Lifecycle of an X11 Window

Spring DECUS Symposium 1993

❑ Window Tree Validation

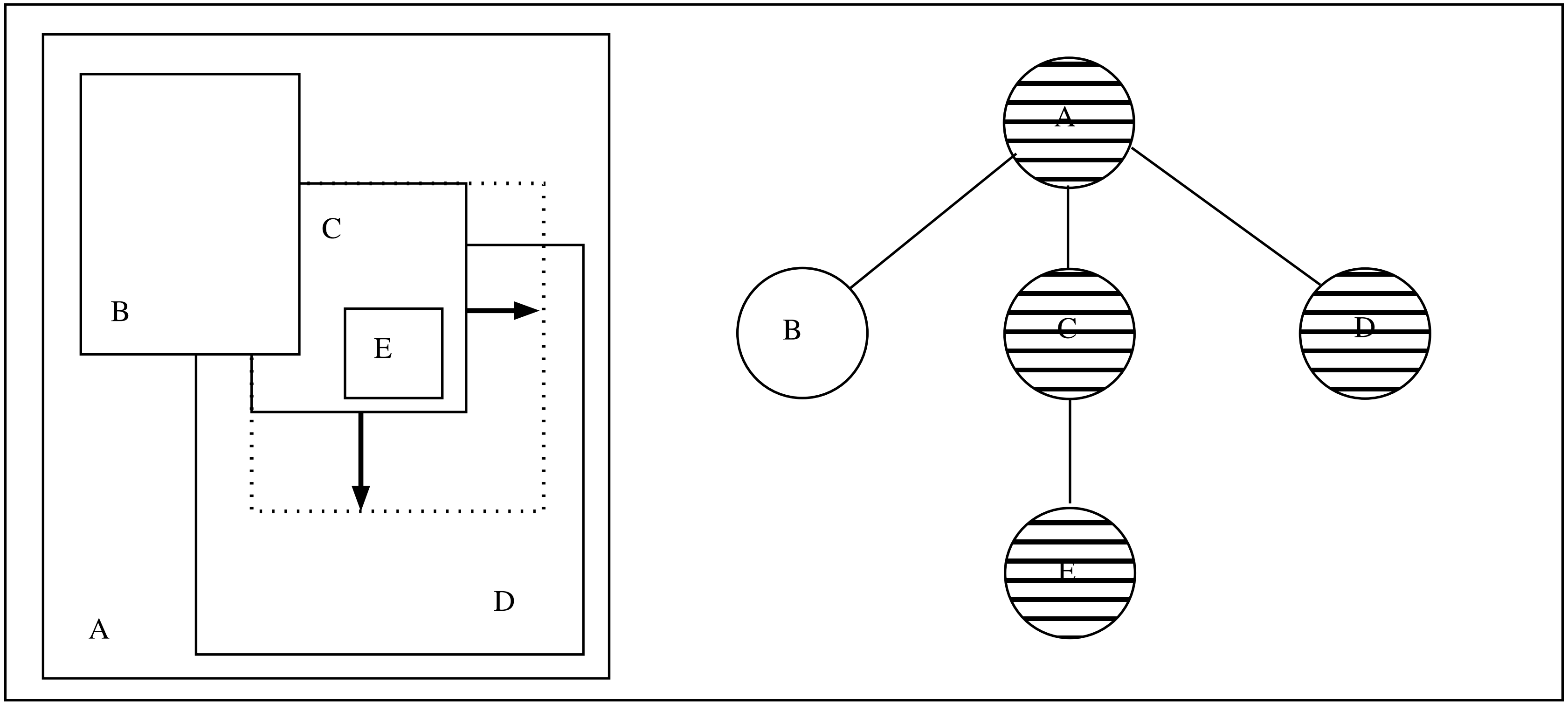
- Everytime a window is modified the server must determine which windows are affected and modify their clip regions
- Insures proper overlap of windows
- Drawing operations effect only the target window
- Implemented as part of the DDX

Window Tree Validation

- Ignores unmapped windows
- Only affects the branch of the Window Tree in which the modified window resides
- Only affects the windows lower in the stacking order between siblings

Lifecycle of an X11 Window
Spring DECUS Symposium 1993

- **Window Tree Validation Example**



❑ Window Tree Validation

- Occurs for the following reasons
  - A New window has been mapped
  - A window has been unmapped
  - A window has been moved
  - A window has been resized
  - Stacking order between siblings has been changed

❑ Window Tree Validation

- Computes the new clip list for the window
- Causes an Expose event to be sent to the client if need
- Invalidates the current Graphics Contexts
- If DDX requests to be called, then the DDX routine for handling Clip notifications is called

❑ Moving a window

- Clients request moving a window
  - via XMoveWindow()
  - via XConfigureWindow()

❑ ConfigureWindow Request

```
struct configureWindow {
        unsigned char  opcode;     /* this value is 12 */
        unsigned char  pad;        /* unused */
        unsigned short length;     /* length of this request */
        unsigned long  id;         /* window affected */
        unsigned short mask;       /* What changed */
        unsigned short pad2;       /* more unused */
};
```

ConfigureWindow Request

- When moving a window
  - *Mask* indicates that the values passed in are (x,y)
  - (X,Y) position is then next two longwords past the end of the ConfigureWindow request

Moving a window in the Server

- ConfigureWindow Request is dispatch to ProcConfigureWindow() (byte-swapping as needed)
  - Length of packet must be >= 12
  - Window Id is validated
- ProcConfigureWindow() in turn calls ConfigureWindow()
  - Old position of window is saved
  - New position is calculated in screen coordinates
  - Window is marked for validation
  - PositionWindow() routine is called in the DDX layer

❑ Moving a window in the server

- All of the window's children are moved
- Validate the Window Tree starting at the Parent of this window
- Handle exposures generated by this request

Lifecycle of an X11 Window
Spring DECUS Symposium 1993

❑ Destroying a window

- Client requests deleting a window
  - via XDeleteWindow()

❑ DestroyWindow Request

```
struct destroyWindow {
        unsigned char   opcode;     /* this will be 4 */
        char            pad;        /* unused */
        unsigned short  length;     /* will be 2 longwords */
        unsigned long   id;         /* resource id(*) */
};
```

* This can be a Window, Pixmap, Font, GC, etc

❑ Destroying a window in the server

- DestroyWindow Request is dispatch to ProcDestroyWindow() (byte-swapping as needed)
  - Length of Packet must = 8 bytes
  - Window Id is validated
  - Window is unmapped, removed from Window Tree and associated data structures are freed

Lifecycle of an X11 Window
Spring DECUS Symposium 1993

❑ Further exploration

- Documentation
  - X Window System Server by Elias Israel & Erik Fortune, Digital Press
  - X Window System by Robert Scheifler, et. al., Digital Press
- Source Code
  - MIT X11R5 release*
  - OpenVMS 5.5-2 Source Listings Kit

* available from a wide varity of sources around the world

Lifecycle of an X11 Window
Spring DECUS Symposium 1993

- **Questions & Answers**