

Le Modèle d'Evolution en Plasticité des Interfaces □ Apport des Graphes Conceptuels

Alexandre Demeure

Gaëlle Calvary

CLIPS-IMAG

BP 53

38041, Grenoble cedex9, France

□Alexandre.Demeure, Gaëlle.Calvary□@imag.fr

RESUME

Cet article traite de la plasticité des Interfaces Homme-Machine, c'est-à-dire de leur capacité à s'adapter à leur contexte d'usage dans le respect de leur utilisabilité. Par contexte d'usage, on entend un triplet <utilisateur, plate-forme, environnement>. L'adaptation est un processus en trois étapes comprenant la reconnaissance de la situation, le choix puis la mise en oeuvre de la réaction. Cet article traite du choix et de la mise en oeuvre de la réaction. Il propose un graphe des descriptions et un modèle d'évolution pour respectivement capitaliser les IHM et spécifier l'adaptation. Ces notions sont basées sur les Graphes Conceptuels.

MOTS CLES : Plasticité des interfaces, modèle d'évolution, graphe des descriptions, graphes conceptuels.

ABSTRACT

This paper deals with Plasticity of User Interfaces (UI). Plasticity refers to the ability of UIs to withstand variations of context of use while preserving usability. By context of use, we mean a triple <user, platform, environment>. Adaptation is a three-steps process involving the recognition of the situation, the computation of a reaction to cope with the situation, and the execution of the reaction. This paper covers both the computation and the execution of the reaction. It proposes a graph of descriptions and an evolution model for respectively capitalizing UIs and specifying the adaptation to perform in case of change of context of use. These notions are based on conceptual graphs.

KEYWORDS : Plasticity of User Interfaces, evolution model, graph of descriptions, conceptual graphs.

INTRODUCTION

Jusqu'ici, les Interfaces Homme-Machine (IHM) étaient confinées à l'écran d'une station de travail et envisagées pour un contexte d'utilisation donné. Il s'agissait, par exemple, de développer un logiciel de présentation pour un écran de résolution 1280x960 installé dans un environnement lumineux et bruyant. Avec les progrès de la technologie sans fil et la miniaturisation des composants

matériels, l'informatique s'infiltré dans le tissu de nos activités quotidiennes [18]. Il en résulte que l'hypothèse implicite de la plate-forme d'exécution unique et du lieu d'interaction fixe ne tient plus en conception d'IHM □ l'adaptation d'une IHM à son contexte d'usage s'impose comme issue à la diversité croissante des plates-formes et des environnements.

L'adaptation des IHM est un processus en trois étapes [2] comprenant la reconnaissance de la situation, le calcul puis la mise en oeuvre de la réaction. Cet article traite du calcul et de la mise en oeuvre de la réaction. Il propose un graphe des descriptions (section 3) et un modèle d'évolution (section 4) pour respectivement capitaliser les IHM et spécifier l'adaptation. Ces notions sont basées sur les Graphes Conceptuels présentés en section 2. La première section situe ce travail dans le cadre général de la Plasticité des Interfaces et fixe le cas d'étude en support à l'illustration. Des perspectives à ce travail sont énoncées en section 5.

CADRE DE TRAVAIL, OBJECTIFS ET CAS D'ETUDE

Cette section localise la contribution de l'article dans le cadre général de la Plasticité des Interfaces. Elle illustre les objectifs sur un cas d'étude.

Plasticité des Interfaces

Le terme *plasticité* fait référence à la propriété physique des matériaux capables de se dilater ou se contracter sous l'effet de contraintes naturelles sans se casser, c'est-à-dire en préservant l'usage commun. Par analogie, une IHM plastique est une IHM capable de s'adapter à son contexte d'usage dans le respect de son utilisabilité. Par contexte d'usage, on entend un triplet <utilisateur, plate-forme, environnement> où [16] □

- L'utilisateur représente l'utilisateur du système interactif.
- La plate-forme recense les requis matériels et logiciels nécessaires à l'interaction.
- L'environnement se réfère aux conditions lumineuses, sonores, etc. pesant sur l'espace physique hébergeant l'interaction.

Calvary [3] perçoit deux leviers d'adaptation

- un remodelage de l'interface. Par exemple, compacter un ensemble de boutons radio en un menu déroulant
- une redistribution de l'interface sur l'ensemble des plates-formes disponibles dans l'environnement. C'est la métaphore du peintre de Rekimoto [14] où la palette et la zone de dessin sont respectivement hébergées par un PDA (Personal Digital Assistant) et un PC. La redistribution peut se manifester par une migration totale ou partielle des IHM.

Les deux leviers, remodelage et redistribution, sont couverts, du point de vue de l'ingénierie, par un même cadre de référence en plasticité [4]. Ce cadre de référence unifie la conception et l'exécution d'IHM plastiques. Il structure (Figure 1)

- le processus de conception autour d'un ensemble de modèles *initiaux* et *transitoires*. Parmi les modèles, initiaux, c'est-à-dire spécifiés par le concepteur, se trouve, outre les modèles du domaine (tâches et concepts) et du contexte d'usage (utilisateur, plate-forme, environnement), un modèle dit d'évolution, dont le rôle est de spécifier la réaction à mettre en oeuvre en cas de changement de contexte d'usage. Cet article en propose une modélisation.

- l'exécution autour d'un processus en trois étapes comprenant la reconnaissance de la situation, le choix puis la mise en oeuvre de la réaction (respectivement R, C, M sur la Figure 1). Ces mécanismes sont embarqués dans les IHM finales (i.e. exécutables ou interprétables) et/ou pris en charge par une infrastructure dédiée à l'exécution. La reconnaissance de la situation ne fait pas l'objet de l'article. Elle s'appuie sur les travaux de Rey [13]. Nous nous focalisons ici sur le choix et la mise en oeuvre de la réaction. Nous enrichissons le cadre de référence par l'introduction d'un graphe des descriptions dont la finalité est de capitaliser les IHM à différents niveaux d'abstraction concepts et tâches, interfaces abstraites (structuration des IHM en espaces de travail), concrètes (réification des interfaces abstraites en objets d'interaction) et finales (code exécutable ou interprétable) (Figure 1).

Nos contributions se situent ainsi, d'une part, en conception par la proposition d'une modélisation de l'évolution, d'autre part, à l'exécution par l'introduction d'un graphe des descriptions et d'un ensemble de mécanismes pour la capitalisation et l'exploitation d'IHM en cas de changement de contexte d'usage. La figure 1 situe ces contributions sur le cadre de référence étendu. La section suivante les illustre sur un cas d'étude.

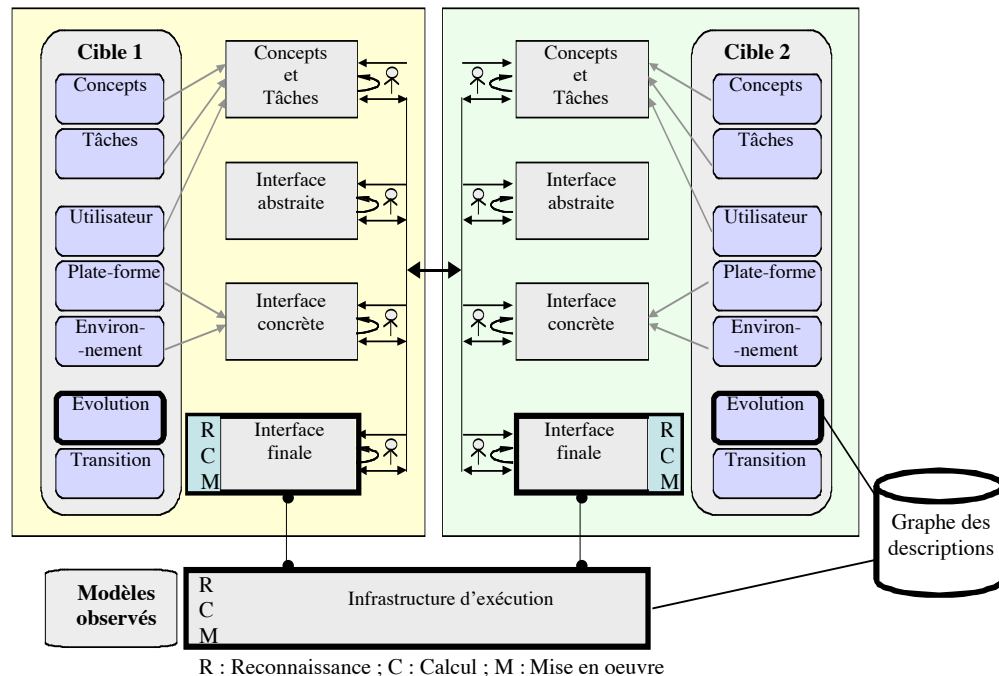


Figure 1 Le cadre de référence en plasticité Adapté de [4]. En gras, la localisation des contributions de l'article.

Cas d'étude ☐ CamNote

CamNote est un logiciel de présentation réalisé dans le cadre du projet européen CAMELEON (d'où son nom CamNote). Il est composé de trois modules capables d'inter opérer à distance ☐

- Un visualisateur de diapositives. Sur PC, ce visualisateur permet l'incrustation de vidéos translucides provenant de caméras. Ces pixels miroirs [11] [17] ne sont pas disponibles sur Pocket PC.
- Un éditeur/visualisateur de notes, des notes pouvant être prises par diapositive ☐
- Une télécommande de navigation permettant de se déplacer dans la présentation.

Supposons que Marc arrive en salle 102 pour dispenser un cours. Il est muni d'un portable contenant sa présentation. Il ouvre le document, le projette à l'écran par le biais d'un vidéo projecteur présent dans la pièce et commence son cours. Il peut à tout moment, à l'insu des étudiants, consulter, sur son portable, les notes associées à la diapositive courante. Ne supportant pas l'immobilité, Marc décide très vite de migrer la télécommande de navigation sur son PDA. La présentation de cette télécommande s'en trouve changée pour s'accommoder de la taille d'écran plus restreinte.

Ce cas d'étude couvre la redistribution et le remodelage d'IHM (migration de la télécommande puis réajustement de sa présentation). Sur cet exemple, le modèle d'évolution spécifie la réaction à mettre en oeuvre lorsque l'utilisateur migre la télécommande du portable au PDA ☐ lancer un composant *télécommande* exécutable sur PDA. Ce composant est stocké dans le graphe des descriptions. Il peut l'être à différents niveaux d'abstraction (concepts et tâches, interfaces abstraites, concrètes et finales).

Ces deux contributions, modèle d'évolution et graphe des descriptions, s'appuient, dans une large mesure, sur les graphes conceptuels [15]. Ces fondements font l'objet de la section suivante.

LES GRAPHES CONCEPTUELS

Cette section présente les graphes conceptuels et en montre l'apport en plasticité des interfaces.

Présentation

Les graphes conceptuels sont un type de réseau sémantique [15]. Un réseau sémantique est un système de représentation graphique des connaissances basé sur des nœuds interconnectés par des arcs. Il prend donc la forme d'un graphe.

Un graphe conceptuel est un graphe biparti étiqueté par des items lexicaux. Une des classes de sommets corres-

pond à des concepts, l'autre à des relations entre concepts.

On peut associer à un graphe conceptuel une formule de la logique du premier ordre. Il existe une sémantique logique consistante et complète, c'est à dire équivalente, pour le modèle des graphes conceptuels comme l'ont montré Sowa [15] et Chein et Mugnier [5].

Le terme de graphe conceptuel proposé par Sowa définissait à la fois un modèle de base assez précis (appelé graphe conceptuel simple) et des extensions (ou idées d'extensions) de manière plus ou moins formelles. Des chercheurs ont depuis développé ce modèle de base, comme Chein et Mugnier [6] qui ont notamment proposé la notion de graphes emboîtés. Un graphe emboîté est tel que les nœuds concepts peuvent inclure d'autres graphes. Nous nous basons sur ces travaux.

Comme le précisent Chein et Mugnier [6], les graphes conceptuels peuvent être considérés comme ☐

- un modèle déclaratif de représentation des connaissances.
- un moyen de résoudre les calculs d'inférence par des algorithmes de graphe.

L'avantage des graphes conceptuels sur la logique du premier ordre ne se situe pas au niveau de leur puissance d'expression brute mais principalement au niveau de leur lisibilité. Leur représentation graphique est plus expressive qu'une formule logique (Figure 2).

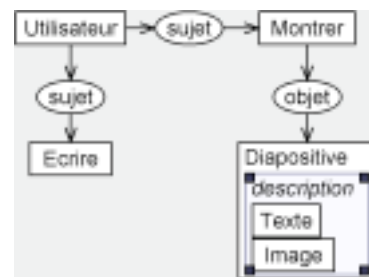


Figure 2 : Un exemple de graphe conceptuel. Ce graphe exprime qu'un utilisateur écrit et montre une diapositive contenant du texte et au moins une image.

Les connaissances exprimées dans un graphe conceptuel n'ont de sens que par rapport à un support donné. Un support définit le vocabulaire de base. Il comprend un ensemble ordonné (treillis) de concepts et de relations (un treillis pour les concepts et un treillis par arité de relations), chaque relation étant munie d'une signature. Les ordres sur ces ensembles sont des liens de spécialisation. La figure 3 en donne un exemple pour CamNote.

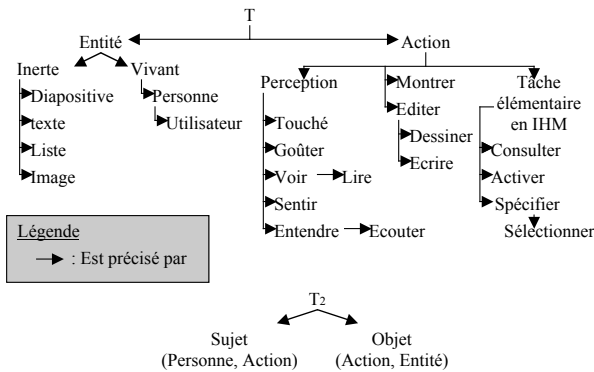


Figure 3 : Une partie du support utilisé dans CamNote. T est le concept universel \square T2 la relation binaire universelle.

En tant que base à l'expression des connaissances, la qualité d'un support est déterminante, un support mal formé pour un problème donné compromettra son exploitation.

Utilité

Grâce à l'opération de projection offerte par les graphes conceptuels, on peut savoir si l'information présente dans un graphe est déductible d'un autre. La projection d'un graphe G dans un graphe H revient à chercher si l'information portée par G peut être déduite de celle portée par H (Figure 4).

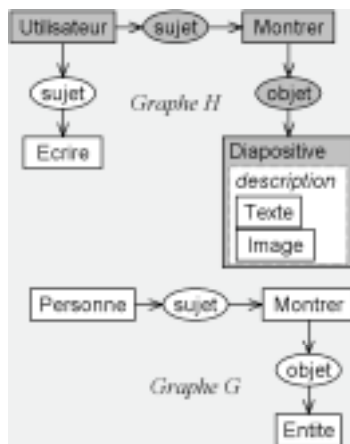


Figure 4 : La projection de G dans H apparaît en grisé.

L'étiquette de l'image par projection d'un sommet n de G dans H est une spécialisation de n. Ainsi sur l'exemple présenté en Figure 4, l'image de Personne est Utilisateur (qui est une spécialisation de Personne d'après le support) et celle d'Entité est Diapositive (qui est une spécialisation d'Entité d'après le support).

En recherche d'information, la projection est utilisée pour comparer une requête aux descriptions de documents. Dans l'exemple, si la requête avait été G cela aurait signifié qu'on cherchait tous les documents dans lesquels «une personne montre quelque chose».

Nous utilisons ici les graphes conceptuels à la fois \square

- dans une optique de recherche d'information. C'est ce qui motive principalement leur utilisation dans le graphe des descriptions. Il s'agit de capitaliser les IHM et de les organiser au sein d'un graphe pour pouvoir, par exemple, retrouver un composant télécommande fonctionnant sur PDA.
- pour leur lien avec la logique. C'est ce qui motive principalement leur utilisation dans le modèle d'évolution pour la détection d'éventuelles contradictions dans les réactions préconisées.

Dans les deux cas, la lisibilité des graphes conceptuels permet d'envisager leur manipulation par les utilisateurs finaux. Les utilisateurs pourront ainsi spécialiser (adaptabilité) les réactions par défaut embarquées dans les IHM. Les graphes sont aujourd'hui mis en oeuvre par la bibliothèque Cogitant [8] développée par l'équipe [CORALI](#) du LIRMM.

LE GRAPHE DES DESCRIPTIONS

Nous justifions dans un premier temps l'existence de cette structure puis en proposons une modélisation et décrivons son exploitation en conception et exécution d'IHM plastiques.

Justification

Nous décrivons les composants logiciels à l'aide de graphes conceptuels. Par exemple, une diapositive peut être décrite par un graphe conceptuel (Figure 4). Les projections étant des opérations coûteuses, l'idée est d'organiser les composants logiciels ainsi décrits au sein d'un graphe des descriptions. Ce graphe assure, par construction, une présélection des candidats. Par exemple, toutes les télécommandes PC, PDA, etc. seront connexes à un nœud *Télécommande*. Notons toutefois que d'autres travaux existent pour accélérer le calcul des projections par pré traitement [12].

Modélisation

Le graphe des descriptions est composé d'un seul type de nœuds, chacun donnant la description de ce qu'il contient (d'où son nom). Cette description est exprimée sous la forme d'un graphe conceptuel. Les relations entre deux nœuds A et B peuvent être de trois types \square

- A hérite de B.
- A est composé de B.
- A implante B.

Ces relations sont contraintes par les graphes et leurs supports, et réciproquement \square

- A hérite de B \square Le graphe de B se projette dans A. Le support de A étant celui de B.
- A est composé de B \square Le graphe de B se projette dans A et son image est B. Les concepts de B sont

ajoutés à ceux de A et éventuellement renommés s'il y a synonymie avec des concepts originaux de A.

- A implante B. Le graphe de B se projette dans A. Le support de A étend celui de B.

Outre la description de ce qu'il contient, chaque nœud peut comporter d'autres informations comme un lien vers un exécutable ou un fichier source par exemple. Ce lien dépendra typiquement du niveau d'abstraction des informations décrites dans le nœud. Remarquons tout de suite que rien ne permet de prouver automatiquement que le contenu proprement dit d'un nœud correspond bien à la description qui en est faite. C'est un problème qui se pose de façon générale lors de la description de documents par des graphes conceptuels. Il n'existe en effet aucun moyen de générer une telle description de façon automatique [7]. La description reste à la charge, et sous la responsabilité, du concepteur.

La figure 5 illustre le graphe des descriptions dans le cadre de CamNote. La section suivante en montre l'utilité en conception.

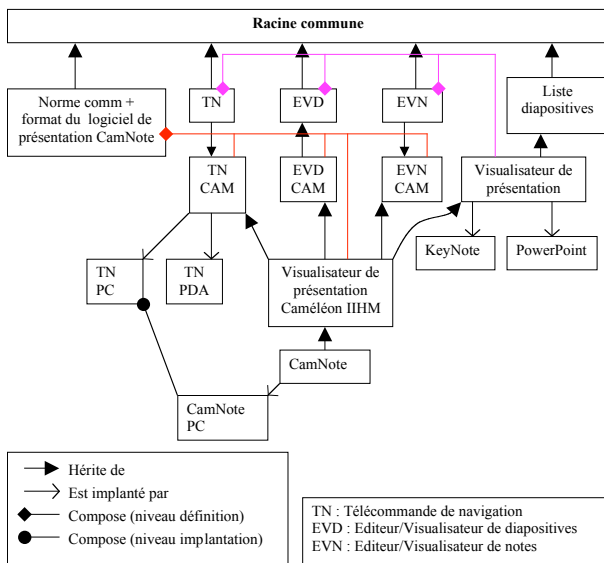


Figure 5 : Un sous-ensemble du graphe des descriptions de CamNote. Chaque nœud comporte un graphe conceptuel qui le décrit. Le graphe conceptuel et le support d'un nœud sont contraints par les relations qu'entretient ce nœud avec le reste du graphe.

Utilisation en conception

Le graphe des descriptions est utile pour inciter 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. Ce graphe soulève typiquement à la conception les questions suivantes

- 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 permettre à d'autres d'y accéder?
- enfin, quelles parties des autres applications pourraient être réutilisées dans ce logiciel

Cette approche favorise modularité et réutilisabilité. Par exemple, le concepteur pourra réutiliser une télécommande fonctionnant sur PDA. La section suivante traite de l'exécution.

Utilisation à l'exécution

À l'exécution, le graphe des descriptions couvre migration et remodelage. Par exemple, lorsque Marc migre la télécommande sur PDA, il s'agit, dans ce graphe, de localiser un composant télécommande fonctionnant sur PDA. Pour cela, une requête est formulée. Elle décrit le composant recherché (Figure 6).

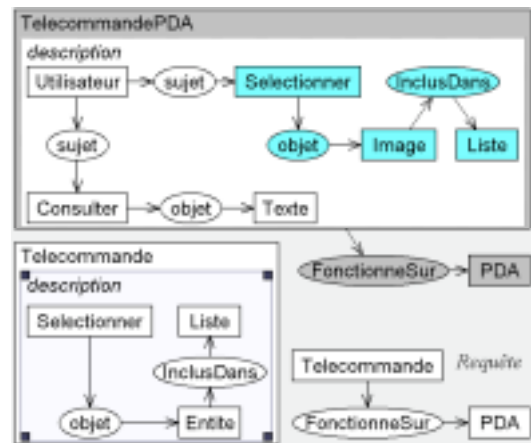


Figure 6 : Formulation d'une requête pour la recherche d'une télécommande fonctionnant sur PDA. Le composant TélécommandePDA répond à la description.

Le parcours du graphe des descriptions se fait récursivement à partir du nœud décrivant le logiciel de présentation en cours d'utilisation. Une fonction de test est utilisée pour indiquer, à chaque étape, si le parcours doit se poursuivre. La sélection d'un candidat s'opère si le graphe requête se projette dans le graphe conceptuel du nœud. Sur l'exemple de la Figure 7, en posant comme fonction de test « Télécommande se projette dans le graphe conceptuel du nœud parcouru », on obtient le marquage suivant (Figure 7).

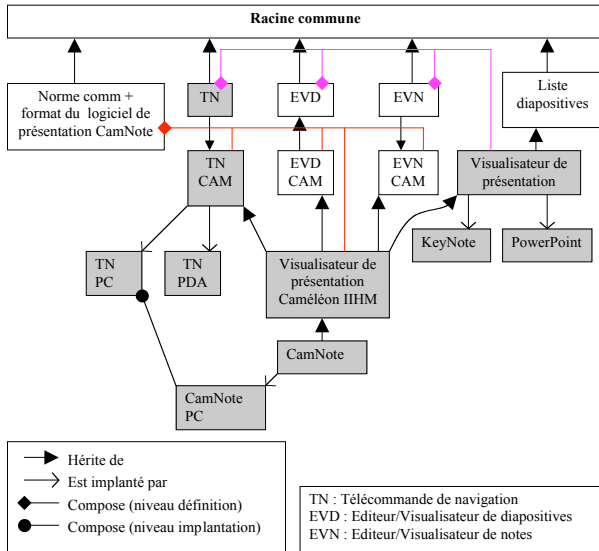


Figure 7 : En grisé, l'ensemble des nœuds parcourus lors de la recherche d'une télécommande fonctionnant sur PDA. Seul "TN PDA" sera retenu.

Remarquons que la requête pourrait être optimisée facilement en spécifiant qu'on s'intéresse seulement aux télécommandes en tant que telles et pas aux programmes qui contiennent des télécommandes. Cela s'obtiendrait en spécifiant que l'image de la projection de la requête devrait avoir comme contexte le contexte global du graphe conceptuel [6] et non un sous-contexte particulier. Un exemple de contexte non global au graphe est donné en Figure 8 où le *Logiciel* contient un composant *Télécommande* mais n'en est pas un. C'est à la fonction de test de spécifier cette contrainte.

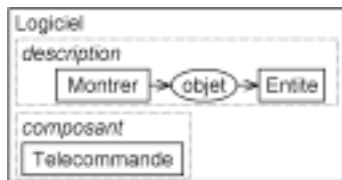


Figure 8 : Dans ce graphe, *Télécommande* a bien une image par projection mais elle n'est valide que dans le sous-contexte *composant de Logiciel*.

Ce principe de parcours illustré sur le cas de la migration couvre aussi le remodelage (recherche d'une description d'interface). La réaction est spécifiée dans le modèle d'évolution. L'accès au graphe est déclenché par ce modèle. La section suivante y est consacrée.

LE MODELE D'EVOLUTION

Cette section couvre la modélisation et la mise en oeuvre du modèle d'évolution.

Modélisation

La modélisation s'appuie sur deux composants : le Proposant et l'Analyseur (Figure 9). Ils composent le modèle d'évolution.

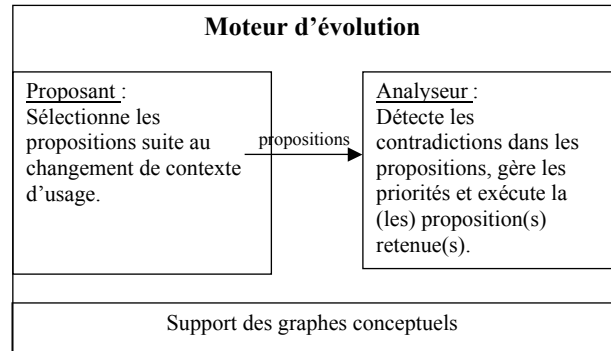


Figure 9 : Synoptique du modèle d'évolution.

Le Proposant est un module chargé de proposer à l'analyseur des réactions suite au changement de contexte d'usage. Notons que l'utilisateur peut lui-même suggérer des réactions. Une IHM est alors nécessaire pour le modèle d'évolution. Les propositions sont des quadruplets comprenant :

- Une fonction à appeler pour mettre en œuvre la proposition (éventuellement paramétrée).
- Un niveau de conseil prenant corps dans un espace à trois dimensions (a) un poids compris entre -1 (déconseillé) et 1 (conseillé) indiquant le niveau de recommandation de la proposition, (b) un qualificatif pouvant prendre pour valeur «De convenance» ou «De survie» selon que la proposition considérée vise à faciliter la vie de l'utilisateur ou à assurer la survie même du logiciel (cas d'un logiciel sur un PDA dont la batterie est faible et qui demande à migrer), (c) une origine indiquant l'auteur de la proposition (concepteur ou utilisateur).
- Une description de l'effet de la proposition si elle est appliquée. Cette description est-elle même exprimée sous la forme d'un graphe conceptuel dont la forme dépendra éventuellement des paramètres donnés à la fonction. La présence d'un graphe conceptuel justifie l'existence d'un support commun dans notre modèle (Figure 9).
- Eventuellement une description en langue naturelle pour les cas complexes.

Un exemple de proposition pourrait être *Proposer(migration, <Concepteur, De Convenance, -1>)* où «*migration*» contiendrait la fonction à appeler ainsi qu'une description sous forme d'un graphe conceptuel. Ces propositions sont envoyées à l'analyseur.

L'analyseur est chargé de sélectionner des propositions parmi celles issues du Proposant. Cette étape couvre la détection d'éventuelles contradictions entre propositions puis la sélection. La sélection s'appuie sur les attributs de la proposition (poids, qualificatif, auteur, etc.). Ces étapes peuvent être manuelles, semi-automatiques ou auto-

matiques. La section suivante en décrit la mise en oeuvre.

Mise en oeuvre

L'implantation du modèle d'évolution est faite en TCL et C++. Le choix de TCL est motivé, en premier lieu, par son caractère interprété (il permettra à l'utilisateur final de surcharger le modèle d'évolution) et en second lieu par sa portabilité et sa bibliothèque graphique TK. Le choix de C++ provient de Cogitant pour la mise en oeuvre des graphes conceptuels.

Le moteur d'évolution est implanté par un ensemble d'automates dont chaque état est une fonction dont l'exécution peut provoquer l'émission d'une ou plusieurs propositions ainsi que la transition vers un autre état (Figure 10).

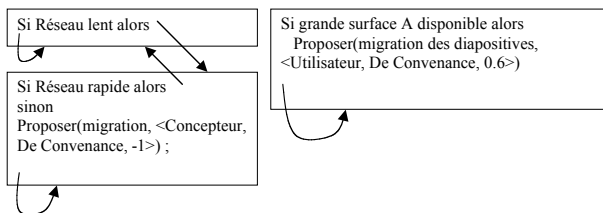


Figure 10 : Un exemple à 2 automates dans le cas de Cam-Note. Celui de gauche déconseille toute migration si le réseau est lent, celui de droite conseille de migrer les diapositives sur la plus grande surface d'affichage.

Les automates ne sont pas la seule implantation possible (les statecharts seraient d'autres candidats). Leur intérêt est de pouvoir structurer les propositions à faire.. Notons qu'en particulier on peut superposer les automates aux arbres des tâches pour émettre des propositions adaptées à la tâche courante. Par exemple, ne proposer la migration des diapositives du portable au PC qu'au moment où l'utilisateur décide de les projeter sur le mur (Figure 11).

En pratique, les fonctions invoquées par le moteur d'évolution sont celles mises à disposition dans l'application. Ainsi le concepteur pourra-t-il restreindre ces fonctions à la stricte API qu'il souhaite. Ce filtrage protégera l'utilisateur contre des invocations dangereuses.

L'Analyseur est implanté selon l'algorithme suivant :

- 1- Séparer les propositions selon leur caractère conseillé ou déconseillé.
- 2- Eliminer tout conseil qui contredit une mise en garde selon les règles suivantes☐
 - a- Toute mise en garde du concepteur est prioritaire sur un conseil de l'utilisateur.
 - b- Si le conseil est de convenance et la mise en garde de survie alors ne pas retenir le conseil.
 - c- S'il sont de même qualificatif, retenir la proposition de poids le plus fort (en valeur absolue).

- d- En cas d'égalité, demander l'arbitrage de l'utilisateur.
- e- Parmi les conseils retenus, vérifier qu'il n'y a pas de contradiction implicite en appliquant des fonctions idoines.

La détection des contradictions explicites entre conseils et mises en garde se fait à l'aide des graphes conceptuels. Si le graphe conceptuel décrivant une mise en garde se projette dans celui émanant d'un conseil ou inversement, il y a contradiction explicite entre les deux. En effet, dans ce cas, on conseille et déconseille à la fois une réaction (Figure 11).



Figure 11 : Exemple de contradiction explicite.

La figure 12 illustre un cas de contradiction implicite.



Figure 12 : Exemple de contradiction implicite. A veut passer à l'image suivante, B à la précédente.

Pour résoudre ce type de contradiction, il faut spécifier, pour chaque concept ou relation, la liste des concepts ou relations qui lui sont contradictoires. Ainsi pour un graphe donné on va☐

- Construire un sous-ensemble des graphes contradictoires à partir des listes de concepts ou relations contradictoires de chaque nœud (l'ensemble des combinaisons possibles en remplaçant les nœuds par un de leur opposé dans leur liste contradictoire).
- Pour chaque couple formé d'un des graphes contradictoires G et d'un graphe issu d'une autre proposition H, tester si G se projette dans H ou inversement. Ce cas correspond à une contradiction implicite. L'arbitrage se fait comme pour les contradictions explicites.

Cette technique requiert un grand nombre de projections. Il convient donc soit de limiter les liens de contradiction, soit de trouver un algorithme plus efficace. D'autres contraintes implicites peuvent exister et être résolues par d'autres algorithmes. Par exemple, le fait qu'un composant ne puisse migrer qu'à un endroit à la fois. Cette exigence peut être détectée en vérifiant que le sous graphe formé par le nœud «migrer☐ et par le sous graphe objet

de la migration, ne se retrouve pas dans d'autres propositions et, si c'est le cas, que la destination est la même. Sinon il y a contradiction.

CONCLUSION ET PERSPECTIVES

Dans cet article, nous avons présenté un graphe des descriptions et un modèle d'évolution pour régir l'adaptation des systèmes interactifs à leur contexte d'usage. Leur modélisation s'appuie sur les graphes conceptuels. Leur implantation fournit des outils au concepteur, et dans une moindre mesure à l'utilisateur final, pour la spécification de l'adaptation. Ces outils couvrent la redistribution et le remodelage, les deux leviers de la plasticité.

A court terme, il reste à finaliser la mise en oeuvre de ces outils, notamment par l'ajout de services tels que la navigation au sein du graphe des descriptions et le maintien de la cohérence. A plus long terme, le graphe des descriptions pourrait aussi servir à identifier la transition à mettre en oeuvre pour accompagner le changement de l'IHM lors de l'adaptation [1]. Par exemple, par l'identification des transformations nécessaires pour passer d'une description à une autre.

REMERCIEMENTS

Ce travail a bénéficié du soutien du projet européen IST CAMELEON (IST-2000-30104). Nous remercions particulièrement Lionel Balme pour sa contribution dans la mise en oeuvre.

BIBLIOGRAPHIE

1. Barralon N., *Interfaces Homme-Machine de Transition*, rapport de DEA d'Informatique Systèmes et Communications, Université Joseph Fourier, Grenoble I, 2002.
2. Calvary, G., Coutaz, J., Thevenin, D. Supporting Context Changes for Plastic User Interfaces: A Process and a Mechanism, in "People and Computers XV – Interaction without Frontiers", Joint Proceedings of AFIHM-BCS Conference on Human-Computer Interaction IHM-HCI'2001 (Lille, 10-14 September 2001), A. Blandford, J. Vanderdonckt, and Ph. Gray (eds.), Vol. I, Springer-Verlag, London, 2001, pp. 349-363.
3. Calvary, G., Coutaz, J. *Plasticité des Interfaces: une nécessité*. Actes des deuxièmes Assises nationales du GDR I3, Cépaduès Editions, J. Le Maître (Ed), Nancy, Décembre 2002, pp 247-26.
4. Calvary, G., Coutaz J., Thevenin, D., Limbourg, Q., Bouillon, L., Vanderdonckt, J. *A unifying reference framework for multi-target user interfaces*, Interacting With Computers, 2003, A paraître.
5. Chein M et Mugnier M.L., *Conceptual Graphs: Fundamental notions*, Revue d'intelligence artificielle, 6, 4, 1992, 365-406.
6. Chein M et Mugnier M.L., *Représenter des connaissances et raisonner avec des graphes*, R.I.A, vol.10, n°1, 1996, pp 7-56
7. Chevallet J.P., *Un modèle logique de Recherche d'Informations appliqué au formalisme des graphes conceptuels. Le prototype ELEN et son expérimentation sur un corpus de composants logiciels.*, Thèse pour l'obtention du titre de docteur en informatique, Université Joseph Fourier 1988.
8. Cogitant, cogitant.sourceforge.net/
9. Genest D., *Recherche d'information par transformation de graphes dans le modèle des graphes conceptuels*, RSTI série IS-NIS, volume 7, n°1-1/2002, pp207-236.
10. Harel, D. Stacharts: A Visual Approach to Complex Systems XX à compléter
11. Morikawa O., Maesako T., *HyperMirror : Toward Pleasant-to-use Video Mediated Communication System*. Actes de la conférence CSCW'98, Seattle, Washington USA. pp.149-158.
12. Ounis I., *RELIEF: Combining expressiveness and rapidity into a single system*, [SIGIR 1998](http://sigir1998.org/), pp 266-274.
13. Rey G., *Systèmes interactifs sensibles au contexte*, rapport de DEA d'Informatique Systèmes et Communications, Université Joseph Fourier, Grenoble I, 2001.
14. Rekimoto, J. Pick and Drop: A Direct Manipulation Technique for Multiple Computer Environments. In Proc of UIST97, ACM Press, 1997, pp. 31-39.
15. Sowa J.F., *Conceptual Structures – Information Processing in Mind and Machine*, Addison Wesley, 1984
16. Thevenin D., *Adaptation en Interaction Homme-Machine : le cas de la Plasticité*. Thèse pour l'obtention du titre de docteur en informatique, Université Joseph Fourier, Grenoble I, 2001.
17. Vernier F., Lachenal C., Nigay L. et Coutaz J. *Interface Augmentée Par Effet Miroir* article de recherche long, IHM'99. (Conférence AFIHM sur l'Interaction Homme-Machine, 22-26 Novembre 1999 Montpellier, France) . pp. 158-165
18. Weiser M., *The computer for the 21st century*, Scientific American, 265(3), 1991, pp. 94-104.