XWindow System

Implementazione open-source di
XWindow System

XOrg

X ebbe origine al Massachusetts Institute of Technology (MIT) nel 1984. La versione corrente di protocollo, X11, apparve nel settembre del 1987.

X.Org Foundation guida il progetto con l'implementazione corrente X.Org Server, disponibile come "free and open source software" sotto licenza MIT.

XOrg è il proseguimento di ciò che è stato XFree86: quest'ultimo, cambiando licenza e rendendosi incompatibile con la GPL, non poteva più essere incluso nelle distribuzioni Linux; è nato un nuovo progetto, nel quale sono confluiti la maggior parte dei programmatori di XFree86.

XOrg

Nel 2005 è stato rilasciato XOrg X11R7.0, il primo grande rilascio del sistema XWindow dopo dieci anni di sviluppo discontinuo. Oltre a driver video aggiornati, troviamo DRI per creare implementazioni di OpenGL che sfruttino l'accelerazione grafica 3D.

DRI è una parte integrale di X.Org 7.x, ed è integrata con Mesa, una implementazione open source delle API OpenGL. Parecchi driver per schede accelerate 3D sono stati scritti con le specifiche DRI, fra cui ATI, Matrox, 3DFX e Intel.

Nel giugno 2012 è stato rilasciato XOrg X11R7.7 che è a tutt'ora l'ultima versione rilasciata.

Il Sistema XWindow

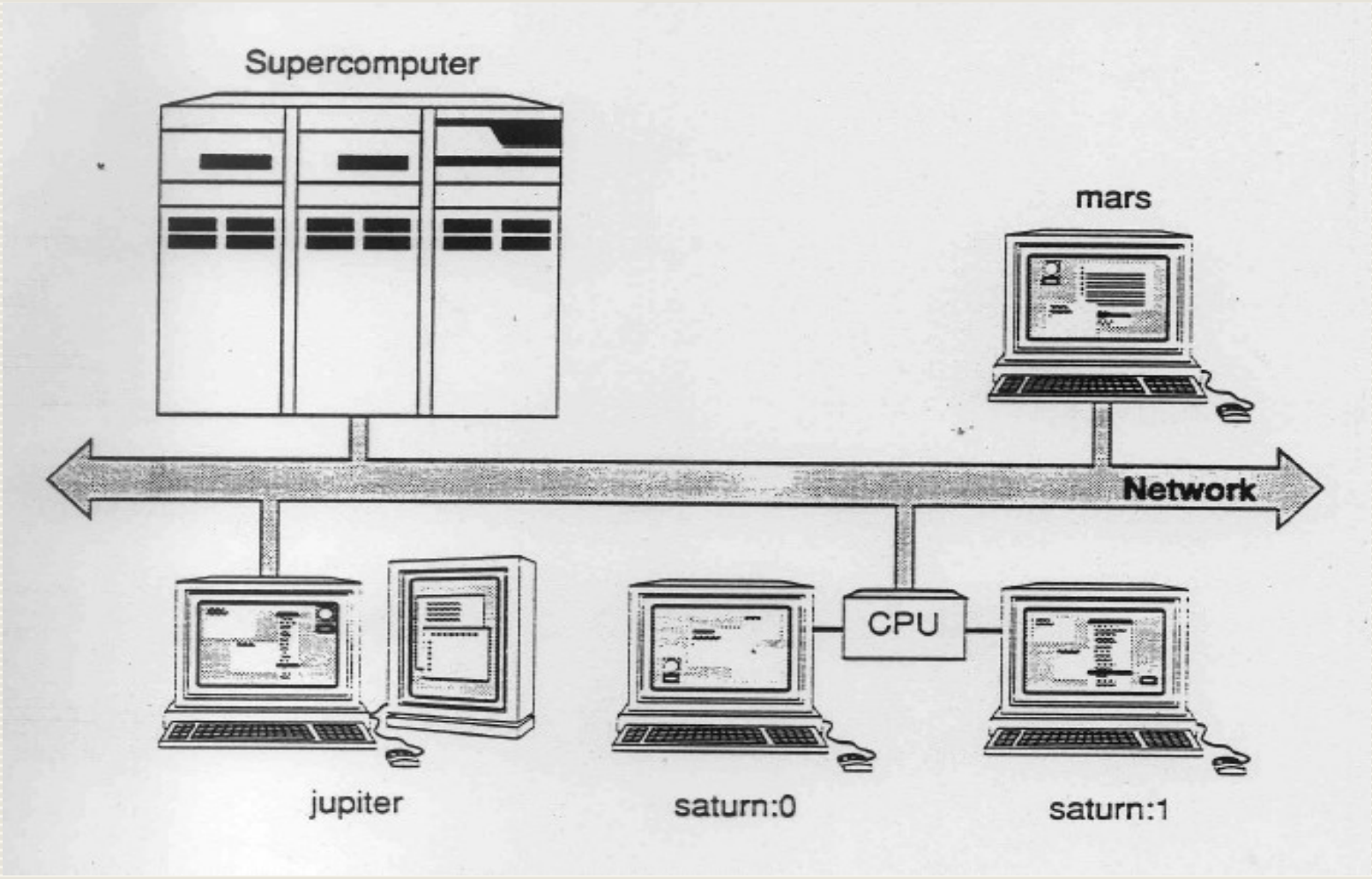
- Indipendente da HW e S.O., permette di realizzare applicazioni portabili su macchine diverse sotto S.O. diversi

- Progettato per sistemi distribuiti, permette a più applicazioni di cooperare attraverso una rete anche con architetture e/o S.O. diversi. Tutto avviene in maniera completamente trasparente all'applicazione.

Nota: l'interfaccia grafica (o Window Manager) può girare su una macchina diversa da quella dove gira l'applicazione.

- X non impone alcun stile di interfaccia (o Window Manager) o di interazione con l'utente. Questo viene definito ad uno strato software sopra ad X

Il Sistema Xwindow



Il Sistema XWindow

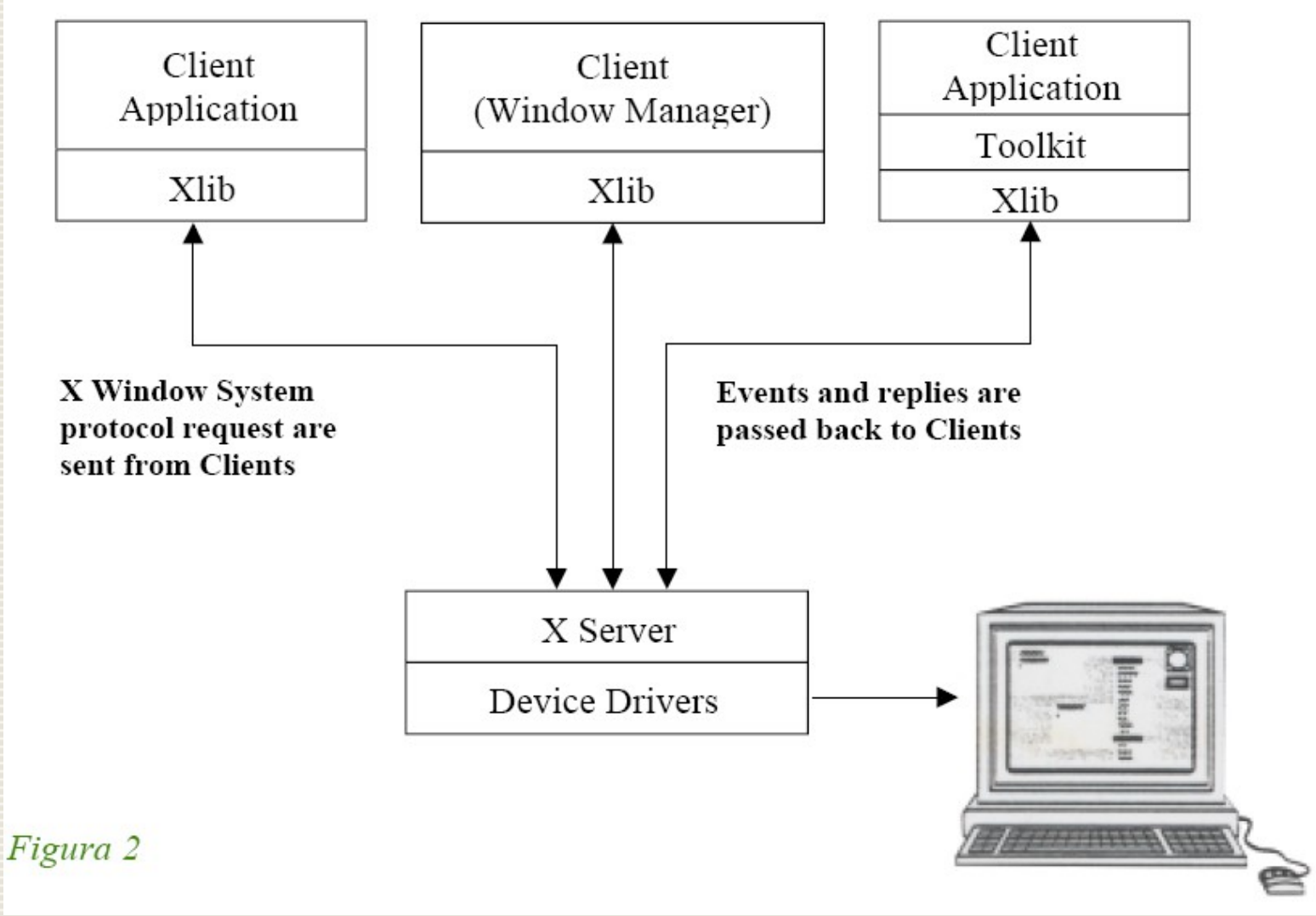
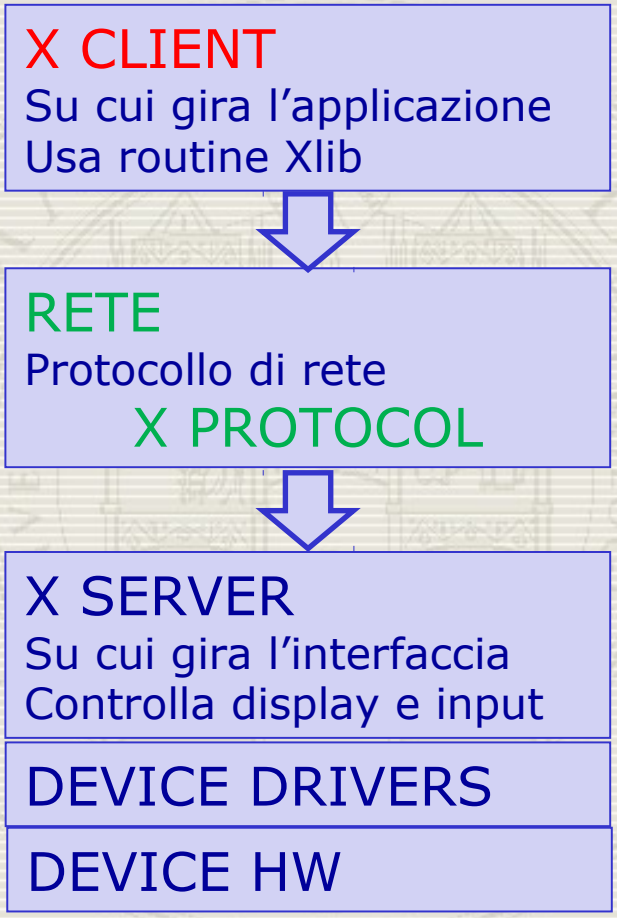


Figura 2

Il Sistema XWindow



X Protocol

Definisce quali pacchetti di informazioni (messaggi) sono scambiati tra Client e Server tramite Rete:

RICHIESTE (Client ➡ Server)
Alloca una risorsa, legge valori correnti di una risorsa, modifica valori di una risorsa. Interazione asincrona: il server mette le richieste in coda e le soddisfa quando ha tempo.

RISPOSTE (Server ➡ Client)
Info di ritorno per certi tipi di richieste (es. alloca una risorsa, ritorna identificatore per accedere alla risorsa).

EVENTI (Server ➡ Client)
Xlib (sul client) mette eventi in coda da cui il programma può leggerli uno ad uno per processarli (interazione asincrona)

X Protocol

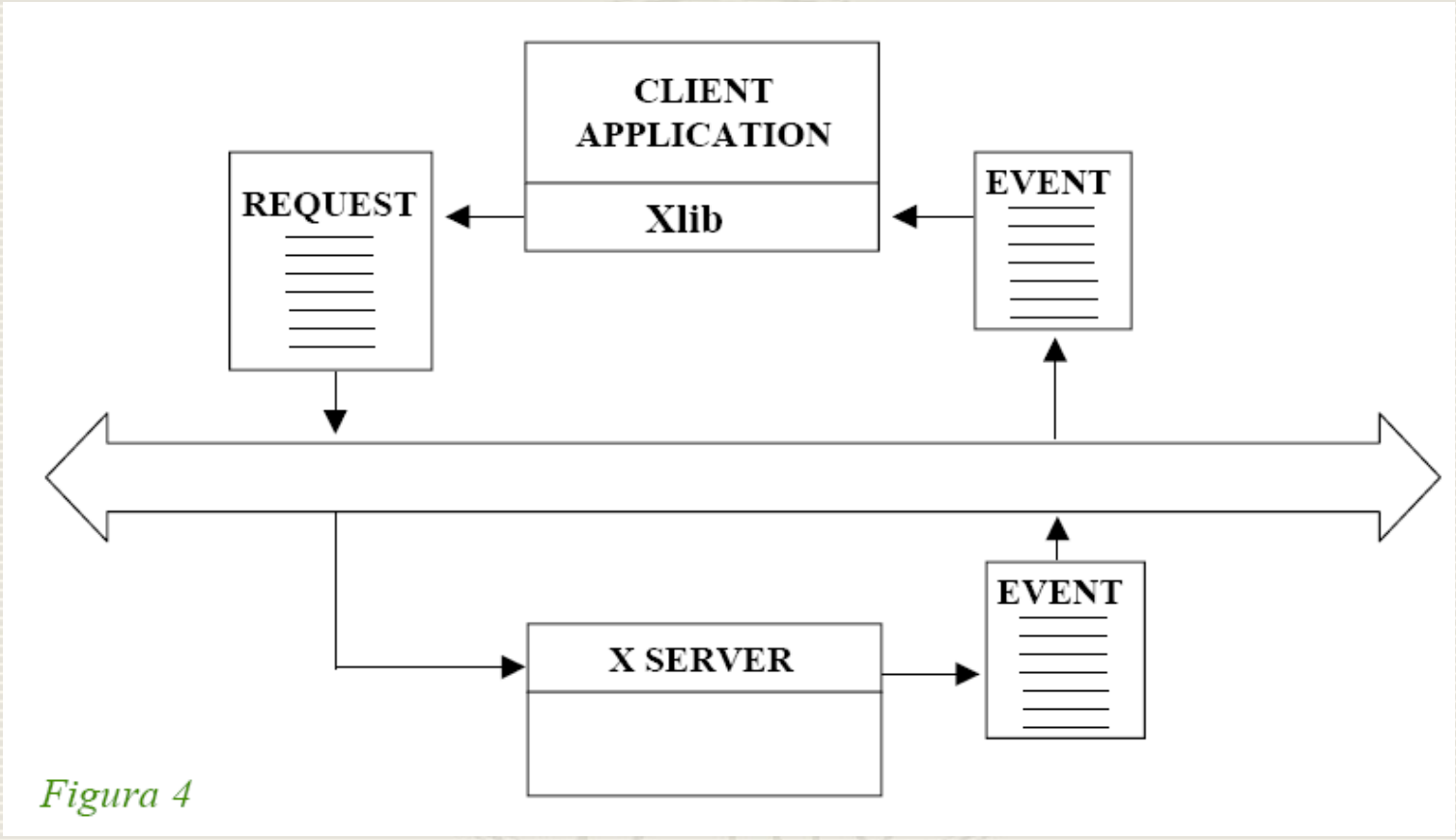


Figura 4

X Server

- Alloca e gestisce risorse (finestre, font, bitmap, pixmap, …) per conto del client: solo l'identificatore della risorsa è noto al client e passato via rete (insieme ai messaggi che si riferiscono a quella risorsa)

- Passa input dai device (tastiera, mouse) ai client

- Passa output (testo e grafica 2D) dai client al display

X Client

Un'applicazione che usa un X Server tramite Xlib.

Xlib

Strato più basso di X. Permette al client di comunicare col server per:

- Ottenere risorse (font, pixmap, finestre, …)
- Manipolare risorse (leggere, assegnare e modificare proprietà)
- Ricevere input (eventi)
- Inviare output (testi e grafica 2D)

Window Manager

•Un particolare X Client con il compito speciale di gestire finestre sullo schermo: posizionare, ridimensionare, iconizzare, gestire stack order, stabilire quale finestra ha il focus per mouse e tastiera, …

•X non fissa un particolare Window Manager, ne esistono vari che utilizzano politiche diverse.

•Un Window Manager aggiunge decorazioni alle finestre (bordo, barra del titolo, controlli per iconizzare, ridimensionare, …).
Queste caratteristiche della finestra non sono gestite dal programma applicativo, che invece gestisce solo l'interno della finestra ed eventuali sottofinestre.

Window Manager

•Solo le "Top – Level" Window di ogni applicazione sono gestite dal Window Manager.
L'applicazione non ha neppure il controllo sulla posizione e dimensione di queste; può dare Hints al Windows manager, ma non è detto che siano accettati.

•Invece l'applicazione ha il pieno controllo sull'interno ed ha la responsabilità di ridisegnare il contenuto (Refresh) quando la finestra cambia dimensioni o la finestra torna visibile dopo essere stata (anche solo parzialmente) oscurata da altre.

X Client che usa ToolKit

•Un Toolkit è uno strato SW sopra Xlib.
Fornisce elementi predefiniti di interfacce (oggetti), che il programmatore può configurare ed assemblare per costruire un'interfaccia.

•Gli elementi di interfaccia sono chiamati WIDGET (Window Object).

Esempio: bottone, check-point, ecc.

•Il programma deve:
- Creare i WIDGET configurandoli come preferisce, definire callback
- Lanciare il Main Loop del Toolkit, che automaticamente cattura eventi e chiama le callback associate

Modello a strati

Xlib: interfaccia a basso livello tra client e server

Xtoolkit (vari): basati su Xlib, forniscono componenti di alto livello per la costruzione di GUI.

OpenGL: libreria grafica ad alto livello che si appoggia su Xlib o direttamente sulla scheda grafica.

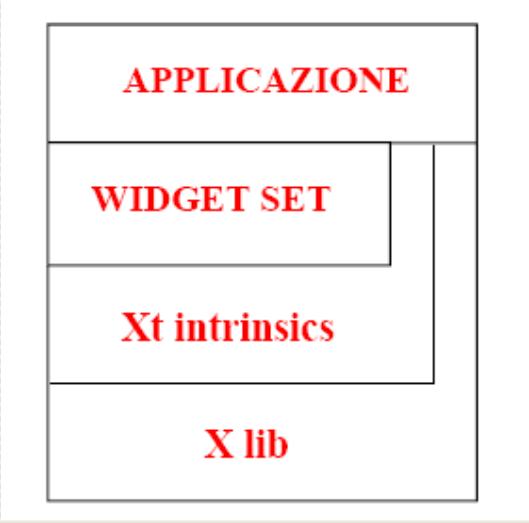
Toolkit (vari): basati su OpenGL, forniscono componenti di alto livello per la costruzione di GUI.

Tipi di ToolKit

•X fornisce un insieme di strumenti per definire e manipolare WIDGET base per implementare un toolkit. Questa è la libreria Xi intrinsics.

•Varie case forniscono WIDGET SET particolari da associare a Xt intrinsics.

Xtoolkit = Xt intrinsics (parte standard X) + Widget set (parte non standard)



Esempi di WIDGET SET:

- OSF Motif
- AT&T OpenLook
- MIT Athena (free, ma molto limitato)

X Client

X Client che usa solo Xlib

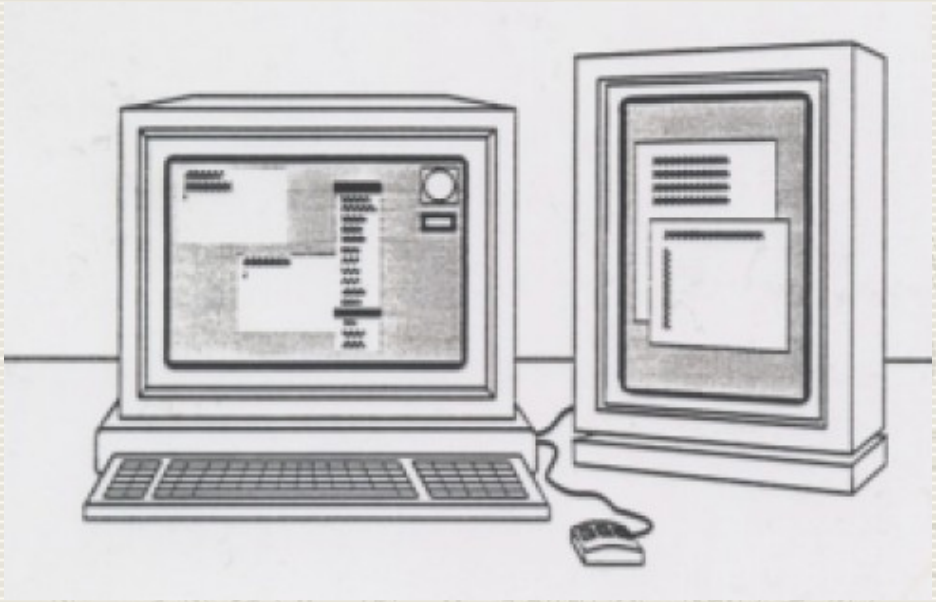
Il programma deve:

- Definire le sue finestre, la loro gerarchia (sotto finestre), la politica con cui le sottofinestre vengono risistemate quando le dimensioni della finestra genitrice vengono modificate, e molti altri dettagli per ogni singola finestra

- Definire quali eventi si vogliono catturare in ogni finestra

- Implementare un ciclo dove si attende un evento, si guarda di che tipo è e in che finestra si è verificato per agire di conseguenza.

Display e Screen



Un display è definito come una workstation grafica con una tastiera, un mouse e uno o più screen.

Connessione con ssh ad un server remoto con X

$ssh -X –l loginname remotehost.domain

(oppure)

$ssh –X loginname@remotehost.domain

quindi

Password: …

L'uso di ssh garantisce una connessione sicura da un XServer locale ad una applicazione remota

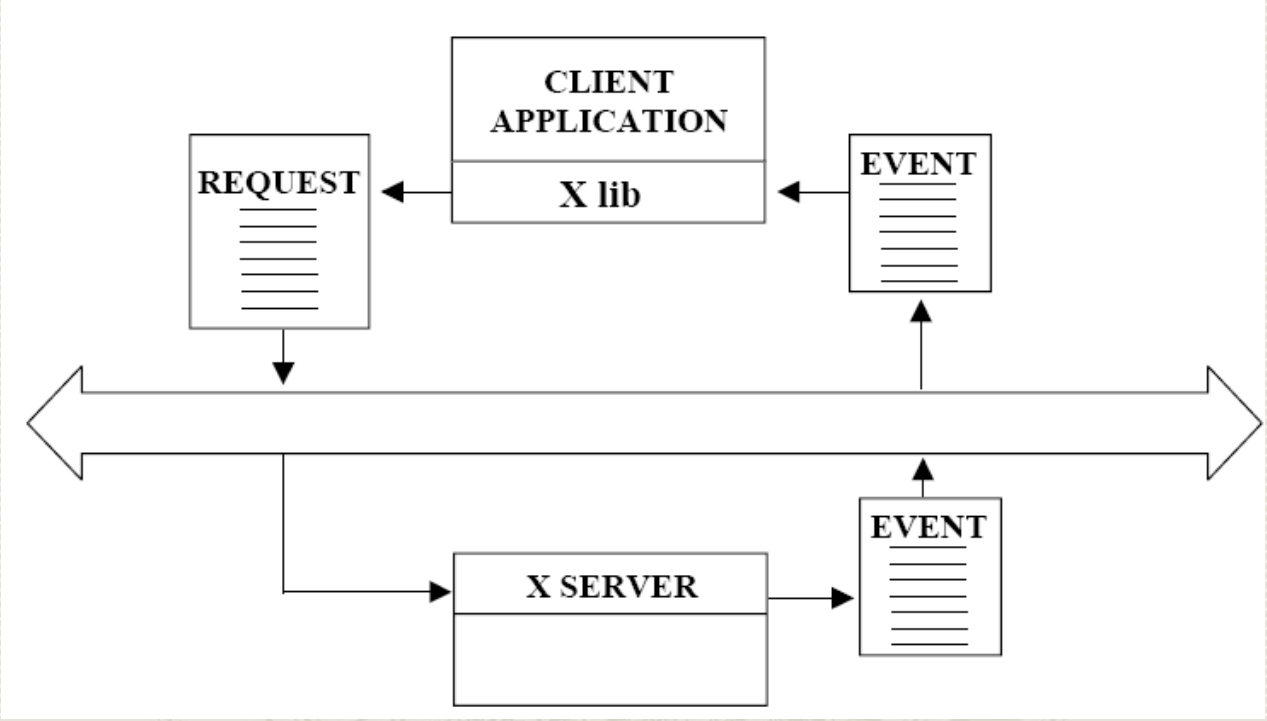
Connessione con ssh ad un server remoto con X

In che ordine si deve procedere?

1. Lanciare l'XServer sull'host locale
2. Aprire una shell (xterm) sull'host locale
3. Eseguire ssh per stabilire una connessione con la macchina remota (comandi visti prima)
4. Dare comandi X sulla macchina remota

Questo permette la visualizzazione dell'output del XClient remoto come se fosse sulla macchina locale

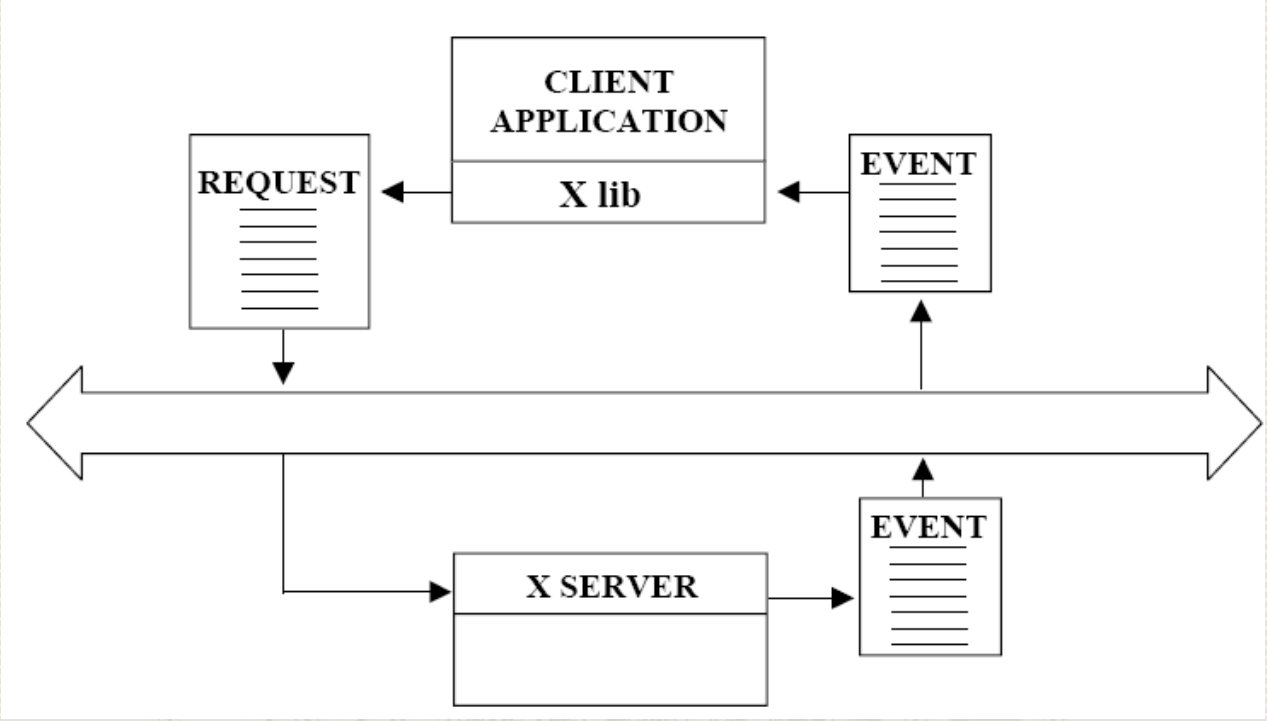
Bufferizzazione



La Xlib svuota i Buffer delle richieste al server sotto tre condizioni:

- Attesa di un evento che ancora non è in coda;
- Richiesta che prevede una immediata risposta;
- Riempimento del Buffer;

Bufferizzazione



Ancora:

- Richiesta esplicita di svuotamento del Buffer:

  Xflush() e/o XSync()

Risorse

XWindow minimizza il traffico sulla rete facendo conservare al server le complesse strutture dati per le risorse.

Una risorsa può essere una:

- Window
- Pixmax
- Colormap
- Cursor
- Font
- Graphics Context
- ….

Il client, per far riferimento ad una risorsa, usa solo un ID.

Proprietà e Atomi

•Una proprietà è un pacchetto di informazioni associate ad una window, resa disponibile a tutti i client che usano quel server.

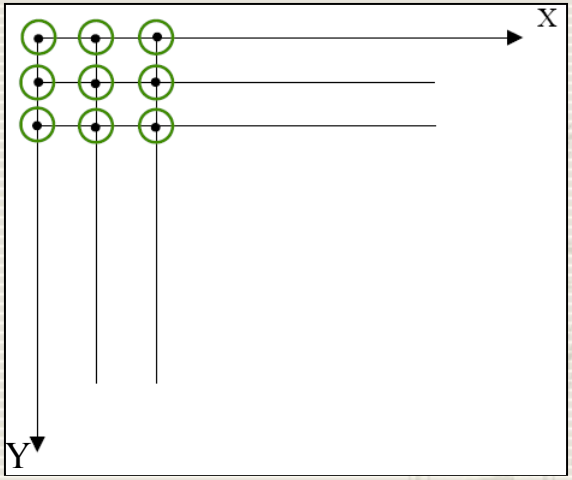
•Le proprietà sono usate dai client per informare altri client e per essere informati sugli altri client.

•Le proprietà hanno un nome ed un ID numerico chiamato Atomo.

Esempio:

Un client usa una proprietà per comunicare e ricevere informazioni dal Window Manager.

Caratteristiche delle Window



Ogni window ha sempre una window genitore.

- Ogni window è interna alla window genitore.
- Una window non può ricevere input se il mouse è fuori dalla sua area.
- L'unica window senza genitore è detta ROOT WINDOW, occupa l'intero schermo ed è creata alla partenza dall' Xserver.

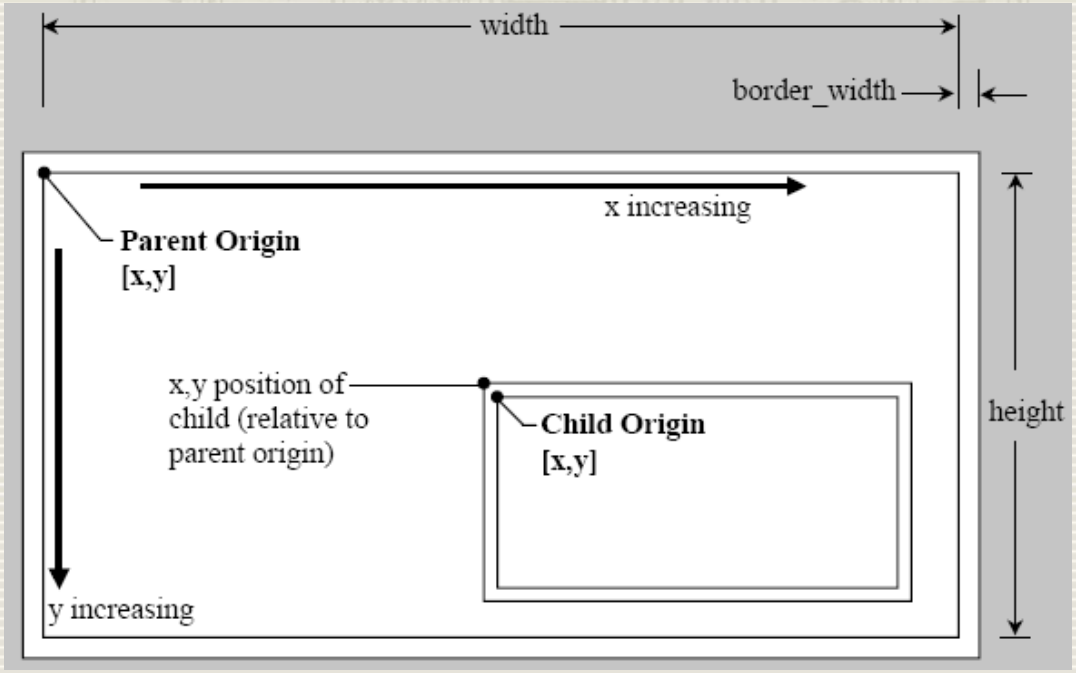
Ogni window ha il proprio sistema di coordinate.

- L'origine è in alto a sinistra e gli assi x ed y puntano a destra e in basso.
- Le coordinate sono intere e coincidono con il centro del pixel.

Caratteristiche delle Window

- Quando una window viene creata le viene associato un ID.
  - Tutte le routine che agiscono su una window usano come argomento l' ID della window.

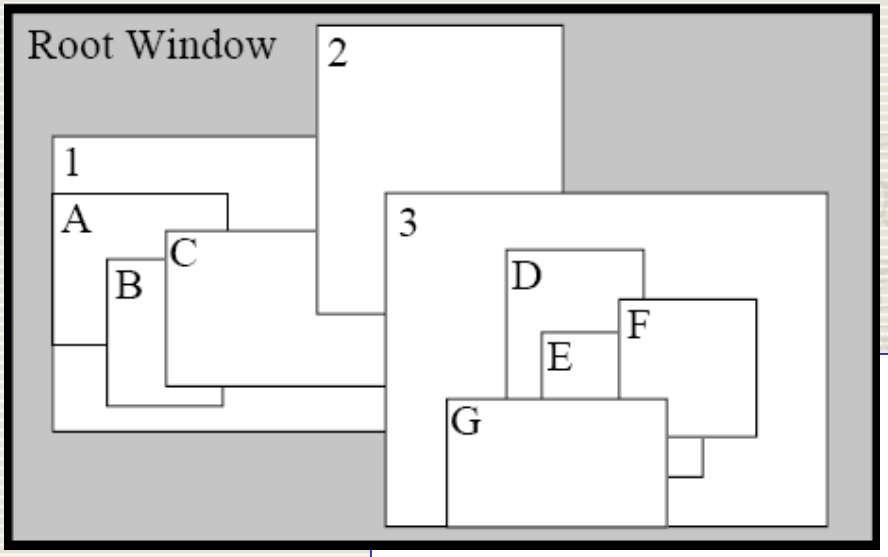
- Una window ha una POSITION, che sono le coordinate del suo vertice alto sinistro, rispetto al genitore: una WIDTH ed HEIGHT in pixel è un BORDERWIDTH.

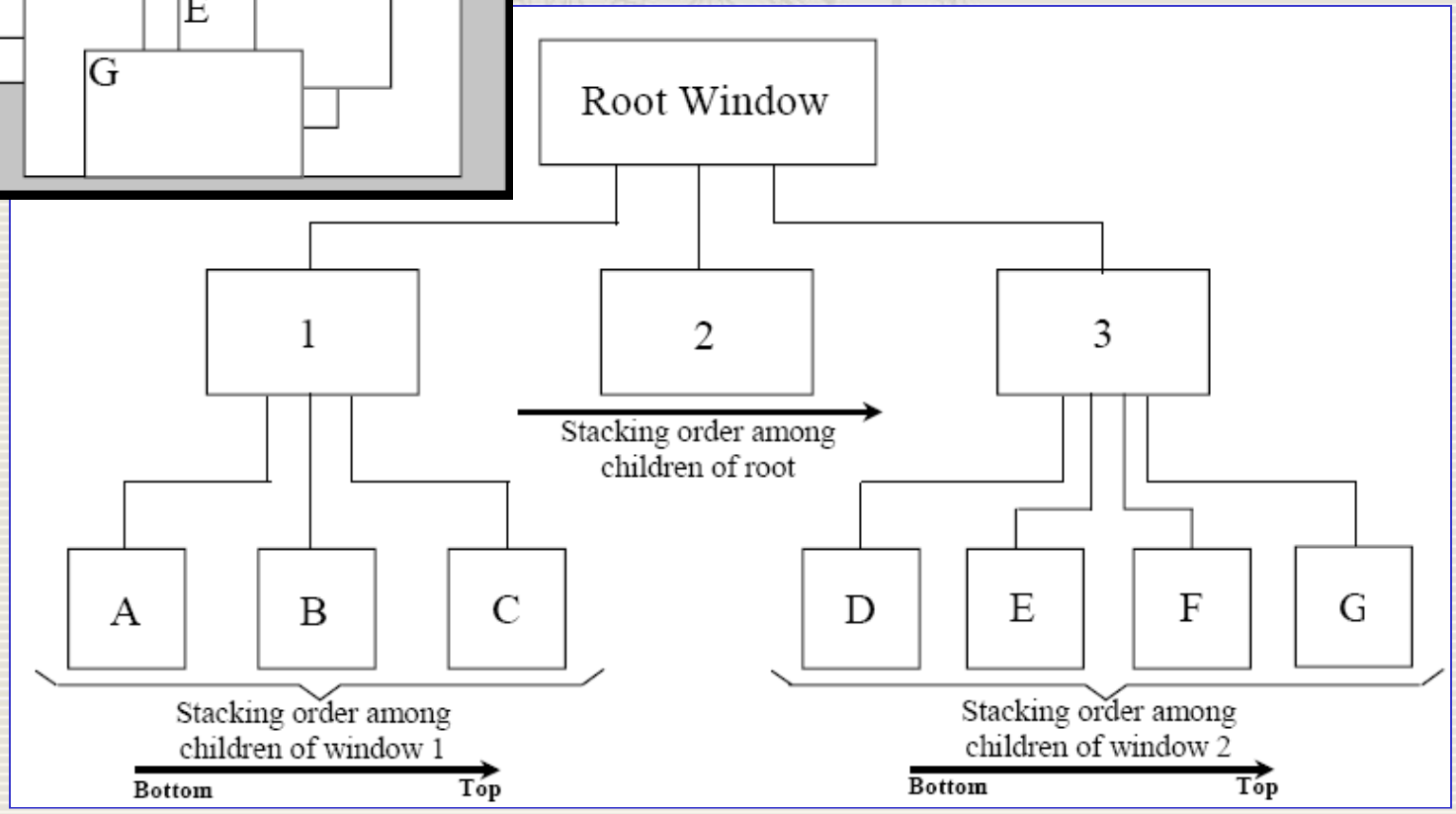


Caratteristiche delle Window

- Window con lo stesso genitore devono avere un ordine nello stack di visualizzazione.
- Queste caratteristiche formano la WINDOW CONFIGURATION, perché influenzano il posizionamento della window sullo schermo.
- Ogni window ha un DEPTH e VISUAL che insieme determinano le sue caratteristiche di colore.
- Una window ha una classe che può essere di INPUT/OUTPUT o solo di INPUT.
- Una window ha un insieme di ATTRIBUTES che controllano molti aspetti della window:
  - colori per il bordo e sfondo
  - Eventi ricevuti (e scartati)
  - Colormap usata per interpretare i valori dei pixel
  - Cursore visualizzato quando il mouse è interno alla window

Gerarchia delle Window





Ordine nello Stack delle Window

- Quando una window interseca altre window aventi lo stesso genitore, quella più in alto nello stack è quella in primo piano e via di seguito.

- Questo ordine può essere alterato chiamando varie routine:

  queste hanno effetto anche sulle window figlie di queste, ma non sui loro genitori.

Mapping e Visibility

Appena una finestra viene creata non risulta visibile.

MAPPING etichetta una window come idonea per essere visualizzata.

Se non è oscurata da window sorelle o da sorelle del genitore può essere disegnata.

Sarà effettivamente visibile quando il buffer viene svuotato.

ATTENZIONE

Si ricorda che l'Xserver, se non viene esplicitamente richiesto (Backing Store attivato), non preserva il contenuto di una window.

Disegno e Graphics Context

XWindow prevede routine per disegnare punti, linee, rettangoli, poligoni, archi, testo, ecc. Le routine di disegno sono dette PRIMITIVE GRAFICHE.

Una risorsa del server, detta GRAPHICS CONTEXT, specifica le rimanenti variabili come larghezza della linea, colore, pattern di riempimento, ecc.

L' ID di un GRAPHICS CONTEXT viene specificato come argomento della routine di disegno.

Eventi

Un client deve selezionare i tipi di eventi che vuole che il server rilevi su ogni window.

TIPI DI EVENTI

- pressione del mouse (3 button)
- movimento del mouse
- focus in/out (tastiera)
- in/out del mouse
- expose
- ….