

Apple Worldwide Developer Conference 2006 Trip Report

August 7-11, 2006

by Lawrence Peng
Directors Office
L-001
Lawrence Livermore National Lab (LLNL)

DISCLAIMER: These notes are the work of the author. They do NOT represent an official position of any LLNL organization.

Some Major Take-Home Points

Leopard will fully enable 64-bit applications, with the ability to run 32-bit and 64-bit applications side-by-side. Universal applications can be packaged as 32/64 bit for Intel and PowerPC. If coded properly, existing device drivers should need minimal tweaking (to operate in 32 or 64 bit mode). There should be no special requirement of 64 bit drivers versus 32 bit drivers. Kernel processes will still run in 32-bit memory space.

NetInfo is dead and gone in Leopard. This includes the NetInfo tools. NetInfo will be replaced by a transparent local data store which has additional functionality. The Directory Services tools (introduced back in OS X Jaguar) are taking over.

Java 1.4.x is being deprecated, so start moving to Java 1.5 as the baseline.

Full Darwin kernel sources for both Intel and PowerPC now posted as of OS X 10.4.7. The rumor that Apple was closing off OS X source code because of the Intel transition was never true. (<http://www.macosforge.org/>)

Microsoft Mac Business Unit announcements--

--Virtual PC is dead for Intel (other good alternatives already available or pending)

--no Visual Basic support in the next version of Microsoft Office for Mac.

(<http://www.schwieb.com/blog/2006/08/08/saying-goodbye-to-visual-basic/>)

On the basis of the WWDC Developer Release, G3 systems are NOT supported.

(<http://www.apple.com/macosx/leopard/index.html>)

(<http://www.apple.com/server/macosx/leopard/>)

Starting with Leopard, UFS disk formatting is no longer supported. Read and write to existing UFS disks is supported. Case-sensitive HFS-Plus is the suggested replacement.

Apple has stated they will pursue UNIX 03 certification for Leopard.

Keynote Address

Steve Jobs (CEO)

Phil Schiller (SVP, Worldwide Product Marketing)

Bertrand Serlet (SVP, Software engineering)

Scott Forstall, (VP for Platform Experience)

http://events.apple.com.edgesuite.net/aug_2006/event/index.html

The keynote opened with a humorous video (in the style of the recent "Get-A-Mac" ads) in which the PC guy says he is delivering a personal message from Steve Jobs. In this message, developers are asked to stop working on "seamless software that just works" and take the rest of year off. But we could help finishing Windows Vista, and other fun quips.

This years attendance shows more than 4200 registered attendees from 48 countries. The Apple Developer Connection now has over 750K registered developers. This years conference features 140 sessions and 100 hands-on labs. Over 1000 Apple engineers are at the conference this week.

The next topic was an Apple retail update. There are now 157 stores, with 17 million visitors last quarter. Of customers which bought Macs, 50 percent of them are new to the Mac platform. Within the last 12 months, Apple retail store has sold 500 million dollars of 3rd party products. In addition, last quarter was best Mac quarter ever with 1.33 million Macs sold. Growth rate much faster than the rest of industry (about 18 percent versus 6 percent) which means Apple is gaining market share. 75 percent of the Macs sold were Intel-based. Notebook market share has increased from 6 to 12 percent US retail since the intro of first Intel products.

The next topic was regarding hardware. Since MacWorld 2006 in January 2006, Apple has transitioned all of the product lines to the Intel architecture, except for the Power Mac and the Xserve. Today the PowerMac fades into history, and the Xserve will follow suite very shortly.

The new desktop is called the Mac Pro, and it is based on the Intel Xeon chipset using the Intel Core 2 microarchitecture (codenamed Woodcrest). Woodcrest is a dual core processor with a clock speed up to 3 GHz, 4 MB shared L2 cache, a 128 bit vector engine, and is 64-bit using the Intel EM64T extensions. The Mac Pro will have 2 Woodcrest processors, and replaces the PowerMac G5 Quad.

Woodcrest was said to have a performance/watt ratio up to 3 times better than G5.

On synthetic benchmarks, Woodcrest improved on the G5 by

2.1X on SPECint_rate2000

1.6X on SPECfp_rate2000

1.4-1.8X on typical high end applications in use today

1.8X for Xcode builds.

Other attributes of the Mac Pro are:

Dual 1 33. GHz front-side buses, 21 GB/second processor bandwidth

4 Channel 256 bit wide 667 MHz, up to 16 GB, twice as wide as G5

4 snap-in hard drive bays up to 2 TB
dual optical drive bays
more front I/O-another USB, and Firewire 800 joins existing Firewire 400
4 PCI-express full length, a double wide slot for graphics cards
8 FB-DIMM slots

One standard retail configuration (Retail price of 2499 dollars)
Dual 2.66 GHz Woodcrest
1 GB RAM
250 GB drive
Nvidia GeForce 7300GT, 256 MB video RAM
16x Superdrive

Nearly 5 million possible build-to-order configurations via the Apple Store. Build-to-order allows for 2 and 3 GHz Woodcrest processors, and ATI Radeon x1900XT or NVidia Quadro FX 4500 graphics cards.

Similar retail configuration is currently about 1K less than Dell.
Starts shipping today!

New Xserve is also built on same Xeon quad architecture running at 2- 3 GHz
5.4X on SPECint_rate2000
3.7X on SPECfp_rate2000 compared to G5 Xserve

The new Xserve has redundant power supplies, up to 2.25 TB disk (80 gb disk for the base configuration), lights out management, and is entirely build to order. Base pricing will start at \$2999 (vs \$3999 for Xserve G5) and is currently 3-400 dollars cheaper than an equivalent Dell. The new Xserve will be available in October.

The Intel transition is now complete for desktops and laptops. Began on January 10, 2006 and finished on August 7, 2006 for a total of 210 days.

The keynote then moved to discussing Mac OS X. Apple says it has 19 million active Mac OS X users, which means that a large fraction of the installed base is now running OS X. OS X Tiger is Apple's best selling software product ever. Over the last 5 years, Apple has released 5 major updates to OS X. OS X Tiger on Intel is effectively a sixth major release, and it was not trivial to get 86 million lines of source code from PowerPC to Intel with virtually no hiccups. Today, there are more than 3000 Universal applications shipping that run on both PowerPC and Intel. Steve extended a big thank you to the developers.

What has the competition been doing? Back at WWDC 2004, with the introduction of OS X Tiger, banners were hung chiding Microsoft to prepare the photocopyers. Apparently Microsoft took this seriously for the upcoming Microsoft Vista. So a few jokes were made at Vista's expense (e.g. search, RSS, mail and calendar apps, logo, and the fact that you still have the registry, dll hell, activation, etc.).

The central point was that as Microsoft struggles to get Vista out the door, Apple is not standing still. Many of Vista's touted features are still in the future for Windows

customers, but have been with the Mac customers since the beginning the OS X (and already iterated several times). It was noted that Microsoft spends over 5 billion on R&D, but these days seems to just end up copying Apple and Google.

Now we move on to a preview of the next major release of OS X, codenamed Leopard. Steve said there is some top secret stuff that will not be shown, as there is no need to have Microsoft start the photocopiers any sooner than needed. Ten new features of OS X Leopard were highlighted.

1) Top-to-bottom support for 64 bit applications

64-bit support to be extended from the Unix layer (as it is on Tiger) through the application frameworks like Cocoa and Carbon. The frameworks remain 32 bit compatible, so you can run 32 and 64-bit applications side by side. emulated or translated.

2) Time Machine

Apple surveys says only 26 percent people backup "regularly", and only 4% do truly automated software to backups. With Time Machine, Apple aims to simplify the backup process. By default Time Machine automatically backs up everything, which means it can restore everything or restore ala carte. You can backup to a hard drive or a server.

When invoked, Time Machine presents the user with an interface (sending you back to the "Big Bang") which shows the state of the filesystem at various slices of time, heading back to the beginning of time. User can go back in time to find the files in question, preview them, and with one click bring them back to the present.

Time machine works with more than just the Finder. Demos were shown with Address Book and with iPhoto. Click restore and data is pulled back to modern time. It can be made to work with third-party apps.

3) Complete Package

Apple has applications currently out in beta, and others shipping on new Macs only. The goal is to have all these applications ship as part of Leopard. Three example applications are Boot Camp (500K downloads so far), Front Row, and Photo Booth.

4) Spaces

In a nutshell this is an implementation of virtual desktops with an Apple flair.

The demonstration showed 4 virtual desktops. One was for Mail and Safari, the second was for create a podcast, the third was for creating web pages via iWeb, and the fourth was for a Final Cut Pro project.

When you click on an application dock item, you are taken to the correct virtual desktop. Using keystrokes or an Expose-like functionality (F8), you can quickly move to different virtual desktops. In addition, when using the Expose functionality,

you could easily move application windows among various spaces as needed.

5) Spotlight

Spotlight is being enhanced for Leopard. You can search other Macs if you have permissions, and you can search servers. Advanced search options will be available (booleans, filetypes, and other Finder search features). Spotlight will also serve as an applications launcher, and be able to store results of Recent Items.

6) Core Animation

Core Animation is a new framework for Leopard in the spirit of Core Audio, Core Image, and Core Video. Together these frameworks allow developers to easily build media rich applications with high production value.

Core Animation allows you to take any scene and decompose it into layers. Layers can contain virtually anything such as text, images video, opengl, quartz compositions, etc. You specify the start and end state, and intermediate key frames as needed, and Core Animation does the rest. Layer properties include various parameters like opacity, filters, shadows, Geometry, Background, Contents, Sublayers, Border, Composite and Mask.

The demo showed an interactive city being built from album covers. Core Animation was also used to build the user interface for Time Machine.

7) Universal Access

Universal Access is receiving several enhancements. VoiceOver is being upgraded. Braille support is being added. Close captioning is being added to QuickTime. Faster and better ways to navigate through the system is being added.

The demo featured VoiceOver and text-to-speech. Apple's standard boilerplate text for press releases was read using the current Tiger implementation, the current Microsoft Vista beta, and the current Leopard beta. In Tiger (which still carries the standard default voice used in OS 9), the audio is understandable, but rather robotic. Under Vista, the voice overall was better, but tripped on many Apple trademark terms. Under Leopard, the voice was much better than Tiger, and sounded somewhat better than Vista. Even when significantly sped up, the voice under Leopard was still very understandable.

8) Mail

Major enhancements are coming to Mail for Leopard. The keynote focused on three items visible to the user.

The ability to create email messages derived from stationery. Stationery templates are HTML-based, so email should look consistent on any modern email client.

The ability to create Notes using a special mail type. Notes can be created from

email messages, and are stored in a separate mail box.

The ability to create To Do's from other notes, email, or other document. You can set due dates, priorities, etc., for each To Do item. There is a system-wide To Do service available to all applications. An example is that To Do's created in Mail appear in iCal, and vice-versa.

9) Dashboard

Dashboard was introduced with Tiger, and more than 2500 widgets are available today. With Leopard, Apple is enhancing Dashboard in two ways.

First is Dashcode for developers. Dashcode provides templates for RSS, Podcasts, images, etc. It is a graphical tool for HTML and CSS. It has a parts library, and can serve as a Javascript source editor and debugger

Second is Web Clip for users. This allows a user to turn any part of a webpage into a widget. The demo showed portions of dilbert.com, eBay, top 10 Apple downloads, and a webcam being converted into widgets. You bring up the website in Safari, click on a new button in the browser which brings up the webpage in Dashboard. You then select the portion of the webpage you want, click done, and apply a Dashboard theme if you want. It is not clear at this time whether widgets created with Web Clip can be saved so that they do not have to be recreated if the user closes the widget.

10) iChat

Some new capabilities include multiple logins, invisibility, animated buddy icons, video recording and tabbed chats.

But going further there are 3 other major features called Photo Booth effects, iChat theater and backdrops. The demo of these features was a chat between Steve Jobs and Phil Schiller. Photo Booth effects uses the same Core Image/Core Video effects present in the Photo Booth applications. Of more interest is being able to make presentations using photos or presentations using Keynote. Backdrops can be photos or video--imagine you are reporting live from Times Square!

Although not discussed, some additional items for Leopard were mentioned. First is that Parental Controls will be significantly enhanced. Second is that iCal is going multiple users and supporting the CalDAV standard. Third is that Xcode 3 is being announced today.

Leopard is expected to ship in Spring 2007. A Developer Preview is put into the hands of developers today

OTHER NOTES:

Below are highlights from some other sessions I attended. As these sessions were not broadcast to the general public, I am keeping the discussion at a higher level.

Mac OS X State of the Union

Bertrand Serlet (SVP, Software engineering)

Simon Patience (VP, Core OS)

Peter Graffagnino (Senior Director for Graphics and Imaging)

Scott Forstall, (VP for Platform Experience)

As mentioned in the keynote, one of the big thrusts of Leopard is 64-bit top-to-bottom. For Tiger on the PowerPC, 64 bit was at the Unix and command line layers of the operating system. You have a 64 bit address space, with a standard programming model, and could run 32 and 64 bit binaries. For Leopard, the aim is to extend 64-bit up the stack to include full application support with the major application frameworks like Carbon and Cocoa.

64 bit systems provide access to a lot more memory than 32 bit systems. Computational speed can be significantly faster depending on what the application is designed to do (e.g. move stuff in the heap in large chunks or in small chunks). Basically it is a tradeoff. At this time, Apple is suggesting that you strongly consider 64 bit versions of your applications if your app is computational intensive and deals with large data sets, or if you are targeting a 64 bit machine.

You need to be aware, as was detailed in a separate session, that the move to 64 bit may entail adopting some newer API's which have been designed to function well in both 32 and 64 bit environments. Older or deprecated API's may be suboptimal, or are not supported in the 64 bit environment. A few examples of API's which are not available in 64-bit are QuickDraw, Sound Manager, or Code Fragment Manager. These are API's which originated with OS 9, and have been deprecated for some time now.

On Intel Macs, the Core 2 architecture is 64 bit via EM64T extensions. One nice thing about this (like with the AMD64 architecture) is that you have more processor registers to work with, and this does help simplify programming on x86. One of the historical problems with x86 programming versus PowerPC programming (and its RISC-like cousins) is that x86 is a very register starved architecture. While 64-bit x86 is still not as good as PowerPC in this regard, it is a significant improvement.

Multicore processors are now the rage for increasing CPU performance. Previously, improving single core performance relied on increasing clock speed. This has traditionally been done through miniaturization, but the cost is in increased power consumption and heat.

As an example of the ramifications of relying on clock speed, some performance versus power statistics were given. For a single core, a 20% increase in clock speed gave a 13% performance gain with a 73% power consumption. In contrast, for the same clock speed, a second core gave you 73% better performance with only a 2% increase power consumption/core.

As a developer, you can access this performance increase in one of three ways. First, do nothing and let the OS do the work for you. Second, you can use threads, although there is a fine art to this and it can be some work to implement and require

manual profiling. Third is to use some new API's (like NSOperation).

Another big part of writing software is memory management. In C, malloc/free is often used. In Objective-C you have reference counting along with autorelease for object memory management. But neither of these methods provide truly automatic memory management, which is also known as Garbage Collection. Garbage Collection provides for greater potential reliability because programmers do have to watch for subtle memory management bugs.

Starting with Leopard, Objective-C v2 has a new runtime and built-in garbage collection. All objects are subject to collection, and calls to old release methods do nothing. The system should integrate well with malloc and CFRetain. Recognizing that there may be good reasons for developers wanting to stay with the previous system for now, Garbage collection is opt-in on a per application basis. The system frameworks work both ways

As has been noted before, the graphics processors (GPU) continue to expand in power and capability. Apple wants to use the GPU to create cinematic experiences by utilizing the increased compute power, better algorithms, making them easier to program, and using the increased pixels to good advantage.

More computing power means not just gigaflops, but also using the GPU and its memory bandwidth (as appropriate) for traditional CPU-like tasks. One example is using the GPU for floating point calculations. As a reality check, precision and accuracy requirements may be rather different depending on the application.

Better algorithms could include task and data parallelism. An example of Task parallelism is to use multi-threaded OpenGL. On internal tests, World of Warcraft is playing up to 2x faster on the new Xeon machines. The API requires one line of code to opt-in. An example of data parallelism is the Core Image framework.

An example of trying to make the GPU easier to program is Core animation. In general, layered animated user interfaces are hard. Among the things you need to do is learn opengl, manage times and or threads, reconcile model and view discrepancies, manage layout for different sizes, and fix bugs in the above hairy code. This is just for rendering, not application code.

Core Animation is the answer. It is a dynamic layering engine which automatically animates object properties. It is media agnostic. Examples of integration with Cocoa are that NSViews powered by Core Animation. Aqua controls on a layer Core Animation is built into the new interface builder

More pixels are generally good until you get too much of a good thing! One solution is a resolution independent ui so more pixels provide more detail for the same physical size. Developers should avoid QuickDraw and move to Quartz, and be ready for 2008 (it was not specified what was happening in 2008).

It is anticipated that things should just work automatically for text for modern code paths, for icons if proper artwork size is used with 512 x 512 for best results, and for

the user interface unless custom drawing routines are being used. Web content should mostly just work, except that images might take some work.

In other news related to the graphics API's.

The PictureTaker Panel is a public API for Leopard. This API is for taking picture snapshots. It supports a consistent user interface, automatic flash, zoom, crop, and rotate. The camera state handled automatically. The API is already used in login panel, address book, ichat, etc.

Photobooth bundled in leopard, and supports custom effects via quartz composer

QTKit Capture API is a professional grade capture solution. It provides frame accurate AV synchronization. It works with internal and external iSights, USB vdc, firewire iidc and dv camers. Captures can be done to file streams and on screen previews. The API integrates with OpenGL.

The iChat Theater API are public for Leopard. "Theater" contents are controlled by the presenting application. Audio is provided through Core Audio. Video is encoded automatically via H264.

The mechanism used by Time Machine was talked about in a very general way. Separate WWDC sessions covered Time Machine in more detail. In a nutshell, Time Machine starts by backing up the entire file system. A new low priority daemon called FSEvent monitors changes that occur (typically every hour), and coalesces all the changes to a new backup. Unchanged files are typically hard linked to the original. The system is designed to work with journaled HFS-plus. An API is available for 3rd party developers to mark files/folder to not back up. In addition, Time Machine can skip files which are in certain hidden directories and known that they do not need to be backed up (e.g. /tmp).

For servers, Leopard should introduce a new Server Assistant and Server Preferences application. Server Assistant walks you through setting up and configuring essential server services. Server Preferences allows you to manage users and groups and the access to server services. A new dashboard widget to show the status of the server will be available. You can also use Server Preferences to auto configure new clients to use the server services.

As always, both the client and server utilize the same kernel. The kernel is open source, and kernel source is posted for both Intel and PPC.

Server services new in leopard server include calendar, backup, Spotlight, Wiki, Podcast Producer, and Ruby on Rails. Server complements client via standards

Finally a comment was made for why do integration. The idea is to provide the ultimate user experience, have an ideal development platform. It allows you to deploy technologies faster while insulating you from hardware details. Finally it provides a dependable baseline of software and hardware.

Development State of the Union

Ted Goldstein

VP of Development Technologies

It was stated that Apple sees 5 Pillars on which to build development tools

Pillar One--Scalability

Xcode and data

Xcode 2.1 at WWDC2005 for the Universal IDE

Xcode 2.3 with DWARF debug format and dedicated network builds

Xcode 2.4 with 64 bit tool support for Intel

As one benchmark for the improvements in Xcode, numbers were cited for building the Finder. On Xcode 1 on PowerBook took 809 seconds vs 113 seconds on MacBook Pro on Xcode 2.4.

As part of the WWDC Leopard DVD seed, we have Xcode 3. In Xcode 3, developers will have a checkbox to build binaries capable of running as either 32 or 64 bit. All languages supported to be 64 bit compatible as well as the frameworks Cocoa, Core Data, and Carbon.

Pillar Two--Productivity

Productivity means Improved tools that streamline your workflow, and more productive programming languages. One of Apple's secret weapons is Objective-C. With Xcode 3, Apple is introducing Objective-C v2.

A principal feature of Objective-C v2 is the support of automatic garbage collection.

Among the many additions to XCode 3 are improved text editing, codeforwarding (collapsing methods), improved access to documentation, refactoring for Objective-C into the IDE, and provides options to convert Objective-C code to Objective-C v2. Interface Builder in Leopard is being refreshed. Interface Builder can now inspect multiple objects and modify multiple objects.

Xcode 2 and 3 will be able to coexist on the Leopard GM (versus one or the other on the WWDC seed today)

Pillars three and four: Performance and Interoperability

GCC 4 generates 10-15 percent faster code using DWARF and 64 bit Intel.

Xray is a new tool in Xcode 3. It is designed to be a unified performance analysis tool. It has a similar look to Garageband, in that you can drag tools in Xray like you can drag musical instrument loops in Garageband. You can create Mastertracks which can record user interface events. These mastertracks can be saved and used in Xcode like a work flow.

Automator 2 (integrated into Xcode)

Mastertrack-like functionality now built in (Watch me do). Drag events out of window into the script window and auto converted to gui applescript code.

Scriptability

Scripting Bridge

Run applescript against cocoa classes, to generate objective-c code. About 81 times faster than pure applescript.

GCC-LLVM (open source tech in exploration stage)

Compiler framework LLVM and OpenGL

Pipelined in C level source code

Specializes function for a specific vector

Assembles block and optimizes

There were some other features touted which are not in the WWDC Developer seed DVD. One of them is Doxygen (spotlight integration) spotlight importer plugin for leopard GM.

Pillar Five is to make tools to easily develop fun applications.

The example cited was the new Dashcode tools for creating widgets.

Graphics and Media State of the Union

Peter Graffagnino (Senior Director for Graphics and Imaging)

There is a bunch of new stuff in Leopard.

In the audio space, we have new API's like the Audio Queue API, Panner Audio unit, QuickTime AudioContext AU insertion, improvements in media hardware control, and inclusion of the OpenAL 1.1 extensions.

In the 2d graphics space, the Quartz 2D acceleration layer expected to be an opt-in. Print dialogs will be enhanced, and the print engine based on CUPS 1.2. Quartz composer is being enhanced with the ability to create custom patches, to be used as a plugin for the WebKit, to be able to animate desktops, etc.

The Preview application is being upgraded to version 4. It contains an updated user interface, contextual searching and improved image editing.

PDF Kit is being enhanced. New stuff includes a document view, new thumbnail view, new model level classes, viewing and editing, integration with the new interface builder, and the ability to reorder pages. Some of the functions here are reminiscent of the freeware app called Combine PDFs (which is built on an earlier version of the PDF Kit).

Core Animation (code name was "Layer Kit" in the WWDC seed). It provides a high production value experience, easy programming model, smooth transitions between states, and dynamic generation of animation between start and end states. Although can do some 3d stuff, it is a 2d tech at heart. Layers are the

building blocks. Layers can have various properties such as Geometry, Background, Contents, Sublayers, Border, Filters, Shadow, Opacity, Composite, and Mask.

In the 3d graphics space, the goal is a multithreaded OpenGL engine. OpenGL 2.1 is the baseline along with initial support for OpenGL ES 2.0 (embedded systems). The implementation will support 64 bit OpenGL and graphics drivers.

The goal is to be using OpenGL in applications from CAD to the cinematic. Applications could range from shading and materials, physics, particle systems and other natural phenomenon. A technology demo was shown by representatives of ATI Technologies called Toy Shop (<http://www.ati.com/products/RadeonX1K/demos/toyshop.html>).

In the imaging space, we already have Core Image (which was introduced in Tiger). With Leopard, a new framework is being introduced call Image Kit. This framework can be thought of as being analogous to PDF Kit). Image Kit is a high level framework for image manipulation (viewing, editing, browsing). Image Kit is powered by Core Image and Core Animation

QuickTime is undergoing some changes with Leopard. First in the move to 64-bit the QT Kit is path to the future, and developers are strongly encouraged to upgrade their code to use it. There is a new QT Kit Capture API. The QuickTime C API is not supported in 64 bit, but continued support in 32 bit. The low level engine is being revamped to be 64-bit clean, parallizable, and clean MVC separation. The old engine is still available for compatibility. The QT Kit Capture API uses new foundation

The H264 codec in QuickTime is being improved as well. H264 will power the next generation of DVD formats. It currently powers iChat and iPod Video. H264 performance will be improved and include alpha video. Alpha video can be thought of in the same way as alpha channels for still images.

The plumbing for handling alpha video has been in QuickTime for awhile but the files are huge (easily up to 100 x larger than H264 encoded files). H264 with alpha allows standard tradeoff of size and quality.

QuickTime is on web, mostly via the QuickTime plugin. The plugin already has robust javascript and AJAX. The plan is to add transparent mode rendering and synchronous rendering with the browser. In addition, the goal is to make the plugin a first class citizen in CSS (Z order and opacity to be respected), and to add further Javascript API enhancements.

Java Overview

Francois Jouaux

WO/Java Technologies Manager

Mac OS X has the best and most tightly integrated Java implementation. Bundled extensions include Java advanced imaging and Java 3d, Webstart, etc.

Since the release of Tiger, 1800 bugs have been fixed. The emphasis was on compatibility. 35 bugs filed on J2SE 5 (Java 2 Standard Edition v5) since April 2006 on 6.5 million downloads. J2SE 5 start up times on new Mac Pro hardware is 2-3x faster. Apple says they have identified 62 for Tiger and 72 Java apps on Leopard which are must not break.

J2SE 6 (codenamed Mustang) previews available on OS X Tiger. Developer Preview 5 available today.

Java integration with Leopard includes new Aqua support. There has been extended Aqua support for AWT for awhile. Now being extended to Swing.

Graphics improvements continue to be made. Quartz rendering has been present for awhile, and support has been added for Sun's 2D renderer. Support has also been added for Java 3D and Java OpenGL (JOGL).

Support for resolution independence will not be in J2SE until version 7 (according to Sun's current roadmap)

With regard to the Java VM and 64-bit machines, 64-bit Java should just work. Running server applications with J2SE 6 is only for Intel Macs. 64-bits blows away the 4 GB heap barrier. The full Java stack is available, including graphics. This should be a benefit for consumers of huge data sets such as the iTunes music store, gene sequencing, and high performance computing. The performance gain or cost will depend on what the application is designed to do.

Going forward, server side functionality will be for Intel only. J2SE 6 previews will be for 32 and 64 bit Intel only. J2SE 5 will be for 32 bit Intel only

On the client side, the baseline will be J2SE 5 only. Java 1.3 is dead Java 1.4 is deprecated and on the way out.

There are several IDEs available from Apple and third parties. Third party tools include NetBeans, IntelliJ, and Eclipse. Eclipse 3.2 on OS X is 3-4 times faster on the new Mac Pro hardware. Of course Apple provides Xcode. In Xcode you can develop Java and C stuff side-by-side, and Xcode 2.4 provides new ANT templates. You have your choice, and teams can use multiple IDEs and build in harmony. You should remember that the Cocoa-to-Java bridge is now deprecated, so from now on you should use JNI to access native code.

What's New in the Kernel

John Wright

Manager, Kernel Technologies

This session was targeted at developers wanting to understand the underlying mechanism of exported interfaces, and wanting to understand changes to existing exported interfaces. This session was not about developing device drivers.

The target customers of the xnu kernel are mostly higher level developer frameworks. Kernel evolution and innovation are driven by both internal and external forces. Internal innovation is driven from "whole widget" optimization and needed optimizations for Apple's development frameworks. Externally, the kernel continues to be influenced by industry trends and standards, partner technology, open source development and developer requests.

The performance philosophy for the xnu kernel is to focus on improving system level performance. Special attention is given to mach messaging and ipc, and to scheduling and interrupt latency. The xnu kernel contains the Mach microkernel, but xnu is a monolithic kernel which is done to minimize transitions across the kernel boundary.

The open source release of the Darwin kernel for Intel is back up starting with OS X 10.4.7. Darwin is very important to Apple, but there were several challenges to resolve before things could be reopened. First there was lots of new code. Second was the need to review and engineer around code which is not open, while preserving the ability to build and run source code. Third is the need to retain transparency of code for developers and for security review. Finally of course is the usual complication of competing priorities (like new products).

The new 64 bit Intel systems inherit Tiger parity. 64-bit is enabled at the Unix layer through selected libs/interfaces via a standard programming model. The 32-bit kernel enables large address for those processes, and is fully compatible with drivers/kexts.

Leopard moves us to full 64 bit application support. Support is provided for all core frameworks including Carbon and Cocoa. This enables applications using those frameworks to get full 64 bit benefits.

Intel 64 bit support is in the kernel although the current memory mapping not guaranteed in the future. IA32 systems look the same below the 4 GB boundary.

EM64T systems run kernel in compatibility mode which means 32 bit processes are compatible with 32 bit drivers. User addressable space is actually 47 bits (Intel architecture limit), which is about 128 terabytes. In addition to larger address space 64 bit tasks get access to twice as many registers in EM64T (16 general purpose registers in 64 bit mode).

The Unix layer employs standard interfaces, and has supported POSIX standard interfaces for quite some time. For Leopard, the kernel will default all behavior to

the UNIX 03 standard, but still support legacy behavior. If target binaries to Tiger or earlier, you get legacy behavior. You should check the man pages for default and legacy behaviors.

A new tool for Leopard for doing systems analysis is DTrace (Dynamic Tracing). This will be a port from OpenSolaris, It allows for profiling, isolating code paths, and exploring behavior across the whole system kernel and user code. It results in near zero overhead in production code. No recompilation required to add dynamic trace points. Static trace points provide more information in contended routines, and static trace points are nops until activated. DTrace is the technology basis for the new XRay tool in Xcode.

In response to the increasing demands of new systems and applications, some infrastructure adjustments are being made. Some of these changes include:

- 1) enhanced scheduler architecture
 - moving from flat processor model to hierarchical model (tasks and queues)
 - automatic load balancing
 - processor idling without dedicated idle thread to reduce context switching
 - maintain interfaces and behavior
- 2) virtual memory enhancements
 - advanced page replacement algorithms
 - changes to the working set algorithms
 - dynamic pager allocating swapfiles on demand
- 3) increasing some default limits
 - maximum of 266 processes per user (from 100)
 - enforcing some limits never enforced before
 - A) RLIMIT_AS: max virtual address space
 - B) RLIMIT_RSS: max resident memory size
 - C) RLIMIT_MEMLOCK: max wired memory size

One reason for enforcing these process and memory limits is avoid crossing the memory barrier and contain rogue applications.

Leopard is adopting the Mandatory Access Control (MAC) framework. The MAC framework was originally developed for TrustedBSD, and ported by the Darwin Security Extension project. The MAC framework is an architecture for authorization points in the kernel. The interfaces are not yet exported and are subject to change, so it is not recommended that developers use them directly in Leopard.

Sandboxing in Leopard is based on MAC framework. There are different sandboxes for each process, with each sandbox defined by a text profile. Sandboxing does not require special privileges to use, and privileges cannot be escalated. The system is designed to be cooperative with applications, and a preview of it is part of the WWDC developer seed.

What's New in Directory Services

David M. O'Rourke

Engineering Manager, Directory Services

There are significant changes occurring in Directory Services for Leopard. First is an architectural overhaul. Second is changes to local configuration database. In addition, there are continual improvements and enhancements to the Active Directory plugin and for supporting OSX Server.

In Tiger, "Directory Services" is a umbrella term for the combination of 3 APIs: lookupd, Directory Services, and memberd. Directory Services provides network directory access. lookupd handles libsystem, results caching, DNS resolution, local database access, and uses Directory Services for network access. memberd handles group resolution and group results caching for the entire operating system. There has always been some redundancy between these 3 APIs.

This is being updated and streamlined for Leopard. There will be a single Directory Services daemon, which means less IPC calls. Directory Services will subsume memberd. As for lookupd, it is being retired, and its functions taken over by the new Directory Services. In the WWDC seed, lookupd is present, but not running. All libsystem API calls are dispatched to Directory Services. In the Leopard GM, both lookupd and memberd will not be running.

With lookupd gone, this means NetInfo is gone. A new local datastore will replace NetInfo. NetInfo has been the historical local configuration store for a very long time and handles remote network access to remote NetInfo databases. NetInfo servers have not been supported since OS X v10.2. NetInfo could hold about 10k entries (string data, not binary data), and had controls for write access but not for read access. In addition, local NetInfo access was handled by a separate rpc based daemon, which meant you had extra IPC even for local data access.

The new local data store (called DSLocal) will be located at `/var/db/dslocal/nodes/default/` and is no longer opaque. NetInfo data structure retained, but put elsewhere. The new data store is a lightweight local implementation in Directory Services. The records are plists, and can store string and binary data. One less daemon running means less IPC. LDAP, AD and NIS are the network directory protocols supported by Leopard.

To repeat, NetInfo is really gone in Leopard. NetInfo manager and all the NetInfo tools will be removed. All local NetInfo data will be migrated during install, and any network NetInfo access will not be supported. This change should be transparent to 99 percent of software.

If you have not already done so, you should migrate from Netinfo tools to the equivalent Directory Services (DS) tools. The DS tools were introduced back in OSX v10.2 Jaguar. Examples of Netinfo toolnames and the corresponding DS toolnames are given below:

--nicl to dscl

--niload to dsimport

- nidump to dsexport
- NetInfo manager to Accounts system preference (to manage accounts)
- NetInfo manager to dseditgroup (to manage groups)
- NetInfo manager to Directory Utility (to manage mounts)
- NetInfo manager to dsenableroot or Directory Utility (to enable root)

A couple of new admin tools were mentioned.

- the Accounts system preference can now edit common advanced user settings
- the Directory Utility application (previously called Directory Access) allows you to edit mount records, and to enable or disable the local root account.

So the summary of the architectural changes for Leopard is as follows:

- directory services, lookupd, and memberd are merging to simplify the system directory resolution infrastructure
- NetInfo no longer supported (in WWDC build, but not in GM)
- removed Bonjour, SLP, SMB, and Appletalk plugins. Their functionality is being picked up by a new network browsing infrastructure adopted by the Finder (the speaker could not reveal any details).

The AD plugin is receiving several enhancements, and continues to be supported in Directory Services. AD is used by about 40 percent of customer sites. Among the enhancements are improved the replica/site support, support packet signing in AD along with SMB file system support for packet signing. This should help make OSX a first class citizen for AD networks. Leopard should now support all security options for joining an OSX system to AD.

Improvements in support of OSX Server

- Upgrading to OpenLDAP 2.3.x, and moving all Apple diffs to overlays to improve open source merging of Apple changes
- support for the Teams services announced as part of Leopard
- schema and access control improvements
- granular access controls allows for choices other than full admin

Improvements are being made to replication scaling. The current design is hub-spoke model which has limited performance scaling. The new design will allow 32 replicas of an Open Directory master and 32 replicas of each replica which allows for a total of 1024 replicas in a two tier hierarchy.

OSX Leopard Server will feature improved Airport base station support. Airport base stations already support RADIUS authentication, and now so does OSX Server using FreeRADIUS (<http://www.freeradius.org/>). In OSX, the FreeRADIUS setup is integrated with Server administration applications. The RADIUS server can handle multiple base stations, which allows users to authenticate to your wireless network with Open Directory name and password. Authentication process supports WPA2 Enterprise Authentication 802.1X.

Managing Clients with Leopard Server

The ability to manage Mac desktops continues to evolve.

Tiger introduced preference manifests and portable home directories.

Leopard introduces improved security (e.g. application launch restrictions), better managemability (hierarchical groups) and more mobility (external accounts).

Application launch restrictions is an approach which was previously introduced in Macintosh Manager. It uses application identities to decide whether an application is blessed to be launched or whether it is blacklisted.

A current weakness in Tiger is that the application bundle ID can be easily changed using TextEdit or even ResEdit, and there is no application launch restriction support for Widgets.

The solution for Leopard is a kernel based mechanism using the kauth API which first shipped in Tiger. On top of it is an enforcement kext which also talks to the application ID and application launch policy. There should be no way to change the application ID or launch policy unless you are working as root. The current Leopard implementation is based on path based policies. Future implementation possibilities might be based on code signing or dynamic permissions granting.

The new Family/Parental controls scheme sits on top of managed desktop, so the managed desktop idea is not restricted to an enterprise/education feature.

Hierarchical groups allow for granular management, and for inherited preferences via group membership. memberd introduced with OSX Tiger enabled ACL support. Computers can now belong to multiple groups, and legacy management via computer lists is supported. If one prefers to operate in the command line, there are command line tools available to show you how nested groups are put together, to debug some managed desktop settings, and to just peek under the hood.

External accounts are the next evolution in managed mobility. Panther had mobile accounts based on cached accounts and credentials plus local home directories. Tiger gave portable home directories based on mobile account plus home sync. Leopard introduces external accounts based on portable home directories plus on disk sync.

External accounts can be managed using Workgroup Manager app in OS X Leopard Server. Administrators can require FileVault to be turned on, and can limit which managed computers allow external accounts. External accounts retain managed settings. External accounts was not yet fully implemented in Leopard WWDC seed.

External accounts loosely equal portable home directories on a portable disk. Plugging the disk into a computer which recognizes external accounts gives you local disk performance with network account. Disk sync can facilitate data backup

and recovery.

Hardware which can be used with external accounts are firewire or USB portable disks. Desktop disks acceptable, but obviously not as physically portable. Flash based devices are not recommended. It is suggested for security that you have FileVault (or something equivalent) enabled on your external disk. Disks which are attached to a non-managed computer can be read or compromised.

Managed desktop features can be using with a Workgroup Manager plugin. You can set application launch restrictions (including Dashboard), mobility, internet sharing, disk management, and support for client setup via server preferences.

The System Imaging Utility is being significantly enhanced. The focus is on ease of use (via Image assistant), ease of re-use using workflow re-execution, and putting power in the hands of administrators by having command line control.

The Automator workflow can expose the internal workings of the system image utility. The System Imaging Utility was intended to be modular, so administrators could construct reusable workflows setup system imaging "erector sets". These workflows can also be initiated via the command line.

Command line system image utility enables you to execute workflow against source volumes repeatedly and with customizations passed in as parameters.

The new System Image Utility is a Leopard only feature, and is still work in progress, so not on WWDC seed. Panther/Tiger images created with their tools, but can be served by Leopard server.

64-Bit In-Depth

Matthew Formica

64-Bit software evangelist

Going 64-bit is considered part of going Universal, so application developers should start thinking about migrating to 64-bit. In Leopard, the full API stack is mostly available to 64 bit binaries. The Leopard kernel will still be 32 bit.

There have been API changes to the Cocoa and Carbon frameworks for 64 bit in Leopard. Individual applications may have other dependencies if they use custom libraries. As a broad rule-of-thumb, any framework deprecated before Leopard is not available to 64 bit applications. However, they may still be present for existing 32 bit compatibility. But if you want to be universal in both the 32 and 64 bit worlds, you should start migrating off any deprecated APIs. Good replacement APIs are generally available for all deprecated APIs.

64 bit on Leopard and Tiger follows the standard LP64 data model used in most all other versions of Unix/Linux. 64 bit means apps can access more data, but you do not want artificial data type constraints. Binary compatibility is needed for 32 bit apps, so some primitive types must remain the same size. No 64 bit changes

should be needed when building 32 bit on Leopard. There is no mixed mode operation--entire process runs as 64 or 32 bit.

The API changes were done for consistency, to remove obvious impedance mismatches, and to clean up the API and to deprecate old APIs. In the end, most APIs from the developer point of view still just work. The changes for Carbon are more extensive than for Cocoa, but in both cases developers should be aware that adjustments have been made to type declarations, enumerations, etc. Some of these adjustments are driven by binary compatibility issues, or for compatibility when the API could be called from both Carbon and Cocoa APIs, so developers should check their code as required. For apps dependent on Carbon, two big potential issues is the required transition from QuickDraw to Quartz, and transition to transition to composited window mode.

API changes for Cocoa

- Objective-C runtime rewritten and all structures are now opaque.
- Objective-C v2 features planned for 64 bit
- NSMovieView and NSMovie are gone. You should use QTKit instead.
- NSQuickDrawView is gone. Use Cocoa drawing system/Quartz instead
- NSMenuView is gone Use nsmenuitem custom view support
- Use Keyed Archiving. Highly discouraged is Unkeyed Archiving

There is a script to help you convert and enable end to end 64 bit for your applications. It will automate/flag much of the conversion hotspots. See the t64transition guide for details (/developer/extras/64bitconversion/cocoa64.tops script).

API changes for Carbon

- Pascal Strings are history for 64-bit. We wish them well and they are gone from the headers. Replacements is with C strings.
- File manager FSSpec going away so use FSRegs are provided
- Memory manager
 - use memmove or bzero instead of things like blockmove, blockzero etc.
- resource manger lives on with some new types and with some formats gone

There is no 64 bit QuickDraw, although QuickDraw is there for existing 32 bit apps. QuickDraw has served us well, but it is time to move on to Quartz and HView. QuickDraw is not thread safe, has a fixed resolution of 72 dots per inch, and is limited to 16 bit integer coordinates. QuickDraw was built around a set of primitives and modes which were pixel based. Quartz 2d is based around paths and is extremely flexible. Quartz does not have an auto redraw machinery so clips must be set manually.

With no 64 bit QuickDraw, this has significant implications for other current APIs.

- Appearance manager is gone--use HITheme
- Procprt based custom mdefs, wdef, and edef gone--use custom HViews
- Font Manager is mostly not available--use Core TextFont API
- most icon services and utilities not available

--Drag Manger QuickDraw related calls no available
--Some event manager routines are obsolete. An example is getmouse().
getmouse returns coordinates in the current graphics port, and there is no current
graphics port. So use HIGetMousePosition instead which is new in Leopard

Window manger for Carbon

--Carbon windows must use compositing mode to draw their content
--Dialog manger--all dialog windows use compositing mode, without any code
changes on your part.

Carbon event manager

--no non-compositing window events available

Other deprecated APIs not available for 64-bit applications include:

--Code Fragment Manager and Device Manager gone
--language analysis manger gone
--desktop manger gone
--sound manager gone, use core audio instead
--list manager is gone, use data browser
--textedit and text services manager is gone, use HITextView, or use an
NSTextView inside an HICocoaView (new for leopard)
--Translation manager gone, use Translation service in HIService framework
--Scrap manger gone, use Pasteboard manager
--Display manager gone, use DirectDisplay API

Other API changes

--Open Transport gone, use CFNetwork or BSD sockets
--Appletalk gone, use TCP or Bonjour
--CFPlugin-base print dialog extensions gone, use Cocoa-based printing dialog
extension API
--to get ppd info, use cups instead of ppplib
--QuickTime C APIs not available to 64 bit apps, use QTKit instead

Java for 64 bit systems for Intel only

Cocoa Java not available to 64 bit.

IOKit

drivers that do DMA to/from buffers physically located above the 2 GB line will need
to change to use a new i/o kit class.

32 vs 64 bit in Leopard and Tiger

OSX will choose to run the 64 bit side of a binary on a 64 bit mac
sometimes 32 bit is the right choice, even for a 32/64 application

32 bit plugins in browsers, pro apps and others

32 bit native libraries for Java, Perl, and other command line "interpreters"

use posix_spawn to run the 32 side, but this interface is not available yuet
until then, use lip -thin

64 bit Leopard apps will run properly on Tiger. OSX picks the right architecture from each Universal binary.

64 bit Intel register names and calling conventions have changed from 32 bit (stack-based named as Exx) to 64 bit (register based named Rxx). Thus all Intel assembly (if applicable) must be updated. OSX calling conventions deliberately made as close to Linux and other Unixes on x86 as possible.

In broad terms, the memory layout of the system

--0 to 4 GB kernel and 32 bit address space

--4 GB to 128 TB 64 bit applications and libraries

--128 TB to some larger memory address X is not addressable

--At memory address X and up is reserved for future kernel usage.

(Memory address X was given as a hexadecimal, but I was not able to write it down before the presentation moved on).

Introducing Teams

Michael Lopp

<http://www.apple.com/server/macosx/leopard>

<http://www.apple.com/server/macosx/leopard/wikiserver.html>

Teams is a new way of enabling people to communicate and work together by sharing contacts, resources, and information more effectively. The above weblinks to Apple's Leopard Server Sneak Peak pages provide some screenshots of the features described. The current implementation in the WWDC seed is primarily for feedback.

Groups (or Teams) receive their own website which is basically an online Wiki Server. The Wiki-powered website has the latest news, upcoming events and provides people access to any type of media or online document. This makes it easy for people within the group to create and edit web page content, hyperlink and crosslink between pages, maintain a history of all past changes, provides web-based access to a shared group calendar, blog communication and podcasting.

Using the Teams Directory application gives you access to shared accounts for people, groups, locations and resources within your organization. You can find information, or define new shared contacts, assign people to groups, logically organize groups based on hierarchy, and manage shared resources. An example (used by Apple itself) is to be able to quickly find people and where they are located, to interact with groups with files and messages, and to know something about the meeting location (does the room have a projector?, where is the room?, show me a map).

Loosely speaking, the Teams block diagram can be thought of as follows. The front end can be composed of a Wiki, Weblog, and Calendar. The plumbing includes the use of tags, RSS, and search. All this sits on top of a Directory Service.

Wikis and Weblogs serve different agendas. The Wiki format encourages accessibility and collaboration, organic creation and organization. Wikis never forget every single change, who did it, why they did it and when. A Weblog tends to be organized by time, involve shorter entries which are pushed and forgotten, and composed of little bits of info that often goes stale.

Tags are to help organize and find information. Tags allow for data to be more naturally structured versus requiring a strict hierarchical structure.

Currently Directory Service access is read-only from Active Directory or OpenLDAP. There is no official bridge to Exchange Server, although there may be a possible third party open source project in progress? Teams should support all modern browsers like Safari and Firefox. Mailing list integration is part of the system.

During the Question and Answer session, feedback was given that (if not already available) tie-ins to single sign on was desired. It was also suggested that support should be available for 3 tier user and groups.

Creating Great Automator Actions: Advanced Topics

Kerry Hazelgren

There is some new stuff for Leopard

- Automator 2
- variables
- recording
- automatic serialization
- performance improvements
- define custom or new data types
- creation of conversion actions
- adding workflow support to your application

Variables allow you to change the behavior of a workflow without modifying the actions. At this point, variables are more global than local. Values can be set by the user or programmatically, and support can be added to existing actions. There are 4 types of variables: text, path, date and AppleScript.

Using the Xcode development tools you can:

- add support for variables to actions
- enable automatic serialization (may not be useful for every action)
- define custom or new data types
- creating conversion actions

As the name implies, a conversion action converts or filters data. It has no user interface, and is executed automatically when called in order to ensure a smooth flow of data. Conversion actions are programmatically the same as a regular action except that it has no main.nib file and its suffix is .caction (rather than .action).

--add workflow support to your application

You can use all the power of Automator in your application. You can control workflow input, customization with variables, and receive the output of an action in your application. Workflows should run as a separate process (will not bring down your app if it crashes, etc). Workflow output can be any kind of Cocoa object. Applescript objects are converted to NSAppleEventDescriptors.

To harness the power of Automator for your application, you need to tie into the Automator framework. The framework supports the MVC model, and contains model (called AMWorkflow), controller (called AMWorkflowController), and view (AMWorkflowView) objects. The view object is not in the WWDC seed, but will be coming soon. It was said that the AMWorkflowView object will be public for the Leopard GM. It is not clear whether that is also the case for AMWorkflow and AMWorkflowController objects.

Inside Darwin Calendar Server (Collaborative Calendaring and Scheduling)

Wilfredo Sanchez Vega

<http://collaboration.macosforge.org/>

The new iCal Server is Apple's calendar sharing solution. It is based on CalDAV and other standards. It work with various clients, including iCal. It is open source, and will also ship on Leopard Server.

Many of us use calendars to manage our time and work and home. However getting calendars to talk to each other is not always easy. Currently we have a data standard called iCalendar (defined in rfc2445). We have a scheduling standard called iTIP (defined in rfc2446). Together iCalendar and iTIP provide a standard calendaring and scheduling spec supported by many products. A standard calendar access protocol has been lacking until now. We need a standard to access that data, and that is called CalDAV (http and WebDAV evolved to CalDAV).

CalDAV leverages existing protocols and formats. The main protocols used are HTTP (cannot author entries), WebDAV (authoring capabilities), iCalendar and iTIP. HTTP (rfc 2616) is used in web browsing and for applications. WebDAV adds a structured and editable data model to http.

CalDAV is the calendar access protocol designed to address real calendaring requirements. Companies using CalDAV include Oracle, Novell, Mozilla, OSAF (Open Source Application Foundation) and IBM. CalDAV has two specifications: CalDAV-access (read, write, and search calendar data on a server in final IETF approval process), and CalDAV-schedule (enables scheduling between user on a server close to IETF last call)

Additional design work on server to server protocols taking place in the Calendaring and Scheduling Consortium, also known as CalConnect (<http://www.calconnect.org/>)

The Apple Calendar server implements a general http/webdav/CalDAV service. It is written in Python, based on the Twisted framework and accessible via Open Directory. The server will scale to handle a moderate to large organization but not yet planning for massive scales like Yahoo or Google.

Twisted is an open source networking framework with sub projects supporting a variety of protocols (http, nntp,imap,pop3, smtp,ssh,irc,ftp,sip,jabber, and more). For example, Twisted Web2 is a sub module within Twisted which implements http. Other projects based on Twisted include Twisted WebDAV and Twisted CalDAV (adds CalDAV access and CalDAV scheduling).

The server is integrated with Open Directory and can populate its repository with users, groups, and resources defined in Open Directory, and user authentication is done via Open Directory. Accounts are managed in Open Directory and the CalDAV server automatically picks up the changes. Kerberos authentication is in progress, and is targeted for the Leopard GM. Archiving is not in the current implementation.

WebObjects and Web 2.0

Daryl Lee

WebObjects Engineering Manager

Right now there is no real industry definition of Web 2.0. So the presentation used a working concept of Web 2.0 being the combination of data sources with new ways to present and organize data. Some cited examples given were Flickr.com, Wikipedia.org, Zimbra.com, and (humorously) Flyakiteosx.com.

Reshaping the web using various techniques like AJAX (asynchronous java and xml), syndication and aggregation, public webservice apis, mashups, tagging, collaboration and microformats

The Core Technologies used are typically nothing really new and include xml/soap, javascript/json (javascript object notation), rss/atom, CSS, and REST.

WebObjects provides several advantages for Web 2.0 sites. It is a proven back end server for right clients. It is standards friendly with a flexible architecture. Finally it has an active community providing additional tools and extensions--the best known is called Project Wonder (<http://wonder.sourceforge.net/>).

What's coming up with WebObjects?

Xcode 2.4

--compiled with JDK 1.5 as the current baseline, and includes 1.4 source for back compatibility. JDK 1.6 supported when available.

- Xcode compatibility fixes
- Security fixes (e.g. OpenBase is now turned off by default)

Deployment changes

- JMX support (Java Management Extensions)
- JavaMonitor deprecation

The build system

- Apache ANT build system for WebObjects in Xcode 3 and Leopard.
- work well with other IDEs that support ANT like Eclipse
- WO Jam build system has been deprecated

The WebObjects open source community has a strong set of tools for developing WebObjects applications, and some of those tools are arguably further along than what Apple now provides. Apple will support and ensure compatibility between runtime and open tools, and will open and make public all of the standards and formats that WebObjects requires.

A few of these open source tools are listed below:

- WOProject (Apache ANT build system for WebObjects)
- RuleModeler (replicates WebObjects Direct to Web rule models plus more)
- Entity Modeler (similar to WebObjects EOModeler)
- Integrated development environments such as Eclipse
- WOLips (Eclipse plugin to develop WebObjects applications)

Further Cocoa Java project creation is being retired starting with Xcode 3. Xcode 2.x still supports building existing projects (existing bridging between Java and Cocoa still works and the runtime is still there). You should use JNI (Java Native Interface) to access native code from Java, or use Cocoa for your GUI apps. Do not create any new Cocoa to Java bridge classes. Do not write any new projects using deprecated client frameworks.

With the deprecation of the Cocoa Java bridge, the existing WebObjects Cocoa Java tools are also being deprecated (<http://lists.apple.com/archives/webobjects-dev/2006/Aug/msg01144.html>).

The deprecated tools and their open source replacements are given below:

- EOModeler/plugin to Entity Modeler
- WebObject Builder to WOLips
- RuleEditor to RuleModeler
- WebServices Assistant to RuleModeler
- WOALauncher (I missed what was given as a suggested replacement tool)

Apple is moving forward with the WebObjects runtime, while the open source community leads the tools. This should lead to a more open format and more example code.

Introducing Time Machine

Robert Ulrich

Sr. Software Engineer

Time Machine is Apple's built-in backup solution for Leopard. Existing backup solutions can be intimidating and inconvenient. Backups may not happen automatically, and it can be difficult to find the information later. The design goals for Time Machine were to be simple to configure, integrated with the system, automated, convenient to use, and compatible.

The basic system requirements for Time Machine are that you are using Leopard, have an external hard disk (USB 2 or Firewire), or an AFP server with a high bandwidth connection (via wired ethernet or Airport). The backup format is journaled HFS Plus.

To configure Time machine you choose a disk, and flip the switch. Time Machine is integrated with the Finder. It is integrated with the installer so that you can backup your system as part of the Leopard install process. Backups are always online and accessible. Backups are viewable on Tiger and Panther, are not encoded or compressed, and not just a single data blob.

Time Machine allows you to restore system from backup. By default, all files needed to restore a system disk are backed up. System changes and updates are captured, since most are not large relative to disk.

Everything is backed up once per day. Rolling backups are kept for the last few hours. By default, automatic backups of changes are done on a rolling basis each hour, and at the end of the day the last hourly backup is kept as daily backup. Users may optionally specify only once per day at specific time.

By default, all backups are kept indefinitely. Users may specify a time period instead. If enabled, when disk gets full, older backups are removed. If there is still not enough room then users should consider buying a larger disk, decrease the preservation period or manually delete backups.

Backups are indexed by Spotlight, though currently standard Spotlight queries do not include backups. Special options will be added to search backups (not implemented in the WWDC seed).

Backups are grouped by machine name and date, with separate entries appear for each disk partition. Special ACLs are added to make backups read only.

Restoring backups

- Data can be restored using Time Machine application
- browse with any file management tool
- Installer can restore entire system from a backup
- Migration Assistant can migrate directly from backup

How Time machine works

- the initial backup contains everything
- Use FSEvents to determine what has changed (new to Leopard, filesystem broadcasts changes to you)
- changed files are copied to a new backup directory
- Unchanged files are hard linked to their previous version
- If a folder is changed, then all contents are linked. To reduce the number of potential hardlinks, unchanged folders are linked with archive directory links.

Bundle files are related and need to be in sync. It is possible that files may change while a backup is being made. The new filesystem API called FSEvents can be called to determine if any copied files have changed. If so, then those changed files are recopied. This process repeats until files are stable or timeout.

You can exclude unnecessary files from backups such as temporary files, cache files, index files and large rebuildable files. There are several ways to exclude files from a backup. One is to use an existing location (temp files in /tmp, cache files in /library/caches). Your application may exclude files (marked as do not backup) by a specific path or by specific item. Your application can call CSBackupItemExcluded() to see if an item is excluded.

Within your application, use FSCopyObject to copy files from a backup. This function will handle archive directory links correctly, and restores original permissions. It is the same copy engine as Finder and Time Machine.

There are some definite limits of Time Machine in the WWDC seed:

- No FileVault support implemented right now. You can do FileVault backups to individual disk images with a password. Backing up individual files is still under consideration since you must be logged in to see FileVault files.
- backup is on a per machine basis
- although planned for the GM release, there is no user interface to exclude specific files from a backup in the WWDC seed.
- Archive of backups is not currently on the plate of to do's.
- Backup over SMB will be supported, and NFS should also work. Users can set exclusion through preferences and maybe via Get Info.
- No tie in to Workgroup manager to let administrators decide what gets excluded.
- As of now, secure delete/trash does not hit the backups, but it is being look at.
- How to handle mobile home directories is still under consideration.

Designing for Security

Simon Cooper

Conrad Sauerwald

This session focused writing software which tries to address both data and system security.

Data Security refers to keeping and managing sensitive data such as passwords, keys, and identities. OSX provides several system level facilities such as file vault and encrypted images, and the keychain for passwords, keys, and identities.

You could try rolling your own methods, but that is discouraged. The use of system facilities provides several benefits. First it means you do not have to write, test, and maintain the code for the platform. Second, you have standardized access via API's. Third, passwords stored in the keychain can be shared with other clients. Fourth, you have the ability to scale with the security needs of the user.

If you really want to roll your own system facilities, then you need to read up on it. Most existing systems are published to prove themselves, and designing and implementing secure cryptographic protocols is hard (low level libraries, algorithms, etc.). Good protocols have gone through many versions.

System Security is only as secure as the system as a whole. One bug could destroy it all. There are potentially many places in which undesired access could occur, or unintended bugs could arise. You may not have to be working with root access as different frameworks and libraries may themselves have different levels of privilege to work their way into the system.

A suggestion was given for structuring your application so that only the portion of the application which needs lower level access actually interacts with the lower level of the system. In this scenario, you now need to use secure IPC to communicate with small tool. You need to watch framework links--linked frameworks needs similar privilege as tool using it). Know now that the tool may inhabit a more hostile environment.

Accessing lower levels of the system may require authorization to determine whether you really have the right to be there. Using Apple authorization methods allows you to have authorization policy live outside of the app and lets admins decide, transparently handles UI outside of your app or tool, and allows delegation. The launchd may be useful as well. It allows you to launch helpers per session and globally with specified privileges in a more sanitized environment.

Apple published a Secure Coding Guide (via ADC) in May 2006. It contains references to others and covers additional topics. Highlighted during the sessions was the topics of buffer and integer overflows. These were picked as they present the most risk to customers

Overflows can be benign in that the result is a crash or data loss. This is annoying but considered mostly harmless from a security point of view. But overflows also serve as springboard vectors for worms, viruses, or spyware.

A sign of an overflow is a crash dump contains ASCII characters with patterns like "AAAA" or "0x41414141" showing up in numerous places. Proving that an overflow is not exploitable can be a big effort. Proof requires all code paths checked, but an exploit only needs to find one path. It is often faster to fix than prove it is benign.

Both the G5 and Intel processors used in the Mac have the NX (no execute) bit turned on by default. While NX is useful, it does not solve everything.

Of course best practices for secure coding recommends:

- Use unsigned variables for calculating sizes
- Use fsave functions for all new code.
 - strcat, strcpy, strncat, strncpy, sprintf, vsprintf, gets are bad
 - strlcat, strlcpy, strlcat, strlcpy, snprintf, vsnprintf, fgets is OK

The use of these functions reduces need for complex length tracking, and makes sure the string length is truncated to the true length needed. For Leopard, Apple is doing this, including the KEXTs.

Avoiding integer overflows is another problem. Compilers do not and cannot help you, and public available tools do not solve the problem well. Apple wants to provide some help in Leopard, but they were not quite ready for WWDC. Presumably this will be coming in later seeds.

Summary for overflows

- be very careful about statically sized, stack based buffers
- avoid the dangerous string functions. You should use language features or frameworks if available
- check for arithmetic overflow. Use unsigned arithmetic
- do not trust user input
- ADC to Reference Library to Darwin to Security for secure coding guide

In the WWDC Leopard seed, a technology preview of Sandboxing is included. Sandboxing is useful for trying to contain damage caused by rogue code. Rogue code has access to everything your app does, and can compromise anything it touches. Minimally, what you want is that if rogue code gets in, it can only damage to what the app can directly do, and nothing else.

Design and implementation characteristics at this time:

- based on the MAC framework (mandatory access control framework)
- different sandboxes for each process
- does not require privilege to use, and cannot escalate privileges
- defined by a text profile (3 examples in /usr/share/sandbox)
- designed to be cooperative with apps

Profile language (provisional pending feedback) characteristics at this time:

- profile consists of rules and filters
- compiled for the policy engine
- 3 examples on seed
- provisional API to have app launch itself into a sandbox

What's New in the File System

Deric Horn

Dominic Giampaolo

The session began with a brief Tiger recap highlighting a few notable additions to the file system for Tiger. Of note was the copy engine API call `FSCopyObject` (manage metadata, extended attributes and resource forks), better support for Extended Attributes, and Access Control Lists (ACL).

ACLs were introduced in Tiger, but off by default. Starting with Leopard, ACLs will be on by default. The APIs are available at the BSD level, and are already in use with the dot-Mac services. Full Finder integration is planned for Leopard.

A few new filesystem API's for Leopard were mentioned, although their function is too obscure to go into detail here.

Other items of relevance to Leopard are:

- 64 bit file ID support
- UFS--read and write UFS, but no UFS formatting support of new drives.
- for new drives. use case sensitive HFS instead of UFS
- deprecation of non-thread safe filesystems

The big presentation was the introduction to the `FSEvents` framework (`FSEvents.framework`). `FSEvents.framework` is a CoreFoundation based framework which was designed to address the question:

- did anything change in this file hierarchy?
- is anything changing in this file hierarchy?

The answers to these questions are application specific, and depends on what the application does. You can view file system events on two levels: low and high. Low level is every single thing that happens, and high level are those related to user visible changes. The raw stream of file system events is huge and complex.

There have always been ways in OS X to monitor the file system, and each version of OS X has introduced further options and refinements. One of the latest enhancements which showed up in Tiger was `fsevents` (located at `/dev/fsevents`), which was first implemented for Spotlight. It provides a raw stream of events of all local generated events, and even local changes to network file systems. However for the purposes of `FSEvents.framework`, `fsevents` is still too raw and sensitive to slow clients.

When designing `FSEvents.framework`, the type of clients considered were such as backup, sync, and the Finder. Clients we know we will not satisfy are things like virus checkers. Some of the limits and constraints for the `FSEvents.framework` were that storing a complete log of everything that happens in the file system is not possible, and the need to filter the stream of events to something manageable. `FSEvents.framework` cannot be all things to all people. It is a good solution but not the only one. It is best when you have a large hierarchy with lots of files to monitor, or if you need a full history of changes.

FSEvents.framework lets you:

- watch an entire file hierarchy for changes
 - get directory level notifications of changes
 - persistent change history
 - fine control over the frequency of updates
- You do not get events for changes to specific files.

When called, the FSEvents.framework invokes a corresponding daemon called `fseventsd`. `fseventsd` is based on `/dev/fsevents`. FSEvents.framework then filters the event stream, sends updates to clients and keeps historical records.

Basic FSEvents.framework concepts

- you monitor a path in the file system name space. You only get events beneath that path.
- the path does not have to exist but for now the path must be absolute. You can watch any number of paths
- all events have an eventid. eventid are 64 bit values and never get recycled

Event History

- the FSEvent.framework stores all event for all time
- it is small and compresses well
- the event history is persistent across reboots
- given a previous eventid you can ask for all events that happened since that eventid. So if your app run once a month, that is ok
- when you receive historical events, you may see many events for the same directory

FSEvents.framework assumptions

- clients have to generate their initial state
- clients have code to do a full scan
- clients can do partial scans
- clients can compare the current state to the initial state
- clients can update their state when events happen

What is coming:

- There will be a GUUID (Globally Universally Unique Identifier) for the event stream on a given machine. If you store eventid you need to also store the GUUID for the event stream.
- some kind of special event indicating a volume was modified by a non `fseventsd` aware system (e.g. external firewire/usb drive, network volumes, etc.).
- misc API clean ups

Summary:

- FSEvents.framework provides notification of changes to the file system
- monitoring a file hierarchy produces event for changes anywhere in that hierarchy
- events are coarse grained, directory level changes
- all events have a unique eventid and there is a full historical record indexed by eventID.

Smart Cards and Other Two-Factor Authentication Solutions: An Enterprise View

Shawn Geddis

Security consulting engineer, Apple Enterprise

Some of the challenges and problems to try to solve are:

- controlled access to corporate resources
- id theft has become a global problem
- worldwide directives are mandating solutions

We need a single form of authentication to:

- ensure nothing has changed since last altered by the author
- keeping private what was intended private
- guarantee that a transaction was done by you, and you cannot deny it.

Two factor authentication is the process of verifying the identity of a user, process, or device, on the bases of at least 2 factors:

- A token which is something serving as a visible or tangible representation of something abstract.
- Enterprise directory services

One method of two factor authentication is the use of smart cards. Smart cards have been natively supported in OS X for awhile.

It can be tough to find a typical smart card. Typically identity and signing keys are generated on card, and the encryption key is generated and escrowed on issuing station. Private information is pin protected. However, the specifics on how things are done are not guaranteed to be publicly available.

Smart cards can be used to enable services, to login, to access system preferences, unlock screensavers, enable web access, VPN, email, etc.

Smart Card usage and support on OS X has typically required that the `/etc/authorization` file have a couple of strings modified. There is built-in support for several smart card standards:

- CAC (Common Access Card) is US government
- BELPIC is Belgian
- JPKI is Japan

CCID (Chip Card Interface Device) compliant USB class card reader is best.

A popular set of standards for getting smart cards, card readers and computers from different manufacturers to work together is from the PS/SC Workgroup (www.pcscworkgroup.com).

A good resource for drivers and source code is from M.U.S.C.L.E. (Movement for the Use of Smart Cards in a Linux Environment). A couple of weblinks are:

- www.linuxnet.com/drivers.html

--www.linuxnet.com/sourcedrivers.html

Smart Card binding to an account is abstracted from Directory Services. Recommended way to link a smart card user with a record in a directory is to add the hash of the public key to the users directory service setup. Configurable mapping of attributes allows for selection of email certificate attributes to be combined with other attributes and used as a lookup key into the Directory Service.

Two examples were given of current enterprise customers using two-factor authentication with smart cards and Mac OS X. They were Genentech and the US Army.

In both cases, the goals were typically for single sign on for services, usability for users, ease of integrating with the existing network, and (in the case of the Army) being able to meet government mandates. To get things to work required involving knowledgeable leads from the various services you wished to access, as well as the external vendors.

A couple of stumbling blocks were noted, of which one is still being worked on. First is the Army's need to use 2 smart cards at the same time. Removal of one card tends to lock up screen. This is still being worked on with Apple. Second is that the OS X server VPN service can offer Secure ID authentication, but not from with the Server Admin application. You need to manually configure the VPN server. There is a Knowledge base article regarding this issue. For the Army this was relevant in order to support Secure ID while starting to plan the migration to using Crypto-login from Crypto Cards.

Summary:

- two factor authentication can provide secure and reliable form of user ID, and can solve critical requirements with the enterprise architecture
- extensive smart card support is built into OS X
- system architecture enables new innovative two factor authentication solutions not previously possible.

11th Annual Apple Design Awards 2006

For each category we list the winner first, followed by the runner-up.

Best Mac OS X Automator Workflow

Build Real Estate Catalog/Ultimate
Productivity Action Pack 1.0
Automated Workflows, LLC
www.automatedworkflows.com/automator/ultimate.html

Lecture Recording Workflow 1.2
University of Michigan School of Dentistry
www.dent.umich.edu/itunes

Best Mac OS X Student Product

Lineform 1.1
William Thimbleby
<http://tribarsw.net/inform/>

PhotoPresenter 2.5.5
Simon Bovet
<http://www.arizona-software.ch/applications/photopresenter/en/>

Best Mac OS X Game

The Sims 2 v1.0
Aspyr Media, Inc.
www.aspyr.com/games.php/mac/10880

WingNuts 2: Raina's Revenge 1.0
Freeverse
www.freeverse.com

Best Mac OS X Developer Tool

TextMate 1.5.2
Allan Odgaard
www.macromates.com

F-Script 1.3.3
Philippe Mougín
www.fscript.org

Best Mac OS X Dashboard Widget

iClip Lite 2.0
Inventive, Inc.
www.inventive.us/iCliquite

WeatherBug Local Weather Widget 1.1.0.9
WeatherBug
www.weatherbug.com/labs

Best Mac OS X Graphics

Modo 201
Luxology LLC
www.luxology.com

Unity 1.5
OTEE (Over the Edge Entertainment)
www.unity3d.com

Best Mac OS X User Experience

iSale 3.1
Equinux USA Inc
www.equinux.com/us

Boinx FotoMagico 1.7
Boinx Software
www.fotomagico.com

Best Mac OS X Scientific Computing Solution

EnzymeX 3.1
Alexander Griekspoor and Tom Groothius
www.mekentosj.com/enzymex

FuzzMeasure Pro 2.0.2
Christopher Liscio
www.supermegaultragroovy.com

Summary

As always, the notes above represent only a sampling of the sessions that were available. The conference had both developer and enterprise IT tracks.

As was said in the keynote, there is still "top secret" stuff in Leopard. The conference sessions held to that, so we will just have to see as time goes on what the secret stuff is all about. Presumably the secret stuff is not anything critical the developer community must know in advance in order to make sure applications are compatible with Leopard.