MajecSTIC 2009
Avignon, France, du 16 au 18 novembre 2009

Une approche de description d'Interfaces Homme–Machine multi-niveaux

Christian Brel

Laboratoire I3S (Université de Nice-Sophia Antipolis - CNRS)
Polytech'Nice-Sophia - 930 route des Colles
06903, Sophia Antipolis Cedex, France

Contact : brel@polytech.unice.fr

Résumé
Les travaux dans le domaine du génie logiciel tendent à offrir toujours plus de possibilités en terme de réutilisation. Les concepteurs de sites web sont confrontés à cette problématique : construire et faire évoluer les applications web en combinant des services provenant de sources diverses. L'opération de composition des services implique de construire une nouvelle Interface Homme Machine (IHM), c'est-à-dire de nouvelles pages web, afin de permettre l'accès et l'utilisation combinée de ces services distants. Puisque le concepteur d'applications web n'a pas accès aux services eux mêmes mais seulement à leurs IHM, il est alors plus intéressant d'exploiter les principes de composition et de réutilisation directement au niveau des IHM. Nous proposons alors d'utiliser de façon combinée différents travaux autour de la composition menés au sein de la communauté IHM. Cet article présente un formalisme de description des IHM permettant de garantir une composition plus cohérente car recoupant des informations provenant de différents niveaux de conception des IHM.

Abstract
Software Engineering composition principles allow for the construction of new applications by reusing other application building blocks. Following such principles, webmasters aggregate services to create a website quickly. Composing services implies creating a new User Interface (UI) - in our case a new web page - to use them. Webmasters cannot access services' description directly but they can leverage information about services' UI easily. Thus, it is more interesting to reuse and compose UIs than the services themselves. We propose a solution based on existing work on HCI to reuse and compose UIs in a consistent way. This article introduces a UI description formalism on which is based our composition approach.

Mots-clés : Interface Homme–Machine, composition, tâches

Keywords: user interface, composition, tasks

---

1. Introduction

La démocratisation d'Internet et son adoption dans la majorité des foyers en fait un outil de communication puissant. Par conséquence, Internet est une excellente vitrine pour les entreprises et les marques associées. Ces dernières proposent des sites Web de plus en plus évolués devenant de véritables applications proposant de la valeur ajoutée (service de réservation ou d'achat en ligne, comparateurs de prix ou de trajets, stockage virtuel de données, ...) pour l'utilisateur et plus seulement des pages d'informations.

Pour attirer les visiteurs, les applications web doivent évoluer en permanence, savoir se renouveler pour proposer toujours plus de valeur ajoutée. Le besoin de visibilité et de réactivité oblige les concepteurs d'applications web à repenser la manière de concevoir ces applications et en particulier leurs Interfaces Homme Machine (IHM). Les applications ne sont plus conçues pour un

usage prédéterminé, elles ne sont plus distribuées par un canal classique d'achat mais largement diffusées sur internet et enfin elles ne sont plus limitées uniquement aux postes de bureau. En effet, les développeurs mettent à disposition, sur des sites web, de plus en plus de services et qui vont permettre à d'autres développeurs de construire sur mesure d'autres services.

Ces services sont éphémères car ils répondent à un besoin nécessaire à un instant t et vont donc évoluer ou être améliorés en fonction de nouveaux services apparaissant au fil du temps. Le concepteur d'une application web devra alors construire de nouveau une IHM (dans son cas, de nouvelles pages web) afin de permettre l'utilisation combinée des différents services.

Nous partons de l'hypothèse qu'il est plus facile de composer la partie visible des services, les IHM, plutôt que les services eux mêmes pour lesquels le développeur n'a pas toujours accès à la description complète. Nous proposons alors d'utiliser de façon combinée différentes formes de composition des IHM et donc de réutilisation de ces dernières. Cette combinaison des approches permet d'aller vers une composition des IHM plus cohérente car recoupant différentes informations provenant de différents niveaux de conception des IHM. Cette nouvelle approche de la composition dite « multi-niveaux » s'appuie sur une façon de décrire les IHM bien particulière. Cet article présente le formalisme de description des IHM pour la composition multi-niveaux (le mécanisme de composition lui-même n'étant pas détaillé ici).

La suite de l'article est organisée comme suit. La section 2 décrit l'étude de cas sur laquelle nous nous appuyons pour illustrer les différentes approches. La section 3 présente les travaux relatifs à la composition d'IHM qui ont inspiré notre approche multi-niveaux. La section 4 introduit le formalisme utilisé pour décrire les IHM dans l'approche de composition multi-niveaux. La section 5 conclut sur l'intérêt de cette approche.

2. Etude de cas : recherche de recettes de cuisine et calcul du prix de revient

Cette section décrit une étude de cas qui va nous permettre dans la suite de l'article d'expliquer les différentes notions et principes abordés. Cette étude de cas utilise des services simplifiés afin de décrire au mieux la solution proposée. L'étude de cas met en éveil les talents culinaires de chacun puisqu'il s'agit d'utiliser deux services qui vont nous permettre de confectionner de bons petits plats :

- le premier est un service de recherche de recettes, l'utilisateur renseigne le nom de la recette recherchée et obtient au final la description de la recette avec notamment les ingrédients à utiliser ;
- le second est un service d'achat d'articles consommables, pour lequel l'utilisateur devra remplir un panier soit en sélectionnant les articles un par un, soit en donnant directement une liste de courses, dans le but d'obtenir le prix total de son panier.

L'IHM associée au service de recherche d'une recette se découpe en trois écrans [1] comme décrit dans la figure 1. Le premier permet à l'utilisateur de renseigner le nom de la recette et de lancer la recherche. Sur le second, l'utilisateur voit apparaitre une liste de noms de recettes avec une petite description rapide. Il peut alors cliquer sur le nom d'une recette et lancer la visualisation de celle-ci. Le dernier écran est la description de la recette en elle-même avec son nom complet, le temps de préparation et de cuisson, les ingrédients à utiliser et enfin les étapes à suivre pour réussir la recette. L'IHM associée au service d'achat d'articles consommables se découpe en 4 écrans [2] dont 3 décrits dans la figure 2. Correspondant au premier écran, la première tâche de l'utilisateur sera de remplir le panier. Pour ce service, l'utilisateur a deux possibilités pour effectuer cette action, soit il ajoute des ingrédients à son panier en les sélectionnant dans des catégories prédéterminées, soit il renseigne une zone de texte qui décrit sa liste de courses avec un ingrédient par ligne. Pour cette deuxième possibilité, une fois sa liste de course validée, il va alors, dans un second écran, pour chaque ingrédient, choisir la quantité voulue et ajouter l'ingrédient au panier. L'utilisateur valide ensuite son panier et voit apparaître, sur le troisième écran, la facture. Il peut alors valider

1. IHM simplifiée provenant du site http ://www.marmiton.org
2. IHM simplifiée provenant du site http ://www.telemarket.fr

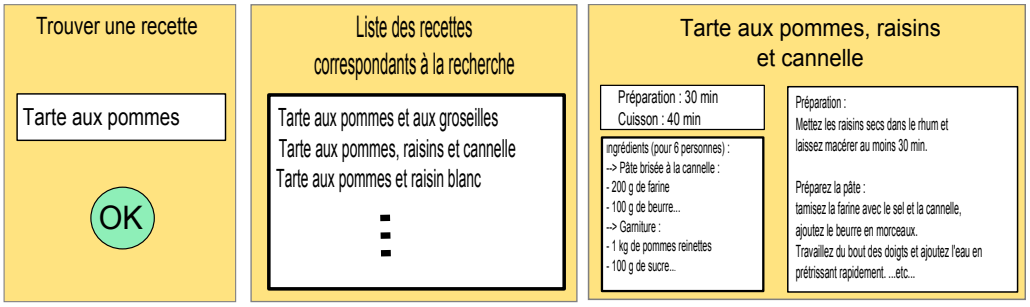

FIGURE 1 – IHM du service de recherche d'une recette.

la facture et effectuer le paiement. Le quatrième écran pour le paiement n'est pas montré dans la figure 2 car il n'apporte rien dans le cadre de notre étude de cas puisque nous voulons seulement obtenir, après avoir effectuée la composition, le coût de la recette. Le but est de composer

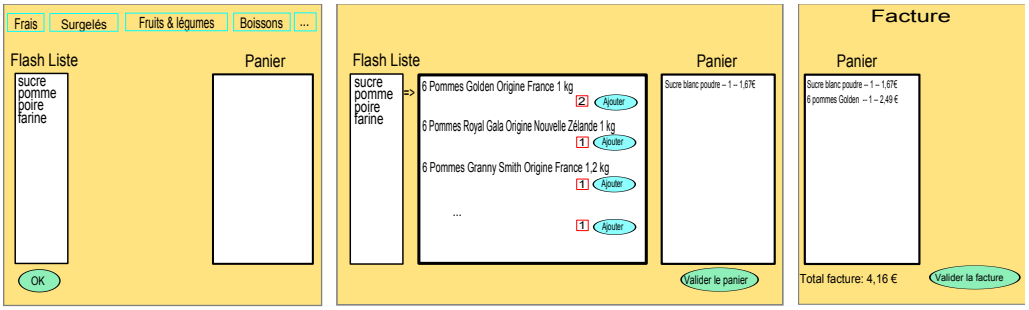

FIGURE 2 – IHM du service d'achat.

ces deux services afin d'obtenir un service de recherche de recette qui nous donne aussi le coût de revient de la recette. Effectivement, lorsque nous choisissons une recette, nous ne savons pas comment évaluer son coût de revient. Nous pouvons constater dans cette étude de cas que ces deux services sont disponibles sur différents sites web. Par conséquent, sans composition, l'utilisateur est obligé de faire la visite de plusieurs sites web pour réaliser une activité donnée (ici, obtenir la recette avec son prix de revient). Une solution serait de reconstruire une application qui nous permette de faire cela. Ce développement peut être fastidieux et très couteux. De plus, dans un monde où l'évolution des applications doit être dynamique et rapide, cette solution ne peut être satisfaisante car le développement complet d'une application ne permet pas d'être assez réactif. Une alternative est de composer des services existants et donc de composer les interfaces qui les accompagnent.

3. La composition d'IHM : travaux existants

Cette section présente certains travaux menés dans le cadre de la composition d'IHM, que celles-ci soient conçues pour le Web ou pour des applications installées directement sur le poste de travail. Ces travaux ne sont pas seulement en relation avec la composition mais aussi avec l'adaptation des IHM à leur contexte d'utilisation.

La communauté IHM préconise de développer des interfaces à des niveaux multiple d'abstraction de manière indépendante afin de produire des IHM de qualité et exploitables dans divers contextes (différents systèmes d'exploitation, différents langages de programmation, différents supports tels que les Pcs, les PDAs ou encore les téléphones, etc) [6]. A partir de n'importe lequel de ces niveaux, le développeur peut obtenir différentes variantes d'une même IHM selon le

contexte d'utilisation. Le passage d'un niveau au suivant se fait plus ou moins automatiquement. Quatre niveaux d'abstraction sont identifiés :

- les tâches utilisateur et concepts du domaine qui décrivent l'IHM selon une perspective domaine, sans préjuger, d'une quelconque représentation. Les concepts du domaine (par exemple, les notions de recette, d'ingrédient, d'article, de prix) sont les entités manipulées par les tâches (par exemple, la recherche de recettes, la validation du panier d'article, etc) ;
- l'interface abstraite qui structure l'IHM en espaces de dialogue (une page web par exemple) et fixe la navigation entre ces espaces. A ce niveau, on reste indépendant de tout contexte d'utilisation ;
- l'interface concrète qui affine l'IHM en introduisant les choix d'interacteurs (boutons, menus, etc.) en fonction d'un contexte d'utilisation donné (l'action « valider » peut être associée à un clic sur un bouton dans une IHM pour un poste de travail ou à une commande vocale dans le cadre d'une IHM sur téléphone portable dans un contexte « mains libres » par exemple) ;
- l'interface finale, qui correspond à implémenter l'interface concrète dans un langage donné (Java SWING, Adobe Flex, HTML, etc) et un système d'exploitation spécifique.

Dans le domaine du Web, les travaux autour de la composition d'IHM se font directement au niveau « interface finale ». Les travaux les plus répandus concernent les mashups c'est-à-dire de « petites » interfaces proposant un service particulier (la météo, les news, pages jaunes, programmes télé, etc). Il existe deux types de travaux qui se différencient par le fait d'utiliser ou non les flots de données et qui répondent en réalité à deux besoins différents : celui de partager des données et de faire différents traitements sur ces données et celui d'avoir un accès rapide à différents services sur une seule page.

Nous pouvons citer IGoogle [3] ou Netvibes [4] comme outils utilisants les mashups pour faire de la présentation et répondre au besoin d'accès rapide aux services. Pour ce type de mashups, la seule composition possible est en réalité le regroupement de plusieurs mashups dans des catégories définies par l'utilisateur. Ces portails Web ne permettent pas d'effectuer des échanges de données entre les différents mashups. Par exemple, pour notre cas d'étude, nous aurions un mashup pour la recherche de recettes et un autre pour l'achat d'articles consommables. Les deux pourraient alors être mis côte à côte, ce qui représente une amélioration puisque les deux services sont alors accessibles tous deux sur une même page web et non plus sur deux pages web différentes.

Popfly [1] de Microsoft ou Yahoo Pipes [2] utilisent les flots de données afin d'effectuer l'enchaînement entre plusieurs mashups. Ces outils répondent au besoin de partage de données. Graphiquement reliées au moyen de flèches, les sorties d'un ou de plusieurs mashups vont correspondre aux entrées du mashup suivant. L'interface du dernier mashup est réutilisée comme interface résultat de la composition. Chaque mashup effectue le traitement pour lequel il est conçu et renvoie en sortie les données traitées. Ici, nos deux mashups de recherche de recettes et d'achat d'articles, seraient alors reliés par le flux de données correspondant à la liste des ingrédients qui sont aussi des articles consommables.

La réutilisation d'IHM fait partie intégrante de la composition des mashups de présentation même si celle-ci correspond simplement à une juxtaposition. Quant aux mashups avec flux de données, la composition se fait directement au niveau des services sous-jacents et seule l'IHM des mashups se trouvant en bout de chaine sont réutilisées.

D'autres approches travaillent en amont de l'interface finale afin de définir la composition indépendamment de toute contrainte liée au contexte ou de choix technique. Dans cette lignée, ComposiXML (Composition of Applications specified in UsiXML) [9] permet de réaliser la fusion de différentes IHM en une seule grâce à un certain nombre d'opérateurs (union, intersection, etc) et à un langage de description d'interfaces utilisateurs, USIXML (USer Interface eXtensible Markup Language) [8]. Dans ce contexte, la réutilisation des IHM de départ peut se faire au niveau « interface concrète ». Cependant, la composition mise en place est basée sur la structure de l'IHM et ne prend pas en compte les informations fonctionnelles liées au comportement du service, ce qui a pour conséquence une grande perte d'informations utiles pour la composition.

D'autres travaux montent encore d'un niveau d'abstraction en réalisant la composition à partir des descriptions des tâches utilisateur [5]. Ces travaux utilisent les arbres de tâches pour décrire l'enchaînement des interactions entre l'utilisateur et l'application. Un arbre de tâches va permettre de définir les tâches que l'interface doit supporter afin que l'interface soit fonctionnelle. Une tâche peut être un but ou une procédure, c'est-à-dire un ensemble de sous-tâches liées. Ainsi, la composition d'IHM consiste à composer les arbres de tâches correspondant à chaque IHM de départ. La composition de deux arbres résulte en la création d'un nouvel arbre avec l'ajout d'une tâche supplémentaire regroupant les tâches similaires dans les deux arbres de départ. Pour notre cas d'étude, nous pouvons retrouver des tâches similaires lors de l'affichage des informations sur la recette et l'affichage du prix total du panier. Pour regrouper ces deux affichages lors de la composition, nous pouvons alors appliquer cette méthode et réunir ces deux tâches d'affichage sous une tâche commune ce qui aura pour conséquence de regrouper ensemble les éléments d'IHM correspondants.

Ces derniers travaux sont intéressants dans la mesure où l'exploitation de l'arbre de tâches permet de prendre en compte l'aspect fonctionnel au niveau de la composition. Par exemple, l'enchainement des tâches (voir figure 3) lancer la recherche et sélectionner la recette dans une liste de recettes, pour le service de recherche de recettes, est une information précieuse pouvant justifier le fait que la deuxième tâche a besoin d'information provenant de la première pour bien se dérouler. Nous pouvons remarquer cependant que l'utilisation seule des arbres de tâches ne suffit pas pour effectuer la composition puisque nous n'avons aucune information sur le découpage et le placement des éléments dans l'IHM.

Jusqu'à présent, les travaux autour de la composition d'IHM ne s'appuient que sur un seul niveau d'abstraction. Nous proposons au contraire de tirer parti des informations relatives à différents niveaux afin de proposer des règles de composition plus complètes : éléments fonctionnels du niveau « tâches », éléments de découpage de l'IHM du niveau « interface abstraite » et éléments de placements du niveau « interface concrète ». La section suivante présente le formalisme de description des IHM que nous exploitons dans notre approche multi-niveaux de la composition.

4. Vers une composition multi-niveaux

Comme nous venons de le voir avec les travaux existants, il n'existe pas de réelle solution pour faire de la composition d'IHM qui exploite à la fois les informations d'ordre fonctionnel liées au comportement du service et les informations d'ordre structurel tels que le placement ou le regroupement/découpage. Cette section présente le formalisme qu'il faut utiliser afin de réaliser une composition d'IHM multi-niveaux.

Le point clef de l'approche est de mettre en relation les différents niveaux d'abstraction liés à la conception d'une IHM [6] : le premier est celui de la conception de l'IHM qui est la description des tâches auxquelles l'interface doit répondre ; le second est celui de la structure de l'interface avec le regroupement des éléments d'IHM ; le troisième, enfin, est celui de la problématique du placement des éléments dans une interface et qui est un problème récurrent lors de la conception d'une interface.

4.1. Arbres de tâches

Comme les travaux de Lewandowski et al. [5], nous partons du principe que la conception d'une IHM passe par l'établissement de l'arbre de tâches correspondant. Comme souligné dans la partie précédente, une tâche peut être un but ou une procédure, c'est-à-dire un ensemble de sous-tâches liées. Pour représenter un arbre de tâches, différents modèles de tâches sont disponibles. Nous avons choisi de nous baser sur CTT [7], un modèle riche permettant d'exploiter les informations sur l'enchaînement des tâches et leur place au sein de l'interface. Nous avons extrait un sous-ensemble de CTT des éléments nécessaires à la description que nous voulions obtenir pour l'arbre de tâches décrivant l'IHM et nous avons ajouté à ce sous-ensemble un pont entre l'arbre de tâche et l'arbre des regroupement d'éléments d'IHM.

Le modèle de tâches (TaskModel) contient une tâche principale. Cette tâche peut contenir des sous tâches ou alors être une tâche élémentaire. Pour chaque tâche nous pouvons lui ajouter une description, les liens de « parentés » qu'elle a avec les autres tâches (la tâche mère, les tâches soeurs). Une tâche peut être optionnelle. Elle appartient à une catégorie c'est-à-dire qu'une tâche va être abstraite (des sous-tâches permettent de la décrire), utilisateur (elle doit être réalisée par l'utilisateur), interactive (elle nécessite une interaction de la part d'un utilisateur ou d'une autre tâche) ou bien applicative (elle doit être réalisée automatiquement par l'application elle-même sans intervention de l'utilisateur).

Les tâches sont reliées entre elles par des opérateurs temporels. Par exemple, nous allons avoir des tâches qui doivent se réaliser l'une à la suite de l'autre, ou encore de manière parallèle ou indépendante, ou bien encore un choix peut s'effectuer entre la réalisation de la première ou de la deuxième tâche, etc...

Si nous reprenons l'interface de recherche d'une recette, nous pouvons décrire le modèle de tâches associé (voir figure 3) comme l'enchaînement de trois tâches principales : lancer la recherche (tâche 1), sélectionner la recette dans une liste de recettes (tâche 2) et enfin visualiser la recette (tâche 3). Hormis les liens de parentés entre la tâche mère « Recherche d'une recette » et les tâches principales ou entre les tâches principales elles-même, nous avons trois catégories de tâches qui apparaissent : la tâche mère est une tâche abstraite, détaillée par ses trois tâches filles dont les deux premières sont interactives et la dernière est applicative puisque ce n'est que de l'affichage d'informations. Les relations temporelles entre les tâches filles sont simples, la « tâche 1 » devra s'exécuter avant la « tâche 2 » et lui transmettra des informations, et nous retrouvons la même relation temporelle entre la « tâche 2 » et la « tâche 3 ». Un autre exemple de relation temporelle est présente dans cet arbre de tâche entre la tâche « AfficherNomRecette » et la tâche « AfficherTempsPreparationEtCuisson » où les tâches doivent s'exécuter obligatoirement (peu importe l'ordre d'exécution entre elles).

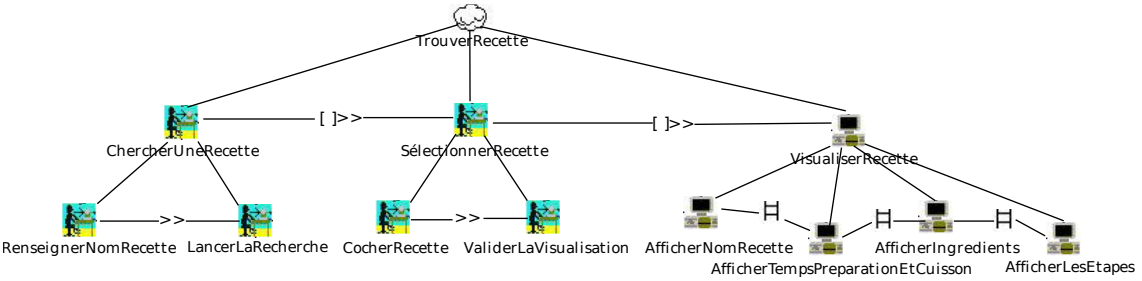

FIGURE 3 – Arbre de tâches du service de recherche d'une recette.

Enfin, notre approche implique de mettre en relation l'arbre de tâches avec les éléments d'IHM puisque nous avons comme but de faire la composition d'IHM multi-niveaux. Ainsi, nous avons une propriété sur les tâches qui permet de relier une tâche à n'importe quel élément d'IHM ou regroupement d'éléments d'IHM. Ce regroupement est défini dans un deuxième arbre comme nous allons le voir dans le prochain paragraphe.

Pour notre étude de cas, la composition des arbres de tâches serait d'ajouter, à l'arbre de tâche de la figure 3, à la tâche « VisualiserRecette » la sous-tâche « AfficherPrixFacture » issue de l'arbre de tâche du service d'achat. Pour cela, il faudra tenir compte des tâches à remplir pour obtenir toutes les informations nécessaires à la sous-tâche ajoutée, les ajouter au premier arbre de tâche et choisir le ou les opérateurs temporels à appliquer à la sous tâche afin de garantir une cohérence au niveau de l'arbre de tâche issu de la composition.

4.2. Regroupement des éléments d'IHM

Dans les travaux existants, la solution pour décrire des interfaces dans le but de faire de la composition est d'en faire l'abstraction. La structure d'arbre permettant naturellement d'exprimer des regroupements, nous avons choisi d'utiliser un arbre pour décrire les regroupements des éléments d'IHM. Notre modèle est composé d'entités (Entity) qui sont soit des ensembles (Ensemble) soit des éléments d'IHM (UIObject). Un ensemble peut contenir des sous ensembles ou des éléments d'IHM. Ce modèle permet de décrire l'interface en la découpant en blocs et par conséquent le fait de grouper des éléments d'IHM va pouvoir être interprété comme la volonté de rapprocher ces éléments d'IHM de manière sémantique.

Au niveau de l'étude de cas, nous pouvons voir dans la figure 4 que tous les éléments d'IHM sont regroupés dans un ensemble « InformationsRecette » qui exprime le fait que ces éléments vont apporter des détails en ce qui concerne la recette visualisée, puis nous avons un sous-ensemble qui est intitulé « Préparation » dans lequel nous allons trouver les informations nécessaires à la préparation de la recette afin d'être prêt à suivre les étapes de la recette.

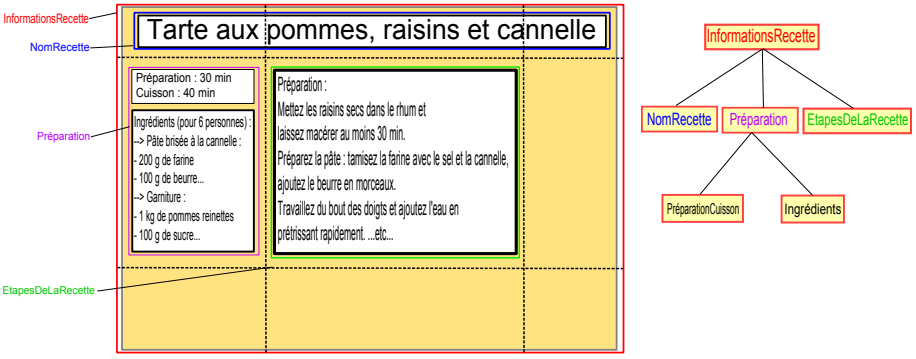

FIGURE 4 – Ensembles des éléments d'IHM pour la visualisation de la recette.

Pour compléter la description de l'interface, il est nécessaire d'exprimer le placement des éléments ou les regroupements d'éléments (ensembles) dans l'interface. Cet aspect est détaillé dans le paragraphe suivant.

4.3. Placement des éléments d'IHM

La description de l'interface se fait à travers un arbre de regroupements qui contient des ensembles et des éléments d'IHM. Après avoir regardé les placements mis en place dans différents langages de programmation (Java, GWT, Flex/MXML, etc...), nous avons choisi de découper les ensembles dans une grille à neuf positions comme décrit dans la figure 5. L'ensemble « Préparation » va se positionner à gauche dans l'ensemble père « InformationsRecette », puis l'élément « NomRecette » va se positionner en haut dans l'ensemble père et enfin, l'élément « EtapesDeLaRecette » va se situer au centre de ce même ensemble père. Avec cette description et le fait que les ensembles peuvent contenir des sous-ensembles, nous pouvons couvrir une grande majorité des placements possibles. Cependant, ce modèle nécessite d'être approfondi et détaillé en partant de plusieurs cas d'étude avant d'être généralisable.

5. Conclusion

Nous avons présenté dans cet article une approche permettant la description d'une IHM suivant un formalisme spécifique à la composition multi-niveaux. Cette approche met en place l'utilisation d'arbre de tâches permettant de décrire les buts dont l'IHM doit tenir compte, buts liés à la description des éléments d'IHM qui se fait au moyen d'une politique de regroupements mêlant aussi l'utilisation d'un placement afin de décrire la position des éléments dans l'interface. En utilisant cette approche de description des IHM, nous avons pour but de faciliter la composition des

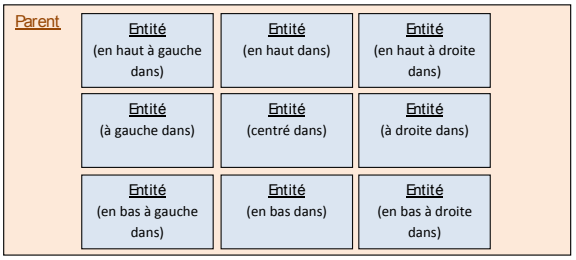

FIGURE 5 – Les neuf positions possibles dans un ensemble.

IHM. Pour cela, nous nous rapprochons du monde des ontologies, afin de décrire les trois points clefs de notre approche dans un langage qui va nous permettre grâce à un moteur sémantique, de pouvoir faire des manipulations des deux arbres (arbre de tâches et arbre de regroupements décoré par les propriétés de placement) plus complexes qu'une manipulation comme nous pourrions l'avoir avec XQuery pour manipuler des fichiers XML traditionnellement utilisés par les travaux de description d'IHM basés sur les langages à balises.

Cette approche permet de différencier deux types de compositions : les compositions automatiques des IHM (comme la fusion, l'union ou encore l'intersection de plusieurs IHM) et les compositions interactives (dirigées par le concepteur d'IHM). Le premier type de compositions s'appuie directement sur la description des arbres de regroupements de notre approche qui ne nécessite pas de connaître les dépendances entre les tâches que doit remplir l'IHM. Par contre, le second type de compositions s'appuie plus particulièrement sur la description des arbres de tâches. Lorsque le concepteur d'IHM indique que, lors de la composition, il souhaite combiner certains éléments de la première IHM avec d'autres éléments de la seconde IHM, nous allons devoir tenir compte des dépendances entre les tâches afin de garantir une composition cohérente (permettant d'obtenir une interface finale fonctionnelle).

Bibliographie

1. Popfly (microsoft). `http://www.popfly.com`.
2. Yahoo pipes (yahoo). `http://pipes.yahoo.com/pipes`.
3. Igoogle. `http://www.google.com/ig`.
4. Netvibes. `http://www.netvibes.com`.
5. Lewandowski A, Lepreux S, et Bourguin G. Tasks models merging for high-level component composition. *Human-Computer Interaction, Part I, HCII 2007, Lecture Notes in Computer Science (LNCS)*, 4550 :1129–1138, juillet 2007.
6. Calvary G, Coutaz J, Thevenin D, Limbourg Q, Bouillon L, et Vanderdonckt J. A unifying reference framework for multi-target user interfaces. *Interacting with Computers*, 15(3) :289–308, juin 2003.
7. Mori G, Paternò F, et Santoro C. Ctte : Support for developing and analyzing task models for interactive system design. *IEEE Transactions on Software Engineering*, pages 797–813, août 2002.
8. Limbourg Q, Vanderdonckt J, Michotte B, Bouillon L, Florins M, et Trevisan D. Usixml : A user interface description language for context-sensitive user interfaces. *AVI'2004 Workshop "Developing User Interfaces with XML : Advances on User Interface Description Languages" UIXML'04*, pages 55–62, mai 2004.
9. Lepreux S, Hariri A, Rouillard J, Tabary D, Tarby J-C, et Kolski Ch. Towards multimodal user interfaces composition based on usixml and mbd principles. *Lecture Notes in Computer Science*, 4552(134) :134–143, juillet 2007.