

Architectures Logicielles Appliquées à l'Interaction Personne-Système

Jérôme Lard

THALES Recherche et Technologie
Domaine de Corbeville
91404, Orsay, France
jerome.lard@thalesgroup.com

Célestin Sedogbo

THALES Recherche et Technologie
Domaine de Corbeville
91404, Orsay, France
celestin.sedogbo@thalesgroup.com

RESUME

Alors que les moyens d'interaction deviennent de plus en plus complexes et sophistiqués, l'interaction en tant que telle demande l'intégration de modalités hétérogènes telles que la voix, le geste ainsi que de nouvelles représentations graphiques. En parallèle, le nombre de terminaux va grandissant (PC, PDA, tablet PC, etc.). Une des solutions qui permet de fournir à l'utilisateur un accès plus large à des systèmes existants et qui peut augmenter le confort d'utilisation des interfaces hommes-machines, est de concevoir des systèmes interactifs intelligents qui s'adaptent dynamiquement aux différents contextes d'utilisation. Nous illustrerons ici le concept de HIC (*Human-system Interaction Container*) qui introduit un changement important dans les modes de conception des systèmes interactifs.

MOTS CLES : interaction peronne-système, architecture d'interaction, modèles de conception logicielle, intergiciel.

ABSTRACT

As today's means of interaction become more and more various and sophisticated, interaction demands the integration of heterogeneous modalities, such as voice, gesture, graphics and animation, as well as appliances, such as traditional laptop and desktop workstations, mobile phones, Personal Digital Assistants (PDA), tablet PC, etc. One solution for providing users with wider access to existing systems and for enhancing user-friendliness of existing interfaces is to design intelligent interaction systems that dynamically adapt to the interaction environment and react appropriately in various contexts of use, without implying any modification to the core application. We will illustrate here the concept of Human-computer Interaction Container (HIC) which introduces an important shift in the field of Human-Computer Interaction (HCI), moving from an application-centric to a user-centric perspective, through the adoption of a service-oriented view of application and user interface capabilities.

CATEGORIES AND SUBJECT DESCRIPTORS: H.5.2 [User Interfaces]: Miscellaneous; D.2.2 [Design Tools

and Techniques]: Modules and interfaces, Object-oriented design methods, User interfaces; H.1.2 [User/Machine Systems]: Human Information Processing; I.3.6 [Methodology and Techniques]: Device Independence, Interaction Techniques, Languages.

KEYWORDS : human computer interaction, interaction architecture, software design pattern, middleware.

INTRODUCTION

Dans la plupart des cas, les interfaces logicielles existantes ne sont qu'un prisme qui permet de consulter les données d'une application et d'agir sur celles-ci au travers de commandes prédéfinies (un logiciel de traitement de texte propose une vue formatée d'un espace de données sans aide particulière pour l'utilisateur). Or, nous devons garder présent à l'esprit qu'un système d'information est un système d'aide et d'amélioration des performances.

Un cycle sans fin

Ainsi, nous constatons que les applications sont de plus en plus puissantes et les interfaces qui les pilotent sont de plus en plus complexes. En parallèle, l'homme ne peut pas démultiplier ses capacités mentales de la même manière que le nombre de fonctions disponibles s'accroît dans une application. La fuite en avant qui consiste à fournir plus de commandes au sein d'une interface pour maîtriser la complexité d'une application n'est pas une solution viable à moyen terme. D'autre part, le fait de proposer une interface graphique générique, en espérant qu'elle conviendra à un nombre croissant d'utilisateurs ne semble pas non plus être une réponse appropriée. En effet, chacun d'entre nous possède des buts bien différents et donc des usages individuels lorsqu'il s'agit d'utiliser un système informatique. Le nombre d'utilisateurs grandissant, il est probable que les modes d'utilisation vont aller en se diversifiant. Cela contredit les concepts présentés actuellement qui cherchent à simplifier l'interface homme-machine en lui donnant un cadre interactif assez pauvre (navigation Web), pour tous types d'application Ceci n'est possible que si on réduit les processus interactifs à leur minimum.

Vers des interfaces plus intelligentes

Nous souhaitons explorer une autre voie dans la conception des systèmes interactifs qui permette au contraire de mieux servir les utilisateurs dans leurs tâches en leur offrant des interfaces permettant de manipuler des concepts de plus haut niveau. Il faut pour cela que l'interface faisant le lien entre l'homme et la machine devienne plus "intelligente" et propose un vrai dialogue à l'opérateur. L'interface doit représenter de manière appropriée les concepts manipulés par un utilisateur qui n'ont sans doute aucune relation avec le modèle de l'application sous jacente.

L'homme dans la boucle système

Il est peu probable que dans des systèmes à risque élevé (tels que ceux utilisés pour le contrôle aérien) l'homme disparaisse de la boucle de traitement des informations. Il reste un acteur important des processus de décision surtout lorsque celle-ci est critique. Tenir compte de l'homme dans la boucle système est un pré requis essentiel de nombreux systèmes d'informations professionnels. Ce type de problématique est rarement prise en compte lors de la conception d'interfaces. Il devient nécessaire de proposer un traitement de l'information qui intègre l'homme comme une composante du système au moment de sa conception. Mais cela ne résout pas complètement les problèmes soulevés ici qui sont plus de complexité dans les applications et dans les interfaces et plus d'informations à traiter. Il est important que le système prenne également des initiatives face au comportement de l'utilisateur et face aux éventuels changements de contexte.. Pour cela, nous devons modifier les modèles classiques qui définissent la séparation entre les logiques de présentation et les logiques d'application. Une nouvelle infrastructure logicielle doit pouvoir intégrer ces notions.

INTERACTION HOMME-MACHINE

Afin de pouvoir définir des services d'interaction génériques, nous avons besoin d'isoler précisément les spécificités du domaine de l'interaction. Malheureusement, la définition du domaine *HCI* est très large. Nous empruntons la définition du comité ACM ici : L'Interaction Homme-Machine est "*une discipline intéressée par le design et l'implémentation de systèmes informatiques interactifs pour l'usage humain, et par l'étude des phénomènes majeurs qui les entourent*".

Partant de cette définition, nous sommes capable de

revenir aux concepts fondamentaux de l'interaction homme-machine. Nous sommes habitués à interagir avec notre environnement par manipulation directe et physique. Lorsque l'environnement devient plus complexe, nous tentons de trouver des solutions pour surmonter nos limites physiques et naturelles. La perception de la complexité d'un tel environnement et le type de contrôle que nous voulons appliquer peut être soutenu par un système informatique. Dans des situations de plus en plus nombreuses, l'ordinateur personnel devient un intermédiaire de choix dans nos relations à l'environnement. L'utilisation d'un système informatique permet même de considérer de nouvelles opportunités d'interaction et de créer son propre champ d'actions possibles: dans un monde virtuel par exemple. L'ordinateur devient aujourd'hui un outil plébiscité lorsque nous voulons communiquer nos idées ou participer à un processus créatif.

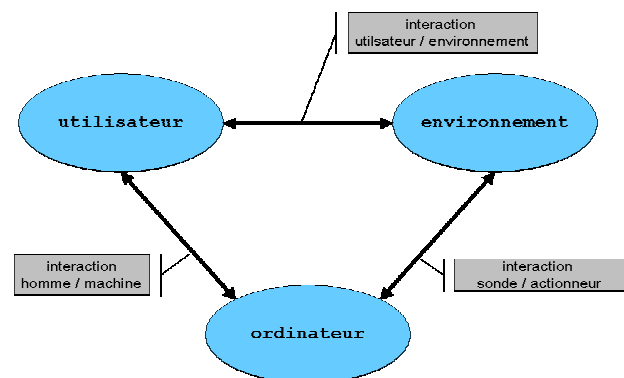


Figure 1: Principe d'action/perception

Nous retrouvons dans le schéma de la figure 1 l'action/perception directe dans ce qui constitue le lien entre utilisateur et environnement. Nous voyons que si nous introduisons un système informatique dans la communication entre ces deux entités, le couple action/perception se sépare en deux nouveaux chemins: l'interaction homme/machine et l'interaction libellée ici sonde/actionneur, signifiant que l'ordinateur perçoit l'environnement par l'intermédiaire de sondes et qu'il réagit grâce à des dispositifs mécaniques, électriques, etc., que nous avons nommés actionneurs. Le cadre de notre étude cherche donc à expliciter les principes fondamentaux qui définissent la communication entre l'utilisateur et l'ordinateur. Nous voulons ainsi en extraire les composantes les plus génériques de manière à les implémenter sous forme de services intégrés à une plateforme d'interaction. Cette plateforme couplée à un nouveau patron de conception logicielle permettra à tout concepteur d'interfaces de concevoir un système interactif, s'adaptant aux différents contextes d'utilisation et à diverses applications.

LE CONCEPT DE CONTENEUR D'INTERACTION

Ce que nous appelons conteneur d'interaction personne-système a pour but d'encapsuler tout composant logiciel dédié à la gestion ou au contrôle de l'interaction personne-système au sein d'un conteneur. Il est conçu de manière à faciliter la séparation logique entre application, interaction et présentation et manipule tous les processus interactifs permettant à une application et à ses nombreuses interfaces utilisateurs de communiquer entre elles. Ce type de découpage offre une interaction indépendante des applications d'une part et des interfaces d'autre part. Ce conteneur logiciel gère les liens entre ces composants, on parle alors de logique interne d'interaction. Celle-ci peut-être comparée pour les systèmes interactifs à la logique applicative et à la logique de présentation. D'autre part ce conteneur est sensible à la modification des contextes des utilisateurs ou du système. Dans le cas d'un changement de contexte de l'utilisateur, le conteneur d'interaction doit être capable d'en tenir compte et de répercuter cet événement au niveau de l'application cible. Dans le cas d'un changement de contexte du système, le conteneur doit être en mesure d'en informer l'utilisateur si cela est nécessaire. Si ces propriétés sont respectées, nous aurons alors réalisé un système dans lequel l'homme et la machine entretiennent un vrai dialogue. Ce conteneur agit telle une passerelle entre les services applicatifs et les services de présentation. Il implémente des services dits d'interaction qui se veulent génériques. Ils n'ont besoin que de modèles pour être configurés en accord avec un utilisateur, une tâche complexe ou encore une logique métier à respecter. Ces services d'interaction incluent par exemple le traitement du dialogue homme-machine, de la tâche et de l'activité des utilisateurs, de la gestion de la multimodalité et de la génération d'une présentation multimédia appropriée. Notre approche cherche à générer ou à ajuster de manière dynamique une présentation qui correspond le mieux aux attentes d'un utilisateur dans un contexte particulier et pour une tâche bien définie. Cette dynamique doit suivre les changements de contexte ainsi que le déroulement de la tâche d'un utilisateur. Pour autant, ce travail n'a pas pour cible la conception automatique d'interfaces homme-machine mais il s'agit plutôt d'étudier les composantes génériques du dialogue homme-machine. Ce dialogue est défini comme une gestion de la communication inter-processus entre des applications de différentes natures.

ARCHITECTURE D'INTERACTION

Nous avons considéré que la plupart des architectures logicielles implémentées à l'heure actuelle sont basées sur un paradigme en trois couches: base de données, application et présentation. Apporter plus d'intelligence à un système c'est introduire une nouvelle couche qui va se charger de la médiation entre l'utilisateur et les objets informatiques qu'il manipule. La représentation fonctionnelle de ce concept est illustré ci dessous:

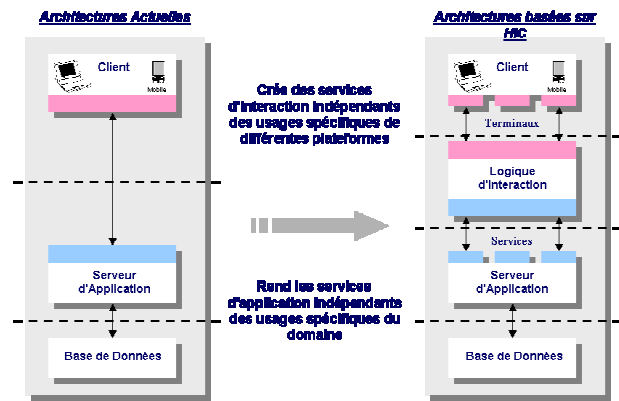


Figure 2: Architecture Fonctionnelle HIC

Nous avons choisi d'implémenter ce paradigme sous la forme d'un middleware spécifique à un domaine particulier : celui de l'interaction personne-système. Nous reviendrons plus en détail sur les spécificités de ce middleware par la suite. Nous pouvons dire que la gestion de l'interaction fournie par cette plate-forme s'applique à la *compréhension mutuelle entre des applications de natures différentes*. En conséquence notre choix est de placer cette interaction au centre des échanges entre le système et les utilisateurs. C'est la solution que nous proposons afin de répondre à la complexité croissante des systèmes et des logiciels. Un système informatique est facile à manipuler lorsqu'il offre un nombre limité de fonctionnalités. Si ces fonctionnalités sont dédiées à des tâches de haut niveau, un tel système est très puissant pour accomplir ce pour quoi il a été conçu. D'un autre côté, les systèmes qui essaient d'être le plus générique possible tendent à proposer trop de fonctions, le plus souvent d'assez bas niveau, au sens où elles sont très proches de celles du système lui-même et pas assez proches de la tâche de l'opérateur. L'architecture présentée à la figure 2 illustre l'introduction d'une nouvelle couche logicielle. Cette nouvelle couche qui contient la logique d'interaction est basée sur des modèles. Les différents modèles que nous pensons utiliser sont les suivants : modèle de tâche, profil utilisateur, modèle de l'organisation, règles métiers ou du domaine d'application.

NOTION DE MIDDLEWARE D'INTERACTION

Un middleware est selon toute vraisemblance un logiciel qui consiste en un ensemble de services permettant à plusieurs processus s'exécutant sur une ou plusieurs machines de communiquer entre eux au sein d'un réseau. Une autre définition prétend qu'un middleware est un bout de code logiciel qui agit comme une couche de conversion ou de translation entre deux processus. Cette dernière définition est plus proche de ce que nous souhaitons implémenter. En effet, en regardant la séparation fonctionnelle de la figure 2, nous nous sommes aperçu que ce que nous voulons insérer entre la logique applicative et la logique de présentation est bel et

bien une couche de conversion bien que particulière. Ce nouveau niveau doit effectuer une conversion entre la syntaxe et la sémantique d'une application et la transformer en une présentation graphique représentative de l'état du système d'une part et d'autre part est chargé d'interpréter les intentions de l'utilisateur pour l'application. Le middleware développé comporte trois niveaux : services standards, services techniques et services d'interaction. Les services standards sont ceux que l'on retrouve dans tous les middleware classiques. Les services techniques viennent supporter l'intégration de services de plus haut niveau que sont les services d'interaction. La figure 3 représente les services d'interaction de la plateforme.

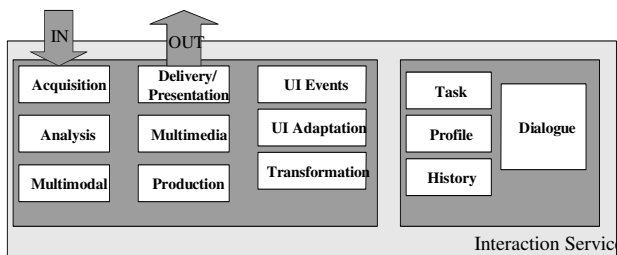


Figure 3: Couche de services d'interaction

CONCLUSION

La plateforme telle qu'elle a été présentée dans cet article a été implémentée et testée dans le contexte particulier d'une application professionnelle pour la gestion du trafic aérien. Il a été possible d'enrichir l'interaction existante avec l'application en ajoutant des fonctionnalités nouvelles sans pour autant modifier l'application de départ. Non seulement l'interaction s'est révélée plus riche mais cette expérience concrète nous a permis d'étudier la conception de différents modèles utiles à la prise en compte des contextes d'utilisation. Ainsi, nous avons défini trois modèles d'utilisateurs, un modèle de l'organisation, des modèles pour les plateformes matérielles et des règles représentatives du domaine d'activité. La plateforme d'interaction une fois implémentée se compose de trois parties distinctes : des composants d'interaction qui représentent les traitements interactifs qui ne sont pas généralisables quelque soit le contexte, le conteneur d'interaction qui assemble ces composants entre eux de manière dynamique et le middleware d'interaction qui propose des services d'interaction génériques disponibles pour toutes les applications du système. Cette architecture complète est représentée en figure 4.

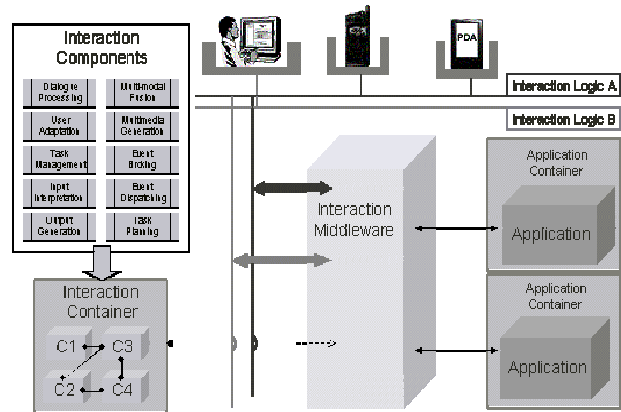


Figure 4: Architecture Fonctionnelle HIC

BIBLIOGRAPHIE

1. Brochers J., A pattern approach to interaction design.
2. Carroll J., HCI Models, Theories and Frameworks: Towards a multidisciplinary Science.
3. Geihs K., Middleware Challenges Ahead. IEEE Computer Society, 2001
4. Gulliksen J., Goransson B., Boivie I., Blomkvist S., Persson J. and Cajander A., Key Principles for User-Centred Systems Design. *Behaviour & Information Technology*, nov-dec 2003, pages 397-409
5. Kirste T., Rapp S., Architecture for Multimodal Assistance Systems. *International Status Conference, HCI*, October 2001
6. Kuniavsky M., Observing the user experience: a practitioner's guide to user research. *Morgan Kaufmann Series*.
7. Sendin M., Lorés J., Montero F. and López V., Towards a Framework to Develop Plastic User Interfaces. *Mobile HCI*, September 2003
8. Schmidt D. C., Buschmann F. , Patterns, Frameworks, and Middleware: Their Synergistic Relationships. 2003
9. Thévenin D., Thèse de Doctorat de l'Université Joseph Fournier de Grenoble, "Adaptation en Interaction Homme-Machine : le cas de la plasticité", Grenoble 2001
10. Wang Z., Garlan D., Task Driven Computing. May 2000