New Evolutions in the X Window System

Matthieu Herrb and Matthias Hopf

Part 2: Xgl

Matthias Hopf
SuSE Labs

suse
A NOVELL COMPANY

Novell®

What is Xgl?

Xgl is:

- Xserver using OpenGL for its drawing operations
- Created mainly by David Reveman
- Prototype presented at X.org developer's conf. 2005

Xgl is ***not***:

- A display mechanism using different protocol or API
  - Still X11 on client side
- Accelerating OpenGL programs
  - Currently slower (indirection, composition manager)

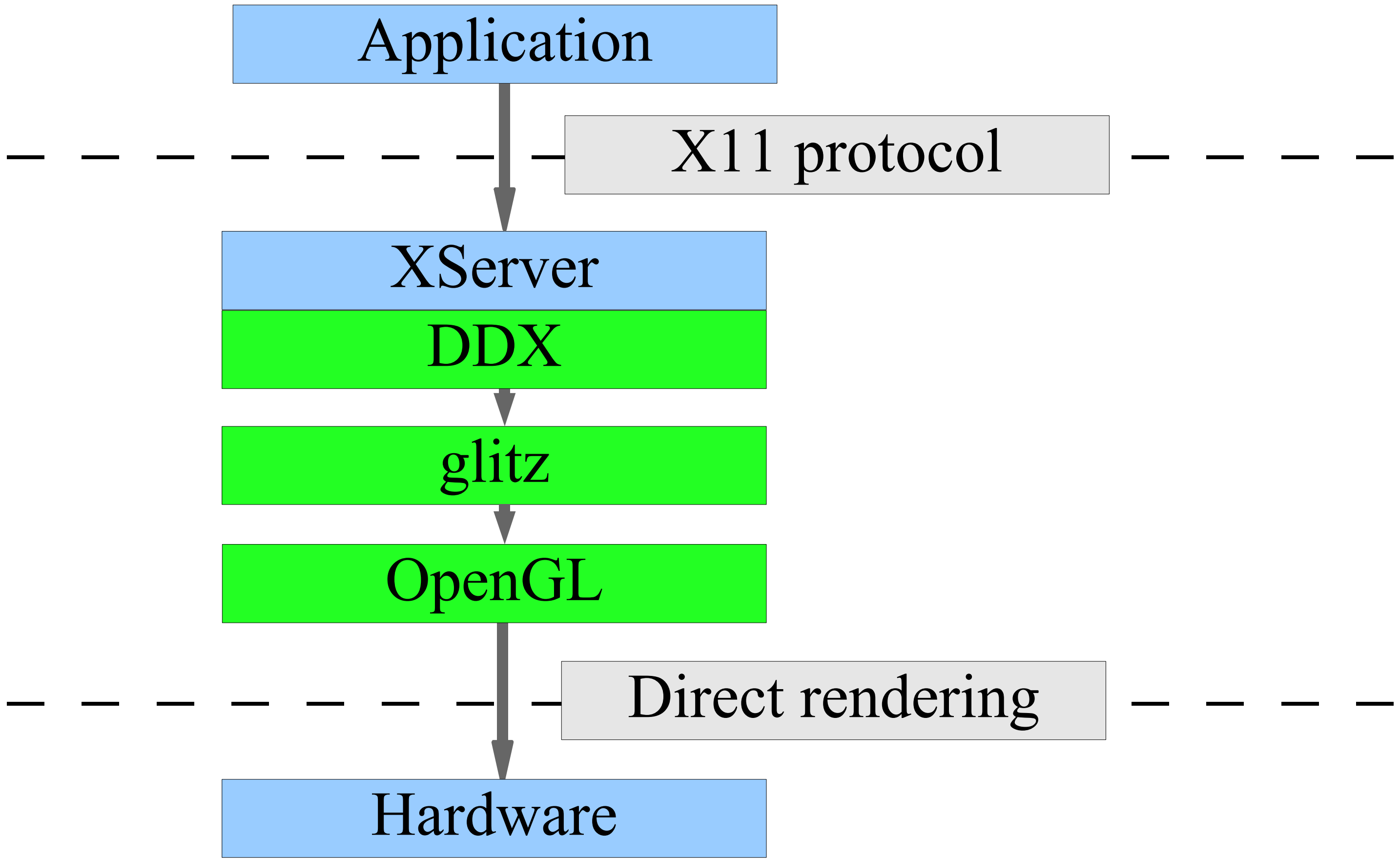
Why OpenGL?

- XAA API considered difficult to extend
  - does not match current rendering use
  - No support for modern gfx hw features
  - API used by Xserver only
- No more hardware drivers inside X server
  - Code separation
  - Run completely unprivileged
- 3D hardware is faster
- Render + Composite accelerated independent of graphics hardware
- Future hardware won't have 2D acceleration
- Provides 3D interface for enhanced desktops like Looking Glass (see Quartz, Avalon)

Flavors

- Xgl
  - Frontend that loads backend plugins
  - Dynamic loading to avoid name space collision
  - Currently based on KDrive Xserver, inclusion in Xorg after modularization
  - Using glitz as mid-level abstraction of OpenGL
- Xglx
  - Similar to Xnest, run in a window
  - Mostly worked on ATM
- Xegl
  - Standalone
  - Early implementation for Radeon R100 & R200

Big Picture


Application
X11 protocol
XServer
DDX
glitz
OpenGL
Direct rendering
Hardware

Hardware Drivers

- libGL.so provides output drivers for Xgl
- Extensions needed for
  - Mode setting (EGL_MESA_screen_surface + others TBD)
  - Offscreen rendering (pbuffers + frame buffer objects)
  - Filtering + color conversion (pixel shader)
  - Hardware cursors (TBD)
  - Address space sharing (TBD):
    needed for running direct rendering applications
- Input
  - Under discussion
  - Most likely to use future Xorg mechanisms

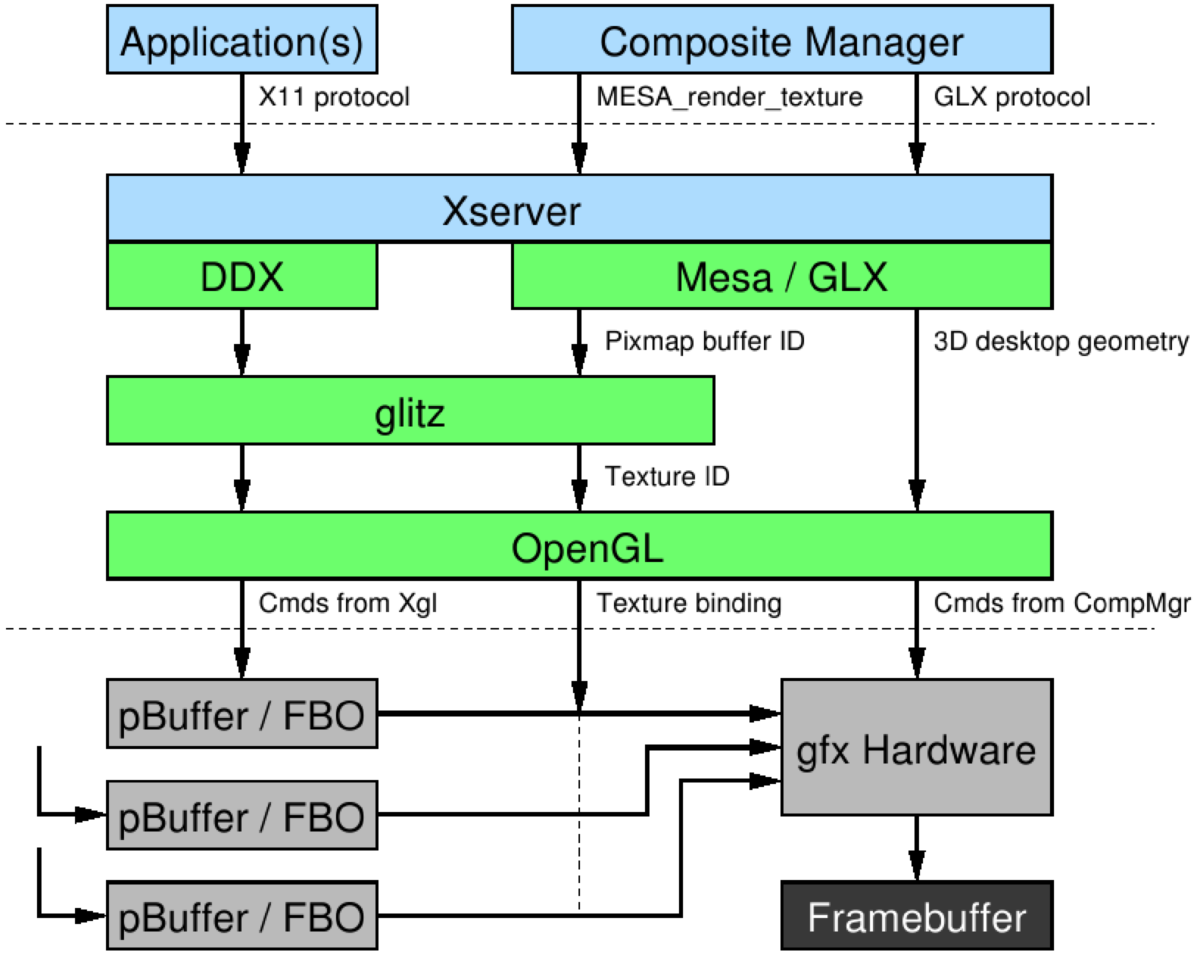
glitz

- OpenGL acceleration for Cairo
  - Abstraction of pixel formats, buffers, geometry, clipping, filters, operators
- Accelerates almost everything of Render extension
- Using textures for Pixmaps
- No software fall-backs

DDX

- Software fall-backs for everything not accelerated
- Accelerated so far:
  - Core: solid and tiled fills, copy area, core fonts
  - Render: trapezoids, glyphs, convolution filters, ...
  - GLX: 1.2
- Unlikely / difficult to be accelerated:
  - Core: Logical ops, plane masks
  - Render: PictOpSaturate
- Only indirect rendering for OpenGL applications
  - Off-screen framebuffer has to be in same address space as textures for composition manager
  - Seamless integration with Composite

Compositing Overview



Compositing

- Rendering to off-screen framebuffer per window
- Composition manager renders these on screen

  -> Effects like transparency etc.
- Windows in 3D space

  -> Composition manager using OpenGL: glxcompmgr
- Bind Pixmaps to textures:
  GLX_MESA_render_texture + indirect rendering
- Currently only supported in Xgl
- Pixmaps stay in graphics memory all the time

Issues

- More acceleration
- Some rendering modes unsupported / wrong
  - Texture storage format is static, pixmaps are not -> Only one picture format per depth
  - Source picture clipping
  - Clamp to transparent for pictures without alpha
- Direct rendering support
- Memory management
- Most important: drivers quality

Near to Mid-term Future

- Disclose source
- Move Xgl into X.org
- Xegl improvements
- Stabilization + more acceleration
- Integration into desktops + window managers
- Getting vendors to support EGL_MESA_screen_surface
- Work on other extension specifications
- First attempts to think about scene graph alike API

BSD related

- Xgl should run on any system providing OpenGL acceleration for X windows
- Unusable without hardware acceleration (DRI)
- Xegl needs OpenGL + OpenGL ES in the console driver

- One big hurdle: availability of open source Embedded OpenGL drivers for existing cards

Conclusion

- X development has resumed
- Will provides enough features for modern desktop environments
- Moving to OpenGL based acceleration
- Challenges remain for Open Source systems

-> Support the X.Org foundation, not only your favourite BSD ;-)