RAPIDSOFT SYSTEMS

Rapidsoft Systems
redefining business relationships

A Rapidsoft Systems' White Paper

© September, 2008

Commonsense approach to Building Scalable Web Applications

In the previous white paper, we described the differences between Microsoft's .NET framework, and PHP for the web development, and how the choice of one technology over other may affect development and operational cost, scalability and other features of your site. No matter what technology you use the fact remains that at some point, you are are going to face the inevitable problem of growth. Your application, that suddenly looked great, will slow down due to increasing load, your server won't be able to cope with sudden traffic burst and then error pages or overloaded server messages will start popping everywhere.

But that need not be the case, if the team that is responsible for developing your site and creating the suitable architecture for it to grow. Both technologies have been used to build large sites (some will say more with .NET as most commercial large sites use .NET). At Rapidsoft Systems, we have built some very large sites with both technologies – so all the discussion here is technology agnostic.

Understanding Scalability

We all know the importance scalability and would like to build a site as scalable as possible. But defining scalability limits of the system is not what a typical PHP or .NET developer is equipped with. They can code and develop a functional site, but guaranteeing that a site performs to a certain standard requires much more deliberate analysis and measurements. Setting a scalability goal is simple, though achieving it all together a different matter. Scalability is a web portal's ability to maintain a site's availability, reliability, and performance as the amount of simultaneous Web traffic, or load, hitting the Web server increases.

As I told earlier a typical code developer is not well equipped to determine the scalability limits of the code, then how do you really predict or calculate the load that your system is going to support? As you expected, to answer this we need to go out of the code and look entire system as a transaction processing black box. Once we do such an abstraction then there are many tools and methods to measure scalability of the system as a whole. Any telecom and queuing theory expert will tell you that understanding rate of arrival and the time it takes to process a request are two important parameters in

understanding any system's behavior under traffic load. But knowing the theory is one thing, and applying that in real domain is another as I explain that with real life scaling experience.

Last year, we, at Rapidsoft Systems, were involved in the development of Jamnow.com, a social networking site where we stream live music from local and famous bands. It has a fantastic and complicated multi-track audio mixing and real time streaming technology beneath its Web Portal face presented to a typical user. During its beta release, we firsthand learnt the lessons of how well planned and executed plans can quickly go out of window under unpredictable load. One night in the last September, a big New York Rock band was supposed to go live 8:00 PM. Since all the Fans who wanted to listen to band they all arrived at about the same time within a narrow window of 1 to 2 minutes at 8;00 pm. As a result we got nearly 20000 hits within a minute and our infrastructure wasn't designed to deal with sudden burst of load. Though we were able to stream the audio - we had a near meltdown experience on the website where users couldn't login and server just couldn't deal with the traffic. So, the lesson is average load is one thing, but handling the peak load is all together different matter. Not all sites have to deal with such a major variation in demands.

Some of the techniques for scaling that we describe here were applied to the site and we were able to overcome that initial problem of load. We never had the same load issue after that, though that doesn't mean there won't be next hurdle to cross in the future.

Simply, The major issues that affect Web site scalability include:

Monitoring and Managing Site Performance

Performance refers to how efficiently a site responds to browser requests according to defined benchmarks. Application performance can be designed, tuned, and measured. It can also be affected by many complex factors, including application design and construction, database connectivity, network capacity and bandwidth, back office services (such as mail, proxy, and security services), and hardware server resources.

Web application architects and developers must design and code an application with performance in mind. Once the application is built, various administrators can tune performance by setting specific flags and options on the database, the operating system, and often the application itself to achieve peak performance. Following the construction and tuning efforts, quality assurance testers should test and measure an application's performance prior to deployment to establish acceptable quality benchmarks. If all of these efforts are performed well, consequently you are able to better diagnose whether the Web site is operating within established operating parameters when reviewing the statistics generated by Web server monitoring and logging programs.

Depending on the size and complexity of your Web application, you may be able to handle anywhere from 10 to thousands of concurrent users. The number of concurrent connections to your Web server(s) will ultimately have a direct impact on your site's performance. Therefore, your performance objectives must include two dimensions:

- the speed of a single user's transaction
- the amount of performance degradation related to the increasing number of concurrent users on your Web servers

Thus, you must establish desired response benchmarks for your site and then achieve the highest number of concurrent users connected to your site at the desired response rates. By doing so, you will be able to determine a rough number of concurrent users for each Web server and then scale your Web site by adding additional servers.

Once your site runs on multiple Web servers, you will need to monitor and manage the traffic and load across the group of servers. Later in the sections - "Hardware planning" and "Techniques for Creating Scalable and Highly Available Sites" we will discuss ways you can do this.

Linear scalability

Perfect scalability-excluding cache initializations-is linear. Linear scalability, relative to load, means that with fixed resources, performance decreases at a constant rate relative to load increases. Linear scalability, relative to resources, means that with a constant load, performance improves at a constant rate relative to additional resources.

Caching and resource management overhead affect an application server's ability to approach linear scalability. Caching allows processing and resources to be reused, alleviating the need to reprocess pages or reallocate resources. Disregarding other influences, efficient caching can result in superior linear application server scalability.

Resource management becomes more complicated as the quantity of resources increases. The extra overhead for resource management, including resource reuse mechanisms, reduces the ability of application servers to scale linearly relative to constraining resources. For example, when an extra processor is added to a single processor server, the operating system incurs extra overhead in synchronizing threads and resources across processors to provide Symmetric Multi-Processing. Part of the additional processing power that the second processor provides is used by the operating system to manage the additional processor and is not available to help scale the application servers.

It is important to note that application servers can only hope to scale relative to resources when the resource changes affect the constraining resources. For example, adding processor resources to an application server that is constrained by network bandwidth will provide, at best, minor performance improvements. When discussing linear scalability relative to server resources, it is implied that it is relative to the constraining server resources.

Understanding linear scalability in relation to your site's performance is important because it not only affects your application design and construction but also indirectly related concerns, such as capital equipment budgets.

Load management

Load management refers to the method by which simultaneous user requests are distributed and balanced among multiple servers (Web, DBMS, file, and search servers). Effectively balancing load across your servers ensures that they do not become overloaded and eventually unavailable.

There are several different methods that you can use to achieve load management:

- Hardware-based solutions
- Software-based solutions, including round-robin Internet DNS or third-party clustering packages
- Hardware and software combinations

Each option has its own distinct merits.

Most load balancing solutions today manage traffic based on IP packet flow. This approach effectively handles non-application-centric sites. However, to effectively manage Web application traffic, it is important to implement a mechanism that monitors and balances load based on specific Web application load.

Now that you have a fairly good understanding of scalability and availability, the next step is to familiarize yourself with the techniques you can use to achieve scalable and highly available Web sites.

Scaling Your Web 2.0 Applications

Understanding your needs is the first step. Anyone can these days develop a websites – but only a few companies understand the need and process for building scalable web applications. We, at Rapidsoft Systems, suggest the following process to meet your performance goals:

1. First step always is to build a functional website with all the features that you need. Test the site as extensively as possible. It is important to test Web applications with multiple browsers and possibly with different screen size monitors.
2. Use a Web benchmarking and load monitoring tool to monitor and measure the performance of the system. Load monitoring tools should be able to generate realistic web traffic as you would expect in real life.
3. Analyze the results and examine the areas where software optimization can take place. Your website performance measuring tool should be able to give you enough data to understand the behavior of the system.
4. Redesign or optimize your Web page generation code and database transactions code
5. Again to the load and performance testing and re-do the optimization
6. Optimize your web server parameters – caching, timers and any other parameter that may affect the system performance.
7. Monitor network, CPU and memory utilization of all servers and develop a correlation map with the arrival traffic.
8. Add more Web servers and database servers to make sure you can serve the peak load as determined by the above load test.
9. Add Web load balancers to the configuration.

When building large scalable site, one should never forget the need for security and disaster recovery. The database systems fail, disk drives have crashes and systems fail. So, it is imperative that a good design should carefully take all these issues in considerations.

Rapidsoft Systems has built numerous large, scalable web sites and our experts will be able to help you in optimizing your current sites. In all our development work, we always assume a scalable implementation, therefore, minimizing the need for expensive reiterative development cycles. We have

developed in-house tools to monitor and bench mark web portal and database transactions. We use these tools to help our customers build scalable web applications.

For more information and specific questions, please contact us at:

Rapidsoft Systems, Inc,
Mailing Address: 7 Diamond Court, Princeton Junction,
New Jersey 08550, USA

www.rapidsoftsystems.com

Phones: **1-609-439 9060 (US East Coast, NJ Office)**
1-408-829-6284 (US West Coast, San Jose Office)
Toll Free (North America): 1-800-946-5490
Fax: 1-831-855-9743

Email: info@rapidsoftsystems.com