre modèle et fait l'objet du paragraphe suivant. Ce niveau correspond à la dernière propriété de CARE, non encore traitée, l'assignation.

2.3 Composant Assignation

2.3.1 Définition

Avec les composants élémentaires et les composants de composition, nous fournissons les bases permettant de réaliser des interactions multimodales. Notre objectif est que notre modèle soit générique, donc non dépendant d'une application particulière. Pour cela, nous identifions un composant pour l'interface avec

l'application. Il s'agit du composant Assignation. Il fait référence à la propriété d'assignation de CARE et permet d'associer une modalité ou un ensemble composite de modalités à une tâche de l'application. Dans [Vernier 01], ce niveau correspond au niveau sémantique. Nous préférons le désigner comme le niveau « sémantico-pragmatique » car l'information fait sens pour l'application (sémantique) tout en étant relative aux tâches de l'application (pragmatique). Il représente ainsi le plus haut niveau d'abstraction de la partie interactive du système et permet la mise en correspondance des informations fournies par le moteur de fusion avec les tâches de l'application.

2.3.2 Symbole

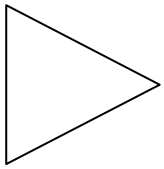


Figure 31 : Symbole du composant Assignation

2.3.3 Propriétés

Le Tableau 12 rassemble les propriétés du composant Assignation. Nous retrouvons quelques propriétés communes avec les autres types de composants comme : le nom du composant, sa version, le nom du créateur, la direction, l'état, les temps de traitement et l'historique.

Ce composant permet de vérifier la propriété ergonomique d'observabilité. Dans certains cas, il est possible que les informations délivrées en entrée de l'application (après traitement physique et logique) doivent être perçues par l'utilisateur. Les informations d'entrée sont fortement couplées aux informations de sortie. Par exemple, dans une voiture, la modalité d'entrée qui informe de la vitesse est utilisée pour les systèmes électroniques comme les régulateurs de vitesse mais elle est aussi directement perçue par l'utilisateur. Dans ce cas, nous définissons que la modalité d'entrée doit être perçue directement et nous définissons ainsi la propriété « observabilité temps-réel ».

Propriété	Description
nom du composant	Le nom du composant commence par « A_ » suivi d'une chaîne de caractères représentant l'assignation.
version du composant	Le numéro de version du composant (Exemple : V1.2).
nom du créateur	Le nom du créateur du composant.
direction	Le sens de l'assignation est caractérisé par cette propriété. La direction « entrée » (resp. « sortie ») illustre le flux de données de l'utilisateur vers le système (resp. du système vers l'utilisateur). La direction « entrée_sortie » concerne les dispositifs permettant des flux utilisateur->système et système->utilisateur.
état	Cette propriété permet de connaître l'état courant de fonctionnement du composant. Il peut être en marche (valeur « marche »), à l'arrêt (valeur « arrêt »), en attente (valeur « pause ») ou en dysfonctionnement (valeur « erreur »).
temps de traitement	Le temps de traitement correspond au moment où le dispositif traite une donnée. Chaque donnée fournit par le dispositif est estampillée par un temps de traitement.
temps maximum de traitement	Cette propriété représente le temps maximum que le composant doit mettre pour traiter et délivrer ses données.
observabilité temps-réel	Cette propriété (valeur « vrai » ou « faux ») définit que les informations délivrée en entrée est rendue observable à l'utilisateur.
historique de chaque propriété	Pour chacune des propriétés, un historique de leurs valeurs est sauvegardé.

Tableau 12 : Propriétés du composant Assignation

2.3.4 Règles et métriques pour le composant Assignation

Une règle est considérée pour le composant Assignation. D'autres pourront être définies.

> **Règle 1** : Si le composant Assignation a la propriété d'observabilité temps-réel, la direction du composant Assignation est entrée et sortie (la valeur de la direction est « entrée_sortie »).

En résumé, le composant Assignation constitue l'interface logicielle entre notre moteur de fusion de modalités et l'application. Il assure la propagation et la réception des données entre les modalités (simples ou composées) et l'application. Un schéma général d'adaptation montre, au paragraphe suivant, l'intégration du composant Assignation entre les autres composants ICARE et l'application.

2.4 Schéma général d'utilisation des composants pour une application donnée

Les composants présentés permettent de concevoir l'interaction indépendamment de l'application. La Figure 32 montre un exemple simple d'un modèle d'interaction. On y retrouve les trois niveaux d'abstraction : le niveau articulatoire, le niveau sémantique et le niveau sémantico-pragmatique. La partie gauche correspond aux modalités et à leur fusion. En entrée, deux modalités sont disponibles. Chacune représente un couple <Dispositif, Système représentationnel>. Elles sont toutes les deux complémentaires et leur complémentarité est associée à une tâche. En sortie, une tâche de l'application, comme par exemple la présentation d'une valeur, est redondante selon deux modalités. En gris sur le schéma, l'application existante reçoit et envoie les données utiles à l'interaction aux composants Assignation qui assure l'indépendance de l'application vis à vis des autres composants de notre moteur de fusion. Un exemple plus complexe est présenté dans le chapitre III. Il s'agit d'une application concrète dont l'interaction multimodale a été développée entièrement avec des composants ICARE.

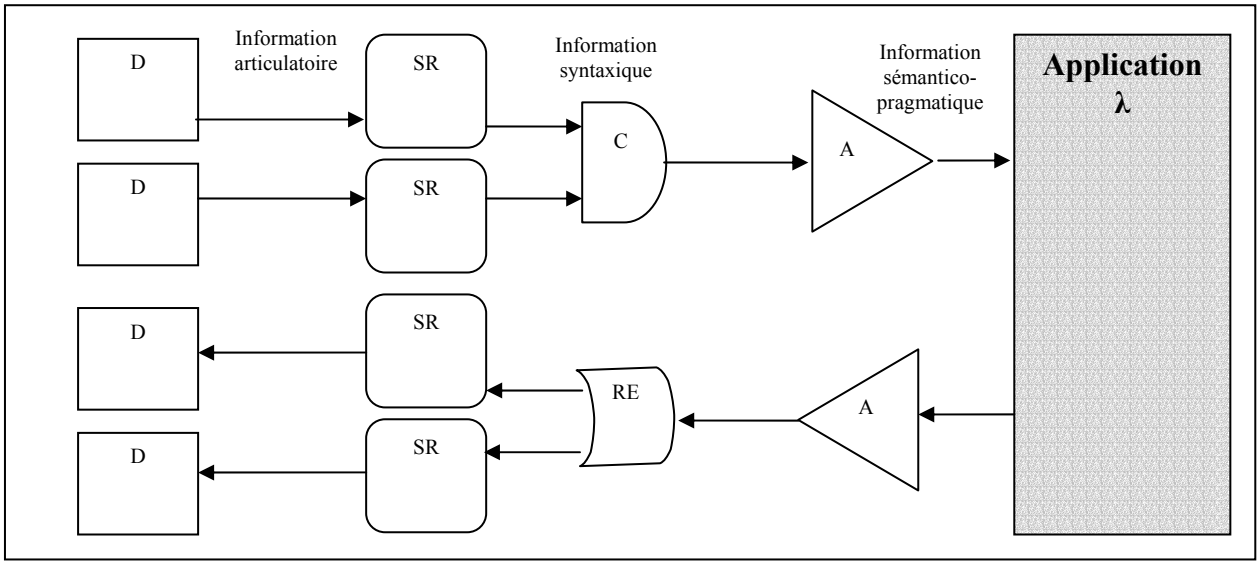


Figure 32 : Exemple d'utilisation des composants connectés à une application existante

Au travers de ce schéma et des composants qui interviennent dans notre plate-forme, nous constatons que notre approche permet de distinguer clairement la partie interaction du reste de l'application, comme dans la plupart des modèles conceptuels d'architecture logicielle.

Par rapport au processus de développement logiciel, il convient de noter que les concepts manipulés pendant la phase de spécification sont explicites jusqu'à la phase de codage. En effet, les spécifications identifient les différentes modalités et leurs compositions par rapport aux tâches de l'application. La conception fournit le modèle correspondant sous la forme d'un schéma semblable à celui de la Figure 32. Ensuite, pendant le codage, les composants de la plate-forme ICARE sont utilisés et assembler directement en suivant le schéma conceptuel.

3 Synthèse

Comme énoncé en introduction de ce chapitre, ICARE fournit plusieurs types de composants pour la création de nouveaux composants. Ces composants sont manipulés durant la phase de spécification-conception du logiciel, c'est-à-dire depuis l'analyse conceptuelle jusqu'à l'implémentation.

ICARE fournit des outils pour concevoir et développer des interactions multimodales. Les modèles des composants élémentaires et les modèles génériques des composants de composition et d'assignation permettent de réaliser des systèmes interactifs complexes. En effet, avec ICARE, une modalité est formée par un ou plusieurs dispositifs associés à un ou plusieurs systèmes représentationnels. La composition permet d'associer des dispositifs ou des langages complémentaires, redondants et équivalents. L'assignation garantit le lien avec le niveau tâche de l'application. La plupart des concepts identifiés dans l'espace problème ont été pris en compte et notamment les notions ergonomiques qui n'étaient présentes dans aucun autre modèle. De plus, les caractéristiques de l'utilisateur et de l'environnement sont aussi intégrées au modèle et à son implémentation.

Par rapport aux propriétés des moteurs de fusion définies au paragraphe 5.1 de l'espace problème, notre approche se caractérise comme suit :

- La multimodalité peut être synergique, exclusive, alternée et concurrente.
- Le choix de la stratégie d'intégration est laissé au concepteur qui peut définir une stratégie précoce ou une stratégie différée.
- L'exploration des possibilités de relation entre modalités se fait en profondeur d'abord.
- Les données qui sont complémentaires sont fusionnées selon leur proximité temporelle.
- Les informations sont transmises au niveau suivant si et seulement si elles sont complètes.
- Les données incompatibles ne sont jamais fusionnées.
- Les composants communiquent entre eux sous forme d'événements et chacun fournit à l'autre une représentation unique des données qui peut néanmoins varier selon la technologie à composants choisie.
- Grâce à la technologie à composants, le modèle que nous fournissons permet de couvrir les étapes de spécification, de conception et d'implémentation du cycle de vie du logiciel.

Le Tableau 13 résume les caractéristiques des moteurs de fusion que l'on peut créer avec ICARE :

Type	Stratégie d'intégration	Stratégie d'exploration	Proximité temporelle	Complémentarité logique	Complétude de l'information	Incompatibilité des données
S,E,A,C	A définir	Profondeur	oui	oui	oui	oui

Représentation des infos	Couverture du cycle de vie	Propriétés ergonomiques	Modèle utilisateur	Environnement	Généricité
unique	S, C, I	oui	oui	oui	oui

Légende Type : S = Synergique, E = Exclusive, A = Alternée, C = Concurrent

Cycle de vie : C = Conception, S = Spécification, I = Implémentation

Tableau 13 : Propriété d'ICARE

Pour illustrer notre modèle à composants pour l'interaction multimodale, nous présentons dans le chapitre suivant une application que nous avons développée avec ICARE.

III. REALISATION LOGICIELLE : APPLICATION MEMO

Pour illustrer notre modèle à composants ICARE, nous avons développé une application, intitulée MEMO, qui constitue un prototype de réalité augmentée mobile. Les applications de réalité augmentée sont un cadre d'étude intéressant de l'interaction multimodale car elles font souvent intervenir plusieurs modalités actives (commandes explicites de la part de l'utilisateur) et plusieurs modalités passives (suivi de l'utilisateur, position, mouvement, etc.). De nombreuses applications de réalité augmentée existent. Citons la Touring Machine [Feiner 97] qui permet d'explorer un environnement urbain, l'application MAGIC [Renevier 01] qui fournit un outil d'aide aux fouilles archéologiques ou encore TROC [Bouchet 02] qui est un jeu où les joueurs doivent retrouver des objets numériques répartis sur un terrain de jeu réel. L'interaction dans MEMO suit les mêmes principes que celles de ces trois applications mais le domaine d'application est différent.

Le paragraphe 1 décrit de façon générale l'application. Les spécifications externes font l'objet du paragraphe 2. La conception est présentée au paragraphe 3 et le codage au paragraphe 4.

1 Description de l'application

Ce paragraphe résume le cahier des charges de l'application MEMO. Nous y décrivons les fonctionnalités, puis la plate-forme matérielle et enfin, les contraintes logicielles.

1.1 Fonctionnalités

L'application MEMO, sorte de GeoNote [Persson 01], permet à un utilisateur de découvrir, lire, poser ou supprimer des post-its (notés mémos) liés à un endroit physique. Par exemple, nous pouvons envisager comme scénario d'utilisabilité, une personne A qui dépose, en partant de son bureau, un mémo sur la voiture d'un collègue B pour lui rappeler le dîner de ce soir : lorsque le collègue B quitte son bureau, il découvre le post-it en arrivant à sa voiture.

Dans la première version de MEMO, nous avons restreint les fonctionnalités : l'utilisateur mobile peut découvrir et visualiser des mémos déjà créés, peut les ramasser, les détruire ou les repositionner ailleurs. La création d'un mémo est exclue de cette première version. L'environnement constitue le monde réel qui entoure l'utilisateur et les mémos sont numériques. Quand l'utilisateur se déplace, il visualise les mémos et peut les manipuler pour les supprimer ou les repositionner. Une indication sonore lui indique le mémo qui est le plus proche de lui. D'autre part, l'utilisateur dispose aussi de l'heure courante.

1.2 Plate-forme matérielle

La plate-forme matérielle de l'utilisateur est constituée d'un ordinateur portable, de lunettes de réalité augmentée semi-transparentes, d'écouteurs, d'un capteur d'orientation. De plus, un microphone et un système de localisation doivent faire partie de notre plate-forme matérielle. Nous n'en disposons pas et nous avons décidé de les simuler par une technique de magicien d'Oz [Bouchet 02].

La Figure 33 montre la plate-forme matérielle.

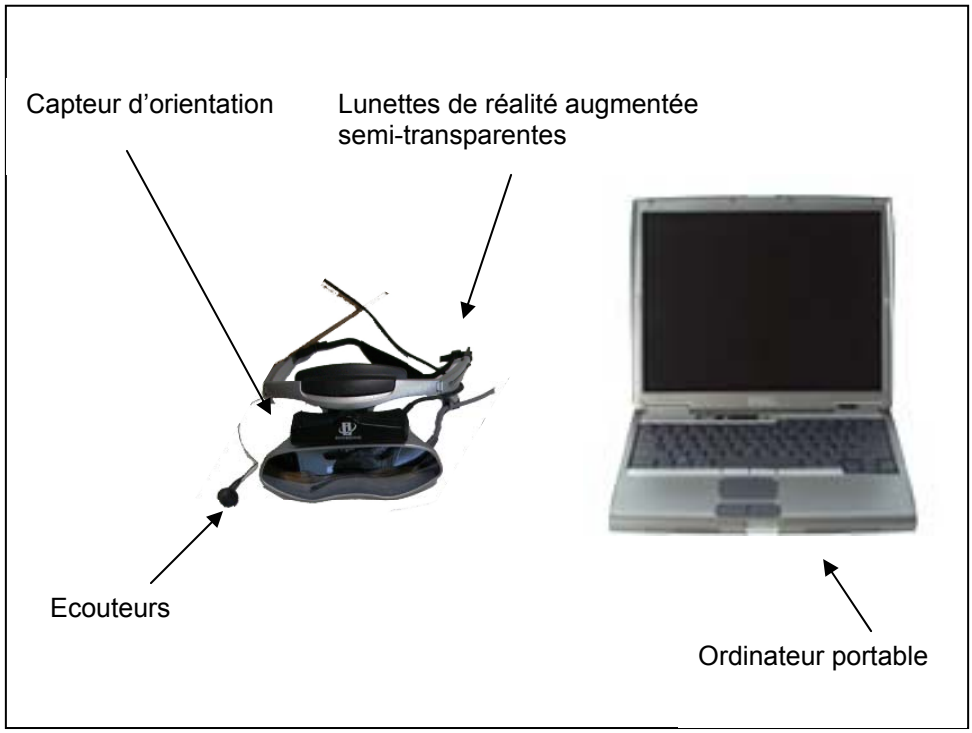


Figure 33 : Plate-forme matérielle de MEMO

1.3 Contraintes logicielles

Pour la mise en œuvre des composants ICARE, nous avons choisi les JavaBeans de Sun comme technologie à composants. Cette technologie a été choisie car les composants répondent à notre modèle et ils peuvent être manipulés et assemblés visuellement. Le travail du développeur est ainsi simplifié.

Le reste de l'application suit la programmation orientée objet et utilise le langage Java et la boîte à outils Java3D pour la visualisation des mémos en 3 dimensions.

Pour le développement, un environnement de programmation est utilisé, il s'agit de la version 8.3 de CodeWarrior.

2 *Spécifications externes*

La phase de spécifications externes regroupe deux étapes. La première consiste à identifier les tâches utilisateurs offertes par le système tandis que la seconde est dédiée aux spécifications de l'interface.

2.1 Tâches de l'application MEMO

2.1.1 Arbre des tâches

Les tâches de l'application MEMO sont décrites avec la notation HTA dont les opérateurs principaux sont rappelés à la Figure 34.

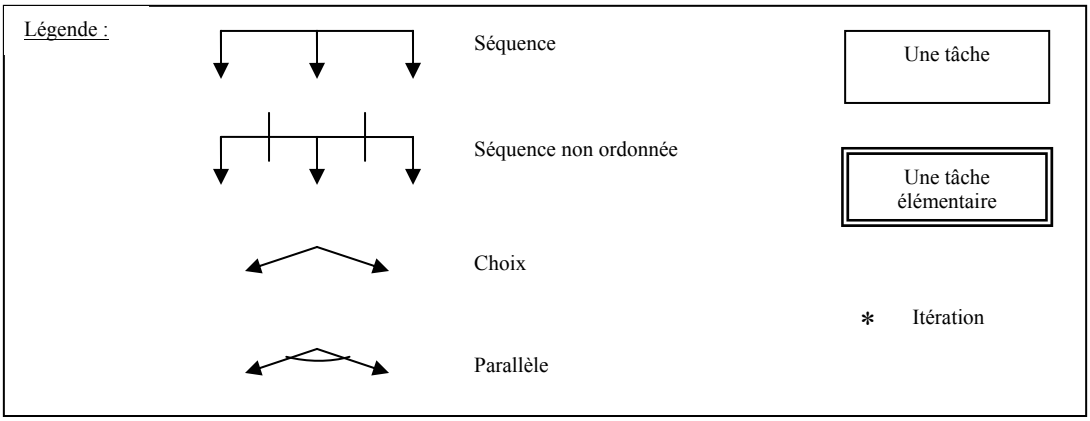


Figure 34 : La notation HTA

L'application MEMO se décompose en deux tâches séquentielles :

1. Utilisation de l'application
2. Quitter l'application

La Figure 35 montre ces deux tâches séquentielles qui sont affinées dans les paragraphes suivants.

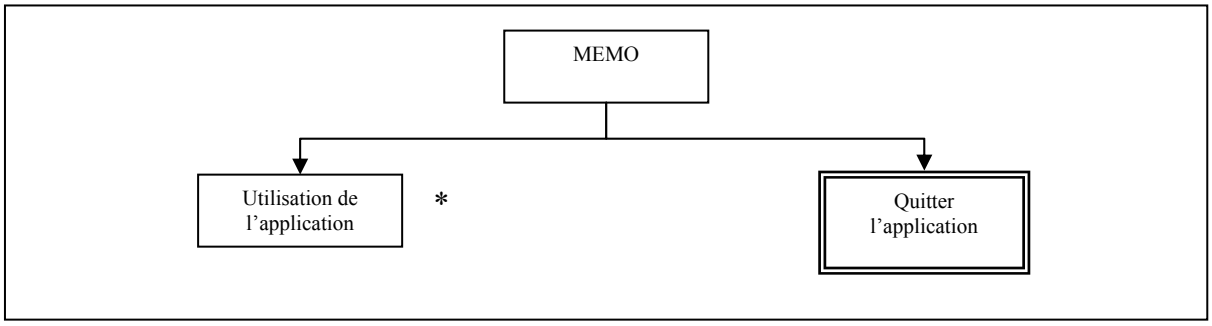


Figure 35 : L'application MEMO

Comme le montre la Figure 36, l'utilisation de l'application se décompose en deux tâches : explorer le terrain et manipuler les mémos.

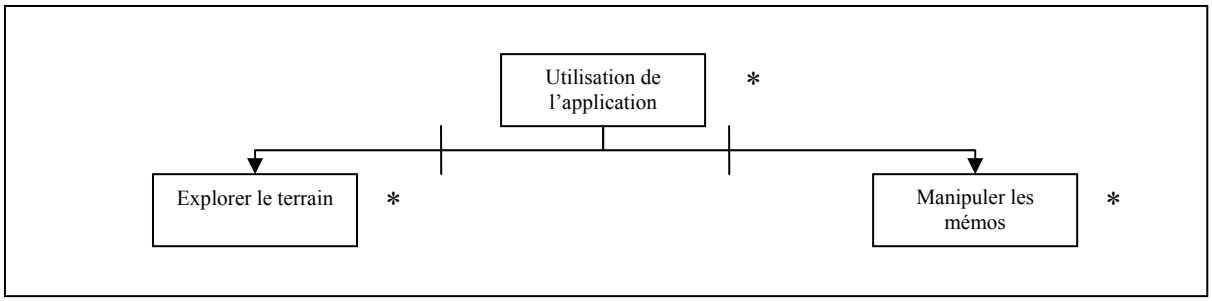


Figure 36 : Utilisation de l'application

La tâche « Explorer le terrain », décrite à la Figure 37 est décomposée en plusieurs tâches qui peuvent être accomplies en parallèle :

1. Localiser le mémo le plus proche
2. Visualiser le terrain réel
3. Visualiser les mémos disponibles sur le terrain
4. Visualiser l'heure courante

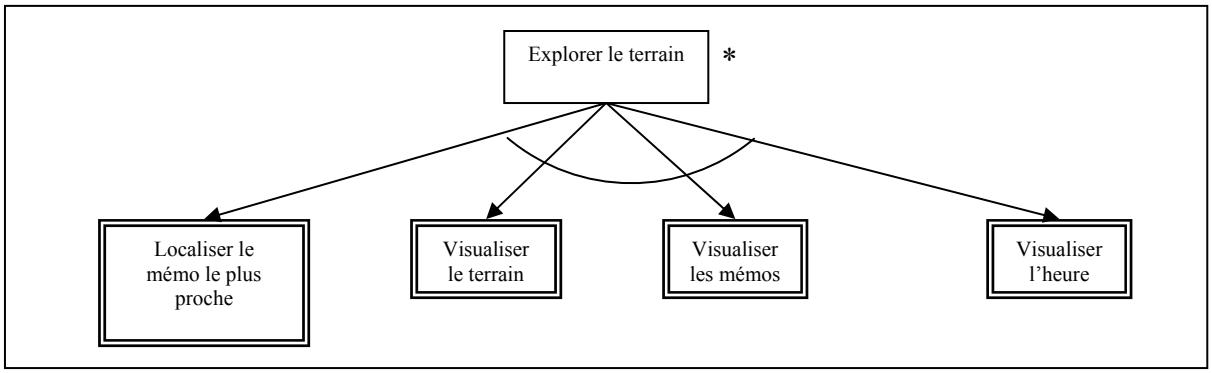


Figure 37 : Explorer le terrain

Comme le montre la Figure 38, la tâche « Manipuler les mémos » consiste à sélectionner un mémo en le ciblant puis à le gérer en le manipulant, différents choix sont alors offerts à l'utilisateur :

1. Ramasser un mémo. Une fois sélectionné, l'utilisateur peut ramasser le mémo. L'intitulé du mémo est alors affiché dans le casque et il n'est plus disponible sur le terrain.
2. Poser le mémo qui a été ramassé précédemment. L'utilisateur pose devant lui (à 1 mètre) le mémo qu'il a ramassé auparavant.
3. Supprimer un mémo.
4. Supprimer un mémo ramassé.

A un instant donné, un seul mémo peut être ramassé. Si un mémo est ramassé alors qu'un autre a déjà été ramassé et non redéposé, l'ancien mémo ramassé est automatiquement détruit.

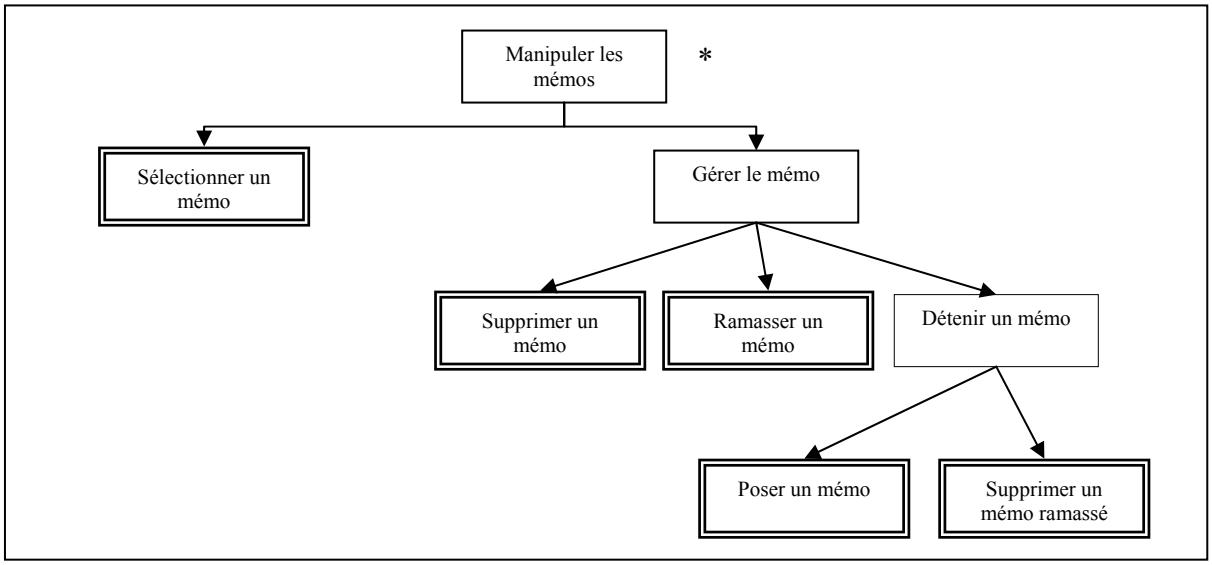


Figure 38 : Manipuler les mémos

2.2 Spécifications de l'interface

Après avoir identifié les tâches du système, nous décrivons l'interface de l'application MEMO, correspondant à l'arbre des tâches identifié.

Nous rappelons que MEMO fonctionne sur une plate-forme matérielle de réalité augmentée décrite au paragraphe 1.2. Par conséquent, la tâche qui consiste à visualiser le terrain est implicite car l'utilisateur visualise le terrain réel qui l'entoure au travers des lunettes semi-transparentes. Seules les spécifications des interfaces concernant les autres tâches sont développées.

2.2.1 Visualisation, sélection et localisation des mémos

Outre le monde réel, plusieurs informations numériques (sonores et visuelles) sont ajoutées à la réalité. Pour localiser le mémo le plus proche, un son en trois dimensions est utilisé et est émis à partir du lieu du mémo. Au niveau visuel et comme le montre la Figure 39, l'heure courante est affichée en vert en bas à gauche, les mémos sont visibles en fonction du positionnement de l'utilisateur et de l'orientation de sa tête. Ici, un seul mémo est visible. Il contient trois lignes de texte écrites en bleu. De plus, un viseur, une croix rouge au centre de l'écran, permet de sélectionner les mémos. Le viseur est statique et positionné au centre des lunettes. L'utilisateur doit bouger la tête pour amener le viseur sur un mémo.

Figure 39 : Visualisation d'un mémo

2.2.2 Ramassage d'un mémo

En visant (et donc en sélectionnant) un mémo, l'utilisateur peut ensuite effectuer plusieurs actions. Il peut par exemple ramasser le mémo et le transporter avec lui. Pour cette tâche, il y a deux façons de procéder (deux modalités équivalentes). La première consiste à viser le mémo et à cliquer sur le bouton gauche de la souris. La seconde nécessite de viser le mémo et de prononcer le mot « ramasser ». Comme le montre la Figure 40, le mémo visé est alors ramassé ; il n'est plus visible à l'endroit où il avait été posé. Cependant, une note écrite en vert au bas de l'écran informe l'utilisateur qu'un mémo a été ramassé. L'intitulé de ce mémo est écrit à côté. Quand l'action est réalisée, un message « MEMO RAMASSE » est inscrit au-dessous du viseur.

Figure 40 : Ramassage d'un mémo

Nous rappelons qu'il n'est pas possible de ramasser plusieurs mémos. Si un mémo est déjà ramassé et que nous en ramassons un autre, le premier est automatiquement détruit.

2.2.3 Pose d'un mémo

Quand un mémo est ramassé, comme décrit au paragraphe 2.2.2, l'utilisateur le transporte. A tout moment, il peut poser ce mémo sur le terrain. Comme pour le ramassage, il y a deux façons de procéder. L'une consiste à viser l'endroit où le mémo doit être déposé et à cliquer sur le bouton gauche de la souris. L'autre consiste à viser l'endroit et à prononcer le mot « poser ». Ensuite, et comme le montre la Figure 41, le mémo est posé sur le terrain juste sous le viseur. Un message « MEMO POSE » s'affiche au-dessous du viseur.

Figure 41 : Pose d'un mémo

2.2.4 Suppression d'un mémo ramassé

Quand un mémo est ramassé, l'utilisateur le transporte. S'il ne vise pas un autre mémo, il peut supprimer ce mémo. Comme pour le ramassage, il y a deux façons de procéder. L'une consiste à cliquer sur le bouton droit de la souris. L'autre nécessite de prononcer les mots « supprimer le mémo ramassé ». Comme le montre la Figure 42, le mémo ramassé est alors supprimé. Le texte descriptif du mémo ramassé disparaît et un message « MEMO RAMASSE SUPPRIME » s'affiche au-dessous du viseur. A noter qu'il est aussi possible de supprimer un mémo sans l'avoir ramassé au préalable, cette manipulation est décrite au paragraphe suivant.

Figure 42 : Suppression d'un mémo ramassé

2.2.5 Suppression d'un mémo

Il est possible de supprimer un mémo sans le ramasser. Comme pour les autres manipulations de mémo, il y a deux façons de procéder. L'une consiste à viser le mémo et à cliquer sur le bouton droit de la souris. L'autre nécessite de viser le mémo et de prononcer le mot « supprimer». Ensuite, et comme le montre la Figure 43, le mémo est supprimé. Le message « MEMO SUPPRIME » s'affiche au-dessous du viseur.

Figure 43 : Suppression d'un mémo

L'interface étant spécifiée, l'étape suivante consiste à concevoir l'architecture logicielle de MEMO.

3 Conception

Dans ce paragraphe, nous expliquons l'architecture logicielle adoptée et le schéma conceptuel réalisé avec ICARE.

3.1 Architecture logicielle

L'architecture conceptuelle globale de MEMO suit en partie le modèle Arch présenté à la Figure 44. Dans ce modèle, cinq modules sont organisés sous forme d'une arche : un noyau fonctionnel, un adaptateur de domaine, un contrôleur de dialogue, un module de présentation et un module d'interaction. Le noyau fonctionnel implémente les concepts du domaine. L'adaptateur de domaine ajuste les représentations des objets conceptuels qui sont transmis entre le noyau fonctionnel et le contrôleur de dialogue. Le contrôleur de dialogue gère l'enchaînement des tâches. Le module de présentation permet de définir une boîte à outils virtuelle qui est concrétisée dans le module d'interaction.

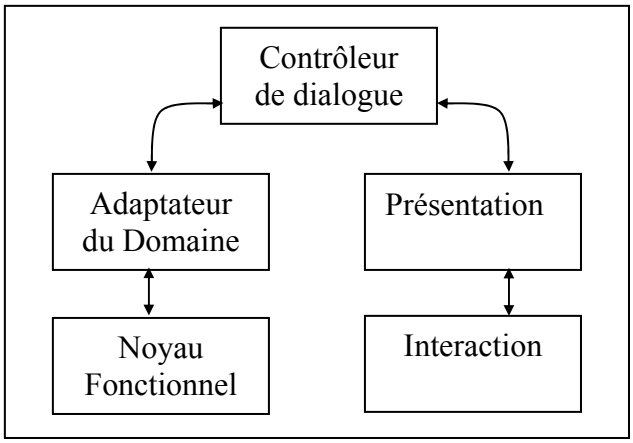


Figure 44 : Le modèle Arch

De ce modèle, seuls les modules noyau fonctionnel, adaptateur du domaine et contrôleur de dialogue sont conservés. Comme le montre la Figure 45, les modules présentation et interaction laissent place aux composants ICARE.

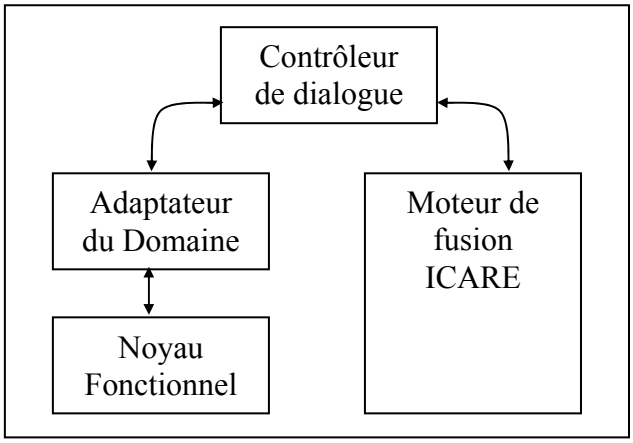


Figure 45 : Le modèle Arch intégrant nos composants ICARE

Le paragraphe suivant décrit précisément l'architecture du moteur de fusion développé avec ICARE, c'est-à-dire les composants et leurs relations.

3.2 Architecture du moteur de fusion

L'interaction multimodale de MEMO est réalisée avec la plate-forme ICARE. A des fins analytiques, nous avons séparé l'interaction en entrée de l'interaction en sortie. Cependant, il est tout à fait envisageable de les réunir dans un seul schéma.

Pour éviter une surcharge du schéma, les composants Utilisateur et Environnement ne sont pas représentés. L'utilisateur est relié aux dispositifs de sortie. Il les prévient si ses capacités sensorielles changent. L'environnement est relié à l'ensemble des autres composants en leur fournissant l'heure et la date courante.

Nous rappelons à la Figure 46 la signification des symboles des Figure 47 et Figure 48.

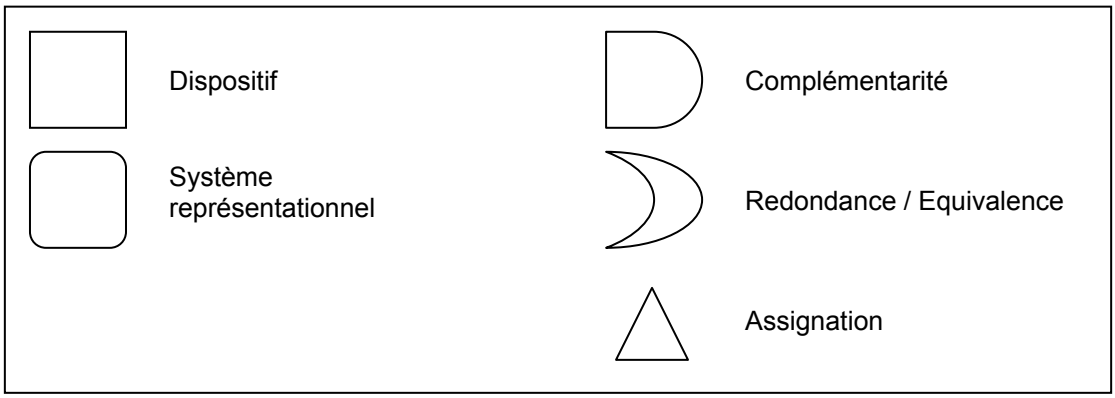


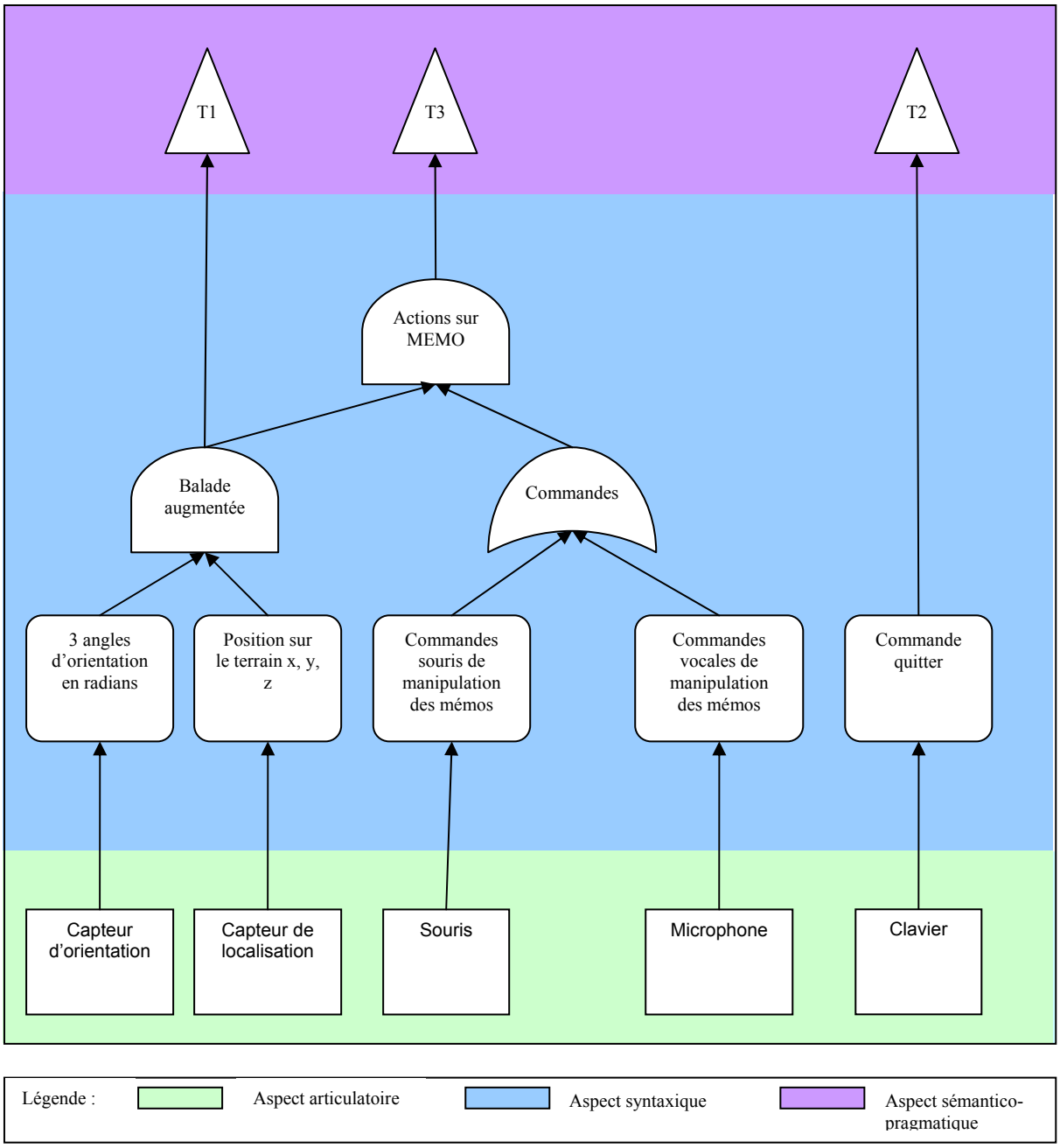
Figure 46 : Signification des symboles d'ICARE

3.2.1 Interaction en entrée

L'interaction en entrée de MEMO est multimodale. Comme le montre la Figure 47, cinq modalités sont définies et certaines sont composées. Nous en retrouvons cinq :

- la modalité d'orientation est représentée par le couple capteur d'orientation (dispositif) et les trois angles d'orientation en radians (système représentationnel),
- la modalité de localisation est représentée par le couple capteur de localisation (dispositif) et la position sur le terrain (système représentationnel),
- la modalité pour sortir de l'application est représentée par le couple clavier (dispositif) et commande quitter (système représentationnel),
- la modalité de manipulation des mémos avec la souris est représentée par le couple souris (dispositif) et commandes souris de manipulation des mémos (système représentationnel),
- la modalité de manipulation des mémos avec la voix est représentée par le couple microphone (dispositif) et commandes vocales de manipulation des mémos (système représentationnel).

La modalité d'orientation et la modalité de localisation sont complémentaires car leur complémentarité permet de savoir où l'utilisateur regarde et donc quel mémo il vise. La modalité de manipulation des mémos avec la souris et la modalité de manipulation des mémos avec la voix permettent toutes les deux de gérer les mémos. Elles sont donc équivalentes et redondantes. En complétant des informations des modalités de localisation et d'orientation, il est possible de déterminer sur quel mémo la commande doit prendre effet.



Les trois composants d'assignation font le lien avec les tâches suivantes :

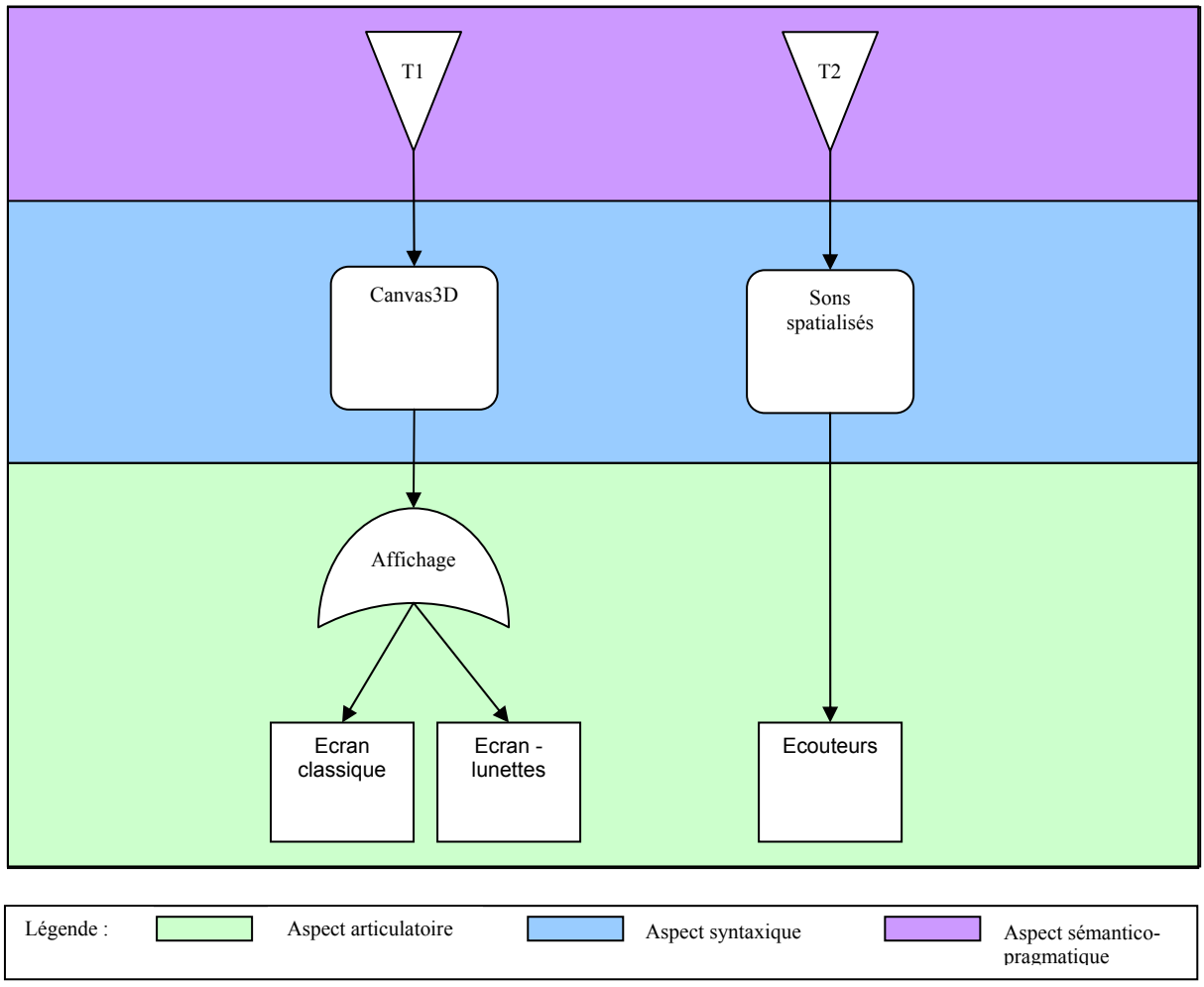
- T1 : mises à jour de l'orientation et de la localisation de l'utilisateur
- T2 : commande pour quitter l'application
- T3 : actions sur les mémos ciblés

Figure 47 : Schéma de l'interaction multimodale en entrée

3.2.2 Interaction en sortie

Dans MEMO, l'interaction en sortie est aussi multimodale. Comme le montre la Figure 48, deux modalités de sortie sont définies :

- la modalité visuelle est constituée du système représentationnel Canvas3D qui est un widget de Java3D pour le rendu des scènes en trois dimensions et deux dispositifs de sortie redondants qui sont l'écran de l'ordinateur portable et l'écran des lunettes semi-transparentes ,
- la modalité sonore est représentée par le système représentationnel sons spatialisés et par le dispositif écouteurs.



Les trois composants d'assignation font le lien avec les tâches suivantes :

- T1 : Visualisation graphique des mémos visibles, de l'heure courante, du retour d'information sur la gestion du mémo sélectionné, de l'intitulé du mémo ramassé.
- T2 : Localisation du mémo le plus proche

Figure 48 : Schéma de l'interaction multimodale en sortie

Les composants ICARE de l'application MEMO étant présentés, nous décrivons maintenant leur implémentation.

4 Implémentation

Dans cette partie nous nous focalisons principalement sur l'implémentation des composants d'ICARE. Le reste de l'application est programmée en Java selon le paradigme de l'orienté objet. Pour la scène en trois dimensions contenant les mémos, la boîte à outils Java3D est utilisée. En ce qui concerne le moteur de fusion, son implémentation est réalisée comme suit :

Tout d'abord, les composants identifiés sur la Figure 47 et la Figure 48 sont créés en suivant les modèles du composant Dispositif, du composant Système représentationnel, du composant Complémentarité, du composant Redondance/Equivalence et du composant Assignation définis dans le chapitre II. Seul le composant Redondance/Equivalence nommé « Affichage » à la Figure 48 n'est pas développé car cette composition est déjà gérée par le pilote de la carte graphique. Les autres suivent les spécifications des JavaBeans [Beans 97].

Les composants étant créés, leur assemblage s'effectue de manière classique à la main ou par manipulation directe dans un environnement de programmation gérant les JavaBeans. Les propriétés des composants sont configurées. Par exemple : la stratégie d'intégration des composants Complémentarité est définie comme « précoce », la modalité de manipulation vocale des mémos est configurée avec un facteur de confiance moins élevé que celui de la modalité de manipulation avec la souris (en cas de redondance, la manipulation avec la souris est privilégiée).

La communication se fait par événement-réaction. Le lien entre deux composants (l'un « source », l'autre « destination ») détermine le traitement (appel de méthode) que doit faire le composant « destination » en fonction de l'événement déclenché.

Ensuite, une méthode d'initialisation du module ICARE et l'intégration des fichiers avec le reste de l'application sont réalisées manuellement. L'application MEMO est alors complètement développée.

5 Conclusion

Le prototype MEMO montre que le modèle et la plate-forme ICARE constituent une approche innovante et adaptée pour le développement de l'interaction multimodale. Les composants de MEMO sont maintenant réutilisables dans n'importe quelle autre application et ils correspondent donc aux premiers composants de la plate-forme. En perspective, le prototype MEMO intégrera un nouveau dispositif d'entrée/sortie et de nouvelles fonctionnalités. Le nouveau dispositif sera un ordinateur de poche sur lequel il sera possible de manipuler les mémos (ramassage, suppression, pose) et d'éditer de nouveaux mémos. De plus, un utilisateur pourra créer des mémos pour soi ou à destination d'autres utilisateurs.

CONCLUSION

1 Synthèse

Nos travaux contribuent au domaine de l'ingénierie logicielle des interfaces multimodales. Avec une approche basée sur les composants, nous fournissons les outils nécessaires pour la mise en œuvre de plusieurs modalités simples ou composées en assurant une cohérence ergonomique dans leur composition.

Le premier chapitre de ce mémoire (espace problème) pose le cadre de notre étude en introduisant les concepts utiles à nos travaux. Ces concepts sont issus des travaux de psychologie cognitive, des définitions des termes « modalité » et « multimodalité », des cadres taxinomiques, de l'étude de différents moteurs de fusion et de l'état de l'art des technologies à composants.

Cet espace problème met en évidence les différents aspects à prendre en compte dans nos travaux. L'étude des moteurs de fusion existants montre clairement leurs limites dans l'intégration du modèle utilisateur, du modèle de l'environnement, des propriétés ergonomiques des systèmes multimodaux et de leur généricité par rapport à l'application. Notre contribution, présentée dans l'espace solution, vise à combler ces lacunes. Ainsi, nous proposons :

- des composants élémentaires symbolisant les modalités (entrée et sortie), l'utilisateur et l'environnement,
- des composants de composition permettant la fusion des composants élémentaires en respectant les propriétés ergonomiques des systèmes multimodaux,
- des composants d'assignation permettant de lier la partie interactive avec le reste de l'application.

Cet ensemble de composants permet au concepteur et au développeur de disposer d'une plate-forme permettant de mettre en place plusieurs modalités et de réaliser leur combinaison dans un moteur de fusion général et paramétrable.

Enfin, nous avons montré la validité de notre approche en développant le système interactif multimodal MEMO avec notre plate-forme.

2 Perspectives

Un premier travail à court terme consiste à étendre le prototype MEMO avec de nouvelles modalités d'interaction et de nouvelles fonctionnalités. Ce faisant, nous étendons notre plate-forme ICARE. Nous souhaitons intégrer un ordinateur de poche comme nouveau dispositif d'entrée et de sortie pour l'édition et la manipulation des mémos. De plus, nous envisageons que MEMO puisse être utilisé par plusieurs utilisateurs et que ces derniers s'adressent personnellement des mémos. Cette extension nous permettra d'étendre la plate-forme ICARE avec de nouveaux composants Dispositifs et Systèmes représentationnels, d'affiner les propriétés du composant Utilisateur et du composant Environnement et d'identifier de nouvelles règles de composition et d'assignation.

Comme autre perspective, nous souhaitons développer un composant logiciel visuel qui permette d'observer l'état de tous les composants élémentaires mis en place dans l'application. Ce composant est destiné en particulier aux développeurs et aux testeurs pour qu'ils puissent vérifier le fonctionnement des composants pendant l'exécution de l'application.

A moyen terme, nous étudierons les différents cas mettant en œuvre plusieurs utilisateurs et plusieurs environnements combinés et nous statuerons sur la possibilité d'utiliser nos composants de composition pour ces deux types de composants.

Outre l'amélioration des composants JavaBeans de cette plate-forme, nous souhaitons utiliser une autre technologie à composants pour mettre en œuvre nos moteurs de fusion. Cela nous permettrait de valider notre modèle à composants ICARE avec une autre technologie à composants.

Enfin, nous envisageons d'intégrer notre plate-forme ICARE dans le cadre d'un contrat avec THALES, noté INTUITION (INTeraction mUltimodale Intégrant les Technologies InnOvaNtes). Ce projet consiste à créer une plate-forme de développement pour les systèmes multimodaux. Notre équipe à la charge de concevoir un outil de développement pour l'interaction multimodale en entrée. La plate-forme THALES doit être validée dans deux applications militaires. De nouveaux composants ICARE seront développés et correspondront aux modalités de ces deux applications. Ce projet permettra de vérifier à nouveaux que notre plate-forme est générique et adaptée.

LISTE DES FIGURES ET DES TABLEAUX

Les figures

Les tableaux

BIBLIOGRAPHIE

[Allen 83] J. Allen, *Maintaining Knowledge about Temporal Intervals*, Communication of ACM 26 (11), pp. 832-843, Novembre 1983

[Barbier 02] F. Barbier, C. Cauvet, M. Oussalah, D. Rieu, S. Bennasri, C. Souveyet, *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.

[Bass 00] L. Bass, C. Buhman, S. Comella-Dorda, F. Long, J. Robert, R. Seacord, K. Wallnau, *Volume I : Market Assessment of Component-Based Software Engineering*, Carnegie Mellon University, Software Engineering Institute, TECHNICAL NOTE CMU/SEI-2001-TN-007, Mai 2000.

[Beans 97] *JavaBeans 1.01 specification*, Sun Microsystems, Juillet 1997.

[Bellik 92] Y. Bellik, D. Teil, *Multimodal Dialog Interface*, WWDU'92, Berlin, Septembre 92.

[Bernsen 93] N. Bernsen, *A revised generation of the taxonomy of output modalities*, Rapport du projet Esprit AMODEUS, numéro RP5-TM-WP11, 1994.

[Bolt 80] R. A. Bolt, *Put that there: Voice and gesture at the graphics interface*, Computer Graphics, pp262-270, 1980.

[Bouchet 02] J. Bouchet, L. Nigay, P. Renevier, L. Pasqualetti, *TROC : un jeu collaboratif sur support mobile exploitant des techniques de réalité augmentée*, Groupe de travail Mobilité et Ubiquité, 2ème assises nationales du GdR i^3, 2002.

[Bourguet 92] M. L. Bourguet, ICPplan : dialogue multimodal pour la conception de plans architecturaux, 19èmes JEP, pp. 369-374, Mai 1992.

[Buxton 83] W. Buxton, *Lexical and pragmatic considerations of input structures*, Computer Graphics, 17 (1), pp. 31-37, 1983.

[Card 83] S. Card, T. Moran, A. Newell, *The Psychology of Human-Computer Interaction*, Chapitre 2, The Human Information-Processor, Hillsdale, New Jersey : Lawrence Erlbaum Associates, p. 23-97, 1986.

[Card 90] S. Card, J. Mackinlay, G. Robertson, *The Design Space of Input Devices, Actes de la conférence Computer Human Interaction'90*, Seattle, p. 117-124, 1990.

[Corba 02] *formal/02-06-65 : CORBA Components*, v3.0 full specification, OMG, Juin 2002.

[Coutaz 94] J. Coutaz, L. Nigay, *Les propriétés « CARE » dans les interfaces multimodales*, actes de la conférence IHM'94, Lille, 1994.

[Coutaz 00] J. Coutaz, *Modélisation en Interaction Homme-Machine*, Support de cours du DEA ISC de l'Université Joseph Fourier, Grenoble, 2000.

[Dubois 01] E. Dubois, *Chirurgie Augmentée, un Cas de Réalité Augmentée ; Conception et Réalisation Centrées sur l'Utilisateur*, Thèse de l'Université Joseph Fourier, Grenoble, 2001.

[EJB 02] Sun Microsystems, *Enterprise JavaBeans Specification*, Version 2.1, Août, 2002.

[Favre 03] J.-M. Favre, *Organisational issues in CBSE*, cours sur les applications à base de composants du DEA ISC de l'Université Joseph Fourier, Grenoble 2003.

[Feiner 97] S. Feiner, B. MacIntyre, T. Höllerer, A. Webster, A Touring Machine : 3D Mobile Augmented Reality Systems for Exploring the Urban Environment, ISWC'97, pp 74-75, Cambridge, 1997.

[Foley 84] J. Foley, V. Wallace, P. Chan, *The Human Factors of computer Graphics interaction techniques*, IEEE computer Graphics and Applications, 4(11), p. 13-48, 1984.

[Frohlich 91] D. Frohlich, *The Design Space of Interfaces*, Multimedia Systems, Interaction and Applications, 1st Eurographics Workshop, Stockholm, Suède, Springer Verlag, p. 53-69, 1991.

[Gaildrat 83] V. Gaildrat, R. Caubet, R. Rubio, Conception d'un modeleur déclaratif de scènes tridimensionnelles pour la synthèse d'images, MICAD'93, Paris, 1993.

[IHM 92] *Compte rendu des ateliers*, IHM'92 Quatrièmes Journées sur l'Ingénierie des Interfaces Homme-Machine, 1992.

[McCall 77] J. McCall, *Factors in Software Quality*, General Electric Eds, 1977.

[Microsoft 02] Microsoft, *MicrosoftCOM White Papers*, les composants Microsoft http ://www.microsoft.com/com/wpaper/default.asp#COMPapers, 2002.

[Microsoft 03] Microsoft, *Plate-forme .NET*, http ://www.microsoft.com/net/, 2003.

[Nigay 94] L. Nigay, *Conception et modélisation logicielles des systèmes interactifs*, Thèse de l'Université Joseph Fourier, Grenoble, 1994.

[Nigay 96] L. Nigay, J. Coutaz, *Espaces conceptuels pour l'interaction multimédia et multimodale*, TSI 96, 1996.

[Norman 86] D. Norman, *Cognitive Engineering, User Centered System Design*, New Perspectives on Computer Interaction, édité par Norman D., Draper S., Hillsdale, New Jersey : Lawrence Erlbaum Associates, pp. 31-61, 1986.

[OSGi 03] OSGi, *OSGi Service Platform*, Release 3, Mars, 2003

[Persson 01] P. Persson, F. Espinoza, E. Cacciatore, *GeoNote : Social Enhancement of Physical Space*, CHI 01, 2001.

[Phipps 00] D. Phipps, *CBD: A Path to Uniform Component Design*, Gartner Group Strategic Analysis Report. Stamford, CT: Gartner Group, Janvier 2000.

[Rasmussen 86] J. Rasmussen, *Information processing and human-machine interaction, an approach to cognitive engineering*, livre édité par Elsevier Science Publishing, 1986.

[Renevier 01] P. Renevier, L. Nigay, *Mobile Collaborative Augmented Reality : The Augmented Stroll*, Engineering for Human-Computer Interaction, 2001.

[Sellen 92] A. Sellen, G. P. Kurtenbach, W. Buxton, The prevention of Mode errors Trough Sensory Feedback, Human Computer Interaction, Lawrence Erlbaum, Vol. 7, No 2, pp. 141-164, 1992.

[Szyperski 98] C. Szyperski. *Component Software: Beyond Object-Oriented Programming*, Addison Wesley, 1998.

[Vernier 01] F. Vernier, *La multimodalité en sortie et son application à la visualisation de grandes quantités d'information*, Thèse de l'Université Joseph Fourier, Grenoble, 2001.

[W3C 02] W3C Multimodal Interaction Framework, http://www.w3.org/TR/mmi-framework/, W3C Note 3, mai 2003.

[Williams 94] S. Willliams, C. Kindel, *The Component Object Model : A Technical Overview*, MSDN Library Microsoft Corporation, Octobre, 1994