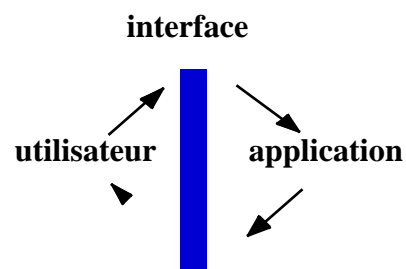


# 1. Généralités sur les interfaces graphiques

## Généralités sur les interfaces graphiques

- Interfaces utilisateur
- Techniques graphiques d'interaction
- Système X

## Programmes interactifs



- Application
  - read()
  - eval()
  - display()

- Utilisateur
  - see()
  - think()
  - act()

# 1. Généralités sur les interfaces graphiques

Interfaces graphiques

## Interfaces textuelles et graphiques

<p><b>Interfaces textuelles</b></p> <ul style="list-style-type: none"><li>o unicité de la source: stdin</li><li>o unicité de la sortie: stdout,stderr</li><li>o granularité (unité d'échange) élevée:<ul style="list-style-type: none"><li>  ligne entière en entrée</li><li>  message en sortie</li></ul></li><li>o nombreux modes:<ul style="list-style-type: none"><li>  mode insertion/commande</li><li>  mode lecture/écriture</li></ul></li><li>o aide laborieuse, explicite</li></ul>	<p><b>Interfaces graphiques</b></p> <ul style="list-style-type: none"><li>o multiplicité des sources : clavier, souris, avec interprétation selon le lieu:<ul style="list-style-type: none"><li>  fenêtre</li><li>  menu, barre de bouton</li><li>  raccourci, équivalent clavier</li></ul></li><li>o multiplicité des sorties : texte, messages, dialogues, clignotements</li><li>o granularité fine:<ul style="list-style-type: none"><li>  l'évènement élémentaire (KeyPress, KeyRelease)</li><li>  évènement symbolique (activer)</li></ul></li><li>o aide implicite facilitée par<ul style="list-style-type: none"><li>  bulles d'aide</li><li>  changement de curseur et/ou de couleur</li><li>  réaction aux actions (clignotement)</li></ul></li></ul>
--	--

**Les interfaces graphiques sont plus difficiles et plus longues à réaliser**

Interfaces graphiques

## Techniques graphiques d'interaction

- o *Formulaires*
  - | Servent à afficher, diriger et modifier un ensemble de variables, au moyen d'*éléments d'interaction* que sont
    - » menus, boutons
    - » listes, textes éditables
    - » variateurs et ascenseurs
- o *Manipulation directe*
  - | Principe qui donne à l'utilisateur *l'illusion* de manipuler directement un objet.
  - | Exemples : programmes de dessin, traitements de textes, pao.
  - | Techniques : déplacer, étirer avec la souris, "glisser-déposer"
- o *Techniques de coopération*
  - | Utilise des protocoles pour l'échange de données, ou de liaison entre applications.
  - | Exemples : couper-coller, presse-papiers, OLE (Object Linking and Embedding)
- o *Dialogues dirigés*
  - | Tutoriels, systèmes d'aide, installateurs, hypertextes, visualisations, animations.
  - | L'interaction se résume à une *navigation* .
  - | Fréquemment, de tels systèmes sont multimédia.

# 1. Généralités sur les interfaces graphiques

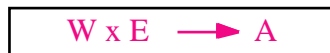
## Quelques dates

- o avril 1981 - Station de travail Star 8010. Conçue au PARC (Palo Alto Research Center) de Xerox. Elle avait
  - | un affichage bitmap
  - | des icônes, des fenêtres
  - | des polices de caractères à espacement proportionnel
  - | une souris
- o janvier 1984 - Premier Macintosh, à interface graphique, avec éditeur de texte et programmes dessin.
- o 1990 Première version commerciale de Windows 3.0.
- o 1986 Première version commerciale du système X (X10R3).
- o mars 1995 Première version des Java AWT
  
- o Toute interface est basée sur la programmation par événements.

## Aspect général

Tout programme dirigé par événement définit une fonction

- de la fenêtre déclenchant ou recevant l'évènement
- de la nature de l'évènement



Windows, Intrinsics, Motif (objets), Java 1.1



Xlib, MacOS, Java 1.0

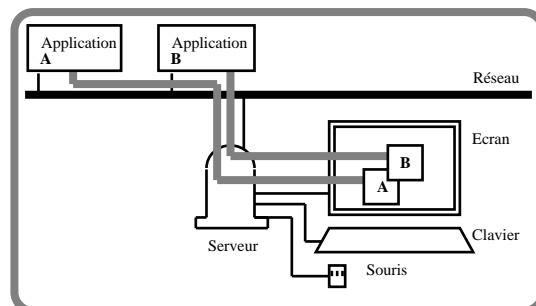


# 1. Généralités sur les interfaces graphiques

## Le système X

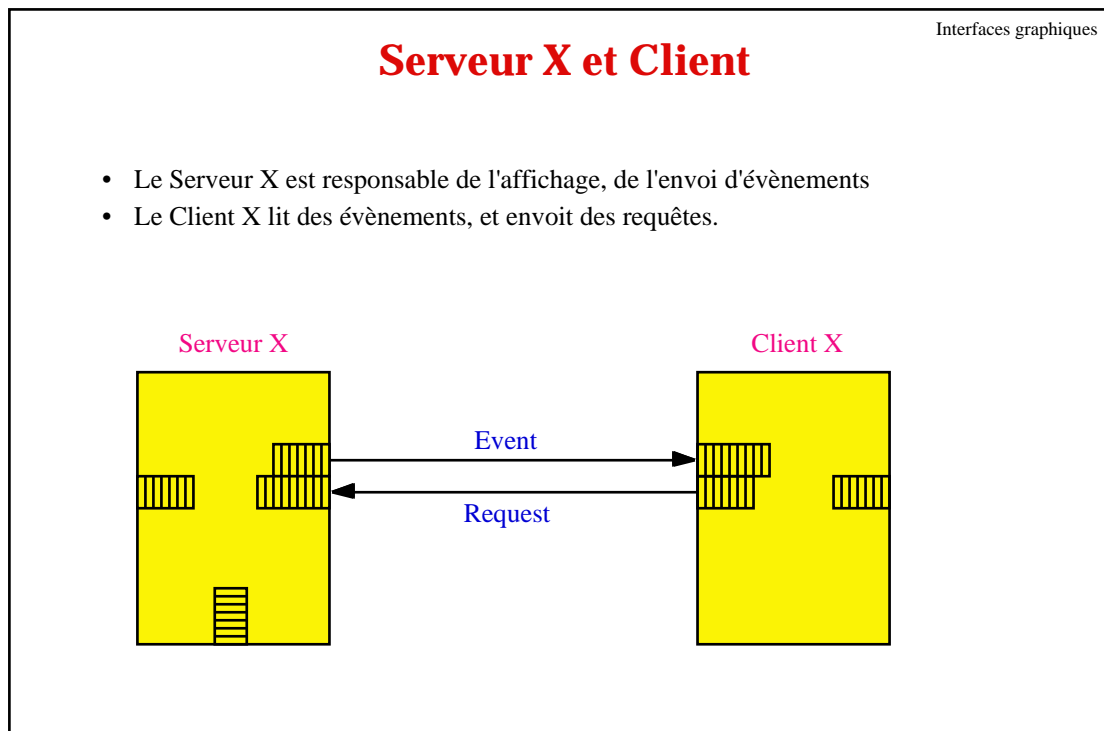
- Systématiquement utilisé sous Unix et Linux
- En plusieurs couches :
  - | Protocole de communication : Protocole X
  - | Système de base niveau : Xlib
  - | Outils de gestion de composants (widgets) : Xt Intrinsics
  - | Toolkits : Athéna Widgets, Motifs
- Tout système pour interfaces graphiques utilise Xlib
- Certains n'utilisent pas Xt Intrinsics : Tcl/Tk, Java

## Présentation générale

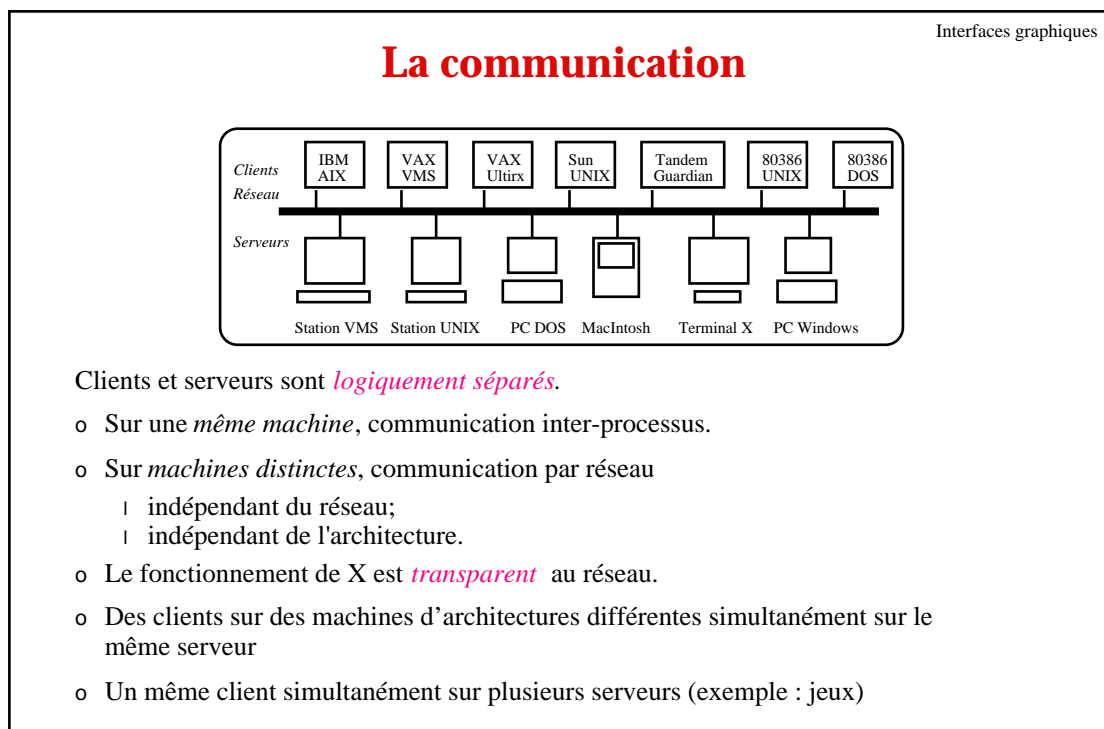


- Un modèle *client-serveur*.
- Comporte 3 composantes :
  - | Le *serveur* - programme qui contrôle le dispositif d'affichage (display) : écran, clavier, souris.
  - | Les *clients* - programmes d'application.
  - | Le dispositif de *communication* entre clients et serveur.

# 1. Généralités sur les interfaces graphiques



9



10

# 1. Généralités sur les interfaces graphiques

Interfaces graphiques

## GUI, API, Framework et les autres

Terminologie

- o **GUI** (Graphical User Interfaces) ou *interface graphique utilisateur* : l'interface graphique, telle qu'elle se présente à un utilisateur.
- o **API** (Application Programmer's Interface) ou *interface du programmeur d'application* : l'ensemble des routines utilisé pour créer des interfaces utilisateur.
- o **Interface Builders** ou *créateurs d'interfaces* : outils (graphiques en général) permettant de développer rapidement ses applications.
- o **Frameworks** ou squelettes d'application : une application déjà écrite "dans ses grandes lignes", et à compléter.
- o Une API évoluée fournit des composants d'interfaces à comportement intégré, comme boutons, menus, ascenseurs.
- o Le niveau de l'API influe sur l'effort du programmeur
- o Les ensembles de composants de création d'interfaces (composant et fonction) sont des *toolkits* (boîtes à outils).

Biblio de base - Programmeur

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

11

Interfaces graphiques

## Le serveur X

### Les fonctions du serveur

- o Il gère les fenêtres
  - ┆ Création, configuration, destruction
- o traite les *sorties graphiques* :
  - ┆ Gestion des *polices* et des *textes*;
  - ┆ Tracés *graphiques* (lignes, arcs, régions);
  - ┆ Gestion des *images* (pixmap) et  *curseurs*.
- o détecte les *entrées* et les transmet aux clients :
  - ┆ Clavier , boutons, mouvements de souris.
  - ┆ Transmission aux clients
- o Gérer les *communications* avec le réseau.

### Ce que le serveur ne fait pas

- o Il n'interprète pas les évènements
- o Il ne redessine pas le contenu d'une fenêtre redécouverte. En revanche, envoie un évènement *expose* ;
- o ne fait pas de zoom
- o ne fait pas de gestion logique des fenêtres (comme agrandissement, élastique, etc.) : c'est le rôle du *gestionnaire de fenêtres*.

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

12



# 1. Généralités sur les interfaces graphiques

## Programmation

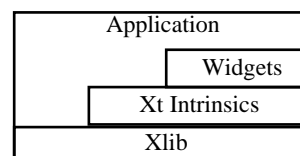
Un programme à interface graphique comporte trois parties :

- o Etablissement de la connexion au serveur.
- o Création de la hiérarchie des fenêtres (resp. des widgets).
- o Gestion de la boucle d'évènements :
  - | Lire un évènement (*read*);
  - | En fonction de la nature de l'évènement et de la fenêtre où il a eu lieu, entreprendre l'action appropriée (*eval*).
  - | envoyer les requêtes correspondantes au serveur(*print*).
- o La boucle d'évènement peut être plus ou moins explicite:
  - | explicite en Xlib
  - | abstraite en Xt Intrinsic
  - | implicite en Java

## Niveaux de programmation

X est structuré en trois niveaux :

- o *Xlib* : niveau de base, pour la gestion graphique.
- o *Xt* ou *Intrinsic* : emploi d'ensembles de widgets. Ne contient que quelques widgets de base.
- o *Widget sets* . N'est pas fourni par le Consortium X (sauf Athena Widget Set).

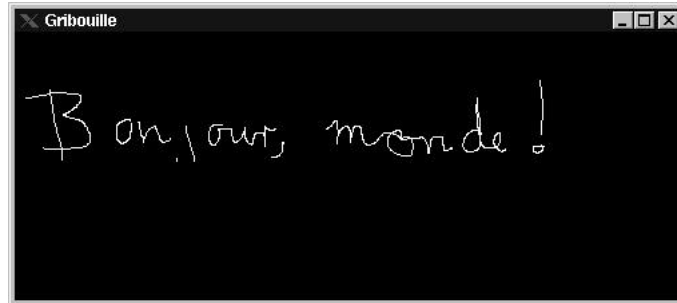


*X Toolkit* est le nom collectif le deux derniers niveaux : les Intrinsic + un ensemble de widgets.

Ensembles de widgets qui ne sont pas basés sur les Intrinsic : *XView*, *IlogViews*, *JavaAWT*, *Tcl/Tk* par exemple .

# 1. Généralités sur les interfaces graphiques

## Un exemple Xlib



- o Crée une fenêtre intitulée Gribouille, de fond noir.
- o Avec un bouton de souris enfoncé,
  - | le curseur change de forme
  - | en se déplaçant, on gribouille
- o Au relachement, le curseur reprend la forme initiale.

## Programme (1)

```
/** gribouille **/  
#include <X11/Xlib.h>  
#include <X11/cursorfont.h>  
Display* dpy; /* le serveur d'affichage */  
int ecran; /* l'ecran, en general c'est 0 */  
Window fen; /* la fenetre de dessin */  
GC ctx; /* contexte graphique */  
Cursor crayon, croix; /* curseurs */  
  
void initialisation() {  
    dpy = XOpenDisplay(NULL); /* demande de connection au serveur */  
    ecran = DefaultScreen(dpy); /* l'ecran (normalement 0) */  
    fen = XCreateSimpleWindow(  
        dpy, /* creation fenetre simple */  
        DefaultRootWindow(dpy), /* le display */  
        0, 0, 500, 200, /* la fenetre mere */  
        4, /* origine, largeur et hauteur */  
        WhitePixel(dpy, ecran), /* epaisseur de la bordure */  
        BlackPixel(dpy, ecran)); /* couleur de la bordure */  
    XStoreName(dpy, fen, "Gribouille"); /* couleur de fond */  
    XMapWindow(dpy, fen); /* nom de la fenetre */  
    XSelectInput(dpy, fen, /* affiche la fenetre */  
        ButtonPressMask | ButtonReleaseMask | ButtonMotionMask); /* evenements recus */  
    ctx = XCreateGC(dpy, fen, 0, NULL);  
    XSetForeground(dpy, ctx, WhitePixel(dpy, ecran)); /* dessins en blanc */  
    crayon = XCreateFontCursor(dpy, XC_pencil); /* pour dessiner */  
    croix = XCreateFontCursor(dpy, XC_crosshair); /* pour deplacer */  
    XDefineCursor(dpy, fen, croix);  
}
```

# 1. Généralités sur les interfaces graphiques

Interfaces graphiques

## Programme (2)

```
/** gribouille **/  
...  
void main (int argc, char* argv[])  
{  
    XEvent ev;  
    short x, y, xx, yy;  
    initialisation();  
    for (;;) { /* Boucle pour les evenements */  
        XNextEvent(dpy, &ev);  
        switch (ev.type) {  
            case ButtonPress :  
                x = ev.xbutton.x; y = ev.xbutton.y;  
                XDefineCursor(dpy, fen, crayon);  
                break;  
            case ButtonRelease :  
                XDefineCursor(dpy, fen, croix);  
                break;  
            case MotionNotify :  
                xx = ev.xmotion.x; yy = ev.xmotion.y;  
                XDrawLine(dpy, fen, ctx, x, y, xx, yy);  
                x = xx; y = yy;  
        } /* du big switch */  
    }  
}
```