Advanced Web Site Management and Techniques Using WebObjects

by Theresa Ray of Tensor Information Systems

Sponsored by Apple Computer, Inc.
Worldwide Developer Relations Group

Advanced Web Site Management and Techniques Using WebObjects

by Theresa Ray of Tensor Information Systems

Designing a cost-effective, manageable, technically powerful web site is not an issue to be taken lightly. You've worked hard to develop your application, but if the users have difficulty connecting to that application, or poor performance from that application due to insufficient server resources, all of your hard work may be in vain. This survival guide addresses the questions and issues faced by companies seeking to provide a scalable, robust deployment for their applications

On-line Technical Documentation

One of the best resources for all WebObjects programmers, regardless of experience level, is Apple's on-line technical documentation. The documentation provided there, specifically the sections on Serving WebObjects and Tools and Techniques, has an excellent introduction to web site deployment and the terminology necessary for continued web site discussions. This survival guide assumes that the reader is familiar with the information presented there. Apple's technical documentation can be found at http://developer.apple.com/techpubs/enterprise/WebObjects/WebObjectsTOC.html

Decisions, Decisions

There are many important decisions to be made in the deployment of a WebObjects web site relating to the configuration of the application and the server:

1) What hardware platform should be used?
2) What operating system should it run?
3) How much memory should the server have?
4) Should the entire application - including web server, WebObjects application, and database - be hosted on a single machine or distributed among multiple machines on a network?
5) How many instances of the web server and WebObjects application should be run?

The answer to these questions is a hotly debated topic. There is no single answer that provides a cost-effective, technologically sufficient answer for the wide variety of applications that can be created. Instead, a number of issues should be considered in determining the answer to these questions. Once these issues are considered, you may find that you have answered at least part of the questions above. The rest will most likely be a trade-off between cost, availability, scalability, and knowledge (do you want to deploy on NT if your company is full of UNIX experts?) The most common issues are listed below.

Number and Type of Users

How many users will your application have? Certain intranet applications have a limited number of registered users. Other internet applications are designed to handle tens of thousands of hits every day (or more). The number of concurrent users is important for determining how many instances of the web server and WebObjects application you need to run. This in turn affects what type of hardware you need for deployment (and whether you need to distribute the application across multiple machines). Keep in mind, though, that the number of concurrent users does not equal the number of WebObjects applications you need to run. As long as your average response times for the application are low, it may be able to accommodate one hundred or more simultaneous users.

Will the users be accessing the application in a fairly even pattern throughout the day (e.g. an intranet-based phone book), or will the application have peak times of heavy use (e.g. an internet application which has just e-mailed it's repository of registered users to announce a special offer)? If particularly heavy peak times will occur, you may need to make a business decision which trades off the number of concurrent users the application can serve against the amount of hardware your company is willing to supply for the application.

Just before deployment, you might want to put log statements in the Session's -awake and -sleep methods in order to determine the response time of your application ([self logWithFormat:@"Awake in session called. Pertinent data = %@", someVariable]). Each log statement is stamped with the system date and time, allowing you to determine the length of time needed for the request. Run through your application and see where the longest response times occur. This can help you determine the maximum number of users your application can handle. There is a little more time associated with receiving the request/responding to the web server, but this test provides a good estimate. If your average response time is 1.5 seconds, your application will be continually busy at just less than 60/1.5 requests per minute.

The web server logs can also assist you in determining the maximum number of users for your application. Were there requests that timed out? What are the time stamps in the log for request and response? Make the same calculation as above, but base them on the server log times.

Application Duration and Performance

Is the application meant to be a 24x7 application, or an 8 to 5 application? If a 24x7 application is desired, the operating system and hardware choices should reflect the reliability required of such a system. Additionally, a 24x7 application will need human support teams available around the clock. Who is responsible for making sure that nothing failed at 3 am? Who is responsible for attending to the problem if it occurs, regardless of the hour? The WOMonitor will be a valuable resource for any deployed WebObjects application, providing many valuable services. The WOMonitor manages the WebObjects.conf configuration file for you, defining the application instances and ports they are running on. Additionally, the WOMonitor has the capability to restart applications if they crash, and to automatically cycle applications after a specified duration and at a user-specified time.

What kind of performance is required of the application? Most WebObjects applications are written for traditional quick-response applications where the process responds to a request in around a second. In fact, for scalable applications, you should design your WebObjects application to respond to most requests in less than a second (application response time). Deploying such an application for only a few users would probably only require one WebObjects application process (unless you wanted a second for redundancy).

However, occasionally someone will write a WebObjects application allowing the users to submit an extensive query to the database - one that requires several minutes or more to execute. In WebObjects 3.5.1 and prior versions, both the request/response loop of WebObjects and the Enterprise Objects Frameworks which connect to the database are single-threaded. This means that while the WebObjects application waits for the result from the database, the user waits for a reply to their submission, and no other requests are served by that WebObjects application process. If there is only one WebObjects application process, everyone's requests queue up until the long query is complete. Therefore, if you are deploying this type of application to even as few as five users, you most likely will want to start multiple WebObjects application processes to avoid the bottleneck created by a long query. If the long query is frequently accessed, you may need to start as many WebObjects application processes as you have simultaneous users.

Thinking Ahead

How many more applications will you be creating in the near future? In order to most efficiently use the WebObjects deployment licenses you purchase, you may want to consider a dedicated machine for just the web server and WebObjects application server if you plan to roll out several applications in the near future. This depends, of course, on the answer to the above issues for each application you intend to roll out. If you

tend to use the same database engine for all your applications, this may also allow for easier administration of those databases by co-locating them on a single , separate machine. Additionally, if you know you will be deploying more applications in the coming future, you may want to consider this in your hardware and operating system choices. This is not a firm requirement, though, since WebObjects also gives you the capability of scaling across additional machines (as opposed to buying a new CPU, more RAM and more disk space for your existing machine). You can either use the WOMonitor to set up the configuration file for deployment across multiple machines, or you can manually edit the WebObjects.conf file.

The WebObjects.conf file is a configuration file that the WebObjects cgi program uses to determine how to connect to the application instances. You should put this file in your web server's cgi-bin directory (or nsapi, or whatever adaptor you are using), where the link or copy to the WebObjects adaptor resides. The file format is as follows:

AppName:instanceNumber@applicationServer portNumber

Consider the following file:

AOLRewards:1@huxley 3000
AOLRewards:2@huxley 3001
AOLRewards:3@bogus 3002
AOLCustomerService:1@huxley 3005

There are three instances of the AOLRewards application defined. The application number is any unique number you wish to use to identify the instance of the application. Two instances of AOLRewards are running on a machine named huxley, while the third is running on a machine named bogus. The port number is used for communication with the application instance. You must specify the application instance number and port number in your startup command in order for the WebObjects adaptor to be able to find the application. The normal startup command you probably used during development:

AOLRewards -d /WWW/sites/aol/htdocs AOLRewards

does not provide the application and port number, so the WebObjects adaptor would not know how to communicate with this instance given the WebObjects.conf file above. You would want to change the startup command to:

AOLRewards -a WODefaultAdaptor -n 1 -p 3000 -d /WWW/sites/aol/htdocs AOLRewards

which uses the -n and -p options to set the application instance and port respectively, allowing the WebObjects adaptor to communicate with this instance of the application. You may append log redirection and backgrounding of the application as you desire to the end of any startup string (such as 1>>file 2>&1 &)

What Does Your Company Already Have?

What resources are already available within your company? Is there a machine available for use as the deployment server - maybe left over from an old project? You might have an existing machine that you can share among multiple projects. If you don't have a single machine with enough idle overhead to accommodate the project, you might be able to distribute the application across several existing machines using the techniques discussed earlier.

What skills do your available administration personnel already have? For large-scale enterprise applications, you usually want to stick with configurations that your company is already familiar with (providing you have experience with enterprise deployment already). But if you're deploying a small-scale intranet application, it might provide a good opportunity for your staff to broaden their skill set.

Security

The type of security your application needs depends on a number of factors. Does your application contain sensitive information which is only available to a select group of personnel? Do those personnel have access to your company's intranet? If so, password protection at the application level (and for access to the database) is probably sufficient.

But what if those personnel need access to this sensitive information from outside the company - via the internet? There are several answers to this question, but the most common configuration is to keep both the WebObjects application and the database within your company's secure network, and to provide a secure web server which is accessible via login and password from the internet. Keeping sensitive information within your company's secure network (or inside the firewall) helps reduce the risk of unauthorized personnel gaining access to your sensitive data.

Physical Machine Characteristics

Your answer to these issues will help determine your answers to the five questions at the beginning of this survival guide. A number of the issues are interrelated. As you add WebObjects application processes to run simultaneously, you will need more RAM for your server, your server size may increase, and your server platform may even need to switch from NT to a UNIX system such as HP-UX or Solaris.

RAM calculations are not too difficult to make. First, determine the amount of memory required by the operating system. Your system administrator can provide

those numbers for UNIX machines, and I would not recommend less than 32 MB for an NT workstation. Then determine the amount of RAM needed by your application. Start the application, and check the amount of memory it is using (use the ps command on UNIX or Task Manager on NT). This is the BASE amount of memory your application requires. As sessions are created and destroyed by your application, your memory requirements will fluctuate. Connect to the application and run through a typical scenario. How has the RAM usage changed? This is the amount of memory needed for a typical session. Add in the same amount of additional memory for every concurrent session you expect to serve. This is the application's memory requirements. Multiply this number by the number of instances you plan to run and add the amount of memory required for the OS (and database if necessary). This is the amount of memory (RAM and swap space) that your machine requires.

Overwhelmed?

If you find that considering the issues listed in this document is overwhelming or that it does not help you answer the questions posed at the beginning, it may be time to contact your Apple Enterprise Technical Support Team for help. Alternatively, you could consult with one of the many companies who have extensive experience with issues such as these (Apple provides a list of consultants on their web site at http://enterprise.apple.com/Alliances/Partners/SIListings.html).

Fortunately, there is no need to feel overwhelmed or under pressure to make a decision that must last for the next five years. The flexibility provided by WebObjects prevents the answers to the questions this document poses from becoming a do-or-die situation. If you initially deploy on NT and find that the server is just not powerful enough for your situation, it is not difficult to migrate the application to Solaris or HP-UX.

My first port of a WebObjects application was back in the days of EOF 1.1/WOF 2.0. I wrote the initial application that accessed an Oracle database on Mach - those old black NeXT machines for those of you new to this technology - and had to port it to Solaris. I'd never really paid attention to the makefiles before, I had no prior knowledge of what was needed to make the port, and WebObjects 2.0 wasn't exactly as slick as WebObjects 3.x regarding cross-platform ports. It took me four hours from start to finish - including the installation of WebObjects on Solaris. Since then, I've done probably every combination of platform ports and never run into serious trouble. I routinely develop on one platform and deploy on another. So this really is a reliable alternative.

One of the only changes you need to make in porting to UNIX deals with linking in the database libraries. On Solaris or HP-UX, you need to statically link in the database libraries and frameworks. Create an additional makefile and include it in the Makefile.preamble of the application. I follow the convention of naming the new makefile MakefileEOF.preamble-operatingSystem-machineName so that the makefile

is only included when the platform requires it. The addition of machine name is optional. I have run into situations where the development and deployment databases were different versions, requiring a different library link order. By specifying machine name, the right makefile is automatically used on the right machine. Add following line to the Makefile.preamble to include the new makefile:

-include MakefileEOF.preamble-$(PLATFORM_OS)-$(HOSTNAME)

A sample of this new Makefile is shown below, and also provided in the DodgeDemo application distributed with WebObjects.

```
# Client library location
INFORMIX_HOME = /informix/informix7
# Statically link in the database EOAdaptor
FRAMEWORKS += -framework InformixEOAdaptor
# Include other .o files - not usually needed
OTHER_OFILES += /informix/informix7/lib/esql/checkapi.o
# Statically link in the database libraries
OTHER_LIBS += -lixsql -lixasf -lixgen -lixos -lixgls -lnsl -lsocket -laio -lm
# Path for the libs
OTHER_LDFLAGS += -L$(INFORMIX_HOME)/lib
```

If the application is initially deployed on a single machine and you find it just isn't enough and your hardware just won't scale any more, you can easily add a new machine to your network and move part of your application (such as the database) to the new machine. If the application's users are experiencing unacceptable performance delays (assuming that the application normally has acceptable response times and your network is not at fault) and your machine's configuration has available overhead, you can add additional web server and WebObjects application processes to the server in just a few minutes. So while there's no need to worry about getting the configuration exactly right the first time, the issues noted above are still important. You'd like your initial configuration to at least be CLOSE to the final setup. After all, no one wants to explain to their boss why the hardware estimate for deployment of the application was too low!

Other Considerations For Managing Your Web Site

• Memory Management: If you've developed your application in the Objective-C language, make VERY VERY certain that you have handled memory allocation and deallocation correctly. The second WebObjects survival guide, "WebObjects Memory Management" addressed this topic (see the references at the end of this guide for a link). If you are developing on NT, the ObjectAlloc program distributed with WebObjects is a good tool for checking the memory management of your application.

On HP-UX or Solaris, you can identify memory leaks by watching the memory used over time by your application. If it grows considerably, you should check your application to make sure that you are deallocating sessions appropriately, and that you have retained and released memory adequately.

• Development versus Deployment: Before your application is deployed, you should decide if you need a develop/test/sign-off/release procedure for changes made to the deployed application. If so, do not allow even simple changes to be made to the on-line application without running through a test phase first. The tester should initially be the developer, but an additional testing/sign-off phase is needed from someone unfamiliar with the code. Ideally, the development and deployment machines and databases are kept completely separate. Obviously, a three-user intranet application does not need to implement such a formal procedure. However, if any clients or customers outside your organization have access to the application, I STRONGLY RECOMMEND implementing such a process. A typical process might include the following rules (the exact format of documents and sign-offs will follow your company's normal procedures):

- Implementation of a revision control program
- No changes are made to the application without a formal Engineering Change Order which details the changes to be made, the test plan for approval of the changes, the budgeted hours for making/testing those changes, and the requested completion date for the changes. The developer should be allowed to bid the time required, and management should approve those hours with a signature or an email.
- Once the code update has been completed and verified by the developer, the test plan detailed in the Engineering Change Order should be initiated.
- After successful testing, the code changes should be released into production, using whatever revision control is appropriate for the company.

Conclusion

Deploying a robust, maintainable web site is as important as good code development. The issues listed in this survival guide should not be overlooked or minimized in the deployment of your application. The site should perform well and be accessible to your users in order for your development to truly be a success. Obviously, the answers to the questions posed here are specific to your business needs. Hopefully this guide has provided a comprehensive list for you to consider while making that transition from development to beta testing and deployment.

References

http://gemma.apple.com/techinfo/techdocs/enterprise/WebObjects
WebObjects Developer's Guide
Enterprise Objects Framework Developer's Guide
http://www.omnigroup.com/MailArchive/WebObjects
http://www.omnigroup.com/MailArchive/eof
http://www2.stepwise.com/cgi-bin/WebObjects/Stepwise/Sites
ftp://dev.apple.com/devworld/Interactive_Media_Resources
http://www.apple.com/developer
http://developer.apple.com/media

About the Author

Theresa Ray is a Senior Software Consultant for Tensor Information Systems in Fort Worth, TX (*http://www.tensor.com)*. She has worked as a consultant on WebObjects projects for a wide variety of clients including the U.S. Navy, the United States Postal Service, America Online, and Proctor and Gamble. Her experience spans all versions of WebObjects, from 1.0 to 4.0 beta, several versions of EOF, from 1.1 to 3.0 beta, AppKit, NEXTSTEP 3.1 to OPENSTEP 4.2, Rhapsody for Power Macintosh, and yellow-box for NT. In addition, she is an Apple-certified instructor for WebObjects courses.

Tensor Information Systems is a Apple partner providing systems integration and enterprise solutions to its customers. Tensor's employees are experienced in all Apple technologies including OPENSTEP, NEXTSTEP, Rhapsody, EOF and WebObjects. Tensor also provides Apple-certified training in WebObjects, Oracle consulting and training, as well as systems integration consulting on HP-UX.

You may reach Theresa by e-mail: *theresa@tensor.com*

Interactive Media Resources Include:

- **Interactive Media Guidebooks**
- **Market Research Reports**
- **Survival Guides—Technical "How To" Guides**
- **Comarketing Opportunities**
- **Special Discounts**

Apple Developer Connection

As Apple technologies such as QuickTime, ColorSync, and AppleScript continue to expand Macintosh as the tool of choice for content creators and interactive media authors, the Apple Developer Connection continues its commitment to provide creative professionals with the latest technical and marketing information and tools.

Interactive Media Resources

Whether looking for technical guides from industry experts or for market and industry research reports to help make critical business decisions, you'll find them on the Interactive Media Resources page.

Apple Developer Connection Programs and Products

ADC programs and products offer easy access to technical and business resources for anyone interested in developing for Apple platforms worldwide. Apple offers three levels of program participation serving developer needs .

Membership Programs

Online Program—Developers gain access to the latest technical documentation for Apple technologies as well as business resources and information through the Apple Developer Connection web site.

Select Program—Offers developers the convenience of technical and business information and resources on monthly CDs, provides access to prerelease software, and bundles two technical support incidents.

Premier Program—Meets the needs of developers who desire the most complete suite of products and services from Apple, including eight technical support incidents and discounts on Apple hardware.

Standalone Products

Apple offers many standalone products that allow developers to choose their own level of support from Apple or enhance their Select or Premier Program membership. Choose from the following products and begin enjoying the benefits today.

Developer Connection Mailing—Subscribe to the Apple Developer Connection Mailing for the latest in development tools, system software, and more.

Technical Support—Purchase technical support and work directly with Apple's Worldwide Developer Technical Support engineers.

Apple Developer Connection News—Stay connected to Apple and developer-specific news by subscribing to our free weekly e-mail newsletter, *Apple Developer Connection News*. Each newsletter contains up-to-date information on topics such as Mac OS, Interactive Media, Hardware, Apple News and Comarketing Opportunities.

Macintosh Products Guide

The most complete guide for Macintosh products! Be sure to list your hardware and software products in our *free* online database at http://www.macsoftware.apple.com

http://developer.apple.com/media

The ultimate source for creative professionals.