Adaptation des Interfaces Homme-Machine à leur contexte d'usage

Plasticité des IHM

Gaëlle Calvary — Alexandre Demeure — Joëlle Coutaz — Olfa Dâassi

CLIPS-IMAG
BP 53
38041 Grenoble Cedex 9
Gaelle.Calvary@imag.fr

RÉSUMÉ. *Cet article traite de la Plasticité des Interfaces Homme-Machine (IHM), c'est-à-dire de la capacité des IHM 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>. Après une analyse de l'espace problème de l'adaptation, cet article propose un cadre de référence en plasticité pour la conception et l'exécution d'IHM plastiques. Ce cadre de référence sert de support à une revue critique de l'état de l'art. Cette revue met en évidence deux manques que nous proposons de combler par une approche à agents et graphes conceptuels. L'article est illustré sur un cas d'étude CamNote consistant en un logiciel de présentation plastique.*

ABSTRACT. *Mobility coupled with the development of a wide variety of access devices has engendered new requirements for HCI such as the ability of user interfaces (UI) to adapt to different contexts of use. By context of use, we mean the triple <user, platform, environment>. This paper first analyses the problem space of adaptation then proposes a reference framework covering both the development and execution of plastic UI. This framework helps in structuring the state of the art. It emphasizes two open issues that we propose to address by an agent and conceptual graphs-based approach. The paper is illustrated on a running example CamNote that supports the presentation of slides in different contexts of use.*

MOTS-CLÉS : *Adaptation, contexte d'usage, plasticité, multiciblage, cadre de référence, évolution, interacteur, approche à agents, graphe conceptuel.*

KEYWORDS: *Adaptation, context of use, plasticity, multi-targeting, reference framework, evolution, interactor, agent-based approach, conceptual graph.*

1. Introduction

Avec les avancées des réseaux sans fil et les progrès en miniaturisation, l'informatique s'infiltre dans le tissu de nos activités quotidiennes (Weiser, 1991) : elle s'intègre à nos objets usuels pour mieux nous assister dans nos tâches, nos mouvements et interactions avec l'environnement. Pour exemples, des prototypes de recherche tels que le musée augmenté (Rekimoto *et al.*, 1995) (Figure 1a), le tableau magique (Bérard *et al.*, 2000) (Figure 1b) ou le mètre augmenté (Lee *et al.*, 2000) mais aussi des produits commercialisés tels que le réfrigérateur qui commande les produits manquants, la montre-caméra-appareil photo ou le « robotichien » qui répond à la voix de son maître. En même temps, nos ordinateurs de poche s'adaptent au contexte d'usage et nous offrent une information située. Pour exemple, le Cyberguide (Abowd *et al.*, 1996), l'assistant de bureau (Yan *et al.*, 2000) ou encore l'assistant conversationnel Welbo (Anabuki *et al.*, 2000). L'assistant personnel devient télécommande universelle (Schilit *et al.*, 1994), permettant à l'utilisateur de piloter n'importe quel dispositif ambiant : cafetière, télévision, éclairage, etc. Rekimoto prolonge la réflexion en implémentant la métaphore du peintre (Rekimoto, 1997) : désormais, l'assistant personnel devient une palette d'outils s'appliquant à des objets gérés par une autre plate-forme. L'interaction multisurface est née : l'utilisateur peut sélectionner la couleur rouge sur son PDA (Personal Digital Assistant) et l'appliquer à un texte géré par un PC et projeté sur le mur (Myers *et al.*, 1998).

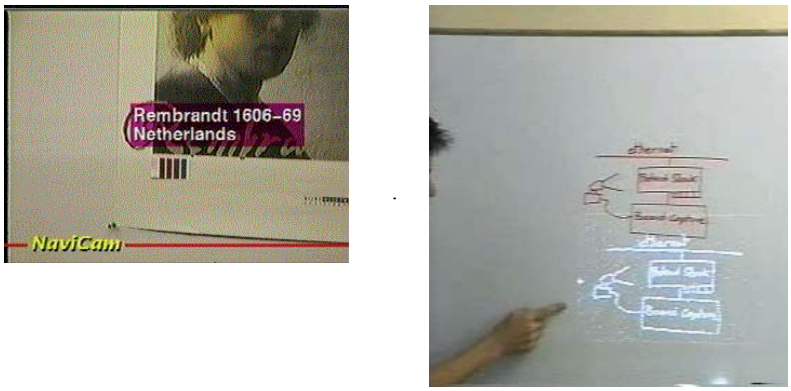


Figure 1. *Exemples d'objets quotidiens augmentés.*

Si d'un point de vue technique, cette informatique de demain, diffuse et mobile (dite ubiquiste) (Lyytinen *et al.*, 2002) est accessible, (Islam *et al.*, 2003) soulèvent

quatre verrous quant à son acceptation « ubiquiste » : en particulier, la multiplicité des dispositifs. Il s'agit de proposer des tâches appropriées sur des dispositifs adaptés, tout en maîtrisant les complexités de développement. Si les approches au cas par cas permettent aujourd'hui d'ajuster l'Interface Homme-Machine (IHM) aux besoins et capacités des utilisateurs pour une plate-forme et un environnement donnés, demain la multiplicité des contextes d'usage compromettra la pratique. Les coûts de développement et de maintenance des différentes versions, accrus par une difficulté de maintien de la cohérence ergonomique entre versions, se heurtent déjà aux exigences économiques. C'est à ce verrou, à la croisée du génie logiciel et de l'interaction homme-machine, que se situe cet article. La voie ici examinée est celle de l'adaptation. L'article traite de la *Plasticité des IHM*, une propriété récemment introduite pour dénoter la capacité d'une IHM à s'adapter à son contexte d'usage dans le respect de son utilisabilité (Thevenin *et al.*, 1999). Le slogan à la base de cette approche est de « spécifier une fois, générer N fois utilisable ». La section 3 définit cette propriété. Elle en cerne l'espace problème sur un cas d'étude *CamNote* présenté en section 2. La section 4 est consacrée à un état de l'art dans le domaine. Il permet d'identifier des manques auxquels nous contribuons en section 5. La section 6 énonce des conclusions et perspectives sur le sujet.

2. Cas d'étude

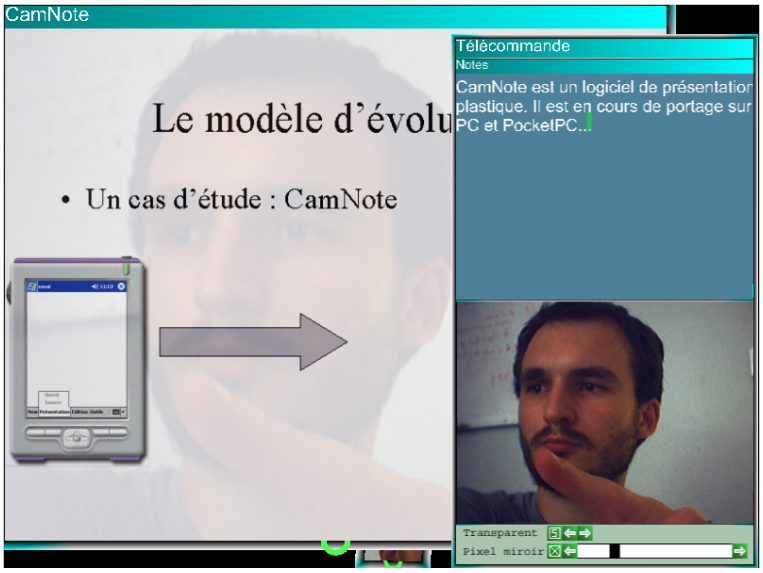
CamNote est un logiciel de présentation réalisé dans le cadre du projet européen CAMELEON (d'où son nom CamNote). Il permet à un enseignant de projeter des diapositives, de naviguer de l'une à l'autre, d'en sélectionner une et de saisir des notes à ce sujet. Il est composé de trois modules disposant chacun d'une version PC et PDA (Figure 2) :

– un visualisateur de diapositives permet de projeter un jeu de transparents. La version PC de ce composant permet l'incrustation de vidéos translucides provenant de caméras. Cette incrustation, dite *pixels miroir* (Morikawa *et al.*, 1998) (Vernier *et al.*, 1999), permet typiquement d'intégrer en dynamique, au sein des diapositives, les gestes ou le visage de l'enseignant (Figure 2a). Les ressources d'un PDA étant limitées, cette possibilité n'existe pas sur PDA. Le PDA peut, par contre, servir à piloter les pixels miroir d'une version PC pour, par exemple, en régler à distance l'intensité lumineuse (Figure 2b) ;

– un éditeur/visualisateur de notes permet à l'enseignant, en mode privé, sur une diapositive donnée, de consulter ses notes ou d'en éditer (Figure 2) ;

– une télécommande de navigation permet à l'enseignant de se déplacer au sein de la présentation (se positionner sur un transparent, passer au suivant ou revenir au précédent) ainsi que de régler l'intensité lumineuse des pixels miroir (Figure 2).

(a) Version PC



(b) Version PDA de la télécommande

Figure 2. *CamNote, versions PC et PDA : (a) version PC composée, d'une part, d'une télécommande (à droite) permettant de piloter les diapositives et pixels miroir, gérer les notes et contrôler la source vidéo, d'autre part, d'un visualisateur de diapositives (à gauche). La présentation porte sur le modèle d'évolution en plasticité. Les pixels miroir sont actifs : le flux vidéo est intégré aux diapositives ; (b) version PDA de la télécommande permettant de piloter la présentation (diapositives et pixels miroir) et consulter les notes. Le flux vidéo n'y est pas observable.*

Les composants de CamNote (visualisateur de diapositives, éditeur/visualisateur de notes et télécommande de navigation) sont conçus pour pouvoir interopérer à distance. Ainsi, la télécommande peut-elle, à tout moment, migrer sur le PDA ; le formateur gagne en mobilité. Le scénario suivant couvre cet exemple de migration.

Marc, enseignant dans le supérieur, 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. Ne supportant pas l'immobilité, Marc décide très vite de migrer la télécommande de navigation sur son PDA. Il peut y piloter le diaporama ainsi que consulter ses notes relatives à la diapositive courante.

La migration de la télécommande suppose une adaptation de ce composant pour s'alléger du contrôle du flux vidéo et s'accommoder d'une taille d'écran restreinte (Figures 2a et 2b). Le scénario de Marc sert de support à l'illustration de l'article. La section suivante définit le vocabulaire à la base de nos travaux ainsi que l'espace problème de l'adaptation.

3. Terminologie et espace problème

Cette section introduit les propriétés de plasticité et de multiciblage puis en cerne l'espace problème selon une perspective utilisateur puis système.

3.1. *Terminologie*

Par analogie avec la plasticité des matériaux qui, sans rompre, s'étirent et se contractent au gré de la chaleur, la *plasticité* d'une interface dénote sa capacité à s'adapter au contexte d'usage dans le respect de son utilisabilité (*Thevenin et al.*, 1999). Par contexte d'usage, on entend un triplet « utilisateur, plate-forme, environnement » où :

– l'*utilisateur* est un utilisateur représentatif du public ciblé. Il est typiquement décrit par ses capacités perceptuelles, motrices et cognitives (Card *et al.*, 1983), notamment ses compétences métier et informatiques, mais aussi par ses caractéristiques culturelles telles que sa langue naturelle. Dans l'exemple de CamNote, l'utilisateur est un enseignant francophone, familier des logiciels de présentation tels que PowerPoint, dont les compétences en informatique se limitent à un niveau utilisateur. Il sait démarrer un ordinateur et manipuler clavier et souris ;

– la *plate-forme* est la structure matérielle et logicielle sous-tendant l'interaction. Par exemple, pour CamNote, un PC et, si possible, un assistant personnel. La taille de l'écran, les dispositifs d'interaction, les capacités de calcul et de communication y sont des informations déterminantes, puisque susceptibles d'influencer l'interaction. Typiquement, sur téléphones portables, proposer des tâches telles que "Saisir notes" ou "Visionner un diaporama" n'aurait pas de sens. La surface d'affichage serait bien trop limitée pour en permettre l'affichage et la manipulation ;

– l'*environnement* se réfère à l'environnement physique accueillant l'interaction. Il est décrit par un ensemble d'informations, périphériques à la tâche en cours mais susceptibles de l'influencer. Par exemple, la luminosité, le bruit, la localisation géographique, la colocalisation sociale, etc. Typiquement, en cours, il est maladroit de conserver le mode sonnerie de son téléphone. Mieux vaut opter pour le mode vibreur plus respectueux de l'environnement social.

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 (Mc Call, 1977). Il relève de l'acceptation d'un système (Nielsen, 1993). 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).

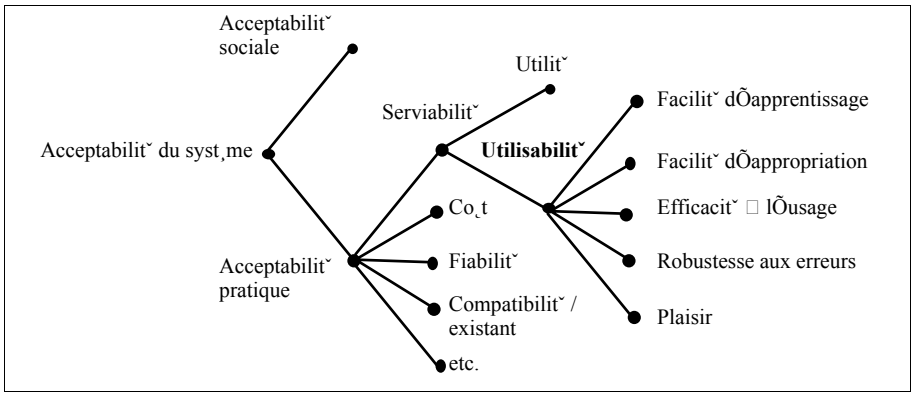


Figure 3. *Un modèle des attributs de l'acceptabilité d'un système. Traduit de (Nielsen, 1993) p 25.*

[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.

Sans métrique, les critères de Nielsen sont difficiles à apprécier. Aussi, l'utilisabilité est-elle, en pratique, évaluée sur les propriétés véhiculées par le système interactif, telles que l'observabilité des concepts manipulés dans les tâches par exemple. Les propriétés définies par l'IFIP (IFIP, 1996) constituent ici une bonne base de raisonnement. Il s'agira de vérifier que, sur l'ensemble des contextes d'usage couverts par le système interactif, l'utilisabilité reste satisfaite. Seules les propriétés mentionnées dans le cahier des charges seront évaluées. On parlera alors d'IHM plastique. Si cette utilisabilité n'était pas préservée, on parlerait d'*IHM multicible*, le terme de *cible* étant ici synonyme de contexte d'usage.

L'adaptation est un processus de type « *Action-Réaction* » (Calvary *et al.*, 2001) dans lequel le maintien de l'utilisabilité relève de la Qualité de Service (QoS). La section suivante examine les parties *Action-Réaction* de ce processus.

3.2. *Espace problème*

Multiciblage et plasticité apparaissent comme des formes d'adaptation. Elles mobilisent un processus de type " *Action-Réaction* " où :

– l'*action* se réfère au changement survenu dans le contexte d'usage. Il porte sur l'utilisateur et/ou la plate-forme et/ou l'environnement ;

– la *réaction* dénote les mesures mises en œuvre par le système et/ou l'utilisateur pour préserver l'utilisabilité, c'est-à-dire l'ensemble des propriétés relatives à l'usage, consignées dans le cahier des charges.

Les sections suivantes développent ces notions d'action et de réaction selon une perspective utilisateur puis système. La perspective utilisateur examine le processus d'"*Action-Réaction*" d'un point de vue de l'usage, c'est-à-dire dans les termes d'un utilisateur non forcément concepteur de systèmes interactifs. De façon complémentaire, la perspective système traite du génie logiciel. L'union de ces espaces fait émerger un ensemble de questions que doit se poser tout concepteur de systèmes interactifs multicibles.

3.2.1 *Perspective utilisateur*

D'un point de vue de l'usage (Figure 4), sans considérer l'ingénierie des logiciels, on recense trois types (ou natures, en référence à la Figure 4) de réactions possibles en cas de changement de contexte d'usage :

– une intervention sur la cible, par exemple, dans le cas de CamNote, baisser les volets si le soleil jaillit afin de maintenir un environnement obscur pour la projection du diaporama ;

– un remodelage de l'interface, par exemple, transformer un jeu de boutons radio en un menu déroulant pour économiser des pixels. Si les choix offerts dans ces boutons radio et menu déroulant restent les mêmes, la portée du remodelage se limite à de la présentation. Si, en revanche, les données manipulées (aussi appelées concepts) ou les tâches utilisateur changent, l'abstraction est modifiée. Que la portée

du remodelage relève de la présentation ou de l'abstraction, il faut se demander à quel niveau d'abstraction ces changements s'opèrent : affectent-ils les concepts du domaine, les tâches élémentaires (c'est-à-dire des tâches uniquement décomposables en actions physiques) ou des tâches abstraites (c'est-à-dire des tâches décomposables en sous-tâches). Sur l'exemple de CamNote, la télécommande fait l'objet d'un remodelage lors de son transfert sur PDA. Les flèches de défilement permettant de naviguer dans le diaporama sont remplacées par les boutons *Précédent* et *Suivant* : c'est un remodelage de type Présentation concernant les tâches élémentaires « *Passer au précédent* » et « *Passer au suivant* ». Le contrôle du flux vidéo devient, en revanche, impossible : c'est un remodelage de type Abstraction et Présentation concernant la tâche abstraite « *Télécommander* » dont « *Contrôler flux vidéo* » était sous-tâche ;

– une redistribution de l'interface sur l'ensemble des plates-formes composant l'environnement. Dans CamNote, la migration de la télécommande du PC vers le PDA est un exemple de *migration partielle*. Si le système interactif dans son intégralité avait migré, on aurait parlé de *migration totale*. Lors d'une migration partielle, une question supplémentaire s'impose : le grain de césure. La coupure dans l'IHM s'effectue-t-elle entre espaces de travail, entre concepts ou au niveau pixels ? Dans CamNote, cette césure s'opère entre espaces de travail. Une césure au niveau concepts signifierait qu'un ensemble de concepts manipulés par une même tâche se scinde lors de la migration. C'est le cas dans Built-It (Rautenberg *et al.*, 1998). Une césure au niveau pixels autoriserait typiquement l'affichage d'un morceau de bouton sur une plate-forme, le complément sur une autre. C'est le cas dans I-LAND (Streitz *et al.*, 1999).

La redistribution, qu'elle soit partielle ou totale, soulève la question du grain de reprise. Quatre grains sont identifiés :

– au niveau session, le contexte d'interaction de l'utilisateur, c'est-à-dire les éventuelles actions physiques qu'il a pu accomplir, est définitivement perdu. Le système interactif est réinitialisé, perdant le bénéfice d'éventuelles interactions passées. Typiquement, si l'utilisateur avait commencé à éditer une note sur PC « *Penser parler de ...* », ce contenu est définitivement perdu lors du transfert sur PDA ;

– au niveau tâche composée, l'achèvement de tâches abstraites (par exemple, pour CamNote, « *Faire une diapositive*») est préservé lors du changement de plate-forme dès lors que ses sous-tâches (« *Editer diapositive* » et « *Editer notes* ») sont accomplies. Ainsi, si l'utilisateur avait fini d'éditer la diapositive et de saisir ses notes, ce contenu se retrouve préservé sur PDA. Par contre, les éventuelles actions physiques amorçant la réalisation d'une nouvelle tâche sont définitivement perdues ;

– au niveau tâche élémentaire (par exemple, « *Editer diapositive », « Editer notes »* ou « *Choisir diapositive »*), ce même principe est reconduit. Les tâches achevées sont restituées telles quelles sur la nouvelle plate-forme. En revanche, les actions physiques ne suffisant pas à l'accomplissement d'une tâche élémentaire supplémentaire sont définitivement perdues ;

– enfin, au niveau le plus fin, les actions physiques sont préservées. Dans ce cas, lorsque l'utilisateur change de plate-forme, son contexte d'interaction est fidèlement reproduit. S'il avait tout juste tapé le « *P* » de « *Penser dire ...* », ce « *P* » est restitué sur PDA moyennant éventuellement un remodelage à l'appui.

Quelle que soit la réaction (intervention sur la cible, remodelage et/ou redistribution), la réaction peut être mise en œuvre par le système et/ou l'utilisateur et/ou tout autre acteur (autre système interactif ou autre individu). Symétriquement, pour la partie *action*, il convient d'identifier l'acteur (ou *initiateur* en référence à la figure 4) du changement. En effet, en reprenant l'exemple de la luminosité qui augmente, il serait malvenu d'abaisser les volets si ce changement est à l'initiative de l'utilisateur. Il a peut-être allumé la lumière. Dans la lignée de (Schmidt, 1999), on distingue trois critères possibles pouvant justifier une adaptation : l'entrée dans un contexte, le fait d'y rester (aussi dit maintien) et la sortie d'un contexte d'usage.

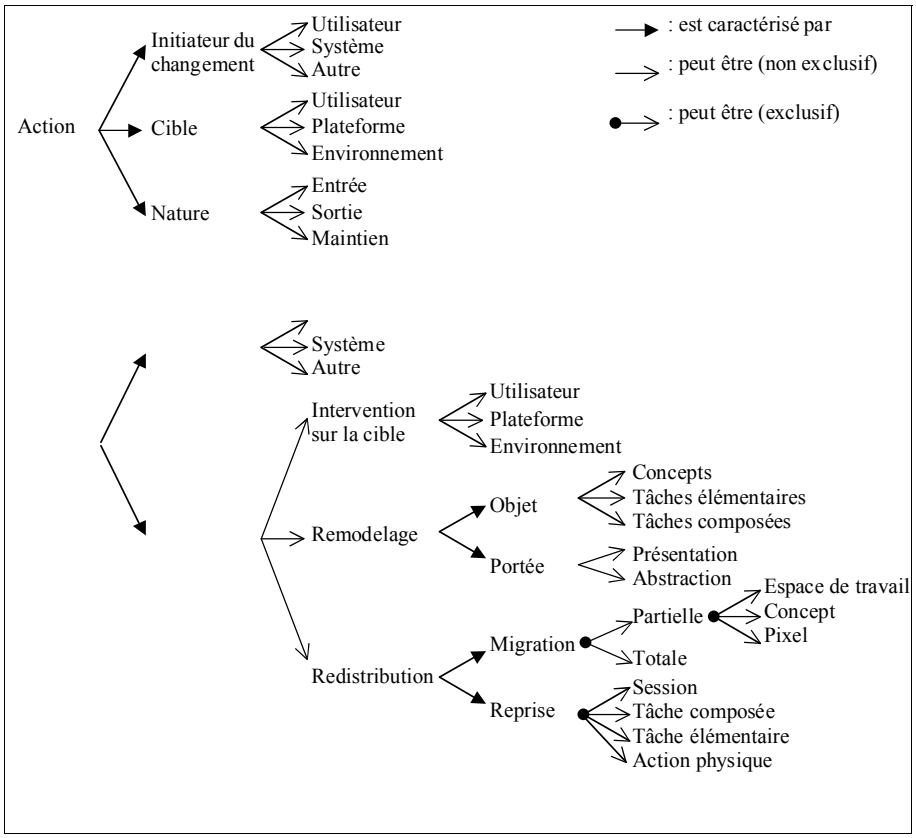


Figure 4. *Espace problème de l'adaptation selon une perspective utilisateur.*

La section suivante présente le dual de cet espace problème envisagé selon une perspective système.

3.2.2. *Perspective système*

La perspective système complète l'espace précédent en suscitant des questions de l'ordre du génie logiciel (Figure 5). Pour la partie *action*, il s'agit de préciser qui de l'utilisateur et/ou du système interactif et/ou d'un tiers (autre système interactif ou individu) est en charge de la capture du contexte d'usage, de la détection d'un changement dans ce contexte d'usage et de l'identification du nouveau contexte. Si ces étapes sont à la charge du système interactif, il faut prévoir, en tant que concepteurs, les capteurs et mécanismes logiciels requis.

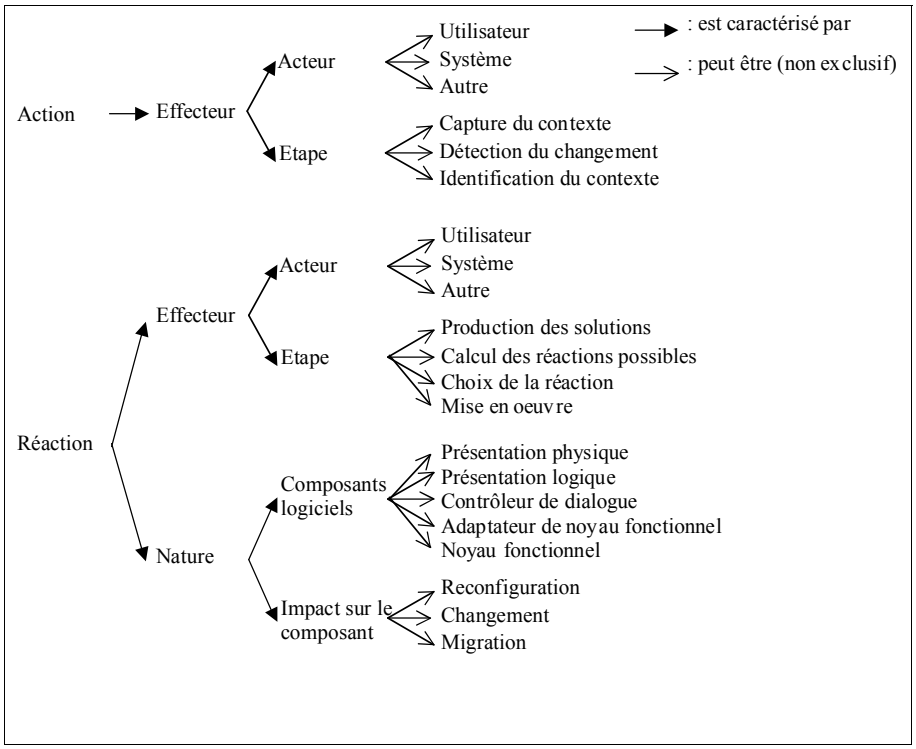


Figure 5. *Espace problème de l'adaptation selon une perspective système.*

De façon symétrique, pour la partie *réaction*, il s'agit d'identifier les acteurs en charge de la production des solutions possibles (l'ingénierie des différentes IHM à la base du remodelage et de la distribution), du calcul des réactions possibles (interventions sur la cible, remodelages et redistributions), du choix de la réaction puis de sa mise en œuvre. Le choix pourra reposer sur la nature des composants logiciels affectés. Conformément au modèle ARCH, ces composants s'échelonnent

de la présentation physique au noyau fonctionnel en passant par la présentation logique, le contrôleur de dialogue et l'adaptateur de noyau fonctionnel. L'impact peut consister en une reconfiguration du composant (le composant est préservé mais commuté en un nouvel état), un changement de composant (le composant n'est plus valide, il est remplacé par un autre) ou en une migration du composant (le composant est préservé mais migré vers une autre plate-forme). Le choix dépend, en partie, du *domaine de plasticité* du composant. Par domaine de plasticité, on entend l'ensemble des contextes d'usage pour lesquels le composant reste opérationnel et utilisable (Calvary *et al.*, 2001). La rupture de plasticité survient dès lors que le nouveau contexte d'usage est situé au-delà de ce domaine de plasticité. On appelle *seuil* cette frontière de plasticité (Figure 6).

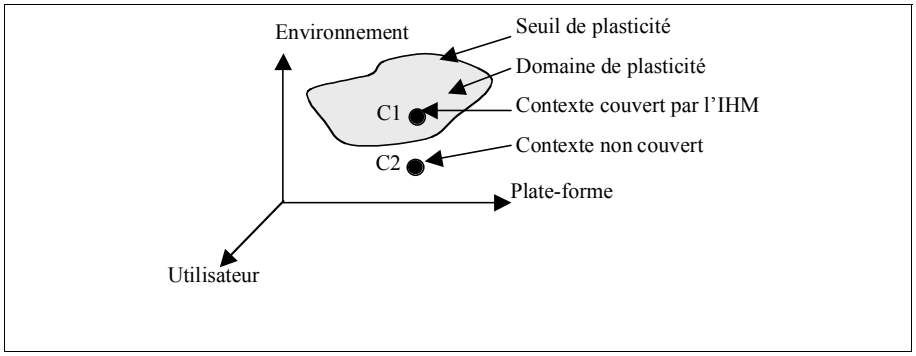


Figure 6. *Domaine de plasticité d'une IHM.*

La connaissance explicite des domaines de plasticité est nécessaire pour le choix de la réaction. La section suivante présente un cadre de référence permettant la production d'IHM de domaines de plasticité connus. Ce cadre de référence servira d'appui à un état de l'art dans le domaine.

4. Cadre de référence et état de l'art

Cette section présente un cadre de référence en plasticité. Ce cadre sert de base à une revue critique de l'état de l'art.

4.1. *Cadre de référence*

Le cadre de référence s'appuie sur les approches basées modèles (Szekely, 1996). Il distingue deux grandes catégories de modèles (Figure 7) :

– les *modèles initiaux* définis par le concepteur en point d'entrée du processus de conception ;

– les *modèles transitoires et finaux* obtenus étape par étape, à partir de ces modèles initiaux, jusqu'à dériver l'interface finale exécutable.

Parmi les modèles initiaux, on distingue :

– les modèles relatifs au domaine qui décrivent les concepts du domaine ainsi que les tâches utilisateur ;

– les modèles relatifs à la cible qui décrivent le contexte d'usage en termes d'utilisateur, de plate-forme et d'environnement ;

– les modèles relatifs à l'adaptation qui décrivent l'adaptation en termes d'évolution et de transition. L'évolution spécifie la réaction à mettre en œuvre en cas de changement de contexte d'usage. La transition préconise un accompagnement de l'utilisateur dans le changement pour une continuité de l'interaction.

Les modèles initiaux alimentent un processus de conception articulé autour de la production de trois modèles transitoires jusqu'à l'obtention de l'interface finale (Figure 7) :

– le lien tâches-concepts répertorie, pour chaque tâche utilisateur, les concepts manipulés par cette tâche. Tâches et concepts proviennent respectivement des modèles initiaux des tâches et des concepts ;

– l'interface abstraite structure l'IHM en espaces de travail et spécifie l'enchaînement entre espaces. Un espace de travail est un « lieu d'activité virtuel offrant les éléments nécessaires à la réalisation d'une ou plusieurs tâches » (Normand, 1992) ;

– l'interface concrète réifie les espaces de travail en fenêtres ou canevas ; leur contenu ainsi que leur enchaînement en objets d'interaction (aussi appelés *widgets* ou *interacteurs*). Les boutons, champs texte, menus déroulants, etc. sont des exemples d'interacteurs ;

– l'IHM finale est une version exécutable ou interprétable de l'IHM concrète.

Le processus produisant, étape par étape, ces modèles est dit *processus de réification*. Les modèles initiaux peuvent y être référencés à tout niveau d'abstraction. Plus ils sont référencés tard dans le processus, plus les modèles obtenus sont génériques, c'est-à-dire de domaines de plasticité vastes. L'opération inverse consistant à abstraire les modèles à partir, par exemple, de l'interface finale est dite *rétro-conception*. L'opération consistant à traduire un modèle en un homologue (modèle de même niveau de réification) pour cibler un autre contexte est dite *traduction* (Figure 7). Réification, rétro-conception et traduction peuvent être effectuées de manière manuelle, semi-automatique ou automatique (Figure 7).

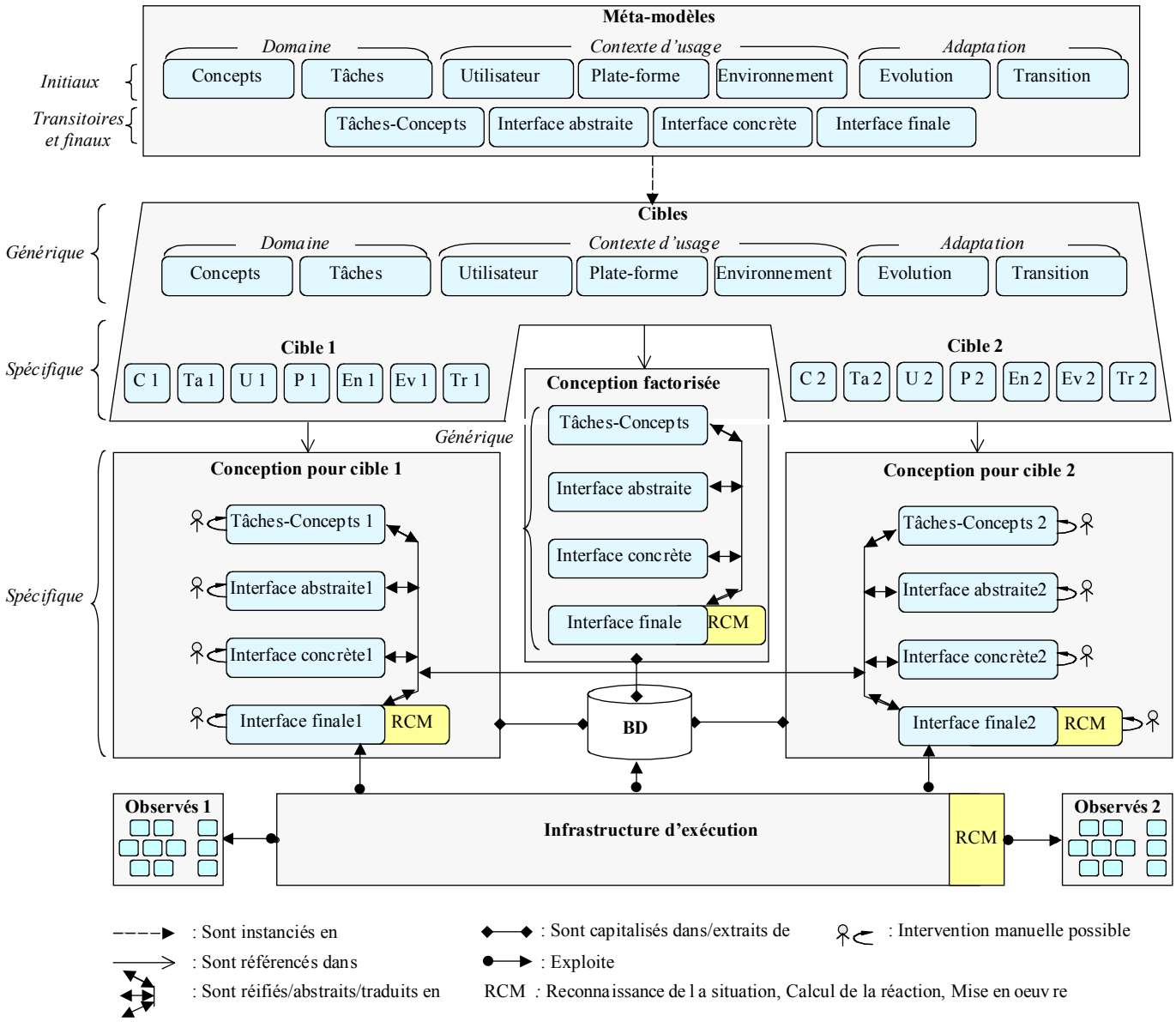


Figure 7. *Le cadre de référence en Plasticité des Interfaces.*

Les modèles, qu'ils soient initiaux, transitoires ou finaux, obtenus par réification, rétro-conception ou traduction, sont des instances de méta-modèles (Figure 7). UML typiquement fournit une base pour l'expression des concepts du domaine ; ConcurTaskTrees pour les tâches utilisateur (Paternò, 1999). A un niveau méta, ces méta-modèles peuvent être vus comme des instances d'un méta-méta-modèle (non représenté en Figure 7 pour des raisons de lisibilité) permettant d'étiqueter le caractère générique/spécifique de toute information modélisée. Par exemple dans CamNote, seule la plate-forme change lors du transfert de la télécommande du PC vers le PDA. Les informations relatives à l'utilisateur et à l'environnement sont inchangées. D'un point de vue du génie logiciel, il est intéressant de pouvoir exprimer cette différence entre modèles génériques (c'est-à-dire communs) et spécifiques. Le cadre de référence permet cette factorisation, à l'origine, soulevée par (Thevenin, 2001) (Figure 7). Chaque modèle, quel que soit son niveau de réification, peut être stocké dans une base de données. Ainsi, lors d'un changement de contexte d'usage, ces modèles sont directement réutilisables.

En termes d'exécution (partie basse de la Figure 7), des mécanismes sont nécessaires pour la reconnaissance de la situation (R dans le RCM de la Figure 7), le calcul (C) et la mise en œuvre de la réaction (M). Ces mécanismes sont embarqués dans les interfaces finales et/ou pris en charge par une infrastructure dédiée à l'exécution. Le calcul de la réaction peut reposer sur les modèles observés, c'est-à-dire la valeur courante de ces modèles au fil de l'interaction (Figure 7). Peuvent être considérés les modèles initiaux mais aussi les modèles transitoires et finaux.

Si ce cadre de référence aide le concepteur à se poser les bonnes questions, il permet aussi de caractériser graphiquement les approches existantes.

4.2. *Etat de l'art : couverture et enseignements*

On trouvera dans Calvary *et al.* (2002, 2003) des états de l'art menés par instanciation du cadre de référence. Cette méthode graphique facilite la perception et la comparaison d'approches souvent voisines. Les enseignements de ces revues montrent que :

– jusqu'aux années 2000, les efforts se concentraient sur des approches à base de réification, avec bien plus un objectif d'automatisation que de multiciblage. Le multiciblage réhabilite ces approches basées modèles mais se heurte au faible appétit des industriels pour les modèles. La rétro-conception apparaît alors pour tenir compte de la pratique et générer les modèles amont à partir des interfaces finales. Vaquita (Vanderdonckt *et al.*, 2001) en est un exemple typique. La combinaison des deux, réification et rétro-conception, devient alors prometteuse pour converger vers des modèles de qualité tout en respectant la pratique industrielle. Mais les outils actuels ne combinent pas encore ces approches. Les traductions commencent à être intégrées, renforçant le sentiment d'une bonne exploration des opérations possibles. En revanche, la capitalisation et la réutilisation logicielles restent des vœux pieux ;

– l'exécution est bien moins étudiée que la conception. Les rares travaux existants comme ceux de Luyten *et al.* (2001) montrent qu'il est vain de travailler à ce niveau sans des méta-modèles solidement réfléchis ;

– en termes de méta-modèles, seules des bases solides existent pour les modèles traditionnels tels que les concepts et les tâches. Ces modèles restent néanmoins à enrichir pour les besoins précis de la plasticité. Les travaux se concentrent aujourd'hui sur la définition du contexte (Dey *et al.*, 2000) et son intégration dans des boîtes à outils (Salber *et al.*, 1999). Les modèles de l'adaptation sont, en revanche, ignorés.

Aussi, au-delà de l'espace problème et du cadre de référence, nos contributions se situent au niveau des méta-modèles. Nous en examinons deux, les interacteurs et l'évolution, qui nous orientent vers des approches à agents.

5. Vers une approche à agents

Les nouvelles exigences de remodelage et de redistribution appuient le génie logiciel dans son incitation à la modularité. Désormais, un espace de travail, un concept voire un pixel peut faire l'objet d'un remodelage ou d'une migration, dictés par un modèle d'évolution. L'idée est alors de raisonner à la granularité des *interacteurs*, de les doter de capacités d'adaptation tout en veillant à la cohérence ergonomique globale du système interactif. Les approches à agents appréciées pour leur modularité deviennent alors des candidats naturels à la mise en œuvre des systèmes interactifs. Cette section introduit la notion de *comet*, c'est-à-dire d'*interacteur plastique*, puis en examine les modèle et mécanismes d'évolution.

5.1. *Les comets, des interacteurs plastiques*

La notion d'« *interacteur* » a été introduite par Faconti et Paterno en 1990 comme étant l'abstraction d'une entité capable à la fois d'entrée et de sortie dans un système interactif graphique (Faconti *et al.*, 1990). Dès 1993, Duke et Harrisson notent que cette notion est intéressante pour raisonner sur les systèmes interactifs en général et que la définition doit donc être revue pour être portée au-delà du graphique (Duke *et al.*, 1993). Depuis, plusieurs contributions à dominante ingénierie (par opposition à l'approche analytique de Duke) ont été apportées, en particulier :

– Vanderdonckt (1997) qui, par sa distinction entre OIC (Objets d'Interaction Concrets) et OIA (Objets d'Interaction Abstraits) fait émerger deux niveaux d'abstraction dans la notion d'interacteur. Il identifie six classes d'OIAs : les OIAs statiques, les OIAs d'action, de défilement, de contrôle, de dialogue et de feed-back ;

– Markopoulos (2001) qui, dans la lignée de Duke *et al.* (1993), adopte une approche flux de données permettant la composition d'interacteurs. Il propose le

modèle d'architecture ADC (Abstract-Display-Control) faisant émerger deux unités : un contrôleur, d'une part, un bloc Abstraction-Présentation, d'autre part ;

– Thevenin (2001) qui recentre les interacteurs sur les concepts et tâches du domaine. Un interacteur est décrit par « un nom ; la liste des données qu'il est capable de représenter ; la liste des tâches qu'il propose ; ainsi que son empreinte graphique » ;

– Crease (2001) qui dote les interacteurs de faculté d'adaptation en sortie. L'interacteur devient multimodal, évoluant dans les mondes graphique et sonore.

De cette revue de l'état de l'art, nous retenons que :

– si Crease innove en dotant les interacteurs de faculté d'adaptation, cette capacité se limite à la sortie et ne couvre pas la réalité mixte (Milgram, 1994) ;

– la qualité de service des présentations n'est pas modélisée. Seul Thevenin amorce la réflexion en chiffrant les requis de l'interacteur en termes de surface d'affichage ;

– les travaux se complètent en termes d'architecture logicielle montrant l'intérêt d'un contrôleur (ADC et Crease) et d'un raisonnement à différents niveaux d'abstraction (Vanderdonckt et Thevenin).

L'idée est alors de marier les deux grains d'analyse, interacteurs et systèmes interactifs, en considérant l'interacteur comme un mini-système interactif doué de faculté d'adaptation. La notion de *comet* (COntext-sensitive Multi-target gadgET) est introduite pour désigner cette nouvelle génération de widgets.

Conformément au cadre de référence en plasticité, une comet est modélisée comme un système interactif :

– structuré en quatre niveaux de réification : liens tâches-concepts (désormais appelés abstractions), interfaces abstraites, concrètes et finales ;

– doué de capacité de réflexivité à tout niveau de réification. Chaque modèle transitoire et final est capable de publier son domaine de validité en termes de modèles initiaux supposés ;

– polymorphe (c'est-à-dire doté d'au moins deux alternatives) à, au moins, un niveau de réification donné (abstractions, interfaces abstraites, concrètes ou finales) ;

– dont les interfaces finales sont dotées d'une qualité de service. Cette qualité de service exprime (a) le coût de mise en œuvre de l'interface (les ressources matérielles et logicielles de la plate-forme, les requis environnementaux et humains – ces coûts correspondent exactement aux modèles initiaux relatifs au contexte d'usage de l'interface finale) (b) les propriétés d'utilisabilité garanties par l'interface (c) les effets de bord potentiels de l'interface sur le contexte d'usage (par exemple, l'augmentation du niveau sonore). Ces effets de bord sont désormais banalisés en la propriété de *viscosité* sur le contexte d'usage. La viscosité n'est pas ici à interpréter au sens de (Green, 1990), c'est-à-dire comme une résistance au changement.

La figure 8 propose une modélisation UML de cette notion de comet. Elle introduit la classe *ComposantComet* en tant que composant réflexif qui connaît et sait exporter ses modèles initiaux. Une comet est composée d'abstractions, d'interfaces abstraites (IU abstraites), d'interfaces concrètes (IU concrètes) et d'interfaces finales (IU finales). Ces composants sont des *ComposantComet* se correspondant mutuellement par réification. Leurs valeurs (c'est-à-dire les valeurs effectives des modèles des concepts, tâches, etc. les définissant) sont spécifiées par le concepteur et/ou calculées par le générateur produisant les modèles transitoires. La comet sera dite plastique (valeur 1) pour un ensemble de propriétés d'utilisabilité P et un ensemble de contextes donnés C si toute propriété de P est satisfaite sur tout contexte de C. L'IU concrète connaît son style (par exemple, le style bouton) et peut dire si son usage est typique dans un contexte donné. Cet usage est un réel compris entre 0 et 1, apprécié par des spécialistes du domaine ou inféré par un outil d'analyse de systèmes interactifs. Les valeurs fournies (*estPlastique* et *usage*) servent à choisir la comet la plus adaptée à un contexte d'usage donné, au regard des propriétés requises.

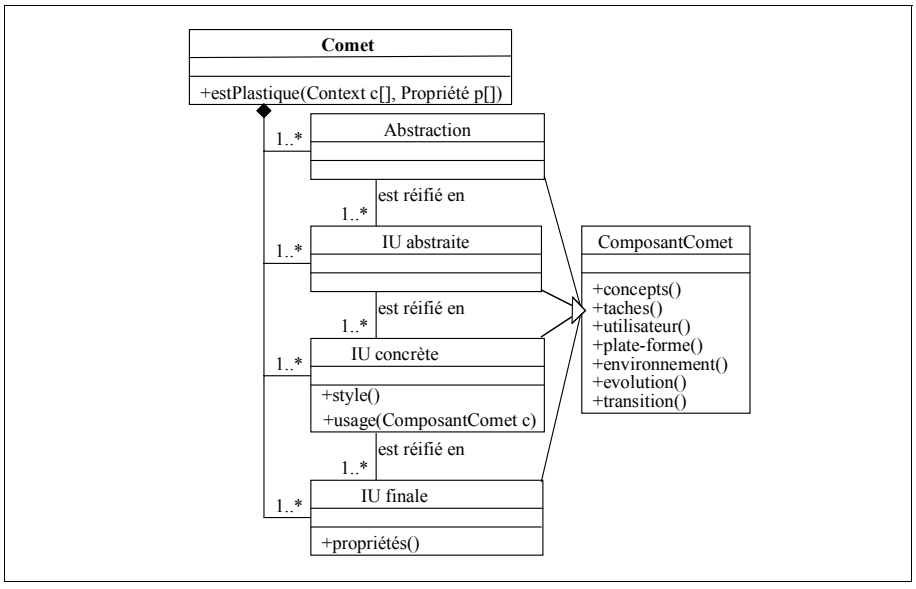


Figure 8. *Modélisation UML de la notion de comet.*

Dans la lignée du modèle d'architecture à agents PAC (Presentation Abstraction Control) (Coutaz, 1987), nous proposons le modèle rePAC peuplant récursivement la facette P d'un agent PAC (Figure 9) :

– l'abstraction du P maintient l'abstraction de la présentation, c'est-à-dire l'état de l'interaction. Cette facette est déterminante pour des migrations d'IHM *à chaud*,

c'est-à-dire des migrations conservant tout ou partie de l'état de l'interaction. En référence à l'espace problème, il s'agit des grains de reprise niveaux tâches composées, tâches élémentaires et actions physiques ;

– la facette contrôle du P assure la gestion de l'adaptation : réception ou diffusion du nouveau contexte d'usage, des adaptations envisageables, de l'adaptation à mettre en œuvre (réception) ou mise en œuvre (diffusion). Le sens de la communication (réception versus diffusion) dépend de la politique adoptée. Les politiques sont examinées ci-après. Pour le calcul de l'adaptation, c'est cette facette contrôle C_p qui gère les capacités de polymorphisme de la comet. Ces capacités peuvent, par exemple, être maintenues sous la forme d'une liste mémorisant, à chaque niveau de réification (abstraction (A), IU abstraite (IUA), IU concrète (IUC), IU finale (IUF)), les différentes alternatives possibles. Par exemple :

$$
\begin{array}{llll}
((A_1 & (IUA_{11} & (IUC_{111} & (IUF_{1111} \ldots IUF_{111m})) \\
 & & \ldots & \\
 & & (IUC_{11n} & (IUF_{11n1} \ldots IUF_{11no}))) \\
 & \ldots & & \\
 & (IUA_{1p} & (IUC_{1p1} & (IUF_{1p11} \ldots IUF_{1p1q})) \\
 & & \ldots & \\
 & & (IUC_{1pr} & (IUF_{1pr1} \ldots IUF_{1prs})))) \\
\ldots & & & \\
(A_t & \ldots & &))
\end{array}
$$

– la facette présentation du P contient la présentation active de la comet. Elle encapsule les widgets traditionnels en les dotant notamment de capacité de réflexivité pour qu'ils puissent exporter leurs modèles initiaux. A terme, ces widgets traditionnels seront distribuables sur l'ensemble des plates-formes pour assurer la redistribution des IHM.

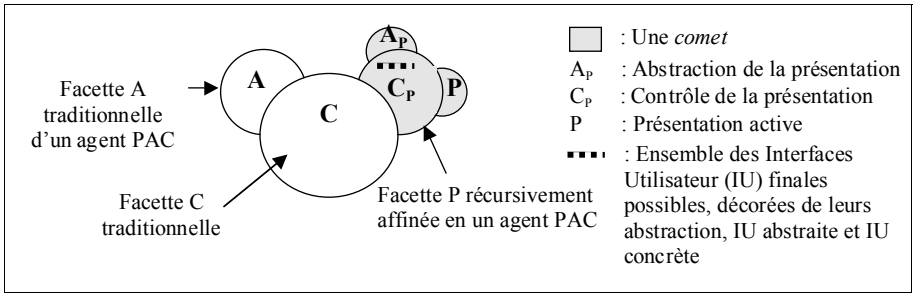


Figure 9. *Architecture logicielle d'une comet : le modèle rePAC*.

Dans rePAC, la communication entre facettes reprend les principes de PAC : les facettes A_p et P ne se connaissent pas directement. Elles échangent via la facette C_p. Le système interactif devient une collection d'agents rePAC, les facettes C des pères et fils se connaissant mutuellement ; idem pour les facettes C_p. La figure 10 donne l'architecture logicielle de la version PC de CamNote. Sur cet exemple, les hiérarchies des facettes C et C_p coïncident. On pourrait envisager des canaux de communication spécifiques entre facettes C_p pour, par exemple, faciliter le maintien de la cohérence ergonomique du système interactif global. Dans ce cas, la facette C_p de l'agent « *Gérer le diaporama* » serait le père des facettes C_p de tous les autres agents.

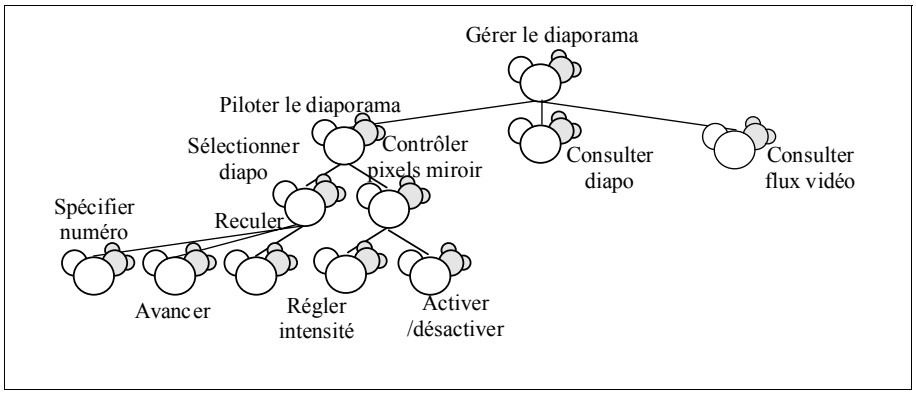


Figure 10. *Architecture logicielle de la version PC de CamNote.*

La section suivante traite des modèles et mécanismes d'évolution.

5.2. *Modèle et mécanismes d'évolution*

Nous identifions quatre classes de stratégies :

– l'adaptation par polymorphisme. C'est une stratégie qui conserve la comet 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 la comet. Elle la remplace par une ou N autres. On distingue trois types de substitution selon leur cardinalité : la substitution 1-1 consiste à remplacer la comet par une autre ; la substitution 1-N remplace la comet par N autres ; la substitution N-1 remplace la comet et N-1 autres par une unique comet ;

– l'adaptation par ajout. Une comet est ajoutée 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. Une comet est supprimée 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 la comet dans sa prise en charge de l'adaptation : reconnaissance de la situation, calcul et mise en œuvre de la réaction. Pour chacun de ces aspects, un degré d'autonomie est accordé à la comet. Ce degré varie de 0 à 1, donnant lieu à quatre classes de politiques :

– la politique *non concertée externe* prive la comet de toute autonomie. L'évolution est calculée et mise en œuvre par un tiers (une autre comet ou l'infrastructure d'exécution) ;

– à l'opposé, la politique *non concertée interne* confère à la comet une autonomie maximale. Elle décide seule de la réaction à mettre en œuvre et l'applique en pleine autonomie ;

– les politiques *concertées* correspondent à une autonomie partielle : une négociation s'établit entre la comet et un tiers (une autre comet ou l'infrastructure d'exécution) pour la prise en charge de l'adaptation. Tandis qu'une version optimiste autorisera la comet 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 œuvre.

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 œuvre.

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 œuvre. 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 asssortie d'un texte en langue natuelle qui en décrit l'effet ;

– les attributs décorent la proposition d'une force, d'un qualificatif et indiquent l'auteur de la proposition (utilisateur/comet). 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 tâches-concepts, 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. Nous nous orientons vers les graphes conceptuels pour leur lisibilité et leur lien avec la logique.

Les graphes conceptuels sont un type de réseau sémantique (Sowa, 1984). Un réseau sémantique est un système de représentation graphique des connaissances basé sur des nœuds interconnectés par des arcs. Il prend la forme d'un graphe biparti étiqueté par des items lexicaux. Une des classes de sommets correspond à des concepts, l'autre à des relations entre concepts. Ces concepts et relations proviennent de supports. La figure 11 présente une partie du support utilisé dans CamNote.

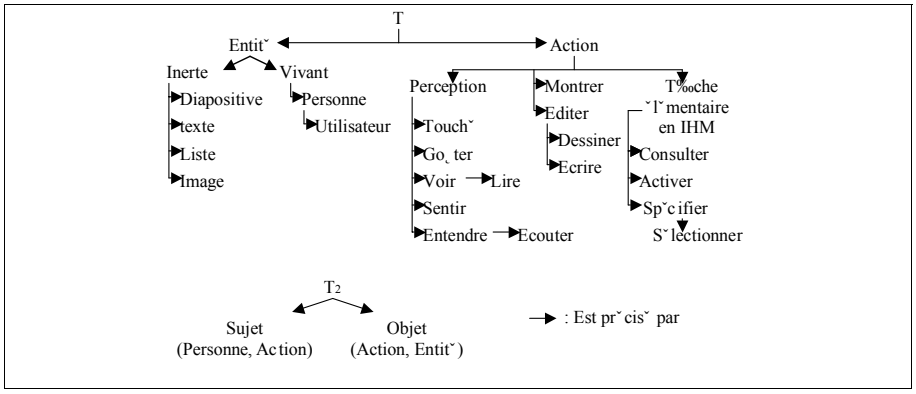


Figure 11. *Une partie du support utilisé dans CamNote. T désigne le concept universel ; T2 la relation binaire universelle.*

La figure 12 montre un exemple de graphe conceptuel exprimant le fait qu'un utilisateur peut écrire et montrer une diapositive contenant du texte et au moins une image. Cet exemple montre la force des graphes conceptuels : leur puissance d'expression et leur extrême lisibilité due à une représentation graphique. Si ces

graphes conceptuels sont, comme l'ont montré (Sowa, 1984) et (Chein *et al.*, 1992) équivalents à une logique du premier ordre, leur atout provient de cette lisibilité.

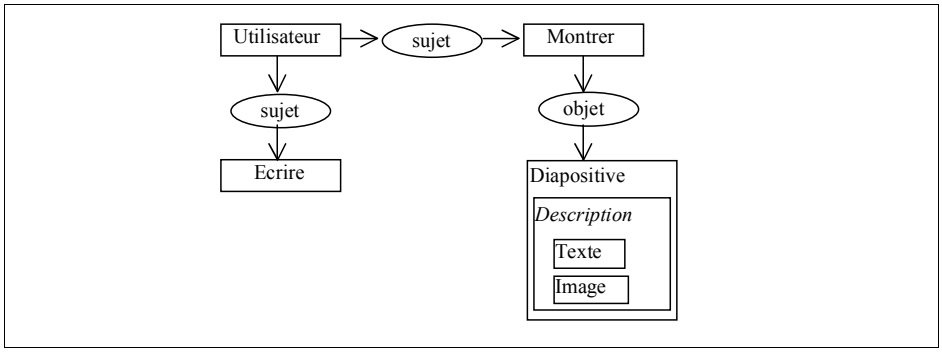


Figure 12. *Un exemple de graphe conceptuel dans CamNote.*

La base de données devient un graphe des descriptions structuré selon trois relations : « *hérite de* », « *est composé de* » et « *implante* » (Figure 13).

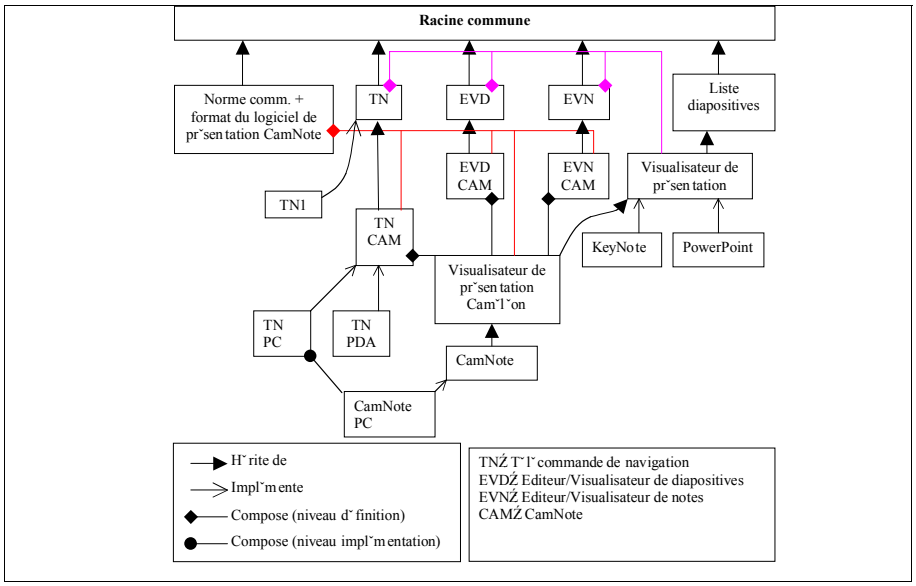


Figure 13. *Un sous-ensemble du graphe des descriptions de CamNote. Chaque nœud comporte un graphe conceptuel qui le décrit. Le graphe conceptuel et le support d'un nœud sont contraints par les relations qu'entretient ce nœud avec le reste du graphe.*

Les relations « *hérite de* », « *est composé de* » et « *implante* » sont contraintes par les graphes et leurs supports, et réciproquement :

– A hérite de B : le graphe de B se projette dans A. Le support de A étend celui de B ;

– A est composé de B : 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.

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 du genre : 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 graphe des descriptions couvre migration et remodelage. Par exemple, lorsque Marc migre la télécommande sur PDA, il s'agit, dans ce graphe, de localiser les composants télécommandes fonctionnant sur PDA. Une requête est formulée : elle décrit, sous la forme de graphe conceptuel, le composant recherché (Figure 14).

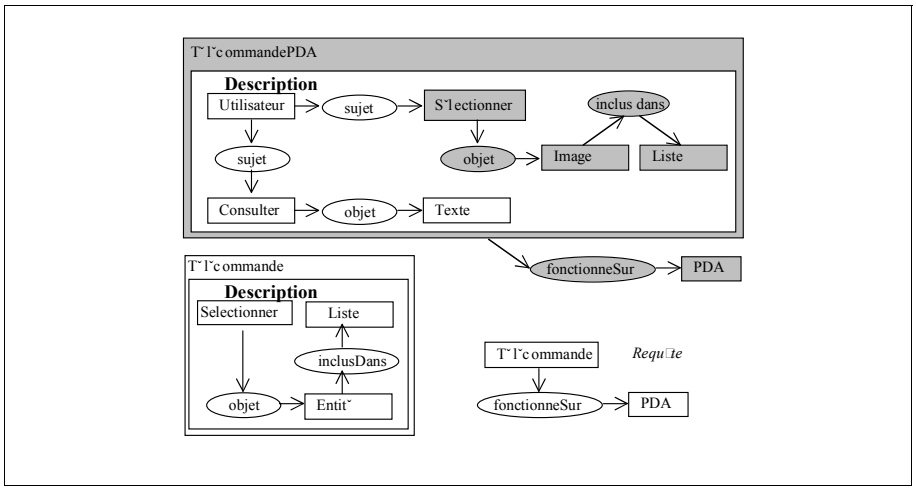


Figure 14. *Formulation d'une requête pour la recherche d'une télécommande fonctionnant sur PDA. Le composant TélécommandePDA répond à la description.*

Le parcours du graphe se fait récursivement à partir d'un nœud, par exemple, celui correspondant à la télécommande en cours d'utilisation (Figure 15). Une fonction de test est utilisée à chaque étape pour indiquer si le parcours doit se poursuivre ou non. La sélection d'un candidat s'opère si le graphe requête se projette dans le graphe conceptuel décrivant le nœud. Sur l'exemple de la Figure 14, en posant comme fonction de test « *Télécommande se projette dans le graphe conceptuel du nœud parcouru* », on obtient le marquage présenté en Figure 15 : seuls les composants « TN1 » et « TN PDA » correspondent à la requête. Dans le cas de CamNote, une négociation est mise en œuvre avec l'utilisateur pour lui proposer ces deux alternatives et recueillir sa préférence.

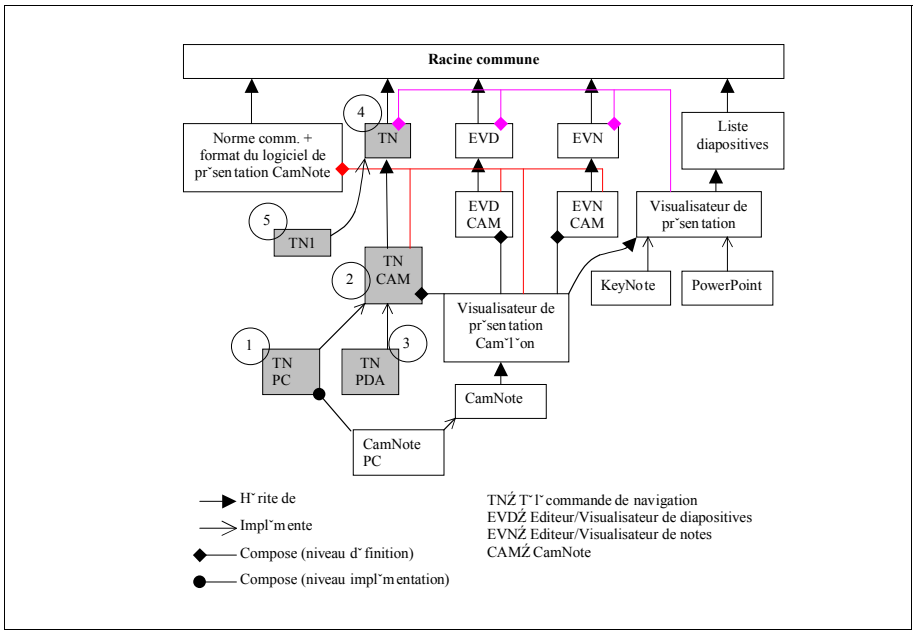


Figure 15. *En grisé, l'ensemble des nœuds parcourus lors de la recherche d'une télécommande fonctionnant sur PDA. L'ordre de parcours des nœuds est indiqué dans les cercles. Seuls « TN PDA » et « TN1 » correspondent à la requête.*

Grâce aux relations « *hérite de* », « *est composé de* » et « *implante* » établies entre les nœuds du graphe, le parcours peut être considérablement réduit. Par exemple, à l'étape 1, il est inutile d'explorer la voie CamNote PC puisque celui-ci n'est pas une télécommande : il en est simplement composé. Le même raisonnement s'applique à l'étape 2 où on ne parcourra que les nœuds « TN PDA » et « TN » pour la relation d'héritage qui les unie à « TN CAM ». Le « Visualisateur de présentation Caméléon » ne sera pas exploré puisque simplement composé d'une télécommande.

En pratique, dans le démonstrateur CamNote, les comets, le moteur d'évolution et le graphe des descriptions sont implémentés en TCL / C++. Le choix de TCL est motivé, en premier lieu, par le caractère interprété de ce langage permettant à l'utilisateur final de surcharger, à la volée, le modèle d'évolution. En second lieu, des atouts supplémentaires tels que sa portabilité et sa bibliothèque graphique (TK) ont été considérés. Le choix de C++ provient de la librairie *Cogitant* du LIRMM (cogitant.sourceforge.net/) utilisée pour la mise en œuvre des graphes conceptuels.

Si par le démonstrateur CamNote, l'approche et notamment sa faisabilité technique ont été validées sur un cas d'étude, le passage à l'échelle est un problème ouvert non encore appréhendé expérimentalement. De façon prédictive, la section suivante traite de la complexité de l'approche. Elle en identifie les freins éventuels et leviers potentiels pour en atténuer le rapport coût/bénéfice.

5.3. Considérations sur la complexité de l'approche

Les approches basées modèles ont de tout temps divisé la communauté. Tandis que certains y voient un succès technologique, d'autres, plus pessimistes, ne sont toujours pas convaincus du bien fondé de l'approche. Dans cette confusion, les praticiens restent prudents et adoptent comme seul critère le rapport coût/bénéfice. Les bénéfices traditionnellement reconnus au paradigme sont les suivants : la capture de la sémantique à différents niveaux d'abstraction (Sukaviriya *et al.*, 1994) ; la définition d'un socle stable et structuré de connaissances, structure d'accueil à la connexion d'outils d'exploration et d'exploitation (Märtin, 1996). L'adaptation en est un nouvel exemple. Les retombées du paradigme sont nombreuses (Szekely *et al.*, 1995) :

– un gain en cohérence et une réutilisabilité favorisés par un partage de connaissances sémantiques entre tous les composants du système interactif ;

– une meilleure intelligibilité, et par voie de conséquence, maintenabilité du système interactif ;

– une extensibilité facilitée par la forme déclarative des spécifications ;

– la promotion des développements itératifs par le caractère exécutable des modèles : même incomplets, ceux-ci offrent aux concepteurs un terrain d'expérimentation concret. C'est le cas typiquement dans CamNote où le caractère interprété du langage permet, par tâtonnements successifs, de converger vers un modèle d'évolution le plus ajusté possible.

Malgré ces atouts, le paradigme ne fait pas l'unanimité. Des inconvénients récurrents lui sont reprochés. Notamment :

– des modèles trop souvent partiels, privilégiant toujours une facette particulière du système interactif (Märtin, 1996) : les tâches dans ADEPT (Johnson *et al.*, 1993), le domaine dans MECANO (Puerta, 1996), la présentation dans Trident (Vanderdonckt, 1997) et l'application dans Mastermind (Szekely *et al.*, 1995). Ici, tous les modèles sont conciliés en un même cadre de référence. Leur spécification

est optionnelle. Le concepteur peut, en particulier, s'affranchir de la spécification des modèles amont (initiaux et transitoires) pour coder directement son interface finale s'il le souhaite. La réutilisabilité en sera réduite. Tout dépend des critères que le concepteur souhaite privilégier dans son approche. De la même façon, la spécification du modèle d'évolution n'est pas un requis. Sa spécification à la granularité de la comet n'en est pas un non plus. Le modèle d'évolution, si modèle il y a, peut être concentré en un modèle unique orchestrant l'adaptation du système interactif dans son intégralité. L'approche modulaire consistant à permettre sa ventilation au sein de comets plus élémentaires est une possibilité à laquelle le concepteur n'est pas contraint de recourir. Un compromis vers lequel il semblerait raisonnable de s'orienter consisterait à doter les comets de type « espaces de travail » (fenêtres et panneaux) de modèles d'évolution veillant, en particulier, à la cohérence ergonomique de leur contenu, et ceci récursivement ;

– des modèles trop souvent spécifiques ou dépendants des plates-formes compromettant, en final, tout espoir de réutilisation (Märtin, 1996). Ici, le cadre de référence incite justement le concepteur à s'interroger quant au caractère générique versus spécifique de toute information ;

– un manque de flexibilité dû à une expressivité limitée des langages de modélisation. Les graphes conceptuels dont la souplesse et la puissance ont fait leur preuve en recherche d'information échappent ici à la critique ;

– les performances limitées dues à une modélisation trop riche et à l'interprétation de celle-ci. L'organisation des règles d'évolution au sein d'automates et la structuration du graphe des descriptions sont ici des leviers intéressants ;

– les difficultés d'utilisation liées à l'apprentissage d'un langage de spécification dédié. Dans certains cas, la spécification s'apparente à de la programmation. Ici, l'usage des graphes conceptuels n'est pas obligatoire pour la description des propositions de réaction. L'intérêt est d'expliciter la sémantique pour, par exemple, détecter des contradictions. Mais ce but peut, dans une moindre mesure, être atteint en se basant simplement sur le nom des fonctions logicielles implémentant la mise en œuvre de la réaction. Il s'agit, par exemple, de déconseiller l'emploi d'une fonction f donnée pour certaines conditions de ses paramètres. Cette alternative présente deux inconvénients : d'une part, elle suppose une connaissance de l'API du système interactif et compromet donc la possibilité de spécification par l'utilisateur final ; d'autre part, le raisonnement sera moins puissant. On perd, en particulier, la possibilité d'exclure certaines réactions pour, non pas leur nom, mais les effets de bord qu'elles engendrent.

Si, en ce qui concerne les modèles, des arguments ou alternatives semblent toujours poindre, il serait néanmoins intéressant d'apprécier le seuil au-delà duquel la spécification est jugée trop onéreuse. L'évaluation devant, par nature, être circonstanciée, un recul est nécessaire pour cette estimation.

Au-delà des modèles, on identifie une deuxième source de complexité : le graphe des descriptions. Si la spécification des modèles était optionnelle, il est, par

contre, ici difficile d'imaginer des réactions de type migration sans une structure à l'appui capitalisant les composants disponibles : c'est le graphe des descriptions. Cette structure a une double vocation :

– capitaliser, lors de leur construction, les systèmes interactifs. La complexité se situe ici au niveau de la structuration : une *bonne* structuration facilitera l'insertion de nouveaux composants. Il s'agira, en particulier, d'estimer la bonne granularité des composants et de mettre à profit la puissance des graphes emboîtés pour décrire chaque nœud à différents niveaux d'abstraction. Rappelons que les graphes emboîtés permettent récursivement de décrire le nœud d'un graphe par un autre graphe. Cet outil devrait modérer les craintes légitimes quant à la complexité du graphe des descriptions. Peut-être un jour des méthodes formelles émergeront-elles, par ailleurs, pour évaluer la qualité d'un graphe, comme c'est le cas aujourd'hui en bases de données avec les formes normales ? Nous n'en sommes pas encore là ;

– soutenir les recherches à l'exécution. Là encore, la qualité de la structuration sera déterminante : une *bonne* structuration permettra d'optimiser le parcours des nœuds. Le choix du nœud de départ sera aussi prépondérant.

Outre la structuration et les algorithmes de parcours du graphe, un autre verrou se pose : celui de la cohérence lexicale et syntaxique du graphe. Comment gérer des relations non bijectives entre les mots et les concepts ? Comment comparer deux graphes et inférer, par exemple, leur équivalence syntaxique ? De tels problèmes ne pourront être fondamentalement résolus que par la définition d'une ontologie par domaine applicatif. Techniquement, on pourrait s'orienter vers des patrons de graphes normalisant l'expression de l'information. Ce travail de « normalisation » devrait se situer à deux niveaux : au sein des graphes conceptuels, par l'adoption d'un vocabulaire standardisant les descriptions selon une grille d'analyse donnée ; au niveau des nœuds par un consensus quant à la définition fonctionnelle des systèmes interactifs. Typiquement, qu'entend-on par « visualisateur de diapositives » ?

De nombreuses questions restent ouvertes. Elles alimentent nos perspectives.

6. Conclusion et perspectives

En conclusion, cet article visait à fournir une vision unifiée et représentative de l'avancement des recherches en matière de plasticité des interfaces. Après un espace problème du domaine, il présente les enseignements d'un état de l'art mené sur la base de la nouvelle version d'un cadre de référence en plasticité. Un rapprochement avec les approches à agents est alors esquissé. Il envisage l'adaptation à une plus fine granularité : celle des interacteurs. Il emprunte à la Recherche d'Information la notion de graphe conceptuel pour décrire sémantiquement les systèmes interactifs. Il met à profit les opérations de projection pour identifier, à l'exécution, le composant vers lequel commuter.

Si le démonstrateur CamNote a aujourd'hui prouvé la complétude et la correction des choix techniques, l'inconnue reste le facteur d'échelle. Quelle est l'utilisabilité des outils proposés ? En particulier, les défauts sont-ils inhérents à l'approche ou résultent-ils d'un biais lié à l'inadéquation des outils ? On pressent typiquement l'intérêt d'un outil de comparaison de graphes mettant en exergue leurs parties communes et spécifiques.

Au-delà des perspectives techniques et méthodologiques, notons que l'IHM, en tant que discipline, confirme ici son caractère d'emprunt. Plus qu'une affinité avec l'IA ou la Recherche d'Information, c'est une incitation à un rapprochement entre communautés techniques qu'il faut voir ici. Personne n'est désormais armé pour appréhender isolément des systèmes interactifs aussi complexes que ceux qui se profilent. De même que le bienfait des alliances avec des ergonomes, sociologues et psychologues est aujourd'hui bien établi et plus que jamais nécessaire, des rapprochements entre communautés techniques s'imposent aussi : l'électronique en fait partie pour le développement d'objets nomades. Aussi, la conclusion de (Coutaz, 1991) est-elle toujours d'actualité : il nous faut "persévérer dans le sens de l'ouverture des frontières entre les disciplines afin que la mise en commun du savoir d'équipes de formation complémentaire conduise à des idées fécondes assorties de solutions sures". Et bien souvent, revisitons les solutions du passé. L'architecture logicielle, par exemple, ou la multimodalité retrouvent ici un souffle nouveau. Finalités hier, ils deviennent aujourd'hui moyens d'adaptation.

7. Remerciements

Ce travail a bénéficié du soutien du projet européen IST CAMELEON (IST-2000-30104). Nous remercions particulièrement Jean Vanderdonckt, membre du projet.

8. Bibliographie

Abowd G.D., Atkeson C.G., Hong J., Long S., Kooper R., Pinkerton M., Cyberguide: A Mobile Context-Aware Tour Guide, GVU Technical Report GIT-GVU-96-27, December 1996.

Anabuki M., Kabuka H., Yamamoto H., Tamura H., Welbo : An embodied Conversational Agent Living in Mixed Reality Space, *Videos of CHI'2000*.

Bérard F., Coutaz J., Crowley J.L., Le tableau Magique, *Ergonomie et Informatique Avancée-Interaction Homme-Machine (ErgoIHM'2000)*, D.L Scapin & E. Vergisson (eds), CRT ILS & ESTIA, Octobre 2000, Biarritz, pp. 33-40.

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.

Calvary G., Coutaz J., Thevenin D., Limbourg Q., Souchon N., Bouillon L., Vanderdonckt J., Plasticity of User Interfaces: A Revised Reference Framework, *First International Workshop on Task Models and Diagrams for User Interface Design TAMODIA'2002*, Bucarest, 18-19 July 2002, pp 127-134.

Calvary G., Coutaz J., Thevenin D., Limbourg Q., Bouillon L., Vanderdonckt J., A unifying reference framework for multi-target user interfaces, *Interacting With Computers*, Vol. 15/3, pp 289-308, 2003.

Card S.K., Moran T.P., Newell A., *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1983.

Chein M., Mugnier M.L., Conceptual Graphs : Fundamental notions*, Revue d'intelligence artificielle*, 6, 4, 1992, 365-406.

Coutaz J., PAC, an Onject Oriented Model for Dialog Design, *In Interact'87*, 1987, pp 431-436.

Coutaz J., Interfaces Homme-Machine : un regard critique, *TSI*, Vol. 10, no 1, 1991, pp. 53-64.

Crease M., *A Toolkit Of Resource-Sensitive, Multimodal Widgets*. PhD Thesis, Department of Computing Science, University of Glasgow, Dec 2001.

Dey A.K., Abowd G.D., Towards a Better Understanding of Context and Context-Awareness, *Proceedings of the CHI 2000 Workshop on The What, Who, Where, When, and How of Context-Awareness*, The Hague, Netherlands, April 1-6, 2000.

Duke D.J., Harrisson M.D., Abstract Interaction Objects, *Eurographics 93*, Spain, 1993.

Faconti G., Paterno F., An approach to the formal specification of the components of an interaction. In C. Vandoni and D. Duce, editors, *Eurographics 90*, pp 481-494, North-Holland, 1990.

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.

Green T.R.G., The cognitive dimension of viscosity: A sticky problem for HCI. In *INTERACT'90* [756], pp. 79-86.

IFIP Book, *Design Principles for Interactive Software*, Gram C. and Cockton G. (eds), Chapman & Hall, 1996.

Islam N., Fayad M., Toward Ubiquitous Acceptance of Ubiquitous Computing, *Communications of the ACM*, February 2003, Vol. 46, N° 2, pp 89-92.

Johnson P., Wilson S., Markopoulos P., Pycok J., ADEPT - Advanced Design Environment for Prototyping with Task Models, *Proceedings of InterCHI'93*, Amsterdam, The Netherlands, 24-29 April 1993, pp 56-57.

Lee J., Su V., Ren S., Ishii H., HandSCAPE: A Vectorizing Tape Measure for On-Site Measuring Applications, *Videos of CHI'2000*, April 2000.

Luyten K., Coninx K., An XML-based runtime user interface description language for mobile computing devices, Karin, *DSV-IS'2001*, Glasgow (Scotland), 2001.

Lyytinen K., Yoo Y., Issues and Challenges in Ubiquitous Computing, *In Communications of the ACM*, December 2002, Vol. 45, N° 12, pp 63-65.

Markopoulos P., Interactors: formal architectural models of user interface software. In Kent, A., and Williams, J.G., (Eds.) *Encyclopedia of Microcomputers*, Volume 27 (Supplement 6), Marcel Dekker, New York, 2001, pp 203-235.

Märtin C., Software Life Cycle Automation for Interactive Applications : The AME Design Environment, *CADUI'96*, J. Vanderdonckt (eds), 1996, pp 57-73.

McCall J., Factors in Software Quality, General Electric Ed., 1977.

Milgram P., Kishino F., A Taxonomy of Mixed Reality Visual Displays. In *IEICE Transactions on Information Systems*, E77-D(12), (1994), p.1321-1329.

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.

Myers B., Stiel H., Gargiulo R., Collaboration using multiple PDA's connected to a PC. *In Proc. of the ACM Conference on CSCW (CSCW'98)*, 1998, pp. 285-294.

Nielsen J., *Usability Engineering*, Academic Press Professional, 1993, p 362.

Normand V., *Le modèle SIROCO : de la spécification conceptuelle des interfaces utilisateur à leur réalisation*, Thèse de l'Université Joseph Fourier-Grenoble I, Spécialité Informatique, Avril 1992, 258 pages.

Paternò F., *Model-based Design and Evaluation of Interactive Applications*. Springer Verlag, Berlin, 1999.

Puerta A., The MECANO Project : Comprehensive and Integrated Support for Model-Based Interface Development, *CADUI'96*, J. Vanderdonckt (eds), 1996, pp 19-35

Rauterberg M., Fjeld M., Krueger H., Bichsel M., Leonhard U., Meier M., BUILT-IT: A Planning Tool for Consruction and Design. *In Proc. of the ACM Conference in Human Factors in Computing Systems (CHI'98) Conference Companion*, 1998, pp. 177-178.

Rekimoto J., Nagao K., The World through the Computer: Computer Augmented Interaction with Real World Environments, *Actes du symposium UIST'95*, 1995.

Rekimoto J., Pick and Drop: A Direct Manipulation Technique for Multiple Computer Environments, *Proceedings of UIST97*, ACM Press, 1997, pp. 31-39.

Salber D., Dey A.D., Abowd G.D., The Context Toolkit: Aiding the Development of Context-Enabled Applications, *In the Proceedings of the 1999 Conference on Human Factors in Computing Systems (CHI '99),* Pittsburgh, PA, May 15-20, 1999, pp. 434-441.

Schilit B.N., Adams N.I., Want R., Context-Aware Computing Applications, *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'94)*, IEEE Press, Santa Cruz CA, December 1994, pp 85-90.

Schmidt A., Implicit human-computer interaction through context. *2th Workshop on Human Computer Interaction with Mobile Devices*. Edinburgh, Scotland, 31 August 1999.

Sowa J.F, *Conceptual Structures – Information Processing in Mind and Machine*, Addison Wesley, 1984.

Streitz N., Geißler J., Holmer T., Konomi S., Müller-Tomfelde C., Reischl W., Rexroth P., Seitz P., Steinmetz R. i-LAND: An interactive landscape for creativity and innovation. *In Proc. of the ACM Conf. On Human Factors in Computing Systems (CHI'99)*, Pittsburgh, May 15-20, 1999, pp. 120-127.

Sukaviriya P., Kovacevic S., Foley J.D., Meyers B.A., Olsen D.R., Schneider-Hufschmidt M., Model-Based Interfaces, What are They and Why Should We Care ?, *UIST'94*, November 2-4, 1994, pp 133-135.

Szekely P., Sukaviriya P., Castells P., Muthukumarasamy J., Salcher E., Declarative interface models for user interface construction tools: the MASTERMIND approach, *Engineering for Human-Computer Interaction*, Bass, L.J. & Unger, C. Eds, Chapman & Hall, 1995, pp 120-150.

Szekely P., Retrospective and Challenges for Model-Based Interface Development. *In Computer-Aided Design of User Interfaces, Proceedings of CADUI'96*, J. Vanderdonckt (eds), Presses Universitaires de Namur, pp xxi-xliv.

Thevenin D., Coutaz J., Plasticity of User Interfaces: Framework and Research Agenda. *In Proceedings of INTERACT'99*, 1999, pp. 110-117.

Thevenin D., *Adaptation en Interaction Homme-Machine : le cas de la Plasticité*. Thèse de l'Université Joseph-Fourier Grenoble I, Spécialité Informatique, 2001.

Vanderdonckt J., *Conception assistée de la présentation d'une interface homme-machine ergonomique pour une application de gestion hautement interactive*. Thèse des Facultés Universitaires Notre-Dame de la Paix, Spécialité Informatique, Juillet 1997.

Vanderdonckt J., Bouillon L., Souchon N., Flexible Reverse Engineering of Web Pages with VAQUITA. *Proceedings of IEEE 8th Working Conference on Reverse Engineering WCRE'2001* (Stuttgart, 2-5 October 2001). IEEE Press, Los Alamitos, 2001, pp. 241–248.

Vernier F., Lachenal C., Nigay L., Coutaz J., Interface Augmentée Par Effet Miroir, *IHM'99* (Conférence AFIHM sur l'Interaction Homme-Machine, 22-26 Novembre 1999 Montpellier, France) . pp. 158-165.

Weiser M., The computer for the 21st century, *Scientific American*, 265(3), 1991, pp. 94-104.

Yan H., Selker T., Context-Aware Office Assistant, *Proceedings of the 2000 International Conference on Intelligent User Interfaces (IUI'2000)*, H Lieberman (eds), ACM Press, New-Orleans LA USA, pp 276-279.