

# P00998 - Dissertation proposal

## Writing a X compositing manager

Arnaud Fontaine (08090091)

Supervisor: Faye Mitchell

13 March 2009

### Contents

<b>1</b>	<b>Introduction and rationale</b>	<b>1</b>
<b>2</b>	<b>Background review</b>	<b>2</b>
2.1	Initial literature review . . . . .	2
2.2	Existing approaches and equivalent software . . . . .	3
<b>3</b>	<b>Methodology and resources</b>	<b>3</b>
3.1	Software, hardware and expert help needed . . . . .	4
<b>4</b>	<b>Project plan</b>	<b>4</b>

# 1 Introduction and rationale

My dissertation is about writing a free software *compositing manager* (Wikipedia 2009-02-25) relying on the X Window System protocol (Wikipedia 2009-02-24). Basically, a compositing manager is a piece of software above the window manager where each graphical program outputs into a separate and independent buffer that can be manipulated before being shown in order to improve user experience. Unlike a *compositing window manager*, a compositing manager does not manage windows but simply implements visual effects such as window translucency, shadows, fading. . . Nowadays, major operating systems provide such technologies, for instance Quartz (Wikipedia 2009-01-28) compositor on Mac OS X, CompizFusion (Wikipedia 2009-03-09a) on Unix-like operating systems, or DWM (*Desktop Window Manager* (Wikipedia 2009-02-19)) on Windows Vista.

More precisely, this project aims to write an efficient, lightweight and responsive compositing manager in C programming language, based on the XCB<sup>1</sup> (*X protocol C-language Binding*) client library for the X protocol. The latter library is available on Unix-like operating systems using the X Window System (usually X.Org<sup>2</sup> based operating systems), thus this project will benefit any of those systems and existing window managers based on that system. The resulting program will be written in a modular way, so that the core code will implement basic primitives whereas more complex features and effects will be implemented as plug-ins, thus targeting platforms from desktop systems to embedded devices. This project also aims to stick to a functional and stable software to improve window manager usability from an end-user point of view (for instance window translucency, application-switcher using live previews instead of just icons, all windows function as seen in *Exposé* (Wikipedia 2009-03-09b). . .), therefore it does not intend at all to provide useless eye-candy effects.

This subject is not tightly related to a particular module taught within MSC program, but it lies on topics address in BSC and MSC modules such as *Software Production* (quality insurance and software testing), *System Programming* (low-level and C programming) and *Computer and Network Security* (writing secure programs in C programming language). Moreover, this

---

<sup>1</sup><http://xcb.freedesktop.org/>

<sup>2</sup><http://www.x.org/wiki/>

project is an extension of my last year Undergraduate project which was about porting a window manager, `Awesome`<sup>3</sup>, to the XCB client library.

I'm particularly interested in this project because I need a stable compositing manager along with my window manager, `Awesome`. Moreover, I really enjoyed working with XCB as my Undergraduate project because I do like low-level programming, particularly in C programming language, while writing a graphical software.

## 2 Background review

### 2.1 Initial literature review

There is no particular documentation describing explicitly how to write a compositing manager, however there are a lot about the architecture of existing software, for instance about Quartz (Wikipedia 2009-01-28, Siracusa 2005-04-28, Apple 2008-10-15) which is an interesting project as it is a fast, complete and widely used compositing manager as part of Mac OS X.

The X Window System provides extensions along with their specification for accessibility and eye-candy effects (Keith Packard 2000-06-15):

- `Composite` (Keith Packard 2007-07-03)
- `Damage` (Keith Packard 2007-01-08)
- `Xfixes` (Packard 2006-12-14)

Along with these extensions, there is also a need to paint the *composed* (or combined) image which may be achieved by using either `Render` (Packard 2005-07-01) or `OpenGL`. (Fedora 2008-05-24) describes the trade-offs involved in both approaches (the former approach seems less efficient than the latter one as it does not take fully advantage of the graphic card possibilities).

---

<sup>3</sup><http://awesome.naquadah.org>

## 2.2 Existing approaches and equivalent software

At the moment, apart from existing compositing window managers such as `CompizFusion` which does not achieve the same purpose than my project, there is only `xcompmgr`<sup>4</sup> written with the old X client library, namely `Xlib`. But, it was developed as a proof-of-concept and it is quite slow because it does not take enough advantage of modern graphic cards possibilities beside of being buggy as well.

## 3 Methodology and resources

Firstly, I have to read in depth papers about compositing manager architecture (Fedora 2008-05-24, Packard 2000-04-19, Keith Packard 2000-06-15, Siracusa 2005-04-28, Apple 2008-10-15) in order to understand the trade-offs between approaches with `Render` or `OpenGL`. Then, I also have to read all the documentations about the X extensions (Packard 2005-07-01, Keith Packard 2007-07-03, Keith Packard 2007-01-08, Packard 2006-12-14) and look for existing libraries that might make development easier. As an example, `Glitz` (Peter Nilsson June 27 July 2, 2004) is a library matching `Render` extension API but it uses `OPENGL` which seems a better approach from a performance point of view. Reading code of `xcompmgr` and `CompizFusion` (both free software) would be valuable as well when trying to understand basic mechanisms.

Secondly, I will run benchmarks on `xcompmgr` to identify the performance bottleneck and then define an appropriate architecture for the project.

Thirdly, I will implement the chosen architecture and write unit tests to ensure the quality of the code. Writing this project will allow to test `XCB` extensions mentioned earlier that have not been so much tested yet. In addition, I will write additional code to complete features I need for my project within the `XCB` util library, namely `xcb-util`, as I did for my Undergraduate project. I will also test the software on several different graphic cards to make sure it run flawlessly almost everywhere.

Finally, the last bit of the programming part will be to run a C profiler in order to identify performance bottleneck and then define a plug-in architecture.

---

<sup>4</sup><http://freedesktop.org/wiki/Software/xapps>

### 3.1 Software, hardware and expert help needed

As this program will be written in C programming language following the C99 standard, only an Unix-like operating system with X Window System set up and a C compiler is needed along with tools to build the project of course. I will have to test my software on different graphic cards (Nvidia, ATI and Intel at least) to ensure that it runs flawlessly on most configurations. If I need help during this project, I may ask for help from mailing lists, such as XCB<sup>5</sup> and xorg<sup>6</sup>, where I may find skilled X developers.

## 4 Project plan

1. Read papers about compositing manager architecture, look for relevant libraries to re-use, understand `xcompmgr` code, and read `CompizFusion` code if necessary.
2. Perform benchmarks on `xcompmgr` to identify design issues and define the basic architecture of the program.

#### **First deliverable.**

3. Write code for `xcb-util`.
4. Check whether XCB extensions used match the X specifications.
5. Write core program.
6. Write documentation and unit tests, fix bugs.
7. Test on different hardware and fix bugs.

#### **Second deliverable, intermediate report and presentation.**

8. Improve performance and design a plug-in architecture to allow loading of external plugins.

#### **Third deliverable and final report.**

---

<sup>5</sup><http://lists.freedesktop.org/mailman/listinfo/xcb>

<sup>6</sup><http://lists.freedesktop.org/mailman/listinfo/xorg>

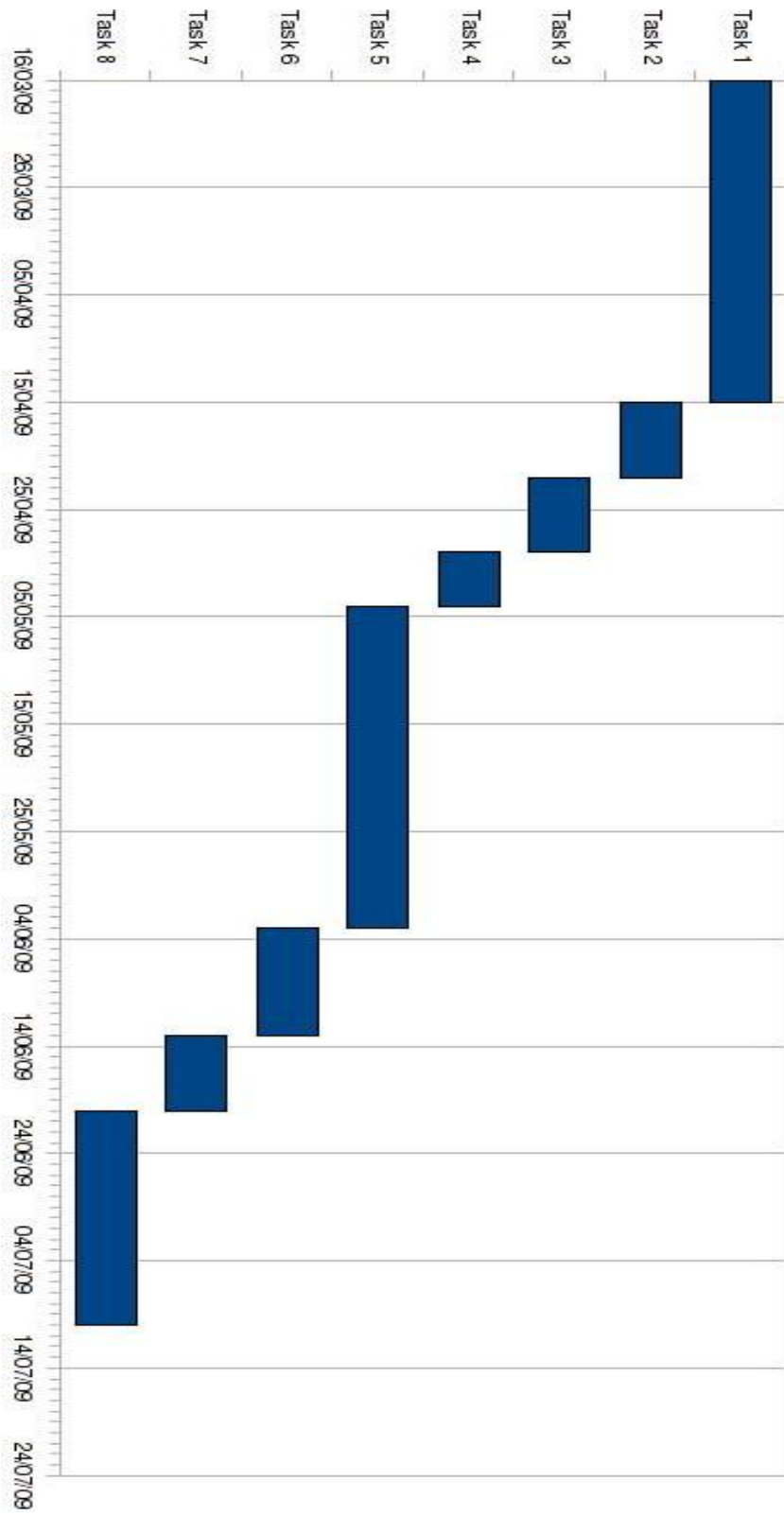


Figure 1: *Time allocation per task (Gantt chart)*

## References

Apple (2008-10-15), 'Max OS X Technology Overview: Graphics and Multimedia Technologies'. *last access: 2009-03-07.*

**URL:** `http://developer.apple.com/documentation/MacOSX/Conceptual/OSX\_Technology\_Overview/GraphicsTechnologies/GraphicsTechnologies.html`

Fedora (2008-05-24), 'Architectures for a Compositing Manager'. *last access: 2009-03-07.*

**URL:** `http://fedoraproject.org/wiki/RenderingProject/CMArchitecture`

Keith Packard, D. J. (2007-07-03), 'Composite Extension'. *last access: 2009-03-07.*

**URL:** `http://cgit.freedesktop.org/xorg/proto/compositeproto/plain/compositeproto.txt`

Keith Packard, E. A. (2007-01-08), 'The DAMAGE Extension'. *last access: 2009-03-07.*

**URL:** `http://cgit.freedesktop.org/xorg/proto/compositeproto/plain/compositeproto.txt`

Keith Packard, J. G. (2000-06-15), 'The (Re)Architecture of the X Window System'. *last access: 2009-03-07.*

**URL:** `http://keithp.com/~keithp/talks/xarch\_ols2004/xarch-ols2004-html/`

Packard, K. (2000-04-19), 'A New Rendering Model for X'. *last access: 2009-03-07.*

**URL:** `http://www.keithp.com/~keithp/talks/usenix2000/render.html`

Packard, K. (2005-07-01), 'The X Rendering Extension'. *last access: 2009-03-07.*

**URL:** `http://cgit.freedesktop.org/xorg/proto/renderproto/plain/renderproto.txt`

Packard, K. (2006-12-14), 'The XFIXES Extension'. *last access: 2009-03-07.*

**URL:** `http://cgit.freedesktop.org/xorg/proto/fixesproto/plain/fixesproto.txt`

Peter Nilsson, D. R. (June 27July 2, 2004), Glitz: Hardware Accelerated Image Compositing using OpenGL, *in* 'Proceedings of the FREENIX Track: 2004 USENIX Annual Technical Conference', USENIX Association, Boston, MA, USA.

**URL:** <http://www.usenix.org/events/usenix04/tech/freenix/nilsson.html>

Siracusa, J. (2005-04-28), 'Mac OS X 10.4 Tiger: Quartz'. *last access: 2009-03-07.*

**URL:** <http://arstechnica.com/apple/reviews/2005/04/macosex-10-4.ars/13>

Wikipedia (2009-01-28), 'Quartz Compositor'. *last access: 2009-03-07.*

**URL:** [http://en.wikipedia.org/wiki/Quartz\\_Compositor](http://en.wikipedia.org/wiki/Quartz_Compositor)

Wikipedia (2009-02-19), 'Desktop Window Manager'. *last access: 2009-03-07.*

**URL:** [http://en.wikipedia.org/wiki/Desktop\\_Window\\_Manager](http://en.wikipedia.org/wiki/Desktop_Window_Manager)

Wikipedia (2009-02-24), 'X Window System protocols and architecture'. *last access: 2009-03-07.*

**URL:** [http://en.wikipedia.org/wiki/X\\_Window\\_System\\_protocols\\_and\\_architecture](http://en.wikipedia.org/wiki/X_Window_System_protocols_and_architecture)

Wikipedia (2009-02-25), 'Compositing window manager'. *last access: 2009-03-07.*

**URL:** [http://en.wikipedia.org/wiki/Compositing\\_window\\_manager](http://en.wikipedia.org/wiki/Compositing_window_manager)

Wikipedia (2009-03-09a), 'Compiz'. *last access: 2009-03-07.*

**URL:** <http://en.wikipedia.org/wiki/Compiz>

Wikipedia (2009-03-09b), 'Exposé'. *last access: 2009-03-07.*

**URL:** [http://en.wikipedia.org/wiki/Exposãl'\\_\\_\(Mac\\_OS\\_X\)](http://en.wikipedia.org/wiki/Exposãl'__(Mac_OS_X))