Le problème de la plasticité dans la conception d'IHM

Alexandre Demeure

Laboratoire CLIPS-IMAG, équipe IIHM
385, rue de la bibliothèque – BP 53
38041 Grenoble Cedex 9, France
alexandre.demeure@imag.fr

RESUME
Cet article décrit mes travaux de thèses, ceux-ci portent sur la plasticité des IHM et plus particulièrement sur la conception et l'exploitation de tels IHM.

MOTS CLES : plasticité, modèles d'évolution, interfaces NG, capitalisation.

ABSTRACT
This paper describes my thesis works, it is about HCI plasticity and more specifically about conception and use of such softwares.

CATEGORIES AND SUBJECT DESCRIPTORS: H.5.2 [GUI].

GENERAL TERMS: Design

KEYWORDS : plasticity, evolution models, NG interfaces, capitalisation.

INTRODUCTION

La plasticité d'une interface dénote sa capacité à s'adapter aux variations du contexte d'usage en termes d'utilisateur, de plate-forme et/ou d'environnement dans le respect de son utilisabilité. L'*utilisabilité* est un facteur qualité logiciel [4]. Il relève de l'acceptation d'un système [5]. Il se réfère à l'adéquation d'un système interactif aux capacités de l'utilisateur. L'utilisabilité complète l'*utilité*, relative à l'adéquation du système interactif aux besoins des utilisateurs, pour caractériser, de façon plus globale, la serviabilité ou « usefulness » du système (Figure 3). La serviabilité se réfère à la faculté du système à permettre à l'utilisateur d'atteindre ses buts[1]. L'utilisabilité en qualifie les facilités d'apprentissage et d'appropriation, l'efficacité à l'usage, la robustesse aux erreurs et enfin, de façon plus subjective, la convivialité ou le plaisir concrètement ressenti à l'usage. Aussi, l'utilisabilité ne se limite pas à des critères de performance dans l'accomplissement de tâches. Elle se réfère, de façon plus générale, à la satisfaction de buts personnels et collectifs[2] (Gilmore, 1995).

L'objet de ma thèse est d'étudier certains points relatifs à la conception de systèmes interactifs plastiques. En particulier il s'agira de formaliser un modèle explicite d'évolution du système et de proposer une méthode de classification des systèmes interactifs afin de favoriser la réutilisation et l'interopérabilité des systèmes interactifs à la conception et à l'exécution.

LE MODELE D'EVOLUTION

L'adaptation du système interactif peut être plus ou moins profonde selon qu'elle porte sur une réorganisation des concepts et des tâches, des interfaces abstraite, concrète ou finale. L'adaptation du système peut ne concerner qu'une sous partie de ce dernier, on parle dans ce cas de l'adaptation d'éléments d'interactions. Dans tous les cas nous identifions quatre classes de stratégies :

- l'adaptation par polymorphisme. C'est une stratégie qui conserve l'élément mais en change la forme. Ce changement de forme peut s'opérer à tout niveau de réification selon trois cardinalités, 1-1, 1-N, N-1, selon que la forme est remplacée par une autre (cardinalité 1-1), N autres (cardinalité 1-N) ou que N formes sont agrégées en une seule (cardinalité N-1) ;
- l'adaptation par substitution. Par opposition au polymorphisme, cette stratégie ne préserve pas l'élément. Elle la remplace par un ou N autres. On distingue trois types de substitution selon leur cardinalité : la substitution 1-1 consiste à remplacer l'élément par un autre ; la substitution 1-N remplace l'élément par N autres ; la substitution N-1 remplace l'élément et N-1 autres par un unique autre *;*
- l'adaptation par ajout. Un élément est ajouté au système interactif. C'est le cas typiquement de la représentation multiple d'un même concept, sollicitée pour de l'insistance temporaire ;
- l'adaptation par suppression : c'est la duale de l'ajout. Un élément est supprimé parce que l'information n'est plus critique ou que la tâche ne fait plus sens dans le contexte d'usage courant.

Les stratégies sont déployées selon des politiques. Ces politiques dépendent de l'autonomie accordée à l'élément dans sa prise en charge de l'adaptation : reconnaissance de la situation, calcul et mise en oeuvre de la réaction. Pour chacun de ces aspects, un degré

[1] « Usefulness is the issue of whether the system can be used to achieve some desired goals » (Nielsen, 1993) p 24.
[2] « Redefine usability as successful fulfillment of user and organisational goals, rather than rapid, error-free performance of unit-tasks » (Gilmore, 1995) p 177.

d'autonomie est accordé à l'élément. Ce degré varie de 0 à 1, donnant lieu à quatre clases de politiques :

- la politique *non concertée externe* prive l'élément de système interactif de toute autonomie. L'évolution est calculée et mise en oeuvre par un tiers (un autre élément ou l'infrastructure d'exécution) ;
- à l'opposé, la politique *non concertée interne* confère à l'élément une autonomie maximale. Il décide seul de la réaction à mettre en oeuvre et l'applique en pleine autonomie ;
- les politiques *concertées* correspondent à une autonomie partielle : une négociation s'établit entre l'élément et un tiers (un autre élément ou l'infrastructure d'exécution) pour la prise en charge de l'adaptation. Tandis qu'une version optimiste autorisera l'élément à appliquer la décision sans accord préalable du tiers, une version pessimiste requerra cet accord avant toute application. La version optimiste s'expose à devoir annuler, en cas d'erreur, les mesures mises en oeuvre.

Le choix de la politique se fera au regard de critères tels que la performance par exemple. Seule la politique non concertée interne est aujourd'hui mise en oeuvre. Le modèle d'évolution est distribué au sein des différents agents composant la hiérarchie. C'est un ensemble de règles du style *« Si condition Alors Proposer(réaction, attributs)* » où :

- la *condition* porte sur le contexte d'usage ;
- la *réaction* spécifie de façon plus ou moins directive la réaction à mettre en oeuvre. Par exemple, « *migrer un composant sur PDA* », « *migrer la télécommande sur PDA* » ou « *remplacer le composant télécommande PC par le composant logiciel télécommandePDA-Version3.4 à exécuter sur le PDA* ». La réaction est aujourd'hui décrite par la fonction logicielle permettant de l'exécuter. Elle est assortie d'un texte en langue naturelle qui en décrit l'effet ;
- les attributs décorent la proposition d'une force, d'un qualificatif et indiquent l'auteur de la proposition (utilisateur/interacteur). La force exprime le caractère conseillé ou déconseillé de la proposition ; le qualificatif en exprime la nature « De convenance » ou « De survie ». Une proposition de convenance relève du confort de l'utilisateur alors qu'une proposition de survie engage le caractère opérationnel du système interactif. C'est le cas typiquement d'un logiciel sur PDA dont la batterie faiblit et dont la migration s'impose.

Ces mécanismes d'évolution supposent :

- une base de données capitalisant les interacteurs disponibles. Ces interacteurs peuvent être de granularité variable. Ils peuvent se limiter aux interacteurs classiquement disponibles dans les boîtes à outils actuelles (par exemple, la notion de choix qui, cette fois, disposera de différentes présentations, en particulier, les boutons radio et menu déroulant) ou s'étendre à des interacteurs métier (par exemple, des télécommandes de diaporama ou même des logiciels de présentation). Ils sont capitalisés à différents niveaux de réification conformément au cadre de référence (liens concepts-tâches, interfaces abstraite, concrète et finale) ;
- des mécanismes de recherche d'information permettant l'exploitation de cette base de données. Ces mécanismes supposent une description de chaque composant capitalisé.

LE GRAPHE DES DESCRIPTIONS

Le GDD organise les descriptions des logiciels, quelque soit leurs niveaux d'abstractions, ces descriptions forment les nœuds du graphe. Il établi les liens qui relient ces logiciels, ces liens peuvent relatifs à une notion d'héritage ou de composition. Nous illustrons cela dans la figure 3 en fin d'article. La description contenue dans chaque nœud comprend :

- une description des ports de communication en WSDL, un langage standard de fait dans le domaine des web-services.
- Un lien vers un exécutable ou un programme source si on a à faire à une version implémenté ou semi implémenté (squelette).
- Une description de l'interface dans un langage pivot à définir.

Le graphe des descriptions est utile en conception et à l'exécution.

En conception, il incite le concepteur à situer son logiciel par rapport à des logiciels existants, déjà référencés. Il favorise une vision large consistant à percevoir un logiciel comme une partie d'un ensemble interconnecté de composants, classés de façon rationnelle. Il soulève typiquement des questions: comment et où s'insère ce nouveau logiciel ? Quelles relations entretient-il avec les autres logiciels ? Quelles parties devrait-il rendre les plus indépendantes possibles pour en permettre la réutilisation ? Et enfin, quelles parties des autres applications pourraient être réutilisées dans ce logiciel ?

A l'exécution, le GDD permet de trouver des remplaçants à un composant donné. Ceci est utile notamment lorsqu'on cherche à migrer une application d'une plateforme vers une autre. Il peut par exemple s'agir de migrer une télécommande d'un PC vers un PALM, auquel cas il faut changer le logiciel lui-même. Dans le cas où la

télécommande de départ est spécifique à un logiciel bien particulier (figure 3), il se peut qu'aucune version n'est été prévue pour le PALM, auquel cas il faudra se rabattre sur une télécommande plus générale (figure 3).

UNDE BOITE A OUTILS ADAPTEES A LA DISTRIBUTION DES INTERFACES

La plasticité des interfaces pose aussi le problème de la distribution de celles-ci sur plusieurs ordinateurs simultanément. C'est notamment le cas dans des projets comme (Lachenal, 2003). Il s'agit idéalement de la mettre en œuvre sans surcoût pour le programmeur et l'utilisateur. D'un point de vue fonctionnel, la répartition pose cinq problèmes :

- α : Découverte des ressources d'interaction, dispositifs de saisie, de pointage, surfaces ou espaces 3D. d'affichage, éventuellement position des utilisateurs.
- β : Mise en relation des espaces logiques et physiques. L'espace logique est un modèle de l'espace dans lequel les interacteurs prennent place, il s'agit classiquement d'un plan mais ce pourrait être un espace 3D. L'espace physique est un modèle de l'espace qui nous entoure, généralement 3D. L'utilisation typique va consister à plaquer des portions d'espaces logiques sur des portions d'espaces physiques (écrans, murs, etc.).
- χ : Expression des relations entre utilisateurs, dispositifs et surfaces/espaces. Il s'agit d'exprimer qu'un dispositif correspond à un utilisateur donné ou qu'une surface est disposée à gauche d'une autre.
- δ : Répartition des éléments présents dans l'espace logique sur l'espace physique. Il s'agira typiquement de distribuer un interacteur sur plusieurs écrans.
- ε : Outils graphiques, offrant primitives et interacteurs, ceux qui peupleront l'espace logique.

Les fonctionnalités précédemment décrites ont des dépendances que nous explicitons dans la figure 1. Une dépendance est symbolisée par un côté commun, une flèche indique le sens de la dépendance.

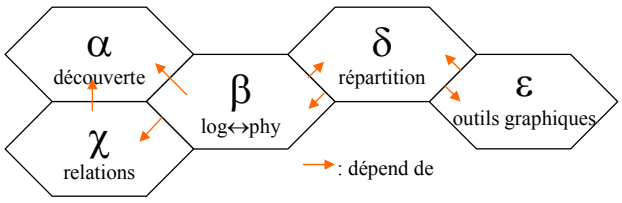


Figure 1: Décomposition fonctionnelle d'une boîte à outil assurant la distribution.

Observons les influences entre fonctionnalités.

- α→(β et χ) : β et χ dépendent de α qui les informe des ressources d'interaction détectées. β en a besoin pour connaître l'espace. χ en a besoin pour calculer les relations entre les entités physiques, par exemple le fait qu'une souris soit proche d'un clavier.
- χ→β : La position des surfaces entre elles dans l'espace physique peut permettre de calculer des mises en correspondance entre les espaces logique et physique. Par exemple, si un écran E2 se trouve à droite d'un écran E1 affichant l'espace logique entre 0 et 1024, on pourra en déduire que l'écran E2 affiche l'espace logique entre 1024 et 2048.
- β↔δ : β émet les vues sur l'espace logique et les surfaces physiques auxquels elles correspondent, permettant à δ d'effectuer les rendus. Le mécanisme de répartition renseigne β sur les zones occupées de l'espace logique. Par exemple qu'une zone contenant des informations n'est pas affichée.
- δ↔ε : ε émet, pour chaque plate-forme, les informations relatives aux interacteurs. Le mécanisme de répartition les répercute sur les autres plates-formes.

Les dépendances entre les modules fonctionnels n'ont pas toutes la même force. Nous voyons deux blocs se dessiner. D'un coté, nous trouvons le mécanisme de répartition qui est très dépendant des outils graphiques. Ils forment ensemble la couche basse de la distribution. De l'autre, nous trouvons la découverte des entités, le calcul des relations et la mise en correspondance entre les espaces physique et logique. Ensemble, ils forment la partie haute, « intelligente » de la distribution. Il nous paraît indispensable de bien séparer au moins ces deux groupes. En effet, il est souhaitable de pouvoir factoriser les algorithmes de mise en correspondance. Il nous semble illusoire de penser qu'une seule BAOIG sera adaptée pour tout. Soit par les technologies mises en oeuvres, soit par leur logique intrinsèque (basée ou non sur des graphes de scène, ayant des interacteurs ou des primitives graphiques seulement, etc.), des BOAG très différentes verront le jour, ciblant chacun un problème précis.

En ce qui concerne la répartition (δ), en supposant qu'on adopte la structure de graphe de scène pour l'organisation des interacteurs (Demeure, 2004) (Ubit, 2004), nous proposons que le partage se fasse au niveau de sous graphes (figure 2). Pour l'ordinateur 1, les nœuds partagés par l'ordinateur 2 seront répliqués dans le sous graphe ayant pour racine le nœud pivot. L'ordinateur 2 fait de même de telle sorte que les sous graphes partagés par 1 et 2 soient les mêmes. Chacun accède à son sous graphe partagé par un nœud point de vue qui détermine la portion d'espace visualisée.

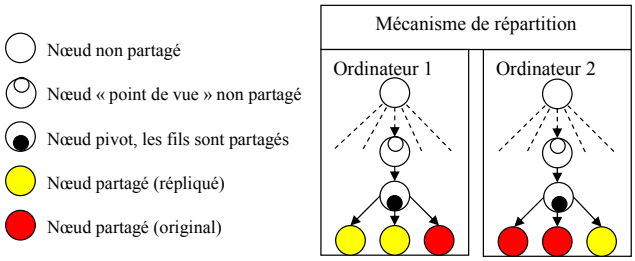


Figure 2: Partage de sous graphes entre ordinateurs.

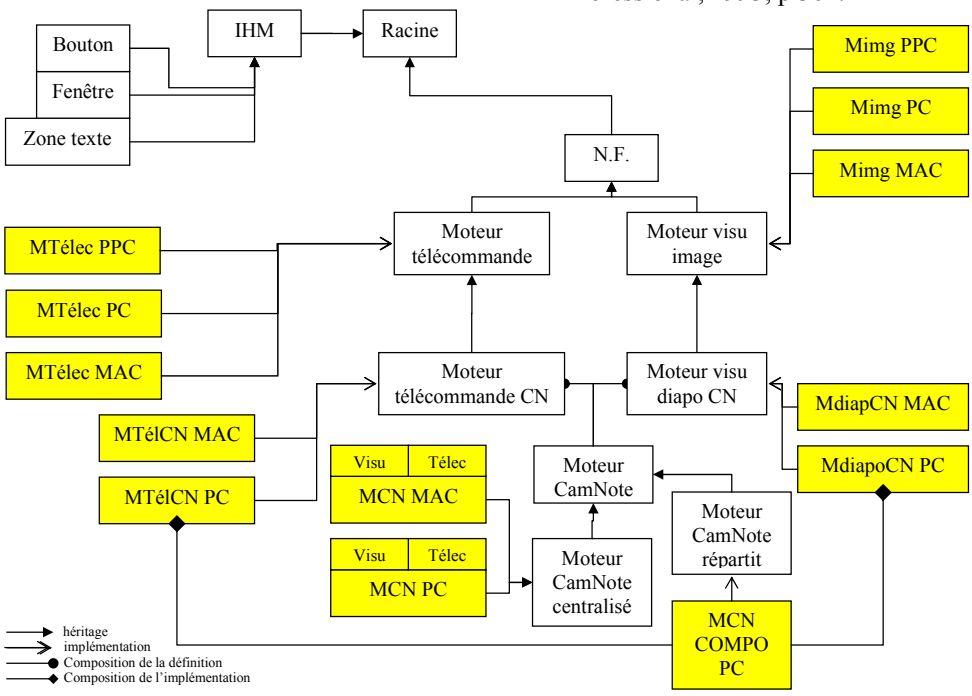
CONCLUSION

La plasticité des interfaces est un sujet qui pose de nombreux problèmes. Parmi ceux-ci je traite les problèmes suivants :

- modélisation de l'évolution de systèmes plastique en explicitant les règles qui déterminent celle-ci.
- capitalisation des systèmes plastique pour facilité la réutilisation en conception et à l'exécution.
- Création d'une boîte à outil supportant la distribution des interfaces au niveau du pixel et exploitant les capacités d'OpenGL.

BIBLIOGRAPHIE

1. Demeure A., Rouillard S. Bérard F., Calvary G., *Requis et pistes pour les futures boîtes à outils d'interaction graphiques.* IHM'04, Namur.
2. Gilmore D.J., Interface Design : Have we got it wrong ?, Human Computer Interaction, *Interact'95*, Nordby, K., Helmersen, P.H., Gilmore, D.J., Arnesen, S.A. (Eds), Chapman and Hall Press, 1995, pp 173-178.
3. Lachenal, C., Coutaz, J. *A Reference Framework for Multi-surface Interaction*. HCII'2003 (Human computer Interaction International) Crete.,
4. McCall J., Factors in Software Quality, General Electric Ed., 1977.
5. Nielsen J., *Usability Engineering*, Academic Press Professional, 1993, p 362.



Un exemple de GDD. On cherche à définir un moteur de présentation à la power point. Les rectangles colorés indiquent que le nœud décrit un exécutable. Chaque nœud contient la description d'un système interactif, ce système peut être définit à un niveau d'abstraction quelconque. Les liens entre les nœuds contraignent les descriptions de ces derniers. Par exemple, en spécifiant qu'un moteur de télécommande permet d'émettre les ordres « suivant » et « précédent », on oblige tous les descendant à faire de même.
Le graphe des descriptions n'est pas autre chose qu'un diagramme de classe maintenu en mémoire. Avec la spécificité que les classes décrites sont des systèmes interactifs écrits dans des langages quelconques.

figure 3 : Un exemple de graphe des déscriptions.