Unix Window Systems

Class Meeting 5

*Notes adapted by Doug Bowman from previous work by other members of the CS faculty at Virginia Tech

(C) 2005 Doug Bowman and Virginia Tech CS Dept.

Why Window Systems?

- A window system provides a graphical user interface (GUI) based on windows, icons, and event-driven interaction
- Increased usability due to
  - Visibility
  - Direct manipulation
  - "Knowledge in the world"

(C) 2005 Doug Bowman and Virginia Tech CS Dept.

Window Systems and Unix

- Unix evolved before window systems and optimized use of the command line
- Modern Unix systems include a window system to combine the advantages of the window system with the availability of a command line for expert use

(C) 2005 Doug Bowman and Virginia Tech CS Dept.

X Windows

- Practically all Unix window systems are based on X Windows (XFree86)
- Standard Version: X11R6
- X:
  - Interfaces with I/O hardware (display, mouse, etc.)
  - Manages the screen space
  - Draws simple graphics
  - Assigns rectangular regions to programs

(C) 2005 Doug Bowman and Virginia Tech CS Dept.

The X Client-Server Architecture

- X designed to work over a network
- X server: software that runs on the machine where the program’s output will be displayed
- X client: program running on the same or another machine
- Client sends drawing and other X commands to the server, which displays the results

(C) 2005 Doug Bowman and Virginia Tech CS Dept.

Historical Use of X

- User sat at an X terminal – graphical terminal that ran an X server, but no OS
- User logged into remote computer running UNIX or other OS supporting X clients
- Separates graphical interface and manipulation from application
- Combine applications running on multiple computers

(C) 2005 Doug Bowman and Virginia Tech CS Dept.
Features of X
- Transparent remote execution
- Gives each program its own virtual screen
- Includes important windowing concepts
  - Window damage
  - Window reveal events
  - Backing store
- X11 programs are highly portable

Window Manager
- Window manager runs on top of X11 (not part of X11 itself)
- Places borders, sliders, and other widgets on windows to provide the interface look and feel
- Routes output from X clients
- Routes input from users
- Examples:
  - kwin — default for KDE
  - metacity — default for GNOME
  - mwm — Motif standalone window manager

Desktop Environment
- Desktop environment organizes display into an integrated environment
- Another layer on top of window manager
- Includes file managers (Nautilus), icons, panels, configuration tools, system-wide menus, etc.
- GNOME (GNU)
  - built on GIMP Toolkit (GTK+)
- KDE (uses Qt C++ libraries)