Atlanta Olympics WOMplex

By Andy Stanford-Clark

This article describes the major components of the Atlanta Olympics WOMplex and attempts to give a flavor of some cutting-edge technology developed for the project.

For the summer of '96, a team of IBM programmers and engineers, based in Southbury, Connecticut, built the official Internet Web site for the Atlanta Olympic Games. The project was immense in every respect—from the amount of content in the site, to the volume of traffic through the site, to the amount of equipment deployed, and the number of hours worked by the team.

WOMplex is an umbrella term that describes the combination of the Web Object Manager (WOM) Internet content and application server, the scalable architecture, the requisite scalable parallel processing hardware, the content authoring environment, the load-balancing systems, the user-level applications on the Web site, and the subsystems that support those applications. In other words, a WOMplex is a scalable, manageable, globally distributable Internet content and application server comprising both hardware and software.

Statistics

The URLs http://www.atlanta.olympic.org and http://results.atlanta.olympic.org together constituted the largest Web server in the world. During the 17 days of the Olympics, the site received just over 192 million hits, with peak daily hits of 16.9 million. Drilling down to the "Sneak Peek" application, which provided a constantly changing montage of captured images from 38 video feeds from the Games in Atlanta, the servers handled up to 21,000 images per hour. What makes these figures especially impressive is that every page served was built dynamically from its component objects at the instant the page was requested.

To provide this level of service 24 hours a day required non-trivial amounts of equipment. The Atlanta site was served by 50 nodes of a distributed IBM RS/6000™ Scalable POWERparallel™ system, spread across five locations in four countries. Most of the processors were at the primary WOMplex site at one of Integrated Systems Solutions Corporation's (ISSC's) major Facilities Management Centers in Southbury. This site provided the required infrastructure for a project of this scale—Uninterruptable Power Supplies(UPS), good physical security, high-bandwidth connectivity into major U.S. Network Access Points (NAPs), and raised floor.

Several nodes—typically some portion of an RS/6000 SP frame—joined the WOMplex at four mirror sites:

- Cornell Theory Center at Ithaca, New York
- IBM U.K. Laboratories at Hursley, U.K.
- University of Karlsruhe near Mainz, Germany
- Keio University near Tokyo, Japan

Each mirror site replicated most of the content of the Olympic site. Notable exceptions were the results system and the rapidly changing Sneak Peek images, which were served only from the primary site at Southbury.

Andy Stanford-Clark

Presented at Get Connected Technical Interchange '96, IBM Hursley U.K., October 1996

System Architecture for Atlanta WOMplex

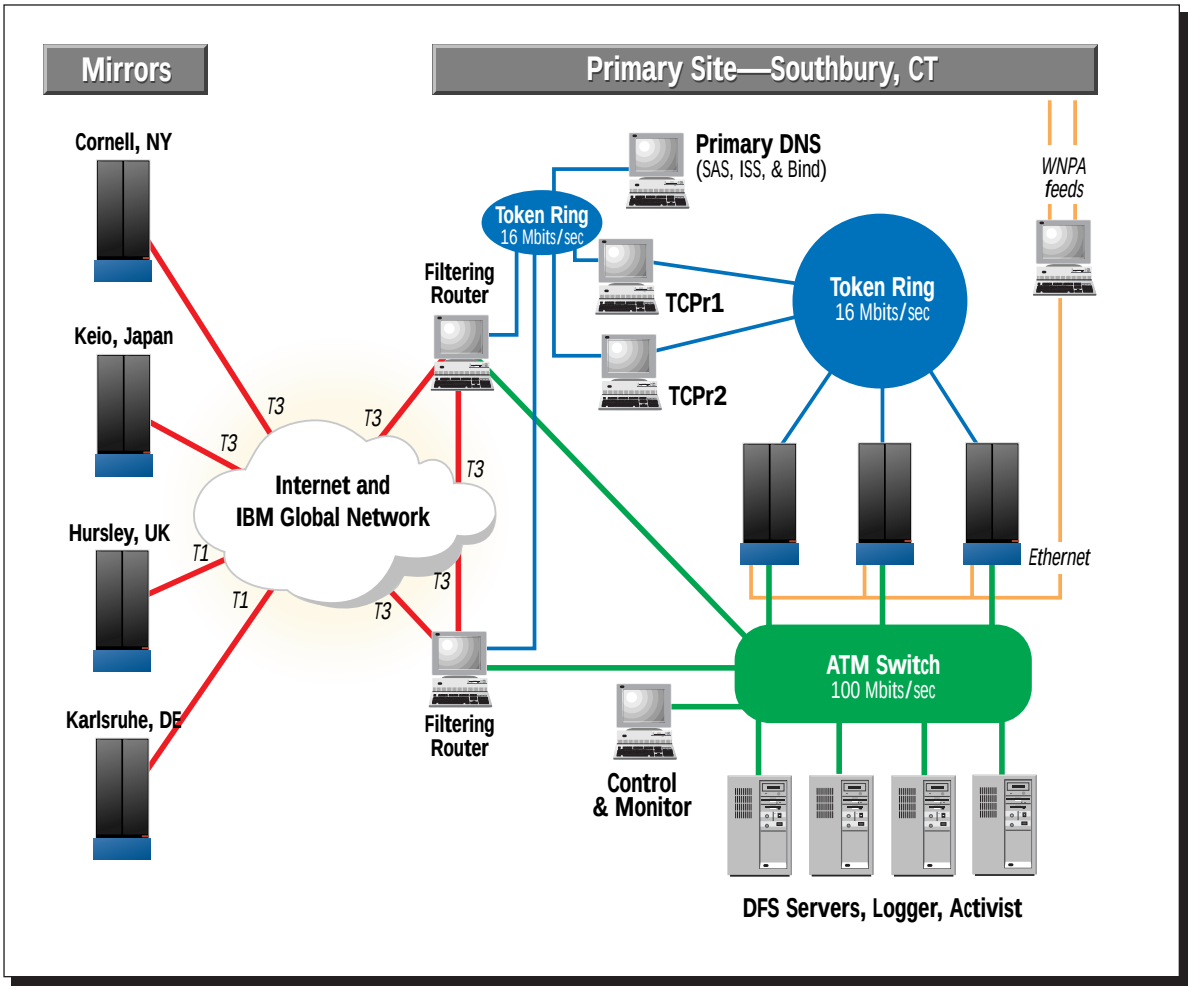

Figure 1. System architecture used for the Olympic games in Atlanta

During the last few months before the games, tickets for seats to events at the games were sold over the Internet by a server based on the IBM Net.Commerce server. In all, this generated over $5 million in ticket sales.

System Architecture

Figure 1 shows the overall system architecture for the Atlanta WOMplex. The "cloud" combines the Internet and the IBM Global Network. T3 connections are DS3 links providing 45 Mbits per second. Filtering routers are Cisco® 7500 Series routers with sets of carefully crafted filter rules to allow only the desired packets in and out from desired sources and destinations. TCPr1 and TCPr2 are the TCP routers, or WOMsprayers—these will be discussed later. Together SAS, ISS, and Bind form the name resolution service. The WNPA feed is a 56 Kbits per second leased-line connection to Atlanta, which provided the feed of results to the databases that were used to drive the Internet Results System. Several leased lines were installed for redundancy. The roles of the machines labeled Logger and Activist are discussed below.

Infrastructure

The IBM RS/6000 SP system provided the WOMplex infrastructure. At the main Southbury site, three frames of the RS/6000 SP were filled with mainly "thin" nodes (equivalent to the RS/6000 Model 390). We did not use the RS/6000 SP high-speed switch in this application—all communication between processors was done over the Asynchronous Transfer Mode (ATM) network.

Control workstations for the main SP frames were in the machine room near the RS/6000 SP. Access to the SP control workstations at the remote mirror sites was handled through X-Windows sessions running across the Internet, using

Secure SHell (SSH) (described below) for secure connections. The Operations Center in Southbury could control every processor in the WOMplex—right down to power cycling.

ATM Network

The outbound network for data from the Web site was routed over a 100 Mbits per second ATM network. ATM uses centralized switching technology to provide a high-bandwidth network. The site networking was highly complex, with multiple Token-Ring, Ethernet™, and ATM networks connected to the Internet through two filtering routers. IBM 8260 switching hubs enabled this level of connectivity. A 16 Mbits per second Token Ring handled the inbound networking. *Note:* Network requirements for a Web site are highly asymmetrical—one packet that contains an HTTP request can provoke the transfer of potentially many megabytes of data from the servers.

Transarc DFS Cell

Transarc's Distributed File System (DFS) ensured that all WOMplex Web servers had access to up-to-date information. (For those familiar with Andrew File System (AFS®), think of DFS as the "grown up" AFS.) DFS is based on the fileset concept, a collection of files handled in the same manner as a single group. Each fileset in a DFS cell resides on a nominated server; it may have mirrors on other nominated servers. Fileset replication happens in two ways: on a timer interval or (often more useful) on a manual "push" from the primary server, when the application or user knows that a batch of updates has been completed.

Each WOM server in the WOMplex was a DFS client configured to request files from the nearest DFS server. At remote mirror locations, the clients retrieved their data from a local server that was kept in sync with the main servers in Southbury. One big advantage of DFS in the Web server environment is client caching. During the Olympics, most clients achieved a 98% cache hit rate.

The primary DFS server in Southbury was distributed across three RS/6000 Model 570 servers and two RS/6000 SP wide nodes. It used 128 GB of IBM 7137 RAID storage, configured with replication and mirroring to give 32 GB of effective storage with three-way redundancy in addition to the RAID failsafe.

The entire cell was administered centrally from Southbury. Problems with filter rules on routers and TCP/IP routing across the Internet make setting up a very large distributed DFS cell a complex process; although when it is running, the system works extremely well. The single DFS cell spanned six locations: Southbury (the main server), Cornell, Keio, Hursley, Karlsruhe (the mirror sites), and Atlanta, where the content was being generated for the Web site.

Internet Connectivity

To ensure the availability of enough bandwidth for the Olympics, four DS3 circuits (T3—45 Mbits per second) were connected into four major U.S. Network Access Points (NAPs): Mae-East, Mae-West, Sprint NJ, and Ameritech™ Chicago. Regardless of the origin of a request, by using a routing algorithm that minimized hop counts, the response was sent to the major NAP closest to the source of the request. Most of the world's major non-U.S. service providers connect directly into one of these four NAPs, so the site was well connected. Multiple circuits also protected the site against any link failure. Packets were automatically routed over other links or could be manually routed around trouble spots, which occurred from time to time at various NAPs.

Security

The entire Atlanta Internet site, the software development environment, and all DFS infrastructure were located outside the IBM firewalls. Consequently, site security was always a high priority for everyone because such a high visibility site was an obvious target for hackers.

High security must be built into a site from the ground up.

High security must be built into a site from the ground up; it cannot be retrofitted onto an existing site. The filtering routers provided the main protection against invasion, and the filter rules for these machines were a major design and maintenance effort. Interworking with the remote mirror sites involved setting filter rules at the remote sites to allow only User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) packets on specific port numbers in and out from specified Internet Protocol (IP) addresses.

A major problem for the primary site at Southbury was the need for extensive access to systems inside the DMZ ("De-Militarized" Zone) at the remote sites; the rest of the world could only have port 80 World Wide Web access. Heterogeneous router technology across the world ensured that many people learned a lot about writing filter rules in a short space of time!

Kerberos and authentication servers were used to secure DFS. Sophisticated password schemes for

the machines were generated and then frequently changed. The apparent inconvenience of complex password schemes was easy to justify in the high profile environment of the Olympics.

Transmission of system control information, including machine root passwords, across the Internet was a potential major exposure. Early adoption of Secure SHell (SSH) for all terminal communication between machines solved this problem. AIX® and Windows® 95 clients for SSH allowed developers to use SSH as a direct replacement for the telnet they would have used otherwise. A version of the SSH client that worked through a SOCKS firewall allowed developers inside the IBM firewall to access the servers and development lab machines.

ISS provided another level of security with the "caching, routing name server," which was deployed as the primary name server for the Olympics site. In this Domain Name Server (DNS) emulation mode, ISS had authority over one or more specified subdomains. If ISS received a name resolution request for a subdomain over which it did not have authority or a name that was not being used for load balancing, then ISS passed the request to one of two backend, ordinary, name servers—typically knowledgeable DNSs that could resolve nearly any request.

The IP address of the original request determines the backend name server that receives the request. ISS configuration has a table of subnet mask specifications. The client IP address, or name server, requesting the name resolution is matched against this table. If a match occurs, the client is considered internal, and passed to one backend DNS; otherwise, it is considered external and sent to the other one.

From the practical standpoint, even with a distributed, multi-site operation, the appropriate client machines can access all of the main site's internal server names and addresses, whereas the rest of the world can only access the limited set of names and addresses (possibly only the Internet address) that the DNS administrator made available. The ISS name server also blocks the DNS Zone Transfer operation, a popular method to access lists of internal machines that are considered hacking targets.

The WOM system has been iteratively developed to support high-volume Web sites for sporting events.

The WOM Web Server

The Atlanta Web site used the latest version of IBM's Web Object Manager system, known as the *WOMserver*. The WOM system has been iteratively developed to support high-volume Web sites for sporting events and IBM external Web sites since 1994. Notable events have included Wimbledon® Tennis, Masters Golf, U.S. Open™ Tennis, and the ACM Chess Challenge (Kasparov versus Deep Blue).

Object-oriented Perl was the language used for the Atlanta server. Supporting applications and subsystems on the site were written in C, Perl, and Java™. Several prototype versions of the WOMserver written entirely in Java were also deployed in the WOMplex for experimental, demonstration, and testing purposes. A specialized binary server, based on the Apache Web server and run on a separate port number, handled binary objects—GIF, JPEG, AVI, MPEG, WAV, Java classes, and Bamba audio and video. All references to binary objects in the HTML source dynamically served as pointers to this port (for example, http://www.atlanta.olympic.org:8080/).

The data flow for entering authored content into the WOMserver had several steps. Content was authored by several types of systems, including traditional HTML editors, image editing tools, and a new generation of WOMtools for directly authoring into the WOM internal object format. This content was then "checked in" to the WOM database. Other systems, based on IBM Digital Libraries technology, generated Web pages from dynamically changing data, such as sports results.

WOMwire was the protocol used for communication among the various tools, including several batch upload tools. This protocol defines the way in which objects and their metadata are transmitted to the WOM system. The WOM database used the DB2/6000™ relational model to store the data and metadata object.

The WOMserver can access data and metadata stored in several formats. For the Atlanta WOMplex, the data and metadata were exported as flat files, called *servables*, from the DB2® database into the DFS filesystem. The DB2 export mechanism also triggered the DFS fileset replication system to push the updated files out to all servers at each WOMplex site.

Web-based applications for the Atlanta site would traditionally have been written as Common Gateway Interface (CGI) programs. However, they were written for the WOMserver in an object-oriented Perl or Java environment. They were run inside the server to avoid the overheads associated with executing CGI applications on traditional Web servers.

WOM architecture allows an executable program to generate any part of a served page;

the applications also fit within this paradigm. Detailed discussion of the WOM Network Application Platform is outside the scope of this article.

The WOMserver is *multi-homed*—it can support several distinct Web sites concurrently in the same server. This makes the WOMplex, with global load-balancing systems, a potential home for many Web sites at the same time—ideal for providing content-hosting services.

Centralized Logging

An important aspect of WOMplex operation is collecting HTTP access and error logs from the servers distributed across the world and bringing them into a central location for monitoring and analysis. The "Grim Reaper" performs this function. The Grim Reaper, a tree-structured plumbing of sockets and UNIX® pipes, streamed log records from each server to a confluence point at each site, then across the Internet to a central collection point on a machine called the *logger* (see Figure 1).

When everything is streaming nicely, the task of the Grim Reaper is straightforward. The problems occur when one of three events occurs:

- The servers produce log data faster than the local Grim Reaper can take it from them.
- Network delays or failures cause holdups in the Grim Reaper plumbing, causing a back-log of data.
- The final receiver of the logs cannot cope with the rate at which they are arriving from the Grim Reaper.

Any of these scenarios, coupled with the simple mandate that no log records can be discarded, make the Grim Reaper's job very difficult. As a result, the Reaper is a sophisticated technology. At any stage, log streams can be diverted into a file on a local disk to hold data, which for some reason, cannot be sent on. Later, when connections are restored or congestion eases, files are spooled back into the network. The Grim Reaper also can deal with requirements such as off-line transfer of batched log data (for example, late at night with less network traffic) and on-the-fly data compression for transmission across slower network connections.

After arriving at the logger machine, the log streams were transferred across a high-speed protocol Common Link Access to Workstation (CLAW) link to an 8-way CMOS 390 Squadron mainframe. The logs were then pre-processed by a series of pipes and filters, then implemented using MVS® Pipes. Individual log records were added to a table in a DB2/MVS database. From here, SQL queries could extract useful information from the logs, including hit counts and rates, online data mining using Activist, and real-time "click tracking." The raw log data was spooled to tape for later analysis.

Content Authoring Environment

The WOMwire protocol defines how authoring and site management tools communicate across the wire with the WOM database. The client part of the WOMwire protocol was implemented as an ActiveX control. The authoring tools, developed for the Windows 95 and Windows NT™ environments in Visual Basic®, C, or C++, could trivially access the WOM database by including the WOMwire control.

WOM objects can contain text, graphics, sound, video, page layout information, or executable code. These objects and their associated metadata can be checked in and out of the system, similar to a programmer's source code control system. The WOMbler created and edited content and managed metadata. WOMexplorer was used to navigate and reorganize a WOMplex site.

WOM objects can contain text, graphics, sound, video, page layout information, or executable code.

Each page on the site fits into an information taxonomy, described by a set of taxonomy tags. These tags position the page somewhere in an *information space*, which has a different hierarchy than the *data space* hierarchy imposed by the HTML links between a set of pages. This exciting new approach to Web site design is outside the scope of this article.

A Visual Basic application called the *Wu-tool classifier* was used to classify existing pages on the site. Users could classify each page into the information taxonomy by choosing from sets of predefined categories that applied to the page in question.

With the WOMpix tool, users could process images by cropping, resizing, reformatting, and labeling, before checking them into the WOM database. WOMpix dealt with many thousands of images that appeared on the site during the Olympics. It also allowed metadata, such as the source agency and photographer, to be recorded with the image. WOMpix could also generate thumb-nail images and image water-marking.

Load Distribution

Load distribution and balancing within the WOMplex uses a two-tier architecture. At each site, local load balancing distributes incoming traffic toward a single IP address across multiple backend WOMservers. Between sites, a global load-balancing system allows logical Web sites to be moved between physical server locations.

Interactive Session Support

Both tiers of the load-balancing system use software called Interactive Session Support (ISS). This is the interactive component of the RS/6000 SP LoadLeveler™ package for batch and interactive load balancing across a set of RS/6000 SP nodes or across other UNIX workstations, notably RS/6000, HP™, Sun®, and Silicon Graphics.

ISSmonitor collates agent responses, starts and stops remote agents (using inetd), and reacts appropriately if agents do not report in for an extended period of time.

Each server runs the ISSagent, which runs a user-defined metric that determines how busy the server is. The metric can be any executable program or pipeline of UNIX commands, providing it returns an integer at the end. For example, a popular choice is `ps -ef | wc -l`, which reports the number of processes currently running on the system and provides a good measure of machine activity. Far more complex metrics can be contrived, including an executable program to generate the metric value. The metric used for the Olympics counted the number of established socket connections on port 80 of each server.

ISSagents periodically report to a central ISSmonitor, which collates agent responses, starts and stops remote agents (using `inetd`), and reacts appropriately if agents do not report in for an extended period of time. The ISSmonitor also uses Internet Control Message Protocol (ICMP) ping packets to ensure that servers are still "alive" before recommending them for use. The ISSmonitor transfers a *load profile* of server activity to the WOMsprayer via a socket. The WOMsprayer uses the load profile to compile a weighted round-robin schedule for incoming traffic. Any server whose ISSagent does not respond is eliminated immediately from the load-balancing schedule to prevent incoming traffic from hitting a non-responsive server and receiving `Connection Refused` error message.

In another operational mode, ISS works as a load-balancing Domain Name Server (DNS). A generic Internet name, such as www.atlanta.olympic.org, is assigned to the cluster of servers. When a client machine or another name server in the hierarchical DNS system requests the IP address for that name, ISS ensures that the name server returns the IP address of the least heavily loaded server to provide the required service. ISS can work in DNS mode in one of two ways.

ISS can cooperate with an existing DNS, by regularly updating the "name to address mapping" database of the name server. Although this works well for low volume, long-lived connections, it is not responsive enough for high-volume, short-lived transactions usually experienced by a Web server.

ISS also operates as a high-performance, load-sensitive DNS, totally replacing the existing name server. This mode of operation can handle Web server traffic and provide a weighted round-robin mechanism for distributing high-volume load evenly across the backend servers.

ISS also offers an enhanced security, caching name server facility, described in the Security section above.

Local Load Balancing—WOMsprayer

Load-balancing software technology from IBM Watson Research has several names: TCP router, Encapsulated Cluster, WOMsprayer, ShockAbsorber, and Network Dispatcher. The *WOMsprayer*, a network router between two physical networks, offers a single IP address to the world. When a TCP connection is made to that address on the inbound network, the WOMsprayer software modifies the Medium Access Control (MAC) address field of the incoming connection request packet and transfers the packet to its outbound network. The MAC address change redefines the final packet destination, and the WOMsprayer arranges for it to be delivered to one of several backend servers.

The backend machines have unique IP addresses and also have the IP address of the WOMsprayer aliased onto their inbound network interface. Consequently, when this modified connection request packet arrives at a backend machine, it appears that the packet was destined for this server; the server does the required "accept" to create the TCP session. The WOMsprayer maintains a table of currently active connections, so any further incoming packets for that TCP session, such as packet acknowledgments, are routed to the correct backend server.

A weighted round-robin schedule determines which backend server receives an incoming session request. This schedule is constructed from the server activity profile that the ISSmonitor periodically supplies to the WOMsprayer. If ISS has not heard from a server or has reason to believe that a server is having problems, the

server is quickly excluded from the WOMsprayer schedule. No new connections are routed to that server until it rejoins the cluster.

The backend server's default route serves as the network path from the server to the client browser. This route generally does not use the same machine as the WOMsprayer. Even if the same server is used, the WOMsprayer does not intercept the returning packets.

Global Load Balancing

Two mechanisms allow Web traffic to be spread effectively across the multiple sites comprising the WOMplex: ISS-DNS and the WOMbat.

ISS-DNS

In its DNS emulation mode, ISS allows the DNS administrator to configure a table of Internet domain names to IP address mappings. This differs from an ordinary DNS in that more than one IP address may be associated with any given Internet name. An ISSagent runs on each server that is designated to serve any given name. According to the health, load, and the configured balancing policy for that name, incoming resolution requests for that name receive the IP address of the most appropriate server.

In the WOMplex, the ISSagents run on the machine that hosts the WOMsprayer for each site. Two balancing policies can be configured: Round and Fail.

Round. This policy does a weighted round-robin of the IP addresses in the list associated with that name. It is distributed according to two factors:

- The server activity profile, as determined from the input provided by the ISSagents
- A server weighting factor, allowing for different server capacity at each site—different numbers of WOMserver nodes in the RS/6000 SPs at each site, in the case of the WOMplex

Fail. This attempts to use the first IP address in the list—the "preferred" site—until that site becomes unavailable. Then, ISS switches to a weighted round-robin of the remaining IP addresses in the list, the same as in the Round policy. This system allows a single, logical Web site (defined by a specific Internet name) to be distributed across multiple physical locations, which could be a cluster of servers controlled by a WOMsprayer. Adjustments to the system configuration can be made dynamically, allowing extra servers to be switched-in, if necessary, to cope with excess load. The system enables automatic fail-over to other servers if a server experiences downtime, which can be fail-over to other servers from a preferred or primary server or omission of unresponsive servers from the round-robin schedule.

WOMbat

WOMbat uses ping to "echo-locate" a client machine and determine the nearest WOMplex site in network round-trip terms. With this knowledge, WOMserver can then divert the client to its nearest server site, and hopefully receive better service, thanks to the better network connectivity to that site.

When a client browser visits a WOMplex site for the first time, the server knows it has not seen that client before, because the client does not have a browser *Cookie* for that site. The server generates a Cookie, a unique key to identify that client in the future, and sends it to the browser to use whenever it visits that site. The Cookie is also stored in the Global Profile Daemon (GPD), a database of profile information, keyed by browser Cookie. The server also invokes— asynchronously in the background—the WOMbat Driver before composing the HTML page requested by the client and returning it to the client.

The WOMbat Driver sends details of the client's IP address and its browser Cookie to a WOMbat daemon at each site in the WOMplex. The WOMbat daemon sends a volley of ICMP echo request packets (ping) to the client IP address and measures the round-trip time of any responses that return within a pre-specified time-out period.

WOMbat daemons send back to the WOMbat Driver the average network response time taken to ping the client machine. The WOMbat Driver waits for a specified length of time for the WOMbat daemons to respond, then determines which server is closest to the client by comparing the responses. Server availability information also can be factored into this calculation, so that the closest server might be rejected in favor of a more distant, but less busy server. The user's profile in the GPD contains the identity of the preferred server.

The next time the user requests a page from the Web server, the server daemon interrogates the GPD using the browser Cookie as a key, and discovers that the preferred server for that client is not this server. The WOMserver then builds the requested page for the client, but rewrites all embedded links to follow-on pages and to

In its DNS emulation mode, ISS associates more than one IP address with any given Internet name.

embedded images as URLs on the preferred server rather than on local links. For example, if the preferred server was www.hursley.atlanta.olympic.org and the local link would normally have been the following:

```
<IMG SRC="/data/image1.gif">
```

then, the link sent back to the client is rewritten as follows:

```
<IMG SRC="http://www.hursley.atlanta.
olympic.org/data/image1.gif">
```

When the client browser follows any of these links, they are retrieved from the preferred server. From that point on, all links will point back to that server, so the client has effectively been "moved" to that server.

WOMplex separates logical services from physical servers, which allows a Web site to be hosted from any site in the WOMplex.

Virtual Addressing

AIX and other UNIX operating systems can alias multiple IP addresses onto a single network interface; that is, the server can "pretend" to have several different IP addresses on the same network at the same time. This was used two ways in the Atlanta WOMplex: multi-homing and service virtualization.

Multi-Homing. A single WOMserver instance can serve several Web sites. The IP site addresses are aliased onto the server's Internet network interface. The WOMserver configuration has separate "trees" of URL data for each hosted address. When a Web page request arrives, the WOMserver interrogates the socket connection to identify which IP address it was sent to, then serves the appropriate page from the correct data tree. Many "standard" Web servers consider this a standard feature, such as Virtual Hosting in Apache.

Service Virtualization. WOMplex separates logical services from physical servers, which allows a Web site to be hosted from any site in the WOMplex. IP aliasing within a site separates logical from physical and frees any dependence on specific hardware. All externally visible IP addresses are aliases, not the "real" base IP address of any machine. Therefore, if a server fails, the service IP addresses supported by the failed server can be transferred readily to another machine. One additional requirement, apart from adding the IP address aliases, is to flush the arp cache on the nearest network router, which forces it to reestablish the new IP to MAC address mappings. The System Network Management Protocol (SNMP) interface to the router can often automate this operation.

Global Profile Daemon

The Global Profile Daemon stores and caches client profile information, using a client browser Cookie as a retrieval key. Any WOM application or the WOMserver itself can store profile information in GPD. Profile data is held as a file of lines containing `name = value` pairs. Keys used for the Atlanta WOMplex were constructed from the concatenation of the following:

- Three randomly chosen letters
- Cookie creation timestamp (seconds since Epoch, plus an offset)
- Cyclic Redundancy Check (CRC) checksum digits

GPD uses a multi-level cache to balance the global distribution requirements of profile data and fast access to user profiles by the WOMserver. Closest to the WOMserver, GPD maintains an in-memory cache with a Least Recently Used (LRU) policy; therefore, people who are known to be currently moving around the Web site are most likely to be in memory.

To locate profiles efficiently, the directory structure has directory names 'A' through 'Z'—three levels deep. For example, the profile with the key ADF1234567123 would be found in the file with the following full path: /dfs/profiles/A/D/F/3217654321FDA.

Note that the name given to the profile file was the reverse of the key string, that is, 3217654321FDA. This is because DFS uses its own hashing algorithm for efficient searching of directory contents—based on the first character of the filename. Consequently, the first character of the filename should be a random character. This avoids having all files in a directory starting with the same first character, and thus all falling into the same hash bucket. We chose three directory levels and 26 entries per level, knowing that DFS caching is optimized for a certain maximum directory content size.

At the next level, DFS caches recently retrieved profiles on the local disk of the server machine. At a site level, recently accessed profiles are stored in a DFS fileset that physically resides on disks at that site. This avoids trans-Atlantic connections for many profiles. Finally, the main DFS server at the primary WOMplex site in Southbury stores the global GPD repository.

When a client comes back to the Web site (at any physical location) with a previously assigned Cookie, the WOMserver requests information on that client's profile from its local (on the same machine) GPD instance. GPD checks the in-memory LRU cache and returns the information if it is there. If not, GPD must fetch the profile from the global repository because that client has not been to this physical site recently.

GPD reads the profile from the global DFS directory and copies the file to a local DFS fileset located physically on the local site. Then GPD sends a UDP broadcast to other servers at its local site informing them that the client with a specified Cookie has come to the site, and may soon request pages from them. Other GPD instances at that site receive the broadcast and go to the local DFS fileset to read the profile information into the memory cache. Thus, only the first GPD at a site has to do the trans-Atlantic journey to get the profile data of a returning client. The other servers (who are likely to receive requests from that client in the near future, thanks to the WOMsprayer) pick it up from a fileset that is local to their physical site.

The CRC digits are built into the Cookie. This makes it more difficult for anyone to tamper with a Cookie while trying to obtain profile information about another user by reverse engineering their Cookie. Cookies with invalid checksums are not accepted by the WOM system.

Applications

This section describes applications that were designed and built for the Atlanta Web site. For various reasons, not all of these went into full production on the site. They were all working in the lab, so we have included them here for the sake of completeness.

Sneak Peek

Sneak Peek was the largest "Net Cam" ever. Live video feeds from 38 locations at the Atlanta Games were fed to "The Fishbowl" in Atlanta. Here they went through a bank of "frame grabbers" and were checked into the WOM system. Images were captured approximately every five seconds from each feed. Because of the high volatility of this data, the images were not stored into the WOM DB2 database, but went directly into DFS. HTML pages to render the latest set of images and to navigate between Olympic venues were generated on-the-fly to pick up the most recent image file names.

The human operator in the Fishbowl received all feeds concurrently and could select channels that currently had activity on them. The navigation options reflected the choices presented by the dynamic HTML pages in the Sneak Peek section of the site. The operator could choose the most stunning images and build a daily "scrapbook" of images worth keeping. These images were titled, cropped, assigned metadata using the WOMpix tool, and checked into the main WOM database.

Sneak Peek images were not mirrored across the world because of their highly volatile nature—they were all served from the primary WOMplex site at Southbury. This became an extremely popular application, and many hundreds of thousands of images were downloaded.

Bamba

Bamba is IBM's streaming audio and video technology for the Web based on G.723 low bit-rate streaming standards. For more information about the technology and to download the decoders and encoders, visit the Bamba home page on the IBM AlphaWorks™ Web site (http://www.alphaworks.ibm.com/).

Bamba has three advantages over other Internet audio and video formats:

- Data is streamed, so you can view and download it simultaneously, avoiding huge waits while the whole file downloads.
- The content is encoded at a low bit rate, making it suitable for transfer over a standard 28.8 Kbit modem.
- G.723 is a standard.

Three forms of Bamba content were used at the Atlanta Olympics:

- **Audio clips:** Recordings of many interviews with athletes, Olympics organizers, and news reports. A regular Web server handles Bamba audio clips in the same way as GIF images. On the Atlanta site, they were one data type served by the binary server.
- **Video clips:** Encoded highlights of the sports events. Bamba video clips are also regular binary content that can be served by an ordinary Web server. They were served from the binary server during the Olympics.
- **Live audio:** Atlanta local radio station, WGST Atlanta, was broadcast over the Internet 24 hours a day during the Olympics. Bamba

Sneak Peek images were all served from the primary WOMplex site at Southbury.

Reflector allowed many people to listen to the radio station concurrently.

Bamba Reflector

In typical use, a small Pentium™ PC would do real-time Bamba audio encoding. It is not realistic to have hundreds or thousands of clients connecting to this machine to listen to the audio stream, so Bamba Reflector was created. The Reflector connects to the transmitter as the only client to that machine, it then accepts connections from Bamba clients. Whatever the Reflector receives from the transmitter is transferred to the clients. There must be a certain amount of buffering and flow-control to deal with clients connected over different speed links. The Reflector intelligently drops Bamba packets without losing stream synchronization for any clients who fall too far behind in receiving Bamba packets. Each Reflector can support many hundreds of client connections concurrently. One Reflector can act as a source to other reflectors, so a whole world-wide tree of Reflectors can potentially serve many thousands of clients across the globe.

The Reflector configuration can limit connections to a specified upper limit or it can be offline—not broadcasting at the moment. When a connection is not possible, it can send out an information message in one of three forms: a pre-recorded audio message built into the client, a text message sent from the Reflector, or an audio message sent from the Reflector. Various combinations are also possible.

ISS load balancing can be used in front of a bank of Bamba Reflectors to balance incoming connection requests to a single advertised Internet name across the available servers to ensure an even load across the Reflectors.

WOMplex Boiler Room

The Boiler Room monitoring system allowed statistics to be gathered from various parts of the WOMplex, its applications, and subsystems. The system structure is described below.

Each application to be monitored had a `funneld` (`funnel` daemon) client. To make this a simple task, developers had client code for Java, Perl, and C available. The protocol used throughout this system was Internet Relay Chat (IRC), transmitting data in a textual format with each monitored system using a different IRC channel number. The client code connected to a `funneld` instance ran on each server. The `funneld` instances connected to other `funnel` daemons in a tree structure. An IRC server was at the root of the tree of `funnel` daemons.

The Boiler Room monitors were Java applets embedded in HTML pages on the WOMplex Web site. These applets used IRC client classes written in Java that connected back to the IRC server. The monitor applications then tuned in to the IRC channels required to construct their displays and waited for information to arrive. A variety of dials, counters, and indicators were implemented as Java classes that could be combined in various ways to display the statistics for any system.

Since the Boiler Room system could also tail log files and receive data through sockets and Interprocess Communications (IPC) message queues, it was easy to hook up monitors for almost anything. A special *chart recorder* client logged IRC channel data for subsequent analysis.

Information Taxonomy

Early in this project we realized that a very large site needs some kind of information taxonomy, or hierarchy, mapped to the site; otherwise, it becomes completely unmanageable and almost impossible to navigate. This important issue required the skills of an information architect.

The metadata for every page, image, and multimedia clip checked into the WOM database included a set of category tags. These tags position the object in the *information space* of the Web site. This information space is dimensionally exclusive from the *data space* imposed by the HTML hyperlinks built into the documents. Tools, such as the Wu-tool, categorize pages on the site. Once the whole site has been categorized, facilities can be provided to navigate the site in information space, as well as the traditional HTML navigation. For example, someone looking at pages about Men's Canoeing might also be interested in other pages that are "close" in the information space. For example, in this case, Kayak events might be of interest.

As an interesting new area for Web site navigation and management, this will undoubtedly play an important role in the future.

Personal Home Page

The combination of generating dynamic pages in the WOMserver and identifying individual clients from their browser Cookie allowed a personal home page to be created for each visitor to the site.

Users could specify to the WOMserver the specific sports, countries, and athletes of interest to follow as the Games progressed. If they provided

Users could specify to the WOMserver the specific sports, countries, and athletes of interest to follow as the Games progressed.

this information, they could then request their personal home page (/myhome.html) from the WOMserver. The Home Page application would build a personalized page for that user, including highlights of the day for the sports, countries, and athletes of interest. This was coupled with Activist, which could suggest links of interest for the user, based on previous browsing history.

Activist

Activist is an online Web log data mining tool. Activist was a DB2 client application connected to the DB2 server running on the CMOS 390 parallel sysplex. This machine was amassing HTTP log data from all the servers in the WOMplex via the Grim Reaper. By following the click-track of each unique client (based on their browser Cookie), Activist would compile a "persona" for that client. Once this was established, it was possible to ask questions of Activist, such as "Based on my browsing patterns in the site so far, show me some links that I might find of interest that I haven't seen yet," or "Show me some links that I might like to see, based on where others like me have been."

The phrase "like me" in this context is derived from a data mining technique called *segmentation*. Segmentation allows a data set to be partitioned into several clusters of individuals who show a statistically significant similarity in certain areas of the data. In this application, if you had looked at Canoeing and Cycling, and many others had looked at Canoeing, Cycling, and Basketball, then Activist would probably offer you some Basketball links to view.

This powerful technology will prove useful in making sense of the vast amounts of log data accumulated by every Web server in the world.

Country and Sport Pages

As information such as results, news reports, news articles, news wire feeds, and photographs flowed into the site during the Games and checked into the WOM database, each page was categorized into its appropriate position in the site's information taxonomy. Every few hours, a batch job was run to compile a home page for every country participating in the games and for each sport. This home page contained links to everything recently added to the site relating to that country or sport. This provided an easy way to follow the progress of a particular country or sport and to catch up on recent happenings in that area.

Guestbook, Search, Feedback

For the Atlanta site, these standard applications were built in the WOMserver application environment and run as dynamic extensions to the server. The Guestbook and Feedback applications made a client/server connection to a remote machine running DB2 that stored the information submitted by the user. Other applications provided a password-protected Web interface to these databases to allow the information to be extracted and analyzed.

Postcards

The Postcards application allowed users to "send a postcard" of any image on the Atlanta site to someone else on the Internet using E-mail. When a full-size version of an image was requested, usually by clicking on its thumb-nail image, one option on the navigation bar of the page containing the full-size image was to "Send a Postcard."

With this option, the user could specify the recipient's E-mail address, a message to go with the Postcard, and the sender's identity. The recipient would then receive an E-mail message indicating that a "Postcard from Atlanta" was waiting. By connecting to the URL given in the E-mail message, the recipient would go to a page that was generated dynamically, based on a unique identifier that was part of the Postcard URL. The page would contain the selected image, the message from the sender, and the sender's name.

Eye on the Olympics

"Eye on the Olympics," a Java applet, provided a way to "keep an eye" on what was happening at the Olympics, without having to continuously surf around the site. The applet used IRC to connect to a server at the WOMplex site. It presented a set of icons: one for each sporting discipline. The client displayed a moving ticker-tape of news flashes. This ticker-tape had an area that could be reached by clicking on a sports icon, which contained results and news stories relating to that sport.

The server used the metaphor of a television station with several "programs"—one per discipline—each created by an automated producer. The producer's job was to choose pieces of information being fed into the Web site and assemble them in a form for presentation in the Eye on the Olympics. The server sent out a *play list* of current articles, then sent new articles that had been added since the last play list was

Segmentation allows a data set to be partitioned into several clusters of individuals who show a statistically significant similarity in certain areas of the data.

broadcast. The client applet received the play list and modified its display behavior accordingly, adding new items as they were transmitted and deleting old items that were no longer in the play list. The producers relied heavily on the information taxonomy to determine which new items being checked into the site were of interest to their programs.

The client applet used HTTP to retrieve non-text items, such as images, from the main Web site, using a URL sent over IRC. The applet also used HTTP to retrieve a file containing the initial set of current content when the applet was first invoked.

Conclusion

The Atlanta Olympics Web site was the largest Web site in the world and handled an immense amount of traffic during the 1996 Summer Games. The WOMplex technology was pioneered to support large sporting events on the Internet. The Olympics gave the WOMble Team a chance to push forward the technology frontiers, and has paved the way for products using WOM technology in the future.

The WOMserver was designed as an Internet application platform, of which a sporting event server was just an application. The suitability of this architecture as an application environment has been proven in the "trial by fire" experienced during the Olympics.

This experience greatly advanced load balancing and global scalability. The systems deployed for the Olympics showed that a Global WOMplex is a viable concept.

The author gratefully acknowledges the contributions of Sean Martin and David Grossman to this article, and also wishes to thank the Team at the IBM Southbury lab.

Andy Stanford-Clark, *IBM Corporation, 150 Kettletown Road, Southbury, CT 06488. Internet: andysc@vnet.ibm.com. Dr. Stanford-Clark is an Internet advanced technology consultant in IBM's Internet Division. His responsibilities include exploring emerging Internet technologies and working with customers to implement Internet-based projects. Dr. Stanford-Clark is currently in the U.S. on international assignment from IBM Hursley, U.K. He has a BS in Computing and Mathematics and a PhD in Parallel Computing from the University of East Anglia in the U.K.*