Quality support for open source software

Dan Poirier

poirier@pobox.com

IBM Corporation

Nov. 3, 2010

Outline

I.Background/introduction

II.Provide and maintain reliability

III.Problem diagnosis

IV.Scaling

I. Background/intro

Who am I?

What the talk will be about

What is IHS?

Who Am I?

Programmer at IBM since 1991

Have worked on networking products for SNA, NetBIOS, IPX, and TCP/IP on DOS, OS/2, Windows, Linux, AIX, HP-UX, Solaris, z/OS

Joined IHS team in 2008

Apache committer since 2009

What the talk will be about

Things that have worked well for us in supporting enterprise customers using our software

(as opposed to a survey of industry best practice or something like that)

What is IHS?

IBM HTTP Server

Based on Apache HTTP Server

Part of WebSphere Application Server

Origins of IHS

- First IHS based on Apache shipped Aug 31, 2000
- Based on Apache HTTP Server 1.3.12
- First provided as a front-end for WebSphere Application Server
- Championed within IBM, before IBM got big on open source, by Bill Stoddard

IHS vs. Apache

Wrote our own SSL and LDAP modules before Apache had them. Our SSL is FIPS certified.

Omits some modules we don't need

Built for 10 platforms: 3 Linuxes (ia32, 390, ppc), Win, AIX, 2 HP-UX (PA-RISC, ia64), 2 Solaris (SPARC, x86_64), z/OS

IHS vs. Apache (2)

- Install and uninstall
- Can be administered from WebSphere Application Server
- Some added diagnostic modules (more on those later)

IHS vs. Apache (3)

apxs for building modules

Try to be binary compatible with most common way of building Apache on each platform

How is IHS used?

Front end to WebSphere Application Server:

- Proprietary module passes application requests along and load-balances across backend servers
- Apache serves static files and can do authentication

How is IHS used? (2)

Bundled with other IBM software:

- As front-end (esp. with other WebSphere products)
- Sometimes as simple web server

Typical customers

WebSphere Application Server customer using IHS to serve static files, pass other requests to App Server

Bigger customers may use IHS as one layer in complex topologies including load balancers, caching proxies, authentication servers, WebSphere proxies, and application servers

What we ship

- Server, most Apache modules
- mod_ibm_ldap, mod_ibm_ssl
- Installer, uninstaller
- Sample configuration files
- Diagnostic modules
- Patch file showing changes we've made to the open source parts

II. Provide and maintain reliability

Reliability

Regression testing

Service stream vs. releases

Proactive problem fixing

Common code

Apache contributions

Test systems

Regression testing

Junit-based test suite

Always add a new test if possible when fixing any problem

Tests aren't exhaustive, but after 10 years, they cover quite a bit

Automation makes it feasible to test on all our platforms

Service stream

After shipping an IHS release, we branch the code for service

We keep strict control of changes, trying never to touch that code except to fix a problem one of our customers has hit

Service stream (2)

We want to keep the risk of breaking things in service as low as possible

We want customers to be confident in our service fixes, not reluctant to apply them

Proactive problem fixing

The only time we make a fix in service before a customer reports a problem is for security vulnerabilities

We also follow the Apache HTTP Server dev list and code updates, to be aware of problems our customers might hit in the future, and be ready

Common code

Apache common across platforms (APR)

IHS code common across releases based on the same Apache version

That way we avoid excessive porting of changes across platforms and releases

Common code (2)

Most of our work is building and testing on all of our platforms

Apache contributions

When IHS finds a problem in open-source code, we contribute a fix back.

If possible, we propose an Apache fix, wait for consensus, then fix IHS the same way, to keep our code as common as possible with Apache

Apache contributions (2)

It's important to give back our fixes, because the more we get out of sync with Apache, the harder to take advantage of Apache fixes in IHS.

It also lets us take advantage of the skills of the open source developers and testing of our fixes by the whole Apache community.

Apache contributions (3)

Example: much of the work to make Apache run on Windows NT was done by IBM.

Apache contributions (4)

We do more than fix bugs:

- Participate in development discussions on dev mailing list
- Assist other users on user list and IRC
- Improve documentation
- Add new features

Keeping up with Apache

- We are gradually migrating our users from mod_ibm_ldap to mod_ldap
- We'll start benefitting from Apache developers
- Customers will benefit because all the available Apache doc on LDAP will also be usable for IHS

Test systems

We have to keep build and test systems for all the platforms and release levels we have to support

III. Problem Diagnosis
When problems happen

Problem diagnosis

Most customers can't run under a debugger, insert extra tracing, etc.

So we need them to be able to easily gather information that we can use remotely to figure out what's going wrong

Must gather

For many large classes of problems, we have documents to tell the customer the information they "must gather" for us

E.g. for an SSL problem, we'd have them turn on additional tracing, recreate, and provide the logs, certificates, IP traces, etc.

IHSdiag Tool

- Our IHSdiag tool collects extensive system info: software levels, configuration, logs, network...
- Analyzes core dumps
- Monitors high CPU

Diag modules

mod_mpmstats (Apache): track thread usage

mod_backtrace: on crashes, log stack trace

mod_whatkilledus: on crashes, log request that was being handled

Diag modules (2)

mod_net_trace: trace traffic at TCP level, unencrypted

IV. Scaling

More customers

Bigger customers

Scaling

Tools

Layers of support

Bounding the uses

Service dates

Availability for service

Tools

- Online problem management tool to track problems, share information among everyone helping
- IBM's RETAIN is a 40-yr-old greenscreen app, but it's fast, stable, and does what we need

Tools (2)

Information-gathering tools previously mentioned

Layers of support: L1

- Take customer calls
- Verify eligibility
- Take initial problem statement
- Route to most likely L2 team

Layers of support: L2

- – Talk to customer
- – Own the problem to its finish
- – Answer configuration questions
- – Gather doc
- – Recognize known problems
- – Spot possible defects and route to L3

L2 needs

- Searchable information about known problems and fixes
- Access to developers (ideally local or at least similar time zone)
- Documentation on how to diagnose problems, what customer doc is needed
- Tools to gather the doc

Layers of support: L3

- Assist L2 as needed with advice
- Diagnose more complex problems
- Make fixes to code in service streams and next release (trunk)
- Package changes for general release

Bounding uses

Set limits on:

- What uses are licensed
- What uses are supported

Service dates

Servicing old releases is expensive

Set well-publicized end-of-service date for each release

Optionally offer support beyond that, but charge accordingly

Service dates (2)

Examples for IHS: The version that shipped with App Server 6.0 in 2005, based on Apache 2.0.47, just went out of service except for extended contracts.

Service availability

Big customers need support 24x7 (and will pay for it)

We have L1, L2 on shifts 24x7

L3 is on call for critical problems

RECAP

- Provide reliable, stable code by regression testing and controlling changes
- Take advantage of open source community but give back
- Add value to the open source

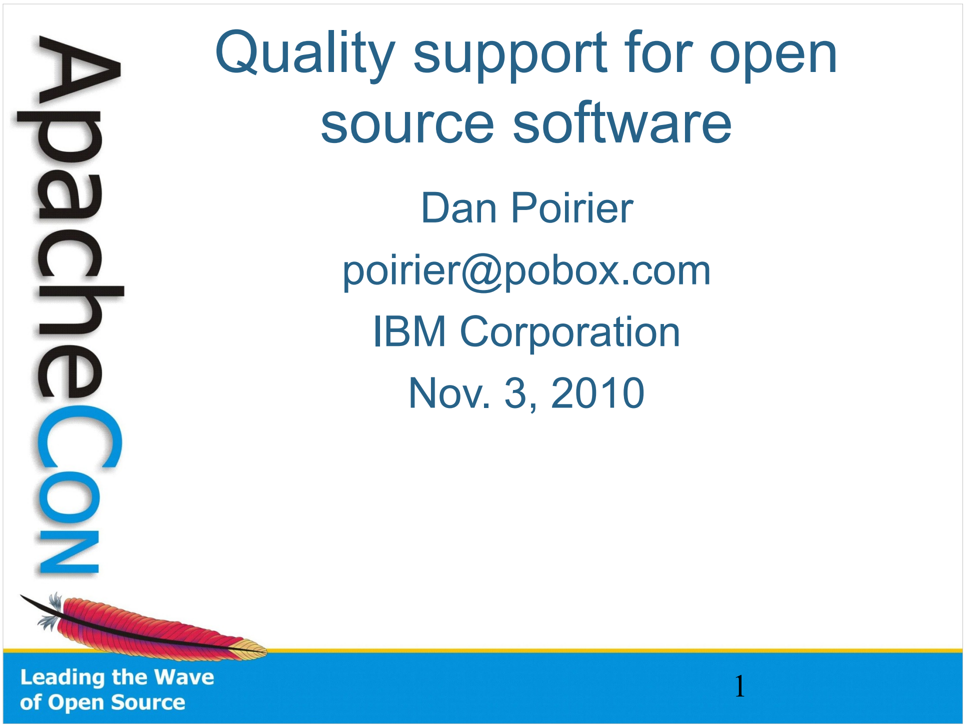
RECAP (2)

Make it easy for customers to gather problem documentation

Scale service by setting bounds on support, breaking down work by specialties

For more information

Links are in the notes (won't fit here)



Quality support for open source software

Dan Poirier

poirier@pobox.com

IBM Corporation

Nov. 3, 2010

ApacheCon

Leading the Wave of Open Source

1

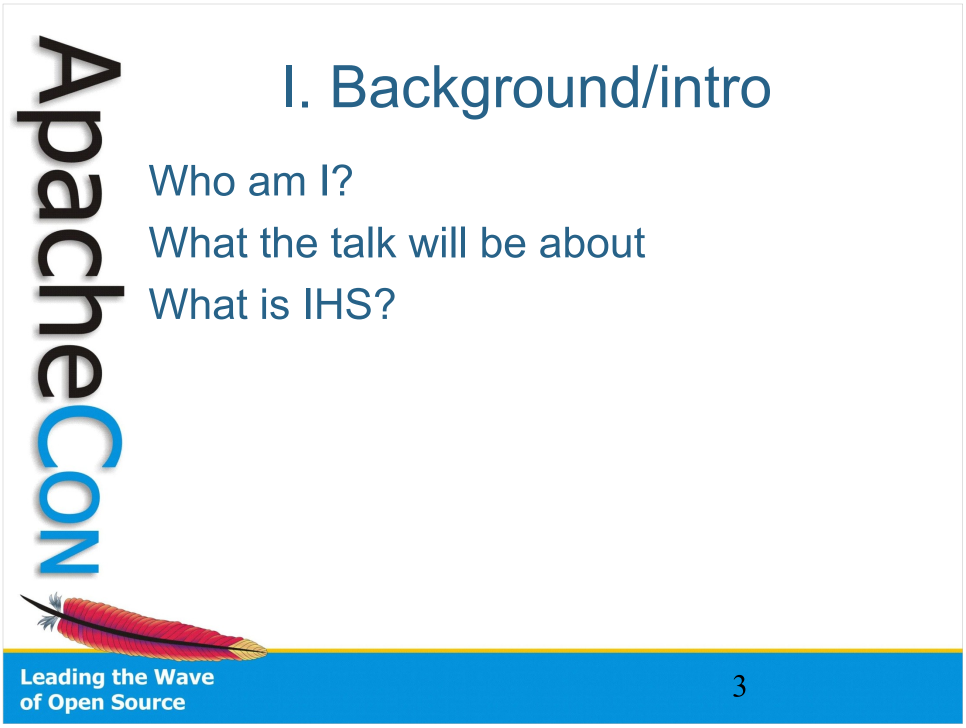
Outline

I. Background/introduction

II. Provide and maintain reliability

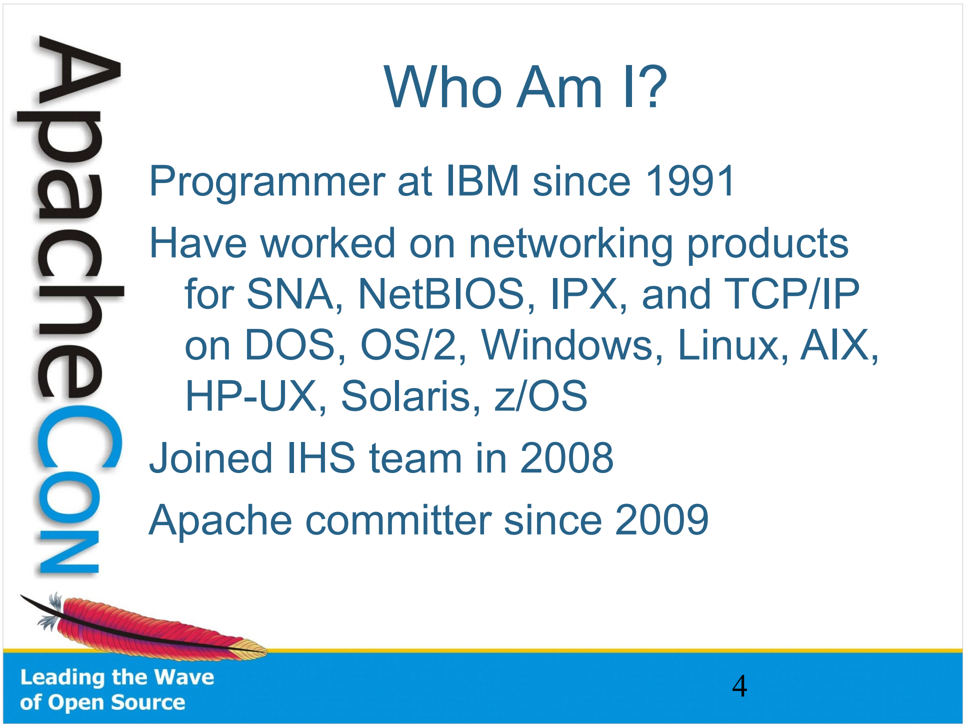
III. Problem diagnosis

IV. Scaling



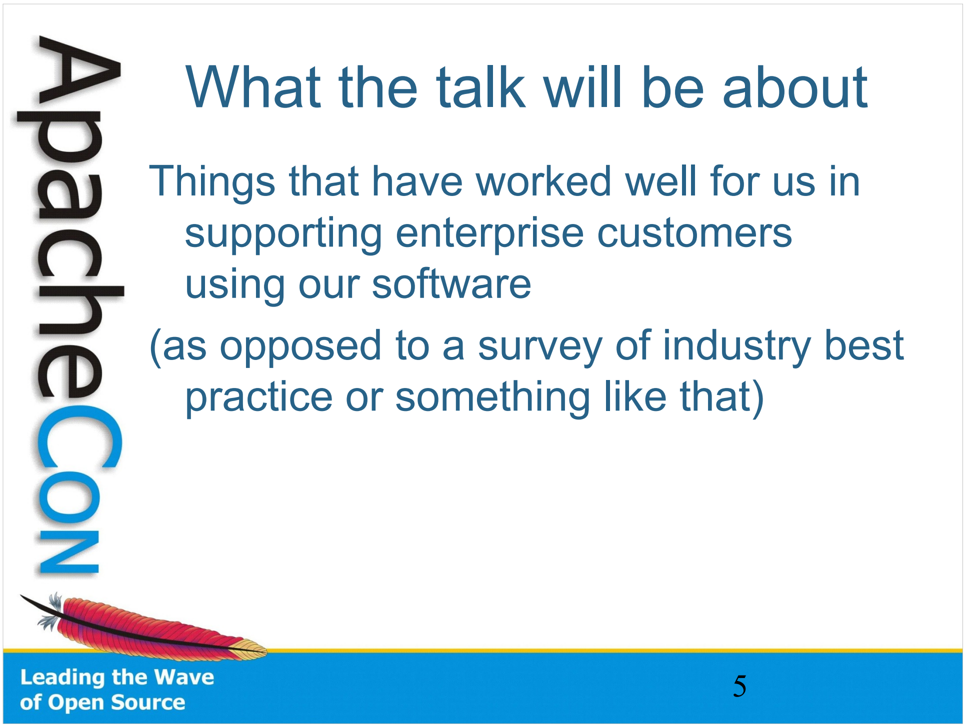
I. Background/intro

- Who am I?
- What the talk will be about
- What is IHS?



Who Am I?

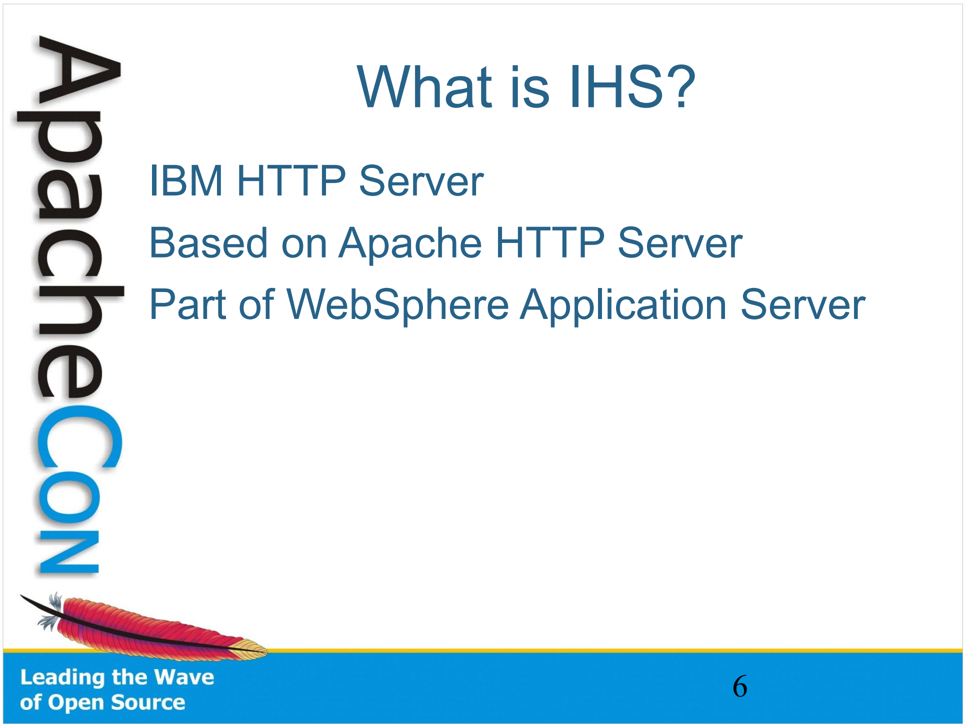
- Programmer at IBM since 1991
- Have worked on networking products for SNA, NetBIOS, IPX, and TCP/IP on DOS, OS/2, Windows, Linux, AIX, HP-UX, Solaris, z/OS
- Joined IHS team in 2008
- Apache committer since 2009



What the talk will be about

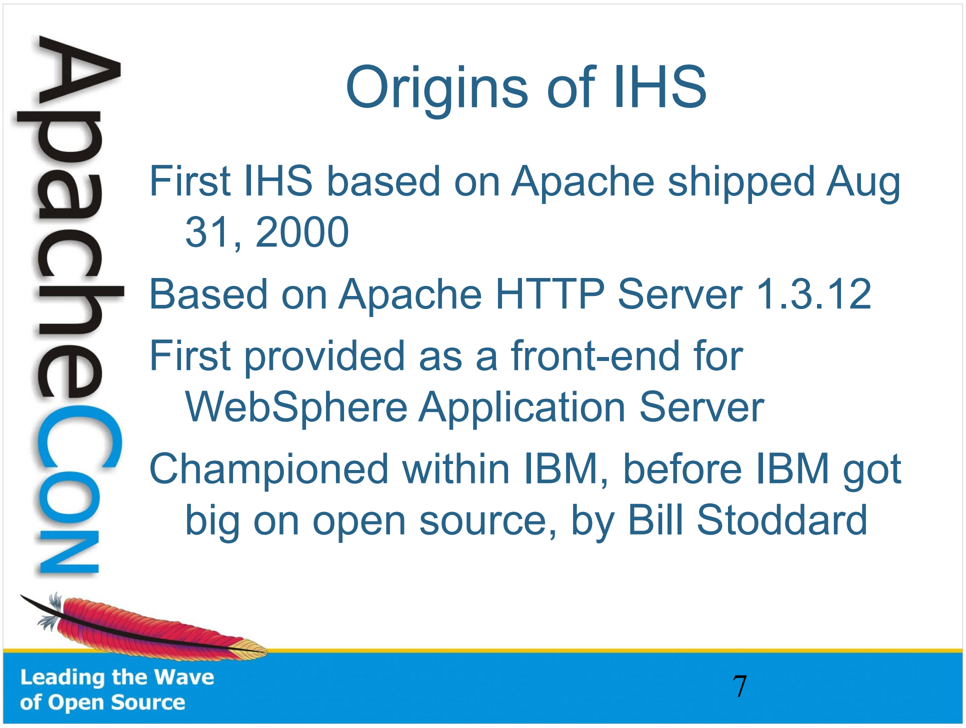
Things that have worked well for us in supporting enterprise customers using our software

(as opposed to a survey of industry best practice or something like that)



What is IHS?

- IBM HTTP Server
- Based on Apache HTTP Server
- Part of WebSphere Application Server



Origins of IHS

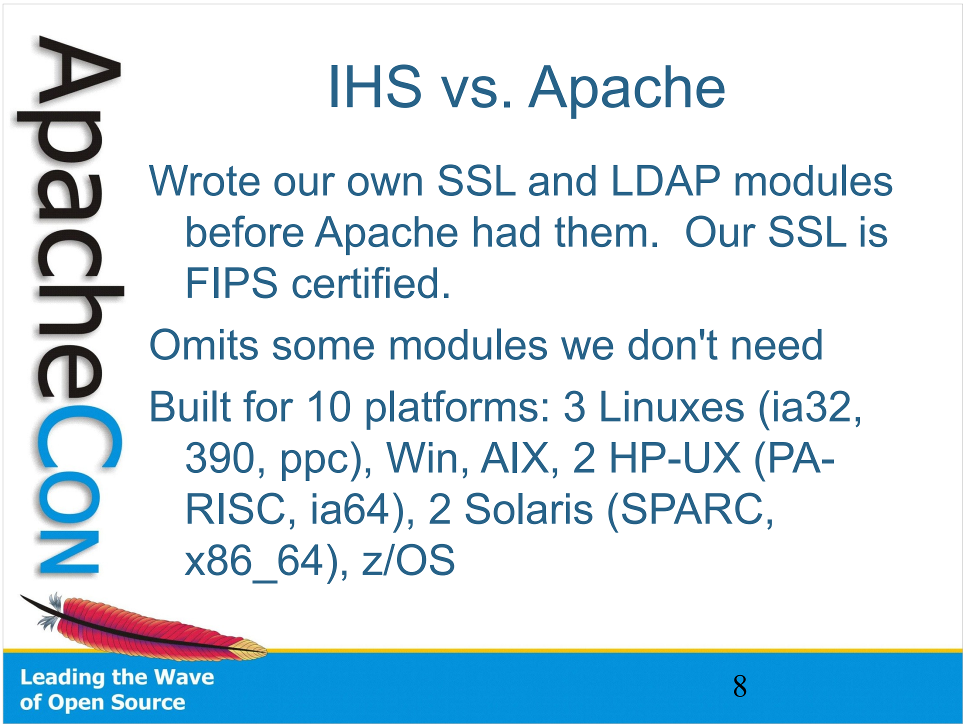
- First IHS based on Apache shipped Aug 31, 2000
- Based on Apache HTTP Server 1.3.12
- First provided as a front-end for WebSphere Application Server
- Championed within IBM, before IBM got big on open source, by Bill Stoddard

ApacheCon
Leading the Wave
of Open Source

There might have been earlier IBM products called "IBM HTTP Server", and there is still another IHS on z/OS only, but they're not built on Apache.

The original intent was probably to be a general-purpose web server product, but the project ended up funded for this narrower usage.

It is now bundled with a variety of other IBM products as well.



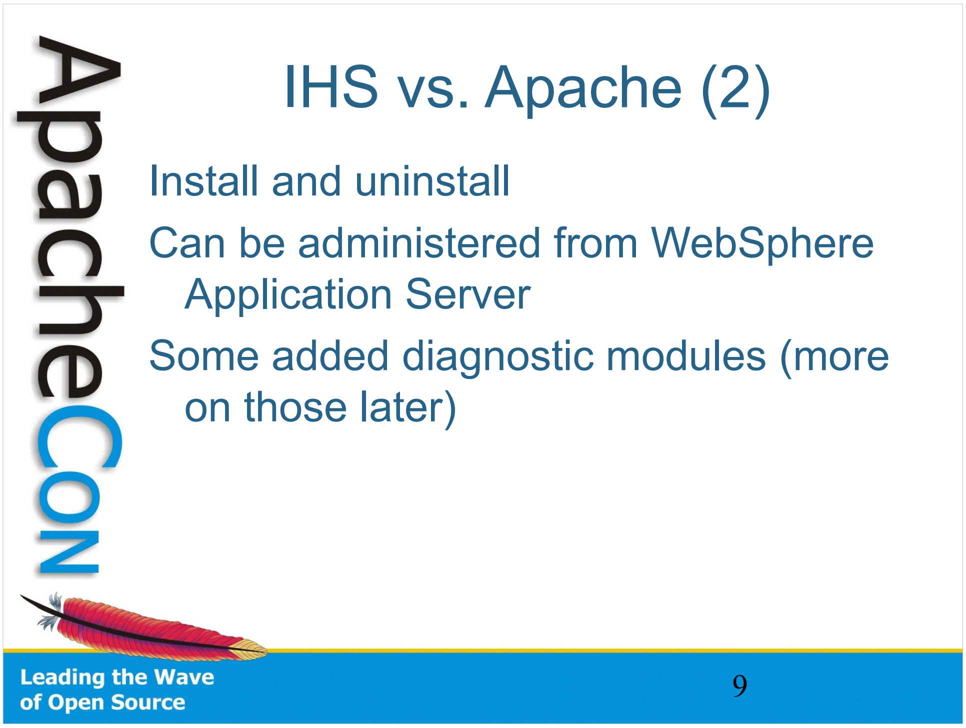
IHS vs. Apache

- Wrote our own SSL and LDAP modules before Apache had them. Our SSL is FIPS certified.
- Omits some modules we don't need
- Built for 10 platforms: 3 Linuxes (ia32, 390, ppc), Win, AIX, 2 HP-UX (PA-RISC, ia64), 2 Solaris (SPARC, x86_64), z/OS

ApacheCon
Leading the Wave
of Open Source

We don't actually omit very many modules. Some modules we omit: mod_dbd, mod_ssl (since we ship mod_ibm_ssl).

FIPS certification, a U.S. Government certification of the encryption we use, is very important to some of our corporate and government customers. Getting FIPS certification is a difficult and expensive process.



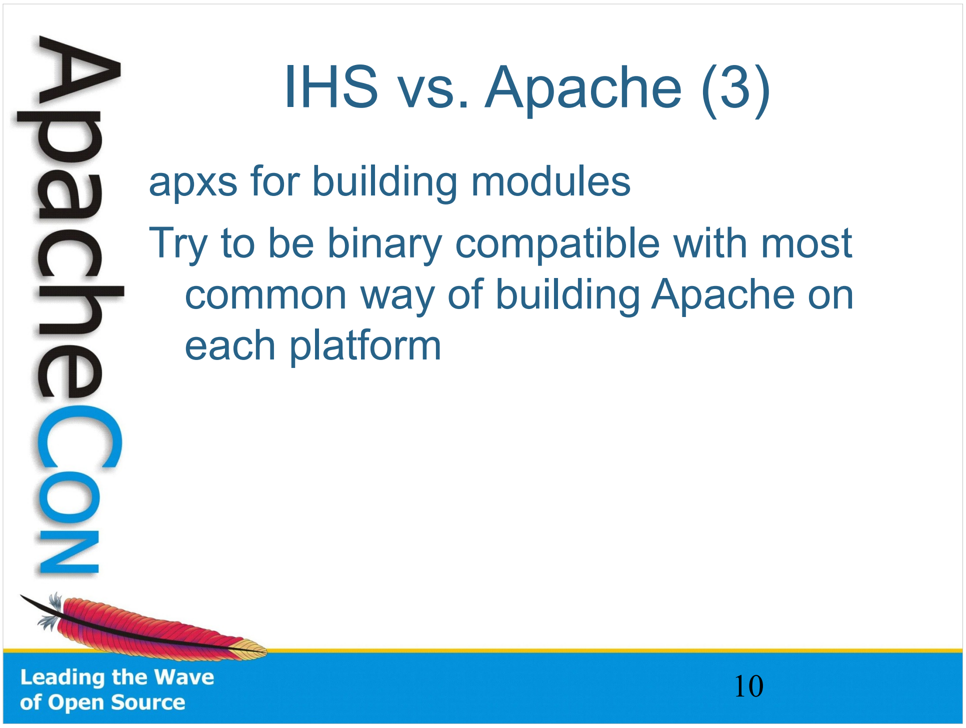
IHS vs. Apache (2)

- Install and uninstall
- Can be administered from WebSphere Application Server
- Some added diagnostic modules (more on those later)

ApacheCon
Leading the Wave of Open Source

We provide graphical install and uninstall programs with a common interface across all of our platforms except z/OS. They all support silent (scripted) install and uninstall.

From App Server, you can start and stop the server, edit the configuration file, and update its configuration for forwarding traffic to WebSphere servers.



IHS vs. Apache (3)

- apxs for building modules
- Try to be binary compatible with most common way of building Apache on each platform

We ship apxs so customers can build their own modules.

In general, modules built for the corresponding Apache version will work with IHS.

This does raise some support issues I'll mention later.

How is IHS used?

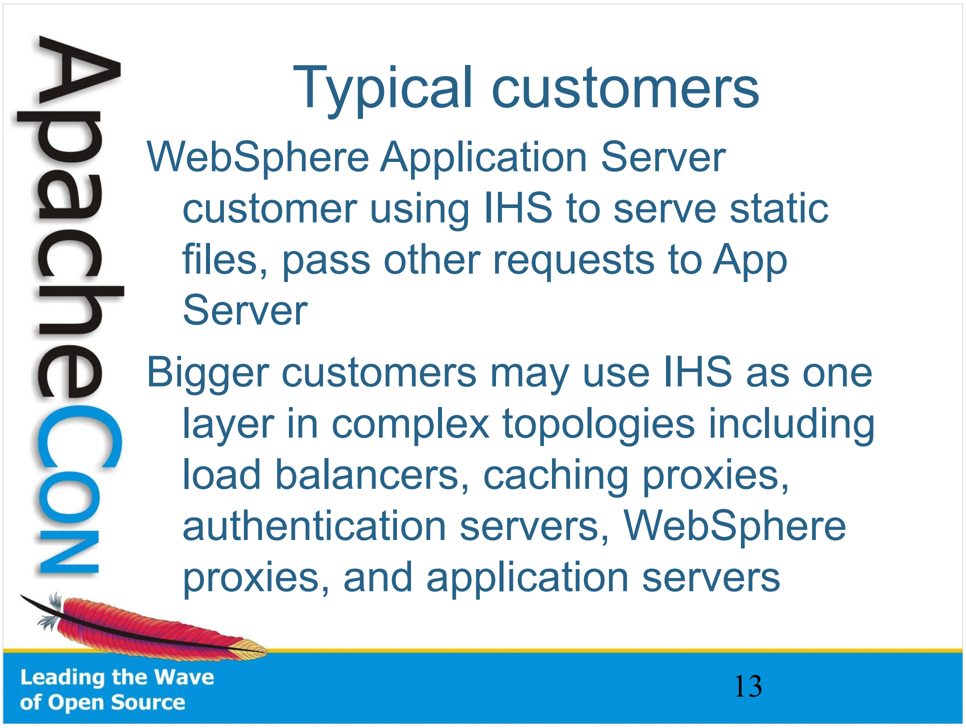
Front end to WebSphere Application Server:

- Proprietary module passes application requests along and load-balances across backend servers
- Apache serves static files and can do authentication

How is IHS used? (2)

Bundled with other IBM software:

- As front-end (esp. with other WebSphere products)
- Sometimes as simple web server



Typical customers

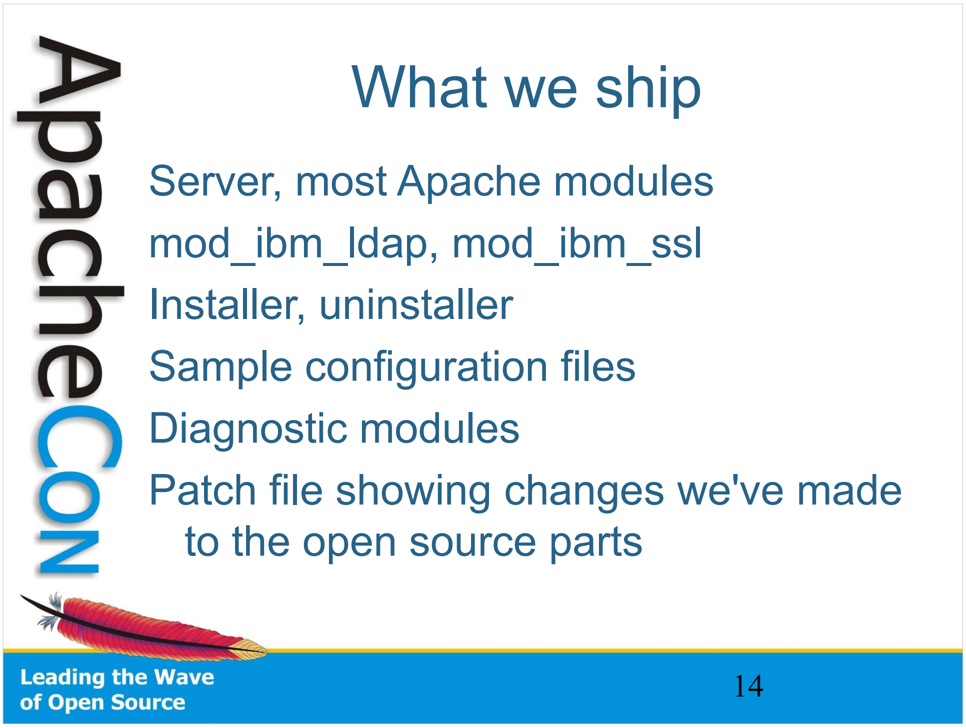
- WebSphere Application Server customer using IHS to serve static files, pass other requests to App Server
- Bigger customers may use IHS as one layer in complex topologies including load balancers, caching proxies, authentication servers, WebSphere proxies, and application servers

ApacheCon
Leading the Wave
of Open Source

Customers come up with all kinds of ways to incorporate IHS.

Many use IHS for authentication, often loading Apache modules not shipped with IHS.

Some use separate instances of IHS, some for static files and others to load-balance application server requests.



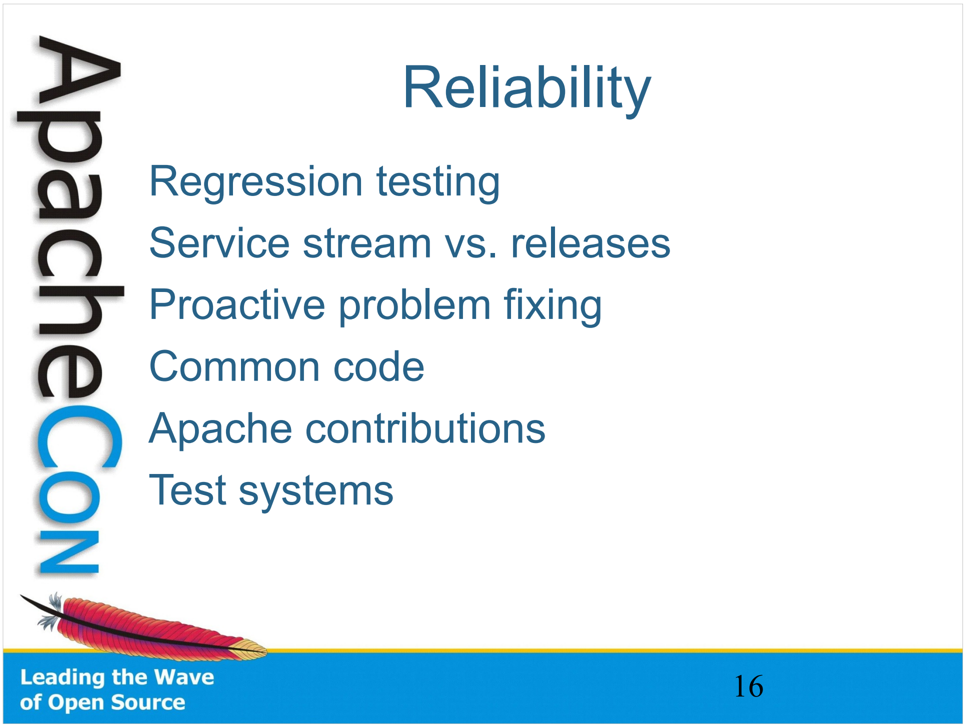
What we ship

- Server, most Apache modules
- mod_ibm_ldap, mod_ibm_ssl
- Installer, uninstaller
- Sample configuration files
- Diagnostic modules
- Patch file showing changes we've made to the open source parts

ApacheCon

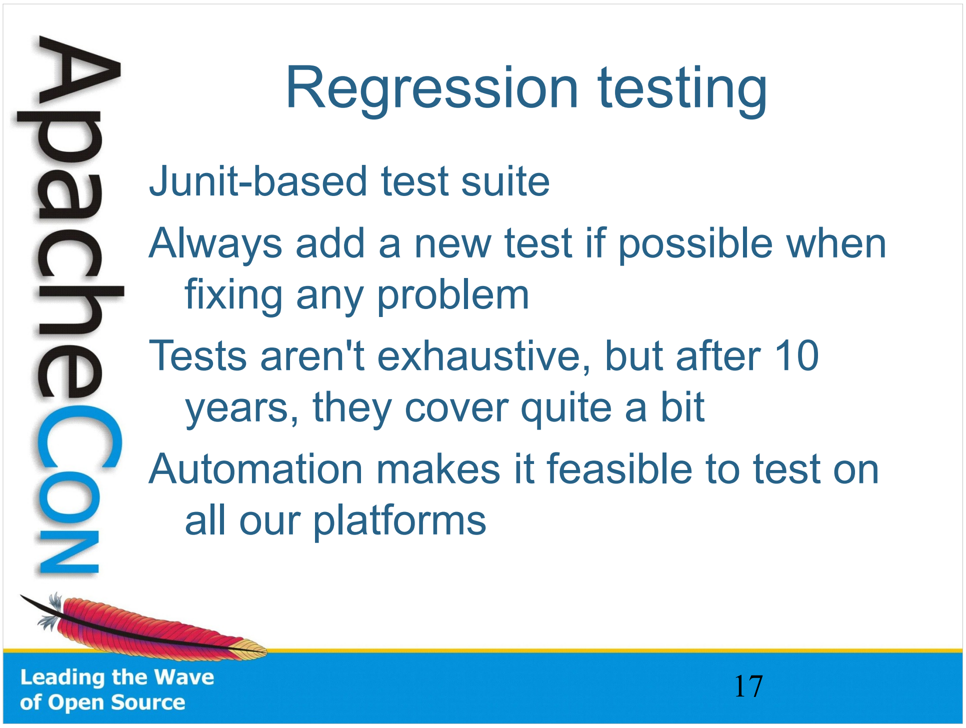
II. Provide and maintain reliability

Leading the Wave
of Open Source



Reliability

- Regression testing
- Service stream vs. releases
- Proactive problem fixing
- Common code
- Apache contributions
- Test systems



Regression testing

- Junit-based test suite
- Always add a new test if possible when fixing any problem
- Tests aren't exhaustive, but after 10 years, they cover quite a bit
- Automation makes it feasible to test on all our platforms

ApacheCon
Leading the Wave
of Open Source

The point of regression testing is to make sure the current fix doesn't break anything we've fixed before, and to make sure later fixes don't break this one.

Having the basic test framework makes it a lot less work to add a new test than starting from scratch.

Just like the tests, we can extend the framework as needed over time, it didn't need to do everything right off the bat.

Service stream

After shipping an IHS release, we branch the code for service

We keep strict control of changes, trying never to touch that code except to fix a problem one of our customers has hit

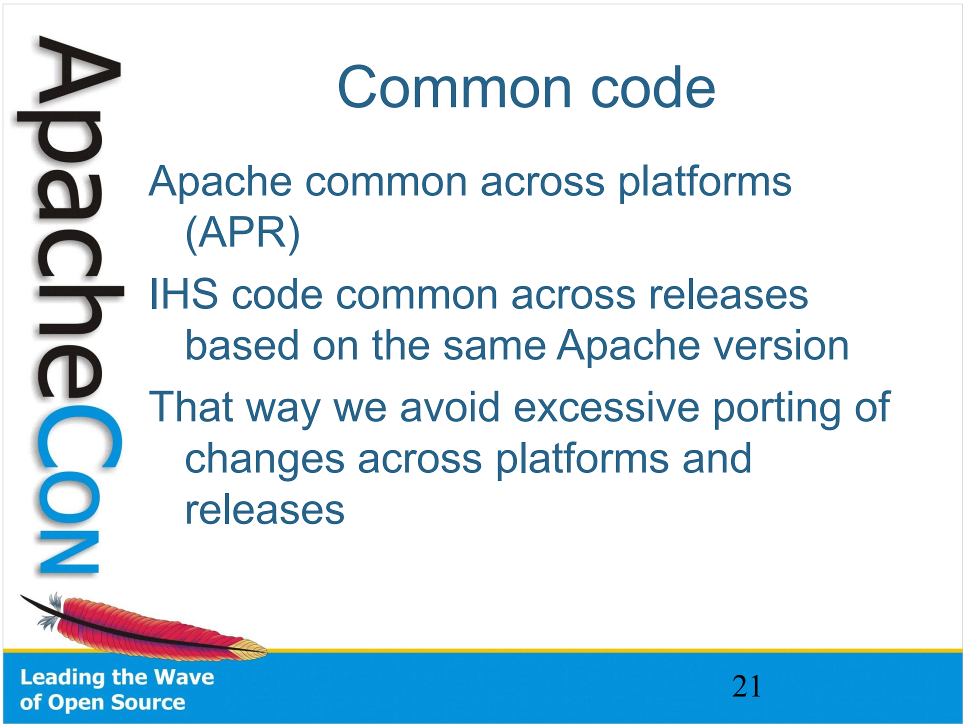
Service stream (2)

- We want to keep the risk of breaking things in service as low as possible
- We want customers to be confident in our service fixes, not reluctant to apply them

Proactive problem fixing

The only time we make a fix in service before a customer reports a problem is for security vulnerabilities

We also follow the Apache HTTP Server dev list and code updates, to be aware of problems our customers might hit in the future, and be ready



Common code

- Apache common across platforms (APR)
- IHS code common across releases based on the same Apache version
- That way we avoid excessive porting of changes across platforms and releases

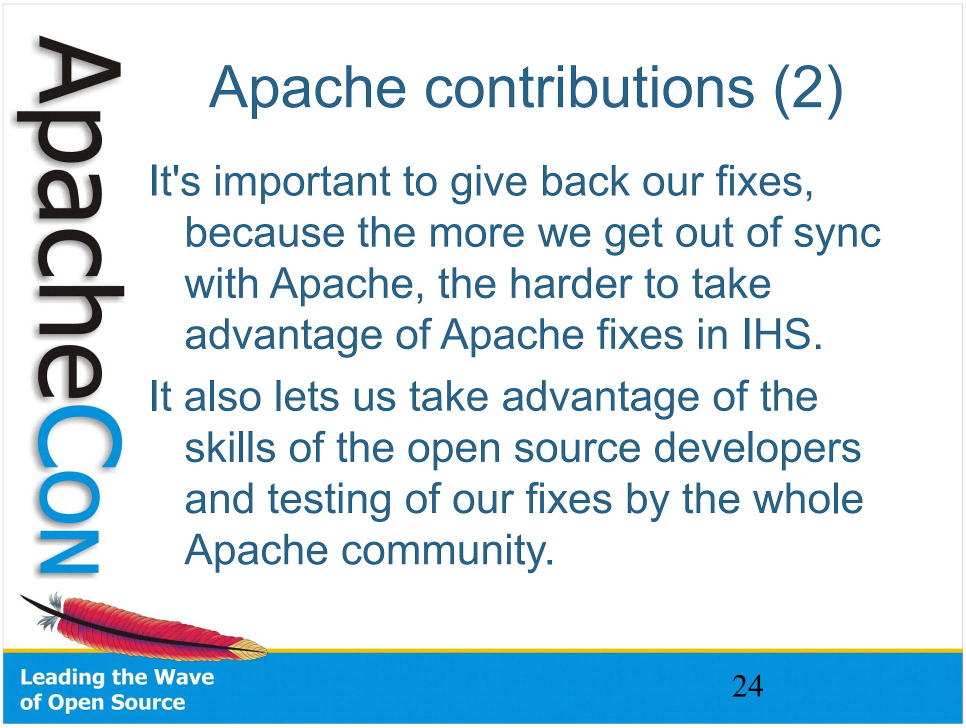
Common code (2)

Most of our work is building and testing on all of our platforms

Apache contributions

When IHS finds a problem in open-source code, we contribute a fix back.

If possible, we propose an Apache fix, wait for consensus, then fix IHS the same way, to keep our code as common as possible with Apache



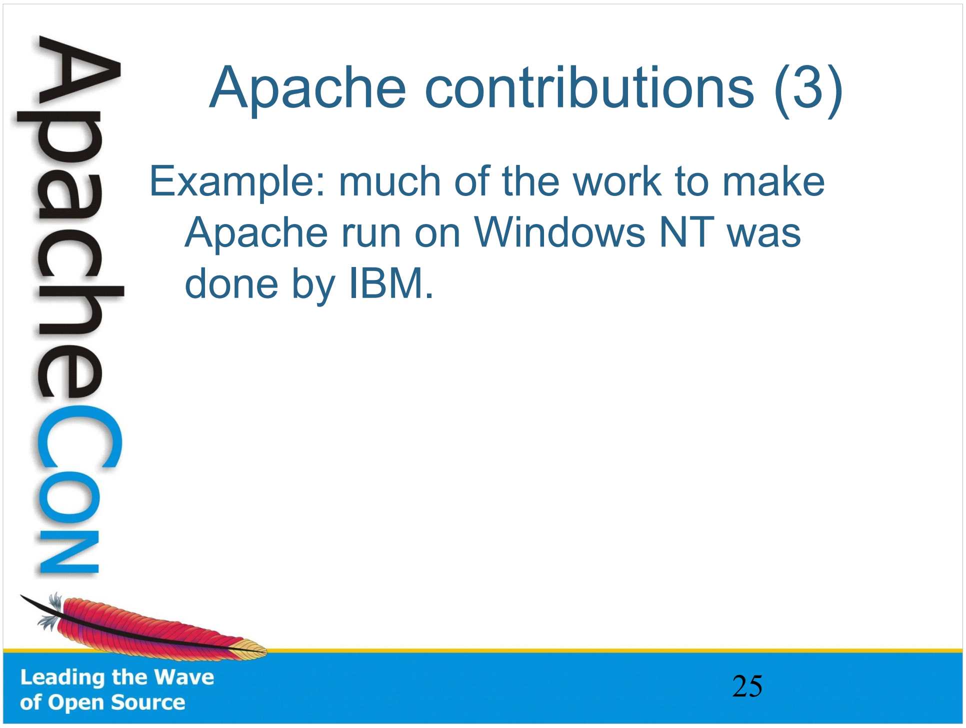
Apache contributions (2)

It's important to give back our fixes, because the more we get out of sync with Apache, the harder to take advantage of Apache fixes in IHS.

It also lets us take advantage of the skills of the open source developers and testing of our fixes by the whole Apache community.

ApacheCon
Leading the Wave of Open Source

Getting the help of the Apache HTTP Server developers, and testing by all the users, is invaluable. It would be hard to attain that level of checking on our code if we kept it proprietary.



Apache contributions (3)

Example: much of the work to make Apache run on Windows NT was done by IBM.

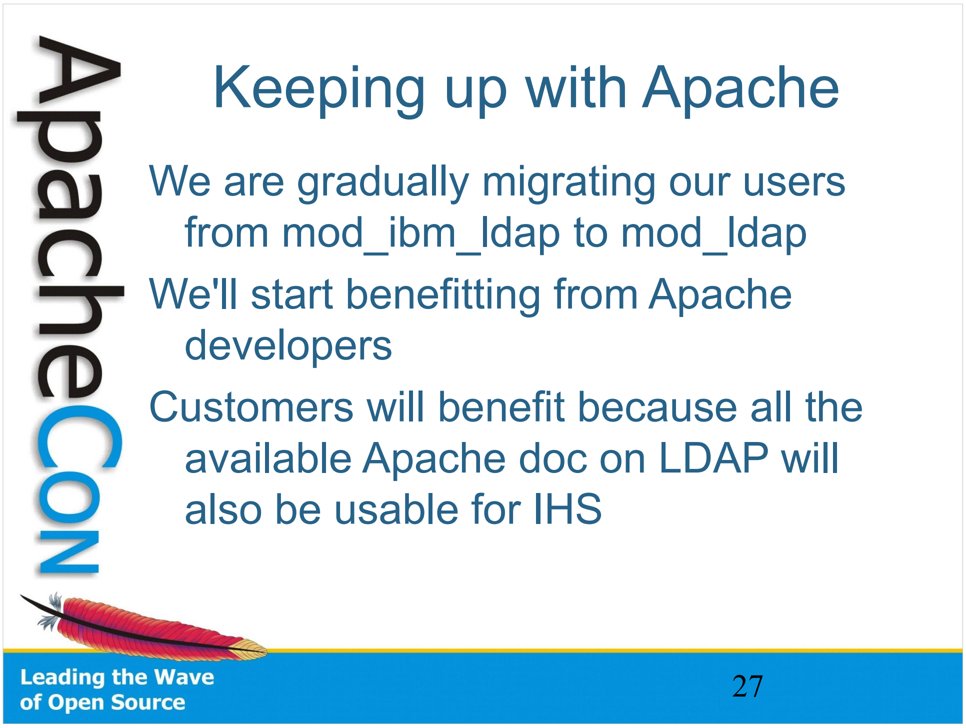
Leading the Wave
of Open Source

See wired article, cited in links at end of presentation.

Apache contributions (4)

We do more than fix bugs:

- Participate in development discussions on dev mailing list
- Assist other users on user list and IRC
- Improve documentation
- Add new features



Keeping up with Apache

- We are gradually migrating our users from mod_ibm_ldap to mod_ldap
- We'll start benefitting from Apache developers
- Customers will benefit because all the available Apache doc on LDAP will also be usable for IHS

Test systems

We have to keep build and test systems for all the platforms and release levels we have to support

III. Problem Diagnosis
When problems happen

Problem diagnosis

- Most customers can't run under a debugger, insert extra tracing, etc.
- So we need them to be able to easily gather information that we can use remotely to figure out what's going wrong

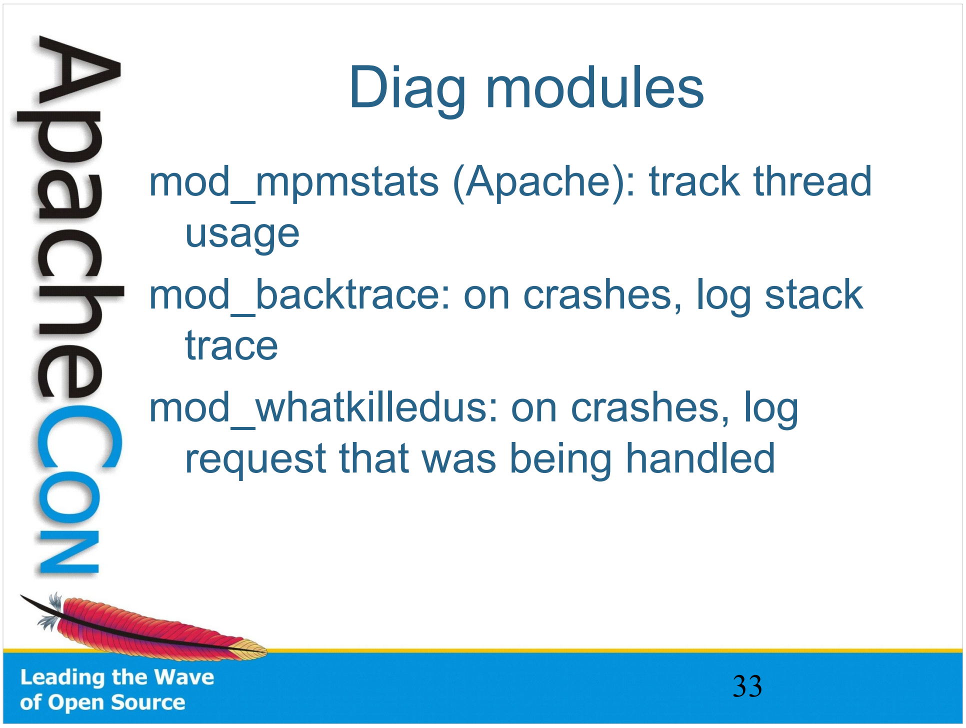
Must gather

For many large classes of problems, we have documents to tell the customer the information they "must gather" for us

E.g. for an SSL problem, we'd have them turn on additional tracing, recreate, and provide the logs, certificates, IP traces, etc.

IHSdiag Tool

- Our IHSdiag tool collects extensive system info: software levels, configuration, logs, network...
- Analyzes core dumps
- Monitors high CPU



Diag modules

- mod_mpmstats (Apache): track thread usage
- mod_backtrace: on crashes, log stack trace
- mod_whatkilledus: on crashes, log request that was being handled

Leading the Wave of Open Source

These are modules we ship in recent versions of IHS specifically for helping diagnose problems. They're also downloadable by customers for use with older versions.

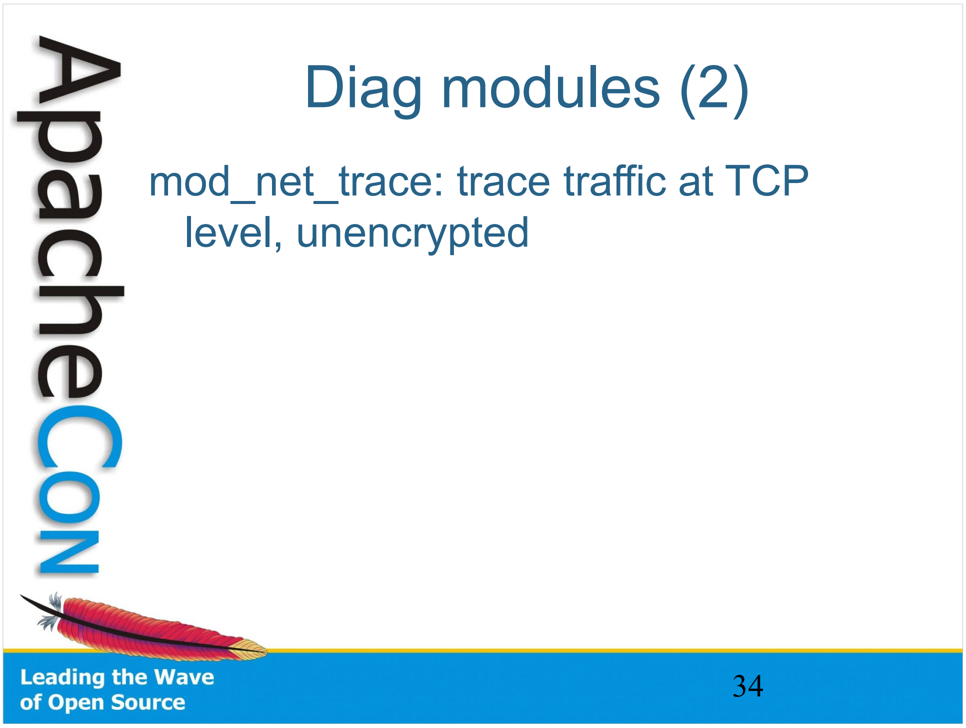
mod_mpmstats ships with Apache, the others are Apache-licensed and also available from http://people.apache.org/~trawick/

mod_mpmstats periodically dumps a line to the error log with statistics about what state threads are in (ready, reading request, sending response, etc.)

An IHS enhancement to mod_mpmstats can also show the name of the module where the most threads are, which helps in spotting hangs and bottlenecks.

mod_backtrace tries to log the backtrace of a crashing thread to the error log before letting the process end, which can save a lot of core file analysis.

mod_whatkilledus logs info about the request in progress on the crashing thread.



Diag modules (2)

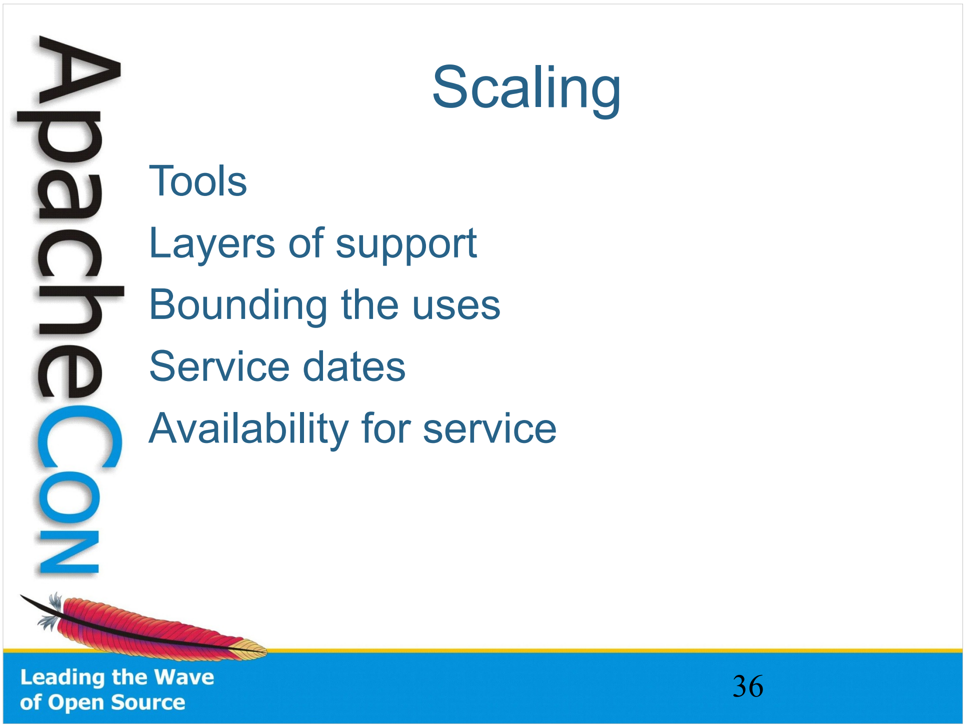
mod_net_trace: trace traffic at TCP level, unencrypted

mod_net_trace has a couple of nice advantages:

- customer doesn't need to know how to use an IP trace tool

- it traces unencrypted data, so we can tell what's going on even with SSL being used

IV. Scaling
More customers
Bigger customers

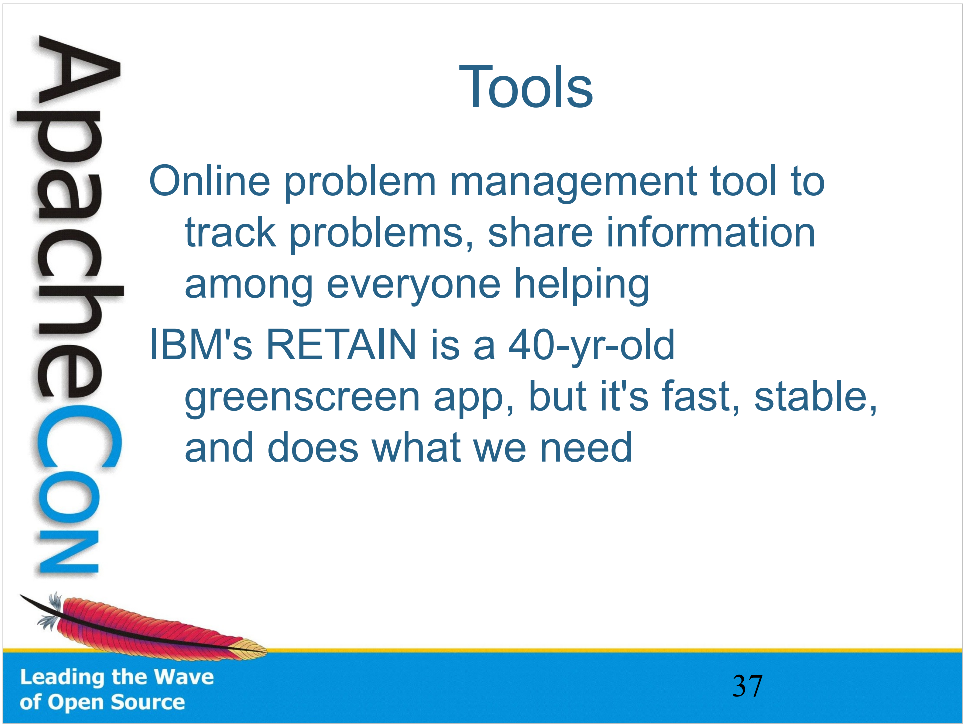


Scaling

- Tools
- Layers of support
- Bounding the uses
- Service dates
- Availability for service

Leading the Wave
of Open Source

Here are some things that help us to scale up our ability to support customers.



Tools

Online problem management tool to track problems, share information among everyone helping

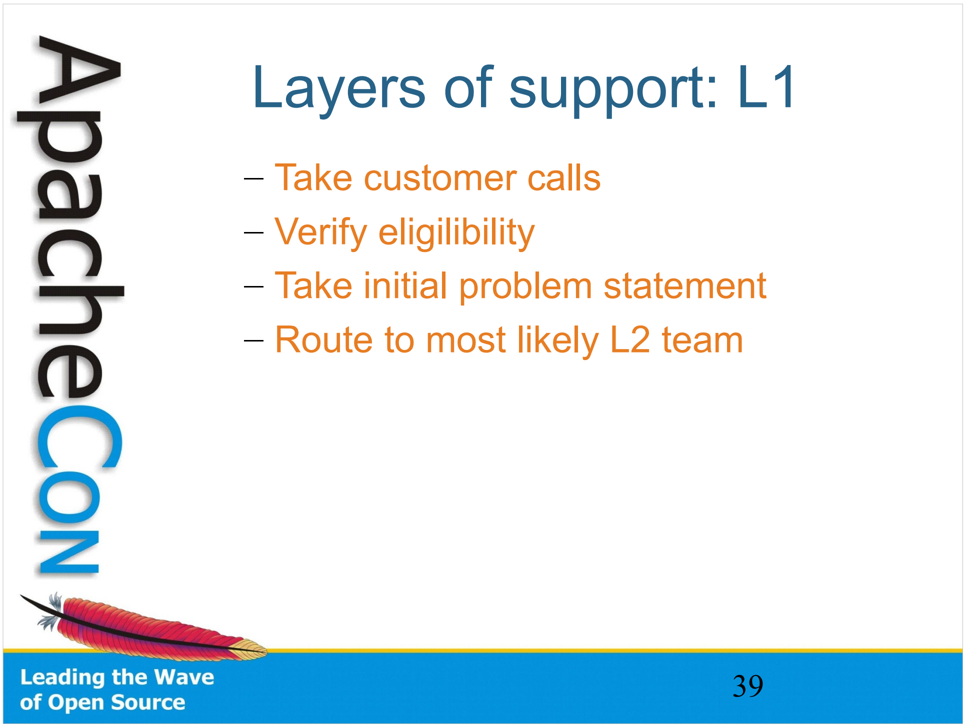
IBM's RETAIN is a 40-yr-old greenscreen app, but it's fast, stable, and does what we need

Leading the Wave
of Open Source

It's hard to underestimate the usefulness of having one place where all the information about a problem can be found.

Tools (2)

Information-gathering tools previously mentioned

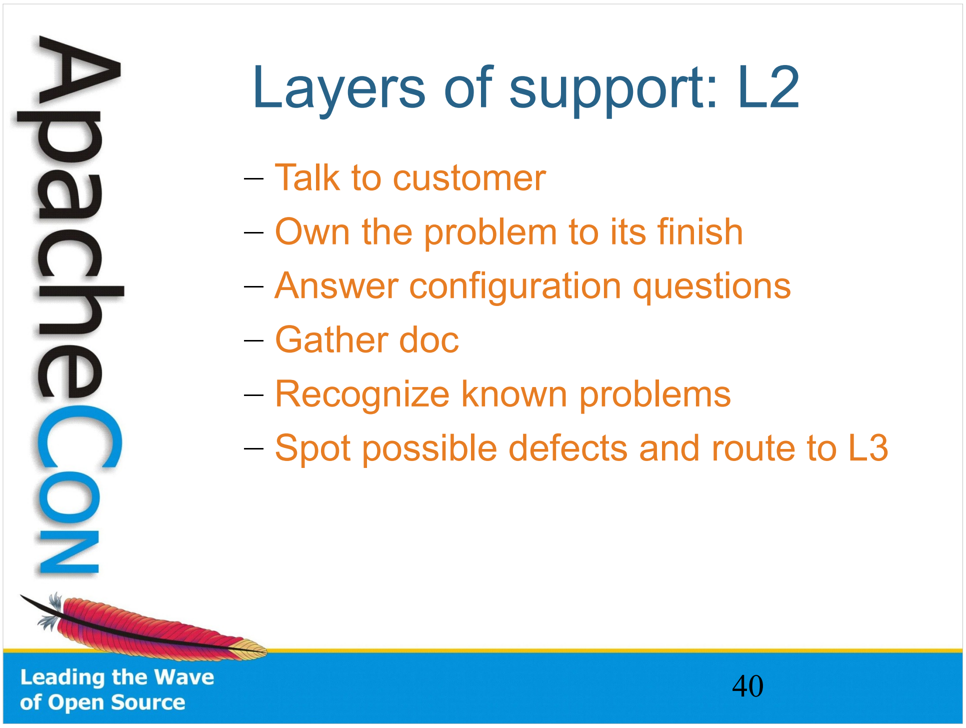


Layers of support: L1

- Take customer calls
- Verify eligilibility
- Take initial problem statement
- Route to most likely L2 team

Leading the Wave
of Open Source

By dividing support staff into different layers, each can focus on the work they're best trained for and most experienced with.

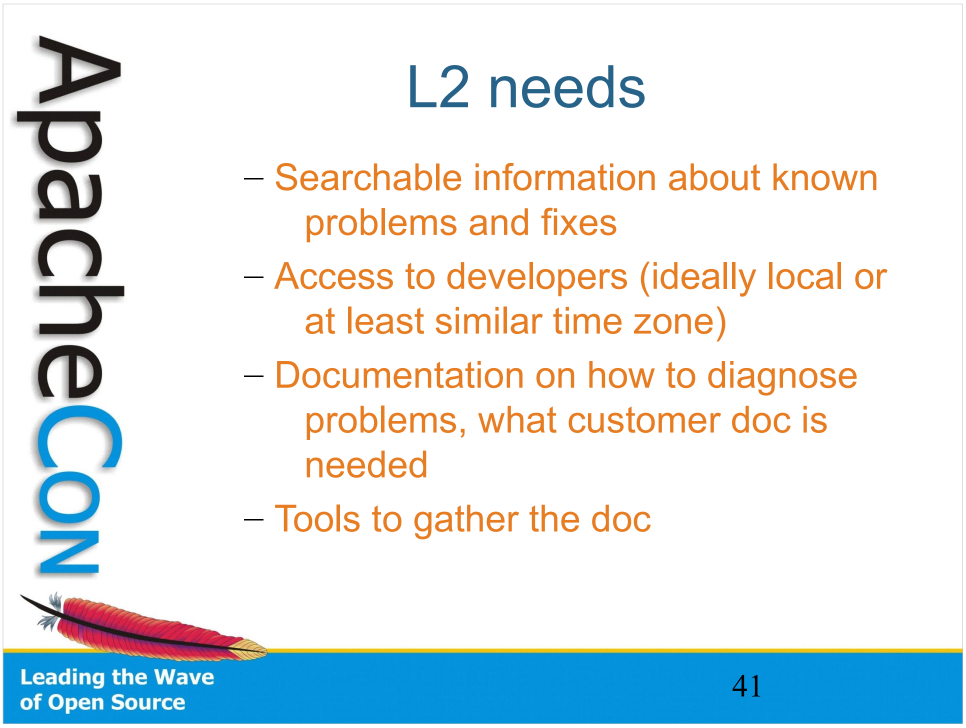


Layers of support: L2

- Talk to customer
- Own the problem to its finish
- Answer configuration questions
- Gather doc
- Recognize known problems
- Spot possible defects and route to L3

Leading the Wave
of Open Source

L2, as the problem owner, is the face of IBM to our customers who need support.

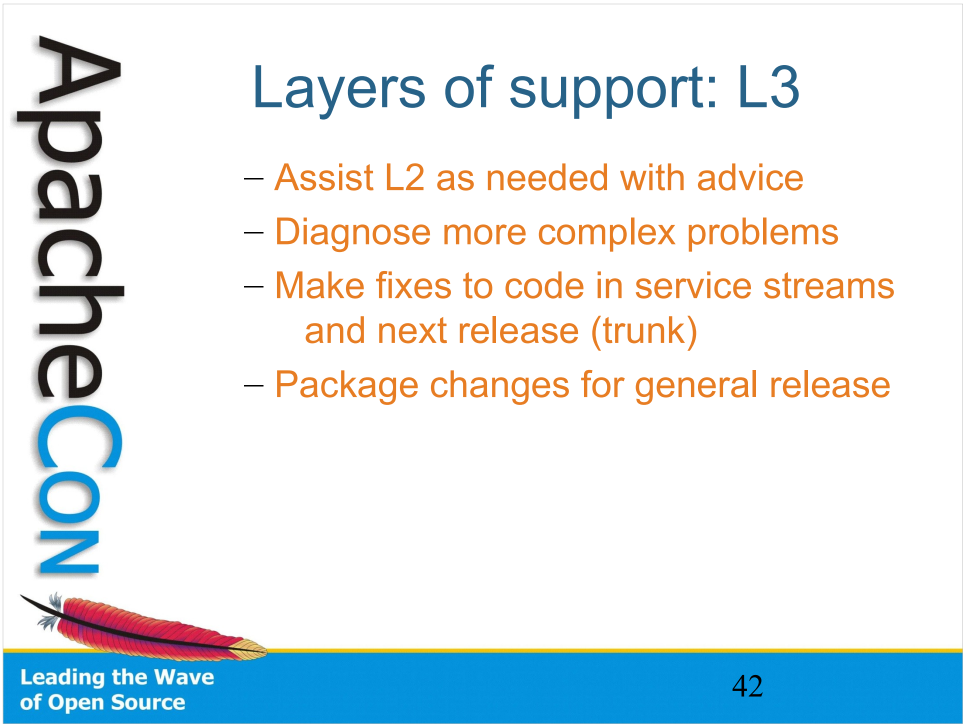


L2 needs

- Searchable information about known problems and fixes
- Access to developers (ideally local or at least similar time zone)
- Documentation on how to diagnose problems, what customer doc is needed
- Tools to gather the doc

Leading the Wave
of Open Source

Supporting L2 with the information and tools they need is critical



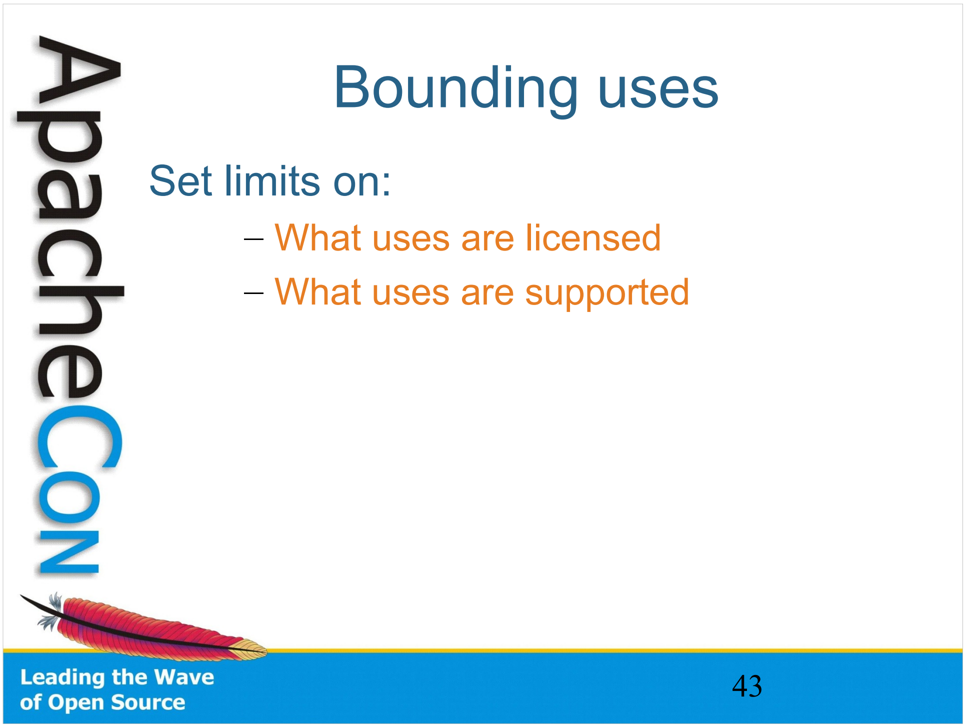
Layers of support: L3

- Assist L2 as needed with advice
- Diagnose more complex problems
- Make fixes to code in service streams and next release (trunk)
- Package changes for general release

Leading the Wave
of Open Source

L3 are the ones who actually work with the code.

They could be the same people as developers (in our case), or a separate "change team" that only works with fixes to released code.



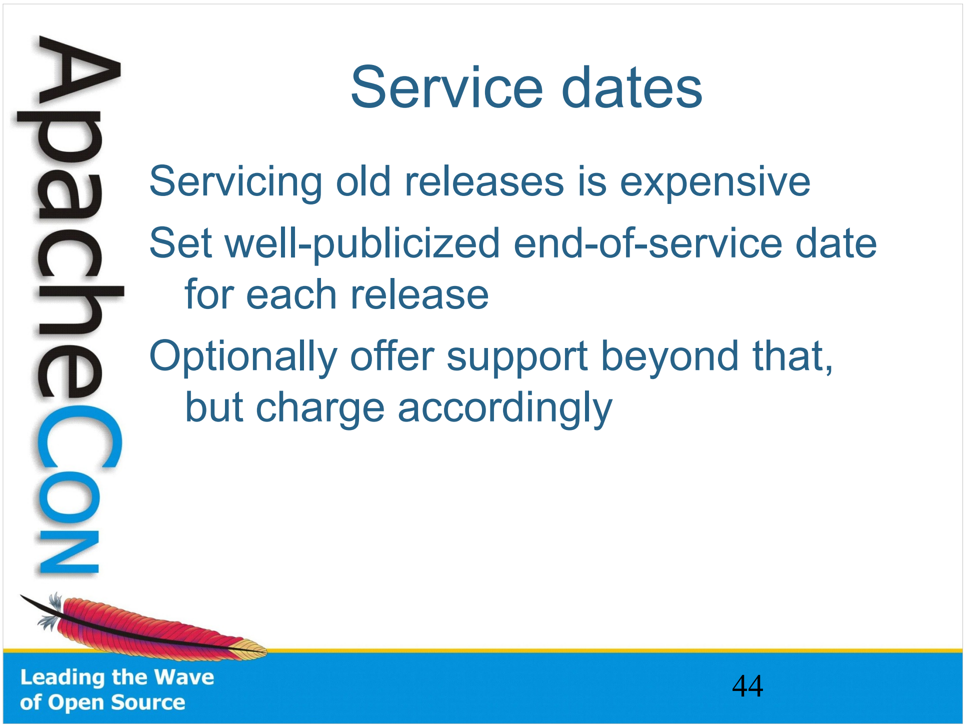
Bounding uses

Set limits on:

- What uses are licensed
- What uses are supported

For support to scale, it helps to set limits on how much support we offer.

Sometimes uses of IHS are licensed but not supported – e.g., someone who gets the free download of IHS from the web is licensed to use it for whatever they want, but they don't get any support for it unless they buy a support contract separately.



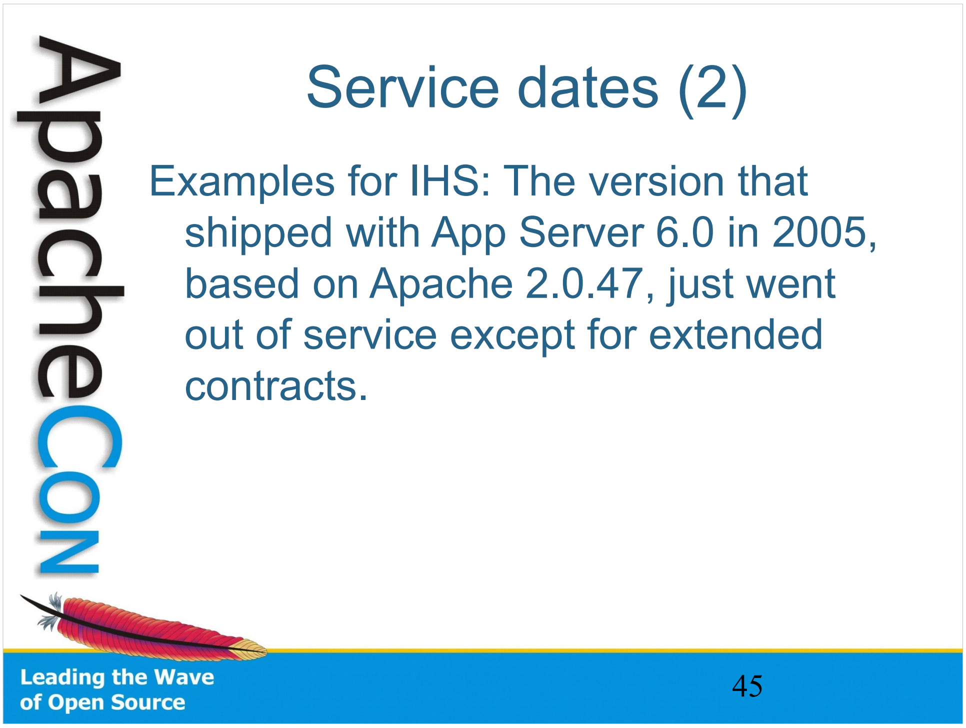
ApacheCon

Service dates

- Servicing old releases is expensive
- Set well-publicized end-of-service date for each release
- Optionally offer support beyond that, but charge accordingly

Leading the Wave
of Open Source

Another way to limit support costs is setting a time limit on how long we support a given release.

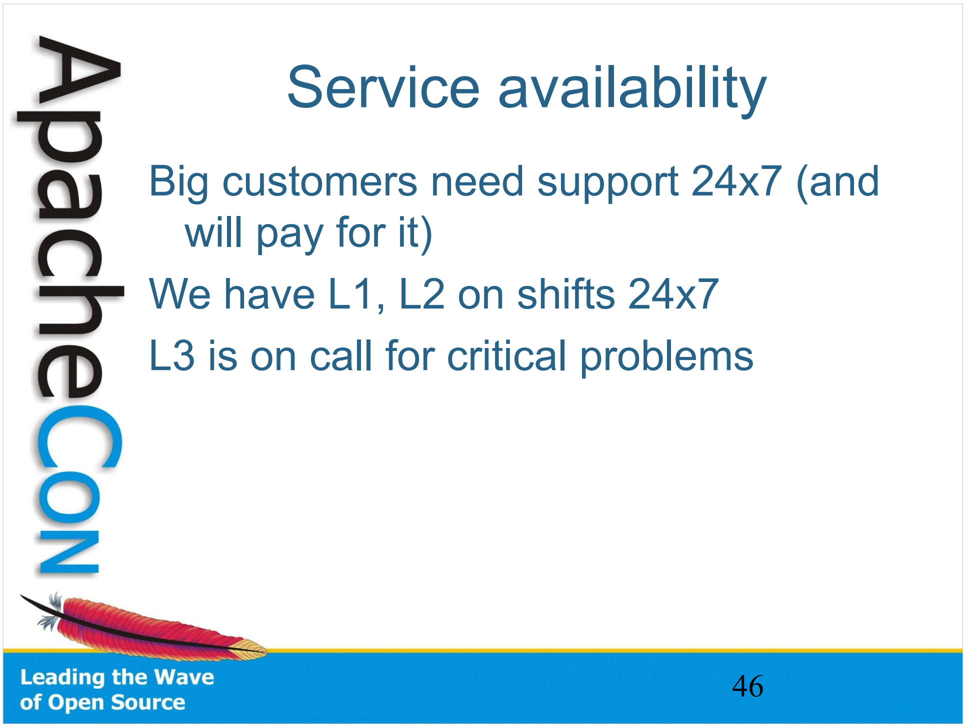


Service dates (2)

Examples for IHS: The version that shipped with App Server 6.0 in 2005, based on Apache 2.0.47, just went out of service except for extended contracts.

Leading the Wave
of Open Source

We'll still be servicing IHS 6.1, also based on Apache 2.0.47, for another year or two (no end date yet announced).



Service availability

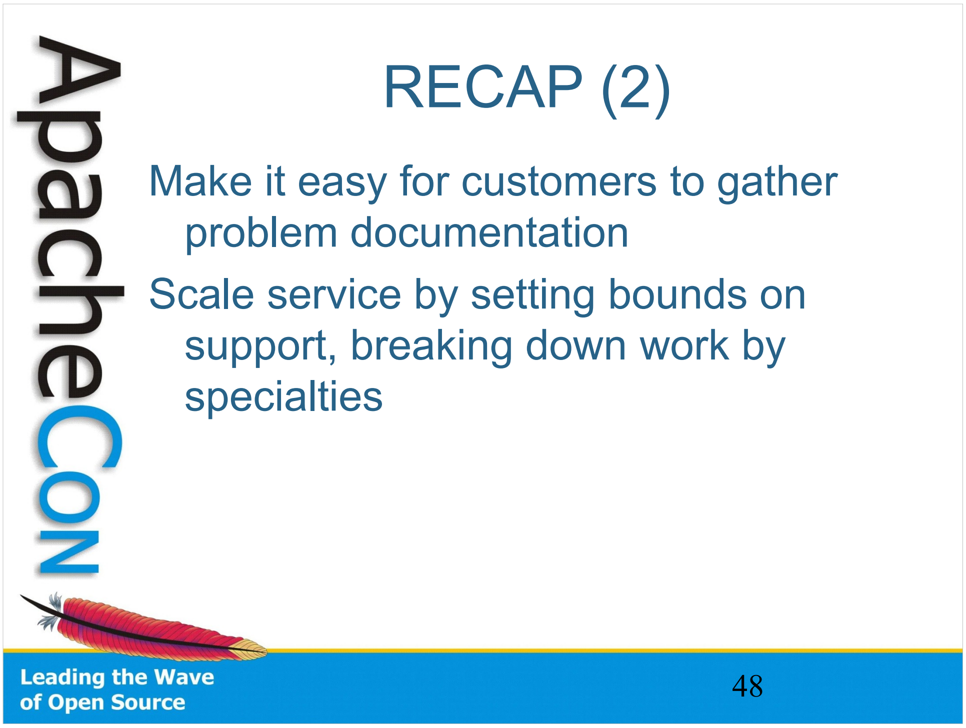
- Big customers need support 24x7 (and will pay for it)
- We have L1, L2 on shifts 24x7
- L3 is on call for critical problems

Leading the Wave
of Open Source

To scale our support to big customers, we need to meet their needs, including being available for critical problems whenever they happen.

RECAP

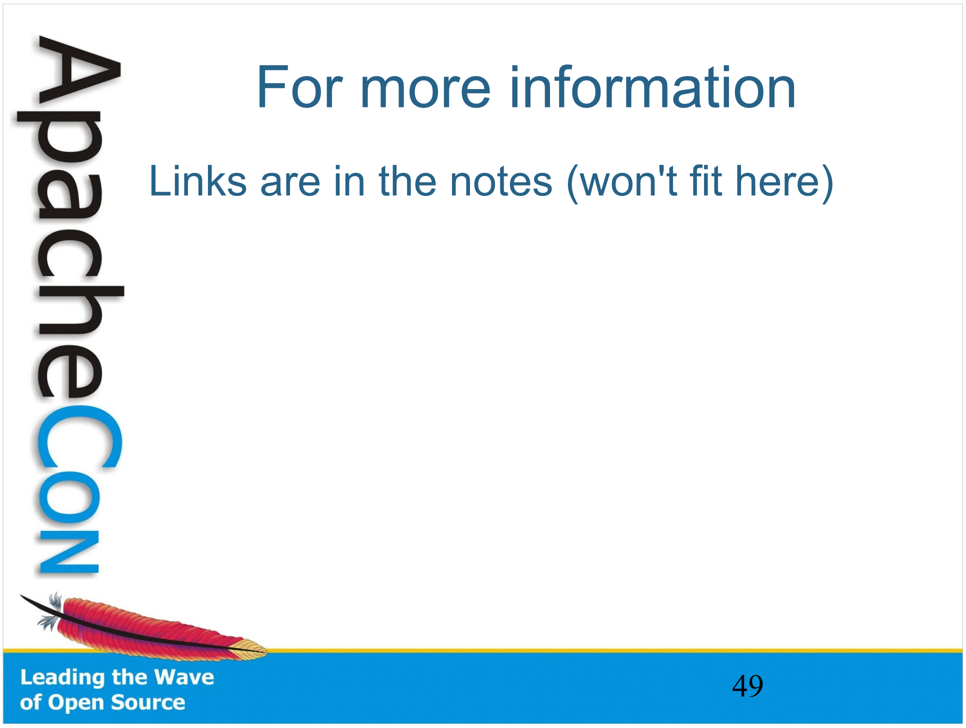
- Provide reliable, stable code by regression testing and controlling changes
- Take advantage of open source community but give back
- Add value to the open source



RECAP (2)

- Make it easy for customers to gather problem documentation
- Scale service by setting bounds on support, breaking down work by specialties

Leading the Wave
of Open Source



For more information

Links are in the notes (won't fit here)

ApacheCon

Leading the Wave of Open Source

IBM HTTP Server:

http://www-01.ibm.com/software/webservers/httpservers/

IHS documentation:
http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.ihs.doc/info/ihs/htmlnav.html

IHSdiag documentation and download:
http://www-01.ibm.com/support/docview.wss?uid=swg24008409

IBM announces it's adopting Apache:
http://www.wired.com/techbiz/media/news/1998/06/13117

Wired article citing IBM's contributions to the Apache web server:
http://www.wired.com/wired/archive/7.05/cobalt_pr.html