Welcome

To Advance through Presentation
Use Page Up and Page Down Keys

Open Transport in Carbon

Vida Amani

Core OS Networking & Communication Team

Introduction

- Information for Network-centric application developers
- Networking in Carbon
- Porting Network-centric application to Mac OS X

Developer Community

- Application developers
- Protocol developers
- Network Device Driver developers

What You'll Learn

- An overview of Networking in Carbon
  - Today and tomorrow
- The features and benefits it offers developers
- How to Carbonize your Network-centric application

Why Carbon

- Easy transition for existing applications to Mac OS X
- Backward compatibility with current Mac OS 8 for new applications

Carbon on Mac OS X

- Greater stability
- Improved responsiveness
- Dynamic resource allocation

Networking in Carbon

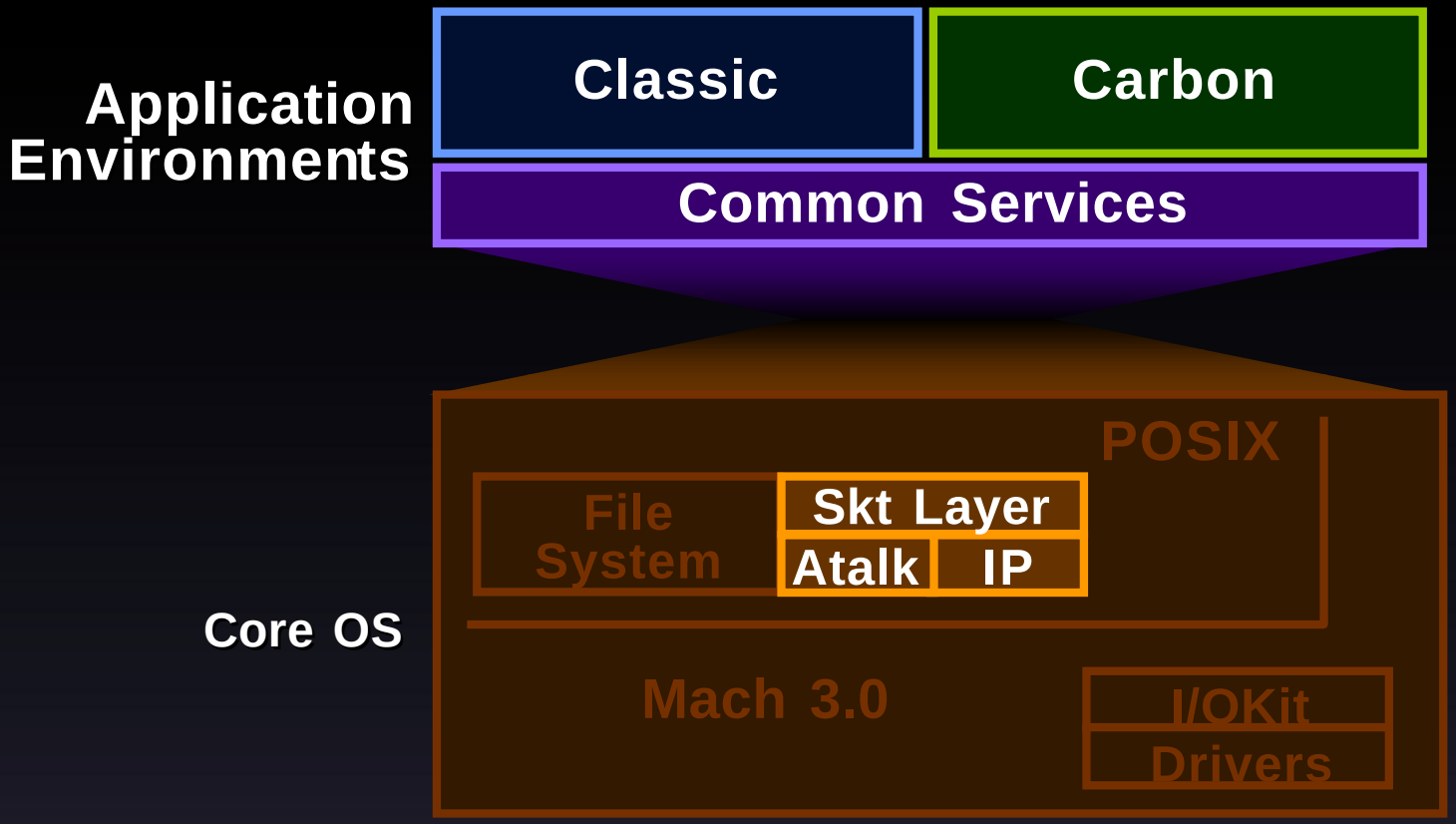
- Goal
- Mac OS 8:
    - Open Transport APIs implemented on STREAMS protocol stacks
- Mac OS X:
    - Open Transport APIs implemented on native BSD protocol stacks

Mac OS X Networking

- Protocols
- Native Network APIs
    - BSD Sockets level APIs for IP
    - Integration of AppleTalk protocol stack with sockets level API to come

Core OS Components

Mac OS X


Application Environments
Classic
Carbon
Common Services
Core OS
POSIX
File System
Skt Layer
Atalk
IP
Mach 3.0
I/OKit
Drivers

OT APIs in Carbon

- A subset of APIs will be supported
- A subset of Protocols/Endpoints currently supported
- A subset of option management and ioctls are supported

OT APIs in Carbon

- Compilation will fail if you use non-supported APIs
- A few APIs have changed
- A few APIs are added

New OT APIs in Carbon

- **OTClientContextPtr**
  - In shared libraries
  - In applications

New OT APIs in Carbon

- Create the OT client context by calling **OTAllocClientContext**
- Pass the client context to **InitOpenTransport**
- After **CloseOpenTransport** call **OTFreeClientContext**

Modified APIs

- Some APIs have changed to take the client context

InitOpenTransport
OTAlloc
OTAllocMem
OTAsyncOpenEndpoint → **OTOpenEndpoint**
OTAsyncOpenInternetServices
OTOpenInternetServices
OTAsyncOpenMapper → **OTOpenMapper**
OTCreateTimerTask
OTCreateDefferedTask
CloseOpenTransport

Client Context Example

- An example of how an Application uses the Client Context

```
#if TARGET_API_MAC_CARBON
#If TARGET_API_MAC_CARBON
OTClinetContextPtr        theOTContextPtr;
err = OTAllocClientContext (0, &theOTContextPtr);
err = InitOpenTransport
                (theOTContextPtr,
    kInitOTForApplicationMask);
#else
err = InitOpenTransport ();
#endif
```

New in OT Carbon

- Universal Proc Pointers (UPPs)

From	OTNotifyProcPtr	to	OTNotifyUPP
From	OTProcessProcPtr	to	OTProcessUPP
From	OTListSearchProcPtr	to	OTListSearchUPP

NewOTNotifyUPP
NewOTProcessUPP
NewOTListSearchUPP
DisposeOTListSearchUPP

DisposeOTNotifyUPP
DisposeOTProcessUPP

Modified APIs

- APIs that are modified to take UPPs

OTNotifyUPP:
- OTAsyncOpenInternetServices
- OTInstallNotifier
- OTAsyncOpenEndpoint
- OTAsyncOpenMapper

OTProcessUPP:
- OTCreateTimerTask
- OTCreateDifferedTask

OTListSearchUPP:
- OTFindLink
- OTFindAndRemoveLink

UPP Example

- An example of how an application uses the UPP type to install a notifier

```
#If TARGET_API_MAC_CARBON
OTNotifyUPP     MyNotifyUPP;
MyNotifyUPP = NewOTNotifyUPP ( MYNotifier);
ep = OTOpenEndpoint (OTCreateConfiguration (“tcp”),
                                0, NULL, &err, NULL);
err = OTInstallNotifier (ep, myNotifierUPP, NULL);
#else
ep = OTOpenEndpoint (OTCreateConfiguration (“tcp”),
                                0, NULL, &err);
err = OTInstallNotifier (ep, myNotifier, NULL);
#endif
```

Protocols in OT Carbon

- Primary protocols supported
  - Today: TCP, UDP, rawIP
  - Future: IPv6
- Protocols under investigation
  - “enet”, Serial
- No plan to support
  - ADSP, DDP, NBP, PAP, ZIP, ATP, PPP

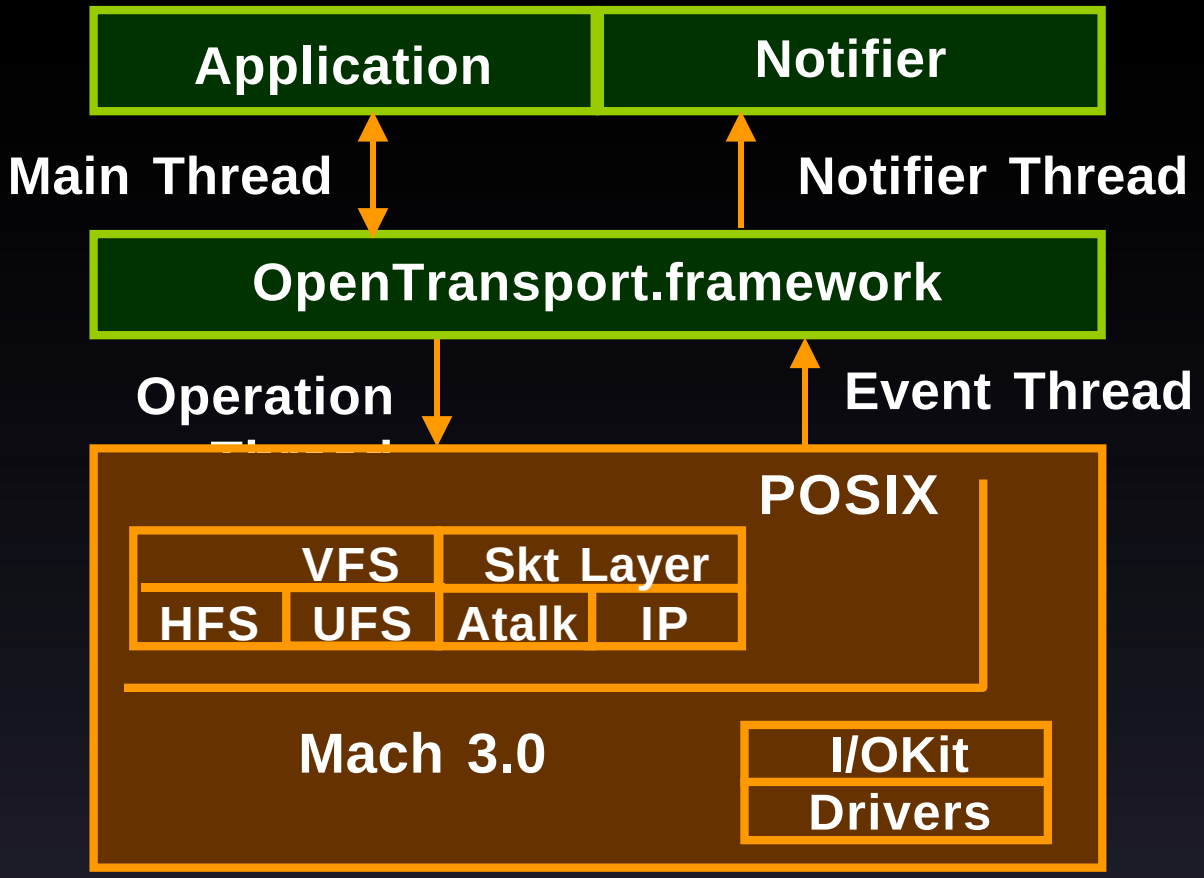
OT Carbon in Mac OS X

- A layer on BSD Socket
- Issues to solve
  - Open Transport and XTI events—non-existent in BSD sockets
  - Asynchronous behavior of OT over synchronous behavior of sockets
  - Differences between XTI APIs and sockets APIs

Architecture

- Defined the OT/ XTI events in the library
- Use of preemptive threads
- Optimization for better performance
    - Event notification mechanism instead of select system call
- Time Manager and Deferred Tasks supported

Architecture



Threading Model

- Thread scheduling in Mac OS X
  - Preemptive
- Thread Manager in Carbon
  - Cooperative
- All threading layered on pthreads

Demo

Vincent Lubet

Core OS Networking &
Communication Team

Building an OT/Carbon Application

- OpenTransport.framework on Mac OS X
- CarbonLib on Mac OS 8
    - Built into the system file
- Build on Mac OS 8 or Classic
- Run the Binaries on Mac OS X and 8

Development Choices

- Developments Tools
  - ProjectBuilder—
    Mach-O based application
  - Metrowerks IDE—
    CFM or Mach-O based application
- Debuggers
  - gdb
  - ProjectBuilder (via gdb)
  - Metrowerks Debugger (via gdb)

Open Transport and IPv6

- Extended APIs will be available for applications that want to be IPv6 savvy
- Applications which are address “agnostic” should just work

IPv6 Guidelines

- Don't attempt to parse address strings
- Use `AF_DNS` address types in `OTConnect`
- Allocate NetBufs for addresses according to `addr` field of `TEndpointInfo` structure
- Use `OTBind(ep,NULL,NULL)` rather than specifying an actual address

Summary

- OpenTransport.framework
  - Ease of transition for Network-centric application
- Carbon is your path to the future of Mac OS
  - Carbon Porting Guide
- Carbon Dater
- developer.apple.com/macosx/carbon
- carbon@apple.com

Related Sessions

609 Mac OS X Networking Overview	Tape
107 Carbon on Mac OS 8	Hall C **Fri., 1:00pm**
910 Mac OS X N&C: Feedback Forum	Hall J2 **Fri., 2:30pm**
106 Carbon on Mac OS X	Hall C **Fri., 2:30pm**

Demo

Q&A

Think different.™

Welcome

To Advance through Presentation
Use Page Up and Page Down Keys