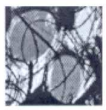
Démarche de Modélisation d'IHM3 avec B

SP 3
LOT 2
LISI/ENSMA

Auteurs : Y. Aït-Ameur, I. Aït-Sadoune, M. Baron, N. Kamel et J-M. Mota
Adresse : LISI / ENSMA - Téléport 2 - 1, avenue Clément Ader - B.P. 40109 - 86960 Futuroscope Cedex - France
Emails : {yamine, idir.aitsadoune, baron, kamel, mota}@ensma.fr

---



CLEARSY
SYSTEM ENGINEERING

ONERA

Table des matières

Table des figures

Chapitre 1

Introduction

1.1 Les interfaces Multi-modales ou IHM3

Un système est qualifié d'interactif lorsqu'il interagit avec son utilisateur en entrée et/ou en sortie. Il est doté en général d'une interface qui permet de gérer les interactions de l'utilisateur vers le système et/ou les interactions venant du système vers l'utilisateur.

Avec le développement de domaines de recherches tels que la synthèse de la parole, la vision par ordinateur et la synthèse d'image, les nouveaux dispositifs d'interaction issus de ces domaines ont conduit à la naissance des IHM multi-modales. L'objectif de ces interfaces est d'approcher la qualité de communication entre l'utilisateur et la machine du niveau de la communication humaine en offrant plus de choix dans l'interaction homme-machine. Ceci permettra le développement de systèmes plus adaptés aux personnes handicapées moteurs. De même, ce type d'IHM est très utile pour les systèmes de sécurité, en permettant la continuité de l'interaction en cas de panne d'un dispositif d'interaction.

Le mot "multi-modalité" est composé du préfixe "multi" qui désigne la multiplicité et le mot "modalité" qui désigne une technique ou une manière d'interagir avec la machine. Cette multiplicité est liée à la variété des canaux d'interaction mis à la disposition de l'utilisateur et aux différentes manières d'utiliser ces canaux. La multi-modalité est un thème de recherche au sein du domaine de l'interaction Homme-Machine assez récent. Richard Bolt [Bol80] a été le premier à proposer l'intégration de deux modalités d'interaction, la parole et le geste avec le paradigme "met ça là". Depuis, plusieurs travaux ont succédé couvrant plusieurs aspects tels que la fixation des concepts et le développement de spécifications, d'outils logiciels et les méthodologies d'évaluation appropriées.

Un des objectifs des recherches dans le domaine des IHM multi-modales est d'augmenter les capacités communicatives de l'ordinateur, en étudiant et en s'inspirant de la multi-modalité de la communication humaine pour améliorer la communication entre les utilisateurs et les ordinateurs. De même, plusieurs avancées conceptuelles ont été réalisées, que ce soit sur la caractérisation de l'usage des modalités ou sur la spécification de mécanismes de fusion de données issues de modalités distinctes.

1.2 Concepts de base

Plusieurs concepts liés à la multi-modalité ont été définis par la communauté des IHM, mais un consensus sur leur définition reste absent. Ces notions sont soit abordées par rapport à l'individu et dans ce cas elles sont dites centrées utilisateurs, soit par rapport à la technologie, et dans ce cas elles sont dites centrées système. On trouve une synthèse des différentes définitions dans [NC96]. Nous reprenons ici quelques définitions caractérisant ce type d'IHM en se basant sur cette synthèse et sur les travaux de [Bel95].

1.2.1 Média

En général le terme média désigne un support technique de l'information. De là, plusieurs définitions ont été données selon plusieurs niveaux d'abstraction allant du niveau purement physique à des niveaux logiques plus élevés. Pour certains, le média est vu comme un dispositif physique ayant le rôle de capteur ou effecteur dans un système informatique. Pour d'autres [BD90], un média n'est pas nécessairement lié uniquement au matériel mais peut être réalisé par un logiciel tel qu'un message électronique par exemple. Pour [Fro91], un média est un système représentationnel nécessitant une interprétation et une compréhension tel qu'un graphe ou un texte en langage naturel.

1.2.2 Mode

Le concept de mode est lié au concept de modalité. En effet, le mode désigne la manière avec laquelle se fait une action et la modalité désigne sa forme. Dans le domaine des IHM, le mode correspond à l'état dans lequel se trouve le système interactif à un instant donné. Il correspond au contexte qui détermine l'interprétation des actions.

Foley et al.[FVC84] définissent le mode par un état ou une collection d'états dans lesquels seul un sous-ensemble de toutes les tâches interactives possibles est disponible.

[Fro91] définit les modes de l'interface par des états à travers lesquels différentes actions de l'utilisateur peuvent produire les mêmes effets. Ainsi, il définit deux modes : langage et Action. Dans le mode action les opérations sont effectuées en s'engageant dans des actions physiques tandis que dans le mode langage, les opérations sont effectuées en utilisant des activités langagières.

1.2.3 Canal

Le concept de canal est lié au concept de média. Il est défini dans [FVC84] par "une interface qui opère une transformation d'énergie". Il fait, ainsi, correspondre à chaque capteur ou effecteur définissant un canal de communication humain, les dispositifs physiques représentant les canaux de l'interface du système chargés de la transformation d'énergie. Par exemple, pour le canal de l'interface "audio", le dispositif correspondant est un microphone.

1.2.4 Modalité

Une modalité est une manière d'interagir avec le système selon une technique. Comme pour le concept de *média*, la modalité est définie selon plusieurs approches : approche orientée utilisateur, approche système ou approche qui mixe les deux.

- Pour l'approche utilisateur, Bersen [Ber94] la définit comme un système représentationnel et la confond ainsi avec le concept média tel qu'il est défini par Alty [Alt91].
- Dans l'approche système nous trouvons les définitions de [Mar95], de [NC96] et [Bou92]. Ces définitions lient la modalité au contenu et à la nature des informations que le système est capable de traiter.
- L'approche mixte est représentée par le modèle "syndésique" de Duke [DBDM94]. Il définit une modalité par les capacités sensorielles et les dispositifs physiques ou logiques engagés dans l'interaction.

Par exemple, [Mar95] définit une modalité comme un processus informatique d'analyse ou de synthèse qui fait coopérer plusieurs processus d'analyse et/ou plusieurs processus de synthèse. Pour [NC96], une modalité est définie par le couple (d, l) où d désigne le dispositif physique et l, un langage d'interaction. Un langage d'interaction est défini par un vocabulaire d'éléments terminaux et une grammaire. Les éléments terminaux sont produits ou captés par les dispositifs d'entrée/sortie.

C'est une définition qui caractérise les échanges entre le système et l'utilisateur car elle identifie et met en relation deux niveaux d'abstraction : le niveau physique (dispositif) et le niveau logique (langage d'interaction).

1.2.5 Fusion et Fission

La *fusion* consiste à combiner les différentes informations provenant de plusieurs médias ou modalités pour former de nouvelles unités d'information. La fusion peut intervenir à différents niveaux d'abstraction. L'exemple le plus illustratif est la fusion de la commande orale "met ça là" avec les clics de la souris.

La *fission* est l'opération inverse de la fusion. Elle consiste à éclater une commande en plusieurs commandes. Un exemple qui explique la fission est la commande "dessiner un cercle dans une fenêtre". La réalisation de cette commande nécessite son éclatement en deux commandes. La première commande est "créer fenêtre", et elle est destinée au gestionnaire des fenêtres. La deuxième commande est 'créer cercle', et elle est destinée à l'éditeur graphique.

En général, l'opération de fusion concerne les interactions en entrée, tandis que l'opération de fission peut concerner aussi bien les interactions en entrée qu'en sortie.

1.2.6 Enoncé

Un énoncé est constitué d'une suite séquentielle d'événements élémentaires multimodaux intervenant dans la réalisation d'une même commande. Une commande est une tâche élémentaire que l'utilisateur veut réaliser. Un énoncé est dit "multi-modal" s'il contient au moins un événement multi-modal et au moins deux événements élémentaires provenant de deux médias distincts. Un événement multi-modal est constitué d'un ensemble d'événements élémentaires provenant de médias distincts, produits dans des voisinages temporels proches et intervenant dans la réalisation d'une même commande. Un événement élémentaire est l'information de base produite par l'interface logicielle associée à un média. La phrase "met ça là" avec un click de la souris constitue un énoncé multi-modal.

1.3 Démarche de modélisation

Dans nos travaux nous ne nous sommes intéressés qu'à la modalité en entrée et donc au processus de fusion. Nous utilisons une approche fondée sur la preuve et le raffinement pour modéliser les IHM3. L'approche proposée consiste à coder un

système de transitions représentant les différents chemins d'interaction permettant d'interagir avec une application.

L'utilisation de B événementiel dans cette étude nous a permis d'identifier plusieurs approches de développement que nous détaillons dans les lots 2 (celui-ci) et le lot 3. Nous avons pris en compte et formalisé différentes notations (en architecture, en modélisation de tâches etc.) utilisées dans la conception des IHM3. Nous nous sommes efforcés à formaliser modèles et propriétés partout où cela nous a paru possible.

Ce rapport est organisé de la façon suivante. Le chapitre 2 présente une approche de modélisation brute de l'interaction multi-modale en entrée. Celle-ci se fonde sur la description de modèles B sans aucune aide méthodologique issue du savoir faire des concepteurs d'IHM3. Le chapitre 3 traite des propriétés d'utilisabilité, en particulier les propriétés CARE. Nous montrons comment ces propriétés sont exprimées et validées avec B. Ensuite, le chapitre 4 traite de la qualité des développements B. Nous prenons en compte l'architecture des IHM3 et montrons comment des abstractions peuvent être réalisées pour permettre de réduire la complexité des développements et des obligations de preuve. Notons que la prise en compte des tâches utilisateur sera abordée dans le Lot 3.

Chapitre 2

Développement d'IHM3 fondé sur la preuve et le raffinement

2.1 Introduction

Nous avons utilisé la méthode B dans sa version événementielle pour modéliser l'interaction multi-modale en entrée. Dans ce chapitre, nous décrivons les principes méthodologiques permettant de représenter cette interaction par une succession de raffinement B sans se soucier des aspects validation de tâches qui seront abordés dans le livrable correspondant au lot 3.

2.2 Description d'une IHM3 avec B

La modélisation d'une IHM3 en B événementiel suit une démarche de conception descendante fondée sur le raffinement. Des événements de haut niveau sont déclarés dans le modèle abstrait racine. Ce dernier est raffiné par l'introduction de nouveaux événements qui précisent le modèle abstrait initial.

L'événement de plus haut niveau décrit l'interaction multi-modale avec le système. Cet événement est décomposé par raffinement en ajoutant différentes informations liées à l'interaction. C'est ainsi que les modalités sont introduites dans les différents raffinements. Elles sont modélisées par des ensembles et des attributs peuvent être associés.

Nous avons appliqué nos travaux aux deux études de cas "MATIS" et "Pages Jaunes", décrites ci-dessous.

Dans ce travail, nous n'avons pas pris en compte l'architecture logicielle utilisée pour la conception de ces applications.

2.3 Etudes de cas

Deux études de cas ont été définies pour la mise en œuvre de la technique formelle fondée sur la preuve.

2.3.1 L'application MATIS

MATIS [Nig94] (Multimodal Airline Travel Information System) est un système d'informations sur le transport aérien développé sur machine NeXT en utilisant l'interface Builder pour la partie graphique et Sphinx, un système de reconnaissance de la parole continue et multilocuteur. MATIS fournit, en réponse à des requêtes de l'utilisateur, des informations sur les vols entre deux villes. La base de données sous-jacente contient actuellement neuf villes américaines, neuf compagnies aériennes et des informations sur chaque vol telles que le numéro du vol, les repas servis à bord etc. L'utilisateur exprime des requêtes au système afin de planifier son voyage. Dans ce but, l'utilisateur a la possibilité entre :

- le langage naturel oral ou écrit, comme "I would like to know the flights from Pittsburgh to Boston serving a meal". De plus, MATIS offre un dialogue élaboré restreint à un domaine sémantique délimité pour lequel l'utilisateur utilise la langue naturelle restreinte : le domaine couvre toutes les informations sur des transports aériens. Ce dialogue est de type question/réponse simple où les demandes et les informations transmises sur un vol sont exprimées en langue naturelle mais où l'enchaînement des requêtes et de négociation n'existent pas ;
- le langage artificiel écrit directement dans les champs d'une requête, comme "PIT" qui est le code de la ville de Pittsburgh.

Par ailleurs, l'utilisateur peut spécifier une requête par manipulation directe avec la souris (métaphore du monde réel). En effet, MATIS offre une interface graphique accessible via la souris qui représente l'univers de l'action par une métaphore du monde réel (cf. figure 2.1). Chaque champ d'une requête fait l'objet d'un formulaire accessible par une fenêtre à l'écran qui présente les outils ; un formulaire permet la spécification par manipulation directe d'une valeur d'un champ particulier d'une requête. Enfin, l'utilisateur peut combiner les langages offerts et la manipulation directe. Par exemple l'utilisateur énonce la phrase "Flights from Pittsburgh to this city" tout en sélectionnant avec la souris la ville de Boston dans la liste des villes. Il peut alors préciser sa demande en saisissant au clavier "1500" dans le champ de l'heure de départ de la requête.

MATIS a été conçue de façon à offrir de nombreuses possibilités à l'utilisateur dans sa tâche de spécification d'une requête. Au travers des exemples ci-dessous nous

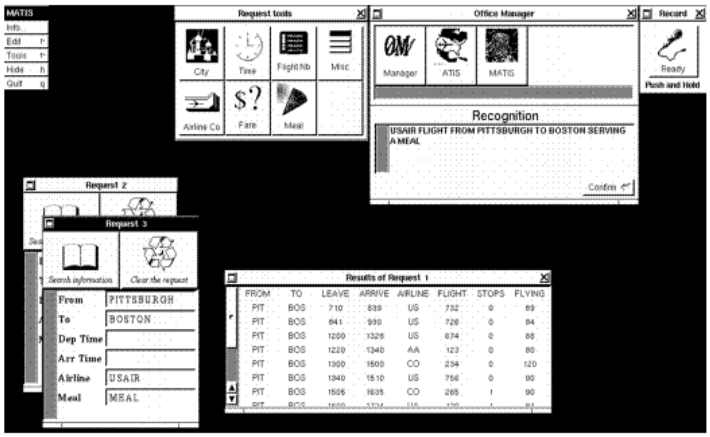


Fig. 2.1 – Copie d'écran du système MATIS

tentons de montrer la multitude de possibilités. Conjugué à un dialogue à plusieurs files d'activité (plusieurs requêtes en cours de spécification) MATIS offre :

1. l'usage synergique des langages et dispositifs que nous illustrons par l'exemple suivant : la phrase orale "Flight from Pittsburgh to this city" accompagnée de la sélection de "Boston" avec la souris ;
2. l'usage synergique des dispositifs uniquement, comme l'exemple suivant : la phrase orale "Flight from Pittsburgh" accompagnée de la saisie au clavier de la phrase "serving a meal and arriving in the afternoon". Le langage utilisé est ici le même ;
3. l'usage alterné des langages et dispositifs : ceci correspond au cas 1 avec séquencement des actions. Par exemple, l'utilisateur articule une phrase puis sélectionne avec la souris ;
4. l'usage alterné des dispositifs : ceci correspond au cas 2 avec séquencement des actions. Par exemple, l'utilisateur commence par articuler le début d'une phrase et la finit par une saisie au clavier ;
5. l'usage concurrent des langages et dispositifs, comme l'exemple suivant : l'utilisateur spécifie une requête en la dictant en langage naturel, tout en complétant une autre requête par saisie directe dans le formulaire. L'usage concurrent permet la spécification de plusieurs requêtes en parallèle ;
6. l'usage concurrent des dispositifs uniquement, comme l'exemple suivant : l'utilisateur spécifie une requête en la dictant en langage naturel, tout en saisissant

au clavier une autre requête en langage naturel ;

7. l'usage exclusif des dispositifs et langages : typiquement MATIS peut être utilisé comme un système vocal uniquement.

2.3.2 L'application Pages Jaunes

L'étude de cas Pages Jaunes CLIPS, que nous appellerons plus simplement Pages Jaunes par la suite dans le document, est développée par l'équipe IIHM du laboratoire CLIPS-IMAG, est une application multi-modale implémentée en utilisant la plate-forme ICARE (Interaction-Care). Notons que nous ne nous sommes intéressés qu'à la multi-modalité en entrée (de l'utilisateur vers le système Pages Jaunes), c'est pourquoi, nous ne détaillons pas la multi-modalité en sortie.

Cette application permet de rechercher un contact en spécifiant le nom (Name) d'une personne et une adresse (Address) de localisation. Suite à l'envoi de la requête de recherche de la personne (Search), un plan est affiché. Sur ce plan, l'utilisateur peut naviguer et cibler sa recherche. L'application Pages Jaunes (cf. figure 2.2) regroupe un ensemble de commandes permettant à l'utilisateur d'accomplir la tâche (spécifier nom, spécifier adresse, envoyer requête, déplacement haut/bas/gauche/droite sur la carte, agrandissement avant/arrière, ...). Il peut déclencher ces commandes en utilisant différentes modalités d'interaction : la voix, la souris et le clavier, ou une combinaison de ces modalités, en s'appuyant sur l'utilisation individuelle et synergique des différentes modalités d'entrées. De cette manière l'utilisateur peut remplir ces champs en utilisant uniquement la voix en prononçant le nom de la personne et le lieu, ou en combinant de façon complémentaire les modalités de la voix, de la souris et du clavier (sélection du champs puis prononciation du nom ou alors en choisissant oralement le champs et en saisissant la valeur par le clavier). Les modalités peuvent également être utilisées de manière redondante dans le cas par exemple où l'utilisateur spécifie par la voix la commande "agrandissement" et en parallèle appuie sur la touche du clavier correspondante à cet effet, une seule commande d'agrandissement sera alors transmise au système.

2.4 Application à la modélisation des études de cas.

Nous montrons dans cette section deux développements par raffinement des deux études de cas présentées précédemment. Notre modélisation ne tient pas compte des différentes notations ou analyses qui ont été réalisées préalablement. Nous avons réalisé ce développement de manière descendante directement à partir du texte dé-

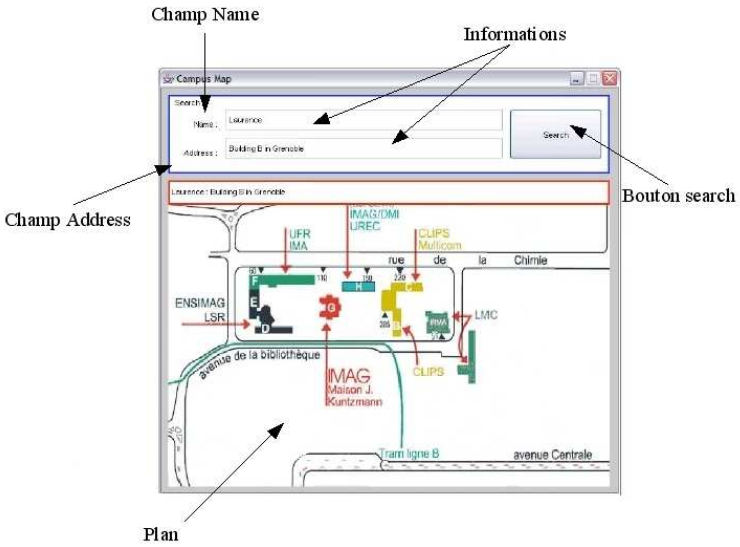


Fig. 2.2 – IHM de l'application Pages Jaunes CLIPS

crivant cette étude de cas et/ou en observant l'exécution du système correspondant puisque celui-ci était disponible.

2.4.1 Modélisation de l'application MATIS

Cette étude a été réalisé dans le cadre des travaux de [AS05]. Le développement complet n'est pas détaillé dans cette section. Nous nous intéressons aux événements les plus important qui décrivent les interactions réalisées entre l'utilisateur et l'application.

Machine abstraite

Deux événements essentiels décrivent ce modèle. $QueryReady$ et $QuerySending$. Le premier événement indique que la requête est prête à être exécutée ($querySending := 1$) alors que le second indique que les résultats de la requête sont prêts ($queryReady := 1$). Les événements décrits ici sont abstraits et aucun détails sur la préparation de la requête et le calcul de ses résultats (cf. figure 2.3).

VARIABLES
$queryReady, querySending, ...$
INVARIANT
$queryReady \in \{0,1\} \wedge querySending \in \{0,1\} \wedge ...$
$queryReady \neq querySending$
...
EVENTS
...
$QueryReady$ = **SELECT**
$queryReady = 1 \wedge$
...
THEN
$queryReady := 0 \parallel ...$
$querySending := 1 \parallel ...$
...
END ;

$QuerySending$ = **SELECT**
$querySending = 1$
THEN
$queryReady := 1 \parallel$
$querySending := 0 \parallel$
...
END

FIG. 2.3 – Machie abstraite de MATIS

Raffinement

Ce raffinement a pour objectif d'expliciter le déroulement de la préparation de la requête et le calcul des résultats en rajoutant de nouvelles variables et de nouveaux événements à la machine abstraite précédente. L'événement *QueryReady* est raffiné par quatre événements qui servent à récupérer les paramètres de la requête que sont la ville de départ ($Evt_{InputDeparture}$), la ville d'arrivée ($Evt_{InputArrival}$), l'heure du départ ($Evt_{HourDeparture}$) et l'heure d'arrivée ($Evt_{HourArrival}$). L'événement *QuerySending* qui modélise le comportement du noyau fonctionnel de l'application est quand à lui raffiné par deux événements ($QueryOK$ et $QueryKO$) qui calculent les résultats de la requête lancée avec les paramètres précédents (cf. figure 2.4).

Pour les paramètres de la requête, nous observons le comportement de l'événement du choix de la ville de départ à travers l'événement $Evt_{InputDeparture}$ au moyen des trois modalités disponible dans le système (la voix, la souris et le clavier). Les variables $MouseModal$, $KeyboardModal$ et $SpeechModal$ du modèle B événementiel sont associées à ces trois modalités. Cet événement est déclanché par une action de l'utilisateur au travers des différentes modalités (exprimées dans la garde de l'événement) et modifie la variable *PlaceDeparture*. Cette dernière per-

SETS
$State = \{on, off\}$;
$VOLS$
CONSTANTS
vol,
$townDeparture, townArrval, timeDeparture, timeArrival$
PROPERTIES
$vol \in NAT \rightarrow VOLS \wedge$
$townDeparture \in VOLS \rightarrow 1..9 \wedge townArrival \in VOLS \rightarrow 1..9 \wedge$
$timeDeparture \in VOLS \rightarrow 0..23 \wedge timeArrival \in VOLS \rightarrow 0..23$
VARIABLES
$PlaceDeparture, ...$
$MouseModal, KeyboardModal, SpeechModal, ...$
$queryReady, querySending, ...$
$result, index, ...$
INVARIANT
$PlaceDeparture \in 1..9 \wedge ...$
$MouseModal \in State \wedge KeyboardModal \in State \wedge SpeechModal \in State \wedge ...$
$queryReady \in \{0,1\} \wedge querySending \in \{0,1\} \wedge ...$
$index \in NAT...$
$result \subseteq VOLS \wedge ...$
...
EVENTS
$Evt_{InputDeparture}$ = **SELECT**
$Evt1 = 1 \wedge$
$(MouseModal = on \vee KeyboardModal = on \vee SpeechModal = on)$
THEN
$PlaceDeparture :\in 1..9 \parallel Evt1 := 0$
END ;
...
$QueryOK$ = **SELECT**
$querySeending = 1 \geq 0 \wedge index \geq 0 \wedge$
$townDeparture(vol(index)) = PlaceDeparture \wedge townArrival(vol(index)) = PlaceArrival \wedge$
$timeDeparture(vol(index)) = HourDeparture \wedge timeArrival(vol(index)) = HourArrival$
THEN
$result := result \cup \{vol(index)\} \parallel$
$index := index - 1$
END ;

$QueryKO$ = **SELECT**
$querySeending = 1 \geq 0 \wedge index \geq 0 \wedge$
$NOT(townDeparture(vol(index)) = PlaceDeparture \wedge townArrival(vol(index)) = PlaceArrival \wedge$
$timeDeparture(vol(index)) = HourDeparture \wedge timeArrival(vol(index)) = HourArrival)$
THEN
$index := index - 1$
END ;

...

Fig. 2.4 – Raffinement du modèle abstrait de MATIS

met de sauvegarder la valeur de la ville de départ (9 villes numérotées de 1 à 9). Enfin la variable $Ev1$ est utilisée pour jouer le rôle de variant assurant ainsi un déclanchement ordonné des différents événements que décrivent le modèle B événementiel. Dans cet événement, il y a des variables issues de la partie présentation ($MouseModal$, $KeyboardModal$ et $SpeechModal$) et des variables issues de la partie noyau fonctionnel ($PlaceDeparture$). Il existe trois autres événements ($Evt_{InputArrival}$, $Evt_{HourDeparture}$ et $Evt_{HourArrival}$) pour la récupération des autres paramètres de la requête ($PlaceArrival$, $HourDeparture$ et $HourArrival$). La garde de l'événement $QueryReady$ est renforcée par l'expression ($Ev1 = 0 \wedge Ev2 = 0 \wedge Evt3 = 0 \wedge Evt4 = 0$) qui indique que tous les paramètres de la requête sont saisies.

Le noyau fonctionnel permet de calculer les résultats de la requête à travers les événements $QueryOK$ et $QueryKO$ qui parcourent l'ensemble des vols (avec comme index la variable $index$ initialisée à la taille de la table des vols de la BDD vol) et de renvoyer comme résultat les vols correspondants aux critères de la requête dans la variable $result$ (les conditions de la requête sont exprimées dans la garde de l'événement $QueryOK$). $townDeparture()$, $townArrval()$, $timeDeparture()$ et $timeArrival()$ sont des fonctions qui associent pour chaque vol, la ville de départ, la ville d'arrivée, l'heure de départ et l'heure d'arrivée. Enfin, l'événement $QuerySending$ indique que le résultat de la requête est prêt ($queryReady := 1$). Ce dernier événement est déclenché à la fin du parcours de l'ensemble des vols ($index = 0$).

Bilan : preuves de propriétés

Le Tableau 2.1 résume les résultats obtenus sur l'étude de cas. 94 obligations de preuves (OP) sont générées. Elles ont été toutes prouvées avec le prouveur automatique, sans aucune intervention de la part du concepteur. Ce qui nous permet de conclure que 100% des obligations de preuves ont été démontrées automatiquement.

Modèle	Obv	OP	Auto	Interactif	%Pr
$MATIS$	184	18	18	0	100
$MATIS\_ref\_1$	576	76	76	0	100
$Total$	760	94	94	0	100

TAB. 2.1 – Bilan des preuves pour l'étude de cas MATIS

2.4.2 Modélisation de l'application Pages Jaunes

Trois modèles ont été définis. Ils sont reliés par une relation de raffinement. Le premier introduit les événements de haut niveau. Le deuxième présente les événements de la présentation. Enfin, le troisième introduit les interactions multi-modales avec les différentes possibilités d'interaction. Dans la suite, nous décrivons une représentation simplifié du développement de l'IHM3 associée à l'étude de cas Pages Jaunes. Comme pour MATIS, nous avons volontairement réduit les modèles B événementiel afin de rendre la lecture plus aisée.

Le modèle racine

Nous ne décrivons que les événements liés à la multi-modalité en entrée, c'est pourquoi, les événements liés à la sortie ne sont pas présents dans ces développements.

MODEL $IMAPPY$
VARIABLES
$Query\_sending, Query\_ready, Nn, Na$
INVARIANT
$Query\_sending \in 0..1 \wedge Query\_ready \in 0..1 \wedge$
$Query\_sending \neq Query\_ready$
ASSERTIONS
$(Query\_ready = 1) \vee (Query\_sending = 1)$
INITIALISATION
$Query\_sending, Query\_ready := 0, 1$

EVENTS
$Query =$
SELECT $Query\_ready = 1$
THEN
$Query\_sending := 1 \parallel$
$Query\_ready := 0$
END ;

$Result\_query =$
SELECT $Query\_sending = 1$
THEN
$Query\_sending := 0 \parallel Query\_ready := 1$
END
END

Fig. 2.5 – Modèle racine représentant l'application Pages Jaunes

Deux événements $Query$ et $Result\_query$ décrivent ce modèle. Le premier indique ($Query\_sending := 1$) que la requête est prête a être envoyée alors que le second indique que le résultat de la requête est prêt ($Query\_ready := 1$). Les deux

variables $Query\_sending$ et $Query\_ready$ décrivent la synchronisation de ces deux événements. Notons que les événements décrits ici sont abstraits (aucun élément relatif aux différentes modalités d'interaction) et n'indiquent que l'ordre correct de déclenchement. Dans la suite de la présentation, nous nous intéressons au raffinement de l'événement $Query$ qui décrit la saisie du nom et de l'adresse selon différentes modalités.

Identification des événements de la présentation

L'événement $Query$ est décomposé de sorte à prendre en compte les différentes possibilités de saisie du nom et de l'adresse de la personne à rechercher. Ainsi, on pourra :

- saisir le nom un nombre arbitraire de fois ;
- saisir l'adresse un nombre arbitraire de fois ;
- lancer ensuite la recherche.

Notons que l'on peut entrelacer la saisie du nom et de l'adresse (saisir l'un puis l'autre dans n'importe quel ordre et retour).

Nous nous concentrons sur les événements introduits pour raffiner les deux événements principaux du niveau abstrait.

REFINEMENT $IMAPPY\_Ref\_1$
REFINES $IMAPPY$
SETS
$string$
VARIABLES
$Query\_sending, EV1, Nn, Na, map, name, address$
INVARIANT
$EV1 \in 0..3 \land Nn \in \mathbb{N} \land Na \in \mathbb{N} \land$
$map \in BOOL \land name \subseteq string \land address \subseteq string \land$
$(Query\_ready = 1 \Rightarrow EV1 \neq 0)$
...

FIG. 2.6 – Modèle racine représentant l'application Pages Jaunes

Nous avons introduit de nouvelles variables pour la description de l'interaction en entrée. Tout d'abord, nous décrivons les variables $name$ et $address$ de type string qui récupèrent respectivement le nom et l'adresse. Les variables entières Nn et Na désignent respectivement le nombre de fois où l'utilisateur va entrer un nom et le nombre de fois où il va entrer une adresse. Enfin, la variable EV1 (variable entière décroissante) initialisée à 3 est un variant qui décrit l'ordre de déclenchement de chaque événement.

EVENTS
Name_Address =
SELECT $EV1 = 3$
THEN
$EV1 := EV1 - 1 \parallel Nn :\in \mathbb{N} \parallel Na :\in \mathbb{N}$
END ;

Input_Name =
SELECT $EV1 = 2 \wedge Nn \neq 0$
THEN
$Nn := Nn - 1 \parallel name :\in \mathbb{P}(string)$
END ;

Input_Address =
SELECT $EV1 = 2 \wedge Na \neq 0$
THEN
$Na := Na - 1 \parallel address :\in \mathbb{P}(string)$
END ;

Search =
SELECT $EV1 = 2 \wedge Nn = 0 \wedge Na = 0$
THEN
$EV1 := 1$
END ;

Query =
SELECT $EV1 = 1$
THEN
$Query\_sending, EV1 := 1, 0$
END ;
END

Fig. 2.7 – Événements Raffinant l'événement *Query*

Les événements de la figure 2.7 décrivent les nouveaux événements correspondant aux interactions de la présentation. L'événement $Name\_Address$ initialise itérateurs Nn et Na (grâce à l'opérateur $:\in$ qui considère un élément quelconque dans un ensemble) alors que les trois événements $Input\_Name$, $Input\_Address$ et $Search$ décomposent (raffinent) l'événement abstrait $Query$. Lorsque ces trois événements ont été déclenchés, ils rendent le contrôle à l'événement abstrait $Query$ qui ne fait que le transmettre aux autres événements. Notons qu'à ce niveau, nous avons pris en compte les déclenchements des interactions sans entrer dans les détails de leurs définitions. Cela sera abordé dans le prochain raffinement.

Identification des modalités

REFINEMENT $IMAPPY\_ref\_2$
REFINES $IMAPPY\_ref\_1$
SETS
$HCI = \{NONE, NAME, INPUT, SEARCH, ZOOM\_IN, ZOOM\_OUT, HERE,$
$INPUT\_TEXT, ENTER\_KEY, LEFT\_ARROW, RIGHT\_ARROW,$
$CLICK\_ON\_NAME, CLICK\_ON\_ADDRESS, CLICK\_ON\_SEARCH,$
$CLICK\_ON\_MAP, UP\_ARROW, DOWN\_ARROW\}$
CONSTANTS
$SPEECH, KEYBOARD, MOUSE$
PROPERTIES
$SPEECH \subseteq HCI \wedge KEYBOARD \subseteq HCI \wedge MOUSE \subseteq HCI \wedge$
$SPEECH = \{NAME, INPUT, SEARCH, ZOOM\_IN, ZOOM\_OUT, HERE\} \wedge$
$MOUSE = \{CLICK\_ON\_NAME, CLICK\_ON\_ADDRESS,$
$CLICK\_ON\_SEARCH, CLICK\_ON\_MAP\} \wedge$
$KEYBOARD = \{INPUT\_TEXT, ENTER\_KEY, LEFT\_ARROW,$
$RIGHT\_ARROW, UP\_ARROW, DOWN\_ARROW\}$
ABSTRACT_VARIABLES
$modality, EV11, ...$
INVARIANT
$modality \in HCI \wedge$
$EV11 \in 0..2 \wedge ...$
ASSERTIONS
...
INITIALISATION
$modality := NONE \parallel EV11 := 2 \parallel ...$

FIG. 2.8 – Introduction des différentes interactions multi-modales

La dernière étape de raffinement consiste à identifier et à introduire les différentes interactions multi-modales en entrée mises en jeu (cf. figure 2.8 et figure 2.9). Ainsi, on décrit les ensembles $SPEECH$, $KEYBOARD$ et $MOUSE$ indiquant et typant les différentes interactions multi-modales possibles. Notons que ces ensembles peuvent être enrichis ou réduits sans remettre en cause la modélisation effectuée. La

variable *modality* décrit la modalité activée à un instant donné. Elle est initialisée à NONE. La variable EV11 quant à elle indique que le focus est sur la saisie du nom.

Nous procédons à la décomposition (raffinement) de l'événement $Input\_Name$ et $Input\_Address$ qui feront intervenir les modalités voix et clavier/souris. Pour cela, il est nécessaire d'enrichir le modèle B par la déclaration de nouveaux événements. Le nom est entrée en deux étapes : saisie du champ à saisir (le nom) par la voix ou le clavier/souris puis entrée du nom (valeur du nom) par la voix ou bien par clavier/souris. Ces deux événements peuvent être itérés un nombre arbitraire de fois. Les suffixes $field$ et $value$ indiquent respectivement les événements associés au choix du champ à remplir et à la valeur entrée pour ce champ. Dans la suite, seuls les événements raffinant l'événement $Input\_Name$ sont présentés.

À ce niveau, il faut observer que tous les événements peuvent être déclenchés. C'est-à-dire, qu'il est possible de saisir le nom par la voix ou par le clavier/souris uniquement, ou bien par une combinaison des deux.

EVENTS
$Name\_address = ...$

$Input\_Name\_Field =$
SELECT $EV1 = 2 \wedge Nn \neq 0 \wedge EV11 = 2$
THEN
$EV11 := EV11 - 1 \parallel$
$modality :\in \{CLICK\_ON\_NAME, NAME\}$
END ;

$Input\_Name\_Value =$
SELECT $EV1 = 2 \wedge Nn \neq 0 \wedge EV11 = 1$
THEN
$EV11 := EV11 - 1 \parallel$
$modality :\in \{INPUT, INPUT\_TEXT\} \parallel$
$name :\in P(string) \parallel$
$Nn := Nn - 1$
END ;

$Input\_Name =$
SELECT $EV1 = 2 \wedge Nn \neq 0 \wedge EV11 = 0$
THEN
$EV11 := 2$
END ;

$Input\_address = ...$
$Search = ...$
$Query = ...$
$Result\_query = ...$
END

Fig. 2.9 – Événements raffinant l'événement $Input\_Name$

Bilan : preuves de propriétés

Le Tableau 2.2 ci-dessous illustre les résultats obtenus sur l'étude de cas. 59 obligations de preuve sont automatiquement générées (OP). Seules 3 d'entre elles ont nécessité l'utilisation d'une preuve interactive i.e. le concepteur prend en main la preuve. Dans notre cas, le concepteur n'a fourni aucun effort puisque celui-ci s'est contenter de faire appel au prouveur de prédicats. Ce prouveur uniquement disponible dans l'environnement de preuve interactive est néanmoins entièrement automatique. Ces 3 obligations de preuves ayant été démontrées uniquement par l'utilisation du prouveur de prédicats. Nous pouvons donc conclure que 100% des preuves ont été démontrées automatiquement.

Modèle	Obv	OP	Auto	Interactif	%Pr
$IMAPPY$	55	10	10	0	100
$IMAPPY\_ref\_1$	176	22	22	0	100
$IMAPPY\_ref\_2$	142	27	24	3	100
$Total$	373	59	56	3	100

TAB. 2.2 – Bilan des preuves pour l'étude de cas Pages Jaunes.

2.5 Conclusion

Les différents développements menés à ce stade ont été obtenus à partir d'une modélisation directe des applications correspondant aux études de cas. Nous ne nous sommes pas intéressés à la complexité des développements et des preuves. Ce point est étudié dans le chapitre 4 par l'étude de différents scénarios de raffinement et d'abstraction du noyau fonctionnel et de modularisation des développements. La modélisation des tâches utilisateur permettant d'appréhender les comportements des utilisateurs lors de leur interaction avec le système n'a également pas été abordée, elle sera traitée dans le lot 3.

Nous aborderons dans le chapitre suivant la validation des propriétés d'utilisabilité selon différentes approches.

Chapitre 3

Modélisation de propriétés d'utilisabilité des d'IHM3 fondée sur la preuve et le raffinement

3.1 Introduction

Parmi les objectifs de la modélisation d'IHM3 présentée dans le chapitre précédent, la vérification de propriétés d'utilisabilité constitue une préoccupation majeure. En effet, ce type de propriétés est souvent validé par l'expérimentation et par le test. Nous proposons d'étudier la modélisation et la vérification de ce type de propriétés à l'aide de la preuve et du raffinement.

Ce chapitre est organisé de la façon suivante. Nous présentons dans un premier temps une formalisation générale des propriétés CARE considérant le temps puis nous montrons comment ces propriétés ont été formalisées et vérifiées à l'aide du raffinement et de la preuve. Plusieurs approches de vérification sont présentées et illustrées sur l'étude de cas MATIS.

3.2 Vérification de propriétés

En plus des propriétés de validité et de robustesse que l'on considère en génie logiciel, la conception d'IHM en général et d'IHM3 en particulier nécessite la prise en considération des propriétés permettant de caractériser ces interfaces homme-machine. C'est pourquoi une autre catégorie de propriétés a été introduite : les propriétés d'utilisabilité comme l'insistance, l'observabilité ou l'honnêteté ont été

définies.

Nous assistons ces dernières années à l'apparition de nouveaux types d'interfaces plus complexes en termes d'interaction et de présentation. Pour ce type d'IHM, une catégorie particulière de propriétés a été définie dans le cas des IHM3. Il s'agit des propriétés CARE qui font l'objet de l'étude que nous présentons ci-dessous.

3.3 Les propriétés CARE

Les propriétés CARE [CNS+95], définies initialement dans l'espace TYCOON [MB, Mar97], ont servi de base pour définir les propriétés d'utilisabilité des interfaces multimodales. Elles définissent les différentes manières avec lesquelles l'utilisateur peut combiner les modalités pour réaliser ses tâches. Elles sont définies comme suit :

1. **Complémentarité :** désigne l'usage de plusieurs modalités pour la réalisation d'un but.
2. **Assignation :** désigne l'usage exclusif d'une seule modalité pour la réalisation d'un but, et aucune autre modalité ne permet de le réaliser.
3. **Equivalence :** désigne le choix offert à l'utilisateur pour réaliser son but avec une modalité de son choix parmi un ensemble de modalités permettant de le réaliser.
4. **Redondance :** désigne l'usage de plusieurs modalités pour le même but, et d'une manière parallèle.

Eléments de base

Dans [CNS+95], les auteurs définissent les propriétés CARE et donnent une expression formelle pour chacune d'elles. Cette expression formelle repose sur les concepts d'état, de but, de modalité, et de relation temporelle qu'ils définissent comme suit :

- Un *état* est un ensemble d'attributs mesurables à un instant donné et qui caractérisent une situation.
- Un *but* est un état qu'un agent veut atteindre. Un agent peut être le système ou l'utilisateur.
- Une *modalité* est une méthode d'interaction qu'un agent utilise pour atteindre un but. La fonction $Reach(s, m, s')$ exprime la capacité de la modalité m de permettre à un agent d'atteindre l'état s' à partir d'un état s en une seule étape. Une suite d'états successifs est appelé *trajectoire d'interaction.*

– Une *relation temporelle* caractérise l'utilisation des modalités à travers le temps. L'utilisation de ces modalités peut être simultanée ou en séquence dans une fenêtre temporelle définie par un intervalle de temps.

Comportements séquentiel et parallèle

A partir des notions définies dans la section précédente, Il est possible de définir les comportements parallèle et séquentiel des modalités permettant de décrire les propriétés CARE d'un système multimodal à états.

Soit M un ensemble de modalités. Ces modalités peuvent être utilisées en parallèle si, dans une fenêtre temporelle, elles sont actives en même temps. L'utilisation parallèle des modalités d'un ensemble M dans une fenêtre temporelle tw est définie par :

$$\boxed{\begin{array}{l} Parallel(M, tw) \Leftrightarrow (Card(M) > 1) \wedge (Duration(tw) \neq \infty) \wedge \\ (\exists t \in tw, \forall m \in M \ Active(m, t)) \end{array}}$$

Avec :
– $Active(m, t)$ est un prédicat qui exprime que m est active à l'instant t.
– $Card(M)$ est le nombre de modalités dans l'ensemble M ;
– $Duration(tw)$ est la durée de l'intervalle du temps tw.

Les modalités de l'ensemble M sont utilisées en séquence dans une fenêtre temporelle tw si au plus, une seule modalité est active à un instant donné et si toutes les modalités de l'ensemble M sont utilisées dans la fenêtre temporelle tw. L'utilisation séquentielle des modalités d'un ensemble M dans une fenêtre temporelle tw est définie par :

$$\boxed{\begin{array}{l} Sequential(M, tw) \Leftrightarrow (Card(M) > 1) \wedge (Duration(tw) \neq \infty) \wedge \\ (\forall t \in tw(\forall m, \ m' \in M \ Active(m, t) \Rightarrow \\ \neg Active(m', t)) \wedge (\forall m \in M, \exists t \in tw \ Active(m, t)) \end{array}}$$

Définition formelle des propriétés CARE

Les propriétés CARE peuvent caractériser la relation entre les états et les modalités en quatre types :

Equivalence. Les modalités de l'ensemble M sont dites équivalentes pour atteindre l'état s' à partir de l'état s, s'il est nécessaire et suffisant d'utiliser une de

ces modalités. L'ensemble M doit contenir au moins deux modalités. Cette définition est donnée formellement par :

$$\boxed{\begin{array}{l} Equivalence(s, M, s') \Leftrightarrow \\ (card(M) > 1) \wedge (\forall m \in M \ Reach(s, m, s')) \end{array}}$$

Cette propriété exprime la possibilité de choisir entre plusieurs modalités sans aucune contrainte temporelles entre elles.

Assignation. Une modalité m est dite assignée à l'état s pour atteindre l'état s', si aucune autre modalité ne permet d'atteindre l'état s' à partir de s. Contrairement à l'équivalence, l'assignation exprime l'absence de choix. L'équivalence

$$\boxed{\begin{array}{l} Assignement(s, m, s') \Leftrightarrow Reach(s, m, s') \wedge \\ (\forall m' \in M Reach(s, m, s') \Rightarrow m' = m) \end{array}}$$

et l'assignation expriment le choix ou l'absence de choix de modalités pour atteindre un but. L'assignation ne dépend pas de la fenêtre temporelle.

Redondance. Les modalités de l'ensemble M sont utilisées de manière redondante pour atteindre l'état s' à partir de l'état s, si elles sont équivalentes et si elles sont toutes utilisées dans la même fenêtre temporelle. Elles sont décrites formellement par :

$$\boxed{\begin{array}{l} Redundancy(s, M, s', tw) \Leftrightarrow Equivalence(s, M, s') \wedge \\ (Sequential(M, tw) \vee Parallel(M, tw)) \end{array}}$$

Complémentarité. Les modalités de l'ensemble M sont dites complémentaires pour atteindre l'état s' à partir de l'état s dans une fenêtre temporelle tw, si elles sont toutes utilisées pour atteindre l'état s' à partir de s et aucune modalité ne permet à elle seule d'atteindre s'.

$P(M)$ désigne l'ensemble de partitions de M.

$Reach(s, M, s') \Leftrightarrow \forall m \in M, Reach(s, m, s')$, ce qui signifie que l'état s' peut être atteint à partir de l'état s en utilisant les modalités de l'ensemble M.

Ces propriétés permettent de mesurer la flexibilité et la robustesse des IHM multimodales. En effet, l'équivalence est une propriété qui définit aussi bien la flexibilité en offrant le choix à l'utilisateur que la robustesse de l'IHM. En cas de panne d'un dispositif l'interface continue à être utilisable. L'assignation, et contrairement à l'équivalence est restrictive. Elle ne permet de réaliser une tâche donnée qu'avec la

$$Complementarity(s, M, s', tw) \Leftrightarrow (Card(M) > 1) \wedge (Duration(tw) \neq \infty) \wedge$$
$$(\forall M' \in P(M)(M' \neq M \Rightarrow$$
$$\neg Reach(s, M', s'))) \wedge Reach(s, M, s') \wedge (Sequential(M, tw) \vee Parallel(M, tw))$$

modalité qui lui est assignée ce qui engendre la non utilisabilité de l'interface dans le cas où le média correspondant est en panne. La redondance permet d'augmenter la robustesse de l'IHM, mais charge le système. La complémentarité permet la flexibilité de l'interaction mais risque de générer des problèmes de synchronisation et d'interprétation cognitives.

3.4 Expression et Vérification de propriétés CARE avec B

Assurer l'utilisablité d'un système interactif multimodal revient à valider le modèle de ce système contenant essentiellement les expressions de cette propriété. Nous avons abordé dans le premier chapitre l'aspect définissant les propriétés d'utilisabilité d'IHM3. Elles sont représentés par les propriétés CARE (complémentarité, assignation, redondance, équivalence) dont les définitions ont été données à la section 3.3 page 22.

Dans cette section nous donnons trois manières différentes pour exprimer et vérifier ces propriétés dans une spécification d'un modèle d'un système interactif multimodal codé en B événementiel ([AS05, AAASBM06]). Ces trois approches se fondent sur :

- la description de ces propriétés par les gardes des événements ;
- la description de ces propriétés par un modèle de tâches CTT ;
- l'expression des propriétés par de nouveaux événements intégrés au modèle.

Nous montrons sur l'étude de cas MATIS comment ces propriétés sont vérifiées à l'aide de B événementiel.

3.4.1 Description par les gardes des événements

Pour exprimer les propriétés CARE, nous enrichissons les gardes des événements sous forme de prédicats manipulant les attributs de l'IHM. Les pré-conditions associés doivent exprimer conjointement les quatre propriétés.

Pour mieux illustrer notre démarche, nous considérons l'événement *Evt* à réaliser, cet événement représente la tâche à réaliser par l'utilisateur. Pour la réalisation

de cette tâche, deux modalités sont disponibles : $m1$ et $m2$. L'événement Evt est donc raffiné en deux nouveaux événements : $Evtm1$ réalisant la même tâche que l'événement Evt en utilisant la modalité $m1$, et $Evtm2$ réalisant la tâche de l'événement Evt avec la modalité *m2*.

$Evt =$
SELECT
$G1 \ \wedge$
$((m1 = on \wedge m2 = on) \ \vee$
$(m1 = on \wedge m2 = off) \ \vee$
$(m1 = off \wedge m2 = on))$
THEN
$S1$
END
...

$Evtm1 =$	$Evtm2 =$
SELECT	**SELECT**
$G11 \wedge m1 = on$	$G12 \wedge m2 = on$
THEN	**THEN**
$S11$	$S12$
END	**END**

Interprétation des gardes de l'événement *Evt* avec les propriétés CARE.

Complémentaire et/ou Redondance. Si les deux modalités $m1$ et $m2$ sont activées et les gardes de $Evtm1$ et $Evtm2$ sont vérifiées, on se retrouvera dans deux cas possibles : on dit qu'il y a *redondance* si $Evtm1$ et $Evtm2$ peuvent se déclencher en même temps et aboutir à la réalisation du même but. Il y aura ainsi deux événements qui rendront la main à l'événement de l'abstraction ; on dit qu'il y a *complmentarit* si $Evtm1$ et $Evtm2$ se complètent pour réaliser le même but.

$((\mathbf{m1 = on} \wedge \mathbf{m2 = on}) \vee$
$(m1 = on \wedge m2 = off) \vee$
$(m1 = off \wedge m2 = on))$

Assignation et/ou Equivalence. Si une des deux modalités $m1$ ou $m2$ est activée et si une des gardes de $Evtm1$ et $Evtm2$ est vérifiée, on se retrouvera dans le cas ou un unique événement est déclenché. Dans ce cas une seule modalité est activée donc un seul événement ($Evtm1$ et $Evtm2$) rend la main à l'abstraction. On dit qu'il y a *assignation* de la tâche Evt a une seule modalité seulement. Par ailleurs, il y a *quivalence* entre $m1$ et $m2$ dès lors que l'événement de l'abstraction possède la main à partir de l'événement $Evtm1$ ou $Evtm2$.

$((m1 = on \wedge m2 = on) \vee$	$((m1 = on \wedge m2 = on) \vee$
$(\mathbf{m1 = on} \wedge \mathbf{m2 = off}) \vee$	$(m1 = on \wedge m2 = off) \vee$
$(m1 = off \wedge m2 = on))$	$(\mathbf{m1 = off} \wedge \mathbf{m2 = on}))$

Conclusion. Dans cette approche, les propriétés qui sont facilement exprimables sont *la redondance* et *l'équivalence*. Cependant *la complémentarité* est difficilement visible, elle se confond avec la propriété de *redondance*. En effet, nous n'avons pas assez d'informations sur les tâches exécutées par les événements $Evtm1$ et $Evtm2$.

3.4.2 Description par modèle de tâches

Cette approche consiste à définir des scénarios sous formes de tâches CTT traduites par des modèles B.Les scénarios CTT doivent faire apparaître les propriétés CARE et doivent aussi valider un modèle B.

Pour mieux illustrer cette démarche, nous proposons de construire les différents modèles autour d'une tâche T à réaliser par l'utilisateur. La tâche T correspond aux déclenchement de deux événements $EV1$ et $EV2$ en séquence.

$$T = EV1 >> EV2$$

Nous proposons de réaliser la tâche T par deux modalités différentes $m1$ et $m2$. L'événement $EV1$ est raffiné en deux événements : $EV1m1$ qui réalise l'événement $EV1$ en activant la modalité $m1$ et $EV1m2$ qui réalise l'événement $EV1$ en activant la modalité $m2$. L'événement $EV2$ est également raffiné en deux événements : $EV2m1$ pour la réalisation de $EV2$ avec la modalité $m1$ et $EV2m2$ pour la réalisation de $EV2$ avec la modalité $m2$. Ainsi la tâche T sera raffinée en *Tref* suivante :

$$Tref = (EV1m1 [] EV1m2) >> (EV2m1 [] EV2m2)$$

Vérification de l'équivalence et de l'assignation. Pour mieux montrer la propriété d'*assignation*, la tâche *Tref* est raffinée à son tour en $Tref1$ qui est la même tâche que T mais réalisée seulement avec la modalité $m1$. *Tref* peut être aussi raffinée en $Tref2$ qui est la même tâche mais réalisée avec la modalité $m2$. Comme $Tref1$ et $Tref2$ réalisent la même tâche *Tref*, on conclut l'équivalence des deux modalités $m1$ et $m2$. L'*assignation* est vérifiée par ce que *Tref* peut être raffinée soit en $Tref1$ soit en $Tref2$ pour être réalisée par une seule modalité.

$$Tref1 = (EV1m1) >> (EV2m1) \quad ou \quad Tref2 = (EV1m2) >> (EV2m2)$$

Vérification de la redondance. Pour vérifier la *redondance*, la tâche *Tref* est raffinée en $Tref11$. Cette dernière consiste à réaliser la tâche *Tref* avec les deux modalités $m1$ et $m2$ en parallèle. Dans ce cas les deux événements $EV1$ et $EV2$ constituant la tâche T sont déclenchés par les deux modalités pour chacun. Nous parlons de *redondance* d'information au niveau des événements $EV1$ et $EV2$ vue qu'ils sont réalisés par les deux modalités en parallèle.

$$Tref11 = (EV1m1 >> EV2m1) \parallel (EV1m2 >> EV2m2)$$

Vérification de la complémentarité. Nous proposons de raffiner la tâche *Tref* soit en $Tref12$, soit en $Tref21$. $Tref12$ réalise la tâche *Tref* avec les deux modalités, mais en les combinant de manière à réaliser l'événement $EV1$ avec la modalité $m1$ et l'événement $EV2$ avec la modalité $m2$. $Tref21$ réalise la tâche *Tref* mais avec une autre combinaison, $EV1$ est déclenché avec la modalité $m2$ et $EV2$ est déclenché par la modalité $m1$. Dans les deux cas, nous parlons de *complémentarité* entre les deux modalités $m1$ et $m2$ pour la réalisation de la tâche *Tref*.

$$Tref12 = (EV1m1) >> (EV2m2) \quad ou \quad Tref21 = (EV1m2) >> (EV2m1)$$

Conclusion. Faire apparaître les propriétés CARE par des modèles de tâches offre la possibilité de vérifier toutes les caractéristiques d'utilisabilité d'une IHM3.

3.4.3 Enrichissement par des événements exprimant les propriétés

Le principe de cette approche est d'ajouter un raffinement au niveau d'un modèle en B événementiel contenant des événements dont la garde est l'expression d'une propriété CARE et un invariant garantissant la vérification de cette propriété. Cette approche permet de vérifier les propriétés au niveau de la spécification, à la fin d'un événement ou à l'exécution d'une tâche bien précise. En d'autres termes, elle permet de placer des points d'observation pour les propriétés CARE.

Pour illustrer cette approche à travers des exemples, nous considérons l'événement *Evt* réalisant la tâche T avec deux modalités $m1$ et $m2$. *Evt* est raffiné en deux événements $Evtm1$ réalisant la tâche T avec la modalité $m1$ et $Evt2$ réalisant la tâche T avec la modalité $m2$.

Nous définissons deux variants $v1$ et $v2$ initialisés à la valeur N. Ces deux variants sont décrémentés à chaque fois que l'une des deux modalités est activée.

$Evtm1 =$	$Evtm2 =$
SELECT	**SELECT**
$G11 \wedge m1 = on$	$G12 \wedge m2 = on$
THEN	**THEN**
$S11 \parallel v1 := v1 - 1$	$S12 \parallel v2 := v2 - 1$
END	**END**

Vérification de la redondance et/ou complémentarité. Nous proposons de vérifier la propriété de *redondance* à la fin de l'exécution de l'événement *Evt*. L'événement *EvtRedondance* peut être vu comme un point d'observation de la propriété

de *redondance* qui est déclenché lorsque la propriété est vérifié à la fin de l'événement *Evt*. L'invariant assure aussi que la propriété est toujours vérifiée lorsque cet événement est déclenché. La *redondance* apparaît par le fait que les événements du raffinement $Evtm1$ et $Evtm2$ se réalisent en même temps et aboutissent à la réalisation du même but.

INVARIANT
...
$(redondance = TRUE) \Rightarrow (v1 = 0 \wedge v2 = 0)$
EVENTS
...
$EvtRedondance =$
SELECT
$(v1 = 0 \wedge v2 = 0)$
THEN
$redondance := TRUE$
END

Pour le modèle que nous proposons, la *redondance* est assurée par le passage des deux variants $v1$ et $v2$ à la valeur **0**. Cela assure le déclenchement des deux événements $Evtm1$ et $Evtm2$ pour la réalisation de la tâche T.

Le même schéma peut être repris pour modéliser la propriété de *complémentarité*. A la différence, il faudra s'assurer que les deux événements $Evtm1$ et $Evtm2$ se complètent pour réaliser l'événement *Evt*.

INVARIANT
...
$(complementaire = TRUE) \Rightarrow (v1 < N \wedge v2 < N)$
EVENTS
...
$EvtComplementaire =$
SELECT
$(v1 < N \wedge v2 < N)$
THEN
$complemantaire := TRUE$
END

La décrémentation des deux variants dans notre modèle (sans même atteindre la valeur 0) assure que la tâche T a été réalisée par les deux modalités $m1$ et $m2$ sans que $Evtm1$ ou $Evtm2$ ne réalise la tâche T complètement seul. Il y a *complémentarité* des deux modalités pour la réalisation de la tâche T.

Vérification de l'assignation et/ou de l'équivalence. L'événement *EvtAssignation* peut être vu comme un point d'observation de la propriété d'*assignation* déclenché lorsque la propriété est vérifiée à la fin de l'événement *Evt*. L'invariant assure aussi que la propriété est toujours vérifiée lorsque l'événement *EvtAssignation* est déclenché. La propriété d'*équivalence* entre les deux modalités peut être déduite

du fait que l'événement *Evt* peut être réalisé par l'une des deux modalités. L'inconvénient est que l'on ne peut pas placer un événement pour observer l'équivalence tel que le modèle est décrit.

INVARIANT
...
$(assignation = TRUE) \Rightarrow ((v1 = 0 \wedge v2 = N) \vee (v1 = N \wedge v2 = 0))$
EVENTS
...
$EvtAssignation =$
SELECT
$((v1 = 0 \wedge v2 = N) \vee (v1 = N \wedge v2 = 0))$
THEN
$assignation := TRUE$
END

Dans notre modèle, l'assignation est vérifiée par le passage de l'un des variant à la valeur **0** sans que l'autre ne change de valeur initial **N** : soit $v1$ est à 0 et $v2$ à N, dans ce cas la tâche est assignée à la modalité $m1$. Soit $v1$ à N et $v2$ à 0, dans ce cas la tâche est assignée à la modalité $m2$.

Conclusion. Ce modèle ne permet pas de vérifier la propriété d'équivalence. Cela s'explique par l'absence de la notion de mémorisation de la trace des tâches. Pour y arriver, il faudrait prendre en considération le fait qu'une tâche se soit déroulée à des temps différents et avec deux modalités différentes.

3.5 Conclusion

Nous avons présenté un modèle général de représentation des propriétés CARE. Ce modèle considère le temps ainsi que les fenêtres temporelles. Ce point n'a pas été pris en compte dans la modélisation B événementiel proposée.

Nous avons proposé trois approches, fondées sur B événementiel, permettant d'exprimer et de vérifier les propriétés CARE. B événementiel offre plusieurs approches différentes de vérification de ces propriétés. Les différentes approches sont complémentaires et ne permettent pas toutes de vérifier les propriétés. Seule l'expression de tâches permet de vérifier toutes les propriétés.

Une autre approche fondée sur l'utilisation des composants ICARE développés au CLIPS-IMAG sera décrite dans le lot 3. Elle est constructive et utilise des composants qui codent les propriétés CARE en leur sein.

Chapitre 4

Différents scénarios de développement d'IHM3 fondés sur la preuve et le raffinement

4.1 Introduction

Nous étudions le cas particulier de la conception de systèmes interactifs multi modaux. La grande majorité des modèles d'architecture de systèmes interactifs, tel que ARCH [BHH+88], MVC [Bur92] ou PAC [Cou87], distinguent trois composants essentiels : le noyau fonctionnel, la présentation et le contrôleur de dialogue.

- *Le noyau fonctionnel* représente l'ensemble des fonctions et services offerts par le système avec lequel l'utilisateur interagit ;
- *La présentation* est le composant disposant des différentes fonctions permettant de gérer la saisie et l'affichage d'informations consommées ou produites par le noyau fonctionnel, ou bien de gérer les informations de présentation ne nécessitant pas de liaison avec le noyau fonctionnel ;
- *contrôleur de dialogue* organise la communication entre les deux composants précédents.

Nous nous intéressons à la modélisation et à la validation formelle du contrôleur du dialogue. Suite à la décomposition précédents, plusieurs scénarios de développement sont possibles et sont étudiés dans ce chapitre. Une stratégie complète de conception des IHM3 qui repose totalement sur la méthode B événementiel est présentée.

4.2 Quatre scénarios de développement d'une IHM3 avec B

Dans un premier temps, nous présentons la modélisation, en B événementiel, des composants noyau fonctionnel et présentation réalisés séparément. $MODEL_{NF}$ le modèle B décrivant le noyau fonctionnel et $MODEL_{IHM}$ celui qui décrit la présentation. Chacun des ces composants est un système de transition. Ce système est modélisé en B par un état déclaré dans la clause **VARIABLES** et un ensemble d'événements dans la clause **EVENTS**. Ces événements décrivent les changements d'état (les transitions). Les clauses **INVARIANT** et **ASSERTIONS** comportent les propriétés pertinentes de ces systèmes. Enfin, ces systèmes sont construits par une suite de raffinement qui préserve les propriétés.

$MODEL_{NF}$	$MODEL_{IHM}$
...	...
VARIABLES	**VARIABLES**
Var_{NF}	Var_{IHM}
INVARIANT	**INVARIANT**
$I(Var_{NF})$	$I(Var_{IHM})$
ASSERTIONS	**ASSERTIONS**
$A(Var_{NF})$	$A(Var_{IHM})$
INITIALISATION	**INITIALISATION**
$Var_{NF} := ...$	$Var_{IHM} := ...$
EVENTS	**EVENTS**
$Evt =$	$Evt =$
SELECT	**SELECT**
G_{NF}	G_{IHM}
THEN	**THEN**
S_{NF}	S_{IHM}
END ;	**END ;**

Le composant contrôleur de dialogue est obtenu à partir du produit des deux composants qui représentent respectivement le noyau fonctionnel et la présentation. L'état du composant obtenu est un ensemble $Var = \{Var_{NF}, Var_{IHM}\}$ composé des variables du noyau fonctionnel et de la présentation. Les événements sont de la forme :

$Evt =$
SELECT
$G_{NF} \wedge G_{IHM}$
THEN
$S_{NF} \parallel S_{IHM}$
END ;

où G_{NF} et S_{NF} correspondent respectivement à la garde et à la substitution de la partie noyau fonctionnel. A l'inverse, G_{IHM} et S_{IHM} correspondent à la garde

et à la substitution de la partie interface utilisateur. La preuve des modèles B obtenus dépend de la façon de construire ce produit. Cette construction exploite le raffinement.

Nous montrons dans ce qui suit quatre scénarios de développement différents [AAASB06]. Ces derniers sont le résultat de la formalisation de l'opération de composition des parties noyau fonctionnel et présentation d'un système interactif. Cette opération n'existe pas dans la méthode B événementiel. Nous exploitons la technique de raffinement pour composer les deux parties qui caractérisent la structure d'un système interactif.

4.2.1 Scénario 1 : raffinement du noyau fonctionnel

Le développement débute par la spécification de la partie noyau fonctionnel. Ci-dessous, nous présentons le modèle $MODEL_{NF}$ avec l'événement Evt spécifie par des attributs du noyau fonctionnel. L'invariant $I(Var_{NF})$ du modèle ne concerne actuellement que les variables de la partie noyau fonctionnel. Par ailleurs, quand la garde de l'événement Evt est vraie (G_{NF} est vraie), cet événement est déclanché et la substitution S_{NF} est alors exécutée.

$MODEL_{NF}$
VARIABLES
Var_{NF}
INVARIANT
$I(Var_{NF})$
...
EVENTS
$Evt =$
SELECT
G_{NF}
THEN
SNF
END ;

Dans la spécification du raffinement de l'abstraction précédente, nous introduisons de nouveaux attributs décrivant des éléments de la partie présentation. Ainsi nous composons la partie noyau fonctionnel avec la partie présentation au moyen de la technique de raffinement de manière à construire le contrôleur de dialogue. D'autres événements relatifs à la partie présentation peuvent être également introduits.

On rajoute un invariant de collage $JCol(Var_{NF}, Var'_{NF})$ qui traduit la correspondance entre les variables concrètes (Var'_{NF}) et abstraites (Var_{NF}) du noyau fonctionnel. Un invariant $I(Var_{IHM})$ relatif aux attributs de la partie présentation caractérise les propriétés de l'IHM. Ainsi la complexité des preuves du modèle

$MODEL_{CD}$ dépend essentiellement de la complexité de l'invariant résultant. Par ailleurs, l'événement Evt a été enrichi par les éléments de la présentation G_{IHM} et S_{IHM}.

$MODEL_{CD}$
VARIABLES
$Var_{NF}, Var'_{NF}, Var_{IHM}$
INVARIANT
$JCol(Var_{NF}, Var'_{NF}) \wedge I(Var_{IHM})$
...
EVENTS
$Evt =$
SELECT
$G_{NF} \wedge G_{IHM}$
THEN
$S_{NF} \parallel S_{IHM}$
END ;

4.2.2 Scénario 2 : raffinement de la présentation

Ce scénario, contrairement au précédent, débute par la spécification abstraite de la partie présentation. Ci-dessous, nous présentons la spécification du $MODELE_{IHM}$. L'invariant $I(Var_{IHM})$ du modèle ne concerne que les variables de la partie présentation. L'événement Evt est décrit par la garde G_{IHM} et la substitution S_{IHM} exprimées sur les variables de la présentation.

$MODEL_{IHM}$
VARIABLES
Var_{IHM}
INVARIANT
$I(Var_{IHM})$
...
EVENTS
$Evt =$
SELECT
G_{IHM}
THEN
SIHM
END ;

Dans le raffinement de l'abstraction de la partie présentation, nous introduisons de nouveaux attributs issus de la partie noyau fonctionnel. L'événement Evt est raffiné pour composer la partie de la présentation et les nouveaux événements liés au noyau fonctionnel. Nous enrichissons la garde par le prédicat G_{NF} et la post-condition par la substitution S_{NF}.

Cet événement se déclanche quand les gardes de l'interface utilisateur et du noyau fonctionnel sont vraies et modifie ainsi les valeurs des attributs du système interactif complet.

Nous rajoutons un invariant de collage $JCol(Var_{IHM}, Var'_{IHM})$ qui traduit la correspondance entre les variables abstraites (Var_{IHM}) et les variables concrètes (Var'_{IHM}) de la présentation. Nous retrouvons également un invariant $I(Var_{NF})$ qui caractérise les propriétés du noyau fonctionnel.

$MODEL_{CD}$
VARIABLES
$Var_{IHM}, Var'_{IHM}, Var_{NF}$
INVARIANT
$JCol(Var_{IHM}, Var'_{IHM}) \wedge I(Var_{NF})$
...
$Evt =$
SELECT
$G_{IHM} \wedge G_{NF}$
THEN
$S_{IHM} \parallel S_{NF}$
END ;

4.2.3 Scénario 3 : produit de la présentation et du noyau fonctionnel

Un modèle modélisant le contrôleur de dialogue du système interactif est dérivé à partir des deux modèles développés séparément correspondant respectivement aux parties noyau fonctionnel et présentation. Dans le code ci-dessous, nous montrons les deux modèles indépendants. La spécification de la partie noyau fonctionnel est décrite par le modèle à gauche. La spécification de la présentation est décrite sur le second modèle à droite. Chaque modèle possède des attributs, un invariant et un événement nommé respectivement Evt_{NF} et Evt_{IHM}.

$MODEL_{NF}$	$MODEL_{IHM}$
VARIABLES	**VARIABLES**
Var_{NF}	Var_{IHM}
INVARIANT	**INVARIANT**
$I(Var_{NF})$	$I(Var_{IHM})$
...	...
EVENTS	**EVENTS**
$Evt_{NF} =$	$Evt_{IHM} =$
SELECT	**SELECT**
G_{NF}	G_{IHM}
THEN	**THEN**
S_{NF}	S_{IHM}
END ;	**END ;**

La dérivation du composant contrôleur de dialogue se fait en composant les deux modèles précédent au moyen d'un raffinement. Plus précisément, nous fusionnons les deux événements Evt_{NF} et Evt_{IHM} en un seul Evt. La garde de l'événement Evt est obtenue par une conjonction des prédicats des gardes de Evt_{NF} et Evt_{IHM}. De même

la post-condition résultante est établie par une exécution parallèle des substitutions S_{NF} et S_{IHM}. Par ailleurs, l'invariant résultant est obtenu par la conjonction des invariants des modèles composés.

$MODEL_{CD}$
VARIABLES
Var_{NF}, Var_{IHM}
INVARIANT
$I(Var_{NF}) \wedge I(Var_{IHM})$
...
$Evt =$
SELECT
$G_{NF} \wedge G_{IHM}$
THEN
$S_{NF} \parallel S_{IHM}$
END ;

Notons que la spécification de l'événement Evt est la même que celle obtenue après raffinement des spécifications du premier et deuxième scénarios. Toute fois, l'invariant résultant est différent. Ce type de composition, notamment par la conjonction direct des invariants amène à une plus grande complexité des obligations de preuve puisque l'intérêt du raffinement de la méthode B n'est pas utilisé (absence de décomposition).

4.2.4 Scénario 4 : produit de la présentation et d'une abstraction du noyau fonctionnel

Dans ce scénarios, nous considérons la partie noyau fonctionnel comme étant abstraite. Ainsi, nous employons une simple variable entière qui détermine si l'état du noyau fonctionnel est modifié ou pas.

Nous donnons ci-dessus deux modèles concernant les parties noyau fonctionnel et présentation. Sur la gauche, nous utilisons la variable var_{NF} pour sauvegarder l'état de la partie noyau fonctionnel. Si l'événement Evt_{NF} est déclanché, c'est-à-dire quand var_{NF} est égale à un, l'état du noyau fonctionnel est modifié à zéro. Concernant l'événement Evt_{IHM} il est identique à celui du troisième scénario, l'invariant de la partie noyau fonctionnel est ici beaucoup plus simple (expression logique sur des variables booléennes entières).

$MODEL_{NF}$	$MODEL_{IHM}$
VARIABLES	**VARIABLES**
var_{NF}	Var_{IHM}
INVARIANT	**INVARIANT**
$var_{NF} \in \{0,1\}$	$I(Var_{IHM})$
...	...
EVENTS	**EVENTS**
$Evt_{NF} =$	$Evt_{IHM} =$
SELECT	**SELECT**
$var_{NF} = 1$	G_{IHM}
THEN	**THEN**
$var_{NF} := 0$	S_{IHM}
END ;	**END ;**

Nous montrons ci-dessous le résultat de la composition obtenu par le technique de raffinement. En comparant avec le troisième scénario, G_{NF} et S_{NF} ont été remplacés par la modification des variables entières var_{NF}. L'invariant du raffinement reste du point de vue forme identique au troisième scénario. Toutefois la complexité des obligations de preuve du raffinement est moindre puisque l'invariant de la partie noyau fonctionnel est plus simple et ne s'applique qu'à des variables booléennes entières.

$MODEL_{CD}$
VARIABLES
var_{NF}, Var_{IHM}
INVARIANT
$var_{NF} \in \{0,1\} \wedge I(Var_{IHM})$
...
$Evt =$
SELECT
$G_{IHM} \wedge var_{NF} = 1$
THEN
$S_{IHM} \parallel var_{NF} := 0$
END ;

4.3 Application à l'étude de cas Matis

Nous présentons dans cette section une application des scénarios de développement à l'étude de cas MATIS . Une comparaison des quatre scénarios est réalisée sur la base du nombre d'obligations de preuve générées par chaque scénario. En gros, MATIS est une application multimodale qui permet d'avoir des informations sur les horaires de vols en utilisant la voix, la manipulation directe avec le clavier et la souris, ou une combinaison de ces modalités.

Nous ne décrivons pas le développement complet de l'étude de cas MATIS dans cette section, nous nous intéressons qu'aux événements les plus importants afin de mettre en valeur les différents changements effectués sur le modèle dans chaque

scénario. Une définition des variables et événements utilisés dans tous les développements a été donné dans la section 2.4.1.

4.3.1 Scénario 1

Nous décrivons dans cette section l'application du premier scénario à l'étude de cas. Le code suivant correspond à la première étape (modélisation de la partie noyau fonctionnel). Ce modèle décrit la récupération des attributs de la requête à travers l'événement $Evt_{InputDeparture}$ et $QueryReady$, et aussi le calcule de la requête par les événements $QueryOK$ et $QueryKO$. La variable $result$ quand à elle, contient les résultats de la requête (cf. figure 4.1).

Dans la seconde étape, et dans le but de composer le noyau fonctionnel avec la présentation, nous enrichissons le modèle précédent en ajoutant trois variables ($MouseModal$, $KeyboardModal$ et $SpeechModal$) pour représenter les modalités utilisées par l'utilisateur. Nous les retrouvons au niveau de la garde de l'événement $Evt_{InputDeparture}$. Le code de la figure 4.2 montre les parties enrichies par le raffinement.

4.3.2 Scénario 2

Ce scénario est le dual du précédent. Nous présentons le modèle correspondant à l'interface utilisateur (cf. figure 4.3). Nous retrouvons les attributs relatifs aux différentes modalités activées par l'utilisateur ($MouseModal$, $KeyboardModal$ et $SpeechModal$) ainsi que les événements qui sont déclanchés par cette activation.

Nous raffinons ce modèle pour le composer avec le noyau fonctionnel. Nous introduisons les attributs qui permettent de récupérer et de sauvegarder les paramètres de la requête ($PlaceDeparture$) ainsi que les attributs qui permettent de la calculer ($result$ et $index$). Nous introduisons également de nouveaux événement pour le calcule de la requête ($QueryOK$ et $QueryKO$) et qui en même temps raffinent l'événement $QuerySending$. Le modèle obtenu (cf. figure 4.4) est identique dans son dernier raffinement au modèle résultant de l'application du scénario 1 à l'étude de cas.

4.3.3 Scénario 3

Pour ce scénario, nous commençons par présenter deux modèles différents qui décrivent les deux composants, présentation et noyau fonctionnel, de l'étude de cas

VARIABLES
$PlaceDeparture, ...$
$queryReady, querySending, ...$
$result, index, ...$
...
EVENTS
$Evt_{InputDeparture}$ = **SELECT**
$Evt1 = 1 \wedge$
THEN
$PlaceDeparture :\in 1..9 \parallel Evt1 := 0$
END ;
...
$QueryReady$ = **SELECT**
$queryReady = 1 \wedge$
$(Evt1 = 0 \wedge Evt2 = 0 \wedge Evt3 = 0 \wedge Evt4 = 0)$
THEN
$queryReady := 0 \parallel querySending := 1$
$index := card(vol) \parallel ...$
...
END ;

$QueryOK$ = **SELECT**
$querySeending = 1 \geq 0 \wedge index \geq 0 \wedge$
$townDeparture(vol(index)) = PlaceDeparture \wedge townArrival(vol(index)) = PlaceArrival \wedge$
$timeDeparture(vol(index)) = HourDeparture \wedge timeArrival(vol(index)) = HourArrival$
THEN
$result := result \cup \{vol(index)\} \parallel$
$index := index - 1$
END ;

$QueryKO$ = **SELECT**
$querySeending = 1 \geq 0 \wedge index \geq 0 \wedge$
$NOT(townDeparture(vol(index)) = PlaceDeparture \wedge townArrival(vol(index)) = PlaceArrival \wedge$
$timeDeparture(vol(index)) = HourDeparture \wedge timeArrival(vol(index)) = HourArrival)$
THEN
$index := index - 1$
END ;

$QuerySending$ = **SELECT**
$index = 0 \wedge querySending = 1$
THEN
$queryReady := 1 \parallel querySending := 0$
...
END

Fig. 4.1 – Modèle du noyau fonctionnel de MATIS

VARIABLES
$PlaceDeparture, ...$
MouseModal, KeyboardModal, SpeechModal, ...
$queryReady, querySending, ...$
$result, index, ...$
INVARIANT
MouseModal $\in State \wedge$ **KeyboardModal** $\in State \wedge$ **SpeechModal** $\in State \wedge ...$
...
EVENTS
$Evt_{InputDeparture}$ = **SELECT**
$Evt1 = 1 \wedge$
(MouseModal = on $\vee$ **KeyboardModal = on** $\vee$ **SpeechModal = on)**
THEN
$PlaceDeparture :\in 1..9 \parallel Evt1 := 0$
END ;
...
$QueryReady$ = **SELECT**
$queryReady = 1 \wedge$
$(Evt1 = 0 \wedge Evt2 = 0 \wedge Evt3 = 0 \wedge Evt4 = 0)$
THEN
$queryReady := 0 \parallel querySending := 1$
$index := card(vol) \parallel ...$
...
END ;

$QueryOK$ = **SELECT**
$querySeending = 1 \geq 0 \wedge index \geq 0 \wedge$
...
THEN
$result := result \cup \{vol(index)\} \parallel$
$index := index - 1$
END ;

$QueryKO$ = **SELECT**
$querySeending = 1 \geq 0 \wedge index \geq 0 \wedge$
$NOT(...)$
THEN
$index := index - 1$
END ;

$QuerySending$ = **SELECT**
$index = 0 \wedge querySending = 1$
THEN
$queryReady := 1 \parallel querySending := 0$
...
END

Fig. 4.2 – Raffinement du noyau fonctionnel en rajoutant la présentation de MATIS

VARIABLES
$MouseModal, KeyboardModal, SpeechModal, ...$
$queryReady, querySending, ...$
INVARIANT
$MouseModal \in State \wedge$
$Keyboard \in State \wedge$
$SpeechModal \in State \wedge ...$
$queryReady \in \{0,1\} \wedge$
$querySending \in \{0,1\} \wedge ...$
...
EVENTS
$Evt_{InputDeparture}$ = **SELECT**
$Evt1 = 1 \wedge$
$(MouseModal = on \vee KeyboardModal = on \vee SpeechModal = on)$
THEN
$Evt1 := 0$
END ;
...
$QueryReady$ = **SELECT**
$queryReady = 1 \wedge$
$(Evt1 = 0 \wedge Evt2 = 0 \wedge Evt3 = 0 \wedge Evt4 = 0)$
THEN
$queryReady := 0 \parallel querySending := 1$
END ;

$QuerySending$ = **SELECT**
$querySending = 1$
THEN
$queryReady := 1 \parallel querySending := 0 \parallel$
...
END

Fig. 4.3 – Modèle de la présentation de MATIS

VARIABLES
$PlaceDeparture, ...$
$MouseModal, KeyboardModal, SpeechModal, ...$
$queryReady, querySending, ...$
$result, index, ...$
INVARIANT
$PlaceDeparture \in 1..9 \wedge ...$
$index \in NAT...$
$result \subseteq VOLS \wedge ...$
...
EVENTS
$Evt_{InputDeparture}$ = **SELECT**
$Evt1 = 1 \wedge$
$(MouseModal = on \vee KeyboardModal = on \vee SpeechModal = on)$
THEN
PlaceDeparture : $\in$ **1..9** $\parallel$ $Evt1 := 0$
END ;
...
$QueryReady$ =
...
END ;

$QueryOK$ = **SELECT**
$querySeending = 1 \geq 0 \wedge index \geq 0 \wedge$
$townDeparture(vol(index)) = PlaceDeparture \wedge townArrival(vol(index)) = PlaceArrival \wedge$
$timeDeparture(vol(index)) = HourDeparture \wedge timeArrival(vol(index)) = HourArrival$
THEN
$result := result \cup \{vol(index)\} \parallel$
$index := index - 1$
END ;

$QueryKO$ = **SELECT**
$querySeending = 1 \geq 0 \wedge index \geq 0 \wedge$
$NOT(townDeparture(vol(index)) = PlaceDeparture \wedge townArrival(vol(index)) = PlaceArrival \wedge$
$timeDeparture(vol(index)) = HourDeparture \wedge timeArrival(vol(index)) = HourArrival)$
THEN
$index := index - 1$
END ;

$QuerySending$ = **SELECT**
$index = 0 \wedge querySending = 1$
THEN
$queryReady := 1 \parallel querySending := 0$
...
END

Fig. 4.4 – Raffinement de la présentation en rajoutant le NF de MATIS

MATIS. A gauche les attributs et les événements du composant présentation et à droite les attributs et les événements du composant noyau fonctionnel (figure 4.5).

En utilisant le raffinement, nous composons les deux modèles pour obtenir un modèle qui correspond au contrôleur de dialogue (figure 4.6). Nous reprenons tous les attributs des deux modèles et nous combinons les mêmes événements. Pour ces derniers, la garde qui en résulte est obtenue par une conjonction des deux gardes des événements des modèles précédents. La substitution résultat est obtenue par une exécution parallèle des deux substitutions qui composent les événements initiaux. Le modèle obtenu comprend les mêmes événements qui existent dans le dernier raffinement de l'application des deux scénarios précédents à l'étude de cas MATIS.

4.3.4 Scénario 4

Dans ce scénario, nous modifions la partie du noyau fonctionnel du scénario 3 en exprimant la garde d'un événement et ses substitutions à travers le comportement d'une variable booléenne entière. Cette variable vaut 1 pour exprimer une garde qui est évaluée à $TRUE$, et qui est substituée par la valeur 0 pour exprimer les changements effectués sur les différents attributs du noyau fonctionnel. Ce dernier modèle est présenté dans le code de la figure 4.7 ainsi que le code de la partie présentation qui est le même que celle utilisée dans le scénario 3.

Le modèle obtenu après fusion des deux modèles (nous avons utilisé l'opération de raffinement pour la composition des deux modèle) permet de vérifier d'une part l'IHM et d'autre part que le contrôleur de dialogue communique correctement avec le noyau fonctionnel. Nous avons remarqué que certaines variables de l'abstraction du noyau fonctionnel telles que $Ev_{QueryReady}$ et $Ev_{QuerySending}$ ont le même comportement que les variables utilisées comme des variants dans la présentation (*queryReady* et *querySending*), donc nous n'avons gardé dans le modèle du contrôleur de dialogue que celles de l'abstraction du noyau fonctionnel pour ne pas charger le modèle résultant. C'est le cas également de la variable $Ev_{inputDeparture}$ avec la variable $Evt1$ (figure 4.8).

4.3.5 Résultats obtenus

Nous avons exposé des scénarios différents et nous les avons appliqué sur une étude de cas. La comparaison se fait sur le nombre d'Obligations de Preuve (OP) générées par chacun des scénarios. Le tableau suivant illustre les résultats obtenus.

A la lecture de ce tableau, nous constatons que les différents scénarios produisent un nombre d'OP différents selon les scénarios. Il faut aussi remarquer que le scénario

VARIABLES $MouseModal$, $KeyboardModal$, $SpeechModal, ...$ **INVARIANT** $MouseModal \in State \wedge$ $Keyboard \in State \wedge$ $SpeechModal \in State \wedge ...$ ... **EVENTS** $Evt_{InputDeparture}$ = **SELECT** $Evt1 = 1 \wedge$ $(MouseModal = on \vee$ $KeyboardModal = on \vee$ $SpeechModal = on)$ **THEN** $Evt1 := 0$ **END ;** $QueryReady$ = **SELECT** $queryReady = 1 \wedge$ $(Evt1 = 0 \wedge$ $Evt2 = 0 \wedge$ $Evt3 = 0 \wedge$ $Evt4 = 0)$ **THEN** $queryReady := 0 \parallel$ $querySending := 1 \parallel ...$ **END ;** $QuerySending$ = **SELECT** $querySending = 1$ **THEN** $queryReady := 1 \parallel$ $querySending := 0 \parallel ...$ **END**	**VARIABLES** $PlaceDeparture, ...$ $result, index, ...$ **INVARIANT** $PlaceDeparture \in 1..9 \wedge ...$ $result \subseteq VOLS \wedge ...$ $index \in NAT \wedge ...$ ... **EVENTS** $Evt_{InputDeparture}$ = **SELECT** $Evt1 = 1 \wedge$ **THEN** $PlaceDeparture :\in 1..9 \parallel$ $Evt1 := 0$ **END ;** ... $QueryReady$ = **SELECT** $queryReady = 1 \wedge$ $(Evt1 = 0 \wedge$ $Evt2 = 0 \wedge$ $Evt3 = 0 \wedge$ $Evt4 = 0)$ **THEN** $queryReady := 0 \parallel$ $querySending := 1 \parallel$ $index := card(vol) \parallel ...$ **END ;** $QuerySending$ = **SELECT** $index = 0 \wedge$ $querySending = 1$ **THEN** $queryReady := 1 \parallel$ $querySending := 0 \parallel ...$ **END** $QueryOK$ = **SELECT** ... **END ;** $QueryKO$ = **SELECT** ... **END ;**

Fig. 4.5 – IHM et NF de MATIS

VARIABLES
$PlaceDeparture, ...$
$MouseModal, KeyboardModal, SpeechModal, ...$
$queryReady, querySending, ...$
$result, index, ...$
...
EVENTS
$Evt_{InputDeparture}$ = **SELECT**
$Evt1 = 1 \wedge$
$(MouseModal = on \vee KeyboardModal = on \vee SpeechModal = on)$
THEN
$PlaceDeparture :\in 1..9 \parallel Evt1 := 0$
END ;
...
$QueryReady$ = **SELECT**
$queryReady = 1 \wedge$
$(Evt1 = 0 \wedge Evt2 = 0 \wedge Evt3 = 0 \wedge Evt4 = 0)$
THEN
$queryReady := 0 \parallel querySending := 1$
$index := card(vol) \parallel ...$
...
END ;

$QueryOK$ = **SELECT**
...
END ;

$QueryKO$ = **SELECT**
...
END ;

$QuerySending$ = **SELECT**
$index = 0 \wedge querySending = 1$
THEN
$queryReady := 1 \parallel querySending := 0$
...
END

Fig. 4.6 – composition de l'IHM et NF de MATIS

VARIABLES $MouseModal$, $KeyboardModal$, $SpeechModal, ...$ **INVARIANT** $MouseModal \in State \wedge$ $Keyboard \in State \wedge$ $SpeechModal \in State \wedge ...$ ... **EVENTS** $Evt_{InputDeparture}$ = **SELECT** $Evt1 = 1 \wedge$ $(MouseModal = on \vee$ $KeyboardModal = on \vee$ $SpeechModal = on)$ **THEN** $Evt1 := 0$ **END ;** $QueryReady$ = **SELECT** $queryReady = 1 \wedge$ $(Evt1 = 0 \wedge$ $Evt2 = 0 \wedge$ $Evt3 = 0 \wedge$ $Evt4 = 0)$ **THEN** $queryReady := 0 \parallel$ $querySending := 1 \parallel ...$ **END ;** $QuerySending$ = **SELECT** $querySending = 1$ **THEN** $queryReady := 1 \parallel$ $querySending := 0 \parallel ...$ **END**	**VARIABLES** $Ev_{InputDeparture}, \cdots$ $Ev_{QueryReady}$, $Ev_{QuerySending}$ **INVARIANT** $Ev_{InputDeparture} \in 0..1 \wedge ...$ $Ev_{QueryReady} \in 0..1 \wedge$ $Ev_{QuerySending} \in 0..1$ ... **EVENTS** $Evt_{InputDeparture}$ = **SELECT** $Ev_{InputDeparture} = 1 \wedge$ **THEN** $Ev_{InputDeparture} := 0$ **END ;** ... $QueryReady$ = **SELECT** $Ev_{QueryReady} = 1 \wedge$ $Ev_{InputDeparture} = 0 \wedge ...$ **THEN** $Ev_{QueryReady} := 0 \parallel$ $Ev_{QuerySending} := 1$ **END ;** $QuerySending$ = **SELECT** $Ev_{QuerySending} = 1$ **THEN** $Ev_{QuerySending} := 0 \parallel ...$ **END**

Fig. 4.7 – IHM et l'abstraction du NF de MATIS

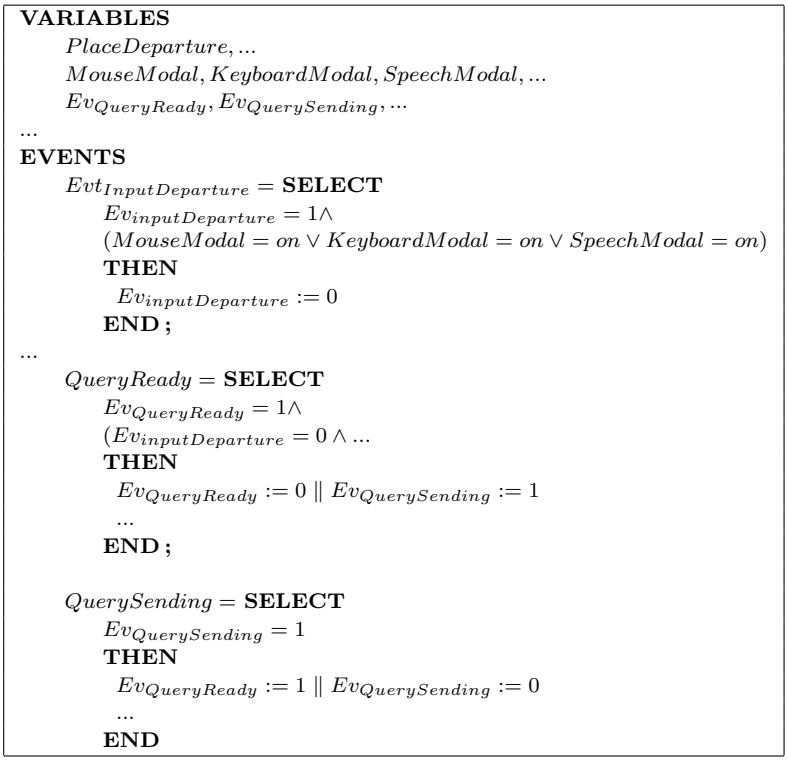


Fig. 4.8 – Composition de IHM et l'abstraction du NF de MATIS

Scénario	OP Tri	OP	Nb Raf
1	1133	127	3
2	1063	81	3
3	760	94	2
4	189	41	2

Tab. 4.1 – Résumé des Obligations de Preuve (OP) des quatre scénarios de l'étude de cas MATIS (Scénario = Scénario de développement, OP Tri = Obligation de preuve triviales, OP = Obligations de Preuve, Nb Raf = Nombre de Raffinement

4 montre que la technique d'abstraction choisie réduit le nombre d'OP à prouver. Par ailleurs, les scénarios 1 et 2 produisent un nombre d'OP plus élevé du fait du nombre de raffinements nécessaires, mais plus simple que dans le cas de la composition parallèle du scénario 3 malgré le nombre plus faible d'OP de ce scénario.

4.4 Conclusion

Cette section nous a permet d'arriver à un résultat essentiel. Ce résultat concerne le scénario de développement à choisir pour valider une IHM3 parmi les quatre proposés. Les résultats nous ont montrés que le scénario 4 (composition de l'abstraction du noyau fonctionnel avec la présentation) génère moins d'obligations de preuve et propose des invariants moins compliqués que dans les autres cas ou le noyau fonctionnel est intégré. Ce scénario permet de vérifier d'une part l'IHM et d'autre part que le contrôleur de dialogue communique correctement avec la présentation.

Chapitre 5

Bilan

5.1 Résultats obtenus

Les travaux effectués et rapportés dans ce document ont permis de :

- modéliser l'interaction multi-modale en entrée avec B événementiel. Cette modélisation se fonde sur la décomposition d'un événement décrivant un état de départ et un état d'arrivée de l'interaction avec introduction d'actions de base et des modalités. Cette décomposition n'a suivi aucune méthode ni respecté de notation. Elle a été expérimentée sur deux études de cas ;
- modéliser des propriétés d'utilisabilité. Ces propriétés expriment des caractéristiques de l'interaction comme la complémentarité ou la redondance. Elles ont été exprimées à l'aide différentes approches. L'approche fondée sur l'expression de tâches utilisateur a permis de représenter toutes les propriétés ;
- définir différents scénarios de développement permettant d'améliorer la qualité de ces dernier en termes de complexité des développements et des preuves des différentes obligations de preuve. Une abstraction du noyau fonctionnel a été possible grâce à la séparation entre noyau fonctionnel et dialogue opérée par la plupart des modèles d'architecture définis par les concepteurs d'IHM3.

5.2 Insuffisances et perspectives

Les travaux présentés dans ce document ne couvrent que partiellement les notations définies par les développeurs d'IHM3. En particulier, l'analyse de tâche utilisateur et les propriétés CARE à la base de l'analyse d'utilisabilité.

Pour ce qui est des tâches utilisateurs, elles doivent être prises en compte au plus tôt dans le développement, alors qu'elles sont dans la plupart des cas validées par le test et l'expérimentation sur le système finale. Nous montrerons dans le LOT 3 comment des expressions d'algèbre de processus permettant d'exprimer des tâches utilisateur sont validées par des modèles B a priori. Nous montrerons également, mais dans le LOT 4, comment ces modèles B peuvent être animés pour la validation de tâches.

Concernant les propriétés CARE, nous montrerons dans le LOT 3 une démarche fondée sur les composants ICARE développés au CLIPS-IMAG. Ces composants vérifient par construction ces propriétés. Nous définirons une modélisation de ces composants en B et nous montrerons comment un modèle de l'IHM3 peut être raffiné pour atteindre ces composants au dernier niveau de raffinement.

Bibliographie

[AAASB06] Y. Ait-Ameur, I. Ait-Sadoune, and M. Baron. Etude et comparaison de scénarios de développements formels d'interfaces multi-modales fondés sur la preuve et le raffinement. In *MOSIM 2006 - 6ème Conférence Francophone de Modélisation et Simulation. Modélisation, Optimisation et Simulation des Systèmes : Défis et Opportunités*, Rabat, 2006.

[AAASBM06] Y. Ait-Ameur, I. Ait-Sadoune, M. Baron, and JM. Mota. Validation et vérification formelles de systèmes interactifs multi-modaux fondées sur la preuve. In *18° Conférence Francophone sur l'Interaction Homme-Machine (IHM)*, pages 123–130, Montréal, 2006.

[Alt91] J. Alty. Multimedia - what is it and how do we exploit it ? In D. Diaper and N. Hammond, editors, *Acte de la conférence Human-Computer Interaction*, pages 31–44. People and Computers VI, 1991.

[AS05] I. Ait-Sadoune. *Vérification et validation formelle d'IHM Multimodales fondées sur la preuve. Utilisation de la Méthode B.* Mémoire d'ingénieur d'état en informatique, INI, Alger, juin 2005.

[BD90] M. Blattner and R. Dannenberg. Chí90 workshop on multimedia and multimodal interface design. *SIGCHI Bulletin*, 22(2) :54–58, 1990.

[Bel95] Y. Bellik. *Interfaces Multimodales : concepts, modèles et architecture.* PhD thesis, LIMSI- Université d'Orsay, 1995.

[Ber94] O. Bersen. Foundations of multimodal representation. a taxonomy of representation modalities. *Interacting with Computer*, 6(4) :347–371, 1994.

[BHH+88] L. Bass, E. Hardy, K. Hoyt, R. Little, and R. Seacord. The arch model : Seeheim revisited, the serpent run time architecture and dialog model. Technical Report CMU/SEI-88-TR-6, Carnegie Melon University, 1988.

[Bol80] R. Bolt. Put That There : Voice and Gesture at the Graphics Interface. In *SIGGRAPH'80 Proceedings*, volume 14, pages 262–270. ACM Press, 1980.

[Bou92] M-L. Bourguet. *Conception et réalisation d'une interface de dialogue personne-machine multimodale.* Doctorat d'université, Institut National Polytechnique de Grenoble, 1992.

[Bur92] Steve Burbeck. Application programming in smalltalk-80 : How to use the model-view-controller (mvc). Report, 1992.

[CNS+95] J. Coutaz, L. Nigay, D. Salber, A. Blandford, J. May, and R. Young. Four easy pieces for assessing the usability of multimodal interaction : The CARE properties. In *IFIP Int. Conf. on Human-Computer Interaction Interact'95*, pages 115–120, London, 1995. Chapman & Hall.

[Cou87] J. Coutaz. Pac, an implementation model for the user interface. In *IFIP TC13 Human-Computer Interaction (INTERACT'87)*, pages 431–436, Stuttgart, 1987. North-Holland.

[DBDM94] D. Duke, P. Barnard, D. Duce, and J. May. Syndetic model for human-computer interaction. Technical report, ID/WP35, ESPRIT BRA 7040 Amodeus-2, 1994.

[Fro91] D. Frohlich. The design space of interface,. In *Multimedia Systems, Interaction and Applications, 1st Eurographics Workshop*, pages 53–69. L. Kjelldahl(Ed.), Springer Verlag, 1991.

[FVC84] J. Foley, V.Wallance, and P. Chan. The human factors of computer graphics interaction techniques. *IEEE computer Graphics and Applications*, 4(11) :13–48, 1984.

[Mar95] J-C. Martin. *Coopération entre modalités et liage par synchronie dans les interfaces multimodales.* Doctorat d'université, Ecole Nationale Supérieure des Télécomunications, Paris, 1995.

[Mar97] J.C. Martin. TYCOON : Theoritical Framework and Software Tools for multimodal interfaces. In *Intelligence and Multimodality in Multimedia Interfaces.* AAAI press, 1997.

[MB] J.C. Martin and B. Beroule. Multimodal interfaces based on types and goals of cooperation between modalities. In *IMMI-1, 1st International Workshop on intelligence and Multimodality in Multimedia Interfaces : research and applications*, Edinburgh.

[NC96] L. Nigay and J. Coutaz. Espaces conceptuels pour l'interaction multimédia et multimodale. *TSI, Spécial Multimédia et Collecticiel*, 15(9) :1195–1225, 1996.

[Nig94] Laurence Nigay. *Conception et Modélisation Logicielle des Systèmes Interactifs : Application aux Interfaces Multimodales*. Doctorat d'université (phd thesis), Université Joseph Fourier, 1994.