compiz:
The Next Generation Desktop

Matthias Hopf

SUSE R&D / Novell

Overview

- All UN*X desktops use X11
- More than 20 years old
- A lot of new development started after XFree86 → X.org fork
- Enhancement by extensions (like in OpenGL)

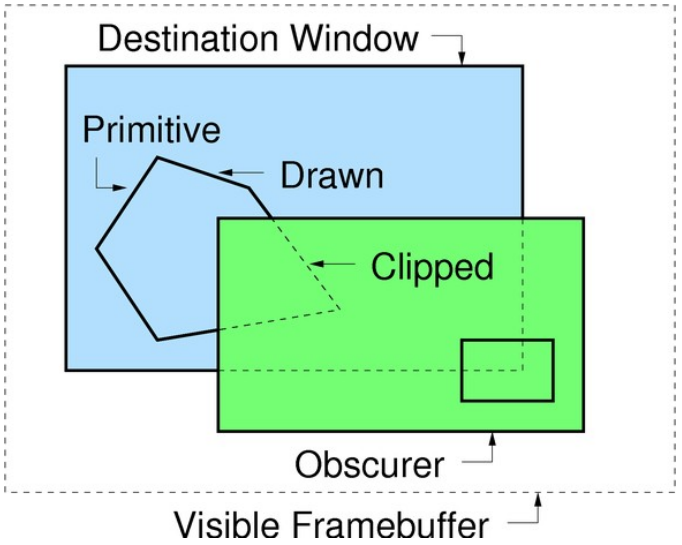
Components of a Desktop

- Window manager
  - Window handling
  - Virtual desktop policy
  - Effects & eye candy
- Desktop, panel, filemanager
- Applets
- Applications
- Libraries & conventions for
  - Configuration, cut&paste, drag&drop
  - Toolkit, look&feel

→ compiz is actually only one part of a desktop

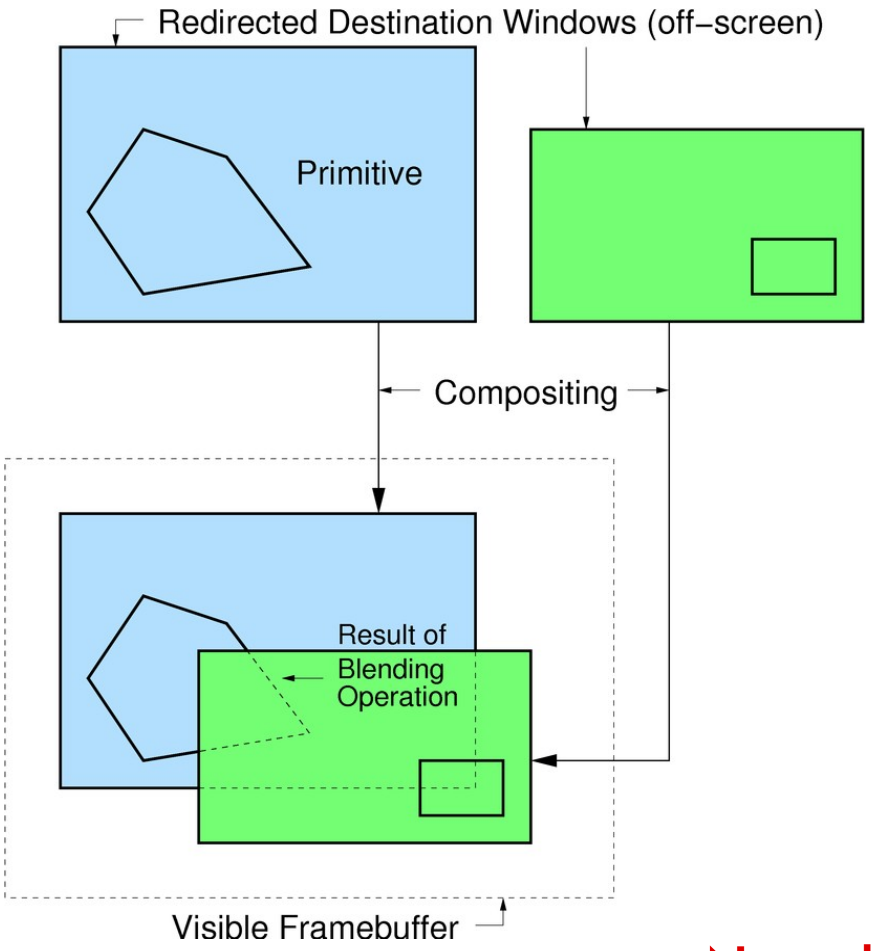
The Non-Composited Desktop

- Desktop rendering paradigma remained the same for >20 years:
  - Create primitives from rendering commands
  - Clip primitives against obscurers
  - Render remaining partial primitives to the framebuffer
- Has several fundamental problems
  - No semitransparency
  - Window contents cannot be transformed
  - Update flicker



The Composited Desktop

- Render windows to off-screen buffer
- Composite windows to screen on a regular basis
- Gives you window transformations, cross-window transparency, thumbnails
- Updates can be made flicker free



Composition Manager

- Compositing can be done by a user process (CM)
  - Controls where Window content is displayed
  - May apply transformations
    - Alpha blending → semitransparent windows
    - Bilinear and projective transformations
    - Color space conversions
    - Transition effects
    - Scaling: Magnifying & thumbnailing
- Composite with modern rendering techniques
  - Render
  - OpenGL

Necessary X11 Changes

- **Render**: New primitives & rendering model
  - New graphics primitives
    - Better suited to modern toolkits
    - Semitransparent glyphs → antialiased fonts
    - Polygons, projections, filters, gradients...
  - Porter & Duff compositing
    - Finally semitransparent primitives
- **Compose**: Redirect windows to off-screen pixmaps
- **Damage**: Tracking window content changes in user space

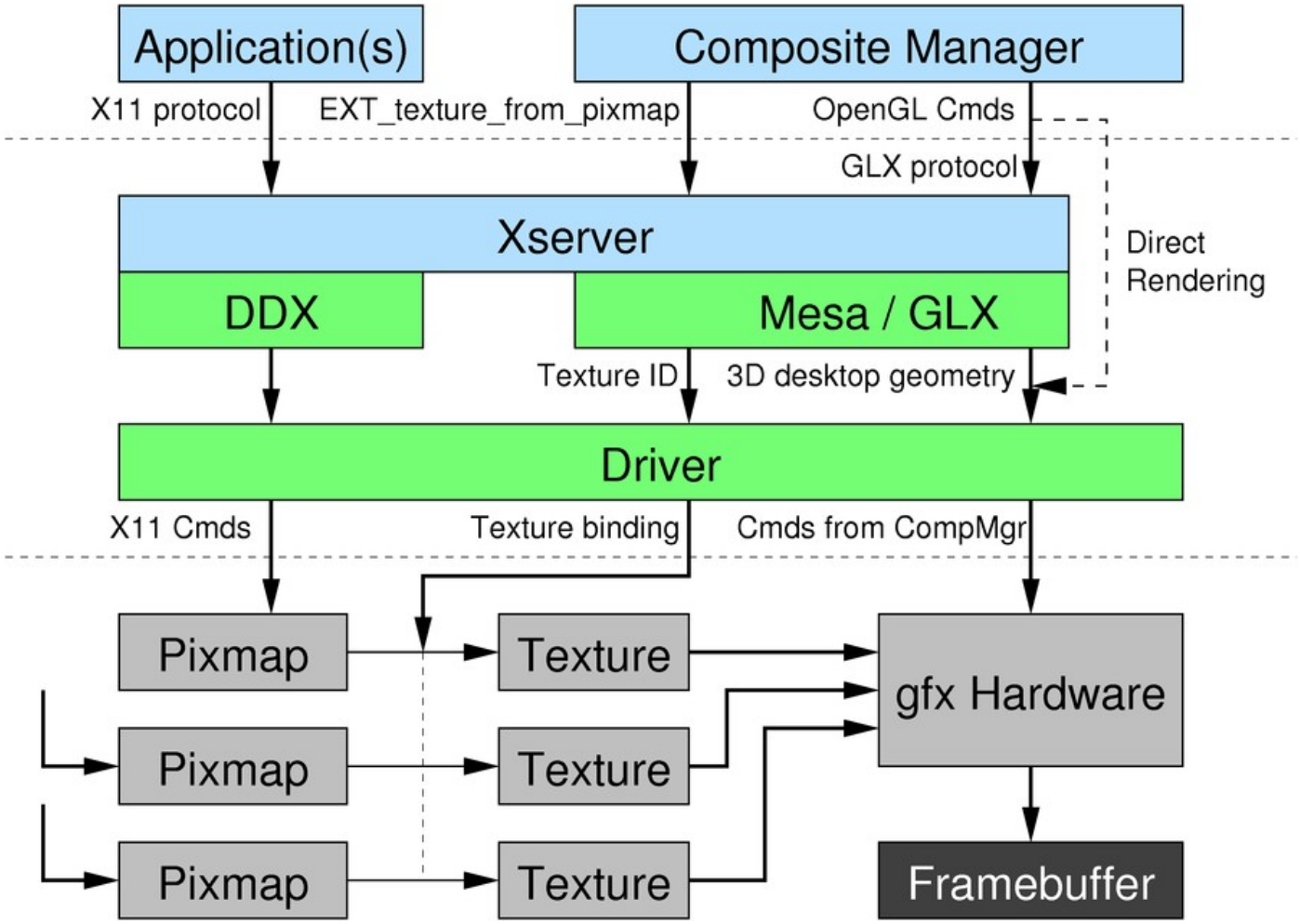
Compositing Window Manager

- Compositing doesn't 100% match X model
  - Window contents are RGB, but decoration is RGBA
- Many effects only possible in tight integration with window manager
  - Minification
  - Application switcher
  - Exposé
  - Virtual desktops
- compiz
  - Most advanced compositing window manager so far
  - Using OpenGL for compositing
  - Flexible plugin architecture

OpenGL Based Composition

- Windows in 3D space
  - Allows for more flexibility
  - Great transition effects
- Requirement 1: EXT_texture_from_pixmap
  - Bind off-screen pixmaps with window contents to texture for compositing
- Requirement 2: texture space in pixmap space
  - Same address space for rendering window contents and compositing
  - Pixmaps stay in graphics memory all the time
  - Typically implemented by indirect rendering
- Supported by Xgl, AIGLX, NVIDIA's driver

Compositing Model Overview



What is Xgl?

Xgl is:

- Xserver using OpenGL for its drawing operations
- A new acceleration architecture for X
- First presented at X.org developer's conf. 2005
  - Created mainly by David Reveman
- The first implementor of EXT_texture_from_pixmap

Xgl is ***not***:

- A display mechanism using different protocol or API
  - Still X11 on client side
- Accelerating OpenGL programs
  - Currently even slower (indirection, composition manager)
- Doing the cool transition effects
  - That is compiz' job

What is AIGLX?

AIGLX is:

- **A**ccelerated **I**ndirect **GLX** protocol
  - Hardware acceleration for indirect rendering over X11 protocol (may be on remote computer)
- Solving a long-standing issue
- Providing EXT_texture_from_pixmap
- Considered cleaner than Xgl (no second Xserver)

AIGLX is ***not***:

- Comparable to Xgl
  - No new acceleration architecture (XAA, EXA, glucose)
- The first solution to solve the indirect rendering issue
  - NVIDIA's driver did that for a very long time

What is about NVIDIA's driver?

NVIDIA is:

- Accelerating indirect OpenGL for a long time
- Providing EXT_texture_from_pixmap
- Allowing direct rendering even for these textures

NVIDIA is ***not***:

- GPL compatible
  - At least many kernel developers think so

Flavors and History of Development

- Upstream compiz in freedesktop.org's git
  - Focuses on changes in the base architecture
  - David Reveman commits here
- Compiz-Quinn
  - Semi-fork that follows upstream development
  - Testbed for experimental patches
  - Only community-driven development
  - Several patches contradict main developers' view and don't fit the plugin model well
  - Licensing
- Beryl
  - Name change after Compiz-Quinn forked fully
  - Active community, but few core changes

Flavors and History of Development (2)

- Reunification
  - compiz remains upstream version
  - Some patches ported over from beryl
  - Very active development now
  - Beryl plugins ported to upstream compiz, currently forming a compiz-extra plugin package
  - New “Composite Community” web site, forum
- Still quite some way to go...

Practice: Running Xgl

- Use it like Xnest: run in a window
- Xgl :1
  - -fullscreen
  - -accel window:pbuffer
  - -accel glx:pbuffer
  - -accel xv:fbo

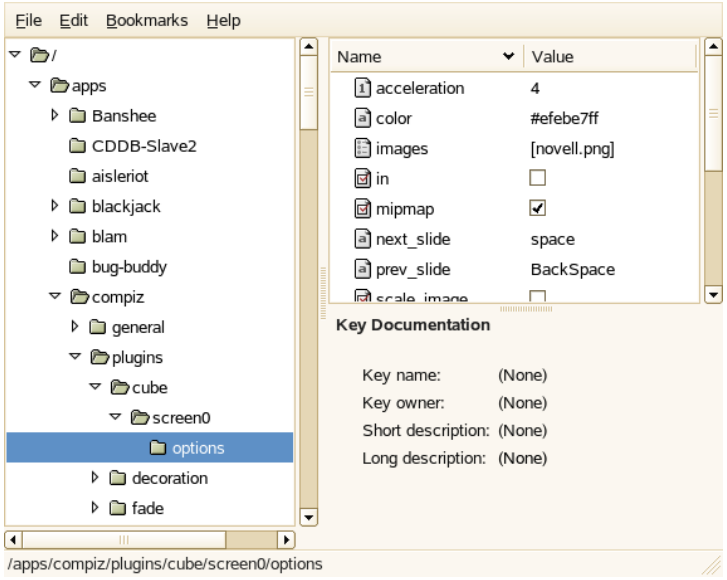
- As a X.org “replacement”:
  - ln -snf /usr/bin/Xgl /var/X11R6/bin/X
  - Xgl automatically starts Xorg underneath
  - Have to make sure acceleration options are passed

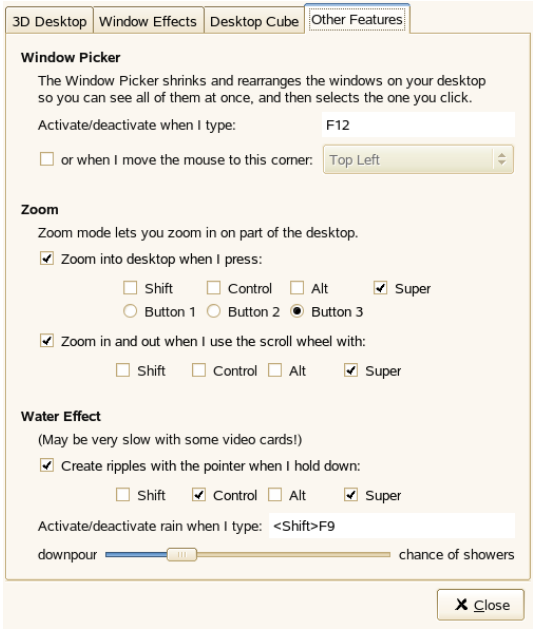
Running compiz

- Use it like a regular window manager
- Or substitute after login:
- compiz --replace gconf &
  - Replaces running window manager
  - gconf & ccp modules autoload configured modules
- {gtk,kde}-window-decorator &
  - Work independently from desktops
  - Additional window decorator from community
- More info on web / Wiki:
  - www.compiz.org

Configuring compiz

- Low-level: gconf-editor
  - Almost everything inside plugins
  - Fine-grained access



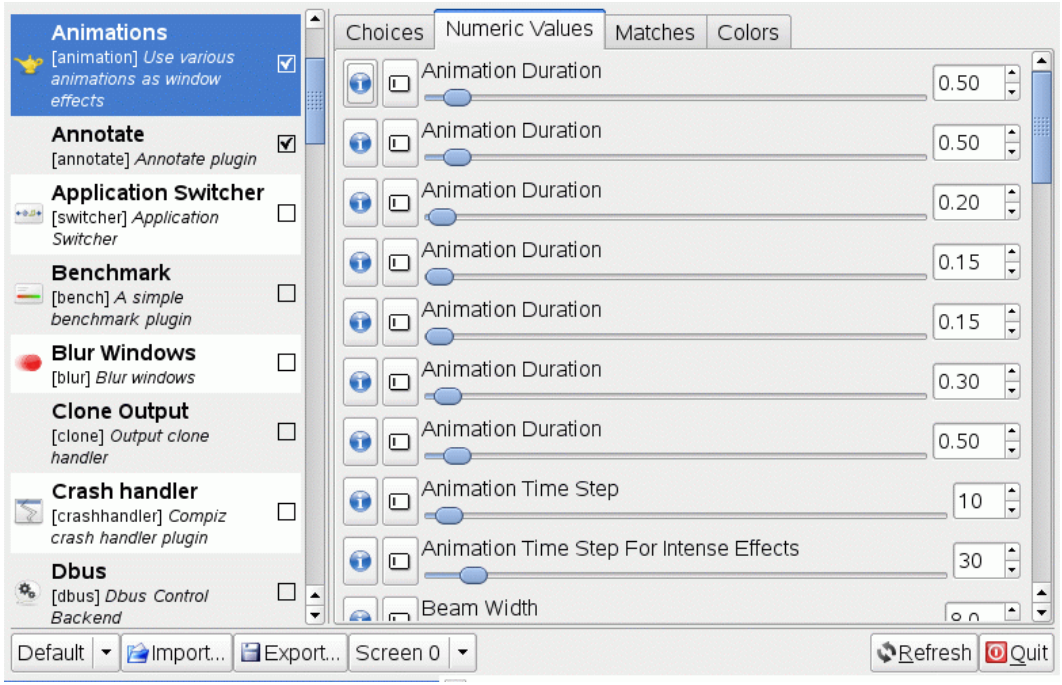


- High-Level tools
  - gnome-xgl-settings

Novell

Configuring compiz (2)

- Community based configuration system:
  - ccp plugin
  - ccs-settings
  - Currently only way to configure some community plugins



What Needs To Be Improved

- X.org
  - Dynamic reconfiguration
    - RANDR 1.2 (drivers), input hotplugging (infrastructure)
  - Color management
  - Accessibility
  - 3D Driver stabilization
- compiz
  - Input transformations, software cursor (being worked on)
  - Retained mode, video interface (prototype)
  - Integration into desktops (still a bit rough), drawing synchro
  - More plugins with real functionality (not eye candy) needed
  - Port to XCB instead of Xlib
  - Stability

Nothing's Like the Real Stuff...

Demo