Keith Packard

keithp@keithp.com

July 29, 2005

Experience

HP (via Compaq) (12/01-present)

Member of the Cambridge Research Laboratory. Research projects focused on user interfaces in all guises, from tiny embedded computing devices running either custom operating systems or uCLinux through multi-machine projector-based media walls.

X window system research included the design and implementation of the X Composite and X Damage extensions which provide for manipulation of application presentation through external applications enabling a wide range of new user environments. This novel new architecture provides capabilities far in excess of those available in other window systems while remaining compatible with existing applications.

Designed and implemented the cairo 2D rendering library supporting X, PostScript, PDF, Windows and Mac OS X output while providing best in class rendering using a PDF 1.4 compatible imaging model. Encouraged adoption of this library by many major free software projects, including Gtk+, Mozilla, Mono and Inkscape.

Developed new font naming and selecting library called fontconfig to unify application use of fonts in free software systems. Promoted library to free software projects by providing relevant patches to their code, including Mozilla, Qt, Gtk+, Tk and X. Fontconfig is now used across all of these applications as the fundemental font configuration mechanism, eliminating a common source of font access difficulties.

SuSE, Inc (11/99-11/01)

Enhanced the XFree86 implementation of the X Window System. Designed and implemented a new rendering extension. Built a new X driver architecture for embedded devices. Wrote and presented numerous papers at technical and industry conferences. Worked with other hardware and software vendors on X related projects, including the Linux port for the Compaq iPAQ.

Developed new text rendering mechanism to improve readability on LCD screens by modulating intensity of individual color elements. Integrated this algorithm into X applications to provide best in class text presentation for window system graphics.

Network Computing Devices, (3/95-10/99)

Designed and implemented X-based products including X terminals, Windows-based X servers and X-based multi-user NT systems. Interacted directly with NCD customers

and resellers to promote X based solutions in general and NCD products in particular. Presented several technical papers and tutorials.

MIT X Consortium (3/88-5/92)

Member of a small group (2-7) of people directed by Robert Scheifler responsible for the development and standardization of the X Window System. Involved in almost all of the related standards efforts, both within the X Consortium and with national standards bodies (ANSI, IEEE). In charge of X server development at MIT for release 3, 4 and 5. Presented many technical papers and tutorials at an international collection of X-related conferences.

Tektronix Inc. (12/83-3/88)

Designed early X terminals. Worked with a team developing an X10R4-based integrated C development environment based on an incremental C compiler

Education

Reed College, Portland, Oregon. BA, Mathematics, 1986.

Publication

All of these are available from `http://keithp.com/~keithp/talks`.

Twin: An Even Smaller Window System For Even Smaller Devices

With embedded systems gaining high resolution displays and powerful CPUs the desire for sophisticated graphical user interfaces can be realized in even the smallest of systems. While the CPU power available for a given power budget has increased dramatically, these tiny systems remain severely memory constrained. This unique environment presents interesting challenges in graphical system design and implementation. To explore this particular space, a new window system, Twin, has been developed. Using ideas from modern window systems in larger environments, Twin offers overlapping translucent windows, anti-aliased graphics and scalable fonts in a total memory budget of 100KB.

Ottawa Linux Symposium, July 2005.

Getting X Off The Hardware

The X window system is generally implemented by directly inserting hardware manipulation code into the X server. Mode selection and 2D acceleration code are often executed in user mode and directly communicate with the hardware. The current architecture provides for separate 2D and 3D acceleration code, with the 2D code executed within the X server and the 3D code directly executed by the application, partially in user space and partially in the kernel. Video mode selection remains within the X

server, creating an artificial dependency for 3D graphics on the correct operation of the window system. This paper lays out an alternative structure for X within the Linux environment where the responsibility for acceleration lies entirely within the existing 3D user/kernel library, the mode selection is delegated to an external library and the X server becomes a simple application layered on top of both of these. Various technical issues related to this architecture along with a discussion of input device handling will be discussed.

Ottawa Linux Symposium, July 2004.

The (Re)Architecture of the X Window System, with Jim Gettys

The X Window System, Version 11, is the standard window system on Linux and UNIX systems. X11, designed in 1987, was "state of the art" at that time. From its inception, X has been a network transparent window system in which X client applications can run on any machine in a network using an X server running on any display. While there have been some significant extensions to X over its history (e.g. OpenGL support), X's design lay fallow over much of the 1990's. With the increasing interest in open source systems, it was no longer sufficient for modern applications and a significant overhaul is now well underway. This paper describes revisions to the architecture of the window system used in a growing fraction of desktops and embedded systems

Ottawa Linux Symposium, July 2004.

Xr: Cross-device Rendering for Vector Graphics, with Carl Worth

Xr provides a vector-based rendering API with output support for the X Window System and local image buffers. PostScript and PDF file output is planned. Xr is designed to produce identical output on all output media while taking advantage of display hardware acceleration through the X Render Extension.

(This is the first paper on the cairo graphics library, which was originally called Xr for X Rendering library).

Ottawa Linux Symposium, July 2003.

X Window System Network Performance, with Jim Gettys

Performance was an important issue in the development of X from the initial protocol design and continues to be important in modern application and extension development. That X is network transparent allows us to analyze the behavior of X from a perspective seldom possible in most systems. We passively monitor network packet flow to measure X application and server performance. The network simulation environment, the data capture tool and data analysis tools will be presented. Data from this analysis are used to show the performance impact of the Render extension, the limitations of the LBX extension and help identify specific application and toolkit performance problems. We believe this analysis technique can be usefully applied to other network protocols.

Usenix Annual Conference, San Antonio Texas, June 2003.

Font Configuration and Customization for Open Source Systems

Font configuration and customization has traditionally been left to each application. Fontconfig is a library designed to provide a common system that can serve to ease application development and provide users with the ability to confidently install new fonts with the expectation that they will be used by most applications. Fontconfig provides the ability for multiple configuration interfaces to affect a wide range of systems without requiring custom code for each new system. Fontconfig provides a range of services to allow applications to pick those appropriate without being forced to use the entire interface. Wide acceptance of the Fontconfig mechanisms will improve system consistency without requiring a radical redesign of existing applications.

Gnome Users and Developers European Conference, Seville, April 2002.

The Xft Font Library: Architecture and Users Guide

The Xft library was written to provide X applications a convenient interface to the FreeType font rasterizer and the Render extension. As FreeType provides for no configuration or customization, Xft also performs this task. Xft provides new font naming conventions, sophisticated font matching and selection mechanisms and sufficient abstractions to permit common applications to benefit from Render extension based text output while still working on X servers without support for this extension.

XFree86 Technical Conference, Oakland, November 2001.

Design and Implementation of the X Rendering Extension

The X Rendering Extension addresses many of the short-comings inherent in the core X rendering architecture without adding significantly to the protocol interpretation or implementation burden within the server. By borrowing fundamental image compositing notions from the Plan 9 window system and providing sophisticated and extensible font rendering, XFree86 is now much more able to support existing applications while encouraging new developments in user interfaces. More work remains to be done in areas where best practice is less well established, including precise polygon rasterization and image transformations.

Usenix Annual Technical Conference, Boston, June 2001.

The X Resize and Rotate Extension - RandR, with Jim Gettys

The Resize and Rotate extension (RandR) is a very small set of client and server extensions designed to allow clients to modify the size, accelerated visuals and rotation of an X screen. RandR also has provisions for informing clients when screens have been resized or rotated and it allows clients to discover which visuals have hardware acceleration available.

Usenix Annual Technical Conference, Boston, June 2001.

Translucent Windows in X

The X Translucent Window Extension is described which solves the general translucency problem by assigning alpha values for pixels in occluding windows. These values

are used to blend the occluding window contents with the occluded region for display. The details of managing translucent window hierarchies, re-parenting translucent windows and X visual differences between blended pixels are discussed.

Atlanta Linux Showcase, October 2000.

Efficiently Scheduling X Clients

The X server is charged with providing window system services to many applications simultaneously, and needs a scheduling mechanism to distribute it's limited resources among these applications. The original scheduling mechanism was simplistic and caused graphics-intensive applications to starve interactive applications.

A new scheduling mechanism has been designed which fairly distributes time among the requesting applications while at the same time increasing performance by a small amount. Descriptions of the original and new scheduling mechanism and empirical measurements demonstrating the effects of scheduling within the X server are included along with a discussion on how the design was arrived at.

Usenix Annual Technical Conference, San Diego, June 2000.

A New Rendering Model for X

The rise of inexpensive Unix desktop systems in the last couple of years has led to the development of new user-interface libraries, which are not well served by the existing X rendering model. A new 2D rendering model is being developed to serve this new community of applications. The problem space and proposed solutions are discussed.

Usenix Annual Technical Conference, San Diego, June 2000.

Font Support in WinCenter Pro: Creating an Application Specific Font Server.

One of the challenges in implementing WinCenter Pro (an X based multi-user NT system) was to efficiently render text. Using strike-format bitmaps or scan converting each glyph into a list of rectangles consumes significant network bandwidth and takes significant time for the X server to render. For reasonable performance the system must use the X text rendering primitives. Substituting an existing X font for each NT font fails to preserve application appearance. WinCenter Pro exports the NT fonts to the X server using the X Font Service Protocol providing both high performance and pixel-perfect results. The overall architecture along with some of the technical issues involved are presented.

Tenth Annual X Technical Conference. The X Resource, O'Reilly & Associates, Issue Seventeen, 1996.

A Pseudo-Root Extension: X Window System Nesting on a Budget.

The notion of encapsulating a screen inside a sub-window has been around a long time. Rob Pike's 'layers' went as far as possible; the only way to create nested windows was to run a new copy of the window system from within a window. Release 6 provided an

X server which could do the same (Xnest). However, both of these suffer from performance problems as each client request must be delivered over two network connections and be interpreted by two window system servers. The X Window System provides for nested windows already, and has only a few references in the protocol to which window is the magic "root" of the window hierarchy on the screen. By changing which window appears to be the root in the context of a particular client, a full-speed window system encapsulation has been achieved. Functionality to encapsulate additional global resources is also included.

Ninth Annual X Technical Conference. The X Resource, O'Reilly & Associates, Issue Thirteen, 1995.

Design and Implementation of LBX: An Experiment Based Standard

Unlike other X Standards efforts, the design of Low Bandwidth X (LBX) could not be done in a vacuum. With the goal of LBX to provide a usable X environment via an extremely limited bandwidth channel, the only way to effectively design the system was to experiment with different ideas and measure which actually worked better. Some of these experiments could be done with pencil and paper, other required extensive development. Many design decisions were made using parts of the eventual implementation. The process, along with the experiments, results and design will be presented together.

Eighth Annual X Technical Conference. The X Resource, O'Reilly & Associates, Issue Nine, 1994.

The Layout Widget: A TeX style Constraint Widget Class

The X Toolkit geometry management process is extremely flexible and powerful; however the existing composite widget classes make it difficult for the application developer both to simply design an application layout, and even more important, to make the layout work in a wide variety of environments.

The Layout widget class is described which uses a stretch/shrink model similar to TeX to constrain the layout of an application in a manner which allows the geometry of the children to respect the desires of the application designer, while adapting to its environment, both in terms of the changing geometry allocated to the widget, and to the changing needs of the child widgets. The specification of the child layout is entirely contained in a resource which is interpreted at run time.

Seventh Annual X Technical Conference. The X Resource, O'Reilly & Associates, Issue Five, 1993.

Using XTrap to Help People with Manual Disabilities

The XTrap extension provides a mechanism to interpose a complicated application between the X input devices (pointer and keyboard) and the X server. Using this mechanism, an interesting system for reducing the amount of manual ability required to operate X clients is investigated and compared with other systems.

Sixth Annual X Technical Conference. The X Resource, O'Reilly & Associates, Issue One, 1992.

X Selection Mechanism

While the existence of the selection mechanism in X may be wide known, the details of using it are not. This paper, while not a research paper per se, attempts to join the technical details with some practical experience. Along with this paper, a sample application was written which provides a working example of the ideas presented here.

Fourth Annual X Technical Conference, Boston, 1990.