Apache in the real world - beating the inhouse bias

Good evening ladies and gentlemen. Or is it morning? My watch is still set to local time for a different location.

That brings me to the first problem I struck jumping between NT and Linux. What is UMT?

UMT

I worked on a dozen operating systems. I am used to GMT. When I found UMT, it took a while to find a clear reliable description of UMT. I gather UMT is what Amercans call GMT.

Perhaps Americans feared GMT would be interpreted as Government Mandated Time. If it is government mandated then clearly no one would use it.

Perhaps the name change was so Americans can claim they invented time.

All I suggest is expanding the typical Apache document to relate UMT to GMT on first mention. It will help people use Apache on NT. It will help people familiar with MVS and other operating systems.

Aim

To explain how an all Microsoft IT department justify, utilize and benefit from Apache.

To explain how an Apache promoter can promote Apache in to Windows environments.

Here are the business and technical cases for using Apache instead of IIS or other proprietary web servers.

Apache or IIS?

"Microsoft is the safe choice. Nobody is fired for choosing Microsoft."

20 years ago you could just substitute IBM for Microsoft.

To beat the incumbent, you need to be more than better, you need to be

VISIBLY BETTER!

Here is what makes Apache a good choice from a business perspective and what to do to make Apache the best choice. Listen if you are considering spending money to make use of Apache or spending money to promote Apache.

Apache in the real world - beating the inhouse bias

Save time

All web servers require some time to set up, learn and set up again. The process is universal because most software has to be set up to let you begin the training you need to set up the software.

I learnt about Apache on NT 4.0. Later I learnt how to use IIS on NT. Then I learnt to use Apache on Solaris. Each step should have been easier because I know more about web servers. The fact is IIS was harder to learn even with the prior experience. This is a common experience with Microsoft products. Why?

Microsoft tend to differentiate their products by replacing commonly used terms with Microsoft inventions. When I set up Apache, the options tended to have meaningful names similar to the terms people use to talk about web sites.

IIS used terms that did not relate to the rest of the web world. When I wanted to do X with IIS, I had to first find out what Microsoft called X then find out where setting X is hidden in a multitude of screens.

It is harder to get a person experienced with the web than it is to get someone trained on IIS. I suggest you get the people with web experience and use their first choice for a web server, Apache.

The webmaster may ask for Linux or Solaris to run Apache. Feel free to stick to NT if your site is currently 100% NT. You can develop, test and perform quality assurance on NT and easily change later when needed. You would not have that easy option if you selected IIS.

At some stage you will look at Linux or Solaris for production. If your site is large, you will run your web server on hardware separate from the database server and network controllers. If your web site is small, look at having the site hosted on a shared server.

Once your site is hosted, you will not need to know what operating system is used.

For the larger site with separate servers, you can use any operating system for your web server and still retain NT for hosting MS SQL Server. Use the operating system recommended by the person managing Apache.

Apache gives you more choice, more reliability, fewer restrictions and a lower long term cost. In the web server race, it is the horse to bet on.

Save money

You can train a person to use IIS in one week on a AU$2500 course. They will not understand the web.

You can take a person with web experience and train them to use IIS in one week on a AU$2500 course. They will not understand the IIS specific problems or why a small IIS maintenance update totally disables the web site.

I can point to occasions when ex Microsoft IIS "experts" could not understand what IIS is doing (or not doing).

What does Apache have that is magically different? Open source.

Open source lets the expert trace a mystery problem right through the code used to build the software creating the problem. You always have somewhere to go to get the problem fixed. While the problem may cost a lot to fix, at least you will get it fixed and save the $300,000 marketing cost you would otherwise need to replace all the customers you lose from ongoing problems.

As an example, I refer to St George Bank and stgeorge.com.au. St George is a bank with a function rich online facility. Unfortunate to use the online facility you have to use Microsoft's Internet Explorer 5.0. For security reasons, I do not have Internet Explorer 5.0 on any of my workstations. To use IE and maintain security, I would have to set up another workstation just for IE. That means all sorts of network security risks and set up costs.

It is easier to change banks than to safely install IE anywhere in my network. What will St George have to spend to replace my business? What will they have to spend to attract a few dozen companies to replace the businesses working toward leaving St George after hearing my recommendations on various bank's Internet facilities?

With IIS you have to first sell people IE 5.0 then try to sell them your product. Net month you will update IIS and suddenly find you have to make all your customers update to IE 5.5.

Prevent your web server doubling your marketing costs by getting a browser neutral web server.

Apache in the real world - beating the inhouse bias

Open source

Have a look at your car. Can you see the spark plugs? If yes then you have an older design car where anything can be repaired in a storm. You can keep your car going through flood, dust and electronic toll gates.

You can also recycle your car by just replacing a few dollars worth of parts.

If you cannot see the spark plugs, you probably have a computer controlled anti emission monster that has to be towed to a huge repair shop in a major city no matter how minor the problem.

You also have a car that can only be recycled by a $20,000 melt down and remanufacture from scratch.

Apache is the FJ Holden or Wileys Jeep that can be repaired in the bush, on the beach or out in the desert.

IIS is the plastic car that you use once and throw away. All repairs on IIS are performed by buying the new model. The new model comes with new problems so you have to wait for the next new model.

With Apache, you can see individual wires, clean battery contacts and top up the oil because everything is open, everything is visible. You can read the source code. That is why open source allows reliability, predictability and emergency repair.

Peer review

I hear people praise peer review as a way of building reliable programs. Peer review was fashionable in the 70s yet never produced reliable code. Why is today any different?

Back in the 70s, peer review meant exposing source code to like minded people, the person at the next desk and the boss that hired you both. Everyone on the peer review team would dot the i's and forget to cross the t's. In the 70s, peer review reinforced both good and bad habits.

Open source peer review lets dissimilar people argue over the value of crossing t's. The peer review may involve thousands of people, some from Europe, some from America and those brilliant people living in sunny Sydney, the Olympic city.

Someone in foggy London can peer through the sleet and smog at the Apache source code on the 12" monochrome screen attached to their 386 PC and send in a question to an Apache newsgroup.

The Sydney sider slipping through the surf can pop up the post on his or her WAP enabled waterproof mobile phone and post a reply saying the "," should be replaced by ";".

Down in the south of India, someone can post a coriander, cumin and chill fired dissention.

Everyone can join in. Even left handed people.

At the end of the discussion, an agreement will be reached. People will test both "," and ";" then publicize the results. If there is a difference, the better choice will prevail.

Peer means a person of equal rank or standing, equal before the law. Proprietary software makes everyone equally ignorant and peer review useless. Open source gives everyone the change to be equally informed and makes peer review of popular products, like Apache, perfect.

Monolithic programs

Think of a car. Imagine buying a car then finding that switching the headlights on also swaps the brake pedal with the accelerator.

This is a general problem with monolithic programs. Even large programs compiled from lots of little building blocks have the same problem. A small change in one place produces untold meaningless changes in many places.

Someone changes a building block so the headlights work better and a side effect is loss of the brake function.

I used to fix cars for friends. Mainly wiring changes. The cars would have a small number of wires driving headlights, brake lights and indicators. If there were the slightest fault with the wires, switching the indicator on would make the brake lights blink. Switching the headlights on would kill certain lights or cause others to go on permanently.

All I did was run a few extra wires. I made the headlight wiring discrete from the battery through the switch to the headlight and back to the battery. The same for the brake lights. In effect I replaced a monolithic system with a set of smaller utilities that did just one job and did it reliably.

Apache in the real world - beating the inhouse bias

Small discrete programs make installation, maintenance, testing, quality assurance and training easier. Anyone looking for a web server should look for a web server that works the same no matter what you choose for an operating system, scripting language or database.

Operating system neutrality

This is one of my few bitches about developing in the Apache web environment. That are few web servers that cross platforms and, from what I have seen, none attempt to solve this problem.

When I switch between Windows and Solaris/linux, I have to change "d:\asite\adirectory\" to "usr/asite/adirectory/". Fortunately NT understands / so I can start with "d:/asite/adirectory/".

These are my thoughts on achieving operating system neutrality and I use PHP as the example scripting language.

I do not use Windows 95 or 98 so do not know what they understand. If they still insist on \, perhaps Apache could have a standard variable set up at installation time to contain \ or /. When I have to write a script to calculate a directory I could include the contents of the variable $appropriate_slash.

Apache could also support some standard variables:

$server_other
$site_other

When I set up an Apache server, I put common external files in a directory structure parallel to cgi-bin. Referring to these files within scripts can be a pain. I usually set up a variable names $server_other and populate it from $DOCUMENT_ROOT.

I could set up an extra variable in the Apache .conf file but on hosted sites, the people controlling the servers usually ban access to the main .conf file.

If Apache came with a nice predefined variable and instructions on setting it up for various operating systems, the main scripting languages could jump on the variable as a way of standardizing scripts. ISPs would automatically point the standard variable to the base directory for the structure containing cgi-bin and other external files.

The same could be done on a per site basis so each virtual site in the server would have a matching external directory structure.

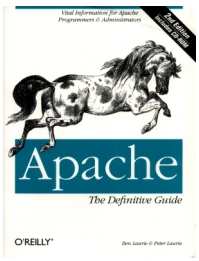
The next step would be the improvement of the include/require structure in languages like PHP. I do not know if Apache could help or if apache.org could support a common approach. The main problem is having an include path that includes a virtual site name.

I set up site xyz.com and the PHP.ini file with "/xyz/include;/common/include". When I add the next virtual site, I have a problem trying to include something that has the same name but different values. It would be nice to set the PHP include path to $site_other/include, $server_other/include. I could then set the PHP.ini include path once and forget it. 90% of the hosted sites I work on would get a one off setup then never change.

Apache in the real world - beating the inhouse bias

Books

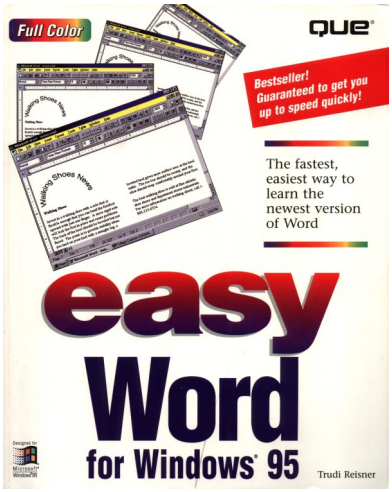
When I read about Apache, the book I read most often is Apache The Definitive Guide by Ben Laurie and Peter Laurie, published by O'Reilly:



This is a great book for people used to setting up software similar to Apache and able to experiment on a spare machine. I would not recommend it to others. There is a need for a better beginners guide.

I read some other books on Apache. I would not recommend any of the other books to the people who would not cope with Apache, The Definitive Guide. What is needed for the rank beginner in a Windows environment?

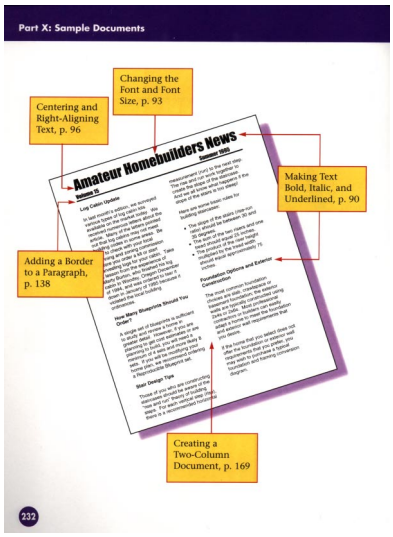
I would start with something like the Easy series from Que. I enclose illustrations from Easy Word for Windows 95 by Trudi Reisner:



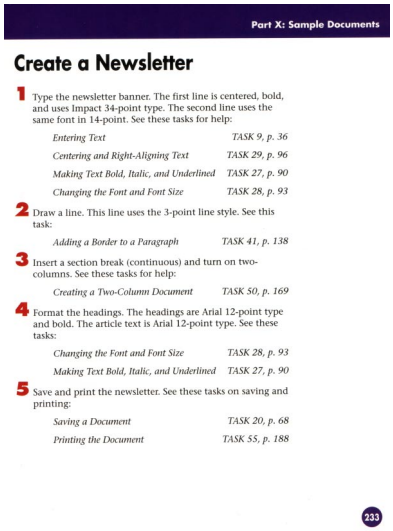
The use of diagrams, step by step illustrations and blowups in the Easy series, lets anyone step through just about any product.

Apache in the real world - beating the inhouse bias

Note the diagrams are presented as images for the visually oriented:

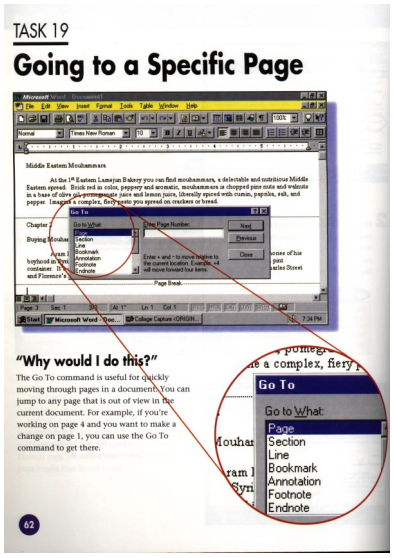


and text for everyone else:

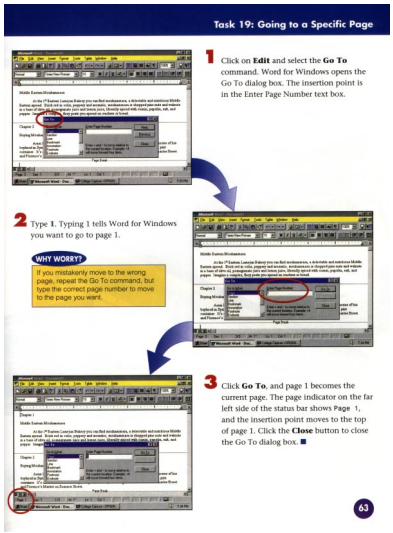


Apache in the real world - beating the inhouse bias

Blowups let people see where on a screen, they should focus and see what to focus on:



Arrows let people follow the logic of the illustrations without continually referring to the text:



Next I would add diagrams showing how Apache steps through decision making processes. How does Apache step through applying information from the .conf files in the config directory and the web site directories? How does Apache step through selecting the right web server? There are quite a few decision processes that could be easily explained with diagrams.

If you are writing books for Apache, this section is for you.

If you are buying books on Apache for your staff, at this stage I would suggest letting your staff choose a few to suit their differing styles of learning.

Decision maker

How many times do you sell Apache to an Information Technology Manager then find the site using IIS? It is a pain. The solution is to talk with the decision maker.

I will step through some decision making aspects that throw the IT decision out of IT's hands.

- The IT Manager has discretionary power to sign cheques up to $10,000. Everything else goes to the General Manager. IIS is $1,000. Apache is free. Surely the IT Manager can sign for Apache?

Not always. The IT Manager may have to aggregate all purchases for a project. Apache is one of many products needed for an Intranet project. The total is $35,000. The General Manager has heard of Microsoft but not Apache.

- The IT Manager has discretionary power to sign cheques up to $10,000. Everything else goes to the General Manager. The General Manager knows about Apache. Surely Apache will win?

Not always. The General Manager has already signed a corporate agreement with Microsoft. The agreement includes 1 copy of IIS. It seems silly to not use the "free" copy. Once one IIS is in use, it is easier to stick with what people know.

- The IT Manager wants Apache. The General Manager wants Apache. The project is contracted out.

The contractors may simply install a 120 day evaluation of IIS, complete their work then walk out. The IT Manager moves the test site in to production. nobody is willing to delay the project to replace IIS with Apache. The short term solution is to buy a licence for IIS.

- Management does not care. The contractor wants to use Apache. This must be a win?

Management does not specify a web server in the contract. They give the contract to IBM. IBM contract out all the work to an agency. The agency contract out to HelpNet. Out of ignorance, the Agency insert IIS in to the contract.

Contractors

I mentioned contractors as an influence in what happens. Here are some of the worse chain of influences I have seen.

An agency called me and asked me to assemble a team to build an Intranet site. The Agency supplied some information including the fact that they received the contract from IBM. IBM accepted the contract for the original customer with little detail in the specifications and no one to complete the work. The agency thought I could complete the work quickly and share the $250,000 IBM was offering. I, and my helpers, were booked out for 6 months so I turned down the contract.

A week later another agency called about the same project. They had received the contract from one of IBM's competitors. Apparently the first agency could not find someone to do the work so they let the contract to one of the big outsourcing companies. That outsourcing company immediately handed the contract on to another agency.

The second agency, the fourth in the chain, suggested I quote $50,000. After discussion, they revealed they accepted the contract at $70,000. I turned the contract down again.

Apart from the fact that $200,000 disappeared in nothing but agency fees, I noticed the contract gain a few details not inserted by the customer, IBM or the first agency. The contract now specified the contractors had to have experience with PhotoShop and various Microsoft products. Even though the contract did not seem to specify which software should be used for the web site, it did specify the developer should have several years experience with Microsoft SQL Server.

Which database do you think I would choose if I was a Microsoft SQL Server specialist? Which operating system would I need? Having installed NT and SQL Server, which web server would I choose off the

same Microsoft CD?

If you want to sell Apache in to NT servers, talk with everyone. Let the database developers hear about success stories where NT shops run SQL Server behind Apache.

SAS is big in sites that used to have IBM mainframes. Find recommendations from SAS developers.

Some companies install SAP then change the whole of their IT infrastructure to suit SAP. Who has every published information from SAP in an Apache based web site? How did they do it? You want to be able to tell SAP users.

ISPs

Millions of people published pages using FrontPage. Nobody notices because all the pages look the same and all have the same JavaScript errors that make the pages useless.

Unfortunately a number of ISPs dumped UNIX because they bought Microsoft's story about FrontPage. The ISPs attacked the FrontPage "One page" web site hosting market. Somewhere along the way they dumped those pesky people with the Unix experience because the Unix people would not learn NT.

Now the world is populated by huge ISPs that are, in turn, populated by 17 year old kids who went from High School to an MSCE "Boot Camp" to being Senior NT Systems Administrator all in 5 weeks.

Front Page extensions are now available for Linux and some ISPs offer the combination. I suggest what is more important is to get the "instant web site" market to ask for Apache. Look at the Intel publicity campaign for "Intel Inside". Web pages do not know if they are published to a browser on a Pentium or an Athlon chip yet some people demand Intel based web servers.

The only x86 style processor in the whole world that is not "Internet ready" in the Intel processor with the divide error. People still prefer the expensive, slow buggy brand because television told them to.

Perhaps a long term approach to selling Apache is to get Apache in to the TV shows for preschoolers. One well known nursery rhyme could be presented as an IIS based web site falling off the wall. The rescuers would be a bunch of Apaches ready to put the site together again.

Reliability, A, Serviceability

IBM made the term RAS famous. Microsoft made NT more reliable than Windows 95. Microsoft made NT more serviceable than Windows 95. What IBM and Microsoft missed is Predictability. Here is a classic example of Microsoft's approach to predictability:

While typing this document in Microsoft Word 97 , I closed the file, made lunch then opened the file again. Despite the .rtf file extension, Word opened the file as something other than RTF. I had to close Word and open the file again to see it as an RTF document. The same happens with .doc documents. They randomly open as text files and I get to see all the document's formatting controls.

When I last opened this document, the English words were highlighted as misspelled. I checked the language setting and found it had reverted to American. Why is it that every time I open a document, I have check which language Microsoft's product chose?

The next day I opened the document and the language had changed again. This time the language setting in the style had "(FE) English (United States), (Other) English (Australian)".

First there was the operating system location setting then the keyboard language setting. By Office 97, the keyboard language setting had only a random effect and applying any maintenance randomly reset the settings. Each release of Microsoft Office looks in different places. Word works differently to Excel.

Think of the poor person installing Microsoft software for a living. His/her customer complains about a word document having the wrong behavior. To work out what Word will do, you have to know:

- operating system release
- operating system service level
- operating system settings
- MS Office release
- MS Office service level
- MS Office settings
- MS Word release

MS Word service level

MS Word settings

The document settings

Unidentified seemingly random factors

I hope Apache never reaches that stage. Replace RAS with PRAS. Predictability is number one priority. Back in the days of 10,000 people using 1 mainframe via dumb terminals, the predictability of the mainframe was not important. Whatever the mainframe did was what it was supposed to do. The most important task was simply to keep it going. Reliability.

Now the same 10,000 people have their own computer and the 60 support staff are flat out trying to find out why 5,780 of the people see document measurements in inches instead of normal measurements.

There are software update systems that can update 10,000 PCs to a new maintenance level of MS Office in a few minutes. That means every printer in the whole organization instantly stopping to because Word is suddenly asking for letter size paper instead of A4. Predictability is what we need.

To put Apache on the desktop of every PC, you need to look at replacing Microsoft's PowerPoint with a HTML based equivalent. PowerPoint users have years invested in learning how to make PowerPoint work despite it's design faults. Microsoft may never be able to fix PowerPoint because it would then be incompatible with the millions of documents already created.

If you give the same people a combination of Apache, PHP, MySQL and the like, give them absolute reliance on the Apache portion. Give them diagrams showing where Apache picks up settings and in what order they are processed. Give the diagrams to PHP and all the other people producing Apache based products so all those products can say where the settings come from.

What is in all the system variables? Why does $PHP_SELF give one result on NT and another on Linux? I ask this in an Apache forum because the PHP documentation says some of the system variables are supplied by Apache. The PHP documentation does not say which come from Apache or what processing happens on the way. I presume $PHP_SELF is based on a PHP interpretation of an Apache variable. Without reading the source code, I cannot see what happens.

Online help

Apache, The Definitive Guide by Ben and Peter Laurie, does not tell me exactly what variables are available from Apache. Books are naturally out of date. Relating books to the Apache online documentation is difficult. The easiest answer is to have the Apache documentation on the CD with Apache. That creates the problem of getting Apache working before accessing the documentation explaining how to get Apache working.

Could Apache be set up to run to servers per machine? The first is configured to work a bit like the Microsoft help system. It is associated with pages named **.help**. It starts up whenever a .help page is accessed and just sits there. If someone is experimenting with the configuration files on the regular Apache, the ApacHelp© server just keeps on running on it's own files.

You would need an NT version of apache that could be started via an association with a file. The association for .help files would point to "c:\Program files\Apache\Apache.exe" "%1". Apache would start up, look at the input type then read help.conf. help.conf would specify a separate configuration to the html.conf file.

The right configuration of Apache and the right support software could lead to a replacement for the Windows help system and a huge uptake in the computer based training area. You would not give up on anything currently in Apache, just add an ability to run a local configuration stated by association and the ability to keep it running when someone is testing the configuration of another Apache.

MS Access

What does MS Access have that I cannot get from Apache, PHP, MySQL and phpMyAdmin? $500 billion of publicity? Relationships? SQL generator?

An MS Access killer could be a short time away. Why is it important? Lots of people create something in Access and share it will 9 work mates. Suddenly 90 people are viewing the data all over the company. The company is in breach of it's licensing agreement for MS Access. The company would like to upgrade to MS SQL Server but simply cannot find the people to perform the conversion.

What if the original work was in PostGreSQL? What conversion is there? PostGreSQL on the desktop

would save a lot of future work and licensing problems. There is still a little bit of work needed to kill MS Access. Everything seems to be in place to get the work complete.

An SQL generator is easy to build and there are already some close to MS Access's level.

Relationships are already in PostGreSQL.

If you give me $500 billion dollars, you will get publicity.

Conclusion

Apache is a better long term choice than IIS. Choosing Apache produces no disadvantages and many advantages.

Choosing a non Microsoft product may make Microsoft work a lot harder to get your business in the future.

Choosing Apache will give you the maximum choice.