Multitouch & Natural User Interface : Opportunités pour une approche Bottom-Up

Laurent Salat

TACTINEO
11, rue de l'Ave Maria
75004, Paris, France
lsalat@tactineo.com

RESUME
Nous proposons d'analyser le problème de la complexité des interfaces logicielles sous l'angle de l'approche cognitive qu'elles proposent à l'utilisateur.

Nous inspirant des travaux réalisés par R. George et J. Blake sur les NUI et le modèle OCGM, nous définissons les notions d'interfaces orientées Top-Down et Bottom-Up et proposons que les NUI favorisent la mise en place d'interfaces orientées Bottom-Up notamment dans les scénarios où l'apprentissage par tutorat ne peut être assuré.

Nous définissons enfin les principes d'une architecture logicielle orientée Bottom-Up et donnons deux exemples de dispositifs orientés Bottom-Up.

MOTS CLES : Natural User Interface, Architecture logicielle, Approche Bottom-Up, Approche Top-Down.

INTRODUCTION
Les notions d'approches Top-Down et Bottom-Up caractérisent deux démarches cognitives complémentaires. Ces deux notions ont été déclinées dans de nombreux domaines tels que le management, la gestion des connaissances, les neurosciences, l'architecture et l'informatique.

L'approche Top-Down vise à passer de la connaissance générale d'un objet complexe à sa connaissance détaillée par découpages successifs en composants de plus en plus simples. L'approche Top-Down est avant tout analytique et formelle. Elle est plus particulièrement adaptée aux scénarios ou la personne maitrise déjà les concepts du domaine.

A l'inverse, l'approche Bottom-Up vise à passer de la connaissance détaillée d'objets simples à la connaissance d'objets complexes par agrégations successives. L'approche Bottom-Up est avant tout créative. Elle permet l'émergence de nouveaux résultats par la pratique.

CARACTERE TOP-DOWN DE NOS LOGICIELS
Les interfaces utilisateurs logicielles deviennent de plus en plus complexes au fur et à mesure qu'augmente le nombre des fonctionnalités offertes. Pourtant, la plupart des utilisateurs n'utilisent qu'un sous-ensemble réduit des fonctionnalités [1]. Ces interfaces « tout en un » proposent le même ensemble de menus et d'outils sans prendre en compte l'expérience de l'utilisateur ou la tache qu'il doit réaliser [2].

Des solutions pour répondre à ce problème ont déjà été proposées sous la forme d'interfaces adaptatives ou d'interfaces adaptables. Les interfaces adaptatives visent à modifier automatiquement l'interface en fonction de l'utilisation faite du logiciel. L'exemple d'une telle solution est le système de menus adaptatifs de MS Office 2000 [3]. Les interfaces adaptables permettent à l'utilisateur de modifier l'interface en fonction de ses besoins [2, 4].

Nous formulons l'hypothèse que la principale difficulté soulevée par les interfaces actuelles est qu'elles placent de facto l'utilisateur dans une approche cognitive de type Top-Down. Avant de pouvoir réaliser une tache simple, l'utilisateur doit d'abord appréhender la structure abstraite d'un outil complexe. Cette approche peut être considérée comme adaptée aux cas où un apprentissage préalable par tutorat est proposé à l'utilisateur (approche analytique). Elle est par contre insatisfaisante pour un scénario d'utilisation grand public [5] où l'apprentissage de l'utilisateur néophyte est souvent réalisé par l'action, via un processus de type essai/erreur.

NATURAL USER INTERFACE
Inspiré par les idées de B. Buxton sur les Natural User Interface (NUI) comme interfaces réutilisant les aptitudes déjà acquises par l'utilisateur [6], J. Blake propose la définition suivante des NUI : « A natural user interface is a user interface designed to reuse existing skills for interacting directly with content » [7]. J. Blake définit la diminution de la charge cognitive associée à l'apprentissage et à l'utilisation de l'interface comme un objectif important des NUI [8] .

Les applications multitouch implémentant les principes de NUI sont pour l'instant fonctionnellement simples. Nous pensons cependant que le nombre de fonctionnalités offertes grandissant, ces applications rencontreront des problèmes similaires aux applications GUI si elles placent, de manière identique, l'utilisateur dans une approche cognitive de type Top-Down.

NATURAL USER INTERFACE & APPROCHE BOTTOM-UP

OCGM

R. George et J. Blake proposent le modèle OCGM comme équivalent pour les NUI du modèle WIMP pour les Graphical User Interface. Dans le modèle OCGM, les composants fondamentaux de toute application sont les Objets, les Conteneurs, les Gestuelles et les Manipulations. Le choix de ces 4 concepts s'appuie sur les travaux de J. Piaget concernant le développement cognitif des enfants. La perception de catégories d'objets apparait à partir de 9 mois. La capacité de reconnaitre des relations de contenu/conteneur est observée à partir de l'âge de 6 mois. L'aptitude à reconnaitre et utiliser des gestuelles pour communiquer est observée entre 9 et 12 mois. La capacité de manipuler des objets est observée à partir de l'âge de 6 mois. En s'appuyant sur des aptitudes acquises très tôt dans notre vie, OCGM vise à diminuer la charge cognitive requise pour l'apprentissage et l'utilisation de l'interface [9].

La notion de jeux

P. Smith et A. Pellegrini définissent le Jeu comme une activité caractérisée par l'emphase mise sur les moyens plutôt que sur la fin (le processus est plus important que l'objectif ou le but final), par la flexibilité (les objets font partie de nouvelles combinaisons) et par l'affect positif. La notion de Jeux correspond elle a une activité plus structurée et possédant un but. La pratique du Jeu est généralement observée entre 2 et 6 ans, celle des Jeux à partir de 6 ans [10].

J. Piaget associe aux 4 stades du développement de l'enfant, 4 types de jeux :

- Stade sensori-moteur (de 0 à 2 ans) : les jeux d'exercices,
- Stade préopératoire ou intuitif (à partir de 2 ans) : les jeux symboliques,
- Stade des opérations concrètes (à partir de 6 ans) : les jeux de construction,
- Stade des opérations formelles (à partir de 11 ans) : les jeux de règles.

Durant le stade des opérations concrètes, l'enfant construit une structure intellectuelle lui permettant de manipuler des opérations mentales de façon logique à partir d'objets se trouvant dans son champ de perception. Durant le stade des opérations formelles, l'enfant met en place les schèmes définitifs qu'il utilisera tout au long de sa vie et qui lui donnent accès à la logique formelle portant non plus uniquement sur des objets réels mais sur des hypothèses. C'est durant ce stade que la pensée dépasse le réel pour l'insérer dans le possible [11].

La notion d'apprentissage

C. George met en exergue deux 2 types d'apprentissages différents : l'apprentissage par l'action résultant d'un mécanisme de rétroaction (l'analyse des résultats des actions réalisées permet l'apprentissage) et l'apprentissage par tutorat (transmission du savoir par un enseignant ou un support de tutorat) [12]. L'apprentissage spontané par l'action est généralement plus lent que l'apprentissage dirigé mais aboutit à des connaissances plus stables et généralisables [13].

Application aux Natural User Interface

Considérant les points précédents nous proposons les définitions suivantes :

Une interface utilisateur orientée Bottom-Up est une interface s'appuyant sur les aptitudes acquises durant le stade des opérations concrètes et favorisant l'apprentissage par l'action.

Une interface utilisateur orientée Top-Down est une interface s'appuyant sur les aptitudes acquises durant le stade des opérations formelles et favorisant l'apprentissage par tutorat.

Dans le prolongement des travaux de R. George et J. Blake [9], nous proposons que les NUI priorisent l'utilisation d'interfaces orientées Bottom-Up afin de favoriser l'utilisation d'aptitudes acquises tôt durant l'enfance, afin de diminuer la charge cognitive associée à l'apprentissage et à l'utilisation de l'interface et afin de diminuer l'occurrence et la durée des phases « Top-Down » (l'analyse de l'interface) entrecoupant les phases Bottom-Up (la réalisation de la tache globale).

Nous pensons d'autre part que cette approche est plus particulièrement adaptée aux cas où l'interface adresse une cible utilisateurs pour laquelle l'apprentissage par tutorat ne peut être envisagé.

PRINCIPES D'ARCHITECTURES ORIENTEES BOTTOM-UP

Nous définissons une architecture orientée Bottom-Up comme une architecture logicielle modulaire permettant à l'utilisateur de construire ses propres outils logiciels à partir de composants élémentaires et facilitant de facto la création d'interfaces orientées Bottom-Up.

Une architecture orientée Bottom-Up s'appuie sur 4 notions :

- Les espaces de travail,

- Les données,
- Les opérateurs,
- Les magasins.

Les espaces de travail

Dans une architecture orientée Bottom-Up, la notion d'application est remplacée par la notion d'espace de travail.

Un espace de travail est créé à la demande de l'utilisateur comme un espace vierge ou offrant un ensemble minimal de fonctionnalités communes à tous les espaces de travail.

Un espace de travail contient des données et des opérateurs explicitement ajoutés par l'utilisateur.

La notion d'espace de travail peut être associée à la notion de Conteneur d'OCGM.

Les données

Les données correspondent aux contenus que l'utilisateur emploie pour réaliser ses taches.

Les données sont typées.

Les données peuvent être simples (texte, image, video, …) ou composées (tableau, carte,…).

La notion de données peut être associée aux notions d'Objet ou de Conteneur d'OCGM.

Les opérateurs

Les opérateurs sont des composants offrant un nombre limité de fonctionnalités (opérations) possédant une forte cohérence fonctionnelle. Par exemple, un opérateur « Palette de police » rassemble un groupe d'opérations telles que Gras, Italique, Souligné, Barré , Taille, Couleur, …

Les opérations s'appliquent à des types de données typés.

Un opérateur doit pouvoir être utilisé dans de multiples contextes. Par exemple, l'opérateur « Palette de polices » doit pouvoir être utilisé aussi bien dans un espace de travail implémentant un éditeur de texte que dans un espace de travail implémentant un outil de présentation, un tableur, …

La notion d'opérateur peut être associée aux notions d'Objet, de Conteneur, de Manipulation ou de Gestuelle d'OCGM.

Les magasins

Les magasins sont des ressources locales ou réseau offrant à l'utilisateur d'acquérir des données ou des opérateurs au sein d'un espace de travail.

Les magasins doivent proposer à l'utilisateur une ou plusieurs méthodes pour sélectionner les opérateurs qu'il souhaite récupérer au sein d'un espace de travail (catégorisation, description, classification par niveau d'expertise, par popularité, …).

Les opérateurs offerts par un magasin s'appliquent à des types de données définis au sein du magasin. La définition d'un type de donnée peut être partagé par plusieurs magasins.

La notion de magasin peut être associée à la notion de Conteneur d'OCGM.

Bénéfices attendus de ce type d'architecture

Dans une telle architecture, la réalisation de taches simples nécessite l'apprentissage d'un nombre limité d'opérateurs explicitement choisis par l'utilisateur.

Au fur et à mesure que les taches à réaliser deviennent plus complexes, l'utilisateur enrichit son espace de travail avec de nouveaux opérateurs.

L'apprentissage des opérateurs est donc incrémental et continu selon le modèle d'approche Bottom-Up que nous avons proposé.

En fournissant non pas des applications « tout en un » mais des opérateurs réutilisables dans de multiples contextes, ce type d'architecture devrait également viser à favoriser la « créativité » et l'émergence de nouveaux schémas pour remplir une tache donnée.

EXEMPLES D'INTERFACES ORIENTEES BOTTOM-UP

La reacTable

La reacTable est un instrument de musique utilisable par un ou plusieurs musiciens permettant la synthèse de sons modulaire. La reacTable est également un dispositif implémentant une Tangible User Interface s'appuyant sur une surface translucide et un jeu d'objets (marqueurs). Chaque objet est associé à une fonction pour la génération, la modification ou le contrôle du son. En plaçant les objets sur la surface puis en modifiant la proximité ou l'orientation de ces objets, le musicien met en place la chaine de traitement sonore et agit sur les paramètres des différentes fonctions [14].

Du point de vue de son utilisation, la reacTable présente toutes les caractéristiques d'une interface orientée Bottom-Up.

Si l'architecture logicielle de la reacTable ne peut pas être qualifiée de Bottom-Up, d'un point de vue ergonomique, le dispositif est bien construit autour des 4 notions proposées :

- L'espace de travail : la table,
- Les données : les sons synthétisés,
- Les opérateurs : les marqueurs,
- Le magasin : l'endroit ou sont disposés les marqueurs non utilisés.

L'iPhone et l'AppStore

Le smartphone d'Apple (l'iPhone) et l'AppStore qui lui est associé sont un autre exemple d'interface orientée Bottom-Up. Les modules logiciels accessibles sur la

plateforme se présentent souvent sous la forme de micro-applications offrant un nombre limité de fonctionnalités et répondant à une problématique précise. Ces applications sont utilisables dans un contexte logiciel unique mais sont, de par le caractère mobile du dispositif, utilisables dans différents contextes fonctionnels.

L'architecture du dispositif répond en partie à la définition d'une architecture orientée Bottom-Up :

- L'espace de travail : l'iPhone,
- Les opérateurs : les applications de l'AppStore,
- Le magasin : l'AppStore,
- Les données : les données consommées par les applications.

La principale divergence du dispositif par rapport à l'architecture que nous avons définie réside dans l'insularité des applications au sein de l'espace de travail.

L'iPhone et l'AppStore représentent une évolution notable dans l'architecture logicielle de nos systèmes [15] et nous paraissent un premier pas important vers un modèle d'architecture orientée Bottom-Up.

CONCLUSIONS

Nous avons introduit la notion d'interfaces orientées Top-Down et Bottom-Up et dans la continuité des travaux réalisés par R. George et J. Blake sur le modèle OCGM, nous avons proposé que les NUI favorisent l'utilisation d'interfaces orientées Bottom-Up.

Certaines évolutions récentes, comme le modèle de l'AppStore d'Apple et ses micro-applications ou l'évolution des interfaces Microsoft Office vers un modèle moins abstrait, plus réifié et plus orienté vers la tâche à accomplir [16], nous paraissent être des indicateurs encourageants pour approfondir cette approche.

BIBLIOGRAPHIE

1. J. McGrenere. "Bloat": The Objective and Subject Dimensions. In Proceedings of CHI 2000..
2. J. McGrenere, R.M. Baecker, and K.S. Booth. An evaluation of a multiple interface design solution for bloated software. In Proceedings of CHI '02, pages 164–170. ACM Press, 2002.
3. J. Harris. Combating the Perception of Bloat (Why the UI, Part 3). In "An Office User Interface Blog". http://blogs.msdn.com/jensenh/archive/2006/03/31/565877.aspx.
4. W. Stuerzlinger, O. Chapuis, D. Phillips and N. Roussel. User interface façades: towards fully adaptable user interfaces. In Proceedings of UIST'06, the 19th ACM Symposium on User Interface Software and Technology, pages 309-318, October 2006. ACM.
5. A. Capobianco, N. Carbonell. Conception d'aides en ligne pour le grand public : défis et propositions. in A. Drouin, G. Eude, J.-M. Robert (Eds.), Actes du 8ème Colloque Francophone Ergonomie et Informatique Avancée (ERGO-IA'2002), Biarritz, 8-10 Octobre 2002, Bidart (64210) : ESTIA & ESTIA Innovation, pp. 309-335.).
6. B. Buxton. Interview Channel 9 au CES 2010. Janvier 2010. http://channel9.msdn.com/posts/LarryLarsen/CES-2010-NUI-with-Bill-Buxton/.
7. J. Blake. NUI reuse existing skills (updated NUI definition). Avril 2010. Blog "Deconstructing the NUI". http://nui.joshland.org/2010/04/nuis-reuse-existing-skills.html.
8. J. Blake. Multitouch on Windows. Edition Mannings. Chap.1 p.14. http://www.manning.com/blake/MEAP_Blake_ch01.pdf
9. R. George, J. Blake. Objects, Containers, Gestures, and Manipulations: Universal Foundational Metaphors of Natural User Interfaces. In Proceedings of CHI10. http://nui.joshland.org/2010/02/ocgm-universal-foundational-metaphors.html.
10. P. Smith, A. Pellegrini. Apprendre en jouant. In Encyclopédie sur le développement des jeunes enfants. http://www.enfant-encyclopedie.com/documents/Smith-PellegriniFRxp.pdf.
11. J. Piaget. L'épistomologie génétique. Editions PUF. P13-58.
12. J.L. Roulin. Psychologie cognitive. Editions Breal.
13. C. George. Comment conceptualiser l'apprentissage. Revue française de pédagogie. N°72 – Juil_Aout_Sept 1985.
14. S. Jorda, M. Kaltenbrunner, G. Geiger, R. Bencina. The Reactable. In Proceedings International Computer Music Conference 2005.
15. D. Hinchcliffe. The app store : The new "must-have" digital business model. ZDNet. http://blogs.zdnet.com/Hinchcliffe/?p=1172.
16. J. Harris. The Story of Ribbon. Présentation Mix2008. http://www.slideshare.net/guest3bbe8d/u-x09-harris?from=ss_embed.