Composition d'Interfaces Homme-Machine en contexte : approche par planification automatique

Yoann Gabillon, Gaëlle Calvary, Humbert Fiorino

Composition d'Interfaces Homme-Machine en contexte : approche par planification automatique

Yoann Gabillon — Gaëlle Calvary — Humbert Fiorino

Université Joseph Fourier, Grenoble INP, CNRS UMR 5217
Laboratoire d'Informatique de Grenoble
385, avenue de la Bibliothèque
38400 Saint-Martin-d'Hères

{yoann.gabillon, gaelle.calvary, humbert.fiorino}@imag.fr

RÉSUMÉ. *En informatique ambiante, les objectifs de l'utilisateur peuvent émerger opportunément. Il devient, dès lors, nécessaire de savoir générer à la volée des systèmes interactifs. Un système interactif est composé d'un noyau fonctionnel et d'une Interface Homme-Machine (IHM). Cet article traite de la composition d'IHM pour un objectif utilisateur et un contexte d'usage (utilisateur, plate-forme, environnement) donnés. Un état de l'art sur la composition d'IHM positionne notre travail et en montre la complémentarité par rapport aux travaux existants. Le principe est de composer un modèle de tâches puis de composer l'IHM concrète à l'aide d'une boîte à outils d'interacteurs définis au niveau tâches. La composition du modèle de tâches se fait par planification automatique. L'étude montre que les planificateurs existants ne répondent pas au problème. Aussi, un planificateur a été spécifiquement développé pour l'IHM. Son utilisation est illustrée dans un prototype Compose. Le travail est original à deux titres : d'une part, son approche « Composition de modèles de tâches » est une extension de la littérature ; d'autre part, la composition d'IHM est un nouveau cadre applicatif pour les algorithmes de planification.*

ABSTRACT. *In ubiquitous computing, user needs may opportunistically emerge along the variation of the context of use. Thus, there is a need for dynamically composing interactive systems. An interactive system is made of a functional core and a User Interface (UI). This paper deals with the composition of UIs to support opportunistic user needs in a given context of use (user, platform, environment). A state of the art about UI composition shows the originality of our work: the composition of a task model. The composition of the concrete UI is then delegated to a toolkit of interactors defined at the task level. The composition of the task model is done by automated planning. The work shows that current planners do not fulfill Human Computer Interaction (HCI) requirements. Therefore, a specific planner has been developed to compose UIs. This planner is used in Compose, our proof of concept. The work is original in two points: first, by the high level of abstraction the composition is performed; secondly, by the use of automated planning in HCI.*

MOTS-CLÉS : *Interface Homme-Machine, Composition, Contexte d'usage, Planification automatique.*

KEYWORDS: *User Interface, Composition, Context of Use, Automated planning.*

1. Introduction

En rupture avec l'informatique traditionnelle souvent cantonnée à un ordinateur muni d'un clavier et d'une souris, l'informatique ambiante (Weiser, 1991) s'ouvre sur une diversité de *contextes d'usage* : diversité des *utilisateurs* (âge, compétences, préférences, etc.) ; diversité des *plates-formes* d'interaction (capacités de calcul, de communication et dispositifs d'interaction) ; diversité des *environnements physiques* (niveaux lumineux, sonores, etc.) et *sociaux* (milieu professionnel, privé, etc.) dans lesquels évolue l'utilisateur. En effet, en informatique ambiante, l'utilisateur n'est plus seulement un informaticien : tout utilisateur est concerné. Ses plates-formes d'interaction ne sont plus les traditionnels ordinateurs « boîte noire » : tout objet du monde physique peut jouer un rôle dans l'interaction. L'environnement n'est plus seulement un bureau : l'utilisateur peut interagir en tout lieu (dans la rue, le tramway, sa maison, etc.).

A cette diversité, s'ajoutent une variabilité des contextes d'usage (par exemple, passage du milieu professionnel au domicile) et une émergence des besoins de l'utilisateur. En effet, l'utilisateur découvre de façon opportuniste des ressources d'interaction et des systèmes interactifs offerts par son environnement (bornes wifi, murs interactifs, ordinateurs, etc.). Dès lors, il devient nécessaire de composer des systèmes interactifs compatibles du contexte d'usage courant.

Un système interactif est un système constitué d'un *Noyau Fonctionnel* (NF) et d'une *IHM* (Figure 1). Le *NF* regroupe l'ensemble des traitements indépendamment de toute représentation à l'utilisateur. L'*IHM* fait les choix de présentation compte tenu du contexte d'usage et des *propriétés ergonomiques* à satisfaire. Par *propriétés ergonomiques*, on entend des propriétés relatives à l'acceptabilité du système, c'est-à-dire à son adéquation aux capacités de l'utilisateur (Nielsen, 1993).

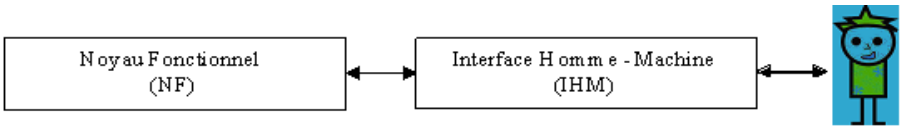


Figure 1. *Décomposition fonctionnelle minimale d'un système interactif*

Composer un système interactif, c'est composer un NF, une IHM et assurer un connecteur entre ces deux fonctions. La composition du NF relève de la communauté Génie Logiciel (GL) ; la composition de l'IHM de la communauté IHM. Plusieurs approches se distinguent pour composer le NF : tissage de comportements par Programmation par Aspect (AOP) (Kiczales *et al.*, 1997), composition de modèles (Perroin *et al.*, 2009, Acher *et al.*, 2010), composition de composants (Balme, 2008), adaptation des transformations (Favre *et al.*, 2004) en

Ingénierie dirigée par les modèles (IDM) et composition par rapport aux ressources de composants web orientés services (Kuter *et al.,* 2003, Traverso *et al.,* 2004, Klusch *et al*., 2006). La composition de services web traite de la variabilité de la tâche sous l'angle du NF : les composants web sont fonctionnellement décrits à l'aide d'un langage pour les composer comme BPEL4WS (Andrews *et al.,* 2003) ou OWL-S (Martin *et al.,* 2004) en omettant l'IHM et son utilisabilité.

En ingénierie de l'interaction Homme-Machine, la variabilité du contexte d'usage est étudiée pour des contextes d'usage prévus à la conception (Thevenin *et al.,* 1999). La *tâche de l'utilisateur* est toujours supposée connue à la conception et constante à l'exécution : « envoyer un message », « voir une vidéo en streaming », etc. Les variations entre contextes d'usage sont aussi répertoriées permettant en conséquence la conception d'Interfaces Homme-Machine (IHM) sur mesure. Ainsi, si la composition de plates-formes est techniquement possible (Hinckley, 2003, Rekimoto *et al.*, 2001), la composition en contexte d'IHM est un problème ouvert. Cet article traite de la composition d'IHM en contexte à partir de tâches utilisateurs variables.

Un cas d'étude est tout d'abord présenté pour illustrer le sujet (section 2). Un état de l'art est ensuite dressé sur la composition d'IHM (section 3). Cet état de l'art permet de situer notre contribution : nous proposons un système de composition d'IHM en contexte, fondé sur la planification automatique (section 4). Un prototype intégrant ce planificateur est présenté (section 5). Les défis restent nombreux et montrent l'intérêt, pour d'autres communautés, de considérer l'IHM comme cadre applicatif (section 6).

2. Cas d'étude

Le cas d'étude est le fruit d'une étude sociologique (Gabillon *et al*, 2009) menée avec vingt-six personnes « grand public » de profils sociodémographiques différents en termes de genre, d'âge, de profession et de lieu d'habitation. Le but de l'étude était double : d'une part, mesurer l'intérêt du grand public vis-à-vis d'un système de composition d'IHM en contexte ; d'autre part, faire émerger des scénarios pertinents. Le cas d'étude suivant en est un.

Victor est un jeune étudiant qui vit à New York. Il est en vacances dans un hôtel à Philadelphie. Il est 23 heures. Soudain, il ne se sent pas bien et ressent le besoin d'une assistance médicale. Il est déjà tard. Il ne connaît personne dans cette ville. Il lance son assistant personnel *Compose* et spécifie son objectif en langage naturel : « Je veux voir un médecin » (Figure 2).

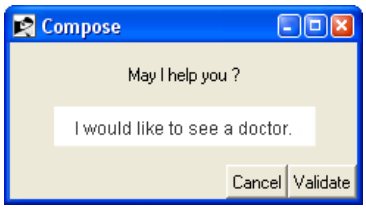


Figure 2. *L'IHM de Compose pour spécifier l'objectif de l'utilisateur.*

Compose calcule à la volée des solutions médicales adaptées au contexte de l'utilisateur : Victor peut soit rentrer chez lui à New York pour se faire soigner, soit rester sur place (Figure 3a). Victor préfère rester à Philadelphie. *Compose* lui propose alors quatre possibilités (Figure 3b) : consulter le médecin de garde, se rendre à l'hôpital le plus proche, appeler SOS médecin ou les pompiers. Victor choisit de consulter le médecin de garde.

(a) Deux possibilités de lieu pour l'assistance médicale

(b) Quatre solutions médicales

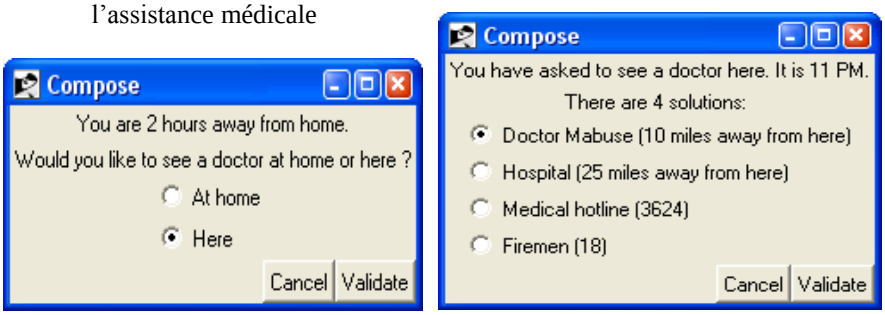


Figure 3. *Deux IHM composées en contexte par Compose pour affiner le besoin de l'utilisateur « Get medical assistance ».*

Trois variantes sont ensuite possibles selon que Victor utilise un mur numérique (Version 1), un Smartphone (Version 2) ou un mur numérique couplé à un Smartphone sans GPS (Version 3).

Version 1 : Mur numérique

Compose calcule à la volée une IHM (Figure 4) pour le mur numérique. Elle permet à Victor d'appeler le médecin de garde. Le numéro est précomposé (en haut de la fenêtre). L'IHM localise également le cabinet médical (sur la carte au milieu de la fenêtre) et fournit des informations complémentaires : ici, la pharmacie de garde et la station service la plus proche (en bas de la fenêtre).

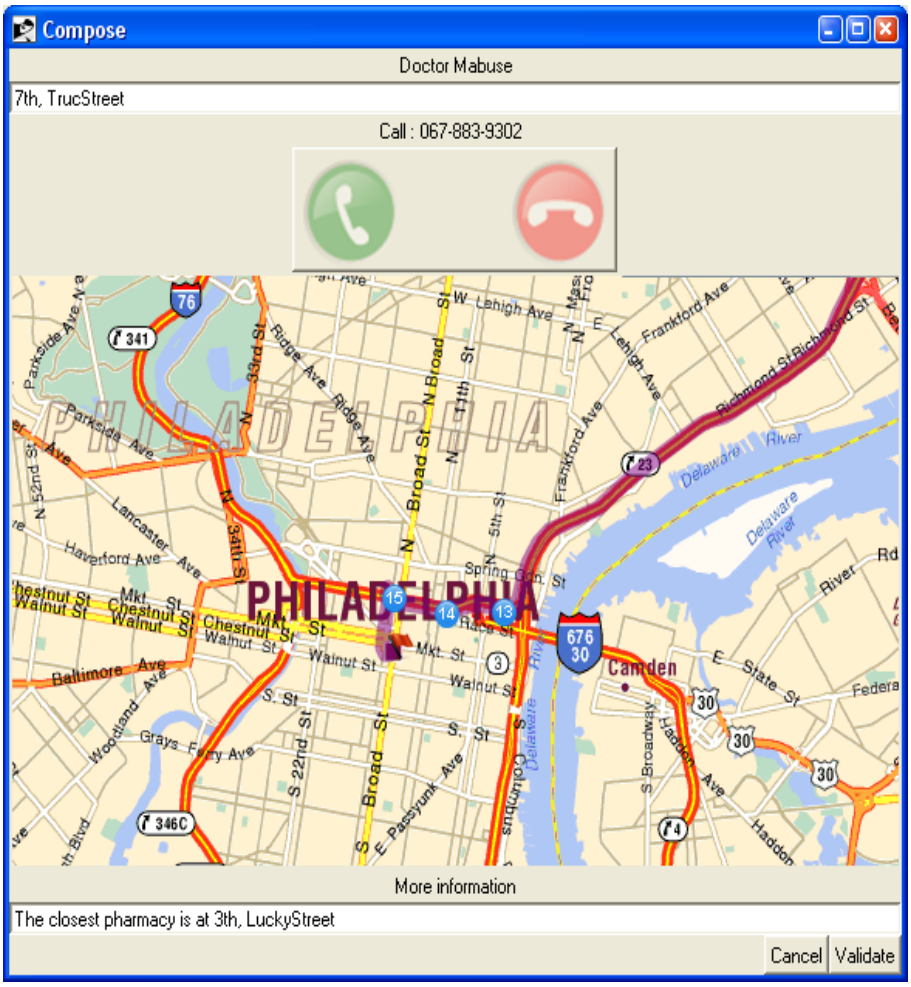


Figure 4. *IHM composée en contexte par Compose pour répondre à l'objectif de l'utilisateur « Get medical assistance ».*

Version 2 : Smartphone

Compose calcule à la volée une IHM (Figure 5) pour le Smartphone. Elle permet à Victor d'appeler le médecin de garde. L'IHM guide Victor vers le cabinet médical en utilisant une carte. Faute de place, l'IHM est organisée en onglets et les informations complémentaires sont omises.

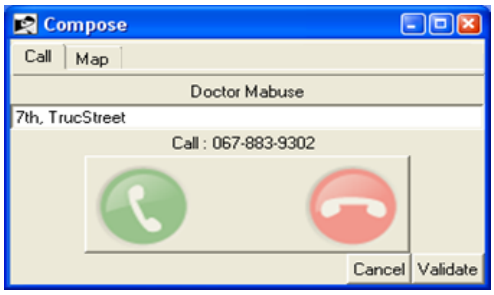


Figure 5. *IHM composée pour un Smartphone.*

Version 3 : Couplage entre un mur numérique et un smartphone sans GPS

Compose calcule à la volée une IHM (Figure 6) pour le mur numérique couplé à Smartphone sans GPS. Elle permet à Victor d'appeler le médecin de garde de son smartphone. L'IHM localise le cabinet médical et fournit des informations complémentaires sur le mur numérique.

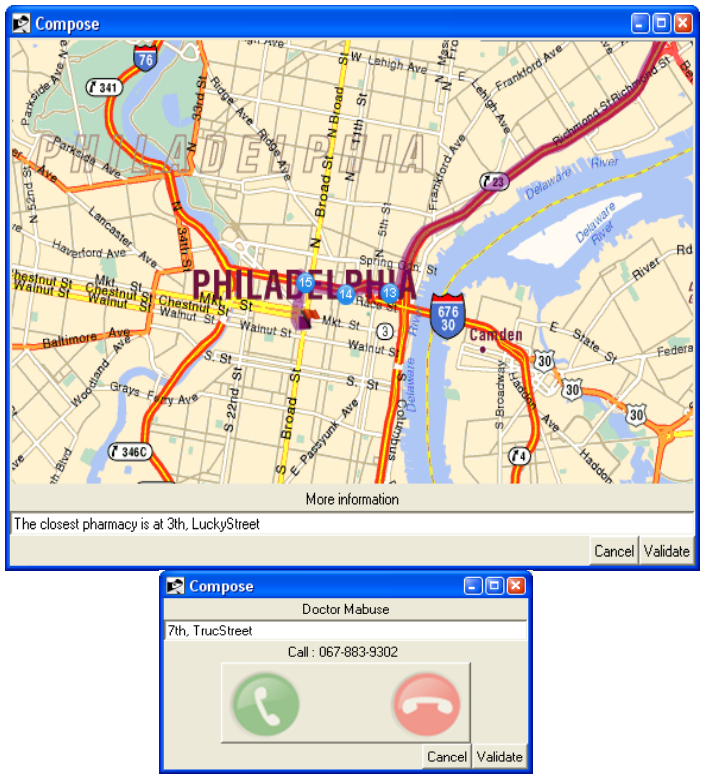


Figure 6. *IHM composées pour un mur numérique couplé à un Smartphone sans GPS.*

D'autres variantes sont bien sûr imaginables selon les trois dimensions du contexte d'usage (utilisateur, plate-forme, environnement).

3. Etat de l'art pour la composition dynamique d'IHM

Cette section parcourt les travaux utiles à la composition dynamique d'IHM. Elle recense les travaux en composition d'IHM puis les acquis en ingénierie d'IHM : processus de conception d'IHM, génération d'IHM à la conception et adaptation des IHM à l'exécution.

3.1. Composition d'IHM

La décomposition fonctionnelle NF-IHM (Figure 1) permet, à gros grain, de classer les travaux de la littérature. AMUSINg (Pinna-Dery *et al.,* 2003) compose, par exemple, des IHM à partir du NF. Il s'appuie sur des modèles de composants préexistants pour inférer une tâche utilisateur et assembler les composants correspondants. Le contexte d'usage est implicite. La composition est réalisée à la conception. Les Mashups (Fujima *et al.,* 2004, Lin *et al.,* 2008) permettent une composition dirigée par les données. La composition est précalculée à la conception et placée sous le contrôle de l'utilisateur final pour la sélection des données et des services à la volée. ComposiXML (Lepreux *et al.,* 2006) permet la composition d'IHM par le concepteur. Les IHM sont graphiques. Le concepteur peut en faire l'union, l'union sans doublon, l'intersection, etc. Le contexte d'usage est là encore implicite. Les IHM sont graphiques et centralisées sur une seule plate-forme.

D'autres travaux s'intéressent au contexte d'usage et à la distribution des IHM sur un ensemble de plates-formes (Stuerzlinger *et al.,* 2006, Nichols *et al.,* 2008). La (dé/re)composition et le tissage (tailoring, Wulf *et al.,* 2008) d'IHM permettent à l'utilisateur de contrôler manuellement la distribution de son IHM sur un ensemble de fenêtres et de plates-formes. L'utilisateur sélectionne manuellement les parties de l'IHM à déplacer. La sélection peut être faite par le système (Paternò *et al.,* 2008) pour s'adapter aux modifications du contexte d'usage.

3.2. Processus de conception d'IHM

Le processus de conception d'IHM s'organise autour de quatre niveaux d'abstraction (Calvary *et al.,* 2003, figure 7). Le *modèle de tâches* (ou *arbre de tâches*, Paternò *et al.,* 1997) décrit la tâche de l'utilisateur en termes d'objectif et de procédure. La procédure décompose récursivement l'objectif en sous-objectifs. Les sous-objectifs sont reliés entre eux par des relations logiques ou temporelles. Par exemple, dans notre cas d'étude (section 2), la tâche « Get medical assistance » se décompose en trois sous-tâches séquentielles : « Choose the city », « Choose the doctor » puis « Contact the doctor ». Chaque (sous-)tâche est liée aux concepts du domaine qu'elle manipule (par exemple, le concept de pharmacie).

Du modèle de tâches peut être déduite une structuration de l'IHM en zones, appelées « espaces de travail ». Les espaces groupent les entités nécessaires à l'utilisateur pour accomplir ses sous-objectifs. La description obtenue est appelée *Interface Utilisateur Abstraite* (IUA). Par exemple (section 2), la tâche « Get medical assistance » structure ses sous-tâches « Choose the city », « Choose the doctor » puis « Contact the doctor » en trois espaces de travail différents. Les espaces de travail rendent observables et éventuellement modifiables les concepts du domaine manipulés dans les tâches.

Dans une approche descendante, les espaces de travail et leur contenu sont ensuite concrétisés en interacteurs (fenêtres, champs texte, images, boutons radio, cases à cocher, son, etc.). Une telle IHM est appelée *Interface Utilisateur Concrète* (IUC) : c'est une description d'IHM. Elle n'est pas encore implémentée dans un langage de programmation. Par exemple (section 2), les copies d'écran sont des IHM concrètes. Elles mobilisent des interacteurs fenêtre, textes (« You are 2 hours away from home, Would you like to see a doctor ? »), boutons radio, boutons (« Validate » et « Cancel »), etc.

Lorsque l'IUC est ensuite implémentée dans un langage de programmation, elle donne lieu à une *Interface Utilisateur Finale* (IUF).

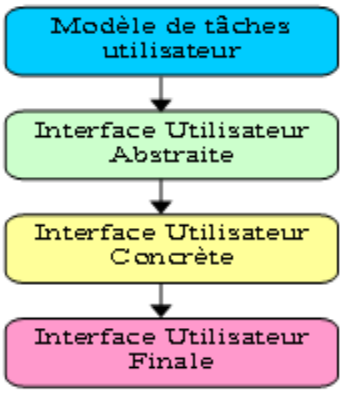


Figure 7. *Les quatre niveaux d'abstraction en IHM, ici enchaînés selon un processus de concrétisation.*

De nombreux travaux ont porté sur la génération à la conception des IUA, IUC et IUF à partir d'un modèle de tâches. Ils font l'objet de la section suivante.

3.3. Génération d'IHM

Depuis les années 90 (Mori *et al.*, 2002), de nombreux travaux ont porté sur la génération d'IHM à partir d'un modèle de tâches donné. Les transformations de modèles étaient « câblées en dur » dans le code des générateurs. Elles n'étaient ni

modifiables, ni extensibles (Johnson *et al.*, 1993), aboutissant en conséquence à des IHM de faible qualité, généralement graphiques, mono-écran, bien loin des IHM modernes (dites « post-WIMP »), ne laissant aucune place à la créativité.

Un saut est franchi dans les années 2004 avec le développement de l'Ingénierie Dirigée par les Modèles (IDM) : les transformations de modèles deviennent des modèles. Dès lors, les transformations sont capitalisables et traçables au fil des évolutions, à la conception comme à l'exécution. De nouvelles possibilités s'ouvrent dès lors aux modèles du processus de conception (Myers *et al.*, 2000). La section suivante explore leur intérêt pour l'adaptation des IHM à l'exécution.

3.4. Plasticité des IHM

En IHM, la *plasticité* (Calvary, 2007) dénote la capacité d'adaptation d'une IHM à son contexte d'usage dans le respect de propriétés centrées utilisateur. Une IHM plastique a, par exemple, la capacité de migrer vers une autre plate-forme lorsque sa batterie faiblit. Elle saura se remodeler pour s'accommoder des caractéristiques de cette nouvelle plate-forme de façon à ce que l'utilisateur puisse poursuive sa tâche avec confort, efficacité et sécurité.

L'ensemble des évolutions du contexte d'usage n'étant pas prévisible à la conception, (Sottet *et al.*, 2007) propose de conserver à l'exécution les modèles de conception (section 3.2). Les modèles couvrent le NF, l'IHM et le contexte d'usage. Ils sont reliés entre eux et forment un graphe (Figure 8). Les relations (*mappings*) entre modèles correspondent aux choix de conception. Ils sont motivés par les propriétés centrées utilisateur à satisfaire. Ils couvrent :

- les mappings intra-IHM entre les niveaux d'abstraction du processus de conception : *Concept-Task* énumère les concepts manipulés dans chaque tâche ; *Task-AUI* assigne des espaces de travail aux différentes tâches ; *AUI-CUI* et *Task-CUI* associent des interacteurs aux espaces de travail et à leur contenu ;
- les mappings intra-contexte : *U-Env* et *Ptf-Env* modélisent respectivement les positions de l'utilisateur (U) et de la plate-forme (*Ptf*) d'interaction dans l'environnement *(Env)* ; *U-Input* et *U-Output* explicitent les dispositifs d'interaction que l'utilisateur (U) exploite en entrée (input) et en sortie (output) ;
- les mappings inter NF-IHM : *Concept-FC* et *Task-FC* établissent les liens entre les données et fonctions du NF (en anglais, FC pour Functional Core) d'une part et les concepts et tâches de l'IHM d'autre part ;
- les mappings inter Contexte-IHM modélisent le déploiement de l'IHM sur les ressources d'interaction (*I-Ptf*) en entrée et en sortie (*I-Input* et *I-Output*).

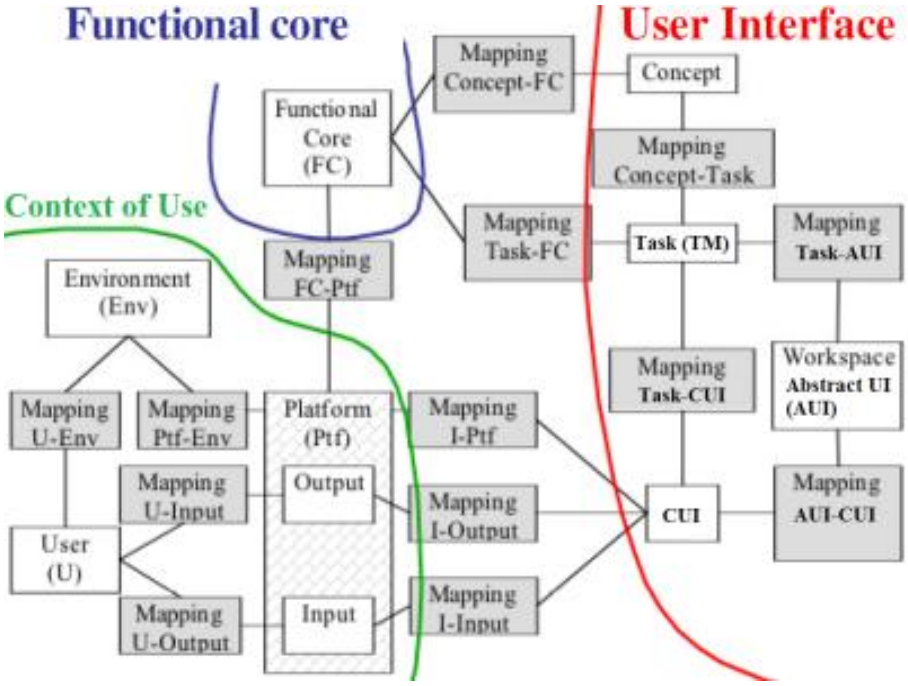


Figure 8. *Graphe de modèles d'un système interactif (Sottet et al., 2007).*

(Demeure *et.al.*, 2004) propose une boîte à outils d'interacteurs plastiques. Un interacteur plastique est un interacteur défini au niveau tâches (par exemple, l'interacteur « choisir ») et comporte plusieurs présentations (par exemple, les boutons radio, un menu linéaire, circulaire, etc.) lui offrant, en conséquence, un potentiel d'adaptation. Ces interacteurs, appelés COMETs, embarquent en fait un micro graphe de modèles tel que défini dans la figure 8.

3.5. Synthèse et positionnement

La figure 9 propose une synthèse de l'état de l'art et positionne *Compose* par rapport à l'existant. Nous retenons un accent porté sur la méthodologie de conception avec, en particulier, l'identification de niveaux d'abstraction en conception d'IHM, mais une exploitation limitée en composition d'IHM. Certains font l'impasse du modèle de tâches comme Amusing qui déduit l'IHM du noyau fonctionnel et ComposiXML qui compose deux IHM au niveau IUC. D'autres le supposent connu à la conception comme les travaux en génération d'IHM et en plasticité. Dans les Mashups, l'utilisateur choisit lui-même les composants et donc, sans le savoir, les IUC qu'il souhaite composer dans les Mashups.

Notre originalité est de composer le modèle de tâches dynamiquement pour permettre l'opportunisme promis en informatique ambiante. Aussi, de façon complémentaire, cet article se concentre sur la composition du modèle de tâches. Nous explorons la planification automatique pour la similitude dans la définition des problèmes.

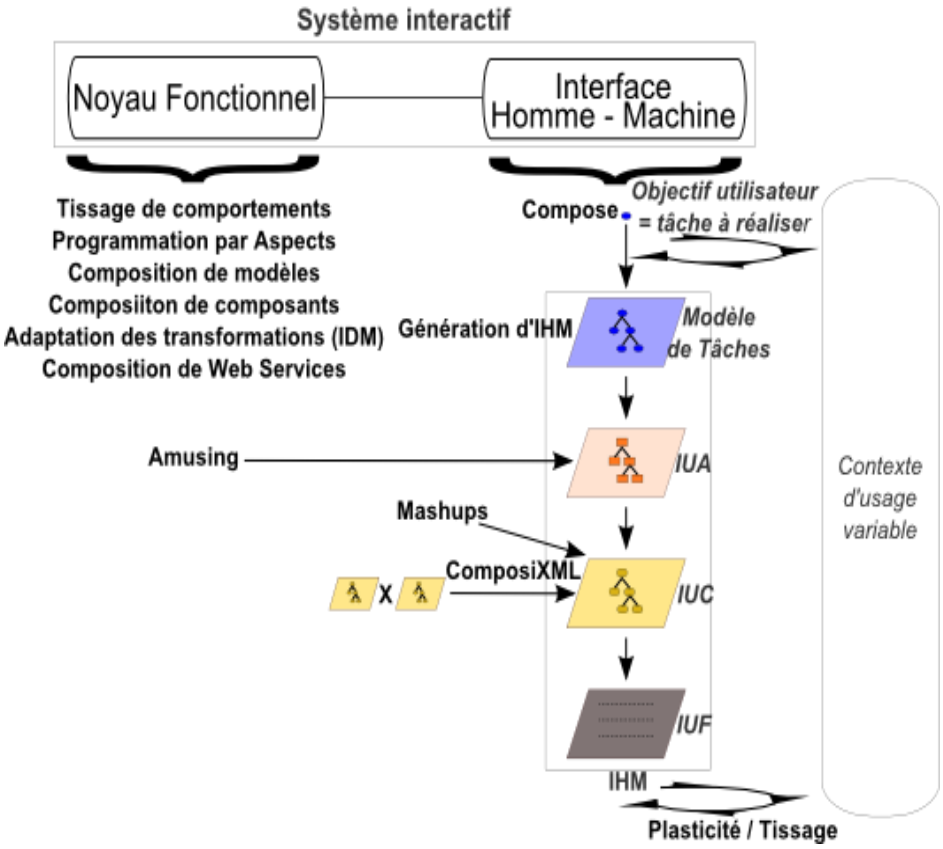


Figure 9. *Synthèse de l'état de l'art et positionnement de Compose.*

4. Approche par planification

Notre objectif est de construire, par des algorithmes de planification (Ghallab *et al.*, 2004), une IHM dont la fonction est de faire passer l'utilisateur de sa situation courante à une situation dans laquelle ses besoins sont satisfaits. De façon plus formelle, la situation courante est *l'état initial* s0 d'un espace de recherche. On représente cet état par un ensemble de propositions logiques :

$$s_0 = \begin{cases} \text{has (Victor, Smartphone)} \\ \text{internet (Smartphone)} \\ \text{at (Victor, Philadelphia)} \\ \dots \end{cases}$$

De la même façon, les besoins de l'utilisateur, c'est-à-dire son *but,* sont représentés par un ensemble de propositions *g*. On appelle *états but* s_g les états contenant *g* :

$$s_g = \begin{cases} \text{found(Victor, medical\_assistance)} \\ \dots \end{cases}$$

L'espace de recherche est un graphe de changements d'états dont les nœuds sont des états et les arcs sont les actions que l'utilisateur peut réaliser grâce à l'IHM (Figure 10).

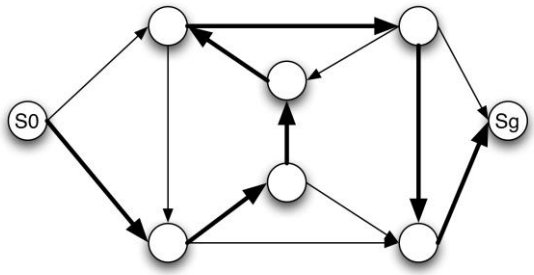


Figure 10. *Graphe de changements d'états*.

Les actions *a* sont définies par des triplets (*precond(a), add(a), del(a)*) où *precond (a)* est un ensemble de propositions logiques représentant des conditions de déclenchement de l'action : une action est *applicable* si et seulement si l'état considéré contient *precond (a). add(a)* et *del(a*) sont respectivement les effets ajoutés et retirés de l'état courant. Si une action *a* est applicable, on calcule l'état suivant par :

$$\gamma(s_i, a) = (s_i - \text{del}(a)) \cup \text{add}(a) = s_{i+1}$$

Notre problème est donc de construire une IHM telle que, quelles que soient les actions de l'utilisateur, cette IHM le mène de l'état initial à l'état but (voir la figure 10). Soit π une liste de longueur *k* de ces actions et *s* l'état courant ; l'état obtenu est défini par :

$$\gamma(s, \pi) = \begin{cases} s, & \text{si } k = 0 \\ \gamma(\gamma(s, a_1), [a_2, \cdots, a_k]), & \text{si } k > 0 \text{ et } a_1 \text{ est applicable à } s \end{cases}$$

En pratique, le fonctionnement du système de composition est présenté dans la figure 11. L'utilisateur exprime ses besoins. Ils sont traduits en un ensemble de propositions logiques *g*. Le système perçoit le contexte d'usage et le représente sous la forme d'un ensemble de propositions logiques *s0*. L'algorithme de planification connaît les actions que l'utilisateur peut réaliser via l'IHM. Ces actions sont exprimées « en intention » sous la forme d'*opérateurs* et de *méthodes*.

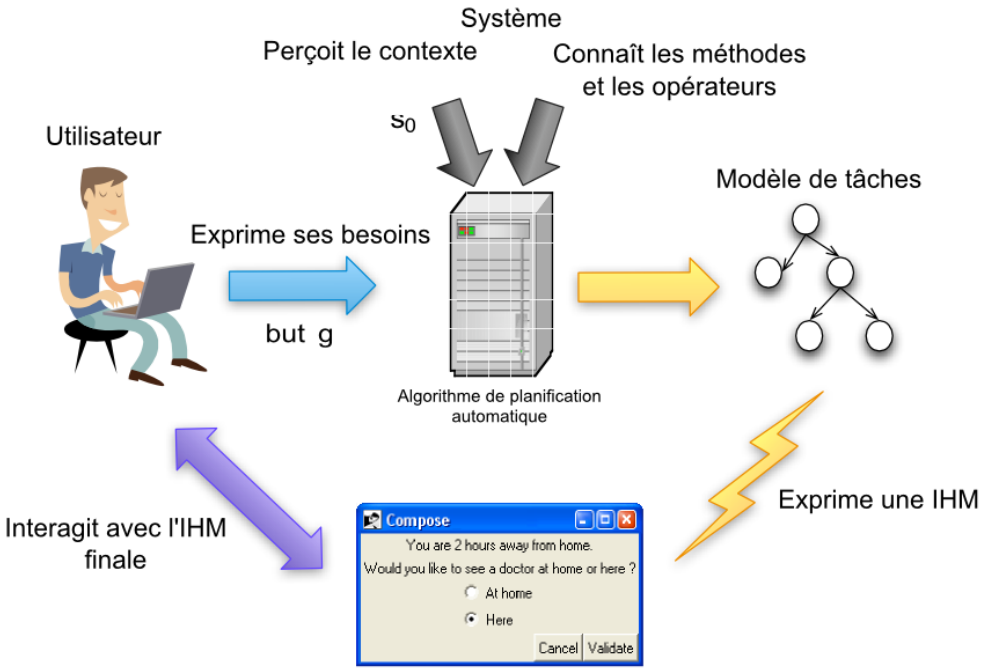


Figure 11. *Fonctionnement du système de composition d'IHM.*

Un opérateur *o* est de la forme *(precond(o), add(o), del(o))*. Il est défini par ses préconditions, ses ajouts et ses retraits qui sont des ensembles de prédicats logiques du premier ordre. Par exemple, dans l'opérateur « *Call_the_office(?u, ?p)* », les préconditions contiennent les prédicats « *has(?u, ?p)* » et « *internet(?p)* » ; l'utilisateur *?u* et la plate-forme *?p* sont des *variables* liées à des *domaines* c'est-à-dire à des ensembles de *constantes*. Ainsi, l'opérateur « *Call_the_office(?u, ?p)* » représente les actions « *Call_the_office(Victor, Smartphone)* », « *Call_the_office (Victor, Wall)* » etc.

Une méthode *m* est de la forme *(type(m), precond(m), subAct(m))* où *precond(m)* définit ses préconditions et *subAct (m)* décompose la méthode en opérateurs ou méthodes à accomplir pour réaliser *m*. Le type de la décomposition exprimé dans *type(m)* est soit une *séquence* soit un *parallélisme (concurrence).* Dans une séquence, les méthodes ou opérateurs de *subAct(m)* sont totalement ordonnés. Dans un parallélisme, les éléments de *subAct(m)* peuvent être réalisés dans n'importe quel ordre. Par exemple, la méthode « *Get_medical_assistance* » correspond à la séquence de « *Choose_the_city* », « *Choose_the_doctor* » puis « *Contact_the_doctor* ». Au contraire, la méthode « *Contact_the_doctor* » correspond au parallélisme de « *Call_the_office* » et « *Find_route_information* ».

A partir de l'état initial s0, du but g et d'un ensemble de méthodes et opérateurs, l'algorithme de planification calcule un modèle de tâches qui permet à l'utilisateur d'accomplir son objectif. Ce modèle de tâches est un arbre. Chaque noeud correspond à une IHM existante. Par exemple, la tâche « *Contact_the_doctor* » correspond à une fenêtre. Une fois calculé, l'arbre de tâches permet de connaître les IHM que la fenêtre contiendra. Par exemple, dans le cas d'étude, elle contient les IHM correspondant aux sous-tâches : « *Call_the_office* » et « *Find_route_information* ».

L'arbre de tâches contient deux types de nœuds : les nœuds internes sont des méthodes totalement instanciées (toutes les variables sont liées à des constantes) de type séquence (« > ») ou parallèle (« II ») ; les feuilles de l'arbre sont des actions c'est-à-dire des opérateurs totalement instanciés (Figure 12).

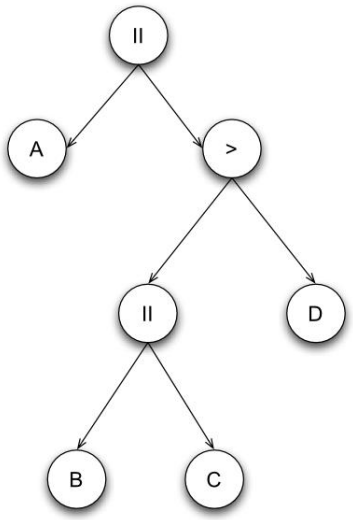


Figure 12. *Arbre de tâches obtenu par le planificateur.*

On appelle *projection* $\wp$ de l'arbre de tâches l'ensemble partiellement ordonné d'actions qui en découle (Figure 13). La relation d'ordre « *C* > *D* » exprime que nécessairement l'action *C précède* l'action *D*. Dans la figure 13, *0* et ∞ sont des actions formelles représentant le début et la fin des interactions.

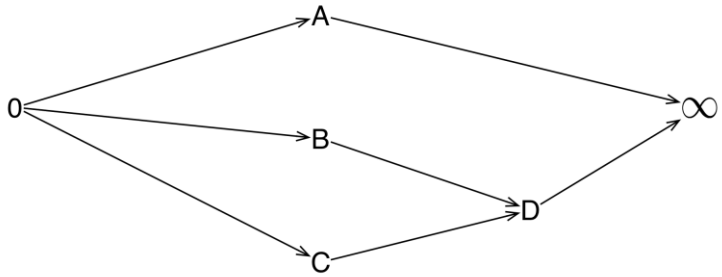
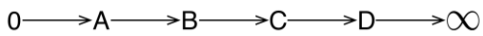


Figure 13. *Projection* $\wp$ *de l'arbre de tâches.*

On appelle *linéarisation* de $\wp$ un ensemble composé des mêmes actions totalement ordonnées par la relation de précédence. Par exemple :



Soit L($\wp$) l'ensemble des linéarisations de $\wp$. Nous sommes à présent en mesure d'exprimer formellement notre problématique.

Soit P = (*s0*, *g*, *A*) un problème de planification où *s0* est l'état initial, *g* est le but et *A* est l'ensemble des actions possibles (en pratique exprimées sous la forme de méthodes et d'opérateurs). Résoudre le problème signifie trouver un ensemble $\wp$ (en pratique, un modèle de tâches) tel que :

$$\forall \pi \in L(\wp),\ g \subseteq \gamma(s_0, \pi)$$

Dans le cas où aucun modèle des tâches n'est trouvé, le problème n'a pas de solution et aucune IHM n'est composée. Dans le cas où π existe, les algorithmes de planification ne garantissent pas que le modèle de tâches correspondant soit le « meilleur », mais qu'il permet à l'utilisateur de réaliser son objectif dans le contexte considéré.

5. Prototype *Compose*

Compose est un démonstrateur de composition d'IHM par planification. La figure 14 en présente une décomposition fonctionnelle. Seules les fonctions « *Model-based composer* » et « *Code composer* » sont implémentées. L'implémentation est effectuée en JAVA.

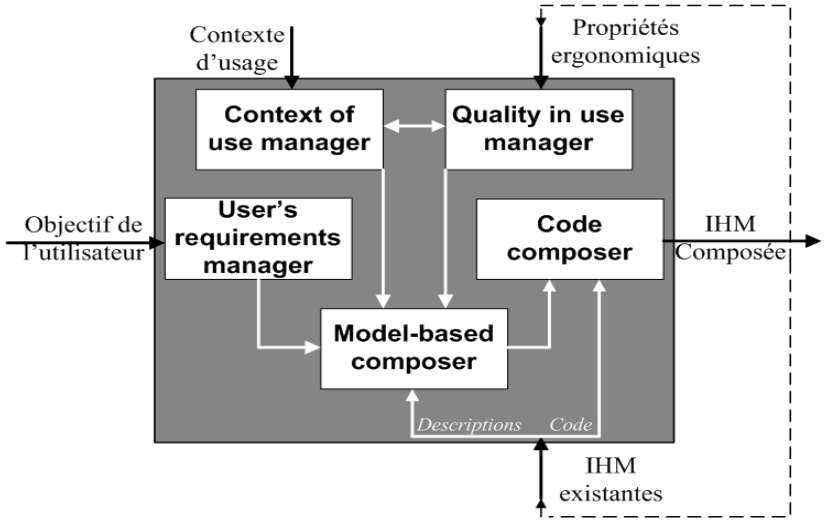


Figure 14. *Décomposition fonctionnelle de Compose.*

La fonction **« Context of use manager »** est chargée de percevoir et d'interpréter le contexte d'usage courant. La fonction **« Quality in use manager »** est chargée de gérer les critères d'ergonomie à privilégier (Scapin *et al.*, 1997). Dans *Compose*, le contexte d'usage et les critères d'ergonomie sont modélisés manuellement. Ces modèles sont traduits en propositions logiques utilisant les objets du contexte. Par exemple, Victor et le mur numérique sont représentés par des objets. Pour exprimer le fait que Victor a accès au mur, on utilise une proposition *« has (Victor, Mur)* ». Cette traduction est automatique à partir d'un modèle du contexte d'usage. La fonction **« User's requirement manager »** est chargée de percevoir l'objectif de l'utilisateur et de l'interpréter en un objectif connu du système. Par exemple, l'objectif « Je veux aller voir un médecin » est interprété en un but modélisé par la méthode « *Get medical assistance* ».

Les fonctions **« Model-based composer»** et **« Code composer »** constituent le cœur de *Compose*. Le **« Model-based composer»** est effectué par le planificateur. Le **« Code composer »** utilise le modèle de tâches produit par le planificateur et le traduit en une IHM composée. Une fois que le niveau tâches est composé par le planificateur, les COMETs (Demeure *et al.*, 2008), associées respectivement aux

opérateurs ou méthodes instanciés, sont déployées sur la ou les plates-formes sélectionnées par le planificateur. Les COMETs embarquent tous les niveaux d'abstraction d'une IHM. La capitalisation de l'IHM composée est souhaitée par les utilisateurs (Gabillon *et al.*, 2009) pour une raison de stabilité (Figure 14).

Dans une première section, les IHM existantes et les arbres de tâches calculés pour le cas d'étude sont présentés. Ces IHM existent sous la forme de COMETs et de descriptions en planification. Dans une seconde section, le **« Code composer »** est décrit. Il compose les codes des IHM existantes à partir de leurs modèles composés par le planificateur.

5.1 IHM existantes et arbres de tâches calculés

Les IHM sont implémentées par des COMETs. Chaque COMET est décrite par un opérateur ou une méthode en planification. Dans le cas d'étude, huit COMETs (Figure 15) sont utilisées. La COMET (f) est un canevas avec un agencement vertical. Dans la version pour le mur numérique, le canevas contient les COMETs (c), (d) et (e). La COMET (g) représente une séquence temporelle entre COMETs. Dans le cas d'étude, cette COMET est utilisée entre les COMET (a) et (b) et les COMETs (b) et (f). Quand la COMET (a) se termine, la COMET (b) se déploie. Quand la COMET (b) se termine, la COMET (f) se déploie.

Les IHM de la Figure 15 sont décrites par des opérateurs ou des méthodes de même nom :

– *Get medical assistance* : les paramètres sont un utilisateur (*?u*). Pour être utilisable, cette méthode nécessite que l'utilisateur ait une plate-forme. Les sous-tâches sont successivement *Choose the city*, *Choose the doctor* et *Contact the doctor*.

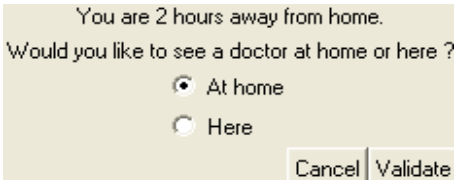
– *Choose the doctor et Choose the city* : les paramètres sont un utilisateur et une plate-forme. Ces opérateurs requièrent que la plate-forme soit connectée à internet (*internet ?p*). Le médecin et la ville sélectionnés sont ajoutés à l'état du monde (*isChosen Victor DoctorChosen et isChosen Victor CityChosen*).

– *Contact the doctor* : les paramètres sont un utilisateur et sa plate-forme. La méthode requiert que l'écran de la plate-forme supportant l'interaction soit petit (isSmallScreen ?p). Cette méthode est utilisée dans les cas où l'espace d'affichage est petit comme la version 2 du cas d'étude. Ses sous-tâches sont parallèlement *Call the office*, *Find route information*.

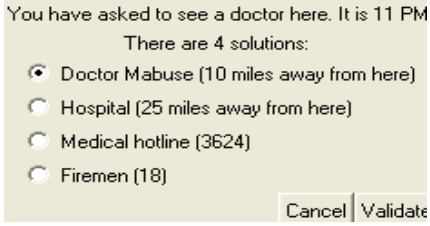
– *Contact the doctor* : les paramètres sont identiques à la méthode précédente. La méthode requiert par contre que l'écran de la plate-forme soit grand (isLargeScreen ?p) et que le critère « Charge de travail – Densité informationnelle » soit prioritaire (*high_Workload_InformationDensity*). Cette méthode est utilisée lorsque l'espace d'affichage est suffisant comme dans les versions 1 et 3 du cas d'étude. Ses sous-tâches sont parallèlement *Call the office*, *Find route information* et *Find nearest chemist*.

– *Call the office, Find route information et Find nearest chemist* : ces opérateurs décrivent respectivement l'IHM (c), (d) et (e). Les paramètres sont un utilisateur et sa plate-forme. Ces opérateurs requièrent que l'utilisateur ait une plate-forme dotée de capacités téléphoniques (*phone ?p*) pour la première ou connectée à internet pour les autres.

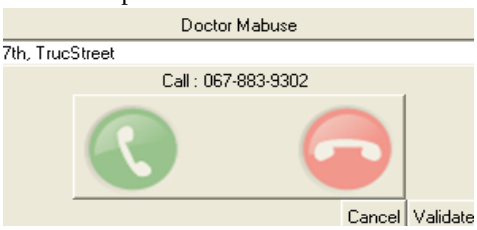
(a) COMET « Choose an option » instanciée pour « Choose the city »



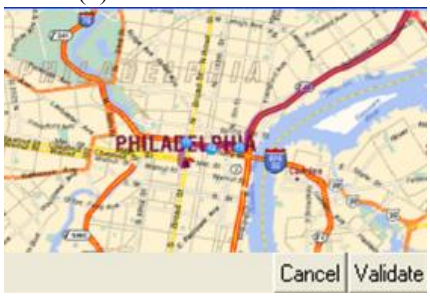
(b) COMET « Choose an option » instanciée pour « Choose the doctor»



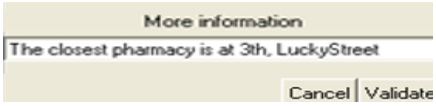
(c) COMET « Call somebody » instanciée pour « Call the doctor »



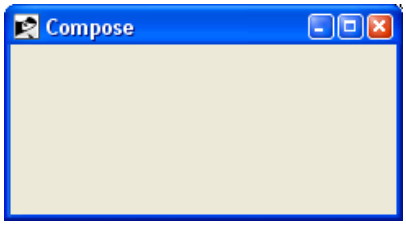
(d) COMET « Find route »



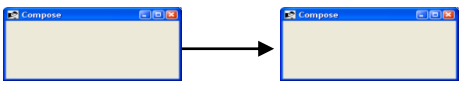
(e) COMET « Display text » instanciée pour « Find nearest chemist »



(f) COMET « Vertical layout » instanciée pour « Contact the doctor »



(g) COMET « Next » instanciée pour « Get medical assistance »



(h) COMET « Interleaving » instanciée pour« Contact the doctor »

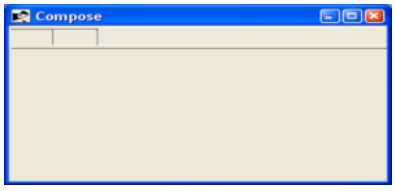
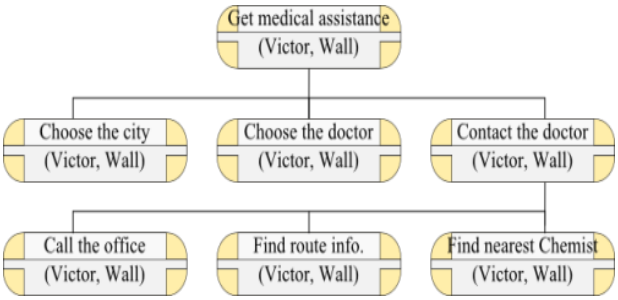
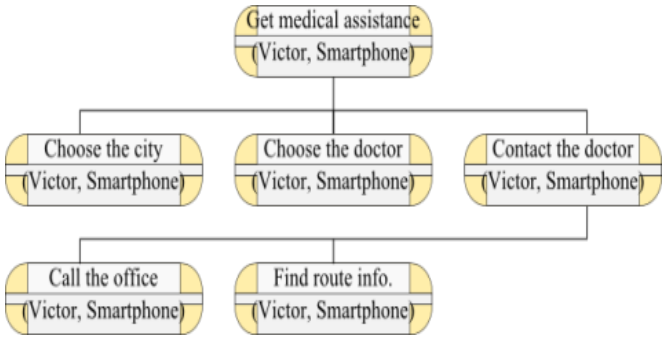


Figure 15. *Huit IHM existantes pour le cas d'étude « Get medical assistance ».*

Le **« Model-based composer»** est le planificateur développé pour la composition d'IHM. Il utilise les opérateurs et les méthodes. Il produit un arbre de tâches selon le contexte d'usage courant et l'objectif de l'utilisateur. Pour le cas d'étude, pour le smartphone et le mur numérique, les arbres de tâches de la figure 16 sont calculés par le planificateur. Pour le mur numérique, sept tâches sont composées. Pour le smartphone, l'arbre de tâches obtenu est différent. Seules six tâches sont composées. La tâche optionnelle « Find nearest chemist » est omise et les sous-tâches de « *Goto doctor* » sont composées dans un onglet. Dans le cas du mur numérique, la méthode instanciée *Contact the doctor* qui a trois sous-opérateurs est sélectionnée car l'écran de la plate-forme est grand (Figure 16a). Dans le cas du smartphone, c'est la méthode instanciée *Contact the doctor* avec deux sous-opérateurs qui est sélectionnée car l'écran est petit (Figure 16b).



(a) pour le mur numérique



(b) pour le Smartphone

Figure 16. *Modèles de tâches obtenus par le planificateur pour le mur numérique et pour le smartphone.*

L'arbre de tâches est fourni au **« Code composer »** qui va remplacer ces opérateurs et méthodes instanciés par les COMETs correspondantes.

5.2. Code composer

Le **« Code composer »** traduit l'arbre de tâches obtenu par le planificateur en un arbre de COMETs. A l'exécution, les opérateurs et les méthodes instanciés sélectionnés sont statiquement associés aux COMETs correspondantes. Ces COMETs sont déployées en fonction des consignes du planificateur. Par exemple, dans *Compose*, la méthode *Get medical assistance* est transformée en une COMET. Cette COMET, déployée sur le mur numérique, affiche ses sous-opérateurs dans des fenêtres en séquence.

Le **« Code composer »** compose les COMETs pour produire une IHM composée. Les tâches de l'arbre de tâches calculées pour le mur numérique sont liées à leurs COMETs respectives (Figure 17). La méthode *Get medical assistance* est liée à la « COMET (g) » qui affichera dans des fenêtres en séquence les « COMET (a), (b) et (f) ». Une fois que toutes les tâches sont liées, l'IHM composée à l'exécution est déployée pour répondre à l'objectif de l'utilisateur.

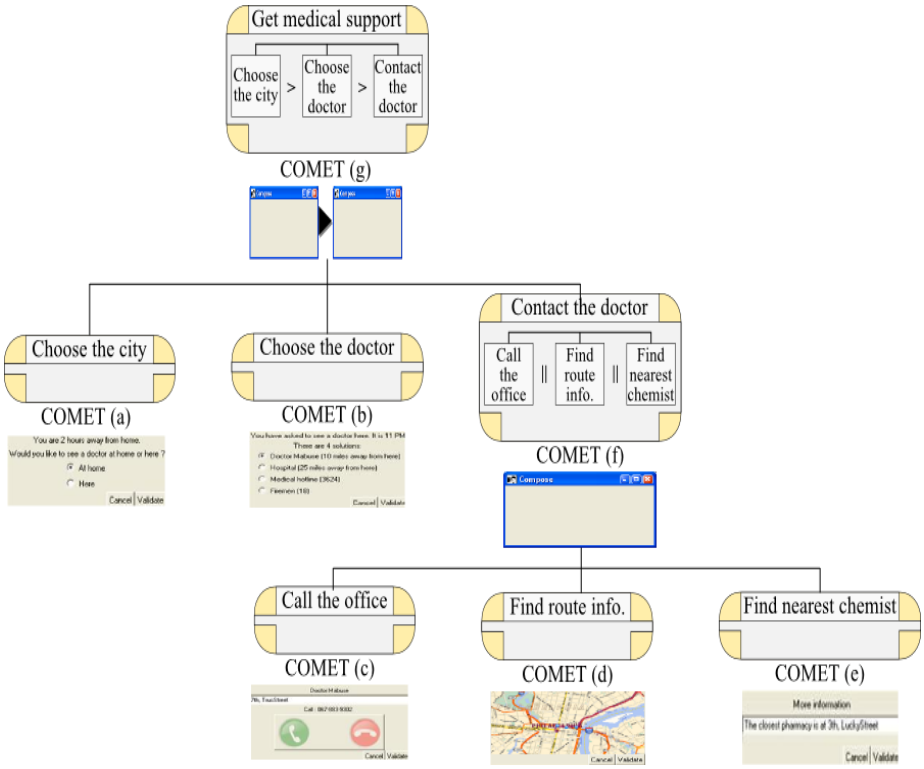


Figure 17. *L'arbre de tâches calculé pour le mur numérique. Chaque tâche est transformée en une COMET.*

6. Conclusion et perspectives

Dans cet article, nous avons traité de la composition d'IHM pour répondre à des objectifs utilisateur émergents. L'originalité est de composer un modèle de tâches. La composition s'appuie sur une base de composants réutilisables (les COMETs) et un contexte d'usage donné. L'algorithme relève de la planification automatique.

Si le démonstrateur *Compose* a montré la faisabilité algorithmique de l'approche, de nombreux verrous persistent. En particulier, la spécification et la satisfaction des exigences ergonomiques sont une difficulté majeure. Les planificateurs raisonnent sur des états du monde et des objectifs à atteindre mais ne traitent pas de propriétés non fonctionnelles. Ici, la difficulté est de taille, les critères d'ergonomie souffrant d'un manque de formalisation.

De même, le temps de calcul sera une difficulté pour l'IHM : il faudra non seulement le prédire pour l'annoncer à l'utilisateur mais assurer en outre une borne maximale acceptable par l'humain. Ce temps de calcul est dépendant du nombre d'IHM à composer. Si l'approche par COMETs favorise la séparation des préoccupations (une description par niveau d'abstraction isolant, en particulier, le niveau tâches du rendu) et prépare en conséquence le passage à l'échelle, la vérification est nécessaire en pratique. Pour cela, il nous faut des bases de données d'IHM réutilisables, bases dont la communauté IHM ne dispose pas. Les travaux menés en langages de description d'IHM (par exemple, UsiXML) devraient faciliter ces capitalisation et réutilisation.

Enfin, l'utilisateur pourrait souhaiter un contrôle sur le processus de composition. Dès lors, la problématique de la composition d'IHM s'ouvre sur la programmation par l'utilisateur final (End-User Programming, Myers, 2009), domaine en pleine expansion aujourd'hui. On voit là toute la richesse mais aussi la difficulté du sujet.

7. Remerciements

Ce travail a été financé par le projet PRESENCE du cluster ISLE (Informatique, Signal, Logiciel Embarqué) de la Région Rhône-Alpes. Il se poursuit aujourd'hui dans le projet européen ITEA UsiXML (2009-2012).

8. Bibliographie

Acher M., Collet P., Lahire P., France R., « Managing Variability in Workflow with Feature Model Composition Operators », *Proceedings of 9th International Conference on Software Composition*, Malaga, Spain, 1-2 juillet 2010, Springer, p. 17-33.

Andrews T., Curbera F., Dolakia H., Goland J., Klein J., Leymann F., Liu K., Roller D., Smith D., Thatte S., Trickovic I., Weeravarana S.. « Business Process Execution Language for Web Services, version 1.1 », 2003.

Balme L., Interfaces hommes-machine plastiques : une approche par composants dynamiques, Thèse de doctorat, Université Joseph Fourier, 2008.

Calvary. G., Plasticité des Interfaces Homme-Machine, Habilitation à Diriger des Recherches préparée au Laboratoire d'Informatique de Grenoble (LIG), Université Joseph Fourier, 2007.

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, n° 3, 2003, p. 289-308.

Demeure A., Calvary G., Coninx K., COMET(s), *A Software Architecture Style and an Interactors Toolkit for Plastic User Interfaces*, Springer-Verlag, 2008, p. 225-237.

Favre J.M., « Foundations of model (driven)(reverse) engineering » *Proceedings of Dagsthul Seminar on Model Driven Reverse Engineering*, Dagsthul, Germany, Internationales Begegnungs und Forschungszentrum, 2004.

Fujima J., Lunzer A., Hornboek K., Tanaka Y., « Clip, connect, clone: combining application elements to build custom interfaces for information access», *Proceedings of the 17th annual ACM symposium on User interface software and technology*, ACM Press, 2004, p. 175-184.

Gabillon Y., Calvary G., Mandran N., Fiorino H., « Composition dynamique d'interfaces homme-machine: besoin utilisateur ou défi de chercheur ? », *Proceedings of the 21st International Conference on Association Francophone d'Interaction Homme-Machine IHM'09*, Grenoble, France, 13-16 octobre 2009, ACM Press, p. 61-64.

Hinckley K., « Synchronous gestures for multiple persons and computers », *Proceedings of the 16th annual ACM symposium on User interface software and technology*, Vancouver, Canada, 2-5 novembre 2003, ACM Press, p.149-158.

Johnson P., Wilson S., Markopoulos P., Pycock J., « ADEPT: Advanced Design Environments for Prototyping with Task Models », *Proceedings of the Conference on Human Factors in Computing Systems INTERCHI*, Amsterdam, The Netherlands, 24-29 avril 1993, ACM Press, p. 56-57.

Kiczales G., Lamping J., Mendhekar A., Maeda C., Lopes C., Loingtier J.M., Irwin J., « Aspect-Oriented Programming », *Proceedings of 11th European Conference on Object-Oriented Programming ECOOP*, Jyvskyl, Finland, 9-13 juin 1997, Springer, p. 220-242.

Klusch M., Gerber M., Schmidt M., « Semantic web service composition planning with OWLSXPlan », *Proceedings of the International Fall Symposium on Semantic Web and Agents AAAI*, Arlington, USA, 13-15 octobre 2006, AAAI Press, 2006, p. 55-62.

Kuter U., Sirin E., Parsia B., Nau D., Hendler J., « Information gathering during planning for web service composition », *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 3, n° 2-3, 2003, p. 183-205.

Lepreux S., Vanderdonckt J., « Towards Supporting User Interface Design by Composition Rules », *Proceedings of 6th Int. Conf. on Computer-Aided Design of User Interfaces CADUI*, Bucharest, Romania, 5-8 juin 2006, Springer, p. 231-244.

Lin J., Wong J., Nichols J., Cypher A., Lau T.A, « End-user programming of mashups with vegemite », *Proceedings of the 13th international conference on Intelligent user interfaces*, Sanibel Island, USA, 8_11 février 2008, ACM Press, p. 97-106.

Martin D.. Burstein M.. Hobbs J.. Lassila O.. McDermott D.. McIlraith S.. Narayanan S.. Paolucci M.. Parsia B.. Payne T.. Sirin E., Srinivasan N., Sycara K., « OWL-S: Semantic Markup for Web Services », *W3C Member Submission*, vol. 22, 2004.

Mori G., Paternò F., Santoro C., « CTTE: support for developing and analyzing task models for interactive system design », *IEEE Transactions on software engineering*, vol. 28, n° 8, 2002, p. 797-813.

Myers B., Hudson S.E., Pausch R., « Past, present, and future of user interface software tools », *ACM Trans. Computer-Human Interaction*, vol. 7, n° 1, 2000, p. 3-28.

Myers, B.A. « Engineering more natural interactive programming systems: keynote talk », *Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems EICS*, Pittsburg, USA, 15-17 juillet 2009, ACM Press, p. 1-2.

Ghallab M., Nau D., Traverso P., *Automated Planning : Theory & Practice*, San Francisco, USA, Morgan Kaufmann Publishers Inc, 2004.

Nichols J., Hua Z., Barton J., « Highlight : a system for creating and deploying mobile web applications », *Proceedings of the 21st annual ACM symposium on User interface software and technology*, Monterey, USA, 19-22 octobre 2008, ACM Press, p. 249-258.

Nielsen J., *Usability Engineering*, Boston, MA, USA, Academic Press, 1993.

Paternò F., Mancini C., Meniconi S., « ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models », *Proceedings of the 13 Interantional Conference on Human-Computer Interaction IFIP TC*, Sydney, Australia, 14-18 juillet 1997, Chapman & Hall, p. 362-369.

Paternò, F., Santoro C., Scorcia A., « Preserving Rich User Interface State in Web Applications across Various Platforms », *Proceedings of the 2nd Conference on Human-Centered Software Engineering and 7th International Workshop on Task Models and Diagrams*, Pisa, Italy, 25-26 septembre 2008, Springer, p. 255-262.

Perrouin, G., Brottier, E., Baudry, B, Le Traon, Y. « Composing Models for Detecting Inconsistencies: A Requirements Engineering Perspective », *Proceedings of Requirements Engineering: Foundation for Software Quality*, Amsterdam, The Netherlands, 8-9 juin 2009, p. 89-103.

Pinna-Dery A. M., Fierstone J., Picard E., « Component model and programming: a first step to manage human computer interaction adaptation », *Proceedings of 5th International Symposium on Human- Computer Interaction with Mobile Devices and Services MobileHCI*, Udine, Italie, 8-11 septembre 2003, Springer, p. 456-460.

Rekimoto, J., Ullmer, B., H. Oba, « DataTiles: a modular platform for mixed physical and graphical interactions », *Proceedings of the conference on Human factors in computing systems CHI*, Seattle, USA, 31 mars-5 avril 2001, p. 269-276.

Scapin D.L., Bastien J.M.C., « Ergonomic criteria for evaluating the ergonomic quality of interactive systems », *Behaviour & Information Technology*, vol. 6, n° 4-5, 1997, p. 220-231.

Sottet J-S., Ganneau V., Calvary G., Coutaz J., Demeure A., Favre J-M., « Demumieux, R. Model-driven adaptation for plastic user interfaces », *Proceedings of the 11th International Conference on Human-Computer Interaction INTERACT'07*, Rio de Janeiro, Brasil, 10-14 septembre 2007, Springer, p. 397-410.

Stuerzlinger W., Chapuis O., Phillips D., Roussel N., « User interface facades : towards fully adaptable user interfaces », *Proceedings of the 19th annual ACM symposium on User interface software and technology*, Montreux, Switzerland, 15-18 octobre 2006, ACM Press, p. 309-318.

Thevenin D., Coutaz J., « Plasticity of user interfaces: Framework and research agenda », *Proceedings of the International Conference on Human-Computer Interaction INTERACT*, Edinburgh, USA, 30 aout-3 septembre 1999, p. 110-117.

Traverso P., Pistore M., « Automated Composition of Semantic Web Services into Executable Processes », *Proceedings of third International Semantic Web Conference ISWC*, Hiroshima, Japan, 7-11 novembre 2004, Springer, p. 380-394.

Weiser M., « The computer for the 21st century », *Scientific American*, vol.3, n° 265, 1991, p. 66-75.

Calvary G., Plasticité des Interfaces Homme-Machine, PhD thesis, 2007. Thèse Habilitation à Diriger des Recherches préparée au Laboratoire d'Informatique de Grenoble (LIG), Université Joseph Fourier.

Calvary G., Serna A., Coutaz J., Scapin D., Pontico F., Winckler M., *Envisioning Advanced User Interfaces for E-Government Applications : A Case Study*, Springer, p. 205-228, 2011. Assar, Saïd ; Boughzala, Imed ; Boydens, Isabelle (Eds.).

Gabillon Y., Calvary G., Mandran N., Fiorino H., « Composition dynamique d'interfaces homme-machine: besoin utilisateur ou défi de chercheur ? », *Proceedings of the 21st International Conference on Association Francophone d'Interaction Homme-Machine IHM'09*, Grenoble, France, 13-16 octobre 2009, ACM Press, p. 61-64.

Gabillon Y., Petit M., Calvary G., Fiorino H., « Automated planning for user interface composition », *Proceedings of the 2nd International Workshop on Semantic Models for Adaptive Interactive Systems : SEMAIS'11 at IUI 2011 conference*, Springer HCI, 2011.

Pellier D., Fiorino H., « Un modèle de composition automatique et distribuée de services web par planification », *Revue d'Intelligence Artificielle*, vol. 23, n° 1, p. 13-46, 2009.

Accepté après révisions le 17 mars 2011

Gaëlle Calvary *est professeur à Grenoble INP. Ses travaux portent sur la plasticité des Interfaces Homme-Machine (IHM) dans le cadre de l'informatique ambiante. Son but est de fournir des modèles, méthodes et outils pour soutenir le développement d'IHM plastiques. Une IHM est dite plastique si elle est capable de s'adapter à des variations du contexte d'usage (utilisateur, plate-forme, environnement) tout en préservant les propriétés attendues par l'utilisateur. L'approche qu'elle a le plus explorée est l'Ingénierie Dirigée par les Modèles.*

Humbert Fiorino *est maître de conférences à l'Université Joseph Fourier. Ses travaux portent sur la planification distribuée et autonome ainsi que sur les systèmes multi-agents. Actuellement, il recourt à ces approches afin de générer automatiquement des interfaces utilisateur et de composer des services web.*

Yoann Gabillon *termine un doctorat en informatique à l'Université Joseph Fourier. Ses travaux traitent de la composition d'interfaces Homme-Machine. Il explore les algorithmes de planification dans le but de produire une IHM composée selon le contexte d'usage et les propriétés ergonomiques à satisfaire.*