La plasticité en interaction Homme-Machine

David Thevenin

Laboratoire CLIPS-IMAG, Equipe IIHM
BP 53 38041 Grenoble Cedex 9 France
david.thevenin@imag.fr

RÉSUME

Les nouvelles technologies ubiquitaires d'accès et de traitement de l'information ouvrent une nouvelle voie de recherche en Interaction Homme-Machine : le développement et le maintient des interfaces utilisateurs sur des systèmes dont les caractéristiques physiques varient. Dans cet article, nous présentons notre approche basée sur la propriété de plasticité, une forme d'adaptation. Nous présentons un processus, de type "Model Based", d'aide à la conception et génération d'interface garantissant cette propriété. Nous illustrons notre approche par l'application d'un sous ensemble du processus pour la conception de l'application "Minimal Media Space".

MOTS CLES : Plasticité, adaptation, adaptativité, adaptabilité.

INTRODUCTION

Le besoin d'un accès ubiquitaire à l'information (au bureau, chez soi, dans le train, etc.), le succès des ordinateurs de poche ou des téléphones portables, les avancées dans les réseaux sans fil, offrent de nouvelles voies de recherche pour la communauté en IHM. Ces nouvelles formes d'interaction se heurtent à de nombreuses difficultés parmi lesquelles le développement et le maintient des interfaces homme-machine sur chaque système. Or la portabilité en exécution, telle que la propose Java, n'est pas une condition suffisante : pour être utilisable, l'interface utilisateur doit être adaptée aux dispositifs d'interaction, par exemple à la taille de l'écran ou à la présence ou non d'un clavier. Dans ce travail, nous introduisons la nouvelle propriété de "plasticité" d'une interface, un type particulier d'adaptation, et proposons un processus de conception et de développement assurant le respect de cette propriété.

En première partie de ce document nous présentons la propriété de plasticité, un sous ensemble du problème général de l'adaptation d'une interface. Dans la deuxième et troisième partie, nous présentons le processus et les requis à la conception d'une interface respectant la plasticité. Enfin en dernière partie, nous présentons une première mise en oeuvre de notre processus.

LA PLASTICITÉ D'UNE INTERFACE

En IHM, l'adaptation d'une interface est caractérisée par deux propriétés : l'adaptabilité et l'adaptativité. L'adaptabilité est la capacité d'une interface à être modifiée par l'utilisateur, et l'adatativité est la capacité d'une interface à se modifier automatiquement, sans action explicite de l'utilisateur. Nous définissons l'espace d'adaptation, par les axes suivants (c.f. figure 1): la *cause* de l'adaptation, l'*objet* qui s'adapte, *quand* l'adaptation se produit et l'*acteur* qui déclenche l'adaptation.

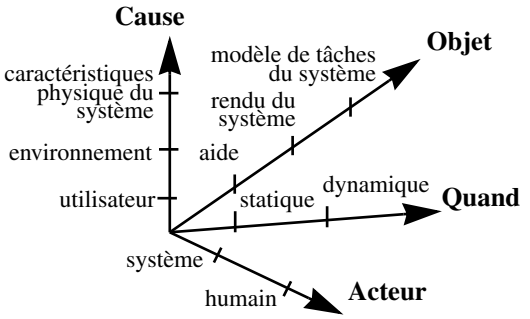

Figure 1: Espace pour l'adaptation d'une interface.

La plasticité d'une matière est sa capacité à se déformer suivant des contraintes, sans se casser. De la même manière, nous définissons la *plasticité* d'une interface, comme sa capacité à s'adapter (se déformer) aux contraintes du matériel ou de l'environnement, en *restant utilisable* (sans se casser). De ce fait, une interface spécifiée pour s'adapter aux variations du matériel, permet *de minimiser le coût de développement et de maintenance*. Dans l'espace d'adaptation proposé plus haut (c.f. figure 1), la plasticité est définie comme suit :

- selon l'axe des *causes*, c'est une adaptation aux caractéristiques physiques de la plate-forme visée, ou une adaptation aux contraintes environnementales;
- selon l'axe du *quand*, l'adaptation est statique, c'est-à-dire que les contrainte peuvent être fixées à la conception, et l'interface est générée en fonction, ou l'adaptation est dynamique c'est-à-dire qu'au cours de l'exécution, l'interface est adaptée en fonction des variations de contraintes;
- selon l'axe des *objets*, la plasticité est une adaptation de l'arbre de tâche et/ou des techniques de rendu;
- selon l'axe de l'*acteur*, le système ou un humain (le concepteur ou l'utilisateur final) provoque l'adaptation.

PROCESSUS DE CONCEPTION ET GÉNÉRATION

Notre objectif peut être défini par le slogan suivant : "Spécifier une fois, générer plusieurs fois". Pour réaliser cet objectif, nous proposons un processus de conception

et de génération d'interface Homme-Machine respectant la propriété de plasticité.

Le processus présenté est inspiré des systèmes du type "Model-Based Interface Development Environments" (MB-IDE) (Nous les appellerons Générateurs Orientés Modèles ou GOMs). Le concept de GOM s'appuie sur l'expression déclarative de la sémantique de l'application et des connaissances nécessaires à la spécification de l'apparence et du comportement d'un système interactif [9]. Le but d'un GOM est d'encapsuler le plus d'informations possible dans des modèles pour faciliter le développement d'une interface, voire l'automatiser. Cette approche est assez sophistiquée puisqu'elle a la prétention de couvrir non seulement tous les composants du modèle Arch mais aussi d'injecter d'autres connaissances spécifiées dans des modèles tels que le modèle de l'utilisateur, de l'environnement, du comportement, des tâches, d'ergonomie, de la plate-forme. Parce que les GOMs ne respectent pas tous les modèles nécessaires à l'implémentation de la plasticité, entre autre le modèle de la plate-forme, nous avons décrit notre propre processus. Son architecture est présentée par la figure 2.

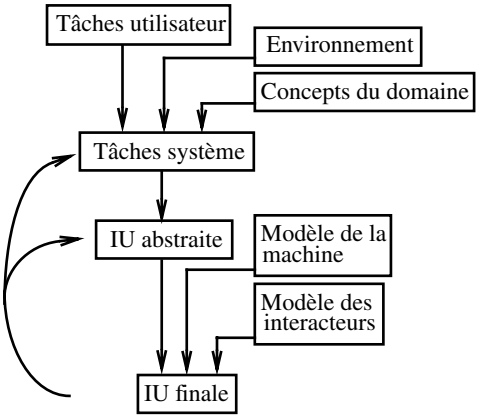

Figure 2: Espace de conception et de génération

Le modèle de l'arbre de tâche utilisateur est une transcription formelle ou semi-formelle de l'activité de l'utilisateur dans le mode réel. Ce modèle est réalisé par un spécialiste en sciences cognitives.

Le modèle de l'environnement décrit le contexte d'utilisation. Ce modèle n'est pas clairement défini dans la littérature. Nous y spécifions les objets qui peuvent avoir un impact ou sont en relation avec la réalisation d'une tâche. Ces objets peuvent être une personne, un lieu, des événements. Par exemple l'usage en un lieu public va forcer l'utilisation du mode vibreur d'un téléphone mobile plutôt qu'une sonnerie.

Le modèle des concepts du domaine décrit les données en relation avec la réalisation des tâches de l'utilisateur. Ce modèle est une description d'un sous-ensemble du noyau fonctionnel.

Le modèle de tâches système décrit comment les tâches utilisateurs sont implémentées dans le système. Il correspond au contrôleur de dialogue du modèle Arch mais augmenté d'information. Dans ce modèle, nous n'allons pas jusqu'à spécifier l'interaction, comme dans UAN, puisqu'elle dépend des dispositifs d'interactions. Nous y reviendrons dans "Modèle de l'arbre de tâche système".

Le modèle de l'interface utilisateur abstraite est la première étape vers la génération de l'interface finale. A partir de l'arbre de tâches système, sont déterminées les informations à représenter et comment elles sont regroupées (sous forme de Présentation [1]).

Le modèle de la machine décrit les possibilités de la plate-forme visée : performances de calcul, performances de communication, dispositifs physiques d'interaction. Pour l'instant nous ne décrivons que les caractéristiques des dispositifs physiques d'interaction : par exemple la surface d'affichage pour un écran.

Le modèle des interacteurs contient une spécification des interacteurs (Widget) disponibles. Ce modèle est vu plus en détail dans "Modèles des interacteurs et de la machine".

L'interface utilisateur finale est la spécification de l'interface finale. Elle est générée automatiquement au semi-automatiquement à partir des modèles précédents.

REQUIS ET HEURISTIQUES

L'étude de l'application "agenda électronique" existant parallèlement sur le Pilot et sur Mac0S/Windows, nous a permis d'extraire des requis et heuristiques. Ces requis et heuristique, dont la liste est non-exhaustive, concernent chacun des modèles vus précédemment.

Modèle de l'arbre de tâche utilisateur

Les concepts du domaine rentrant dans la réalisation d'une tâche doivent être ordonnés en fonction de leur importance dans la réalisation de celle-ci. Par exemple, dans l'application "agenda électronique" le système affiche une date. Cette date est composée du jour dans la semaine (mardi, jeudi, ...) du jour dans le mois (5 ième), du mois (juin ou 6) et de l'année. Le jour dans le mois, le mois et l'année sont considérés comme des informations de première importance dans la tâche de "prise de connaissance de la date". Alors que le jour dans la semaine est considéré comme une information de deuxième importance. [8] ou [5] présentent une modélisation des tâches prenant compte de ce niveau de distinction : "information requise" et "information nécessaire".

Le niveau de première ou de seconde importance, permet de justifier des choix lors du rendu d'une présentation.

Une heuristique de génération pourrait être qu'un concept de première classe devra être directement accessible alors qu'un concept de seconde classe pourra, si la surface d'affichage n'est pas suffisante par exemple, être rendu accessible par une indirection (c.f. paragraphe suivant). Une autre heuristique serait qu'un concept de second classe, parce que redondant avec un concept de première classe, sera supprimé. Dans l'exemple de "l'agenda électronique", sur une station, une date est de la forme "Mercredi 15 octobre 1999" alors que sur le Pilot, elle est de la forme "15 Oct 99". Le jour dans la semaine est considéré comme un concept de seconde classe.

Modèle de l'arbre de tâche système

Dans le contexte de la plasticité, l'accès à un élément de deuxième classe peut nécessiter l'ajout d'une tâche articulatoire. Une tâche articulatoire est tâche système induite par la méthode de rendu et par les caractéristiques de la plate-forme. Par exemple, dans l'agenda, se positionner à une date précise nécessite la sélection de celle-ci dans un calendrier. Dans la version Station le calendrier est directement accessible à l'écran, alors que dans la version Pilot, le calendrier est rendu accessible par une commande. Cette commande "Aller à" est une tâche articulatoire introduite dans l'arbre de tâche du Pilot par manque de surface d'affichage.

Modèles des interacteurs et de la machine

La plasticité dénote entre autre, la capacité de changer de rendu, en particulier d'interacteurs, pour représente le même concept. La figure 3 montre, pour l'exemple de l'agenda, deux interacteurs distincts utilisés pour changer de mode de visualisation des rendez-vous. Le modèle des

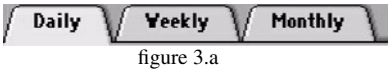

figure 3.a


figure 3.b

Figure 3: Deux intéracteurs différents pour repésenter le même concept. (a) version Station avec des onglets, (b) version Pilot avec des icones.

interacteurs permet de faire un choix approprié en fonction des contraintes physiques. Le modèle doit spécifier pour tout interacteur, une description des données représentables, permettre d'évaluer le coût de rendu et son adéquation. Le choix est fait par une résolution de la fonction *f(données, coûts, adéquation)*. Il s'agit d'un système de résolution de contraintes.

La description des données représentables fournit l'information nécessaire à la mise en correspondance d'un concept à représenter et d'un ensemble d'interacteurs disponibles sur la plate-forme visée. [6] ou [7] ont ouvert la voie pour des systèmes non interactifs. [4] s'intéresse au problème dans le cadre de la génération d'interfaces à manipulation directe.

Le modèle d'un interacteur spécifie le coût d'utilisation de celui-ci. Un interacteur peut être vu comme une modalité et donc être spécifié par le couple *m*=<*r, d*> [3]. Dans le contexte de la plasticité, nous augmentons *m* par les coût Cr et Cd respectivement liés à *r* et *d*.

Enfin la modélisation doit spécifier l'adéquation d'un interacteur pour une tâche donnée. Elle a pour but d'aider au choix de l'interacteur approprié, lors de la génération, et de quantifier une éventuelle dégradation de l'utilisabilité du système. Le problème est traité dans [1] ou [2] pour des système multimodal.

Dans ce qui suit, nous présentons l'application d'un sous ensemble de ces modèles sur MMS.

MMS :

MMS est un mediaspace qui montre le niveau d'activité des membres de notre équipe. Ce niveau d'activité est déterminé par des statistiques sur l'utilisation du clavier et de la souris. Pour l'instant l'application ne fonctionne que sur des stations de bureau. Nous simulons donc différentes surfaces d'écran en faisant varier la taille de la fenêtre de l'application.

Dans cet exemple, nous supposons faire la génération d'une interface capable de s'adapter à la surface d'un écran. Les contraintes physiques sont donc limitées à un couple <x, y> représentant la largeur et hauteur d'un écran en pixels.

L'interface est constituée de trois PUs représentant : un texte indiquant le but du système, une commande pour quitter et une liste de taux d'activités. Les trois PUs sont de première classe. La liste des taux d'activité est composée, pour n personnes connectées, de n noms de personne, de n noms machine et de n taux d'activité. Nous définissons que le nom de personne et le taux d'activité sont de premières classe, alors que le nom de machine est de deuxième classe. La figure 4 montre le flux de données spécifiant les trois premières PUs. La figure 5 montre le flux de données spécifiant un élément de la liste des taux d'activité.

Concept	Direct.	Type données	Card.	Domaine
but du système	sortie	ASCII[]	1	{Activités ... MMS}
liste des activités	sortie	élement d' activité	n	Indéfini
cmd "Quitter"	Entrée	ASCII[]	1	{Quitter}

Figure 4: PU de MMS

Concept	Direct.	Type données	Card.	Domaine
nom personne	sortie	ASCII[]	1	Indéfini
nom station	sortie	ASCII[]	1	Indéfini
taux d'activité	sortie	entier	1	[0, 100], pas =1

Figure 5: PU pour un élément de la liste des activités

Un interacteur est spécifié par son empreinte graphique, sont coût en surface d'affichage, et le flux d'informations qu'il peut représenter. La figure 6 montre un exemple pour un interacteur de type "cases à cocher". Les interacteurs respectant les éléments de base des PUs sont :

- le champ de texte non éditable pour le but du système, le nom d'une personne connectée et pour le nom de la machine d'une personne connectée;
- le bouton pour la commande quitter;
- un entier, une barre de type histogramme ou un graphe avec historique pour le taux d'activité d'une personne.

Direction	Type de donnée	Cardinalité	Domaine
Sortie	entier	1	[0..n], pas x

Figure 6: Exemple d'une spécification pour un interacteur de type bar d'histogramme.

L'interface s'adapte automatiquement en fonction de la surface de la fenêtre imposé par l'utilisateur. L'adaptation est rendue dynamique par un codage architecturé des PUs et un système de résolution de contrainte. La figure 8

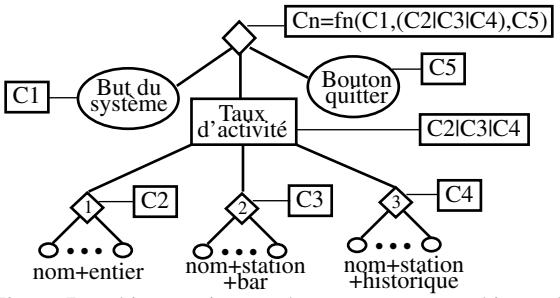

Figure 7: architecture interne des composants graphiques de l'interface et coût associés

montre la représentation interne utilisée pour le rendu. Un noeud en forme d'ovale, représente un interacteur. A ce noeud est associé le coût affichage de l'interacteur. Un noeud en forme de losange représente un groupement d'interacteurs et une somme de coûts d'affichage en fonction du positionnement des interacteurs. Un noeud en forme de rectangle représente un module faisant le choix optimal en fonction des contraintes qui lui sont fournies et les coûts de ses fils. La présentation finale est déterminée par propagation des coûts. La figure 8 montre les trois interfaces produites dynamiquement.

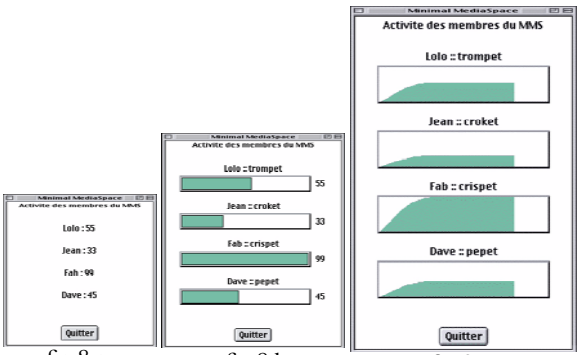

fig 8.a fig 8.b fig 8.c

Figure 8: Variation dynamique du rendu de l'interface.
a) Pour une petite fenêtre, seulement les noms de personne et leur niveau d'activité sont observables. L'activité est représentée par un entier. Le nom de machine, étant de deuxsème classe, n'est pas rendu observable.
b) Pour une fenêtre de moyenne taille, le taux d'activité est représenté avec une bare de type histogramme.
c) Pour une grande fenêtre, l'activité est représentée par un graphe avec historique.

CONCLUSION

Nous avons présenté la propriété de plasticité, qui caractérise la capacité d'une interface à s'adapter aux ressources physiques du système et à l'environnement. Pour assurer cette plasticité, nous avons présenté un processus d'aide à la conception et génération d'interfaces, inspiré des MB-IDEs. Enfin, à travers l'application MMS, nous avons montré la faisabilité de notre approche, par l'application d'un sous ensemble du processus.

Nos premiers modèles trop simples, nécessitent d'être compléter et mieux formalisé. Enfin nous travaillerons sur l'expression d'un ensemble de méthodes et heuristique aidant à la génération.

REFÉRENCES

1. F. Bodart, A.-M. Hennebert, J.-M. Leheureux, J. Vanderdonckt, Computer-Aided Window Identification in TRIDENT. Dans INTERACT'95 pp. 331-336.
2. N.O.Bernsen, Foundations of Multimodal Representations; a Taxonomy of Representational Modalities. Interacting with Computers, 1994, 6, pp. 347-371.
3. J. Coutaz, L. Nigay, D. Salber, A. Blandford, J. May, et R. Young, Four Easy Pieces for Assessing the Usability of Multimodal Interaction: The CARE Properties. Dans INTERACT'95, pp. 115-120.
4. M. Fischer, A Framework for generating Spatial Configurations in User Interfaces. Dans DSV-IS'97, Springer Computer Science, pp. 225-241
5. P. Johnson, S. Wilson, P. Markopoulos, Y. Pycock, ADEPT-Advanced Design Environment for Prototyping with Task Model. Dans INTERCHI'93, pp. 56
6. J. Mackinlay, Automating the Design of Graphical Presentations of Relational Information", ACM Tr. on Graphics,5,2, pp.110-141.
7. S. Roth, J. Mattis, Data Characterization for Intelligent Graphics Presentation. Dans CHI'90, ACM, pp. 193-200.
8. A. Sutcliffe, Task-Related Information Analysis. Int. J. of Human-Computer Studies,(1997) 47, pp. 223-257.
9. UIST'94, Model-Based User Interfaces. What are They and Why Should We Care? UIST Novembre 1994, pp. 133-135.