Démonstration : HephaisTK, une boîte à outils pour le prototypage d'interfaces multimodales

Bruno Dumas, Denis Lalanne, Rolf Ingold

Université de Fribourg
Boulevard de Pérolles 90
1700, Fribourg, Suisse
[prénom.nom]@unifr.ch

RESUME
Cet article décrit HephaisTK, une boîte à outils destinée au prototypage d'interfaces multimodales. L'article présente brièvement l'état de l'art et les défis à résoudre avant de décrire l'architecture de la boîte à outils HephaisTK, ainsi que son langage de description. Finalement, l'article expose les perspectives.

MOTS CLES : Boîte à outils multimodale, interfaces multimodales, fusion multimodale.

ABSTRACT
This article describes HephaisTK, a toolkit for prototyping multimodal interfaces. The article briefly presents the state of the art and the challenges before describing the architecture of the HephaisTK toolkit, along its description language. Finally, the article explains future works.

CATEGORIES AND SUBJECT DESCRIPTORS: H.5.2 [Information Interfaces and Presentation]: User Interfaces – Input devices and strategies, Interaction styles, Prototyping.

GENERAL TERMS: Design, Human Factors.

KEYWORDS: Multimodal toolkit, multimodal interfaces, multimodal fusion, human-machine interaction.

INTRODUCTION

Les interfaces multimodales sont devenues depuis une dizaine d'années un thème majeur de l'interaction homme machine. Néanmoins, cette classe d'interfaces pose un défi à ses concepteurs de par la complexité des différents mécanismes qu'elle nécessite de mettre en œuvre. Certaines modalités d'entrée, telles que la reconnaissance de la parole ou de la gestuelle, font partie de domaines de recherches actifs. Une autre source de difficulté provient du fait que les différentes modalités peuvent potentiellement être combinées entre elles : l'aspect synchronisation entre modalités ainsi que la résolution des coréférences doivent alors être impérativement pris en compte. La création d'une boîte à outils permettant la génération d'interfaces multimodales a déjà été abordée par plusieurs auteurs, en général sous la forme d'un outil intégré gérant l'ensemble de la création de l'application. Bourguet [1] s'est ainsi attachée à créer une boîte à outils en deux composantes séparées : d'un côté MEngine, le moteur multimodal proprement dit, de l'autre IMBuilder, une interface permettant la création d'interfaces multimodales en les décrivant sous la forme de modèles à états. Une autre expérimentation intéressante fut celle de Flippo et al. [5] qui conçurent une boîte à outils basée sur un système multi-agents. Celle-ci devait s'intégrer directement au sein de l'application à laquelle elle donnait des capacités multimodales. Une approche composantes a été choisie pour la création d'une boîte à outils logiciel libre appelée OpenInterface [6] proposant des composantes pour de multiples fonctionnalités et sources d'entrée. Il est intéressant de constater que, dans les cas des boîtes à outils présentées ci-dessus, l'application cliente profitant de chaque outil doit soit être construite autour de l'outil, qui s'impose ainsi comme le cœur de l'application, soit découler directement de la boîte à outils elle-même, qui fabrique (et/ou constitue) l'essentiel du futur produit. HephaisTK, la boîte à outils proposée dans cet article, a été conçue pour pouvoir être rattachée à une application afin de gérer les entrées multimodales de celle-ci, sans pour autant que l'architecture de cette application doive être complètement remise en question : l'intégration de la boîte à outils au sein d'une application cliente s'effectue d'une manière proche de l'intégration de bibliothèques graphiques telles que Java Swing.

HEPHAISTK

HephaisTK est une boîte à outils écrite entièrement en langage Java, et basée sur un système multi-agents appelé JADE. Le choix de ce type d'architecture logicielle permet de pouvoir distribuer HephaisTK sur une série de plate-formes. L'architecture globale de la boîte à outils [3] est détaillée sur la figure 1. Toute source d'entrée est

IHM'08, 2-5 Septembre 2008, Metz, France

surveillée par un agent défini, qui se charge de recueillir les données de la source, de les convertir en un type générique si besoin est et de les transmettre au reste de la communauté logicielle. Cet agent contient également une description des capacités de la source et peut en informer tout autre agent de manière dynamique. Un agent nommé « postman » se charge de recueillir toutes les données entrantes provenant des différentes sources. Il les stocke ensuite dans une base de données, les laissant à disposition du reste de la boîte à outils. Ce « postier » permet à d'autres agents de s'inscrire à une certaine catégorie de messages, puis leur renverra automatiquement tout message de cette catégorie reçu. L'atout de cet agent « postier » est de servir de générateur unique de timestamp, évitant de devoir considérer de potentiels problèmes de désynchronisation entre les sources d'entrée.

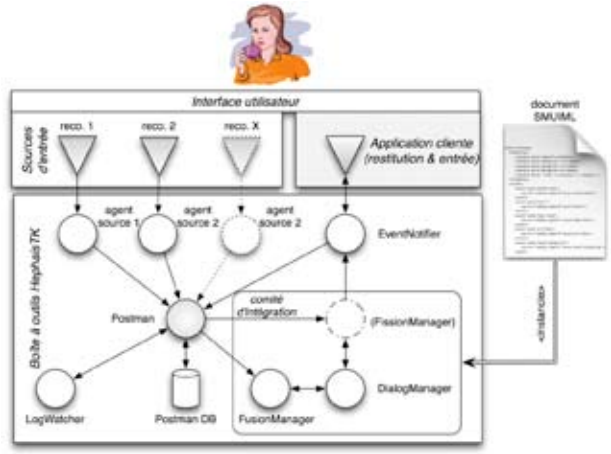

Figure 1 : architecture de HephaisTK.

La décision de signaler à l'application cliente un événement donné revient à trois agents logiciels, regroupés en un « comité d'intégration ». Un agent gère la fusion de modalités, un deuxième agent gère l'ensemble du dialogue humain/machine, alors qu'un troisième agent se charge de la fission multimodale. L'agent chargé de la fusion de modalités peut être configuré dynamiquement pour utiliser différents algorithmes. Actuellement, un algorithme basé sur les « meaning frames » est utilisé pour la fusion des différentes sources d'entrée, cependant HephaisTK a été créée dans le but de tester différents algorithmes de fusion novateurs. HephaisTK a été conçue de manière à pouvoir recevoir des données de sources d'entrée aussi diverses que des reconnaisseurs de parole, des périphériques matériels ou tables multi-utilisateurs.

CONFIGURATION

Les agents gérant le dialogue et la fusion sont paramétrés à l'aide d'un script écrit en un langage XML appelé SMUIML (*Synchronized Multimodal User Interfaces Modeling Language*) [4]. Lorsqu'une application cliente instancie HephaisTK, elle doit notamment spécifier l'emplacement du script qui lui est rattaché. Ce script SMUIML permet de définir le dialogue humain/machine multimodal sur trois niveaux différents : une description du dialogue humain/machine proprement dit, de haut niveau, en ayant la possibilité de définir des entités réutilisables ; une description de l'ensemble des événements d'entrée et de sortie qui vont potentiellement intervenir dans le dialogue décrit précédemment, selon leur modalité ; et une description bas niveau des reconnaisseurs et sources d'entrée utilisés, ainsi que leur éventuelle configuration ; à noter que plusieurs reconnaisseurs différents peuvent être disponibles pour une même modalité. Le langage de script permet la spécification précise de la synchronisation de modalités par la prise en compte explicite des propriétés CARE [2]. Le choix de se concentrer en premier lieu sur la définition d'un langage type XML permettant de décrire l'interaction multimodale a été fait d'abord afin d'avoir une base formelle permettant l'étude de l'expressivité nécessaire à la description d'une interface multimodale ; ensuite, pour obtenir un formalisme qui soit à la fois humainement compréhensible et facile à traiter pour la machine. Enfin, si pour l'instant ces fichiers de configuration SMUIML sont écrits par le développeur, leur organisation devrait permettre à terme leur génération à l'aide d'une interface graphique.

REMERCIEMENTS

HephaisTK est financé par la Fondation Hasler (http://www.haslerstiftung.ch) ainsi que par le NCCR IM2 project (http://www.im2.ch).

BIBLIOGRAPHIE

1. Bourguet, M. L. A Toolkit for Creating and Testing Multimodal Interface Designs. *Posters and Demos from UIST'02*, Paris, Oct. 2002, pp. 29-30.

2. Coutaz, J., Nigay, L., Salber, D., Blandford, A., May, J. and Young, R. Four Easy Pieces for Assessing the Usability of Multimodal Interaction: The CARE properties. *Proc. of INTERACT'95*, Lillehammer, Norway, June 1995, pp. 115-120.

3. Dumas, B., Lalanne, D., Guinard, D., Koenig, R., Ingold, R. Strengths and weaknesses of software architectures for the rapid creation of tangible and multimodal interfaces. *Proceedings of TEI'08*, Bonn, Germany, Feb. 2008, pp. 47-54, ACM Press.

4. Dumas, B., Lalanne, D., Ingold, R. Prototyping Multimodal Interfaces with SMUIML Modeling Language. *Proc. of CHI 2008 Workshop on UIDLs for Next Generation User Interfaces*, Florence (Italy), April 5-10 2008, pp. 63-66.

5. Flippo, F., Krebs, A. and Marsic, I. A Framework for Rapid Development of Multimodal Interfaces. *Proc. Of ICMI'03*, Vancouver, BC, Nov. 5-7, 2003, pp. 109-116.

6. Serrano, M., Nigay, L., Lawson, J.-Y.L., Ramsay, A., Murray-Smith, R., Denef. S. The OpenInterface framework: a tool for multimodal interaction. In *Adjunct Proceedings of CHI'2008* (April 5-10, Florence, Italy), ACM Press, pp. 3501-3506.