A Scalable and Highly Available System for Serving Dynamic Data at Frequently Accessed Web Sites

Jim Challenger, Paul Dantzig, and Arun Iyengar
IBM Research
T. J. Watson Research Center
P. O. Box 704
Yorktown Heights, NY 10598

Abstract:

This paper describes the system and key techniques used for achieving performance and high availability at the official Web site for the 1998 Olympic Winter Games which was one of the most popular Web sites for the duration of the Olympic Games. The Web site utilized thirteen SP2 systems scattered around the globe containing a total of 143 processors. A key feature of the Web site was that the data being presented to clients was constantly changing. Whenever new results were entered into the system, updated Web pages reflecting the changes were made available to the rest of the world within seconds.

One technique we used to serve dynamic data efficiently to clients was to cache dynamic pages so that they only had to be generated once. We developed and implemented a new algorithm we call Data Update Propagation (DUP) which identifies the cached pages that have become stale as a result of changes to underlying data on which the cached pages depend, such as databases. For the Olympic Games Web site, we were able to update stale pages directly in the cache which obviated the need to invalidate them. This allowed us to achieve cache hit rates of close to 100%.

Our system was able to serve pages to clients quickly during the entire Olympic Games even during peak periods. In addition, the site was available 100% of the time. We describe the key features employed by our site for high availability. We also describe how the Web site was structured to provide useful information while requiring clients to examine only a small number of pages.

1. Introduction

This paper describes the system and key techniques used for achieving performance and high availability at the official Web site for the 1998 Olympic Winter Games in Nagano,

SC '98, Orlando, FL
(c) 1998 IEEE 0-89792-984-X/98/0011 $3.50

Japan. The Web site utilized thirteen SP2 systems scattered around the globe containing a total of 143 processors, 78 GB of memory, and 2.4 terabytes of disk space. This amount of hardware was deployed both for performance purposes and for high availability. Performance considerations were paramount not only because the Web site was one of the most popular sites at the time of the Olympic Games but also because the data being presented to clients was constantly changing. Whenever new content was entered into the system, updated Web pages reflecting these changes were made available to the rest of the world within seconds. Clients could thus rely on the Web site to provide the latest results, news, photographs, and other information from the Olympic Games. Our system was able to serve pages to clients quickly during the entire Olympic Games even during peak periods. In addition, the site was available 100% of the time.

One technique which we used to reduce the overhead of generating dynamic pages was to cache them so that they only had to be generated once. A key problem with caching dynamic pages is determining when a cached page has become obsolete. We developed and implemented a new algorithm we call *Data Update Propagation (DUP)* which identifies the cached pages that have become stale as a result of changes to underlying data on which the cached pages depend, such as databases. A stale page can either be invalidated or updated directly in the cache. For the Olympic Games Web site, we were able to update stale pages directly in the cache. This allowed us to achieve cache hit rates of close to 100%. By contrast, an earlier version of our system which did not use DUP achieved cache hit rates of around 80% at the official Web site for the 1996 Olympic Games.

The techniques deployed at the 1998 Olympic Games Web site can be used at other Web sites where it is desirable to provide significant dynamic content. Web site designers often shy away from providing too many dynamic pages because of the overhead that this entails. We have demonstrated that this does not have to be the case. Using our techniques, a single server can serve several hundred dynamic pages per second if the pages are cacheable. There is considerable demand for providing dynamic content efficiently on the Web. Consequently, our technology is being incorporated into IBM products so that it can be used at many Web sites.

The 1998 Olympic Games Web site achieved high availability by using redundant hardware and by serving pages from four different complexes in different geographic locations containing replicated information. If a server failed, requests were automatically routed to other servers. If an entire complex failed, requests could be routed to the other ones. The network contained redundant paths to eliminate single points of failure. It was designed to handle at least two to three times the expected bandwidth in order to accommodate the high volumes of data should portions of the network fail.

The rest of this paper is organized as follows. Section 2 describes the DUP algorithm for

identifying obsolete cached data and keeping caches updated. Section 3 describes the architecture of the Web site and how pages were structured in order to minimize the amount of browsing required to obtain useful information. Section 4 describes the network architecture and techniques used for high availability and load balancing. Section 5 describes the performance of the Web site. Finally, Section 6 contains concluding remarks.

2. Caching Dynamic Pages

Web servers provide two types of pages: static and dynamic. Static pages are served from the file system. By contrast, dynamic pages are created on-the-fly by server programs which execute at the time a request is made. Dynamic pages are essential for situations like the Olympic Games where Web pages are constantly changing. However, dynamic Web pages are often expensive to serve. A static page typically requires 2 to 10 milliseconds of CPU time to generate. By contrast, a dynamic page can consume several orders of magnitude more CPU time to generate [8].

In order to reduce the overhead for generating dynamic pages, our system attempts to cache dynamic pages when they are first created. Therefore, the overhead for creating a dynamic page is only incurred once. Subsequent requests for a cached dynamic page will obtain the page from the cache [6,2]. Server programs generating dynamic pages are responsible for determining which pages get cached. A component known as the *trigger monitor* is responsible for monitoring databases and notifying the cache when changes to the databases occur. Our system deploys multiple caches in order to handle heavy request traffic.

Cached dynamic pages can be served from our system at roughly the same rates as static pages. In order to achieve these rates, the Common Gateway Interface (CGI) [15] for invoking server programs cannot be used because it incurs too much overhead. Instead, an interface such as FastCGI [14], NSAPI [13], ISAPI [11], or ICAPI (by IBM) should be used. Our system used the FastCGI interface for invoking server programs.

When a request for a dynamic page is received, the server program invoked to satisfy the request first determines if the page is cached. If so, the cached page is returned. Otherwise, the program must generate the page in order to satisfy the request. The program must also decide whether or not to cache the newly generated page. A preliminary version of this technology was used at the Web site for the 1996 Atlanta Olympic Games [6,9]. The 1996 Web site achieved hit rates of around 80%. Key innovations gleaned from our experience with the 1996 Olympic Games Web site allowed us to improve cache hit rates to near 100% for the 1998 Olympic Games Web site.

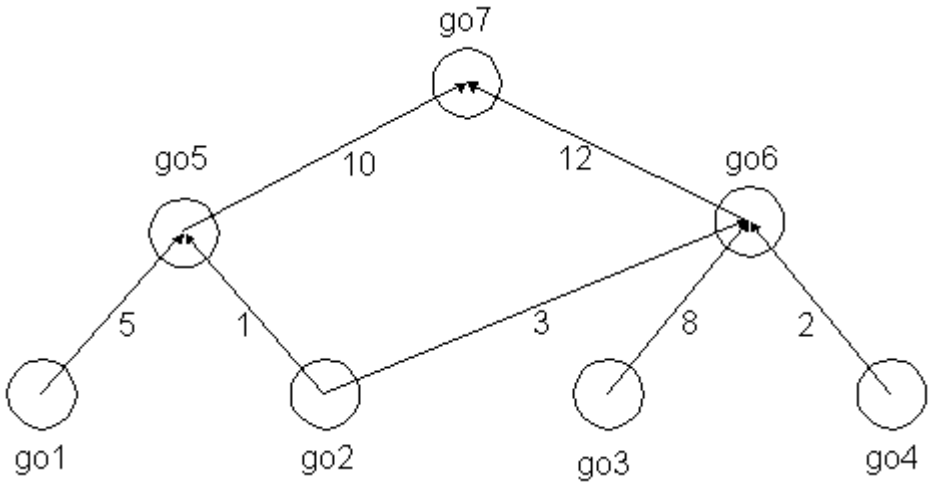
One of the problems with caching Web pages is determining how cached Web pages are affected by changes to underlying data which affect the content of the pages. For the Olympic Games Web sites in both 1996 and 1998, dynamic pages were constructed from databases. When the databases changed due to the arrival of new results, cached pages needed to be updated or invalidated to reflect the new results. A key problem is that identifying which pages are affected by changes to databases is difficult. During the 1996 Olympic Games, a conservative approach was taken whereby a large number of pages were invalidated after database updates. While this preserved cache consistency, significantly more pages were invalidated than were necessary. Consequently, cache miss rates after database updates were high.

At the 1998 Olympic Games, we were able to precisely identify cached objects affected by database changes using the DUP algorithm [7]. DUP maintains correspondences between objects which are defined as items which may be cached and *underlying data* which periodically change and affect the values of objects. It is possible for an item to constitute both an object and underlying data.

An application program is responsible for communicating data dependencies between underlying data and objects to the cache. Such dependencies can be represented by a directed graph known as an *object dependence graph* (ODG), wherein a vertex represents an object or underlying data. An edge from a vertex *v* to another vertex *u* indicates that a change to *v* also affects *u*. Figure 1 shows an example of an ODG. ODG's are constantly changing.

If node *go*2 in Figure 1 changes, the trigger monitor is responsible for detecting the change and communicating the change to the cache. The system determines through graph traversal algorithms which other nodes are affected. In this example, DUP determines that nodes *go*5 and *go*6 also change. By transitivity, *go*7 also changes.

Figure 1: An object dependence graph. Weights are correlated with the importance of data dependencies.



Edges may optionally have weights associated with them which indicate the importance of data dependencies. In Figure 1, the data dependence from *go*1 to *go*5 is more important than the data dependence from *go*2 to *go*5 because the former edge has a weight which is 5 times the weight of the latter edge. Edge weights can be used to quantitatively determine how obsolete an object is. It is often possible to save considerable CPU cycles by allowing pages to remain in the cache which are only slightly obsolete.

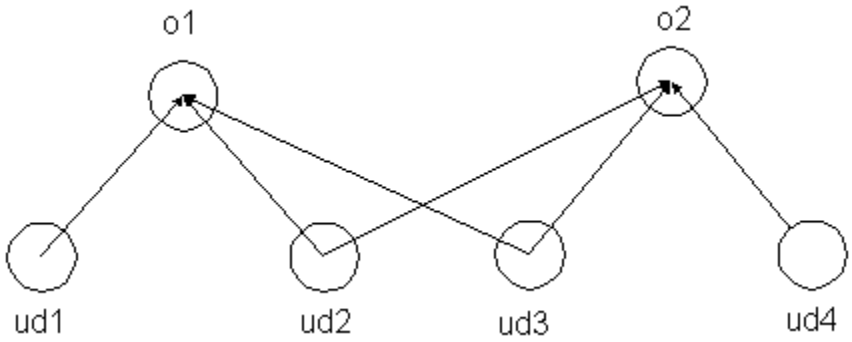
DUP uses graph traversal algorithms in order to determine which cached objects have become obsolete as a result of change to underlying data. These objects are then either updated directly in the cache or invalidated.

In many cases we have encountered, the object dependence graph is a *simple object dependence graph* having the following characteristics:

- Each vertex representing underlying data does not have an incoming edge.
- Each vertex representing an object does not have an outgoing edge.
- None of the edges have weights associated with them.

Figure 2 depicts a simple ODG. DUP is considerably easier to implement if the ODG is simple. A detailed description of how to implement DUP for simple as well as all ODG's is contained in [7].

Figure 2: A simple object dependence graph.



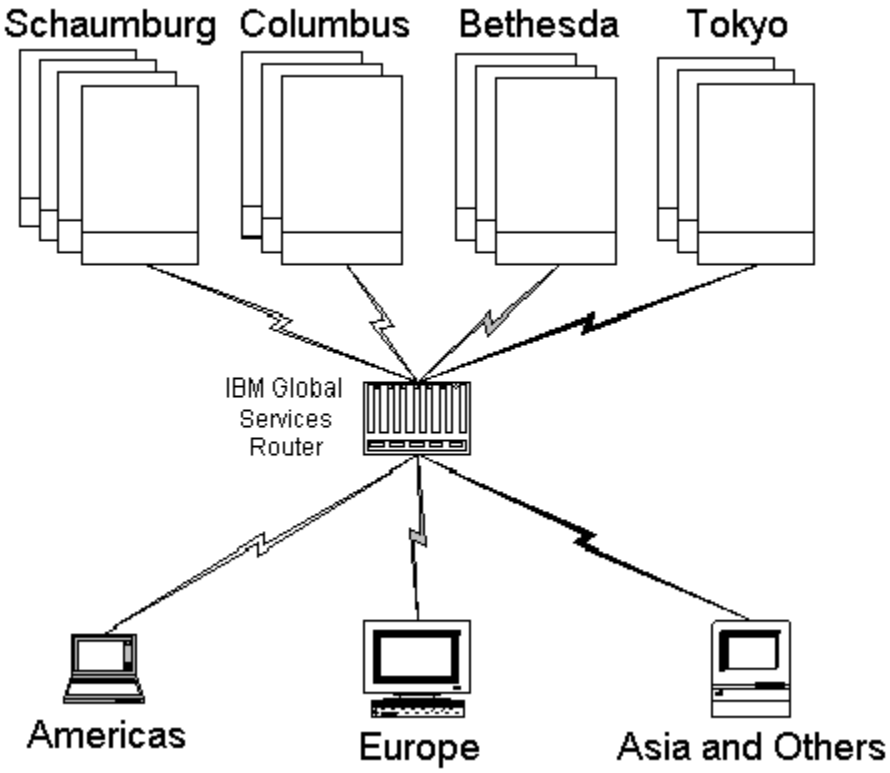
Another key component in achieving near 100% hit rates was prefetching. When hot pages in the cache became obsolete as a result of updates to underlying data, new versions of the pages were updated directly in the cache. Such pages were never invalidated from the cache. Consequently, there were no cache misses for these pages.

Updates to underlying data were performed on different processors from the ones serving pages. As a result, response times were not adversely affected around the times of peak updates. By contrast, processors functioning as Web servers at the 1996 Web site also performed updates to the underlying data. This design combined with high cache miss rates after updates caused response times to be adversely affected around the times peak updates occurred.

3. System Architecture

The Web pages were served from four locations: Schaumburg, Illinois, Columbus, Ohio, Bethesda, Maryland, and Tokyo, Japan. When a client made a request to the Web site http://www.nagano.olympic.org, the request was routed from the local ISP into the IBM Global Services (IGS) network. From here, IGS routers geographically routed the request to the server location closest to the client (Figure 3).

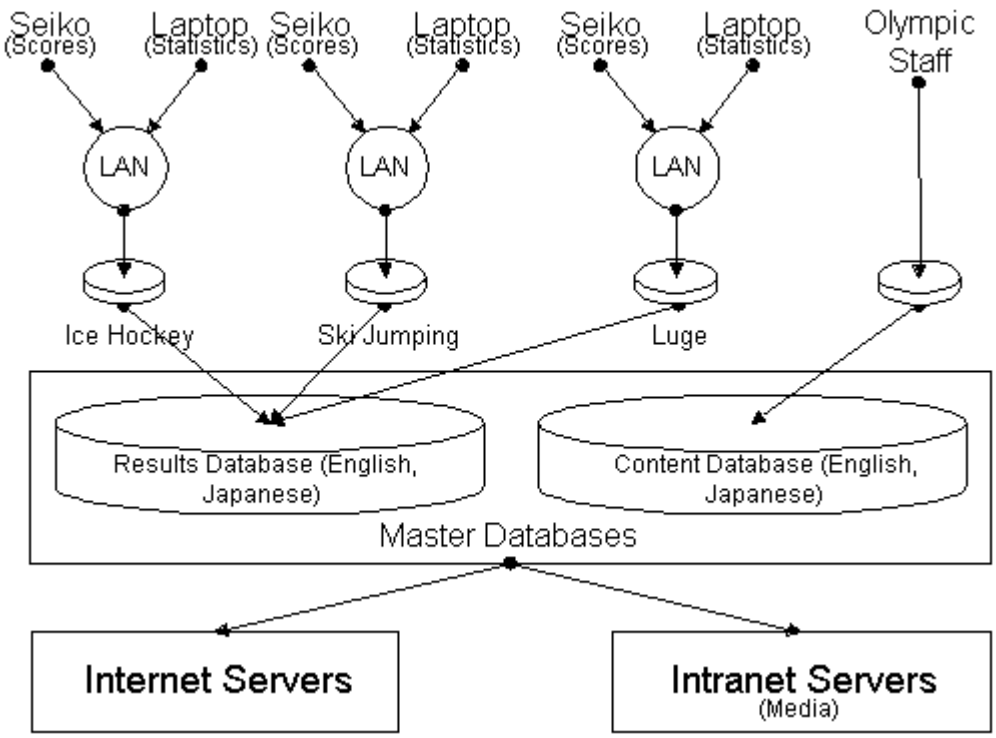
Figure 3: Serving locations of www.nagano.olympic.org



Pages were composed and served from IBM Scalable PowerPARALLEL Systems (SP2) located at each physical site. Each SP2 was composed of ten RISC/6000 uniprocessors, one RISC/6000 8-way symmetric multiprocessor, and associated peripherals. Each uniprocessor had 512 MB of memory and approximately 18 GB of disk space. Each multiprocessor had 1 GB of memory and approximately 6 GB of disk space. In total, the serving systems consisted of 143 processors with 78 GB of memory and 2.4 terabytes of disk space. There were additionally, at each location, a number of machines dedicated to maintenance, support, file serving, networking, routing, and various other functions. A total of thirteen SP2 systems were used: four in Schaumburg and three at each of the other sites. Hardware requirements were calculated from forecasts based on the 1996 Olympic Games site traffic, capacity, and performance, the rate of increase in general Web traffic over the preceding year and a half, and the need for 100% availability of the site for the duration of the games. In retrospect, there was significantly more hardware deployed than was required. This overestimate was partly due to the fact that our caching algorithms reduced server CPU cycles by more than we had expected.

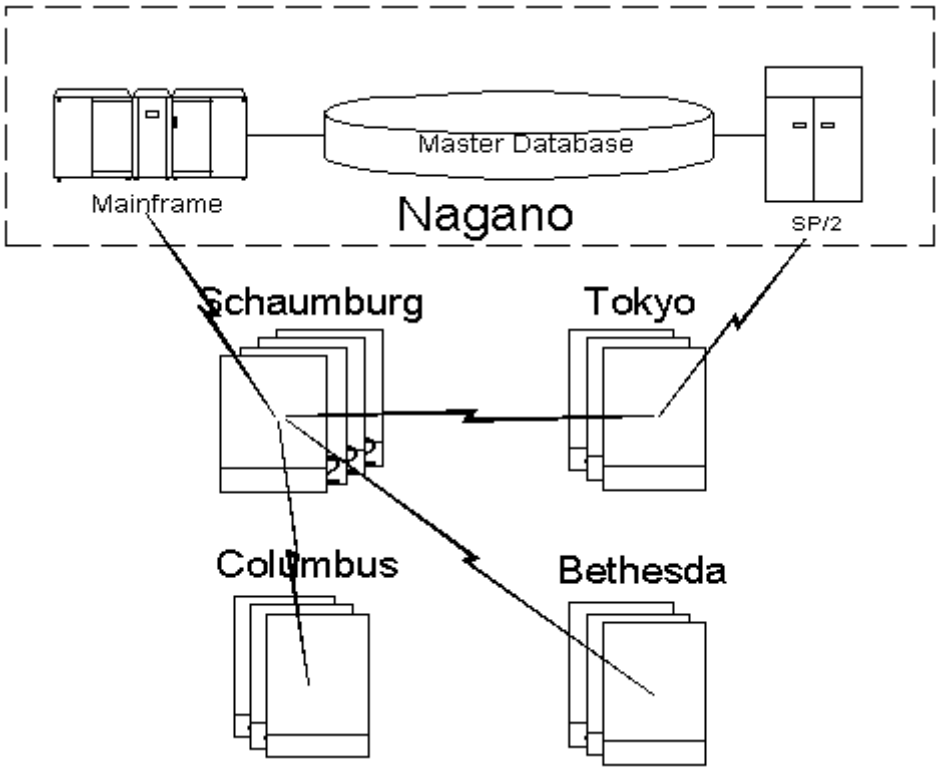
Result data was collected directly from the timing equipment and information keyed into computers at each venue (Figure 4). The scoring equipment was connected via token ring LAN at each venue to a local DB2 database. Each local database transferred its data to the master (mainframe-attached) database physically located in Nagano. This master database is what served both the on-site scoring system and the Internet.

Figure 4: Data flow from Nagano to the Internet and the scoring systems.



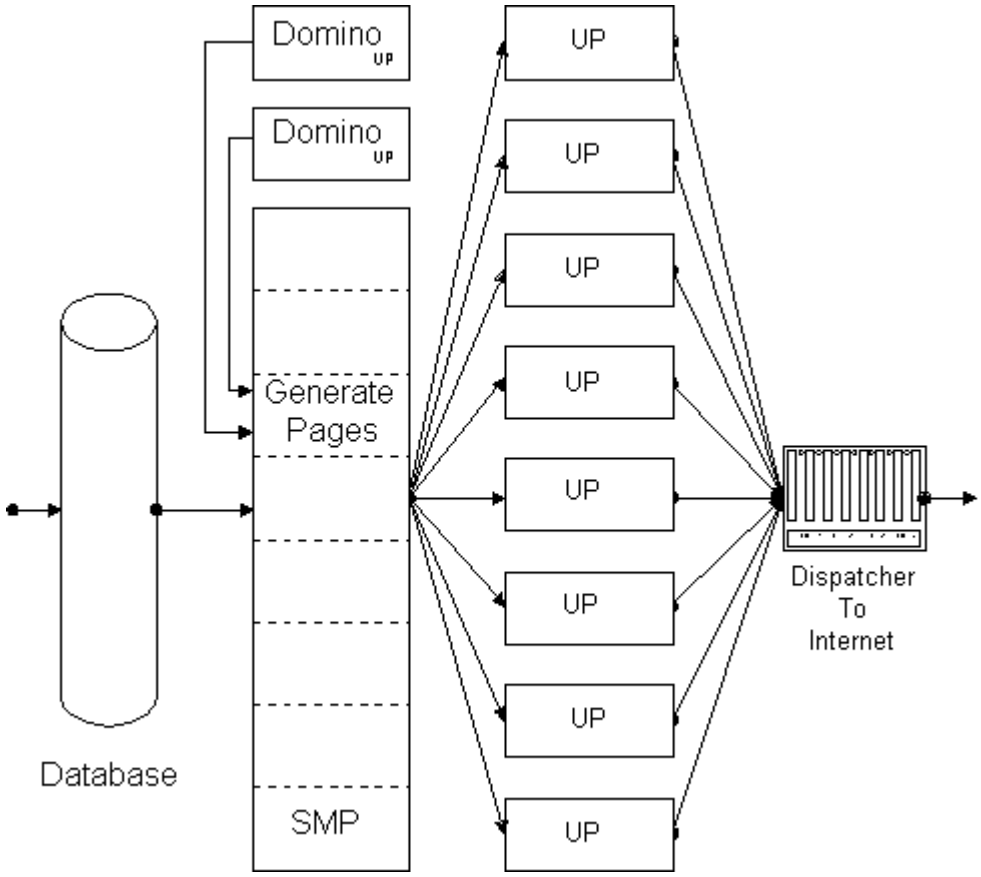
From the master database, data was replicated to the three SP2 complexes in Tokyo and the four complexes in Schaumburg (Figure5). From Schaumburg the data was again replicated to the three machines in Bethesda and the three in Columbus. For reliability and recovery purposes, the Tokyo site was also capable of replicating the database to Schaumburg.

Figure 5: Data flow from the master database to the Internet servers.



Stepping inside a single SP2, we see something like Figure6. Results flowed from the database and editorial content flowed from two of the uniprocessors (UP's) on the SP2 into the triggering, caching, and page rendering code on the SMP. The trigger monitor analyzed the incoming data, issued requests to the local httpd to render relevant pages, and finally distributed updated pages to each of the eight UP's serving the Internet. The UP's were connected to a router distributing requests coming from the IGS routers based on load among the serving nodes. This SP2 configuration was replicated on each of the thirteen serving systems.

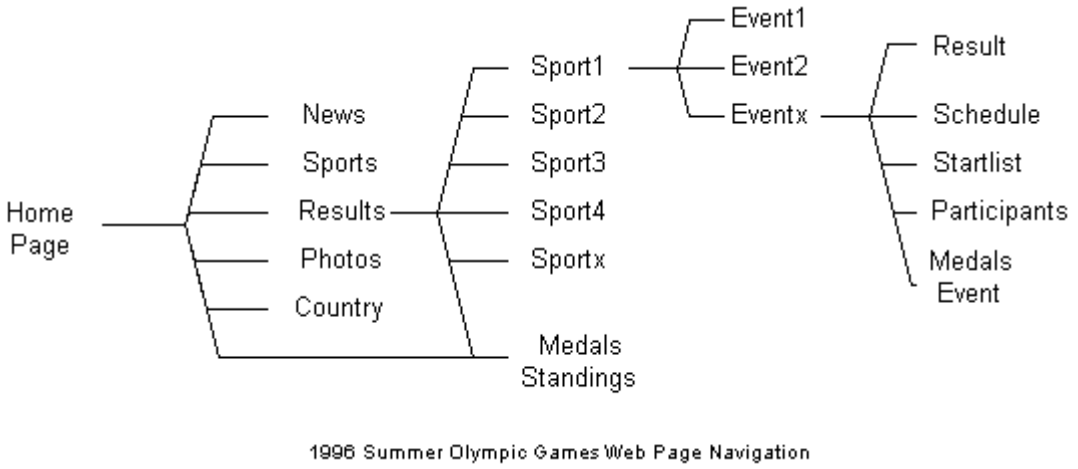
Figure 6: Logical view inside an SP2.



3.1 Web Page Structure and Content

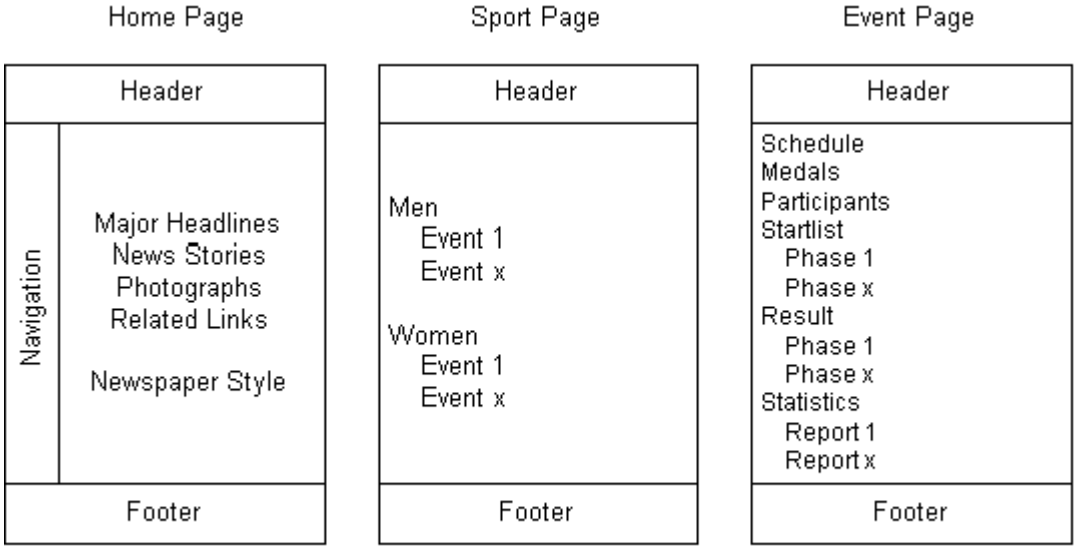
The architecture of the 1998 Olympic Winter Games Web site is an outgrowth of our experience with the 1996 Olympic Summer Games Web site. The Web server logs collected during the 1996 games provided significant insight into the design of the 1998 Web site. From those logs, we determined that most users were spending too much time looking for basic information (e.g. medal standings, most recent results, current news stories). Figure 7 shows the 1996 Web site hierarchy. At least three Web server requests were needed to navigate to a result page. Similar browsing patterns were also required for the news, photos, and sports sections of the Web site. When a client reached a leaf page, there were no direct links to pertinent information in other sections. Due to this hierarchy, intermediate pages required for navigation (see Figure10) were among the most frequently accessed. Figure 8 shows how home, sport, and event pages were designed for the 1996 Olympic Games. Figures 9 and 10 show actual home and results pages respectively.

Figure 7: Page organization for the 1996 Olympic Summer Games Web site.



1996 Summer Olympic Games Web Page Navigation

Figure 8: Page design for the 1996 Olympic Summer Games Web site.



1996 Summer Olympic Games Page Design

Figure 9: Home page for the 1996 Olympic Summer Games Web site. It contains considerably less information than the home page for the 1998 Olympic Winter Games Web site in Figure 13.

(Click here for full-size low resolution image, 115K)

(Click here for full-size high resolution image, 630K)

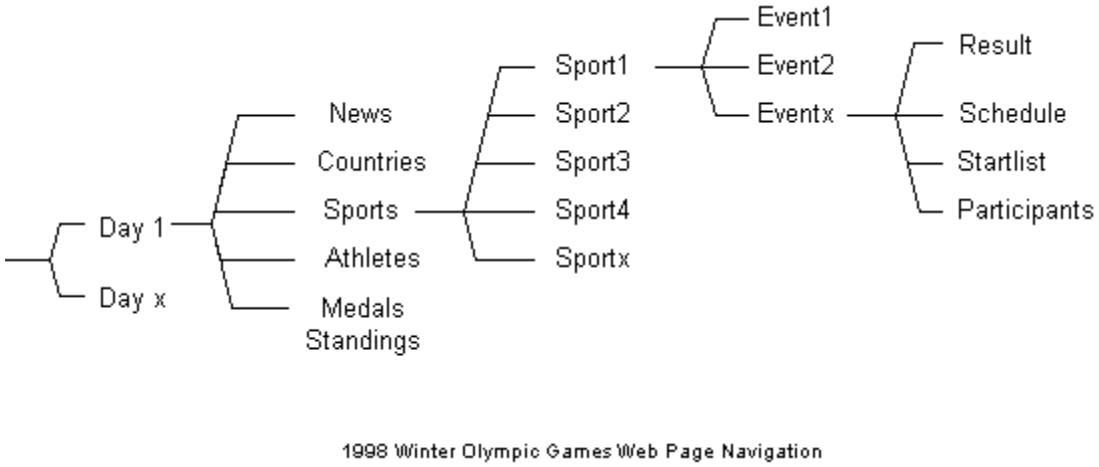
Figure 10: Event page for Women's Tennis Singles at the 1996 Olympic Summer Games Web site. Clients must navigate to other pages in order to obtain useful information. By contrast, virtually every page at the 1998 Web site contained useful information.

(Click here for full-size image, 60K)

The pages at the 1998 Olympic Games Web site were designed to allow clients to access relevant information while examining fewer Web pages. Figure11 shows the 1998 Web

site hierarchy. The most significant change was the addition of another level whereby a different home page was created each day of the Olympic Games. Clients could view the home page for the current day as well as any previous day. The 1996 Web site organized results, news, photos, etc. by sports and events; although country information and athlete biographies were provided, results corresponding to a particular country or athlete could not be collated. Many users of the 1996 Web site felt that this was a limitation. Consequently, the 1998 Web site organized results, news, photos, etc. by countries and athletes as well as by sports and events.

Figure 11: Page organization for the 1998 Olympic Winter Games Web site.



A key objective for the 1998 site was to reduce hits by giving clients the information they wanted on the home page for the current day, such as results from key events currently in progress or recently concluded. Estimates were made that using the page design for the 1996 Web site in conjunction with the additional country and athlete information could result in over 200M hits per day. This figure is over three times the maximum number of hits we received on a single day using our improved page design. Figure12 shows the design of home, sport/event, and country/athlete pages from the 1998 Web site. The logs from the 1998 Olympic Games suggest that over 25% of the users found the information they were looking for by examining the home page for the current day. Figures13 and 14 show actual home and athlete pages respectively.

Figure 12: Page design for the 1998 Olympic Winter Games Web site.

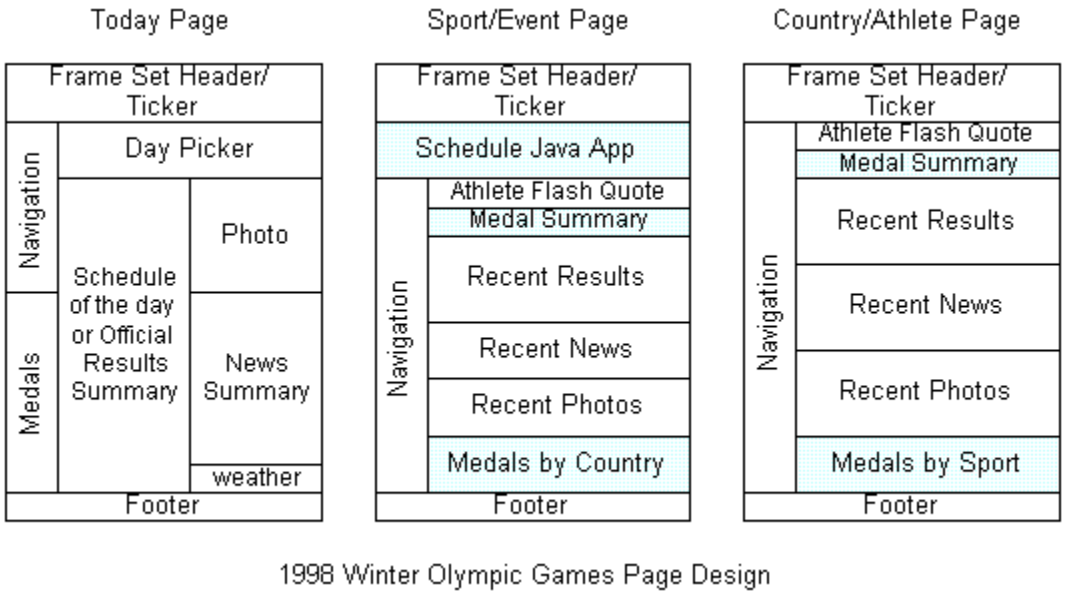


Figure 13: Home page for Day 14 of the 1998 Olympic Winter Games Web site. It contains considerably more information than the home page for the 1996 Olympic Summer Games Web site in Figure 9.

(Click here for full-size low resolution image, 140K)

(Click here for full-size high resolution image, 875K)

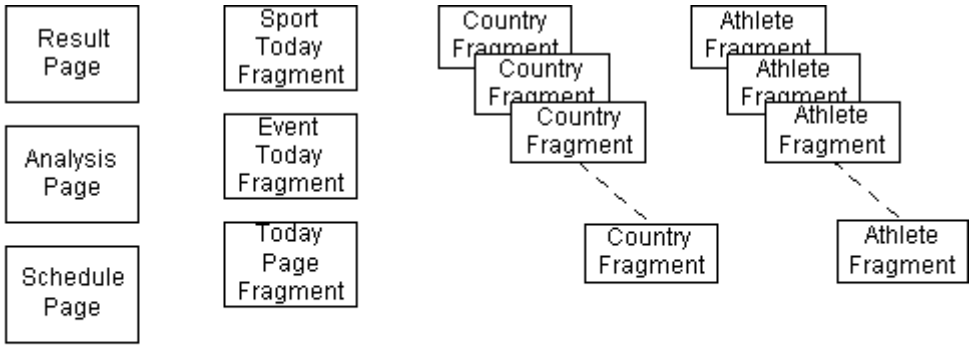
Figure 14: Athlete page for Tara Lipinski at the 1998 site.

Click here for full-size low resolution image (80K)

Click here for full-size high resolution image (446K)

The page redesign led to a significant increase in the number of dynamic Web pages from a few thousand in 1996 to over 20,000 in 1998. In the 1996 Olympic Games, completion of an event typically caused about one or two pages to change. In the 1998 Olympic Games, completion of an event could cause over a hundred pages to change. For example, one typical update to Cross Country Skiing results affected the values of 128 Web pages. Figure 15 shows pages and page fragments which would typically change when new results were received.

Figure 15: Web pages and page fragments typically affected when new results were received by the 1998 site.



The site contained approximately 87,000 unique pages in both English and Japanese; all news articles were also available in French. In addition, the site served a subset of the sport results for the CBS web page. Of these 87,000 pages, approximately 21,000 were dynamically created, reflecting current events within a maximum of sixty seconds after the event was recorded. Up to 58,000 pages were generated or regenerated per day during peak activity, with an average of 20,000 pages generated per day over the course of the event.

The site's content was divided into nine categories:

1. ''Today'': most recent results, stories, medals.
2. ''Welcome'': the ''how to'' and ''what is'' page of the site.
3. ''News'': current and archived new stories.
4. ''Venues'': information on where the sports were performed.
5. ''Sports'': results, scores, etc.
6. ''Countries'': Olympic and general information on participating countries.
7. ''Athletes'': information on the participants.
8. ''Nagano'': information about Nagano, Japan.
9. ''Fun'': sports-related activities for children.

All sports pages were automatically generated in response to database updates. These pages combined items such as scores, medals, and standings with content such as news, photos, and athlete quotes. News and the home page were edited by hand and dynamically combined with results and photos as they arrived. Country information was compiled over the preceding year and dynamically combined with news, photos, and results. Athlete information was extracted from the athlete registration database and dynamically combined with results, news, and photos. Photographs were classified by hand and dynamically inserted into the appropriate News, Results, Athlete, Country, Venue, and Today pages.

4. Network Architecture and High Availability

During the 16 days of the Nagano Olympics, over a million Internet users browsed the Olympic Games Web site. No doubt the average user who accessed the site had no idea of the vast infrastructure that lay behind the scenes, delivering tens of thousands of dynamic Web pages to thousands of users at a time. We wanted the scale of the site to be invisible

to the average user. Our goal in designing the architecture was to ensure that the site continued invisibly working throughout the span of the event to quickly deliver fresh content to users all over the world.

To determine network requirements, we drew on information we had gathered by tracking sporting events on the Web, as well as marketing data projecting Internet growth in general. Based on this information, we projected that the network would have to support up to 100 million hits per day, with a potential peak-to-average ratio of five to one. Given the content we were planning to deliver, we estimated that each hit would request an average of 10 Kbytes of information. This adds up to a maximum of a terabyte of data per day.

One of our initial performance requirements was that a client communicating over a 28.8 Kbps modem should see a maximum response time from the Web site of 30 seconds per page. This requirement was subsequently restricted to IBM Global Network users, since we had little control over the speed with which other Internet Service Providers (ISPs) would be able to route traffic to our network.

4.1 Network Architecture

We needed a network solution that satisfied our requirements for rapid responses and high availability. To guarantee high availability, we provided redundant paths to eliminate single points of failure in the network architecture and made sure there were at least two to three times the needed bandwidth to handle the high volumes of data should portions of the network fail. Load balancing and traffic distribution techniques were also an important part of the overall solution.

A private, high-speed IP network was built to support the movement of content from the source in Nagano to the Web server complexes in Tokyo, Schaumburg, Columbus, and Bethesda (Figures 16, 17). Each of the server complexes had direct connectivity to OpenNet which is part of the IBM Global Network lying outside the corporate firewalls and servicing Internet users around the globe. OpenNet and its network of routers became the foundation for the Nagano Olympics network. In order to provide surplus capacity for Internet users outside OpenNet, additional dedicated lines and line upgrades were established for public access to the Olympic Games Web servers. These dedicated lines provided high-bandwidth connectivity from Olympic host sites to major US Network Access Points (NAPs) which are sites where major networks (e.g. MCI, Sprint, UUNet) connect to each other and exchange data. These lines provided the essential bandwidth for the Olympic Games Network in order to decrease access times for users on other ISPs and ensure that the flood of data didn’t overload and bring down major Internet links.

Figure 16: Network in Japan.

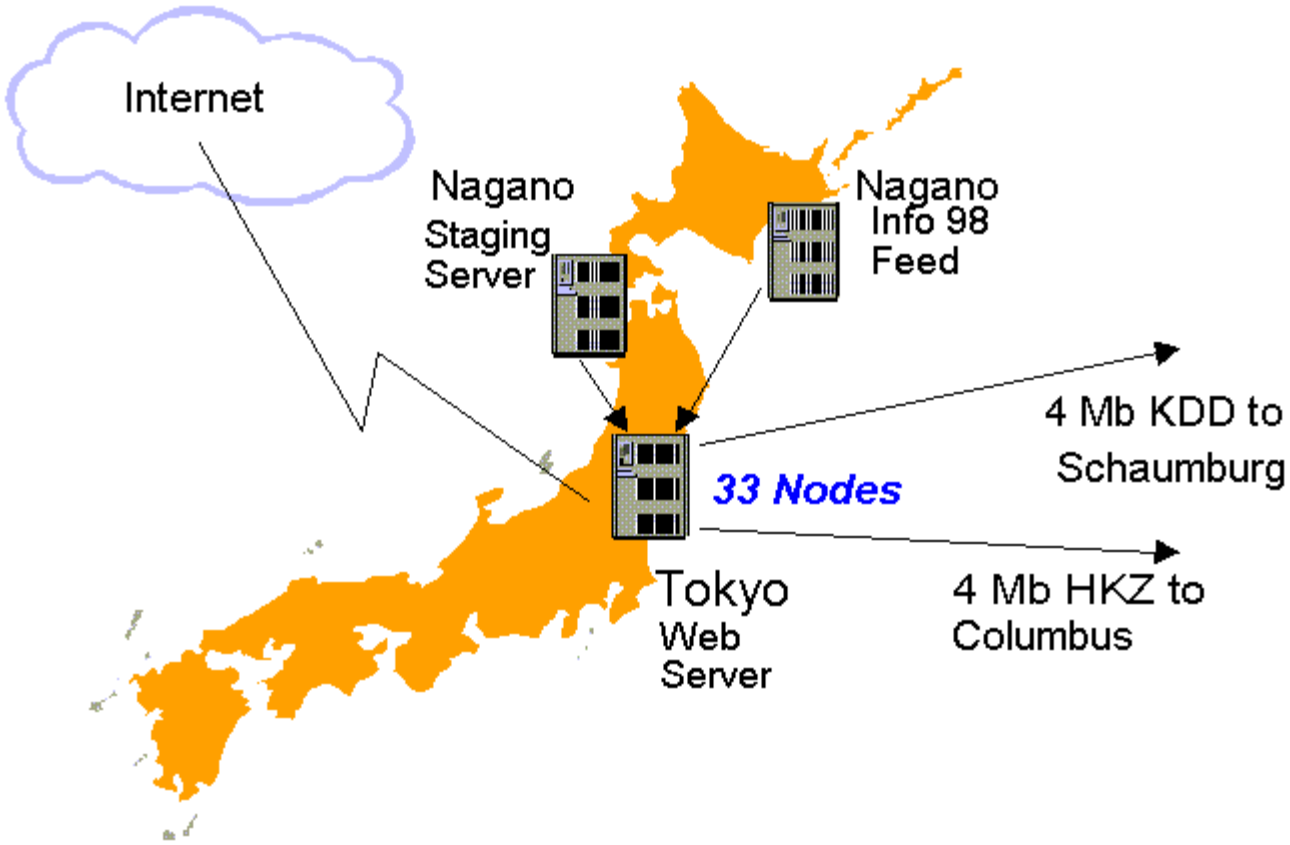
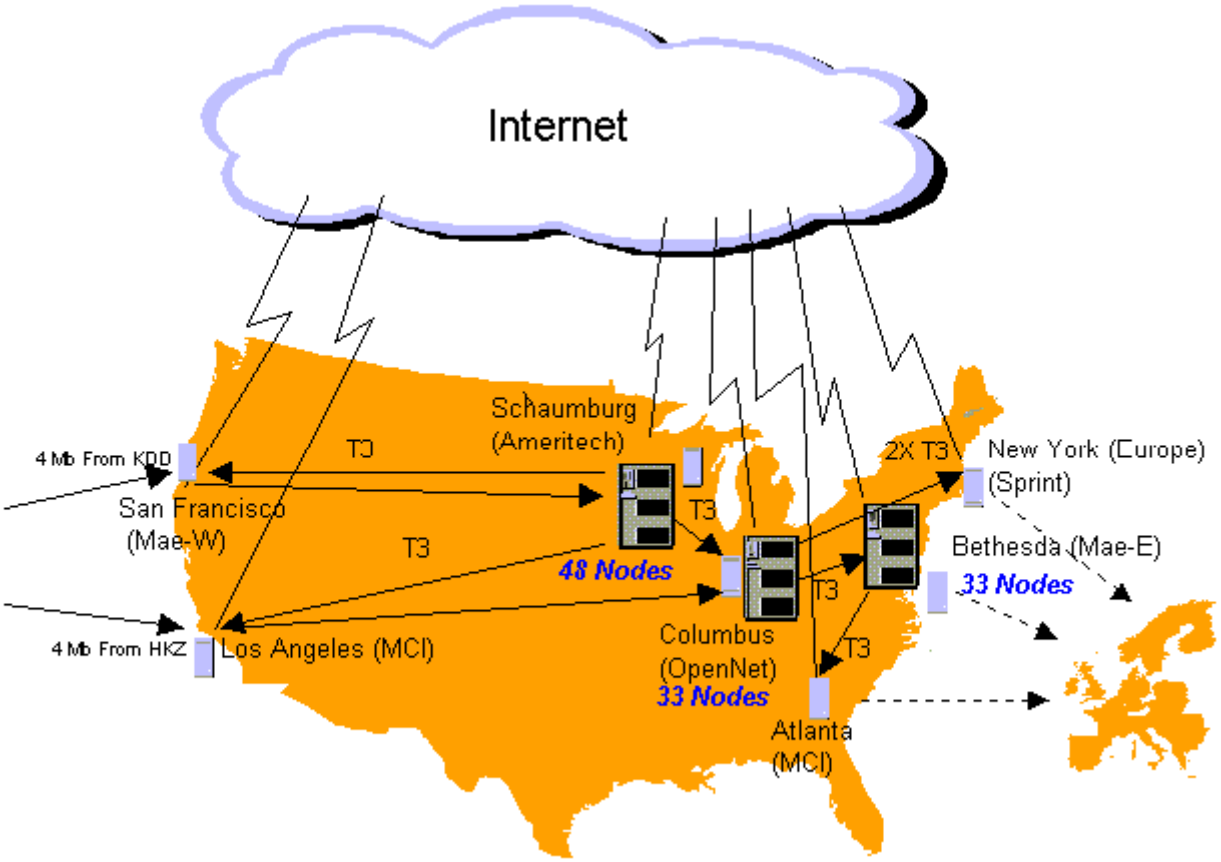
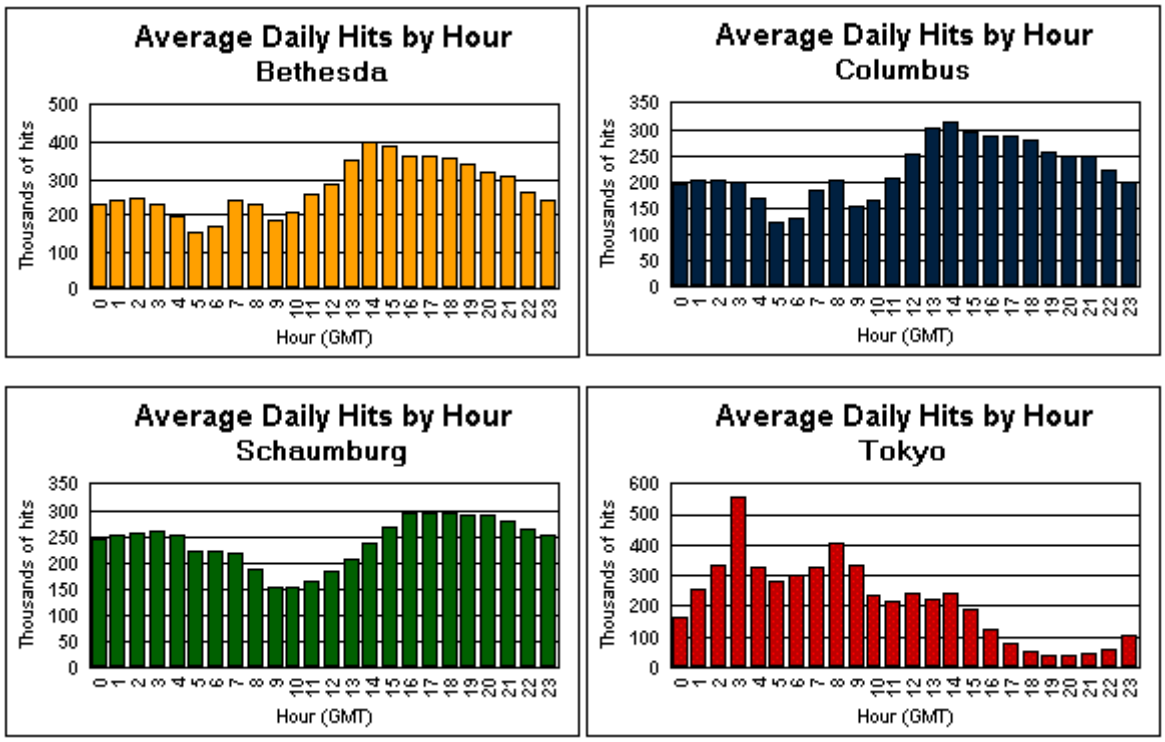


Figure 17: Network in the United States.



Effective load balancing is a necessity with a network expected to carry heavy traffic originating over a wide geographical area. In this case, the geographical area was the entire globe. Furthermore, the flow of traffic could vary dramatically across the entire network. The bar graphs in Figure 18 show the average number of hits by hour over the course of the Olympic Games.

Figure 18: Variations in request traffic over the course of a day. These bar graphs represent averages over the entire Olympic Games.



Since multiple locations were used to host the Web site, one goal was to provide users with the fastest performance possible by routing their requests to the server complex closest to them. However, we needed a way to reroute traffic if a particular site or network link became overloaded. To meet these requirements, we used several common Internet protocols combined in a way that gave us maximum performance, flexibility and reliability. We call the combination of these techniques*Multiple Single IP Routing (MSIRP).*

What makes MSIRP work is the way routing functions on the Internet. IP routing is dynamic by design. Routing can be redirected on the fly to ensure that an IP network will continue to function in cases where failures or changes impact the interconnections between routers and data exchange points. As a result, on the huge interconnected network that makes up the Internet, there are always multiple paths to a single destination. Dynamic routing protocols such as Open Shortest Path First (OSPF) [3] or the Border Gateway Protocol (BGP) [12] determine which route to take by assigning a cost based upon time or other cost metric for each network ‘‘hop’’ traversed between network routers until the destination is reached. After calculating the total cost of each route, data is sent along the least costly path. In network terms, this route is the shortest path to the destination.

For our purposes, it didn’t matter that there were actually different sites corresponding to

the IP address for the Olympic Games Web site. Any router would only see multiple pathways to the ‘‘same’’ destination. When a user’s browser accessed the Web site *http://www.nagano.olympic.org*, the hostname was resolved to the appropriate IP address. Standard IP routing then took over and automatically delivered each request along the path with the lowest cost to the closest site identifying itself as the Olympic Games Web site.

One weakness of single IP routing (SIPR) is that it does not provide flexibility for load balancing. If traffic becomes too heavy at a given site, all one can do is to stop advertising the IP address of the site as corresponding to the requested URL and to route requests for the URL to other sites. This works fine if an entire site ceases to function but is ineffective for balancing traffic among several sites.

To add flexibility, we used multiple SIPR (MSIPR) addresses that all resolved to *http://www.nagano.olympic.org*. Each site advertised all twelve of our MSIPR addresses to the Internet. Since users go to a site by symbolic name rather than IP address, we used round-robin DNS cycling through these addresses. This produced multiple instances of the same route, and traffic arrived at each site evenly distributed among the twelve IP addresses. With all twelve IP addresses to manipulate, we could shift traffic among the sites in 8 1/3% increments.

4.2 Local Load Balancing with Interactive Network Dispatcher

IBM’s Interactive Network Dispatcher [5] was an essential component for implementing the complex network management techniques we used. The purpose of the Network Dispatcher was to accept traffic requests from the Internet that were routed to a complex and to forward the requests to an available Web server for processing. At each complex in the US, four Net Dispatcher servers sat between the routers and the front-end Web servers (Figure 19). Each of these four Net Dispatcher boxes were the primary source of three of the twelve SIPR addresses and secondary source for two other SIPR addresses. Secondary addresses for a particular Net Dispatcher were given a higher OSPF cost.

Figure 19: This figure shows how requests were routed within each of the four complexes serving Web pages.

(Click here for full-size image, 50K)

The Net Dispatchers ran the *gated* [3] routing daemon which was configured to advertise IP addresses as routes to the routers via OSPF. Differing costs were used depending on whether the Net Dispatcher was a primary or secondary server of an IP address. The routers then redistributed these routes into the OpenNet OSPF network. Incoming requests received by any OpenNet router could then make routing decisions, with knowledge of the routes being advertised by each of the complexes, to deliver requests to the Net Dispatcher with the lowest OSPF cost. Typically, this would be the Net Dispatcher at the closest complex which was the primary source for the address assigned to incoming requests; this was given to the browser via round-robin DNS.

The request would only be sent to the secondary Net Dispatcher box for a given address if the primary Net Dispatcher box was down or had failed for some reason. If the secondary Net Dispatcher also failed, traffic would be routed to the primary ND in a differen complex; this is similar to the situation where a system deliberately does not advertise an address in order to move traffic from one complex to another.

A key benefit of this design was that it gave control of the load balancing across the complexes to operators of the Net Dispatchers. No changes were required on the routers to support this design as the routers learned routes from the Net Dispatchers via a dynamic routing protocol.

Each Net Dispatcher box was connected to a pool of front-end Web servers dispersed among the SP/2 frames at each site. Traffic was distributed among these Web servers using the advisors included as part of the Interactive Session Support (ISS) Interactive Network Dispatcher (IND) product. If a Web node went down, the advisors immediately pulled it from the distribution list.

This approach also ensured high availability by avoiding any single point of failure in a site. If a Web server went down, the Net Dispatcher automatically routed requests to the

other servers in its pool. If an SP/2 frame went down, the Net Dispatcher routed requests to the other frames at a site. If a Net Dispatcher box went down, the router sent requests to its backup. If an entire complex failed, traffic was automatically routed to a backup site. In this way we achieved what we call *elegant degradation*, in which various points of failure within a complex were immediately accounted for, and traffic was smoothly redistributed to elements of the system that were still functioning.

5. Olympic Games Web Site Performance

Over the course of the event, 634.7 million requests were serviced. The site was available 100% of the time for the entire Olympic Games. On the peak day (Day 7, Feb 13, 1998), the site served 56.8 million requests over a 24-hour period. By contrast, the 1996 Olympic Games Web site peaked at 17 million hits a day, fewer than any day for the 1998 Olympic Games. The maximum number of hits per minute was 110,414; this occurred around the time of the Women's Figure Skating Free Skating on February 20 (Day 14). The total number of hits to the site as well as the maximum number of hits during a single minute were both determined by independent organizations which audited the Web logs. On July 14, 1998, the *Guinness Book of World Records* recognized the 1998 Olympic Games Web site for setting two world records:

- *The Most Popular Internet Event Ever Recorded* based on the officially audited figure of 634.7 million requests over the 16 days of the Olympic Games.
- *The Most Hits On An Internet Site In One Minute* based on the officially audited figure of 110,414 hits received in a single minute around the time of the Women's Figure Skating Free Skating.

Figures 20, 21, and 22, show summaries of the hits, network traffic, and response times of the site during the event. Figure 23 shows the breakdown of requests by geographic location.

Figure 20: Hits by day in millions.

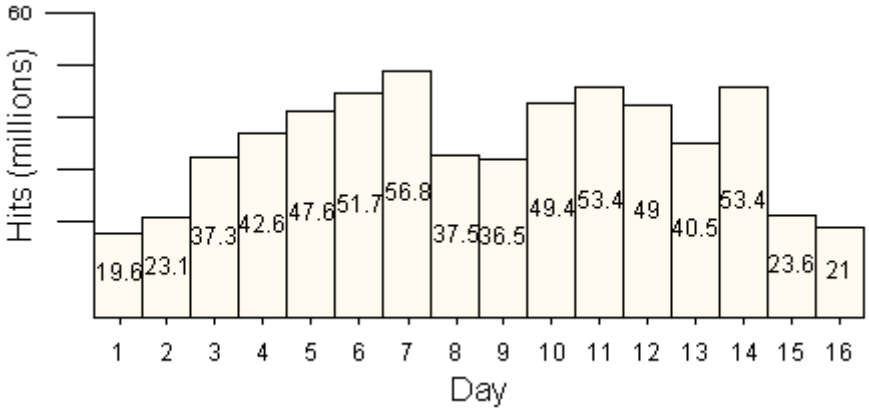


Figure 21: Traffic in billions of bytes.

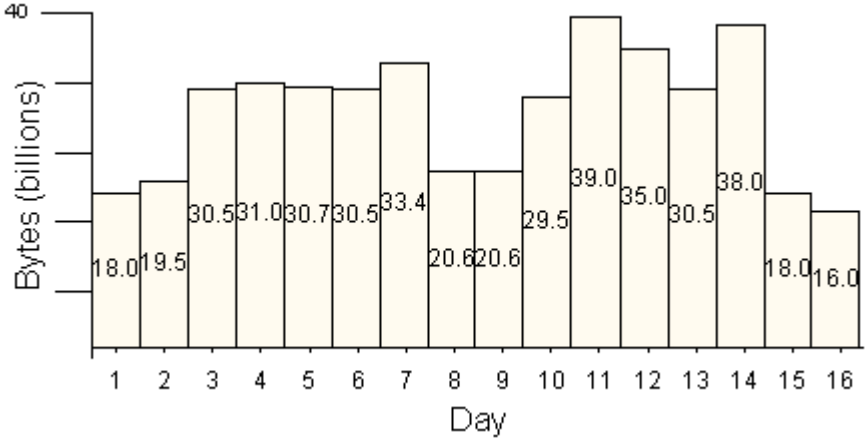


Figure 22: Response times.

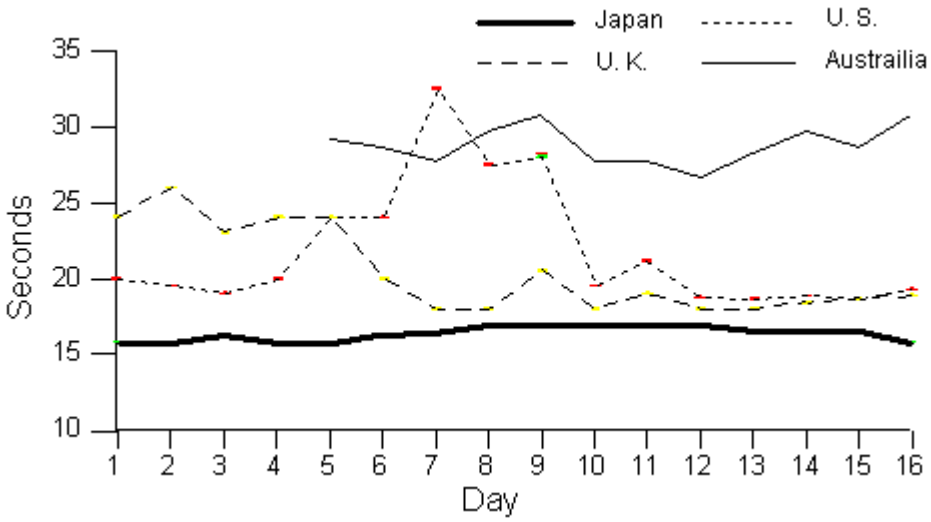
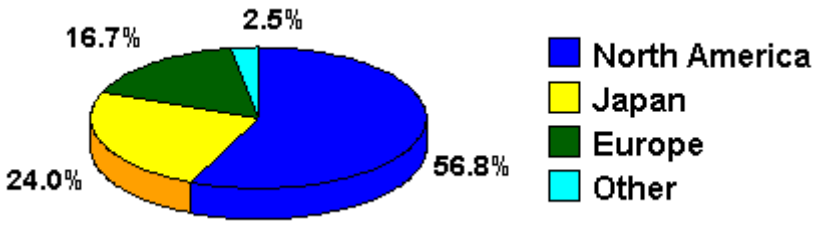


Figure 23: Breakdown of requests by geographic location.



The response times of Figure 22 were measured on a 28.8 Kbps modem connected to the Internet and represent the time to access the home page of the site. Days seven, eight, and nine show somewhat worse response times for the U.S. site. This was caused by problems external to the site, as can be seen by the flat response times for the UK, Japan, and Australia during this same period.

Tables 1 and 2 compare the access times and transmission rates for the Olympic Games Web site to those of home pages of major Web sites around the world measured on Day 14 using 28.8 Kbps modems. These numbers indicate that the Nagano site was one of the most responsive sites on the Internet.

Table: Response comparison, non-USA sites. These numbers and the ones in Table 2 were measured by IBM on a particular day and were not verified by an independent

organization.

Country	Japan	Japan	AUS	AUS	UK	UK
ISP	Olympics	Nifty Service	Olympics	OZMAIL	Olympics	DEMON
Mean Response	16.22	18.15	29.37	25.02	17.36	20.82
Transmit Rate(Kbps)	25.78	22.05	16.82	18.69	25.84	21.28

Table 2: Response comparison, USA sites.

Country	USA	USA	USA	USA	USA	USA
ISP	Olympics	Compuserve	AOL	MSN	NETCOM	AT&T
Mean Response	18.26	19.14	23.91	20.17	19.72	19.71
Transmit Rate (Kbps)	23.31	21.86	19.05	18.60	21.01	20.84

Even during peak periods, the system was never close to being stressed. Virtually all of the delays in response times for the Olympic Games Web site in Tables 1 and 2 were caused not by the Web site but by the client and the client connection to the network. For clients communicating with the Internet via fast links, response times were nearly instantaneous. A key component in reducing CPU cycles was our caching technology. As a result of DUP and prefetching, we were able to achieve cache hit rates close to 100%.

All dynamic pages could be cached in memory without overflow. Therefore, the system never had to apply a cache replacement algorithm. In general, the memory requirements for caching data belonging to a single Web site are significantly less than those required for proxy caches which store data from several Web sites [4,1,10]. The maximum memory required for a single copy of all cached objects was around 175 Mbytes.

One of the most interesting moments was a peak of 98,000 requests per minute during the Men’s Ski Jumping finals on Day 10. Because of time zone differences and geographical routing, 72,000 requests per minute were served from the Tokyo site alone during this period. The Tokyo site had the capacity to service requests quickly even during this peak moment.

6. Summary and Conclusion

We have described the system and key techniques used for performance and high availability at the official Web site for the 1998 Olympic Winter Games in Nagano, Japan. The Web site utilized thirteen SP2 systems scattered around the globe containing a total of 143 processors, 78 GB of memory, and 2.4 terabytes of disk space. This amount of hardware was deployed both for performance purposes and for high availability. Performance considerations were paramount not only because the Web site was one of the most popular sites at the time of the Olympic Games but also because the data being presented to clients was constantly changing. Our system was able to serve pages to clients quickly during the entire Olympic Games even during peak periods. In addition, the site was available 100% of the time.

Caching dynamic Web pages was crucial for achieving good performance. A key problem with caching dynamic pages is determining when a cached page has become obsolete. We developed and implemented a new algorithm we call *Data Update Propagation (DUP)* which identifies the cached pages that have become stale as a result of changes to underlying data on which the cached pages depend, such as databases. A stale page can either be invalidated or updated directly in the cache. For the Olympic Games Web site, we were able to update stale pages directly in the cache. This allowed us to achieve cache hit rates of close to 100% compared to 80% for an earlier version of our system which did not use DUP.

The DUP algorithm for keeping caches updated can be applied to other situations where objects are being cached and underlying data affecting the values of objects are constantly changing. We are currently looking at how DUP can be applied to domains outside the Web.

Acknowledgments

Many people made valuable contributions to the 1998 Olympic Games Web site. Cameron Ferstat, Paul Reed, and Kent Rankin contributed to the network design. John Thompson, Jerry Spivak, Michael Maturo, Kip Hansen, Brian Taylor, Elin Stilwell, and John Chiavelli contributed to the Web site design. Kim Benson provided useful information for this paper. Glen Druce contributed to the news, photos, and Lotus Notes design.

Bibliography

1 P. Cao and S. Irani.
Cost-Aware WWW Proxy Caching Algorithms.
In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, December 1997.

2 J. Challenger and A. Iyengar.
Distributed Cache Manager and API.
Technical Report RC 21004, IBM Research Division, Yorktown Heights, NY, October 1997.

3 D. E. Comer.
Internetworking with TCP/IP.
Prentice Hall, Englewood Cliffs, NJ, second edition, 1991.

4 B. M. Duska, D. Marwood, and M. J. Feeley.
The Measured Access Characteristics of World-Wide-Web Client Proxy Caches.
In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, December 1997.

5 G. Hunt, G. Goldszmidt, R. King, and R. Mukherjee.
Network Dispatcher: A Connection Router for Scalable Internet Services.
In *Proceedings of the 7th International World Wide Web Conference*, April 1998.

6 A. Iyengar and J. Challenger.
Improving Web Server Performance by Caching Dynamic Data.
In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, December 1997.

7 A. Iyengar and J. Challenger.
Data Update Propagation: A Method for Determining How Changes to Underlying Data Affect Cached Objects on the Web.
Technical Report RC 21093(94368), IBM Research Division, Yorktown Heights, NY, February 1998.

8 A. Iyengar, E. MacNair, and T. Nguyen.
An Analysis of Web Server Performance.
In *Proceedings of GLOBECOM '97*, November 1997.

9 Y. H. Liu, P. Dantzig, C. E. Wu, J. Challenger, and L. M. Ni.
A Distributed Web Server and its Performance Analysis on Multiple Platforms.
In *Proceedings of the International Conference for Distributed Computing Systems*, May 1996.

10 S. Manley and M. Seltzer.
Web Facts and Fantasy.
In *Proceedings of the USENIX Symposium on Internet Technologies and Systems* , December 1997.

11 Microsoft Corporation.
(ISAPI) Overview.
http://www.microsoft.com/

12 MindSpring.
The BGP Page.
http://www.mindspring.com/~jlindsay/bgp.html

13 Netscape Communications Corporation.
The Server-Application Function and Netscape Server API.
http://www.netscape.com/newsref/std/server_api.html

14 Open Market.
FastCGI.
http://www.fastcgi.com/

15 Various.
Information on CGI.
http://hoohoo.ncsa.uiuc.edu:80/cgi/overview.html
http://www.yahoo.com/Computers/World_Wide_Web/CGI___Common_Gateway_I
and
http://www.w3.org/pub/WWW/CGI

About this document ...

A Scalable and Highly Available System for Serving Dynamic Data at Frequently Accessed Web Sites

A postscript version of this document is available here.

This document was generated using the **LaTeX**2HTML translator Version 98.1p1 release (March 2nd, 1998)

The command line arguments were:
latex2html `-split +0 -show_section_numbers -no_navigation sc98.tex`.

The translation was initiated by James Challenger on 1998-08-05

Challenger, Dantzig, and Iyengar[0]
1998-08-05