

# Une Architecture en Cascade pour des Systèmes Interactifs Multi-Dispositifs

Pierre Dragicevic

ÉCOLE DES MINES DE NANTES  
4, rue Alfred Kastler  
44307 Nantes Cedex 3  
Pierre.Dragicevic@emn.fr

## RÉSUMÉ

Aujourd'hui encore, rares sont les applications interactives capables d'utiliser plus d'une souris et d'un clavier. Pour profiter des richesses offertes par les dispositifs d'entrée et les techniques d'interaction non standards, il est nécessaire de disposer d'une boîte à outils où la gestion des entrées soit extrêmement flexible. Dans cet article, nous décrivons les principes d'une architecture basée sur le paradigme de *cascade de dispositifs*, permettant au concepteur de tester un grand nombre de méthodes d'entrée potentielles, et à l'utilisateur expérimenté de reconfigurer ces méthodes pour les adapter à ses besoins. Cette architecture constitue une des bases d'une boîte à outils en cours de développement.

**MOTS-CLÉS** : Architecture logicielle, dispositifs d'entrée, techniques d'interaction, boîte à outils.

## INTRODUCTION

L'utilisation conjointe d'un clavier, d'une souris, et d'un nombre limité de techniques d'interaction WIMP constitue depuis des années le procédé universel pour interagir avec l'ordinateur. Depuis, de nombreux travaux de recherche ont démontré que des méthodes alternatives pouvaient rendre l'interaction plus naturelle, plus efficace, ou encore plus adaptée aux compétences ou aux limitations de l'utilisateur [1]. Cependant, très peu d'applications commerciales utilisent ces méthodes d'interaction novatrices [5]. En effet, les boîtes à outils graphiques, et *a fortiori* les applications interactives qu'elles servent à construire, demeurent câblées pour une utilisation exclusive et stéréotypée du clavier et de la souris [2].

Notre objectif est de fournir des abstractions, des mécanismes et des outils permettant de développer des applications interactives auxquelles on puisse adapter un grand nombre méthodes de contrôle et de saisie. Notre travail s'est orienté vers une architecture de type *data flow*. Une boîte à outils basée sur cette architecture est en cours d'implémentation, ce qui nous a permis de valider certains de ses concepts sur des prototypes. Dans ce papier, nous n'en présenterons que les grands lignes. Celle-ci est décrite de façon plus détaillée dans [3], qui comprend également des références à d'autres travaux.

## L'ARCHITECTURE

### Présentation générale

Notre architecture est basée sur le paradigme de *cascade de dispositifs*, proche de celui du *data flow*: dans les applications interactives, l'interprétation des actions de l'utilisateur se fait en général à travers une série de traitements successifs, effectués par des boîtes noires que l'on peut identifier comme étant autant de dispositifs virtuels, mais que nous nommerons plus généralement *dispositifs*.

Par exemple, à chaque fois qu'une touche est appuyée, elle passe de l'état *faux* à l'état *vrai*. Ces événements sont transformés par le clavier en *codes touches*, qui à leur tour sont convertis en *symboles* dont la combinaison est interprétée comme une *chaîne de caractères* dans l'éditeur de texte. Certains codes touches peuvent également servir à des tâches de contrôle (fig. 1).

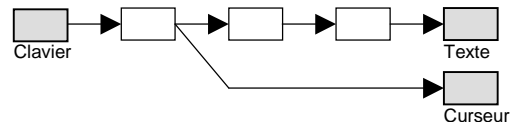


Figure 1: Un exemple de cascade de dispositifs entre le clavier et un éditeur de texte.

### Les dispositifs

Un *dispositif* est une boîte noire qui possède des *canaux d'entrée* et des *canaux de sortie* typés. Il peut effectuer des traitements internes et générer du feedback. Une *connexion* est un arc reliant un canal de sortie d'un dispositif à un canal d'entrée de même type. On peut classer les dispositifs en trois catégories, selon leur provenance :

- **Dispositifs système**: Ce sont des dispositifs partagés, comprenant l'ensemble des dispositifs d'entrée. Leur disponibilité peut varier d'un environnement à l'autre.
- **Dispositifs de la bibliothèque**: Ce sont des dispositifs qui servent d'intermédiaire entre les dispositifs système et les dispositifs d'application. Exemples : contrôle et traitement de données, opérateurs, dispositifs virtuels, adaptateurs, émulateurs.
- **Dispositifs d'application**: Ce sont des points d'entrée vers une application.

## Les configurations d'entrée

Une *configuration d'entrée* est un ensemble de dispositifs et de connexions reliant les dispositifs système (d'entrée) à ceux de l'application. Il existe un grand nombre de configurations d'entrée potentielles pour une application et un ensemble de dispositifs d'entrée donnés. Dans l'exemple de la figure 2, un éditeur de texte expose trois dispositifs correspondant aux interacteurs de Foley [4]. Une configuration synthétique y illustre différentes manières de connecter les dispositifs d'entrée au moyen de techniques d'interaction fournies par la bibliothèque.

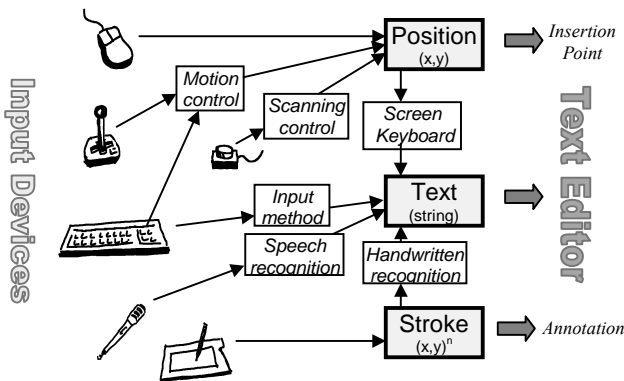


Figure 2: Différentes configurations d'entrée entre un ensemble de dispositifs (souris, manette, clavier, micro, tablette et bouton simple) et un éditeur de texte.

## DISCUSSION

Nous avons décrit les principes d'une architecture permettant de connecter des dispositifs d'entrée à des systèmes interactifs. Un système basé sur une telle architecture, Input Configurator (ICON), est en cours de développement et permet d'ores et déjà de tester cette approche sur des exemples [3]. Ce système écrit en Java™ se greffe sur des APIs d'entrée telles que *Direct Input* et repose sur la sémantique réactive d'Esterel pour l'exécution. Il comprend un jeu de techniques d'interaction et de dispositifs d'entrée, ainsi qu'un éditeur visuel (figure 3).

Un tel système possède de bonnes propriétés pour le développeur : celui-ci peut tester et valider un grand nombre de configurations en fonction des dispositifs disponibles, et rendre ainsi son application plus ouverte aux différentes modalités d'entrée. Ce système peut se greffer sur une boîte à outils graphique existante, et il est possible de rendre une application existante partiellement reconfigurable en en exposant une partie sous forme de dispositifs. L'éditeur visuel permet également à l'utilisateur avancé de reconfigurer les mécanismes d'interaction afin de les adapter à ses besoins.

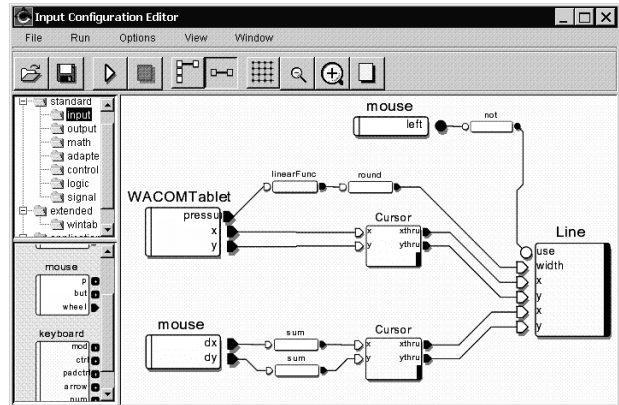


Figure 3: L'éditeur visuel ICON, où est décrite une technique bimanuelle de tracé de lignes pour l'application ICONDraw.

L'architecture en cascade est particulièrement bien adaptée à l'utilisation parallèle et non contextuelle de plusieurs dispositifs. Les applications à contrôle direct (modélisation 3D, illustration graphique, jeux vidéo) gagnent à reposer sur cette architecture. Bien que nous n'ayons pas fait d'étude à ce sujet, nous pensons que les interfaces de ce type peuvent être reconfigurées au moins en partie par des utilisateurs moyens.

Le défi actuel réside dans l'intégration des techniques d'interaction WIMP, celles-ci comportant des mécanismes d'aiguillage complexes et difficilement généralisables, ainsi que des objets transitoires. Une partie du problème a déjà été abordé [2], et un modèle générique permettant de connecter des dispositifs à des widgets est en cours d'étude.

Lorsque le système ICON sera finalisé, nous le testerons sur des applications réelles, et effectuerons des évaluations sur des utilisateurs.

## BIBLIOGRAPHIE

1. Baecker, R.M., Grudin, J., Buxton, W. and GreenberCole, S. (Eds). *Readings in Human Computer Interaction : Toward the Year 2000*. Morgan Kaufmann Publishers, 1995.
2. Dragicevic, P. et Fekete, J.-D. Étude d'une Boîte à Outils Multi-Dispositifs. *Actes des 11èmes journées francophones d'Interaction Homme-Machine IHM 99*, pages 55-62. Cépaduès, 1999.
3. Dragicevic, P. et Fekete, J.-D. Input Devices Selection and Interaction Configuration with ICON. *Actes de la conférence IHM-HCI 2001*.
4. Foley, J.D., van Dam, A., Feiner S.K., Hugues J.F. *Computer Graphics Principles and Practice*. Addison-Wesley, 1990.
5. Jacob, R.J.K. Input Devices and Techniques. *The Computer Science and Engineering Handbook*, ed. by Allen B. Tucker, pp. 1494-1511. CRC Press, 1996.