

Performance Gains to the IBuySpy E-Commerce Application Using ScaleOut™ StateServer

ScaleOut Software, Inc.
June, 2004

Introduction

E-commerce applications need to run seven days a week, twenty-four hours a day, and they need to deliver the responsiveness that their worldwide customers expect. To maintain fast response time in handling client requests, these applications must ensure that session data, such as shopping carts, can be accessed reliably and fast. This can become quite a challenge during busy shopping periods and as Web farms¹ grow to handle increased demand.

ScaleOut StateServer provides a new solution for managing session data that significantly reduces response time while improving availability. Instead of storing fast changing session data in a backend database server, which can become both a bottleneck and a single point of failure, ScaleOut StateServer moves these data into memory (RAM) on the Web or application server farm where they can be rapidly accessed and updated. By intelligently distributing and replicating session data across the farm, ScaleOut StateServer automatically scales its performance and capacity, and it ensures uninterrupted access if a server fails.

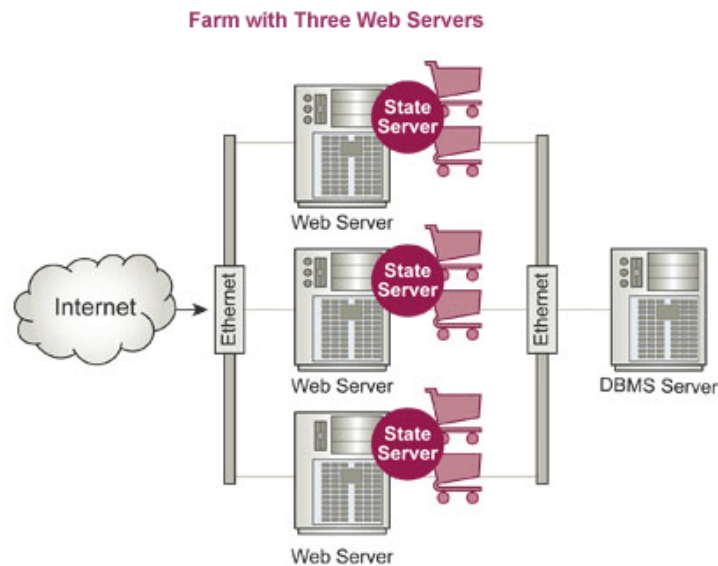
Overview of the IBuySpy Demonstration

The IBuySpy Store is a sample e-commerce Web application selling fictional spy accessories that is made available for download by Microsoft Corporation to demonstrate the capabilities of ASP.NET. (See <http://www.ibuyspy.com/ibuyspy/download.aspx> for more information.) This application illustrates the design of an online shopping site, including the product catalog, user authentication and personalization, shopping carts, and order checkout. By default, IBuySpy stores product and user information, shopping carts, and purchase data in a SQL Server database.

ScaleOut Software has modified the original implementation of IBuySpy to use an in-memory shopping cart that behaves exactly like the original implementation but avoids the database accesses required to store shopping carts in SQL Server. The in-memory shopping cart is stored in an ASP.NET session object (using ASP.NET's "in-process" storage option for session objects). Each session object has a unique session identifier that corresponds to a logged-in user shopping on the site. Note that SQL Server is still used to authenticate users and complete purchases.

¹ A Web farm is a collection of Web servers that cooperate to handle incoming Web requests using a Web load-balancer. Web farms allow a Web site to scale with demand and to survive the failure of a Web server.

While storing shopping carts directly in Web server memory should greatly improve overall response time, this technique alone would require that the Web load balancer direct all requests from a given user to the same server within the Web farm. If a server should fail, its users would lose their session data. The use of ScaleOut StateServer avoids this problem and retains the performance benefits of in-memory storage by transparently replicating ASP.NET session objects across multiple servers and making them uniformly accessible from any server using the unique session identifier. The following diagram illustrates ScaleOut StateServer running on a Web farm:



Once the IBuySpy application was modified to use ASP.NET's built-in session objects to hold shopping carts, ScaleOut StateServer required no additional changes to the application other than ensuring that session objects are serializable. (ASP.NET session objects also must be serializable if they are stored in SQL Server.)

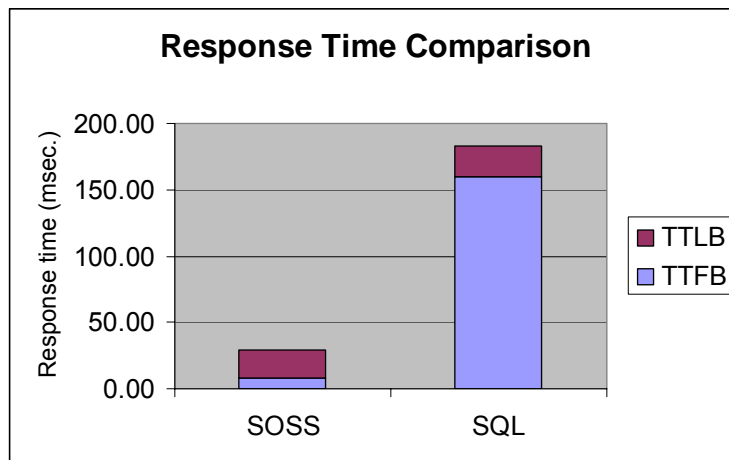
Performance Results

Storing shopping carts in Web server memory using ScaleOut StateServer instead of in a SQL Server database speeds up both accesses and updates to these data. (Note that the use of a database cache does not improve update performance because updates must be committed to the database. Updates also invalidate database cache entries, which can require additional database accesses.) Instrumentation added to IBuySpy showed that six database connections are required to execute the Web page AddToCart.aspx, which adds a selected item to the customer's shopping cart, when using SQL Server to store shopping carts. However, only one database connection is required to execute this page when using ScaleOut StateServer.

To estimate the actual response time improvement, a stress test was performed using Microsoft's Web Application Stress Tool (MCAT). This stress test performed a "typical" shopping session in which a simulated user logs in, selects seven spy accessories, completes a purchase, and then logs out. This single-user test was repeated continuously for ten minutes from a networked client connected to a three-server Web farm which in

turn was connected to a networked SQL Server (as shown in the above diagram except that a single Ethernet LAN was used). Microsoft's Network Load Balancing was used to load-balance Web requests across the Web farm.

The following chart shows the test's average response time to access the AddToCart.aspx Web page. Response times were measured using SQL Server and also using ScaleOut StateServer (SOSS) to hold session objects. These results are shown in separate columns, and each column shows the initial time to receive the first byte (TTFB) and the additional time to receive the last byte (TTLB). The total response time was 6.25 times lower for ScaleOut StateServer, which represents a significant improvement over the use of SQL Server for storing IBuySpy's shopping carts.



Appendix: Details of the Modifications to IBuySpy

Only minimal, straightforward changes to IBuySpy were needed to support the use of in-memory shopping carts. A configuration variable was added to the web.config configuration file to allow switching between the original database version and the new, in-memory version. A new directory was added to the project which contains the four primary C# classes added to the original implementation:

- *Customer*: This object encapsulates all information that is put and pulled from an ASP.NET session object.
- *ShoppingCart*: Exposed as a property of Customer, ShoppingCart is used to store products.
- *ProductCollection*: Exposed as a property of ShoppingCart, ProductCollection is used to add, edit, delete, and show items in the customer's shopping cart.
- *Product*: This object represents an individual product placed in a shopping cart.

ScaleOut Software, Inc.

10900 NE 8th Street, Suite 900, Bellevue, WA 98004
425-450-3216 • www.scaleoutsoftware.com