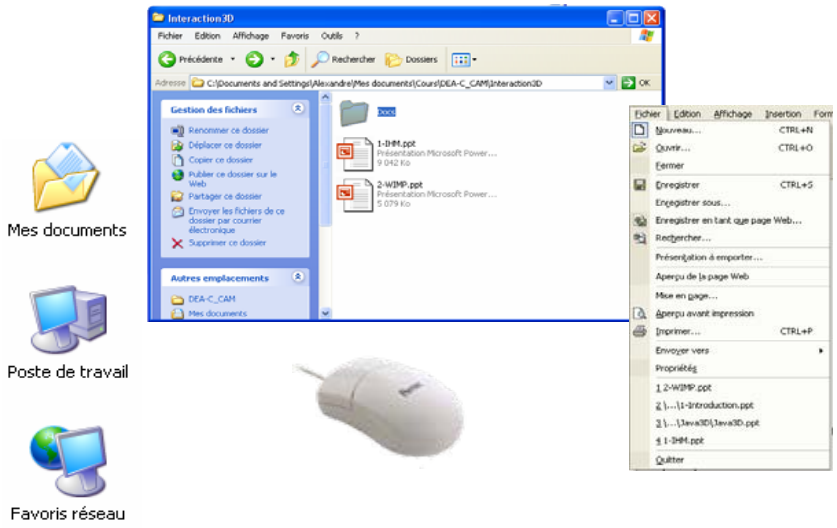
Les fondamentaux de l'IHM et du multimédia

L'Interaction Homme Machine



UE NSY116
Multimédia et Interaction Homme-Machine
2005-2006

Alexandre Topol

Quelques définitions

- Système interactif

  Un *système interactif* est un système dont le fonctionnement dépend d'informations fournies par un environnement externe qu'il ne contrôle pas
  [Wegner, 1997]

  Les systèmes interactifs sont également appelés *ouverts*, par opposition aux systèmes *fermés* – ou *autonomes* – dont le fonctionnement peut être entièrement décrit par des algorithmes

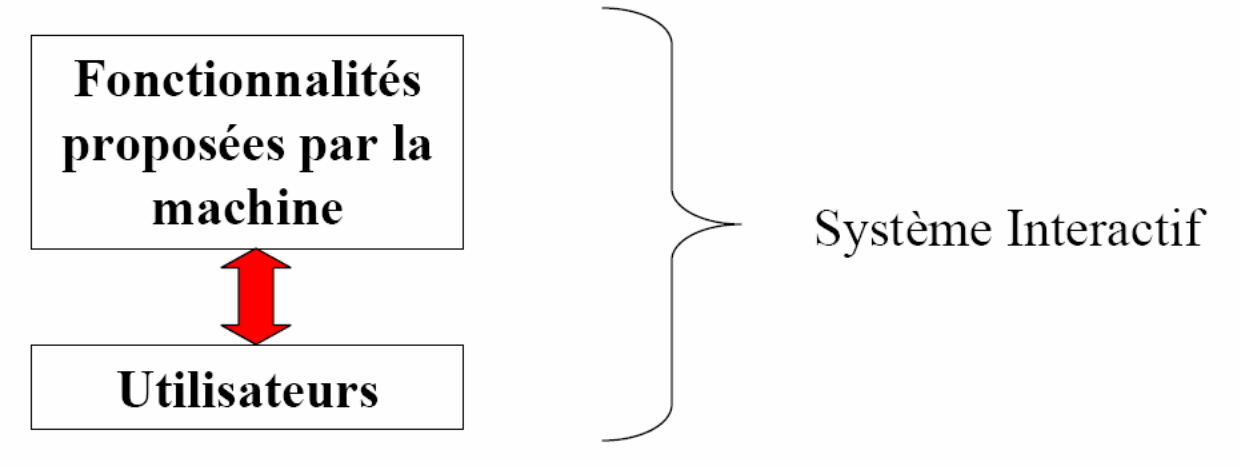
Quelques définitions

- Interface

  L'interface est l'ensemble des dispositifs matériels et logiciels qui permettent à un utilisateur de commander, contrôler, superviser un système interactif

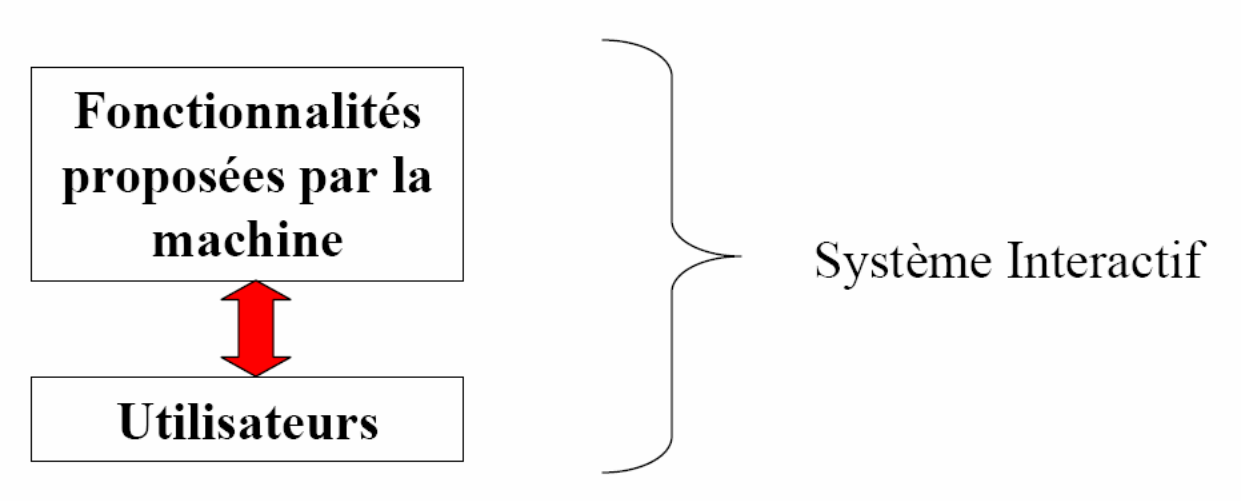
IHM : I comme …

- Interface et/ou Interaction ?
- L'interface désigne le vecteur (le média) par lequel deux éléments communiques
- Un système interactif est composé des deux éléments et du vecteur de communication



IHM : I comme …

- Termes difficilement dissociables
- Interactions : actions (mutuelles) entre acteurs
- Interfaces : dispositifs (techniques), vecteurs de communications



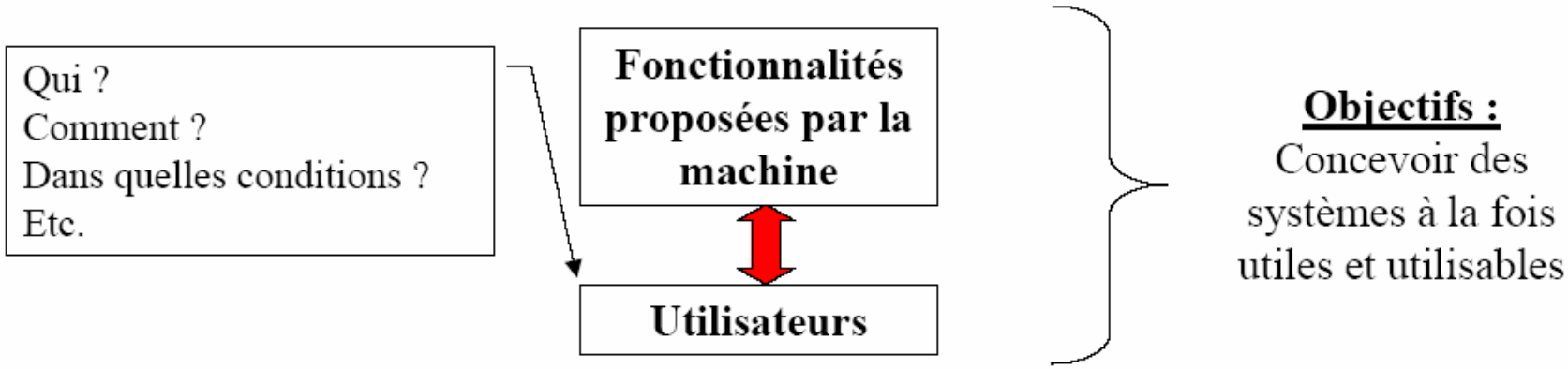
IHM : HM comme …

… comme Homme (être humain)

- des interfaces utilisables

… comme Machine

- des fonctionnalités utiles



D’où la définition …

- L'Interaction Homme-Machine

L'Interaction Homme-Machine est une discipline consacrée à la conception, à la mise en œuvre et à l'évaluation de systèmes informatiques interactifs destinés à des utilisateurs humains ainsi qu'à l'étude des principaux phénomènes qui les entourent

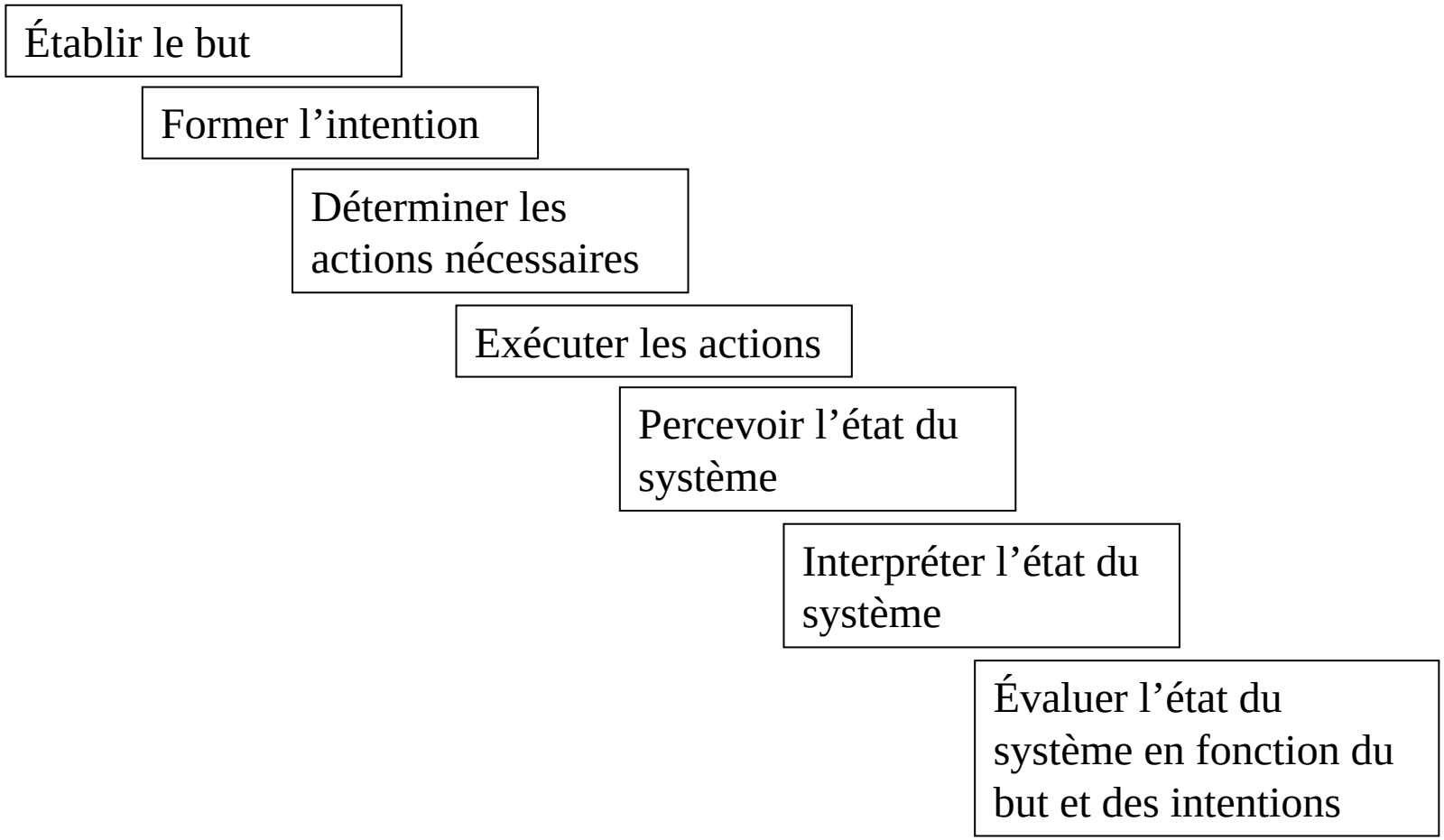
Pourquoi *interaction* et pas *interface* ?

- Les boutons, les menus, les couleurs ou les animations ne suffisent pas à rendre un système *utilisable*
- Comment se mesure l'*utilisabilité* ?
  - temps nécessaire pour apprendre
  - rapidité d'utilisation (*benchmarks*)
  - taux d'erreurs
  - facilité à se souvenir
  - satisfaction subjective
  - etc.

Pourquoi *interaction* et pas *interface* ?

- Ce n’est pas seulement l’interface qui compte, mais l’interaction :
  - la séquence d’actions nécessaires pour accomplir une tâche
  - l’adéquation entre le système et le contexte dans lequel il est utilisé
- Les deux sont indissociables dans les programmes d’aujourd’hui

Le cycle exécution – évaluation (Norman)



Evolution de l'interface graphique

- Interfaces à lignes de commande

  donnent accès à une commande (une fonction) du système

- Menus et écrans de saisie

  donnent accès à une application (un sous-ensemble des fonctions du système)

- Multi-fenêtrage, interfaces iconiques et manipulation directe

  donnent accès à l'ensemble des fonctions du système, et au-delà, à celles du réseau

Evolution de l'interactivité

- Le degré d'interactivité d'un système peut se mesurer au nombre et à la nature de ses échanges avec les utilisateurs
- Deux éléments importants ont contribué à l'augmentation du degré d'interactivité :
  - la possibilité d'exécution en parallèle de plusieurs tâches
  - l'avènement des interfaces graphiques
- Le nombre des échanges a beaucoup augmenté, mais leur nature n'a pas vraiment évolué

Importance des IHM

- Constat :
  - L'informatique envahit la vie:
    - au quotidien : PC, distributeur de billet, borne de réservation (avion-train), téléphone portable, etc.
  - au travail : réseau, bureautique professionnelle
    - systèmes embarqués (airbus), etc.
  - Or, tout le monde n'a pas les mêmes capacités
- Question :
  - que doit-on attendre d'une bonne interface ?

Importance des IHM

- Les sous questions qui en découle :
  - interface invisible, facilité d'utilisation ?
  - apprentissage ?
  - universelle ?
  - évolutive (ajout de fonctionnalités), etc.
- Aspects socio-économiques (rejet, argument de vente) contre les aspects techniques (réalisation et utilisabilité)

Importance des IHM

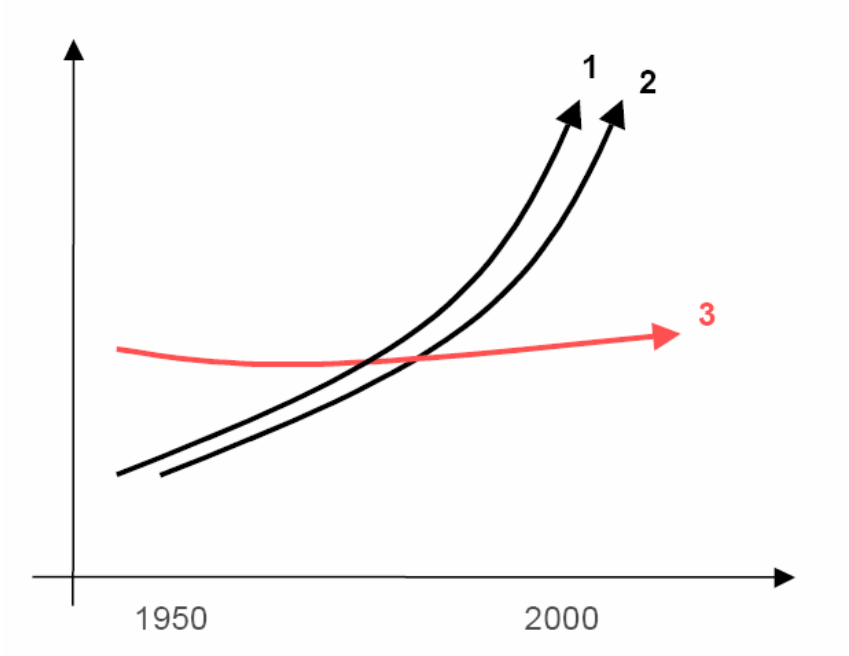
- IHM = carrefour des compétences
- En informatique : ingénierie (génie logiciel), système, développement
- Psychologie cognitive
- Ergonomie
- Sociologie
- L’approche est compliquée par l’intégration et la généralisation de nouvelles techniques d’interaction

Importance des IHM

1 - le matériel progresse sans cesse (Moore)

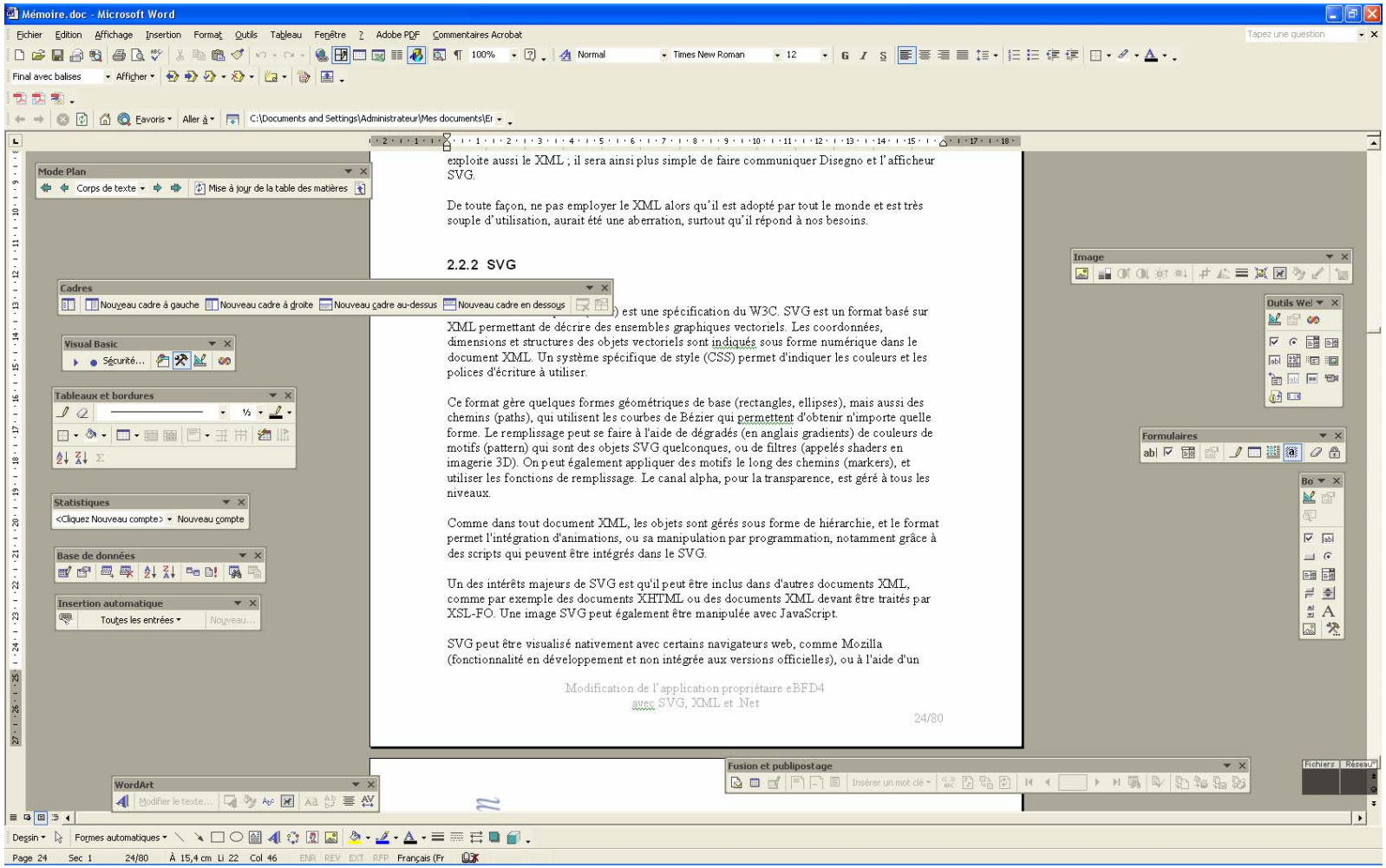
2 - les fonctionnalités promises aussi (Buxton)

3 - l'homme, lui, ne change pas, ou presque…

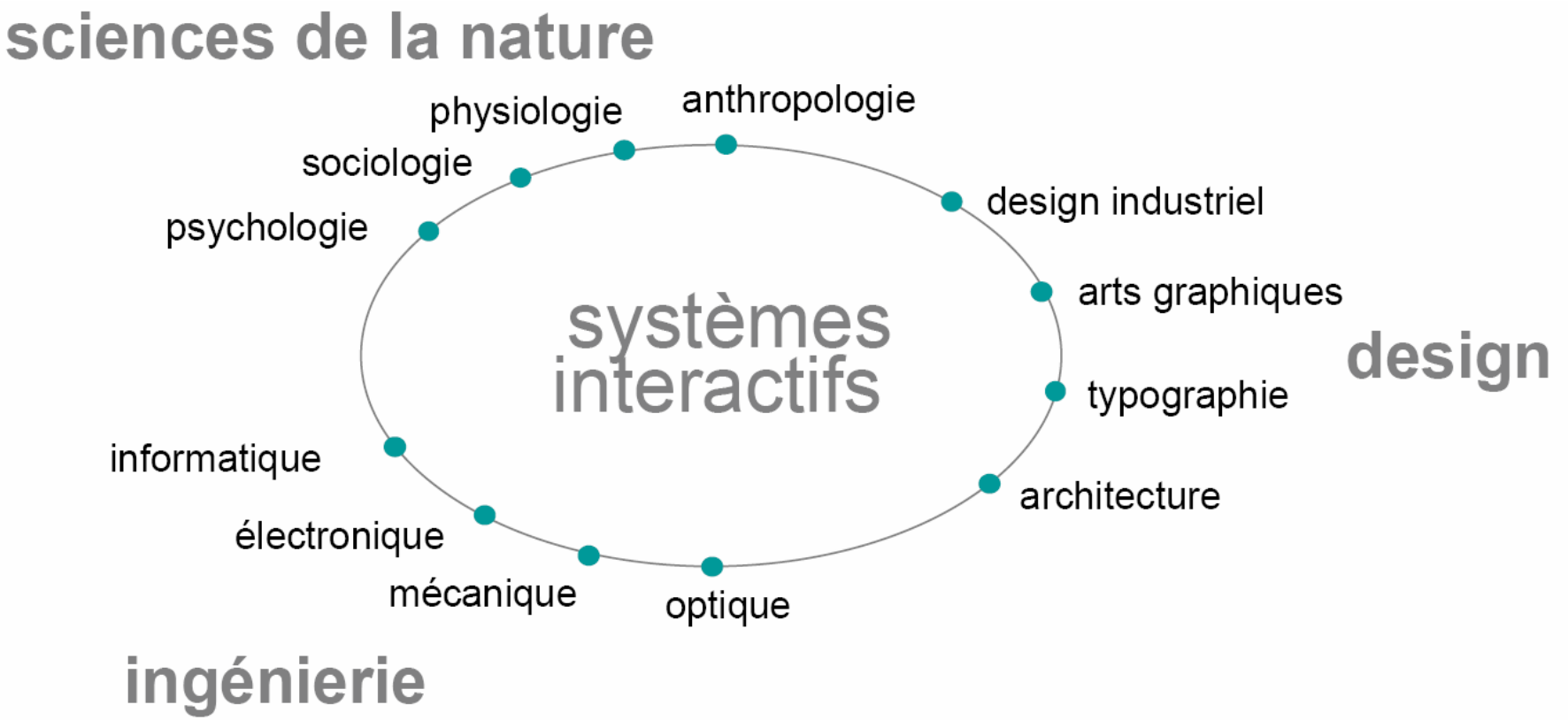


Limites des capacités de *perception* et d'*action* : le temps de la frustration !

Ça peut paraître frustrant



IHM : un domaine pluridisciplinaire



Ce qu’il faut retenir

- L'interactivité croissante a permis de passer de l'ordinateur *partenaire* à l'ordinateur *outil* ou *medium*
- Pourtant, sur bien des points, on est encore loin des visions des pionniers :
  - l'innovation matérielle, historiquement liée à l'innovation logicielle, a été progressivement abandonnée au profit du couple clavier/souris
  - les systèmes graphiques modernes ne sont similaires au Xerox Star qu'en apparence seulement
  - le Web n'est qu'une version réduite de ce qu'imaginaient Bush, Engelbart, Nelson ou Berners-Lee

Conception des interactions

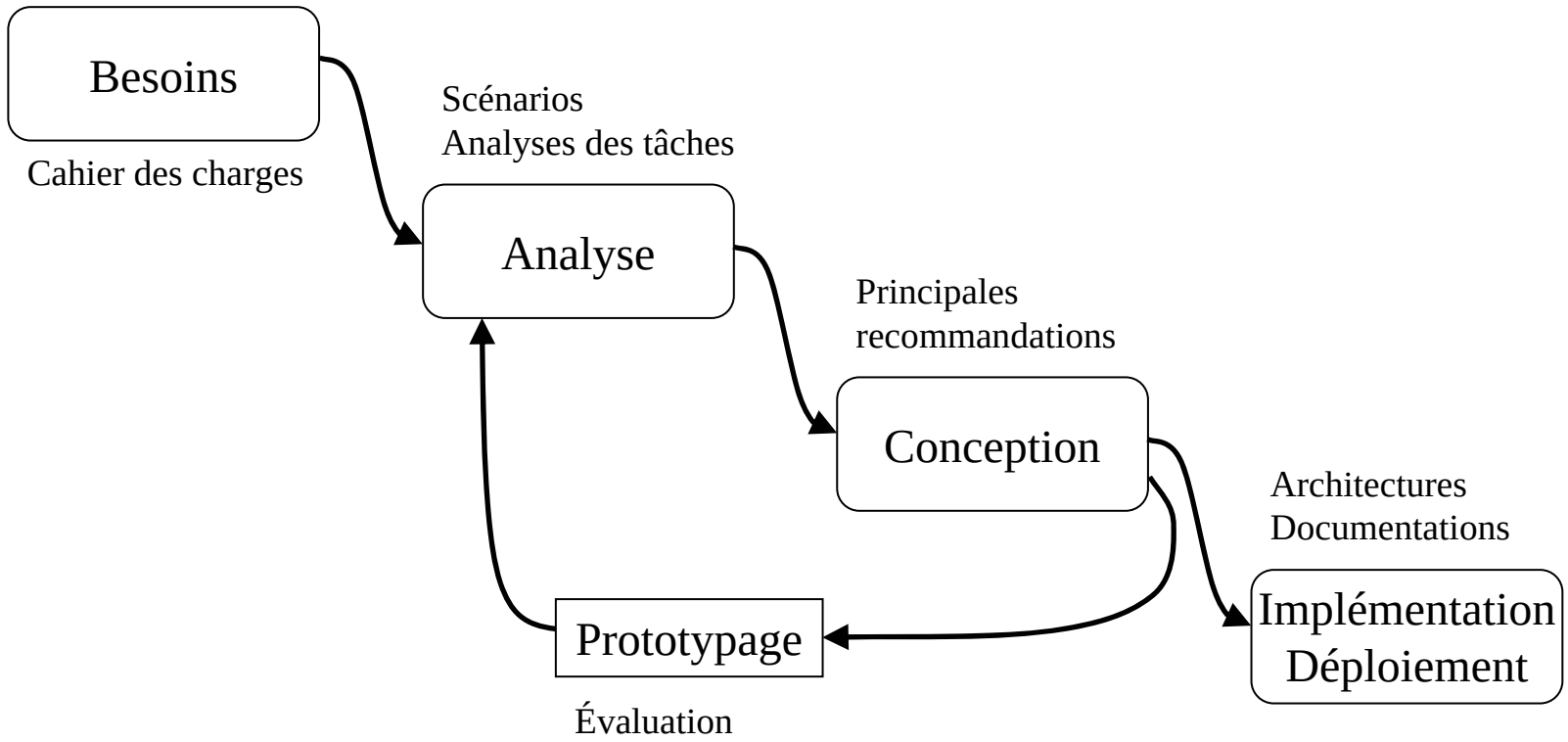
- Qu’est-ce que concevoir une IHM ?
  - Atteindre un **but** en prenant en compte des **contraintes**
- Buts :
  - Quels besoins ?
  - Pour qui est-ce ?
  - Pourquoi est-ce veulent-ils cela ?
- Contraintes
  - Quels matériels ?
  - Quels standards ?
  - Quel coût ?
  - Quel temps ?
  - Quelles sécurités ?

Conception des interactions

- Comme pour toute création, la règle d’or : **connaître son matériel**
- En IHM :
  - Compréhension des ordinateurs :
    - Possibilités/limitations
    - Capacités
    - Outils
    - Plateformes
  - Compréhension des humains :
    - Physiologique
    - Psychologie
    - Social
    - Erreur

Conception des interactions

- Le processus de conception



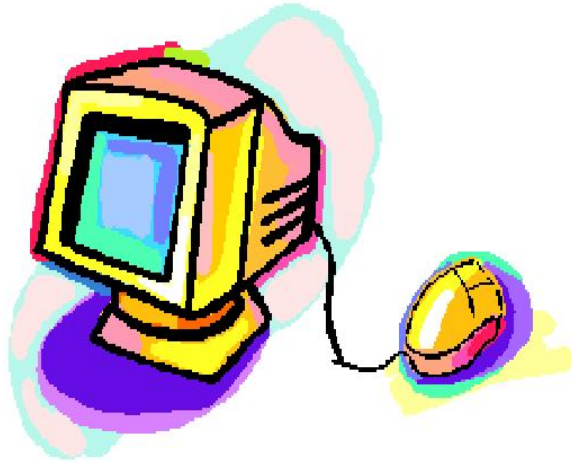
Interfaces WIMP

Windows (fenêtres)

Icônes

Menus, boîtes de dialogue, etc.

Pointage, sélection, tracé



- Avantages :
  - apprentissage rapide
  - environnements standards
  - des techniques d'interaction simples à utiliser et à programmer
- Comment en est-on arrivés là ?

Modèle graphique

- Un modèle graphique est :
  - un ensemble de primitives graphiques
  - un ensembles d'attributs graphiques
  - une sémantique graphique
- Trois grands modèles :
  - Pixellaire – bitmap
  - vectoriel 2D
  - vectoriel 3D
- Exemples : Windows, MacOS et XWindow sont basés sur des modèles graphiques pixellaires et vectoriels 2D

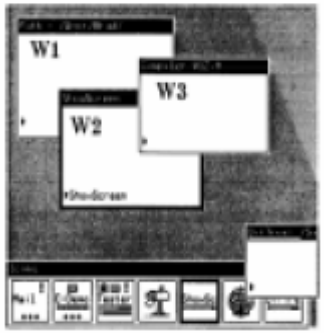
Système de fenêtrage

- Buts : permettre à plusieurs applications de partager l'écran et les périphériques d'entrée
- Une fenêtre est une zone autonome pour l'affichage et/ou l'entrée de données
- Fonctions du système de fenêtrage :
  - affectation des périphériques d'entrée (focus)
  - gestion de session (ouverture, fermeture, etc.)
  - communication entre applications
  - gestion des différentes fenêtres

Gestionnaire de fenêtres

“A window manager is a software package that helps the user monitor and control different contexts by separating them physically onto different parts of one or more display screens”

“Before window managers, people had to remember their various activities and how to switch back and forth”

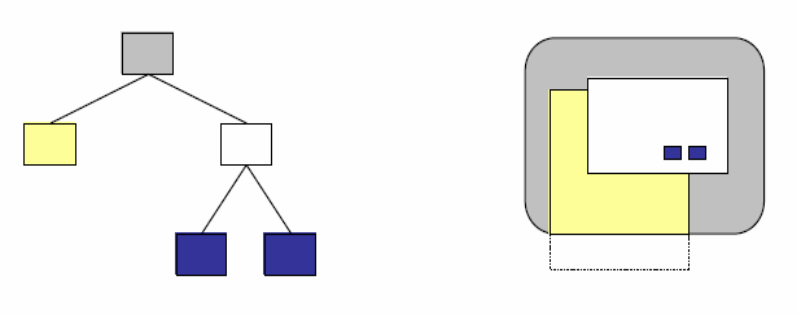


B. Myers. A taxonomy of window manager user interfaces. IEEE Computer Graphics and Applications 8(5), 1988

Modèles de fenêtrage

- Avec ou sans superposition

- Eventuellement hiérarchique



Manipulation des fenêtres

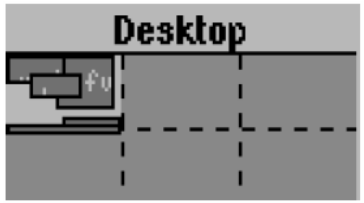
- Opérations "classiques"
  - placer, déplacer
  - dimensionner
  - iconifier, maximiser, restaurer la taille initiale
  - fermer
- Décorations
  - cadre et/ou barre de titre
  - menu, icônes, raccourcis clavier
- Opérations plus originales
  - réduire à la barre de titre
  - toujours au-dessus ou au-dessous

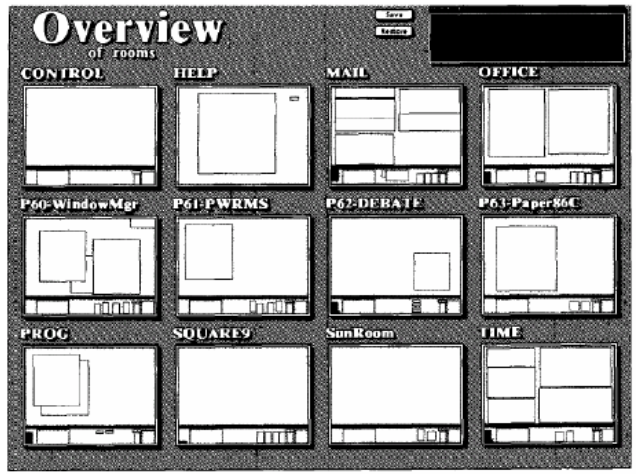
Réaffichage des parties cachées (*Expose event*)

- Par le système de fenêtrage
  - nécessite de mémoriser le contenu des fenêtres
- Par les applications
  - nécessite de transmettre les demandes de réaffichage aux applications (position et taille des zones)
  - peut bénéficier des mécanismes de clipping de la librairie graphique

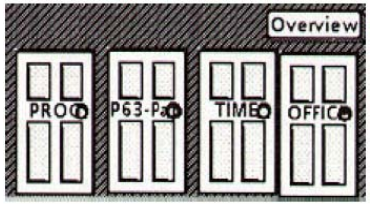
Peu de choses ont changé depuis Xerox Star

- Ecran virtuel





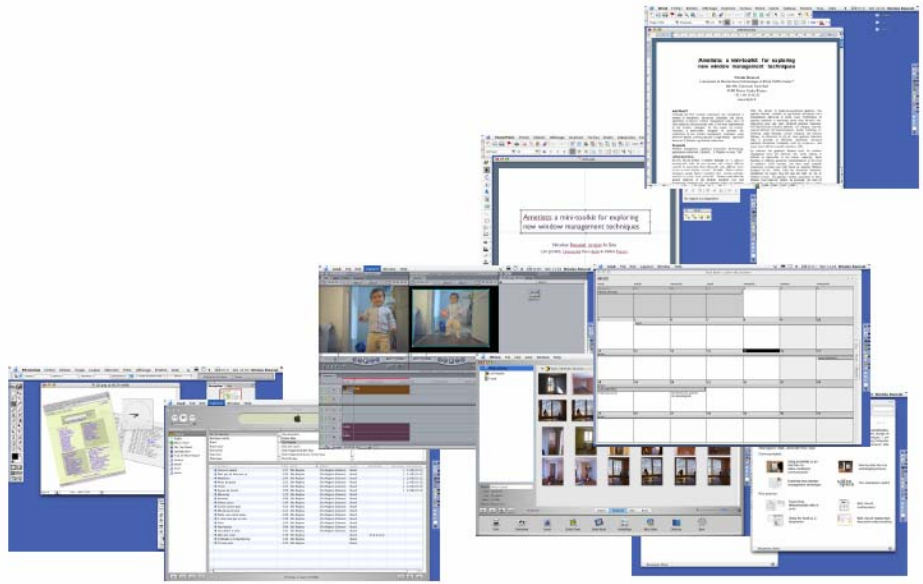
- Galleries (*rooms*)



- Utilisation de multiples écrans réels

Pourtant beaucoup de choses ont changé …

- Les activités se multiplient
  - traitement de texte, tableur
  - Jeux
  - édition d'images, de sons et de vidéos
  - courrier électronique
  - Web
  - messagerie instantanée
  - *media players*
  - etc.



Pourtant beaucoup de choses ont changé …

- Les technologies d'affichage ont évolué

9", 512x342 pixels

23", 1920x1200 pixels

- Les applications aussi
  - elles utilisent de plus en plus de fenêtres (PowerPoint X a 13 palettes, Photoshop 7.0 en a 17)
  - elles semblent conçues pour utiliser tout l'écran

Le pari d'Apple : le GPU plus fort que le CPU

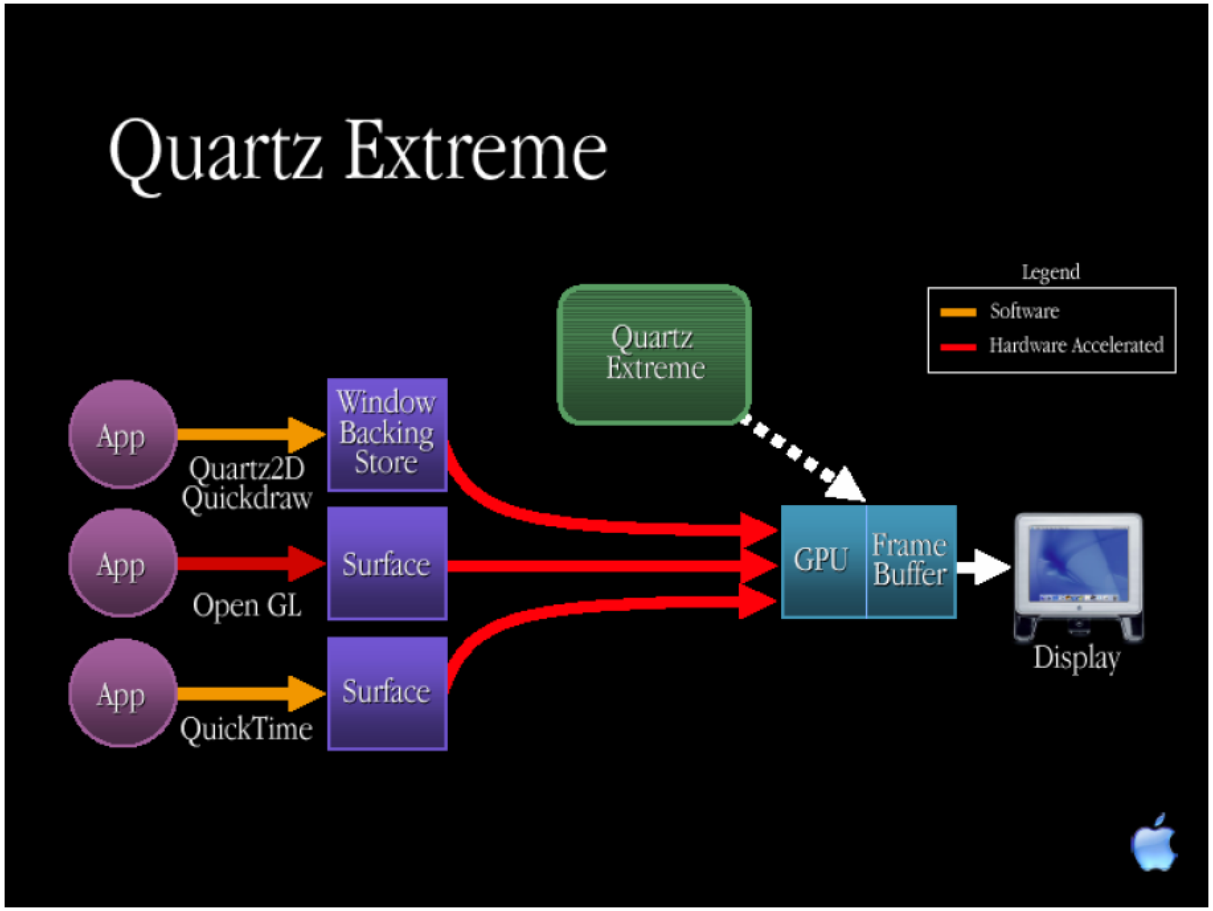
- CPU (Moore) : les performances doublent tous les 18 mois

  G5 = 58 millions de transistors
- GPU : les performances doublent tous les 6 mois

  Radeon 9600 Pro = 75 millions de transistors

  GeForce FX 5200 = 45 millions de transistors
- 2001: Quartz Extreme
  - "window system as a digital image compositor"
  - de nombreux effets ("because we can")

Apple : Quartz Extreme

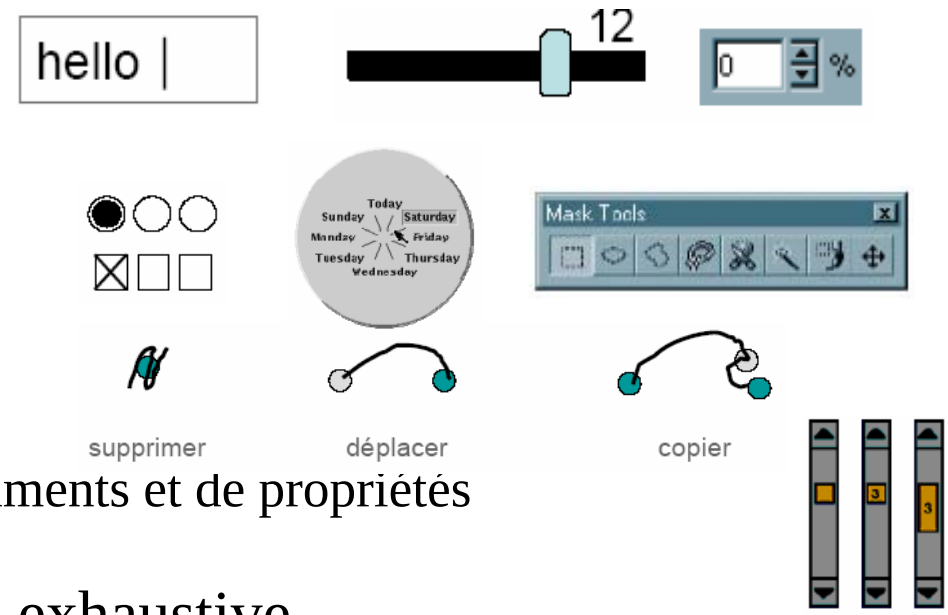


Les *Widgets*

- Le *widget* est un objet graphique interactif
- Il a trois facettes
  - *présentation* : aspect graphique, plus ou moins paramétrable (ex : ressources Xt)
  - *comportement* : réactions aux actions de l'utilisateur, généralement, peu ou pas paramétrable
  - *interface d'application* : lien avec l’application, notification des changements d'état
- Intérêt : permet d’établir des styles d’interaction standard
- Facilite l’apprentissage et la réutilisation de code

Les *Widgets*

- Permettent d’effectuer les tâches élémentaires d’interaction
  - saisie
  - sélection
  - déclenchement
  - défilement
  - spécification d'arguments et de propriétés



- Cette liste n’est pas exhaustive

Evolution des *widgets*

- *Widgets* de base du Macintosh (1984) :
  - bouton (*button*)
  - potentiomètre (*slider)*
  - menu déroulant (*pull-down menu*)
  - case à cocher (*checkbox*)
  - bouton radio (*radio button*)
  - champ texte (*text field*)
  - boîte de dialogue pour les fichiers (file open/save dialog)
- Ajouts ultérieurs : menus hiérarchiques, listes, *combo box*, tabs, listes hiérarchiques

Construction d’interfaces : plusieurs couches

- Bibliothèques de base
  - bibliothèque graphique, système de fenêtrage
  - gestion des entrées/sorties
  - intégration dans le système d’exploitation
- Boîtes à outils
  - architecture permettant de structurer l’application
  - services programmables (ex : préférences, copier/coller)
  - *widgets* réutilisables (ex : boutons, menus)
- Générateurs et langages de scripts

Boîtes à outils

X toolkit / Motif
- X Window
- langage C
- fonctions de rappel

AWT ou Swing
- X Window, MacOS, Windows
- langage Java
- messages

Tk
- X Window, MacOS, Windows
- langage Tcl
- fonctions de rappel, variables actives

Exemple Tcl/Tk (portable et gratuit) :

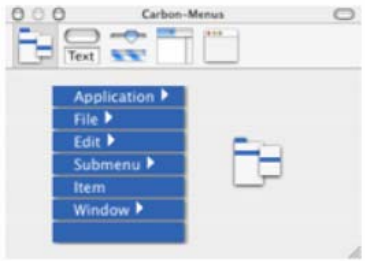
```
button .b -text "Hello world" -command exit
pack .b
```

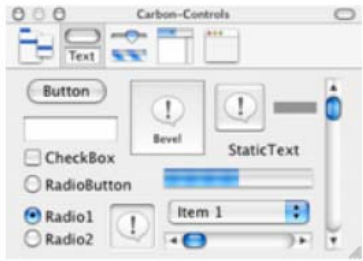
Générateurs d’interfaces

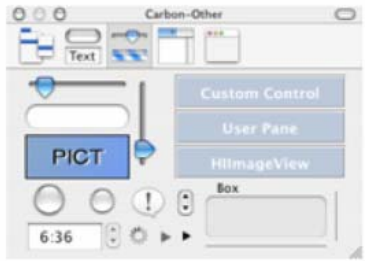
- But : aider (automatiser ?) la mise en oeuvre d’interfaces
- De nombreux noms
  - User Interface Management Systems (UIMS)
  - User interface builder
  - User interface development environment
- Principe
  - placer les *widgets*, modifier leurs attributs (couleur, etc.)
  - les connecter à l’application
  - tester leur comportement
- Exemples : Visual *, Interface Builder

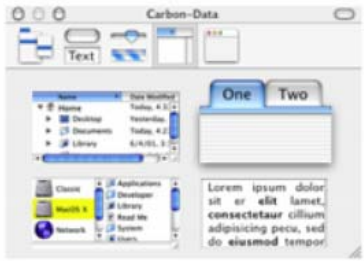
Générateurs d'interfaces

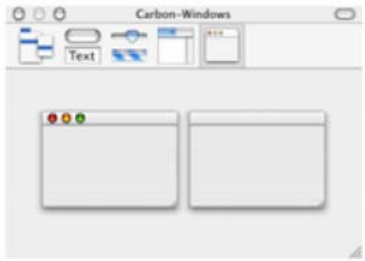
- Inconvénients : What You See Is All You Get
  - solutions partielles
  - le code reste à écrire…
  - difficile de modifier le code généré





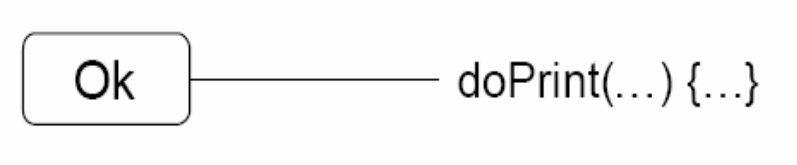




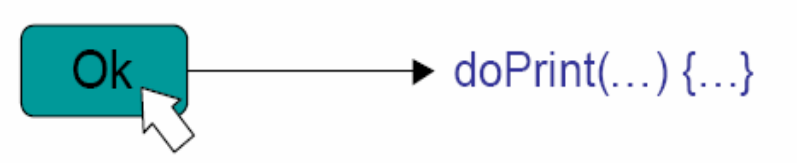


Lien *widget* - application

- Fonctions de rappel (*callbacks*)
  - enregistrées dans le *widget* à sa création

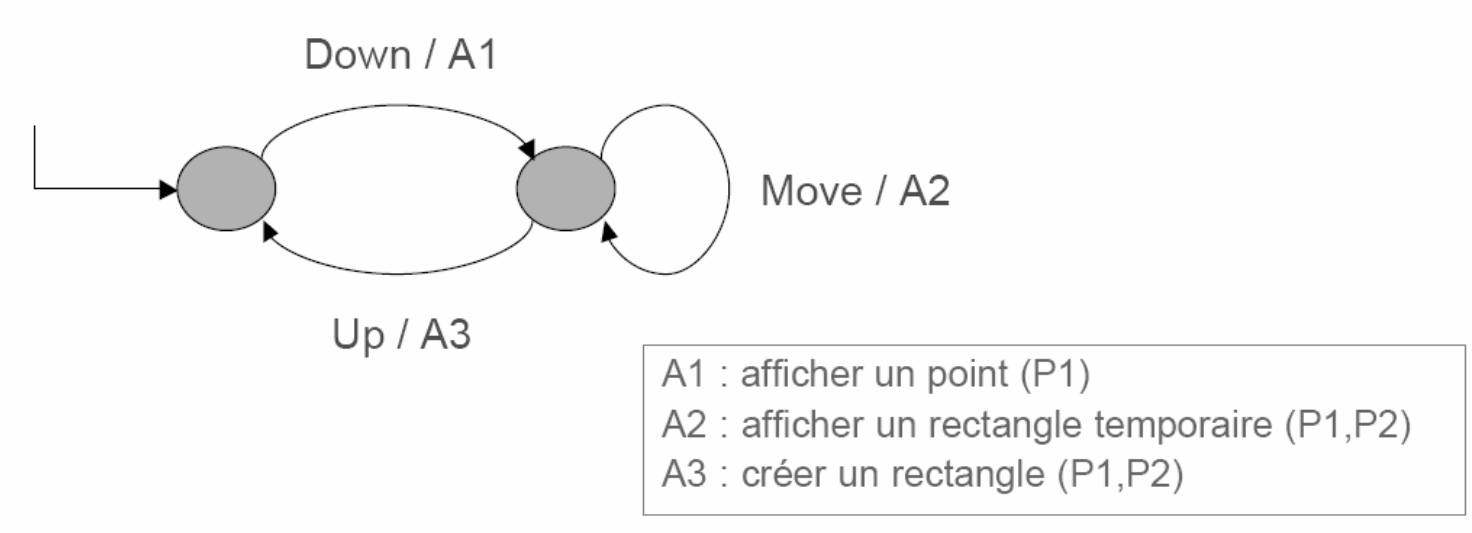


  - appelées lorsqu'une opération du *widget* est activé



Lien *widget* - application

- Machines à états : des automates étendus
  - conditions associées aux transitions
  - actions associées aux transitions
- Exemple : le "rubber-band"



Programmation événementielle

- Evénements liés aux périphériques
  - entrée/sortie du curseur dans une fenêtre
  - utilisation d'un des boutons
  - frappe au clavier
  - etc.
- Evénements liés aux applications
  - création/destruction de fenêtre
  - réaffichage
  - autre

Événements XWindow

```
#include <X11/Xlib.h>                          /* Librairies */
[...]

Display * display;                             /* Variables globales */
int screen_num;
Window win;
XEvent report;

int main(int argc, char * argv[]) {            /* Procédure principale */
  display = XOpenDisplay("");                  /* Initialisations */
  screen_num = DefaultScreen(display);
  win = XCreateSimpleWindow(display, [...]);
  XSelectInput(display, win, ExposureMask | ButtonPressMask);

  while ( 1 ) {                                /* boucle d'événements */
    XNextEvent(display, &event);
    switch ( event.type ) {
      case Expose: [...]
      case ButtonPress: [...]
    }
  }
```

Événements Win32

```
#include <windows.h>                                    /* Librairies */

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

int WINAPI WinMain( [...] ) {
  HWND hwnd;
  MSG msg;
  WNDCLASSEX wndclass;

  /* Remplissage de la structure WNDCLASSEX */
  wndclass.cbSize = sizeof(wndclass);
  wndclass.lpfnWndProc = WndProc;
  [...]
  wndclass.hInstance = hInstance;
  wndclass.hIcon = LoadIcon(NULL, IDI_APPLICATION);
  wndclass.hCursor = LoadCursor(NULL, IDC_ARROW);
  wndclass.lpszClassName = szAppName;
  wndclass.lpszMenuName = NULL;
  /* Enregistrement de la classe de fenêtre */
  RegisterClassEx(&wndclass);
```

Événements Win32

```
  /* Création d'une fenêtre de la classe précédemment déclarée */
  hwnd = CreateWindow(szAppName, "Hello, world!", [...]);
  ShowWindow(hwnd, iCmdShow);      /* Affichage de la fenêtre */
  UpdateWindow(hwnd);
  /* La boucle d'événements */
  while ( GetMessage(&msg, NULL, 0, 0) ) {
    TranslateMessage(&msg);
    DispatchMessage(&msg);
  }
  return msg.wParam;
}

LRESULT CALLBACK WndProc(HWND hwnd, UINT iMsg, [...]) {
  switch ( iMsg ) {
    case WM_PAINT:
    case WM_DESTROY:
      PostQuitMessage(0);
  }
  /* Appel de la procédure par défaut */
  return DefWindowProc(hwnd, iMsg, wParam, lParam);
}
```

Programmation événementielle

- Application non-interactive : “start, do something, stop”
- Ancienne façon de voir les applications interactives : de temps en temps, le système pose des questions à l’utilisateur
- Version moderne actuelle :
  - l’utilisateur contrôle l’application
  - celle-ci attend un signe (notion de boucle principale dans laquelle sont traités les événements)
  - de nombreux états à conserver… c’est compliqué !
- La solution de facilité : l’utilisation de modes

Quoi qu’est-ce un mode ?

- Un mode est un état de l'interface dans lequel les actions de l'utilisateur sont interprétées de façon homogène et différentes des autres modes

- Inconvénient des modes : ils restreignent les fonctions disponibles à un moment donné et obligent l'utilisateur à se souvenir des changements d'état de l'application pour connaître les effets de ses actions

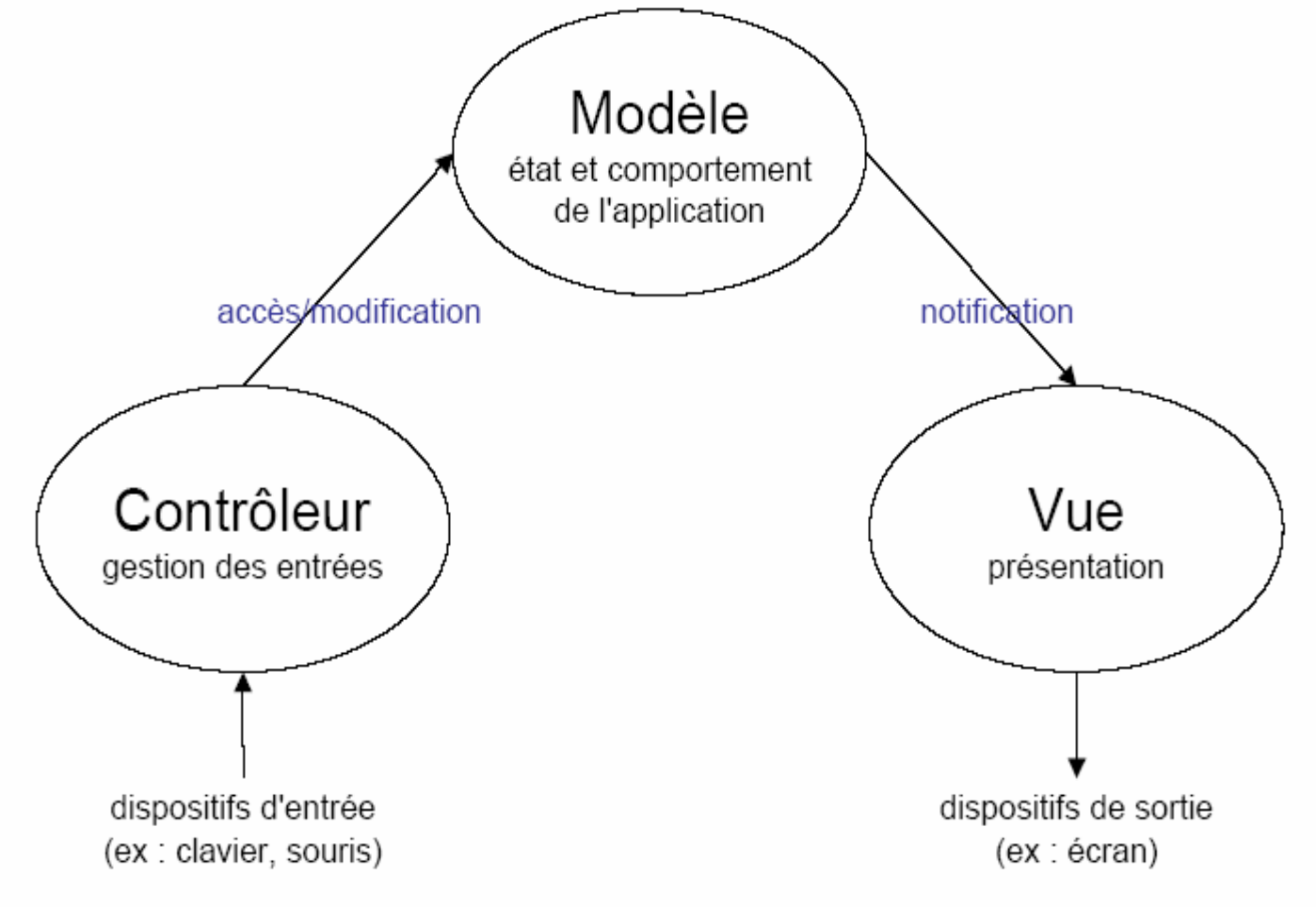
Modes d'interaction

- Modes spatiaux
  - la même action en différents endroits produit des effets différents
  - le + : mémoire spatiale
- Modes temporels
  - la même action à des moments différents produit des effets différents
  - problème : manque d’initiative pour le changement de mode
- Micro-modes
  - modes liés à une action physique

L’architecture MVC

- Origine :
  - modèle issu de Smalltalk-80
  - adopté par de nombreuses boîtes à outils (Java AWT, Microsoft MFC, …)
- Principe :
  - le **modèle** correspond aux données de l'application (structures et fonctions)
  - la **vue** présente des informations à l'utilisateur à partir des données du modèle
  - le **contrôleur** se charge de l'interaction avec l'utilisateur

L’architecture MVC



Manipulation directe

- 3 principes fondamentaux (B. Shneiderman, 1983)
  - représentation permanente des objets et actions d'intérêt
  - utilisation d'actions "physiques" (ex : pointer, déplacer) plutôt que des commandes à la syntaxe complexe
  - opérations rapides, incrémentales et réversibles, dont les actions sur les objets sont immédiatement visibles

"Point and click instead of remember and type"

Manipulation directe

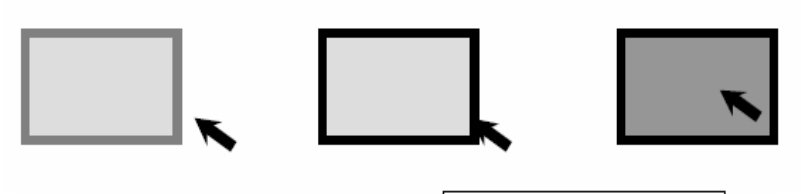
- Mise en œuvre
  - utilisation de représentations graphiques métaphoriques
  - approche sujet-verbe (-compléments)
- Avantages
  - actions génériques (ex: copier)
  - possibilité d'apprentissage rapide, par démonstration par exemple
  - connaissance persistante

Couplage perception/action

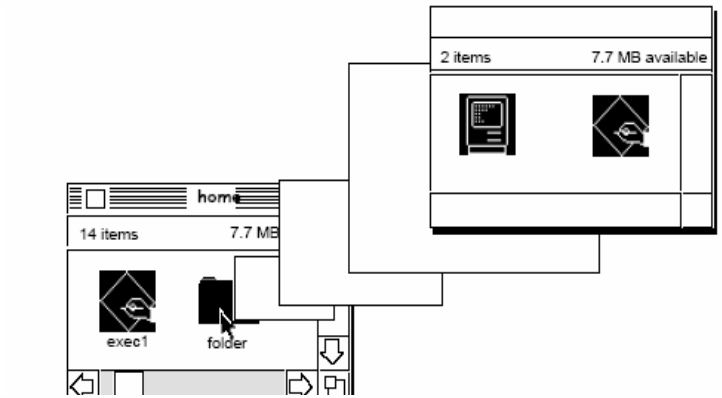
- *Agir pour percevoir*
  - perception de la profondeur par des mouvements de la tête
  - perception de la texture d'un objet en déplaçant le doigt sur sa surface
- *Percevoir pour agir*
  - ajuster les mouvements du bras pour saisir un objet
- Importance de la continuité de ce couplage
- Ex : le glisser – déposer

Le *feedback* (retour d'information)

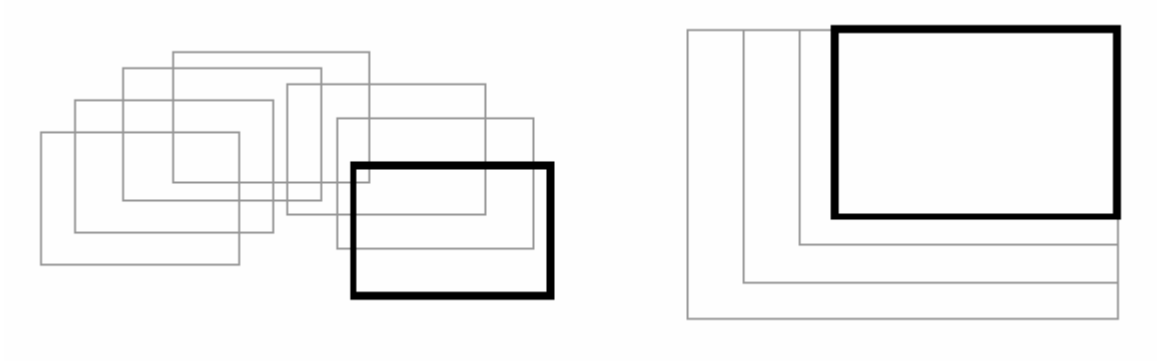
- Pointage



- Sélection



- Tracé, déplacement, transformation



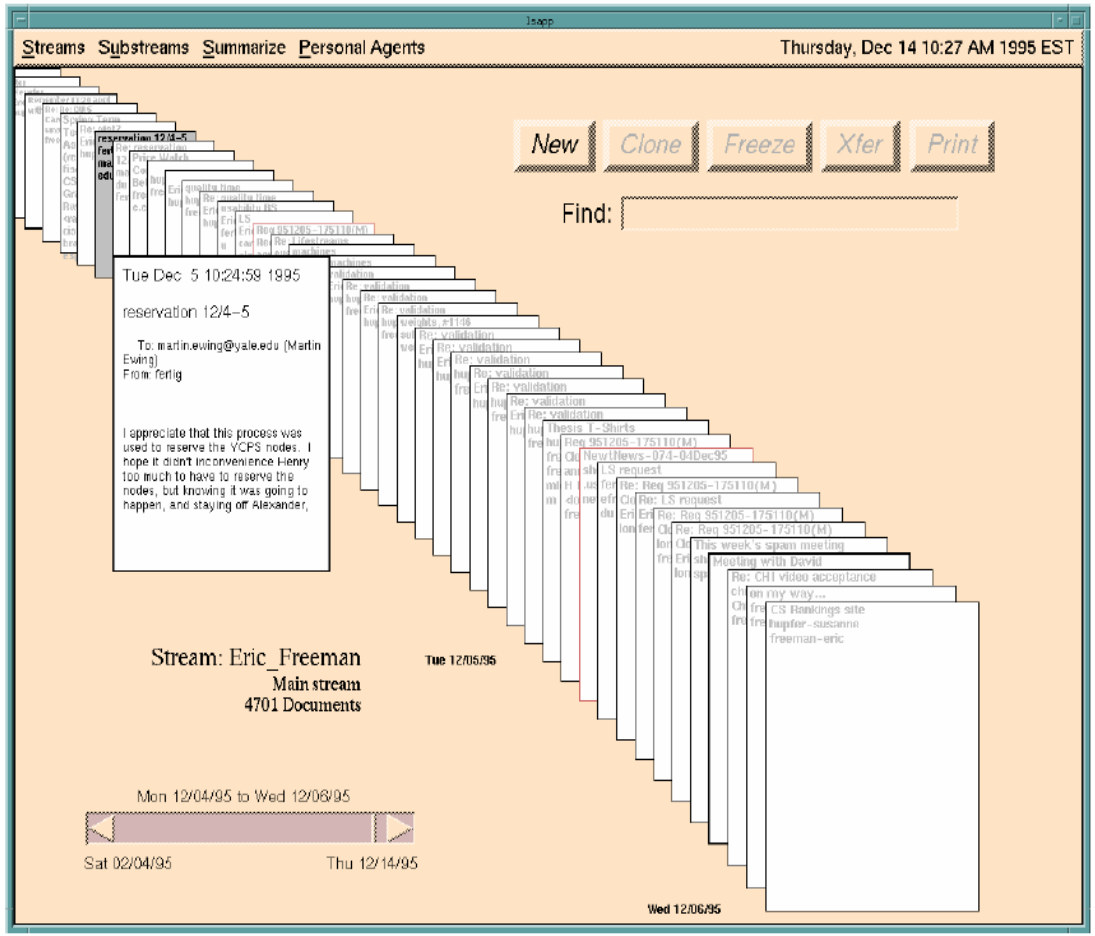
Métaphore

- Problèmes lié à la manipulation directe : quels sont les objets d'intérêt ? comment les représenter ?
- Métaphore : utilisation de concepts connus de l’utilisateur pour introduire le modèle conceptuel du système
- Avantage : modèle naturel qui facilite l'apprentissage, l’utilisateur anticipe le comportement du système
- Dangers : trop esthétique, trop restrictive (trop fidèle)
- Métaphores du monde réel
  - métaphores spatiales : bureau, pièce, etc.
  - métaphores sociales ou techniques : théatre, télévision

Les piles

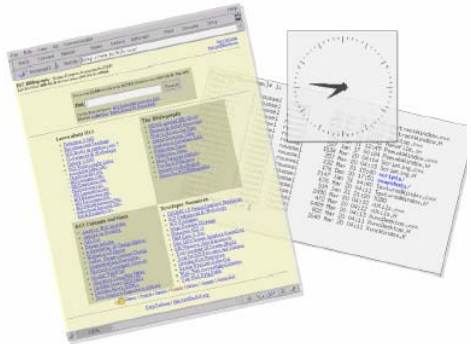
Mander, Salomon & Wong, CHI 1992

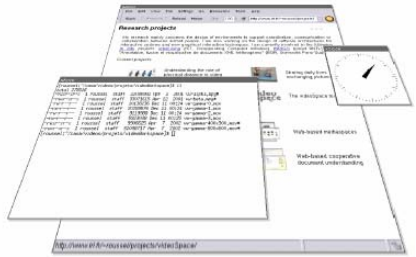
LifeStreams

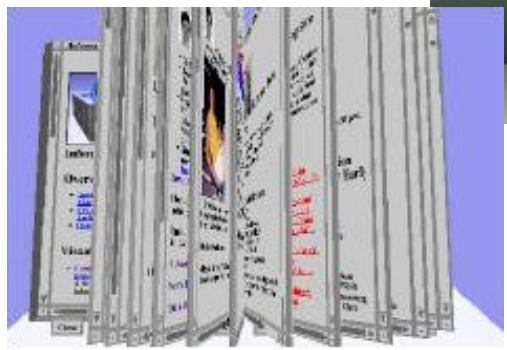


Fertig, Freeman & Gelernter, CHI 1996

Mais aussi …







Quelques références

P. Wegner. "Why interaction is more powerful than algorithms". Communications of the ACM, 40(5):80-91, May 1997.

J. Johnson al. (1989) "The Xerox Star: A Retrospective". IEEE Computer, September 1989.

B. Myers. "A brief history of human-computer interaction technology". ACM interactions, 5(2):44-54, March/April 1998.

http://www.computerhistory.org/

Remerciements à Nicolas Roussel et Stéphane Conversy pour leurs supports de cours