HAL
open science

Gestion des fenêtres : enregistrement et visualisation de l'interaction

Olivier Chapuis

To cite this version:

Olivier Chapuis. Gestion des fenêtres : enregistrement et visualisation de l'interaction. IHM '05: Proceedings of the 17th international conference on Francophone sur l'Interaction Homme-Machine, Sep 2005, Toulouse, France. pp.255-258, 10.1145/1148550.1148590 . hal-00534168

HAL Id: hal-00534168

https://hal.science/hal-00534168v1

Submitted on 9 Nov 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Gestion des fenêtres : enregistrement et visualisation de l'interaction

Olivier Chapuis

Projet In Situ (CNRS, Université Paris-Sud & INRIA Futurs)
LRI, Bâtiment 490, Université Paris-Sud
91405 Orsay Cedex, France
chapuis@lri.fr

RESUME

Les nouveaux modèles de rendu graphique de nos ordinateurs de bureaux permettent d'envisager de nouvelles techniques de gestion de fenêtres plus efficace. On doit donc mieux comprendre comment les utilisateurs manipulent leurs fenêtres et avoir des moyens pour évaluer ces nouvelles techniques. Nous présentons un système d'enregistrement et de visualisation de l'interaction Homme-Fenêtre pour X Window. Les enregistrements permettent de rejouer une session (sans le contenu des fenêtres) à la manière d'une vidéo. Un système de filtrage permet de sélectionner et d'accéder rapidement à des actions de haut niveau. Nous esquissons quelques exemples d'utilisations.

MOTS CLES : Géstion des fenêtres, enregistrement de l'interaction, visualsation de l'interaction.

ABSTRACT

The recent graphical rendering models of desktop computers can be used to explore new window management techniques. In order to evaluate these new techniques, we should have tools that help us better understand how users manipulate their windows. We present a log and visualization tool of the Human-Window interaction for the X Window system. The tool allows to replay a session (without the windows contents) as a video. A filtering system allows to select and easily access high level actions. We give some examples of application of this system.

KEYWORDS: Window management, UI logs, visualisation of the interaction

CATEGORIES AND SUBJECT DESCRIPTORS: H.5.2 [User Interfaces] : Evaluation/methodology, Windowing systems. D.4.9 [Systems Programs and Utilities] : Window managers.

GENERAL TERMS: Experimentation, Human Factors.

INTRODUCTION

La puissance de nos ordinateurs de bureaux ainsi que l'apparition de nouvelles utilisations conduisent à avoir de plus en plus de fenêtres ouvertes simultanément tout en ayant besoin d'interagir entre elles. Ceci conduit à des problèmes sur lesquels la communauté IHM s'est penchée depuis déjà longtemps avec par exemples les "rooms" [5]. La possibilité d'utiliser de plus grands écrans ou plusieurs écrans a récemment motivé des recherches sur la gestion des fenêtres : [7, 11]. De nouvelles manipulations de fenêtres ont aussi été récemment envisagées [1, 3, 8, 13, 2], certaines pour optimiser l'espace, d'autres pour améliorer les interactions entre fenêtres comme le glisser-déposer.

Les possibilités graphiques offertes par nos ordinateurs et par les systèmes de fenêtrage qui les accompagnent (Mac OS X, la future version de Windows et X Window avec ses nouvelles extensions XComposite et XDamage) permettent de penser que certaines de ces nouvelles techniques de gestion de fenêtres vont se généraliser à l'instar d'Exposé de Mac OS X. Dans cette perspective, il semble urgent de mieux comprendre comment chacun manipule ses fenêtres et d'avoir des moyens pour évaluer ces nouvelles techniques.

Nous présentons dans cet article un système d'enregistrement et de visualisation de l'interaction Homme-Fenêtre pour X Window. Les enregistrements obtenus sont d'assez haut niveau grâce à une nouvelle spécification récemment adoptée par les gestionnaires de fenêtres X. Ces enregistrements permettent d'obtenir des données statistiques. De plus ces enregistrements sont suffisemment précis pour pouvoir rejouer une session X (sans le contenu des fenêtres) à la manière d'une vidéo. Un tel lecteur vidéo est décrit dans cet article. Ce lecteur à des capacités de filtrage à la DIVA [9] pour sélectionner des évènements de haut niveau (comme les déplacements interactifs) et ainsi avoir un accès direct à ces évènements. Ainsi, outre des données de nature quantitative (statistique) sur la gestion des fenêtres nous obtenons des informations de nature "qualitative".

GESTION DES FENÊTRES

Une particularité du système X Window est que le gestionnaire de fenêtres est une application (presque) comme les autres. Il en existe un grand nombre, mais il y a peu de différences fondamentales entre eux ni avec les gestionnaires de Mac OS X et Windows. Le modèle des fenêtres en super-

position et la métaphore du bureau sont toujours en vigueur (voir [14] pour un état de l'art sur le sujet). Nous décrivons succinctement dans cette section le rôle du gestionnaire de fenêtres.

Le gestionnaire de fenêtres capture les fenêtres des applications à leur création, installe éventuellement des décorations (barre de titre, boutons et bords) et les positionne sur l'espace de travail. Le gestionnaire de fenêtres est en charge d'activer les fenêtres (i.e., leurs donner l'entrée clavier) et de les positionner dans l'ordre de superposition. Ces opérations peuvent être contrôlées de diverses manières par l'utilisateur. Des outils externes ou internes au gestionnaire peuvent être utilisées comme une barre de tache, un "pager" (voir ci-dessous) ou un menu à la "Alt-Tab" pour un accès direct à une fenêtre. Grâce à la décoration des fenêtres ou à des menus par exemple, l'utilisateur peut accéder aux opérations et manipulations du gestionnaire de fenêtres comme l'icônification et le déplacement.

La plupart des gestionnaires de fenêtres X Window possèdent un espace de travail virtuel. Le modèle le plus commun est une suite de bureaux virtuels indépendants entre eux de la taille de l'écran. Certains gestionnaires de fenêtres proposent de *grands* bureaux virtuels : une matrice $n \times m$ consitutée de *pages* de la taille de l'écran. La navigation dans l'espace de travail virtuel peut revêtir differentes formes. L'un des outils utilisé est un gestionnaire d'espace virtuel ou pour être plus concis un *pager*. Cet outil reproduit une vue miniature de tout l'espace de travail comme dans la Figure 1.

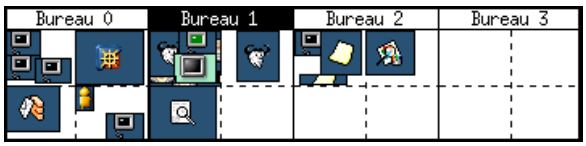

FIG. 1: Un pager qui represente quatre grands bureaux virtuels (2x2).

Certaines fenêtres ont un statut particulier vis à vis de la gestion des fenêtres. Les "panels", "docks" ou barres de tâches sont des outils aujourd'hui essentiels aux utilisateurs pour gérer leurs fenêtres. Le gestionnaire de fichiers en fond d'écran (i.e., le bureau) a aussi un statut particulier, mais de notre point de vue il s'agit plus d'un outil de gestion des documents que d'un outil de gestion des fenêtres.

L'ENREGISTREUR : WMTRACE

Un système d'enregistrement des évènements de gestion de fenêtres pour Windows est présenté dans [6]. Ce système à été utilisé pour produire des données statistiques pour comparer des configurations mono-écran à des configurations multi-écrans. D'autres systèmes plus ambitieux d'enregistrement de l'activité de l'utilisateur sont envisagés dans [4] et [12]. Ces systèmes s'intéressent en fait de manière marginale à la gestion des fenêtres et à leurs manipulations. Nous avons développé un outils *wmtrace* specifiquement dédié à la capture des manipulations de fenêtres.

wmtrace est un programme léger écrit en C qui s'exécute en tache de fond avec une session X Window. Il enregistre l'activité de gestion des fenêtres ainsi que les actions de l'utilisateur (clavier et souris). Cet enregistrement est stocké dans un fichier sous la forme de flux temporels d'évènements (un flux par fenêtre) pour être analysés par la suite. *wmtrace* utilise la Xlib et son système d'évènements, l'ICCCM (Inter-Client Communication Conventions Manual), la spécification EWMH[1] (Extended Window Manager Hints) et les extensions X11 XRecord et Xinerama

L'ICCCM fixe les règles et les mécanismes que doivent respecter les applications X pour communiquer entre elles et avec le gestionnaire de fenêtres. En particulier les applications doivent fournir le titre et la classe de leurs fenêtres et pour les fenêtres transitoires, l'identité de leur fenêtre maître. Le gestionnaire de fenêtres doit indiquer l'état des fenêtres : normal ou iconique. L'ICCCM fixe aussi les règles régissant les opérations de copier-coller entre applications.

L'EWMH est une nouvelle spécification qui étend l'ICCCM. Cette spécification est respectée par la plupart des gestionnaires de fenêtres (en particulier par ceux de GNOME et KDE). L'idée est de pouvoir modulariser la gestion d'un bureau X Window. Avec cette spécification il est possible d'implémenter une barre de tâches, un pager ou une application de bureau indépendamment du gestionnaire de fenêtres. Une conséquence est que l'on peut obtenir toutes sortes d'informations sur l'état des fenêtres et l'espace de travail.

Le gestionnaire de fenêtres doit publier (entre autres) les propriétés globales suivantes : (1) le nombre de bureaux virtuels ainsi que leurs géométries ; (2) le bureau virtuel actif ainsi que la position du point de vue dans ce bureau ; (3) la liste des fenêtres sous deux formes, l'une doit respecter l'ordre d'apparition, l'autre l'ordre de superposition ; (4) la fenêtre active.

Le gestionnaire de fenêtres doit publier les informations suivantes sur une fenêtre : (1) l'état de la fenêtre, qui peut être une combinaison des états suivants : iconifié, maximisé, enroulé, plein écran, modale, caché pour les barres de tâches ou les pagers, toujours visible sur son bureau virtuel, et éventuellement une priorité pour l'ordre de superposition ; (2) le bureau virtuel courant de la fenêtre ou bien sa visiblilité sur tous les bureaux ; (3) les dimensions de la décoration de la fenêtre.

Une application qui respecte l'EMWH publie les informations suivantes sur ces fenêtres : (1) le type de la fenêtre, qui peut être "desktop" (i.e., le bureau), "dock" (par exemple pour une barre de tache), "toolbar" pour une barre d'outils détaché en fenêtre, "menu" pour un menu détachée (pas un menu popup), "splash", "dialog" et "normal" ; (2) si nécessaire une région du bord de l'écran qui ne doit pas être couverte pour assurer le bon fonctionnement de la fenêtre ; (3) si nécessaire une indication pour signifier que la fenêtre se charge de représenter les fenêtres par des boutons (e.g., une barre de tâches pourra utiliser cette indication). Une telle application doit alors publier la géométrie de ces boutons.

Toutes les propriétés ci-dessus spécifiées par l'EMWH et l'ICCCM sont inscrites sur les fenêtres via des propriétés X

[1] http ://freedesktop.org/wm-spec/

(la fenêtre racine est utilisée pour les propriétés globales comme la liste des fenêtres). La Xlib permet de récupérer ces propriétés et toute modification de l'une d'entre elles génère un évènement. Ainsi *wmtrace* peut enregistrer de manière dynamique ces propriétés et construire un flux temporel d'évènements pour chaque fenêtre. La dimension et la position des fenêtres sont obtenues et tracées grâce à la Xlib. Les évènements de création et de destruction de fenêtres qui ne sont pas contrôlés par le gestionnaire de fenêtres (les fenêtres "override redirect", comme un menu popup) doivent être traités de manière séparée. L'extension Xinerama permet de savoir si plusieurs écrans sont utilisés. Si c'est le cas elle permet d'obtenir les dimensions de ces écrans ainsi que leurs positions relatives.

Le systeme d'évènement de la Xlib ne permet pas à une application (ici *wmtrace*) de récuperer *tous* les évènements souris et clavier. Nous utilisons l'extension XRecord pour pouvoir le faire. Chaque évènement souris et clavier ainsi capturé est ajouté au flux de la fenêtre concernée par l'évènement. Nous utilisons aussi XRecord pour capturer les opérations de copier-coller et de glisser-déposer entre applications.

Une attention particulière est prise pour préserver la vie privée. Par exemple, les titres des fenêtres ne sont pas enregistrés car elles peuvent indiquer le sujet d'un courriel ou bien les pages WWW visitées. Seule la classe des fenêtres est enregistrée (le nom de l'exécutable le plus souvent). De la même manière nous enregistrons l'activité clavier sans la valeur des touches.

La grande force de *wmtrace* est qu'il fonctionne avec la plupart des gestionnaires de fenêtres. Ceci peut être un désavantage si l'on a besoin d'informations particulières. Ainsi, l'une des motivations de *wmtrace* est de pouvoir évaluer les nouvelles techniques de gestion de fenêtres implémentées dans Metisse [2]. Lorsque *wmtrace* est executé par le gestionnaire de fenêtres de Metisse, il peut recevoir des informations directement de la part de celui-ci. On enregistre ainsi l'utilisation de ces nouvelles techniques comme le "peeling" [1] ou la rotation des fenêtres.

Par ailleurs, nous avons commencé à utiliser l'interface d'accessibilité de GTK+ pour pouvoir récupérer l'arbre des widgets de certaines applications. Il serait particulièrement intéressant, du point de vue de la gestion des fenêtres, d'avoir des informations sur les onglets, car ceux-ci sont de plus en plus utilisés et permettent de réduire le nombre de fenêtres.

LE VISULISATEUR : WMTRACE-VIEW

Les fichiers d'enregistrements produits par *wmtrace* sont suffisamment précis pour pouvoir rejouer la session X sans le contenu des fenêtres. *wmtrace-view* (Fig. 2) est un programme écrit en JAVA chargé d'extraire des données statistiques et surtout de visualiser de manière efficace une session X enregistrée par *wmtrace*. *wmtrace-view* comporte un "lecteur vidéo". Les fenêtres sont représentées par des rectangles semi-transparent avec une décoration opaque. Ceci permet de voir toutes les fenêtres et d'avoir une idée de l'ordre de superposition. Des couleurs sont utilisées pour différencier l'état des fenêtres : la fenêtre active est représentée en rouge alors que les autres fenêtres normales le sont en bleu. Les fenêtres "override redirect" sont représentées en turquoise. Une fenêtre iconifiée est représentée par un simple rectangle totalement transparent. Des petits rectangles de couleurs au centre des fenêtres indiquent des propriétés particulières. Par exemple, un rectangle vert indique une fenêtre visible sur tous les bureaux virtuels. Tous les bureaux virtuels sont aussi représentés. Le curseur est représenté par un petit carré qui se noircit totalement lorsqu'un bouton de la souris est enfoncé.

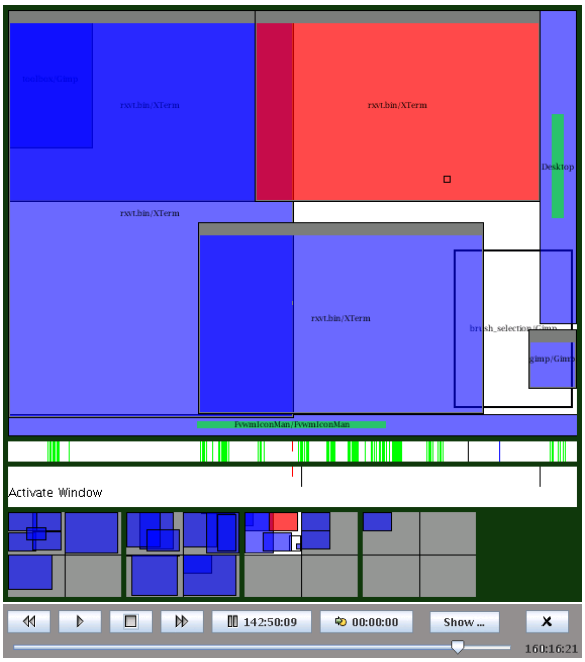

FIG. 2: De haut en bas, le bureau actif, une barre qui represent dix secondes du flux d'évènement souris et clavier, une barre qui représente les évènements sélectionnés (ici "Activate Window"), une representation des bureaux virtuels, les contrôles de la vidéo.

La fenêtre de visualisation vidéo comporte aussi deux barres pour représenter des évènements autour du temps courant. Une barre représente les évènements souris et clavier avec des traits verticaux de couleur qui dépendent de l'évènement. Une autre barre représente des évènements qui peuvent être sélectionnés grâce à des boites de dialogues. On peut par exemple choisir d'observer les opérations de déplacement interactif de fenêtres normales (ni transitoire, ni override rediect). Pour ce faire, on sélectionne l'évènement de déplacement et grâce à une boite de dialogue pertinente pour ce type d'évènements on impose qu'un bouton soit pressé sur la fenêtre en déplacement et que cette fenêtre soit normale. Un clic sur la barre d'évènements positionne alors le flux vidéo cinq secondes avant le prochain déplacement interactif.

EXEMPLES D'UTILISATIONS

Nous conduisons à l'heure actuelle une expérience pilote de type étude de terrain. *wmtrace* est utilisé sur l'ordinateur de travail de sept personnes.

Avec l'un des utilisateurs que nous appellerons Paul, nous avons réalisé une expérience de "configuration participative". Paul utilise un gestionnaire de fenêtres extrêmement confi-

gurable que nous connaissons bien (FVWM). Nous avons visualisé les enregistrements de Paul et observé certains comportements récurrents. Nous avons alors montré à Paul des parties bien choisies de l'enregistrement ce qui lui a permis d'améliorer la configuration de son gestionnaire de fenêtres. A titre d'exemple nous avons observé que Paul déplaçait quasi systématiquement ses nouvelles fenêtres pour les détacher légèrement du bord de l'écran (une quinzaine de pixels) et n'utilisait quasiment jamais l'opération de maximisation. Nous avons alors modifié l'espace de placement et de maximisation des fenêtres pour respecter cet espace d'une quinzaine de pixels. Nous avons pu constater la réduction significative du nombre de déplacement de fenêtres à leur création et une utilisation plus fréquente de l'opération de maximisation (probablement à la place d'un redimensionnement).

Nous avons commencé à observer de manière quasi systématique les opérations de déplacement et de redimensionnement interactif. On aurait pu s'attendre à ce que le déplacement soit utilisé principalement pour permettre une interaction entre deux fenêtres ou bien pour un meilleur arrangement global des fenêtres. Mais nous avons noté que pour la majorité des utilisateurs, la raison principale pour déplacer une fenêtre normale est de la recentrer sur l'écran et ceci souvent au détriment d'une occupation de l'espace qui minimiserait la superposition. Ceci suggère une opération de recentrage réversible. À propos du redimensionnement, nous avons remarqué qu'il est très rare qu'un utilisateur agrandisse une fenêtre tout en modifiant la position de son coin supérieur gauche (en utilisant donc le bord gauche ou supérieur), quitte à déplacer d'abord la fenêtre pour fixer la position de son coin supérieure gauche puis l'agrandir en utilisant son bord droit ou inférieur. Ceci peut justifier l'utilisation de fenêtres à la Mac OS X, sans bord et avec une poignée en bas à droite pour le redimensionnement.

L'une des lois les plus robustes de l'IHM est la loi de Fitts [10]. Cette loi régit le temps de mouvement pour le pointage d'une cible. Nous envisageons d'utiliser nos enregistrements pour évaluer sa validité *in situ* pour des pointages sur les bords et la barre de titre des fenêtres. On peut penser en effet que le contexte puisse affecter la performence. Par exemple, il est possible qu'un pointage sur le bord d'une fenêtre soit réalisé avec des temps de mouvement assez différents suivant qu'il existe ou non une autre fenêtre sous le bord à pointer, car le risque en cas d'erreur de mettre en avant la fenêtre inférieure est important.

CONCLUSION

Nous avons présenté un système d'enregistrement et de visualisation des fenêtres d'une session X Window. Les techniques employées pour obtenir les informations nécessaires pour rejouer les activités de gestion de fenêtres ont été décrites. Nous avons décrit un visualisteur qui permet d'observer l'interaction de l'utilisateur avec ses fenêtres. Des exemples d'utilisations ont été esquissés allant de la conception participative à une étude en situation de phénomènes en générales étudiés de manière abstraite. Nous prévoyons d'utiliser ces outils pour mieux évaluer de façons systématique les interfaces de manipulation de fenêtres.

REMERCIMENTS

Merci à M. Langet pour sa participation au développement de *wmtrace*. Merci aux membres du projets In Situ pour de nombreuses discussions fructueuses.

BIBLIOGRAPHIE

1. M. Beaudouin-Lafon. Novel interaction techniques for overlapping windows. In *Proc. of UIST 2001*, pages 153–154. ACM Press, November 2001.
2. O. Chapuis and N. Roussel. Metisse is not a 3D Desktop ! Technical Report 1407, LRI, Université Paris-Sud, France, April 2005.
3. P. Dragicevic. Combining crossing-based and paper-based interaction paradigms for dragging and dropping between overlapping windows. In *Proc. of UIST 2004*, pages 193–196. ACM Press, 2004.
4. A. N. Dragunov, T. G. Dietterich, K. Johnsrude, M. McLaughlin, L. Li, and J. L. Herlocker. Tasktracer : a desktop environment to support multi-tasking knowledge workers. In *Proc. of IUI 2005*, pages 75–82. ACM Press, 2005.
5. D. A. Henderson and S. Card. Rooms : the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. *ACM Trans. Graph.*, 5(3) :211–243, 1986.
6. D. Hutchings, G. Smith, B. Meyers, M. Czerwinski, and G. Robertson. Display space usage and window management operation comparisons between single monitor and multiple monitor users. In *Proc. of AVI 2004*, pages 32–39. ACM Press, 2004.
7. D. Hutchings and J. Stasko. Revisiting Display Space Management : Understanding Current Practice to Inform Next-generation Design. In *Proc. of GI 2004*, pages 127–134, June 2004.
8. D. Hutchings and J. Stasko. Shrinking window operations for expanding display space. In *In Proc. of AVI 2004*, pages 350–353. ACM Press, 2004.
9. W. E. Mackay and M. Beaudouin-Lafon. Diva : exploratory data analysis with multimedia streams. In *Proc. of CHI 1998*, pages 416–423. ACM Press, 1998.
10. I.S. MacKenzie. Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, 7 :91–139, 1992.
11. G. Robertson, E. Horvitz, M. Czerwinski, P. Baudisch, D. Hutchings, B. Myers, D. Robbins, and G. Smith. Scalable Fabric : Flexible Task Management. In *Proc. of AVI 2004*, pages 85–89, May 2004.
12. N. Roussel, J.D. Fekete, and M. Langet. Vers l'utilisation de la mémoire épisodique pour la gestion de données familière. soumis à IHM 2005.
13. D. Tan, B. Meyers, and M. Czerwinski. Wincuts : manipulating arbitrary window regions for more effective use of screen space. In *Extended abstracts of CHI 2004*, pages 1525–1528. ACM Press, 2004.
14. M. Tomitsch. Trends and Evolution of Window Interfaces. Diploma thesis, University of Technology, Vienna, December 2003. 132 pages.