

## How to Improve J2EE Performance and Reliability

Opinion by Kieran Taylor, Akamai Technologies Inc.

### OPINION



KIERAN  
TAYLOR

JULY 15, 2004

(COMPUTERWORLD) – As more companies utilize the Internet in their business, Web applications have come into widespread use. These Web applications are typically delivered via tools such as load balancers, HTTP Web servers, caching servers, messaging systems, transaction-processing monitors, application servers and databases. A typical enterprise application infrastructure is shown below.

As performance and geographic reach requirements expand, it becomes increasingly difficult to scale the Web site infrastructure. IT managers must continually evaluate capacity plans to keep pace with the expected peak demand, and planning must consider events such as marketing promotions, news events, and other events that inevitably create more planning uncertainty. Errors in planning can result in overloads that can crash sites or cause unacceptably slow response times and lead to lost revenue.

Pre-provisioning extra capacity as insurance against overload is financially unacceptable for most enterprises. Ideally, enterprises want the needed resources when – and only when – they are needed; they do not want to buy extra resources that sit idle when they are not needed. [1] "On-demand" computing provides better utilization of computing resources and represents a model in which computing resources are brought into service as needed.

### On-Demand Application Platforms

Today, on-demand computing technologies have been integrated into centralized enterprise application platforms, and generally offer enterprises improved fault tolerance and better scalability, using intelligent scheduling and load balancing of application workloads.

However, for many of today's Web applications, end users are inherently spread across the Internet. Congestion and failures in this end-user environment are common and because of this, centralized enterprise applications can have unpredictable reliability and performance for end users.

One solution for enterprises is to use an on-demand distributed computing (ODDC) model for improved application performance and reliability. Deployed at the "edge" of the Internet – close to users' access points – this gridlike distributed platform consists of hundreds of servers deployed in many networks around the globe.

Enterprises can deploy applications to this distributed platform; thus, enterprises can fight service bottlenecks and failures, and at the same time provide on-demand scalability, global reach and high performance for the application users.

This widely distributed application environment consists of the end user typically using a browser, the enterprise (origin) running business logic, legacy systems and databases, and the edge servers running an embedd-

able server, such as WebSphere Application Server or Tomcat, that supports the Java 2 Enterprise Edition (J2EE) Web application programming model.

### Developing Applications for an ODDC Platform

The development model remains standard J2EE for edge applications and doesn't require the use of any proprietary application programming interfaces; it's the deployment model that changes, not the programming model.

If your applications generally follow J2EE component programming best practices, adapting the existing application for the edge will be easier. Development for an ODDC platform still relies on standard J2EE development tools and best practices in developing applications, but you must architect your edge-enabled application as two cooperating sub-applications: an edge-side application and an enterprise-side application.

### Presentation Components on the Edge

The presentation components are the most common application components to deploy to the edge. These components access enterprise data via the Java Web services client model. Typically, the Web application will be developed using a framework based on the Model-View-Controller (MVC) architecture. Jakarta's Struts framework, an open-source framework for building Web applications based on the MVC pattern, is well suited for edge deployment.

The View and Controller components of a Struts application are good candidates for distributing out onto the edge network. These components execute on the edge servers and can interact with Model components running at the enterprise.

Depending on the functionality of your application, the extent to which these applications can move onto the edge platform will vary. The edge View and Controller components are bundled, along with other Java classes, into a Web application archive (WAR) and deployed onto the edge server network.

Normally in an enterprise environment, the Web tier will use Remote Method Invocation (RMI) to communicate with the business tier. The edge application can also use Java API for XML-based remote procedure call to call the Web service at the enterprise via Simple Object Access Protocol (SOAP)/HTTP(S). The edge application platform enables an edge application to cache the results of SOAP/HTTP(S) requests to optimize interactions with the enterprise. These requests to back-end systems are handled via industry standard protocols such as HTTP, SOAP, Java RMI and Java Database Connectors.

### Data Access on the Edge

Any of these HTTP-based protocols are useful interfaces that allow you to "bridge" your applications from the edge to the enterprise, but it's still important to avoid excessive communication for edge-origin requests, since end-user latency and origin utilization will increase. Since there is an absolute cost for every round trip from the edge to enterprise, calls should be as effective as possible. Requests should be bundled, and edge caching should be used to store data and other objects across requests.

A marketing promotional contest is an example of an application that can run almost entirely on an edge environment. Because of

the uncertainty of the number of end-user contestants, a distributed application delivery platform is extremely beneficial to assuring a successful outcome. In this scenario, the application might have "random selection" logic to determine if an end user is a winner.

An application can be designed and developed to execute this logic on the edge, offloading the load from the enterprise. In addition, the corporate marketing team can implement various controls on how long the contest runs, how many products are given out, the rate at which they are disbursed, or other controls. The edge application executes the corresponding business logic entirely on the edge and retrieves the control parameters from the enterprise via Web services calls.

### Application Deployment to an ODDC Platform

Once designed, developed and tested, the edge sub-application is uploaded and provisioned on the ODDC network. Plug-ins allow developers to deploy their applications directly from popular application deployment environments, such as WebSphere Studio and Eclipse, to the edge network, thus creating a single environment so developers can easily deploy J2EE applications.

The developer codes an application, simply clicks on "deploy," and the infrastructure and capacity of the ODDC platform are instantly available at the developer's fingertips, giving developers the freedom to innovate and launch richer applications, while reducing

the risk associated with up-front capital investments.

### Summary

To support growing numbers of Web-based applications and avoid the Internet bottlenecks inherent with "silo" serving, companies are increasingly utilizing a "distributed computing" model in an on-demand fashion. By locating computing resources close to the users requesting them, performance and reliability, as well as scale, are assured. Given the proliferation of Web services, this distributed model is increasingly the choice of enterprises eager to innovate without risking huge capital or ownership costs. As a result, enterprises that use this proven model needn't expend time or money on complex capacity planning.

For developers, the On-Demand Distributed Computing model boosts performance so that applications are never "dumbed-down" to handle the vagaries of the Internet. The net result is that enterprises can launch innovative Web services-based applications in far less time and at lower cost than possible with traditional solutions.

Note: [1]- A study conducted by IBM showed that on average Unix servers were only 10% utilized in enterprises (IBM Scorpion White Paper: Simplifying the Corporate IT Infrastructure, 2000)

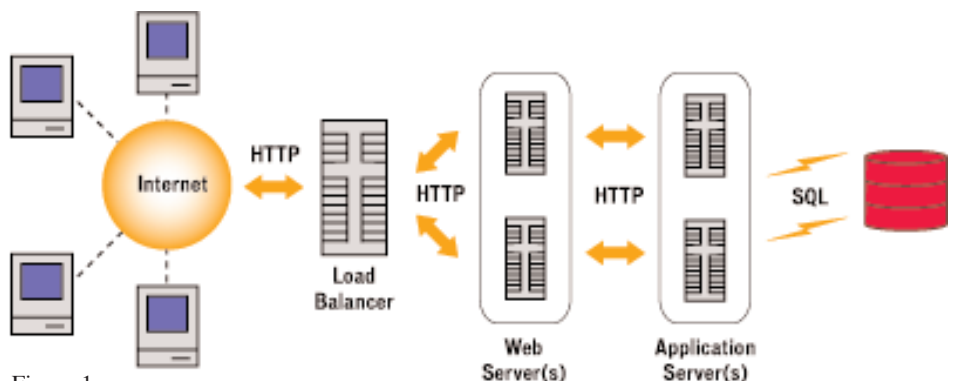


Figure 1

*Kieran Taylor is the director of product management at Akamai Technologies Inc., which provides on-demand computing solutions and services. He can be reached at ktaylor@akamai.com.*

