MASTER RECHERCHE
INFORMATIQUE DE LYON

ADAPTABILITE (MUI)

Présenté par : Hajer KOUNI ,Reim DOUMAT

INDEX

I- INTRODUCTION

Les années récentes ont vu l'introduction de beaucoup de types des ordinateurs, des dispositifs et d'appareils d'enchaînement. Afin d'accomplir leurs tâches, peuples ont maintenant disponible une grande variété de dispositifs informatiques s'étendant au-dessus d'un spectre important: les téléphones de mobilophones, intelligents et de confort, PDA, PC de poche, l'Internet ont permis les télévisions (WebTV), les cahiers, etc..... Tandis que cette prolifération croissante des dispositifs fixes et mobiles équipés du besoin d'accès omniprésent au traitement de l'information, cette diversité offre de nouveaux défis à la communauté de logiciel de HCI. Ceux-ci incluent:

- ♦ construisant et maintien de versions avec des applications simples à travers le multiple de dispositifs
- ♦ examinant l'uniformité entre les versions pour assurer garantir une interaction sans couture à travers multiple de dispositifs
- ♦ bâtir dans ces versions la capacité de répondre dynamiquement aux changements de l'environnement tel que la connectivité de réseau, l'endroit de l'utilisateur, le bruit ambiant ou les états d'éclairage.

Pour adresser ces nouvelles conditions, la notion de (multi- targeting) et (plasticité) sont présentées. Traitez l'adaptation au contexte de l'utilisation de couvrir cette nouvelle diversité sans éclater le coût de développement et d'entretien. Le contexte se rapporte à la signification qui doit être impliquée du texte adjacent. En conséquence, le contexte peut seulement être défini par rapport à un but, ou finalité. Le contexte de l'utilisation d'un système interactif est défini par trois classes des entités:

- ♦ les utilisateurs du système qui sont prévus pour employer le système
- ♦ les plateformes de matériel et de logiciel, cet à dire, l'informatique et des dispositifs d'interaction qui peuvent être utilisés pour agir l'un sur l'autre avec le système
- ♦ l'environnement physique où l'interaction peut avoir lieu.

Adaptation

En HCI, l'adaptation est modelée en tant que deux propriétés complémentaires de système: adaptabilité et adaptativité. L'adaptabilité est la capacité du système de permettre à des utilisateurs d'adapter leur système aux besoins du client d'un ensemble prédéfini de paramètres. L'adaptativité est la capacité du système d'effectuer l'adaptation automatiquement sans action délibérée de la pièce de l'utilisateur. Si l'adaptation est effectuée sur des demandes humaines ou automatiquement

. Ces dernières années, de nombreux efforts ont été faits pour proposer des méthodes de génération d'IHM multi-contextes. On a vu apparaître des termes comme MUI (Multiple User Interfaces) ou *"plasticité des interfaces"* qui se définit comme étant la capacité d'une interface à s'adapter aux contraintes matérielles et environnementales dans le respect de son utilisabilité.

C’est dans cette thématique que seront étudiés les deux articles concernés. Le premier datant de 2001 pour Brusilovsky s’intitule Adaptive Hypermedia. Dans son article, Brusilovsky a présenté l’état de l’art qui touche cette technologie à l’aube de cette époque, et ce par rapport à sa première évaluation faite en 96. Le second article traité et qui est plus récent (2002), rédigé par Moruzzi et qui s’intitule One Model many interfaces, met le point sur un aspet de l’adaptabilité à savoir l’adaptabilité des interfaces aux dispositifs matériels.

II - Adaptive Hypermedia

1. INTRODUCTION

L'hypermedia adaptative est née de la convergence de deux domaines à savoir l'hypermédia et le User Modeling. Cette technologie apporte une solution à une problématique sentie dans les années 90 quand les technologies parentes ont atteint un certain niveau de maturité : l'hypermédia est statique et est la même pour tous les utilisateurs en dépit des multiples différences qui dépendent de chaque individu. L'hypermédia adaptative a pour but de personnaliser l'interaction avec l'utilisateur.

Par exemple , un traditionnel éducateur hypermedia système présentera la même explication statique et suggére la meme page aux étudiants avec différents buts et la connaissance de la matière. Adaptive hypermedia est une alternative à l'approche traditionnelle « one size fits all » dans le développement de hypermedia systèmes. Un étudiant dans un systeme éducateur hypermedia sera donné une présentation adaptée speciallement à sa connaissance de la matière

2. Etat de la recherche en Hypermedia adaptative avant et après 1996

L'année de 1996 peut être considérée comme un moment décisif dans la recherche concernant adaptive hypermedia.

Avant , la recherche est effectuée par quelques équipes isolées. Toutefois, depuis 1996, adaptive hypermedia est allée au travers d'une période de la croissance rapide.

En particulier , plusieurs nouvellex équipes de recherches ont commencé les projets dans adaptive hypermedia, et beaucoup d'étudiants ont choisi adaptive hypermedia comme la surface assujettie pour leur Phd thèses. En outre , plusieurs ateliers directement ou relaté indirectement à adaptive hypermedia ont été retenus.

Il y a deux facteurs principaux qui ont pu estimer pour cette croissance de l'activité de recherche, et un lecteur appliqué qui a une chance à comparer un nombre raisonnable de adaptive hypermedia articles parus en librairie avant et après 1996.

3. AH : Où ? Quoi ? A quoi ?

3.1. Domaines

D'après P.Brusiovsky, six domaines principaux bénéficient des apports de l'AH. Cette classification a été faite dans son premier article. Ici, il met le point sur les principaux domaines qui se sont imposés le plus à savoir : educational Hypermedia, On-line Information sytems et Information Retrieval. Ces applications sont les principales leader ; mais d'autres applications sont possibles mais qui n'ont pas connu le meme rythme d'évolution, ces applications sont : on-line-help system, institutional hypermedia et systems for managing personalized views dans les espaces d'information.

3.2. Adapter quoi ?

Brusilovsky résume la répons à cette question par un schéma qu'il a commencé à élaborer dans son premier article et qu'il a essayé de mettre à jour. Sur ce schéma, présenté ci-dessous, l'auteur classifie les types d'adaptations en deux classes majeures : l'adaptation du niveau du contenu ou adaptive navigation support et l'adaptation du lien meme ou adaptive navigation support. Chaque classe est subdivisée à son tour :

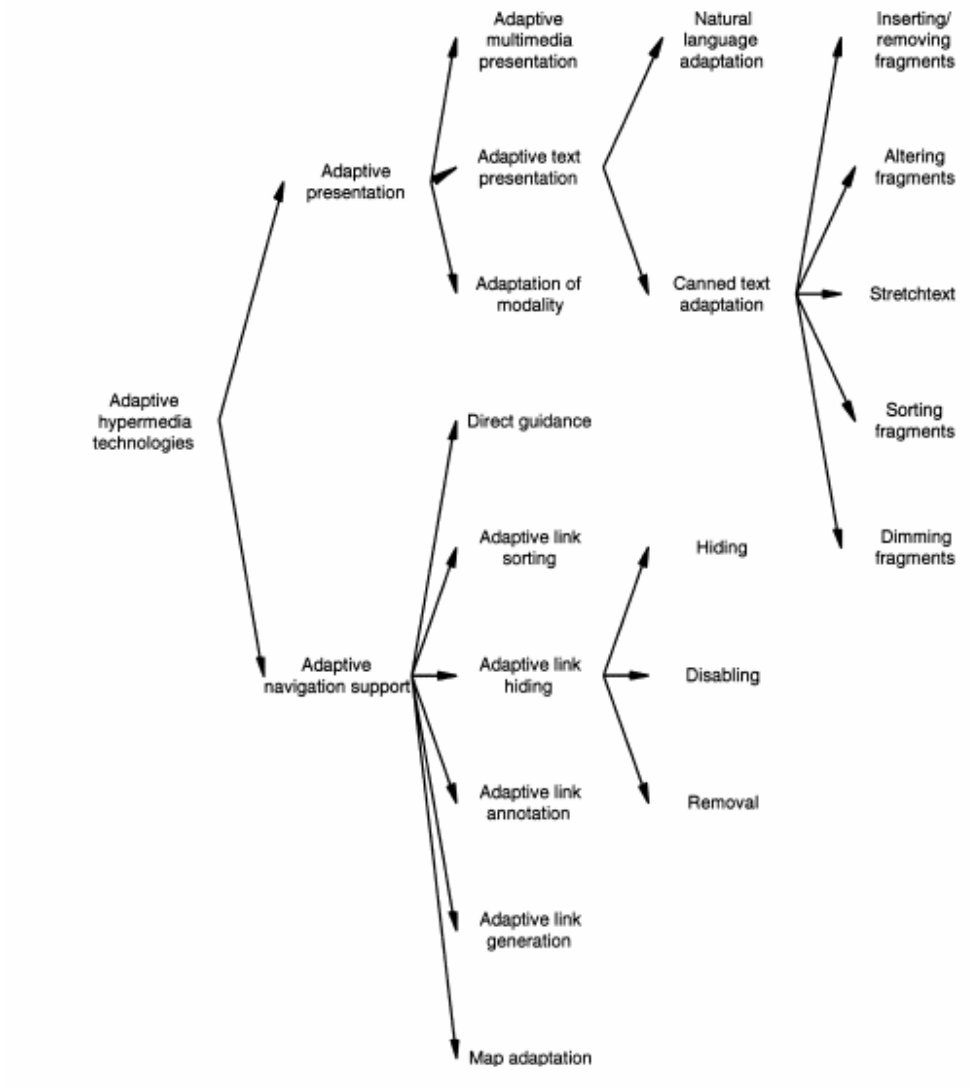


Figure 1. La taxonomie des technologies de l'hypermedia mise à jour

3.3. Adapter à quoi ?

Traditionnellement la décision d'adaptation dans les systèmes adaptatifs ont été basés sur la prise en compte des caractéristiques diverses de leurs utilisateurs représenté dans le modèle d'utilisateur.

Pour pré-1996, l'article adaptive hypermedia systèmes dans la section ``Adapter à quoi" de la revue traiter exclusivement avec les caractéristiques d'utilisateurs.

La situation est différente et elle a évolué avec le développement spectaculaire des moyens informatiques mis à la disposition de ces utilisateurs, ce qui a fait que

quelque chose d'autre que les caractéristiques d'utilisatrices declenche l’adaptabilité.

On suggère ainsi la prise en compte de l’environnement et du contexte élargi de l’utilisateur. L’article suivant, traite un aspect de cette adaptabilité en expliquant mieux la notion de plasticité c'est-à-dire la capacité de l’application à s’adapter à l’equipment mis à la disposition de l’utilisateur et ce sans etre pour autant déformée et inutilisable.

III - **ONE MODEL, MANY INTERFACES**

ISTI, CNR Institute, Via G. Moruzzi 1

1. INTRODUCTION

Les années récentes ont vu l'introduction de beaucoup de types des ordinateurs et de dispositifs (par exemple téléphones cellulaires, PDA's, WebTV, etc...) et la disponibilité d'un éventail si de dispositifs est devenue un défi fondamental pour des concepteurs des systèmes logiciels interactifs. Les utilisateurs souhaitent pouvoir accéder à l'information et à des services indépendamment du dispositif qu'ils utilisent, même lorsque le système ou les changements d'environnement dynamiquement. À cet effet, les applications sur ordinateur doivent fonctionner sur une gamme étendue de dispositifs. Concevant les applications qui exploitent la nouvelle technologie est souvent un problème difficile.

Pour des réalisateurs de logiciel ceci présente le problème de construire des versions multiples avec des applications simples et de doter ces versions avec la capacité de répondre dynamiquement aux changements du contexte. Créant différentes versions des demandes de différents dispositifs engendre le développement supplémentaire et le coût élevé d'entretien d'uniformité de croix-plateforme, complique les problèmes de la gestion de configuration et dilue les ressources disponibles pour la technologie de rentabilité. En plus,les outils de développement courants fournissent peu appui pour créer les applications qui changent dynamiquement en réponse aux changements de leur environnement, ou qui doivent partager des données parmi les types de dispositif hétérogènes. Bien que beaucoup de dimensions doivent être considérées en concevant les applications contexte-dépendantes (acteurs, plateformes, environnements, ressources de système, etc.), en cet article nous nous concentrons sur la façon soutenir une conception adressant le changement des plateformes. De nos jours les compagnies doivent considérer mieux les issues épineuses impliquées en concevant des applications soutenant les plateformes multiples. Actuellement, ils adressent souvent différents contextes d'utilisation par différentes applications de construction, même si soutenant les tâches semblables accédant aux données semblables. Les différentes versions des applications devraient différer dans leur présentation, plutôt que d'essayer d'adresser différentes plateformes d'entrée-sortie en remettant à la côte juste les éléments de l'interface utilisateur. Les versions peuvent également différer dans les tâches qu'elles peuvent normalement soutenir - par exemple, les laisux de PDA lisant des revues, le téléphone laisse commander des livres, et le PC de bureau soutient tous les deux. En outre l'ensemble de fonctionnalité soutenu peut changer dynamiquement. Les progrès technologiques résolvent certains des problèmes de machiner des demandes de dispositifs multiples: Les documents de XML (langue de marge bénéficiaire d'eXtensible) soutenus par des stylesheets de XSL (langue d'eXtensible Stylesheet) laissent créer des présentations adaptées aux besoins du client pour différents dispositifs ou utilisateurs. La langue sans fil de marge bénéficiaire (WML) laisse produire des présentations non tributaires du type d'unité pour une gamme de petits dispositifs d'affichage. Les passages sans fil d'Internet traduisent automatiquement des documents de HTML en documents de WML (bien qu'ils peuvent produire des résultats inutilisables s'ils se fondent sur de grands affichages). L'aide de XSL (et technologies relatives) avec la présentation d'interface utilisateur, mais sont limitées par le fait

qui une conception différente d'interaction peut être nécessaire en se déplaçant entre les types de dispositif radicalement différents. Tandis que ces solutions aident avec des parties du problème, elles ne fournissent pas des conseils à niveau élevé pour garantir la qualité à travers des versions multiples des applications.

2. TRAVAIL RELATIF

L'idée fondamentale est celle au lieu d'avoir des demandes séparées de chaque dispositif qui échangent seulement des données fondamentales, il y a une certaine description abstraite et puis un environnement qui peut suggérer une conception pour un dispositif spécifique qui s'adapte à ses dispositifs et contextes possibles d'utilisation. Ceci s'appelle également la plasticité d'interface utilisateur. Ce problème est un défi de roman pour la conception et le développement modèle-model-based des applications interactives. Les potentialités de ces approches ont été adressées d'une façon limitée. Dans le projet d'ESPRIT de GUITARE un générateur d'interface utilisateur a été développé: il prend des modèles de tâche de ConcurTaskTrees (CTT) et produit des interfaces utilisateur pour des applications d'ERP selon des directives de compagnie. Cependant, la génération automatique n'est pas une solution générale en raison de beaucoup, changeant les facteurs qui doivent être pris en considération dans le processus de conception. L'appui semi-automatique est plus général et flexible: Mobi-D est un exemple d'une approche semi-automatique mais il soutient seulement la conception des applications de bureau graphiques traditionnelles. UIML est une langue appareil-indépendante d'interface utilisateur de XML.
Tandis que cette langue est en apparence indépendante du dispositif et du milieu spécifiques utilisés pour la présentation, elle ne tient pas compte du travail de recherches mené à bien dans la dernière décennie aux approches modèle-model-based pour des interfaces utilisateur: par exemple, la langue ne fournit aucune notion de tâche, il vise principalement à définir une structure abstraite. Le consortium de W3C a récemment fourni la première version d'une nouvelle norme (XForms) cette des présents une description de l'architecture, concepts, traitant le modèle, et la terminologie sous-tendante la prochaine génération des formes d'enchaînement, a basé sur la séparation entre le but et la présentation d'une forme. Si elle montre l'importance de séparer le plan d'étude de la présentation concrète, elle accentue également le besoin des modèles signicatifs de soutenir de telles approches. Plus généralement, la question de s'appliquer des techniques modèle-model-based au développement d'UIs pour les ordinateurs mobiles a été adressée à un conceptuel et à une recherche de niveau

3. LA MÉTHODE PROPOSÉE

Il est possible de soutenir le même type de tâches avec différents dispositifs en changeant l'ensemble de techniques d'interaction et de présentation tenant compte des ressources disponibles dans le dispositif considéré. Un autre, une option plus prometteuse est de considérer différents dispositifs également en ce qui concerne le choix des tâches de soutenir (par exemple téléphone plus approprié à l'accès rapide à l'information limitée, aux systèmes de bureau plus appropriés à la lecture rapide par de grandes quantités de l'information). Notre méthode essaye d'adresser de tels problèmes, et se compose d'un certain nombre d'étapes qui permet à des concepteurs de commencer par un modèle envisagé global de tâche d'une application nomade et puis de dériver les interfaces utilisateur concrètes et efficaces pour les dispositifs multiples:

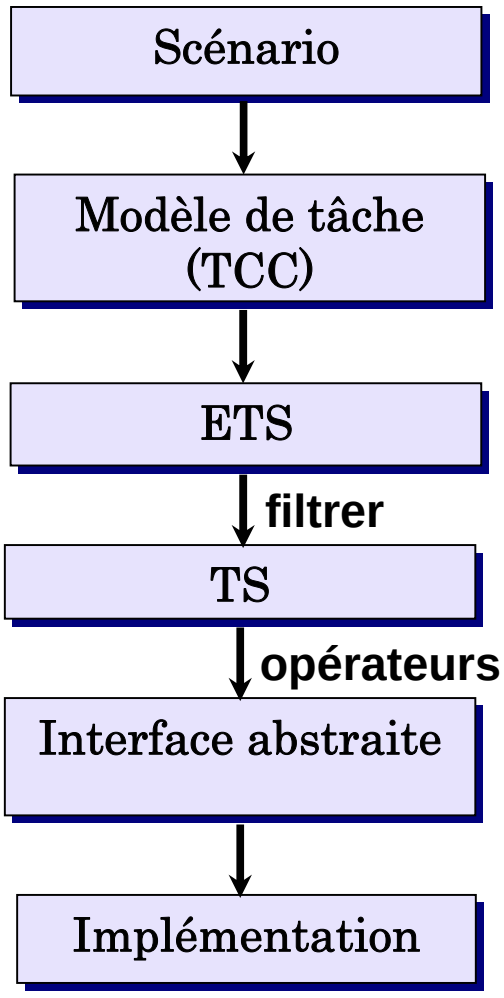
- Modeler à niveau élevé de tâche d'une application de multi-contexte .
- Développant le modèle de tâche de système pour les différentes plateformes ont considéré.

- Du modèle de tâche de système à abstraite d'interface utilisateur.
- La génération d'interface utilisateur.

Nous avons défini des versions de XML de la langue pour modeler de tâche (ConcurTaskTrees), les ensembles permis de tâche et la langue pour modeler les interfaces abstraites et le développement commencé des transformations parmi ces représentations.

4. CONCEPTION DE MODEL DE BASE

Dans notre méthode nous nous concentrons sur les modèles qui peuvent soutenir le développement des interfaces utilisateur tout en préservant la rentabilité, en particulier tâche modèlent indiquer les différentes activités qui sont censées être exécutées dans un système interactif. Des modèles de tâche devraient être développés faisant participer des utilisateurs afin de représenter comment ils préfèrent exécuter des activités.



(Model-based design)

5. L'EXEMPLE ET SON MODÈLE DE TÂCHE

L'exemple considéré est au sujet de la conception d'une demande nomade d'information d' accès de musée. Un scénario possible est un utilisateur qui trouve le musée par hasard tout en surfant l'enchaînement à s/he à la maison propose à quelques amis une visite au musée de marbre. Quand ils arrivent au musée, ils reçoivent un guide mis en application dans PDA qui fournit l'appui audiovisuel pour leur pièce de visite du modèle de tâche de système pour les PDA(Fig. 1).

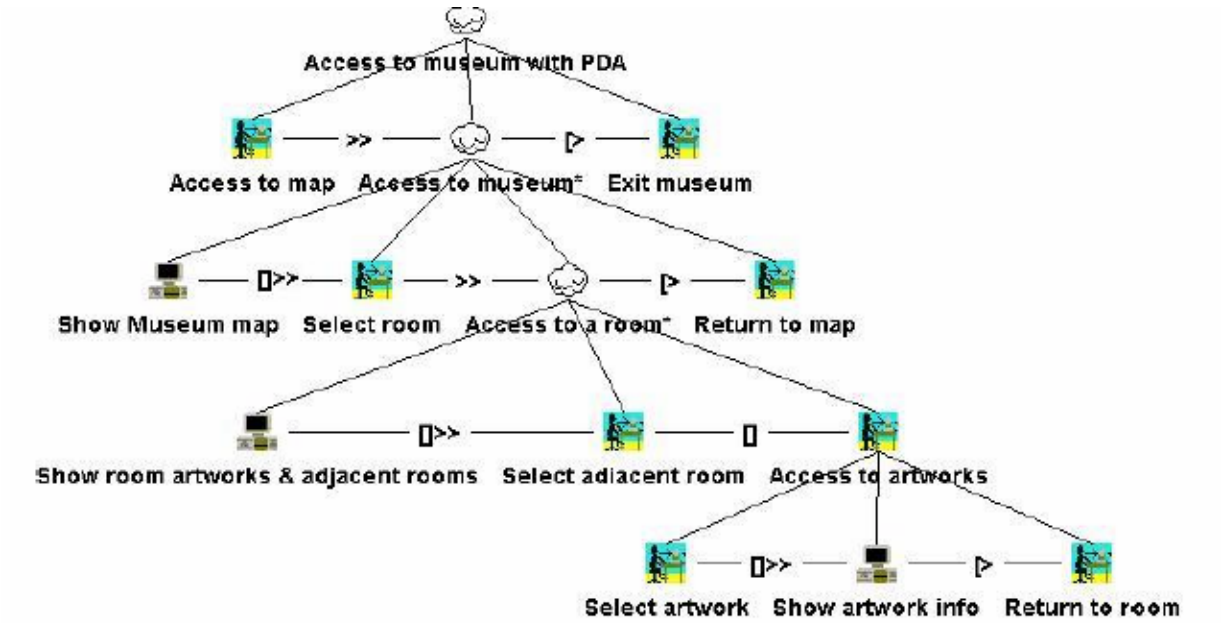


Figure 1. Part of the system task model for the PDA access.

6. DU MODÈLE DE TÂCHE ABSTRAITE À L'INTERFACE UTILISATEUR

Commençant par le modèle de tâche du système, nous visons à identifier les spécifications de l'interface utilisateur abstraite en termes de sa structure statique (la pièce d'" présentation") et comportement dynamique (la pièce de " dialogue")

6.1 La Partie De Présentation

La première étape est de calculer les ensembles permis de tâche (ETSs) selon le modèle de tâche de système. L'outil de CTTE effectue automatiquement l'identification de ces ensembles. Pour l'exemple considéré, les ETSs sont:

ETS1: {Access to map}
ETS2: {Show Museum map, Exit museum}
ETS3: {Select room, Exit museum}
ETS4: {Show room artworks &adjacent rooms, Return to map, Exit museum}
ETS5: {Select adjacent room, Select artwork, Return to map, Exit museum}
ETS6: {Show artwork info, Return to room, Return to map, Exit museum}

6.1.1 L'heuristique pour obtenir un nombre de tâche inférieur place

Une fois qu'ETSs ont été définis, nous devons indiquer quelques règles pour réduire leur nombre en fusionnant deux ETSs ou plus nouveaux en jeux, appelés Task Set ou les raisons de TS.The de faire une telle étape sommes divers:

1. Réduire le nombre initial d'ETSs qui dans certains cas peut être très haut.
2. Maintenir et accentuer dans la même information significative de présentation (comme un échange de données est) même lorsque les tâches impliquées appartiennent différents à ETSs.
3. Éviter les groupes de tâches considérants à plusieurs reprises que tous partagent semblable une structure.

L'heuristique qui ont été identifiés sont la suivante:

_ *H1*: Si deux (ou plus) ETSs diffèrent pour seulement un élément, et ces éléments sont au même niveau lié à un opérateur permettant, ils pourraient être joints ainsi que l'opérateur de commande.

_ *H2*: Si un ETS se compose de juste un élément, il devrait être joint avec un autre ETS qui partage un certain dispositif sémantique.

_ *H3*Si la part d'un certain ETSs la plupart des éléments, ils pourrait être unifiée. Par exemple si tous les éléments communs apparaissent à la droite de l'opérateur de neutralisation, ils pourraient être associés à TS.

_ *H4*: S' il y a un échange d'information entre deux tâches, ils peuvent être mis dans les mêmes TS afin d'accentuer un tel transfert de données.

Par exemple, ETS2 et ETS3 différent par seulement un élément: appliquant H1 ils ont pu être unifiés dans

TS1={*Show museum map, Select room, Exit Museum*}.

En outre, ETS4 et ETS5 part la plupart des éléments: H3 peut être appliqué pour obtenir

TS2={*Select adjacent room, Select artwork, Show Room Artworks & Adjacent Rooms, Return to map, Exit museum*}.

6.1.2 De TSs à abstraite des présentations

L'ensemble de TSs obtenu est l'entrée initiale pour établir les spécifications abstraites d'interface utilisateur se composera d'interactors liés aux tâches de base. Par exemple, la structure de présentation obtenue pour TS2 est:

O (*Show Room Artworks & Adiacent Rooms*, **G** (*Select adjacent room, Select artwork*)) **R** *Return to map* **R** *Exit museum*

C'est parce que l'opérateur de commande accentue le transfert de l'information (indiqué par [ ] > > l'opérateur) entre *Show Room Artworks & Adjacent Rooms* et tous les deux *Select adjacent room* et *Select artwork* tâches (ce qui sont groupés ensemble en raison bien choisi " [ ] " de l'opérateur). Chaque tâche est alternativement dans la relation avec *Return to map* et *Exit museum* parce qu'ils tous les deux apparaissent à la droite d'un opérateur ([>). de neutralisation de la même manière nous pouvons identifier la présentation abstraite associée à TS1 qui est: **O**(*Show museum map, Select room*) **R** *Exit Museum*

6.1.3 La Partie De Dialogue

Quand *static* arrangement de l'abstrait l'interface utilisateur est identifiée, nous doivent indiquer son comportement *dynamique*. À ce but, un rôle important est joué par les prétendues tâches de transition. Pour chaque tâche T réglé, un ensemble de règles (*transition_task, next*_TS)

devrait être donné dont la signification est:quand le *transition_task* est exécuté, l'interface utilisateur abstraite passe de T à *next*_TS.
Par exemple, TS1 a deux tâches de transition: *Select Room* et la tâche *Exit Museum*. Pour exprimer cela par l'intermédiaire de tâche *select room* l'interface abstraite passe de TS1 à TS2 nous pouvons employer la règle suivante:

<*task_set* **TS1** /*task_set* >
<*behaviour*>
<*rule*>
<*transition_task* **Select room** /*transition_task*>
<*next_TS* **TS2** /*next_TS*>
</*rule*>
</*behavior*>

6.2 De l'interface utilisateur abstraite à son exécution

Une fois les éléments de l'interface utilisateur abstraite ont été identifiés, chaque interactor doit être tracés dans des techniques d'interaction soutenues par la configuration de dispositif particulière considérée (logiciel d'exploitation, toolkit, etc.).
Une exécution possible pour la présentation correspondant à TS2 est montrée dans fig. 2, supposée que la salle courante est au sujet d'Archaeology romain.

Figure 2. One presentation of the PDA user interface.

Fig. 3 montre un exemple de différente présentation d'information visuelle entre un système de bureau et un téléphone de WAP: comme vous pouvez voir de la partie inférieure de l'image, alors que dans le système de bureau il est possible d'avoir des détails du travail (titre, type, description, auteur, matériel, et date de création), dans l'interface de WAP nous pouvons avoir seulement de basses images de résolution laissant avoir juste une idée approximative de ce qu'est l'oeuvre d'art, ainsi que l'indication du titre et de la section relative de musée. La partie supérieure de fig. 3 montre comment dans l'outil de CTTE il est possible d'indiquer pour chaque tâche que chaque plateforme (PDA, système de bureau, cellphone ou d'autres plateformes) peut soutenir différents ensembles d'objets manoeuvrés pendant l'exécution de tâche.
Par exemple "*Show artwork info*" la tâche considérée dans l'image, le concepteur a indiqué que le titre est disponible sur toutes les plateformes considérées (tous les checkboxes marqués "*PDA*", "*Desktop*", et "*Cellphone*" ont été choisis), considérant que la description est disponible seulement pour le Desktop.

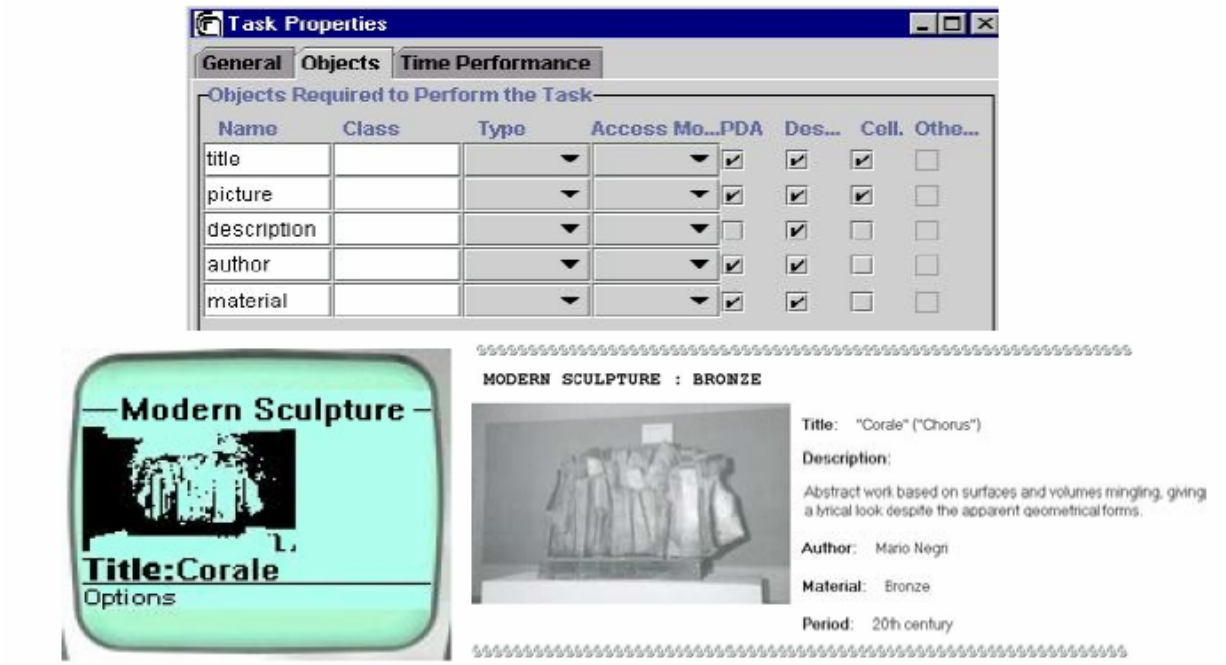


Figure 3. Exemple de différentes présentations d'une oeuvre d'art.

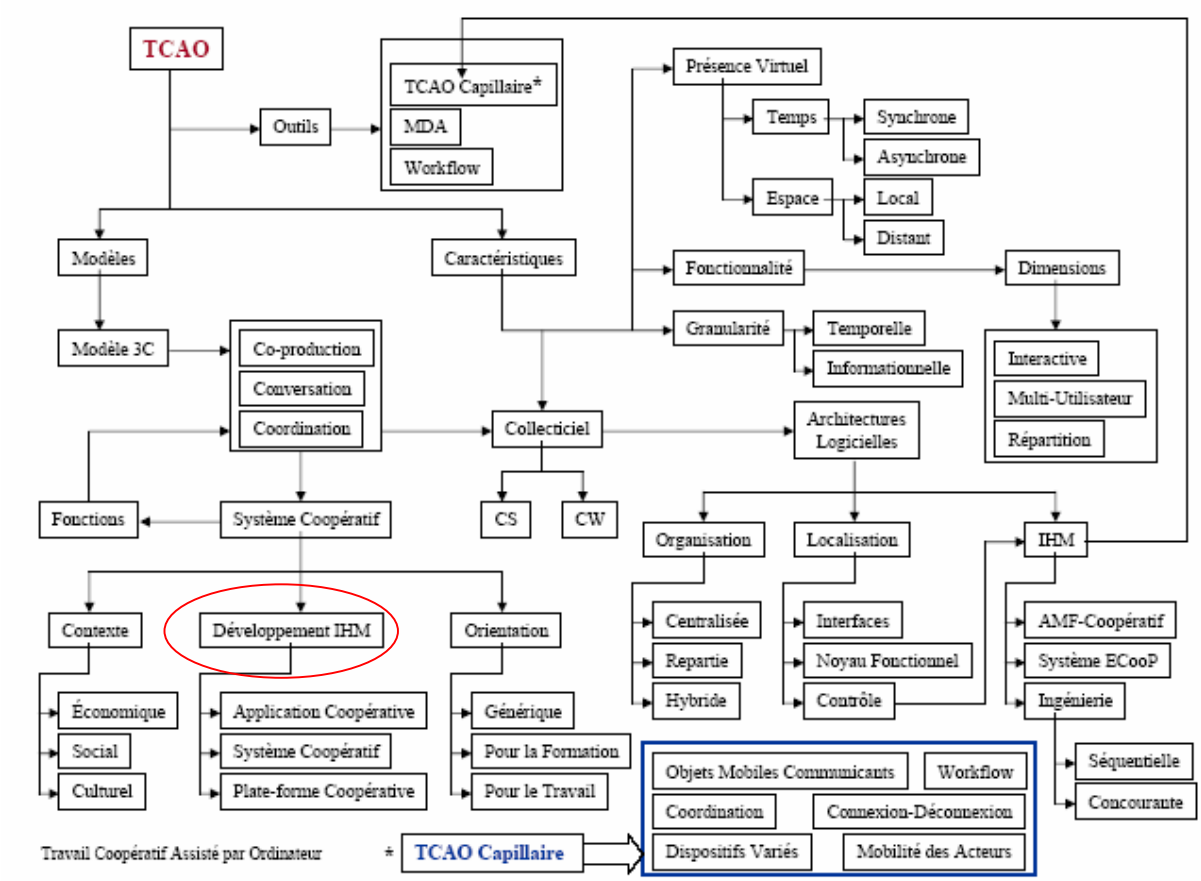
Cette information a été employée pour produire de différentes interfaces utilisateur pour le téléphone cellulaire et le système de bureau: Fig. 3 prouve que le titre est disponible sur les les deux les plateformes tandis que la description est disponible seulement pour le système de bureau.

7. CONCLUSIONS

Nous avons décrit une méthode pour concevoir des applications à plusieurs dispositifs et nomades à partir de leur modèle de tâche et d'employer un certain nombre d'abstractions dans le cycle de conception d'obtenir des applications finales capables soutenir efficacement les activités des utilisateurs par de divers dispositifs consultés dans différents contextes. L'application des aides de méthode maintient un à niveau élevé de l'uniformité à travers les interfaces utilisateur multiples développées. Pendant que l'appui d'outil dans la phase de génération d'interface utilisateur est encore limité, des travaux futurs sont projetés pour le prolonger et pour créer un environnement semi-automatique complet appliquant la méthode présentée.

IV – Carte de concept

La thématique vers laquelle convergent les deux articles touche le niveau IHM dans la panoplie du domaine de système coopératif. Mais cette partie qui constitue une infime partie dans ce domaine est capable à elle seule de créer tout un domaine très large.



Cependant, il est à noter que la complexité de développent d'application de TCAO avec une belle interface à la fois intelligente et qui permet de s'adapter à chaque caractéristiques de ces utilisateurs tout en leur permettant de travailler en coopération prenant en compte toutes les contraintes qui en découlent, reste un défi pour les concepteurs de ce type d'application et les chercheurs.

RÉFÉRENCES

- Brusilovsky, P.: 1996, Methods and techniques of adaptive hypermedia. User Modeling and User-Adapted Interaction 6(2-3), 87-129
- Brusilovsky, P.: 2001, Adaptive Hypermedia. User Modeling and User-Adapted Interaction 11, 87-110
- Calvary,G.,Coutaz,J.,and Thevenin,D.,A Unifying Reference Framework for the Development of Plastic User Interfaces, Proc.of EHCI'2001,Springer-Verlag,2001.
- Puerta,A.R.,A Model-Based Interface Development Environment, IEEE Software,July/August 1997,pp.40-47.