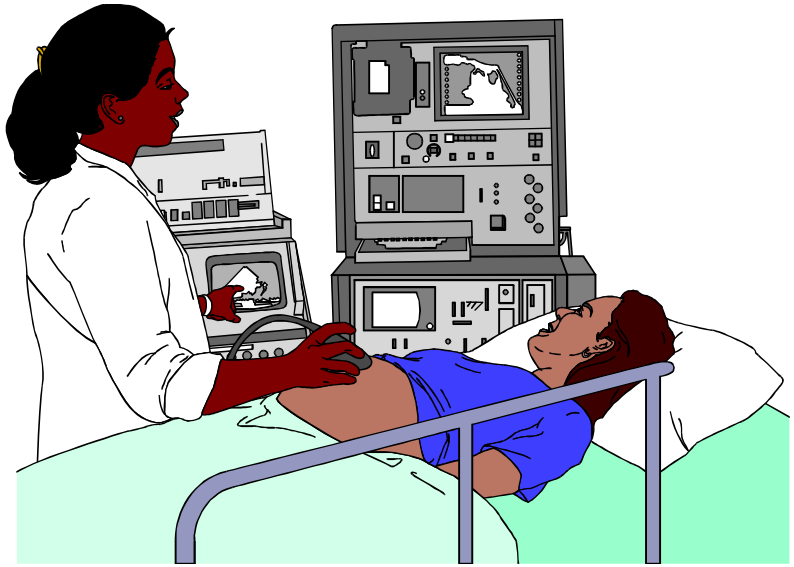
Web Quality of Service



Antoni Drudis
John Mendonca

WebQuality of Service

WebQoS is a web- and application-server performance-enhancement tool set that improves the web access experience for end-users. This goal is reached by providing a GUI-based measurement and management environment that helps the system and network administrator configure and tune his or her web farm so it can satisfy the service level agreements with the owner of the site.

WebQuality of Service

WebQoS solutions span across heterogeneous hardware platforms, (HPPA, Intel, SPARC) operating systems, (HPUX, Windows/95/98/NT) web servers products and versions (Netscape Enterprise Server, Apache) and other internet components such as many browsers and network routers.

Web transactions

Transactions are sets of operations that leave the state inconsistent until the last operation of the set is completed.

Some transactions are physical in nature (e.g. disk I/O operation, memory update) Other are logical (e.g. delete all dependent entries when a master entry is deleted)

Web transactions

Characteristics of web-based transactions:

- Long transactions, with components processed in more than one server: transaction management may be difficult.
- The number of incomplete transactions might be very high: efficiency of locking algorithms may be questioned.
- Silent failures: transaction cancellation notice not sent to the web server

Web transactions

Sessions...

Sessions are sets of transactions that require a consistent environment. Example: consult a product database which contains user- and time-dependent discount offers, and, then, place an order based on these offers.

Web transactions

Example of session:

- Buy a product on a web: update the inventory, issue a payment request, ensure merchandise delivery, create an accounting record, notify suppliers if inventory is low, track the status of related orders to minimize shipping costs, etc.

Web transactions

Characteristics of web-based sessions.

- Sessions are not transactions: the state of the session is kept consistent -across transactions- from the beginning to the end of each session step
- A typical step contains one or more transactions and the user may leave the session at any time without having to undo any previous process.

Web transactions

Characteristics of web-based sessions.

- Transactions are more predictable than sessions because *sessions* derive from the behavior of the user and *transactions* are the result of a product implementation

Data flow

- Client:
  - Browsers
  - Web spiders, wanderers, and robots
- Proxy server
- Gateway
- Routers
- Switches

Data flow

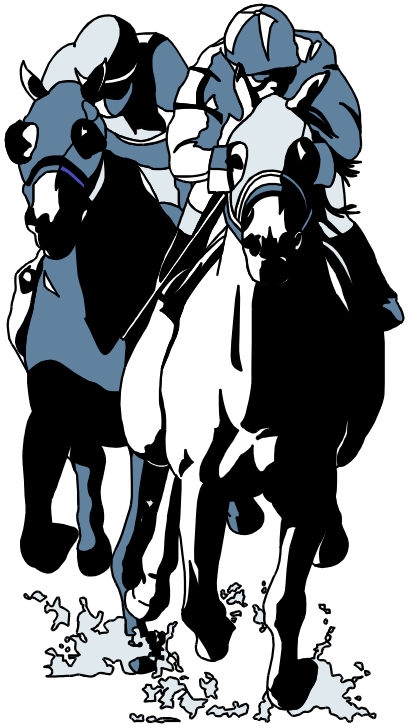
- Firewall
- Localdirector (load balancing)
- Web server
- Application server
- Database server

Web servers

- Typical goals of web servers
  - Scalable architecture: from single host to large farms of cooperating web server instances
  - Fault-tolerant: unattended 24x7 operation
  - Flexible configuration: from single-instance web servers to exploiting the features of the platform
  - Extensible (e.g. MIME)
  - Lightweight (secondary goal)

Web servers

- Typical web server implementation
  - http daemon
  - TCP/IP connections
  - multithreading
  - queue management



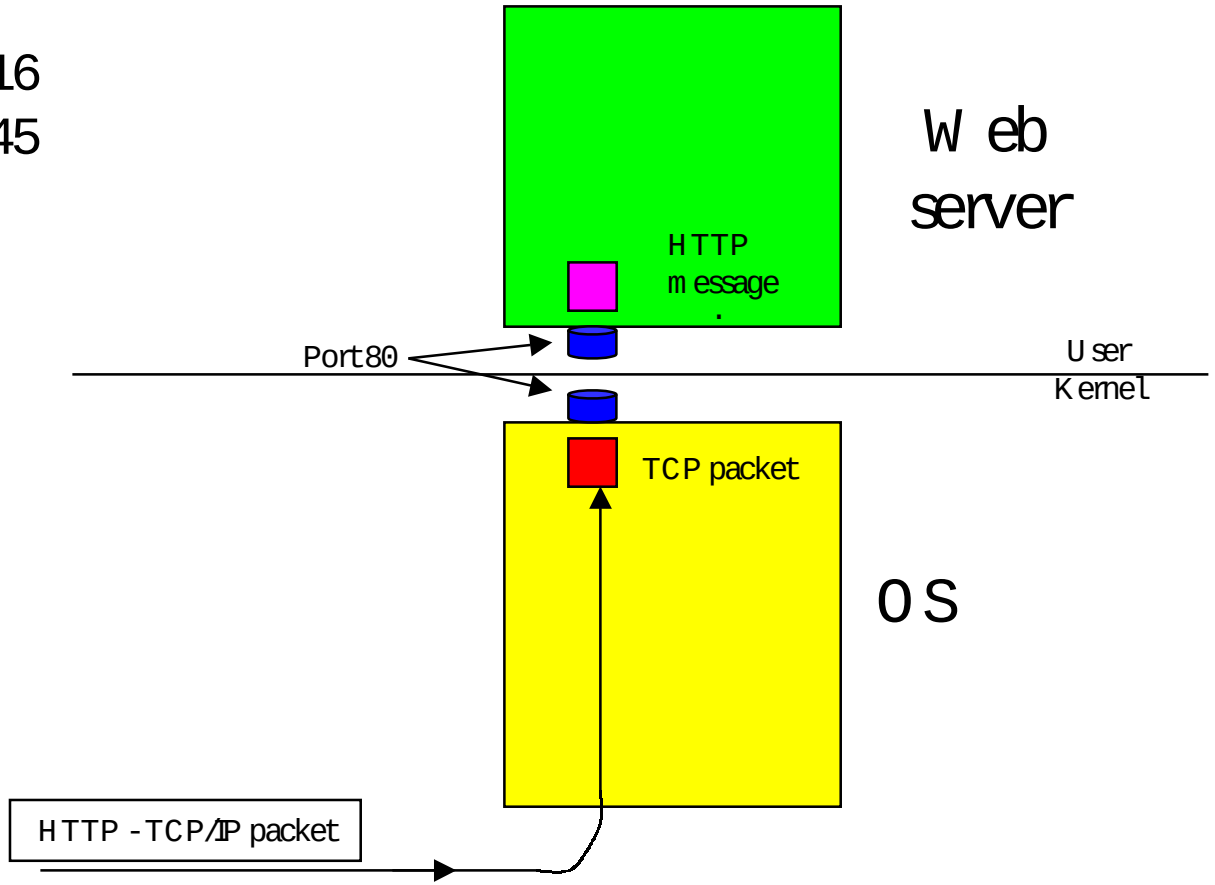
Web servers

- Web server implementation
  - Stateless access
  - Scalable to a large number of clients
  - Extension mechanisms to handle state, transactions, database access, client-side and server-side processing, encryption, etc.

Web servers

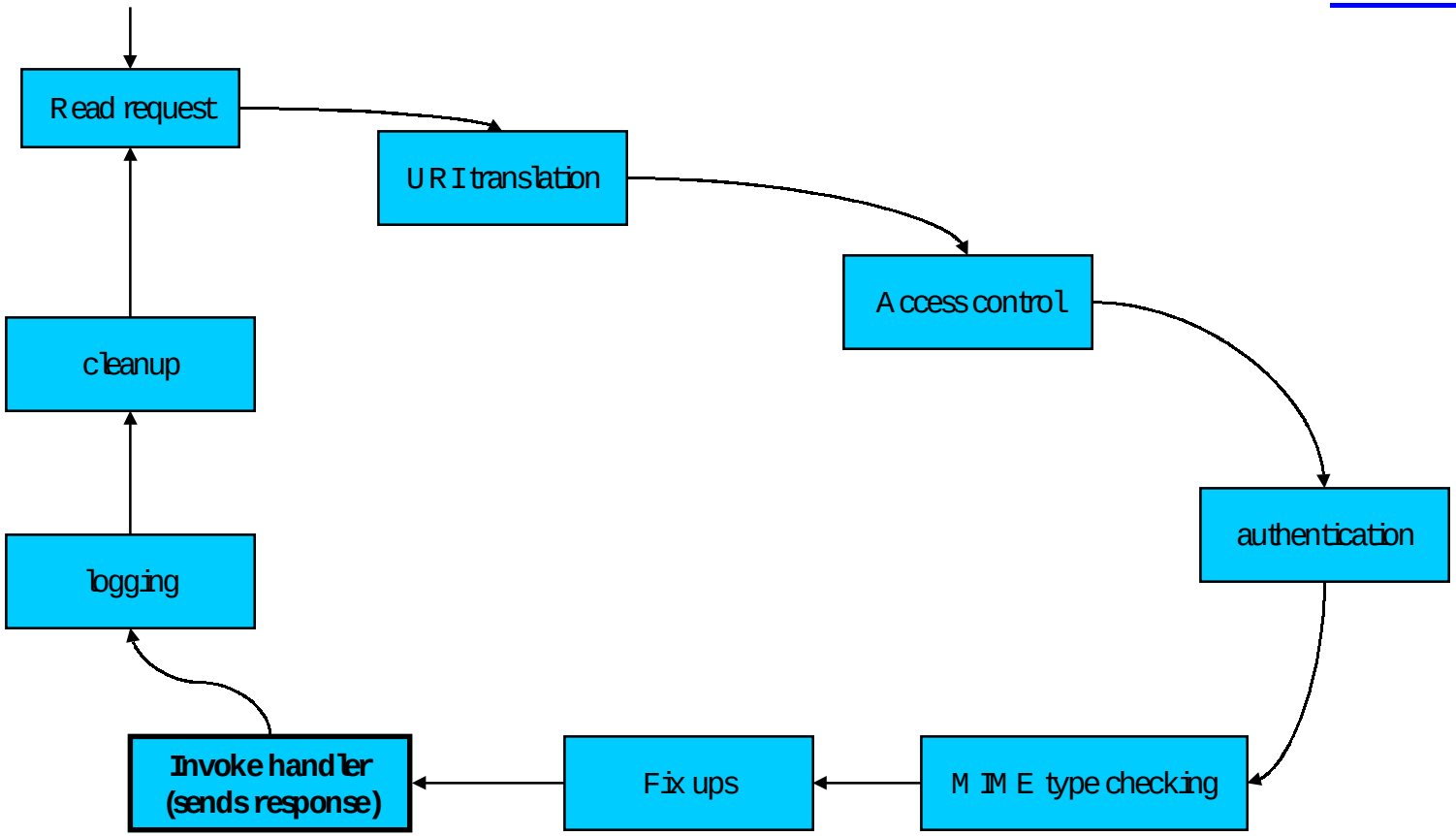
Accepting requests

HTTP 1.1 RFC 2616
HTTP 1.0 RFC 1945



Web servers

Processing requests

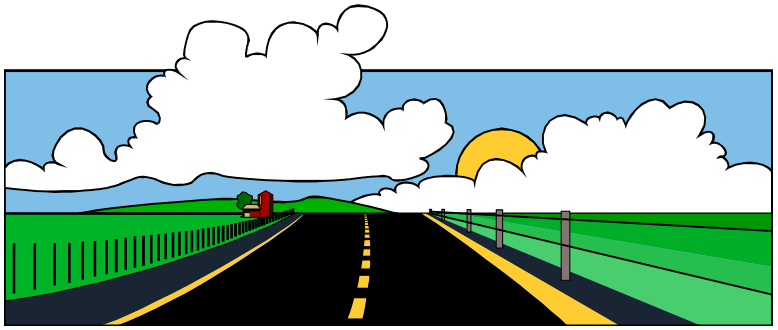


Web servers

Web server performance depends on the number of concurrent users and the available network and server architectures.

Web servers

- Unpredictable and fluctuating demand for web services
- Peaks in resource consumption (disk, CPU)
- Undifferentiated service level (best effort rules)



User expectations

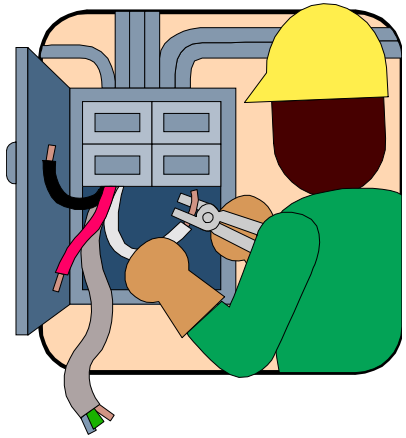
- Users will assume that the server is down if it does not respond to the user requests in a timely manner.
- The web server overload problem is compounded by impatient users hitting the *'reload'* button when their perceived response time exceeds a few seconds.

User expectations

- Server response time grows exponentially with the number of concurrent requests
- Relatively small increases in the number of concurrent users may produce fatal consequences for the e-commerce site.

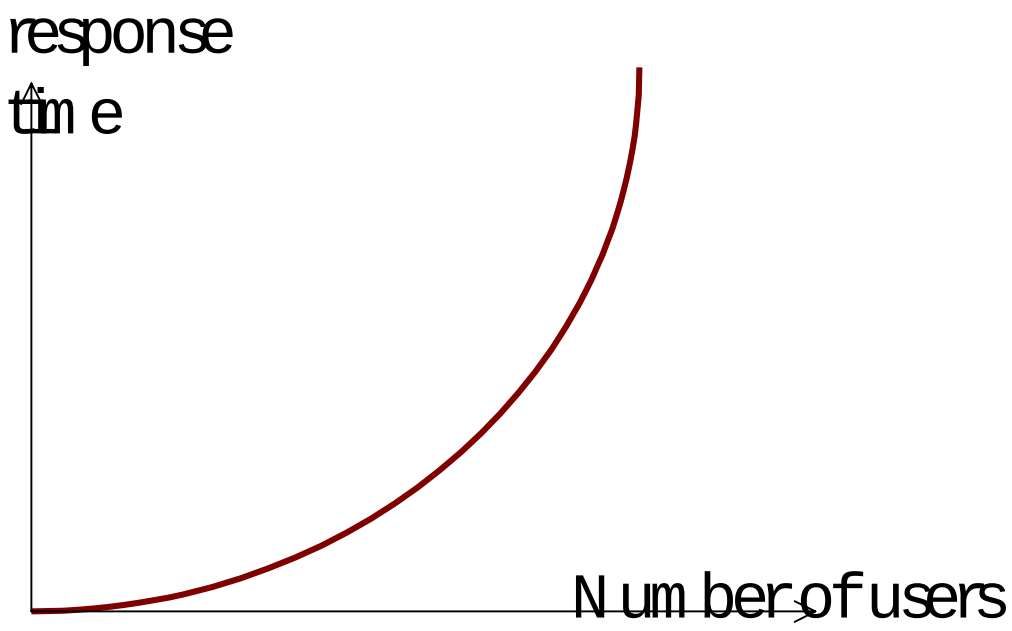
User expectations

Some users will assume that the server is down if it does not respond to the user requests in a timely manner.



Introduction to WebQoS: WebQoS value proposition

Server response time grows exponentially with the number of concurrent requests



Introduction to WebQoS: WebQoS value proposition

When the web server is working at its peak capacity, relatively small increases in the number of concurrent users produce fatal consequences for the e-commerce site.

	Current	10% increase	20% increase	30% increase
Concurrent users	20000	22000	24000	26000
Response time	3.0	3.9	5.7	12.3
Per cent users lost	0	0	20	80
Sales per day	$10 million	$11 million	$9.6 million	$2.6 million

Introduction to WebQoS: WebQoS value proposition

When the web server is working at its peak capacity, relatively small increases in the number of concurrent users produce fatal consequences for the e-commerce site.

	Current	10% increase	20% increase	30% increase
Concurrent users	20000	22000	24000	26000
Response time	3.0	3.9	5.7	12.3
Per cent users Lost	0	0	20	80
Per cent buyers lost	0	0	0	0
Per cent non-buyers lost	0	0	4800/21600= 22.22%	20800/23400= 88.88%
Sales per day	$10 million	$11 million	$12 million	$13 million

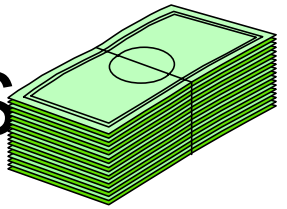
(asuming 10 per cent of users are actual buyers)

Introduction to WebQoS: WebQoS value proposition

An unexpected 30% increase in the number of customers produces extra-sales of $3,000,000 per day as opposed to a loss of $7,400,000 per day when there is no web server capacity protection.

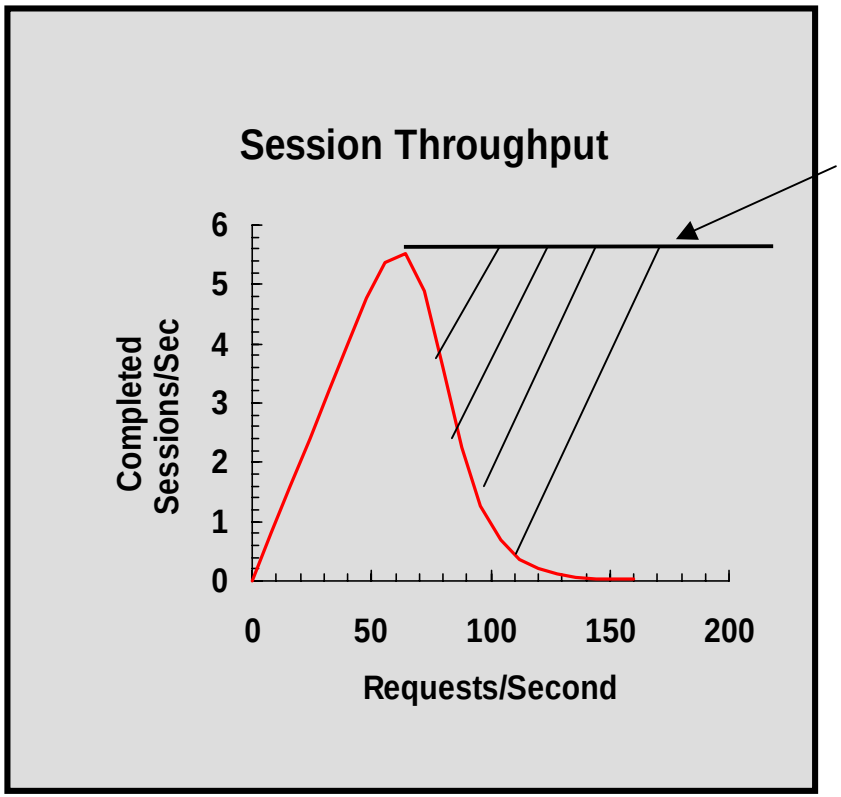
In this example, WebQoS has produced an extra-income of $10,400,000 per day when the load exceeds the web server capacity by 30%

Introduction to WebQoS: WebQoS value proposition



Number of users	Sales without using WebQoS	Sales when using WebQoS
20000	10.0	10
22000	11.0	11
24000	9.6	12
26000	2.6	13

Introduction to WebQoS: WebQoS value proposition



With HP WebQoS

A 20-200% capacity boost!
Higher sales per peak period!
Fewer sales and support calls!
Longer connect times!

Introduction to WebQoS: WebQoS value proposition

- The principle behind WebQoS design is to intercept the request of the less-desirable users (20,800 non-buyers in that e-commerce site) so the target users may have a more pleasant web experience and a positive business relation with the site

Introduction to WebQoS: WebQoS value proposition

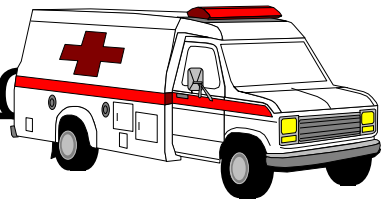
- If the number of users keeps growing over the capacity of the web server, buyers will eventually be redirected to other sites or rejected. In that case, the value proposition of WebQoS is to reduce the frustration of the user who has to wait a long period of time but is still unable to complete the transaction.

Introduction to WebQoS: WebQoS value proposition

Servers using WebQoS provide better service to all users

- Higher-priority users will have better response time
- Lower-priority users will not have to wait for rejection if the server is too busy to provide them a service at this time

Introduction to WebQoS: solution strategy



The analysis of the behavior of the server and the architecture of the network does not suffice to provide an accurate picture of the traffic and process bottlenecks.

Introduction to WebQoS: solution strategy

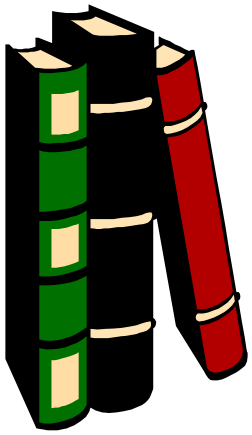
For example, while examining a given network topology for e-commerce, the user may want to replicate some services to minimize the number of packets that traverse subnets. But a dynamic analysis of the real traffic could indicate that the percentage of packets traversing subnets does not warrant the cost of that replication.

Introduction to WebQoS: solution strategy

Because of the fluctuations in demand, server optimization is dynamic: in addition to the architecture of the e-commerce solution, users have to tune their servers based on current and peak demand.

WebQoS addresses that need by adjusting the behavior of the server to the user's demand and the service policies defined by the administrator.

Introduction to WebQoS: WebQoS components



Terminology:

- SLO: Service level objective
- User classes/Request classification
- Service classes/Process groups

Introduction to WebQoS: WebQoS components

- Capacity protection in overload situations
- Process groups to optimize use of system resources
- Classify user requests and prioritize service
- Management interface to analyze server behavior and configure WebQoS profiles

Introduction to WebQoS: objectives and design principles

- Lightweight
- Efficient
- Scalable
- Portable to multiple O.S. and web servers

Introduction to Web QoS: building blocks

- Service
- System
- Site
- SLO
- Corrective actions

Introduction to WebQoS: building blocks

- User request classes
- Process groups
- Service classes
- Sessions
- Management tools

Introduction to WebQoS: building blocks - service

A service refers to something of value that a business is offering to other businesses or consumers. For example, a business may provide email, messaging, documentation publishing, cataloging, and Electronic Commerce services.

Introduction to WebQoS: building blocks - service

- Service: logical grouping of web and/or application servers

  A service is a collection of service components such as web sites, ftp sites, mail sites, customer databases, pricing rules, application logic modules, and so on.

Introduction to WebQoS: building blocks - system

- System: physical host controlled by a single WebQoS management agent

  The system name can be the hostname (for example, hpxx123.hp.com) or an alias (e.g. omocha)

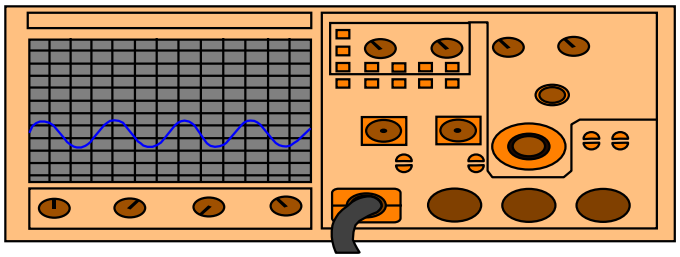
Introduction to WebQoS: building blocks - site

- Site: web server instance or application server

  WebQoS sites are identified by their IP address and port number.

Introduction to WebQoS: building blocks - SLO

- **SLO: Service level objective** in terms of web server performance data that is being managed by WebQoS software.



Introduction to WebQoS: building blocks - SLO

To ensure that your business policies are reflected in your Web applications, they need to be translated into service level objectives (SLOs) and thresholds. SLOs are business-oriented policies and thresholds are operations-oriented policies.

Introduction to WebQoS: building blocks - SLO

Business-oriented service level objectives define the following:

- Response time -- measured from the time a request enters the server to the time it leaves the server.
- Concurrent session capacity -- created when a user's initial request is accepted and maintained until the session times out.

Introduction to WebQoS: building blocks - SLO

Operations-oriented thresholds for capacity protection determine the following:

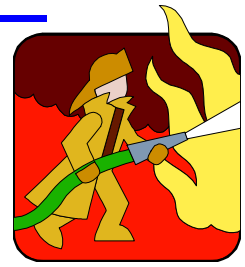
- Average CPU load--measured on the local system.
- Queue depth--the maximum number of service requests waiting that are not yet forwarded to the web server.
- Maximum number of concurrent users accessing the WebQoS hosts.

Introduction to WebQoS: building blocks - SLO

The administrator defines SLOs, thresholds, and their relative priorities. WebQoS can trade off meeting policies based on these priorities.

Information Technology rules concerning response time, throughput, availability, and priorities for users and applications are translated into SLOs and thresholds.

Introduction to WebQoS: building blocks - corrective actions



The WebQoS administrator corrective actions that are executed when the rules are violated.

Corrective actions are a list of prioritized actions an administrator uses to help bring the SLO or threshold into compliance with the rule.

Introduction to WebQoS: building blocks - user classes

User Classes (short for User Request Classes) determine access priority for requests submitted to a web site.

They enable you to give preferential treatment to your most important customers or transactions, allowing you to meet formal or informal service-level agreements.

Introduction to WebQoS: building blocks - user classes

WebQoS User Classes differentiate service requests as they enter the server system by dividing user requests into categories based on application, client or destination IP addresses, destination port number, and URL document paths. High user class requests have higher priority access to the server.

Introduction to WebQoS: building blocks - user classes

- WebQoS supports three User Classes: high, medium, and low.
- During periods of heavy system load, low priority requests may possibly be redirected or rejected.

Introduction to WebQoS: building blocks - user classes

If a request is accepted, it is scheduled based on its User Class priority.

Based on your configured policies, the request might be immediately processed, or it may wait in the queue while other higher priority requests are processed first.

Introduction to WebQoS: building blocks - process group

- Process group: set of processes being managed by the PRM (Process Resource Management) product.

Introduction to WebQoS: building blocks - service classes

- WebQoS allows you to associate each application with a Service Class. Service Classes determine the resource sharing priority given to an application.

Introduction to WebQoS: building blocks - service classes

- You can set resource sharing policies for application belonging to different classes. WebQoS creates an HP-UX Processing Group for each Service Class and sets a resource entitlement for each processing group, based on your input.

Introduction to WebQoS: building blocks - service classes

- This enables you to give preferential treatment to your most important applications, processes, or Web sites when multiple web sites are running on the same server.

Introduction to WebQoS: building blocks - service classes

- While User Classes prioritize access, Service Classes optimize system resource allocations. WebQoS currently supports three Service Classes: high, medium and low.

Introduction to WebQoS: building blocks - service classes

- Prioritizing access to CPU and disk I/O resources affects the performance of an application.

  For example, requests in the high service class execute in an operating environment with a higher percentage of system resources such as CPU.

Introduction to WebQoS: building blocks - sessions

A session is composed of one or more requests to a web site from the same user.

These requests may arrive over one, or over several connections. Once a session has been granted, a user's remaining requests are guaranteed to be forwarded to the web server, unless the session times out.

Introduction to WebQoS: building blocks - sessions

Under periods of heavy system load, new sessions may not be granted. This is controlled by the WebQoS policies.

When new sessions are rejected, redirected, or deferred, existing sessions continue uninterrupted.

User request prioritization and session management are the primary tools used by WebQoS to provide Capacity Protection.

Introduction to WebQoS: building blocks - sessions

The network manager is responsible for updating the timers that define a session, and for establishing the policies that control whether the session is admitted.

Introduction to WebQoS: building blocks - management

WebQoS management features:

- Lightweight: resources used to reach service level objectives cannot be compromised by the load of management tools
- Ease of configuration: administrators do not need to know the internal design of WebQoS to configure the qoslib parameters
- Ease of use and value of data: management screens provide data meaningful to the web server and WebQoS administrator.

Introduction to WebQoS: building blocks - management

WebQoS management can be bundled in the host being managed or spread across three hosts:

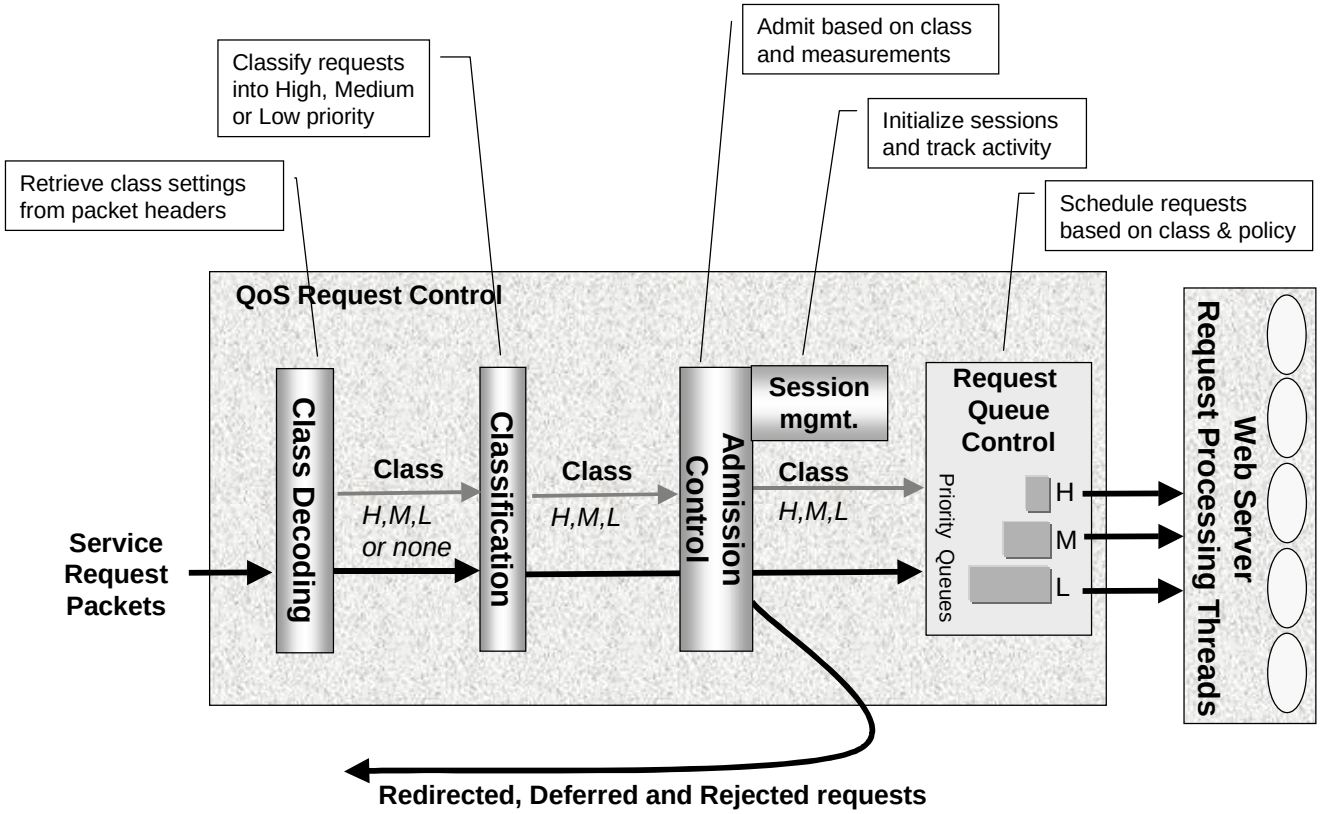
- the host being managed (SCA: service control agent)
- the management host (SCO: service control operator)
- the GUI

WebQoS architecture

WebQoS architecture

- *Web server components*
- *Network components*
- *Configuration and management components*

WebQoS architecture



WebQoS architecture

Web server components at single-host level

- *'qos-ified' web server*
- *WebQoS services (qoslib)*
  - *Queue, accept, classify*
  - *Compute statistics*
  - *Analyze and reinforce SLO compliance*
- *Configuration and management of web servers*

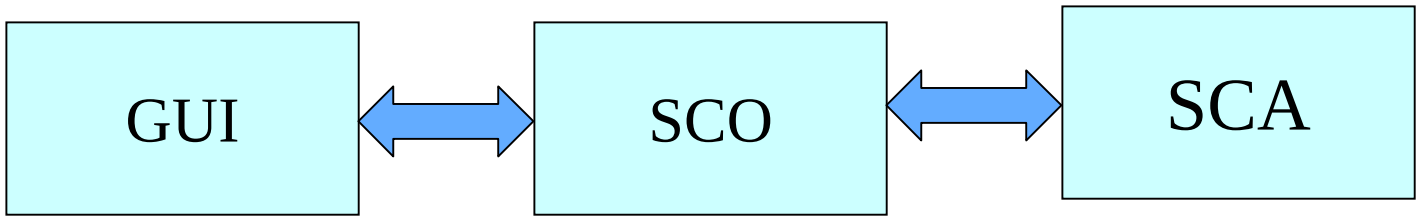
WebQoS architecture

Network components

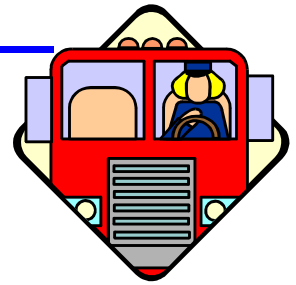
- Library-based tools to set the network TOS
- TCP/IP traffic between WebQoS service control agent (SCA) and WebQoS service control operator (SCO)
- TCP/IP traffic between the configuration and management station (GUI) and the service control operator (SCO)

WebQoS architecture

Management components



WebQoS architecture

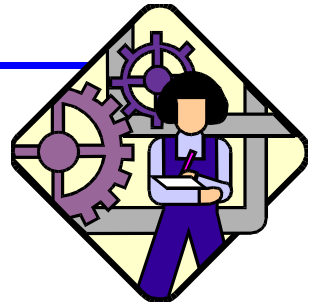


Management components

Service control agent (SCA)

- Gathers status information and sends it to its controller (SCO)
- Caches directives to the SCO when the SCO is not responding to the SCA requests

WebQoS architecture

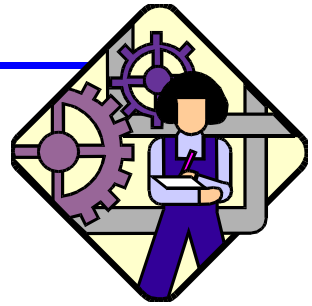


Management components

Service control operator (SCO)

- Communicates with the GUI and the SCA:
  - Maintains the WebQoS configuration database (services, systems, sites, SLOs)
  - Maintains the state of the sites and systems
  - Maintains the history log: status changes in web sites.

WebQoS architecture



Management components

Service control operator (SCO) ...

- Caches configuration and history log to provide better response time
- Manages a configuration and events database

WebQoS architecture

Management components

- GUI: all the (currently) supported WebQoS configuration and management tools are GUI-based.

WebQoS Benefits

HP used Keynote services to analyze the performance of two identical sites, with identical load.

One of them was running the web server using WebQoS and the other was running the web server without WebQoS.

WebQoS benefits

When the system is overloaded:

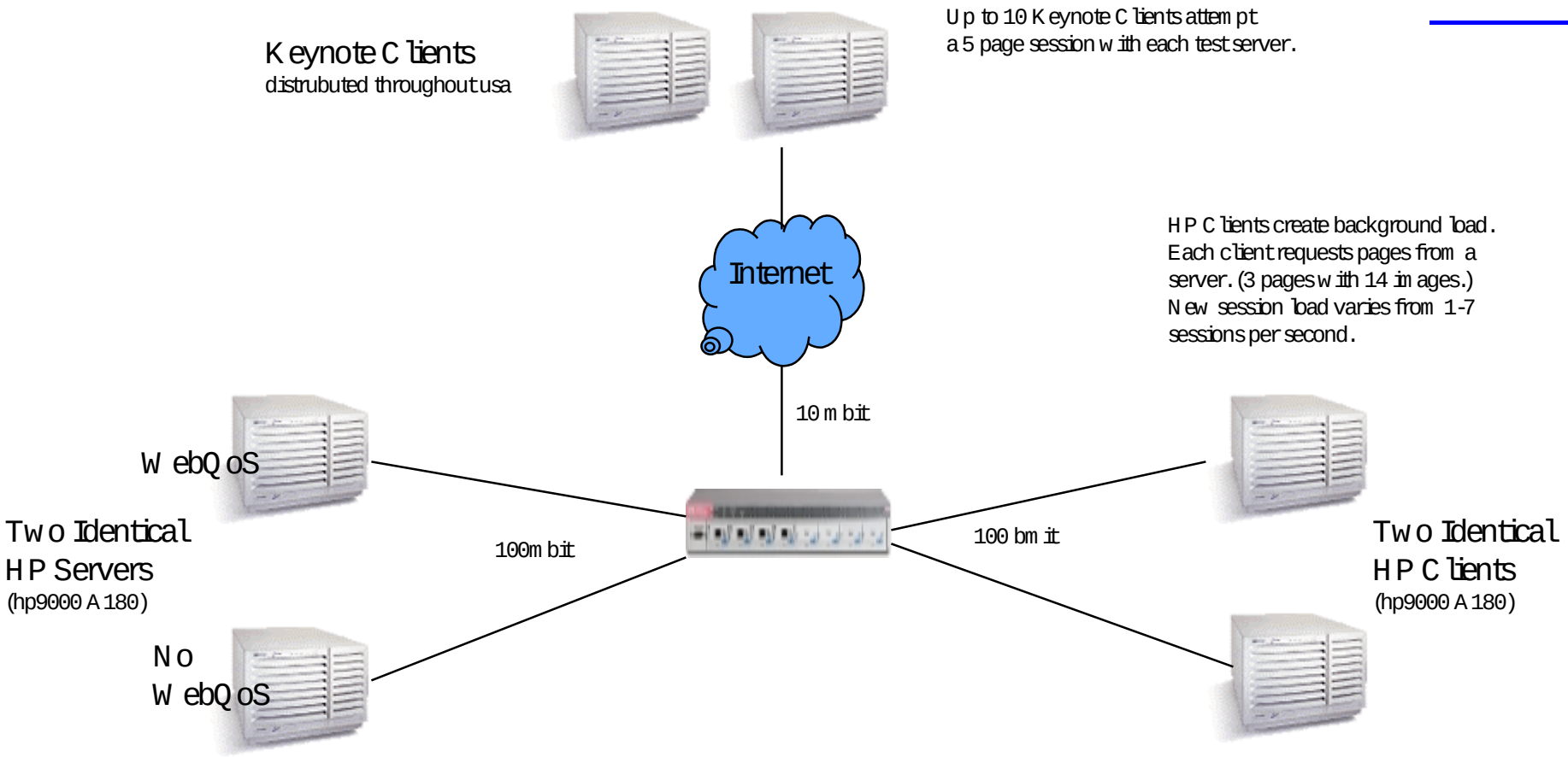
- More visitor sessions are processed (greater sales)
  - Without WebQoS session throughput drops quickly as load increases
  - With WebQoS session throughput is maintained and sessions complete faster. Fewer sessions terminate due to data communication errors.

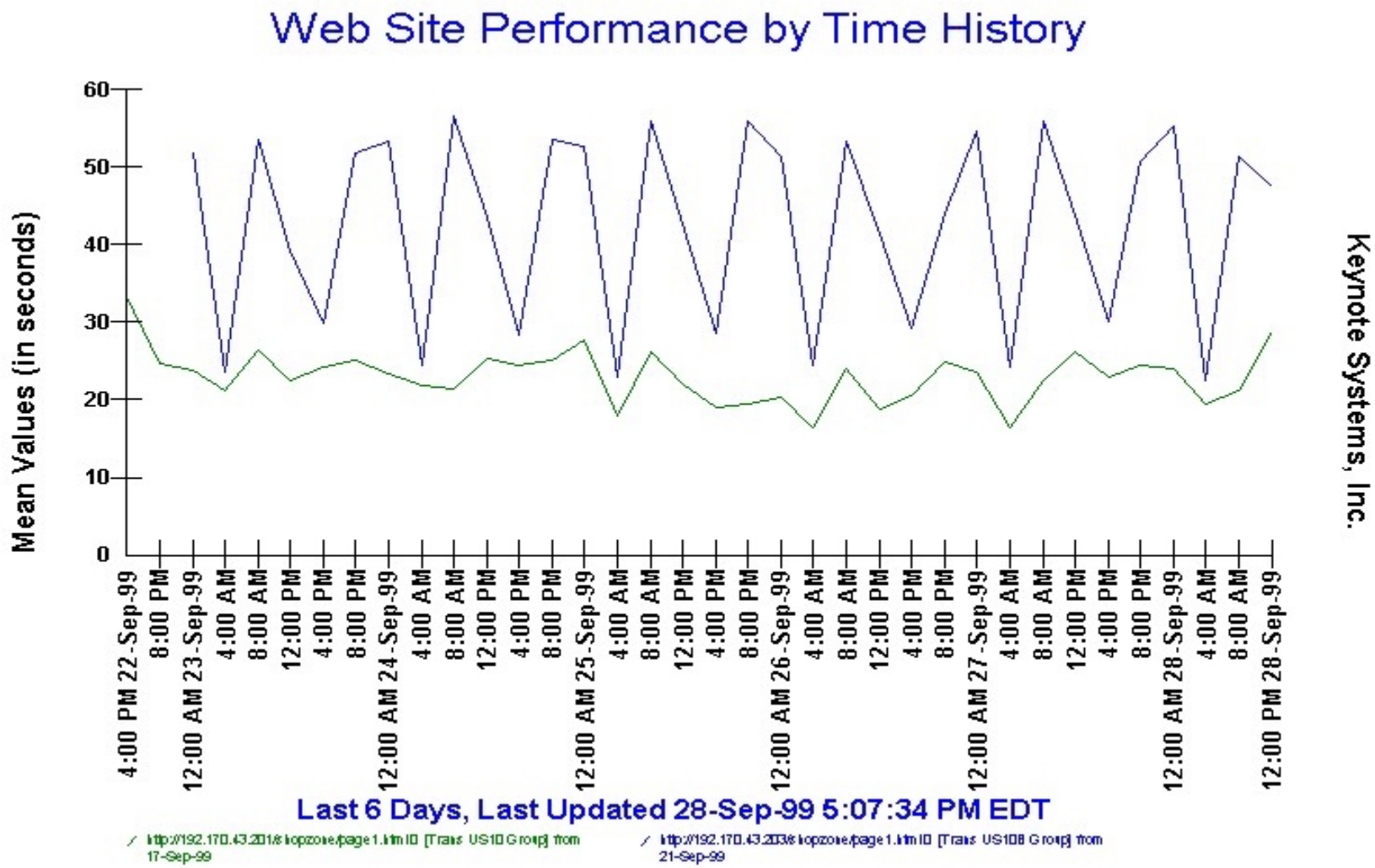
WebQoS benefits

When system is overloaded...

- Improved visitor experience (less time watching hourglass)
  - visitors enjoy better response time
  - visitors that cannot be admitted into a session are quickly deferred,
  - redirected, or rejected instead of being ignored in the middle of a session.

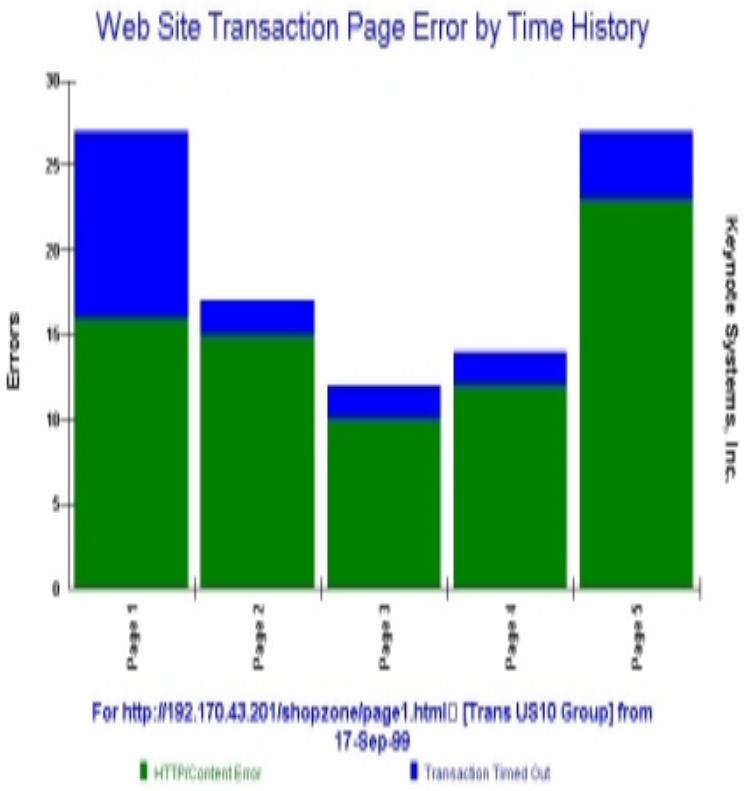
WebQoS test configuration



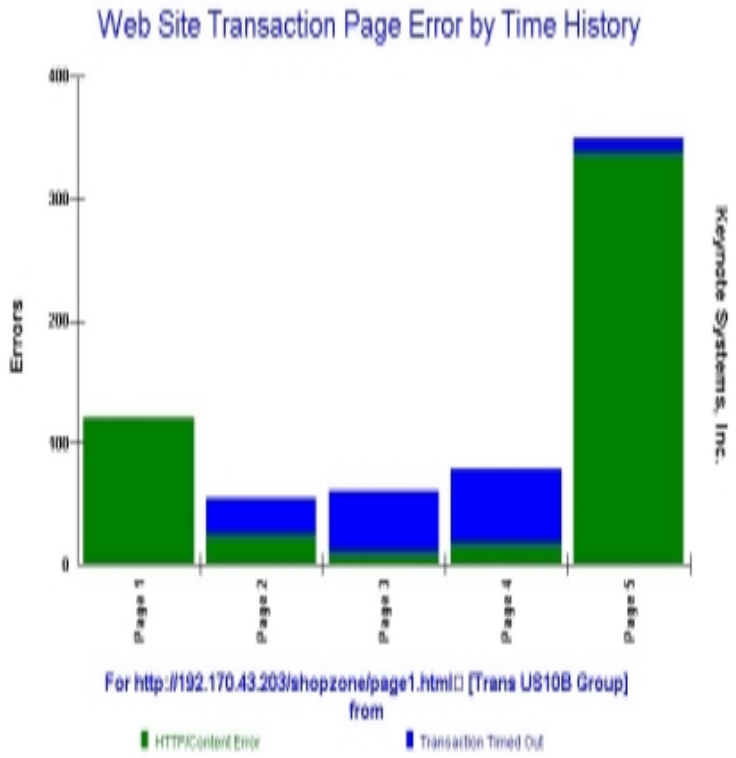

Web Site Performance by Time History
Mean Values (in seconds)
60
50
40
30
20
10
0
4:00 PM 22-Sep-99
8:00 PM
12:00 AM 23-Sep-99
4:00 AM
8:00 AM
12:00 PM
4:00 PM
8:00 PM
12:00 AM 24-Sep-99
4:00 AM
8:00 AM
12:00 PM
4:00 PM
8:00 PM
12:00 AM 25-Sep-99
4:00 AM
8:00 AM
12:00 PM
4:00 PM
8:00 PM
12:00 AM 26-Sep-99
4:00 AM
8:00 AM
12:00 PM
4:00 PM
8:00 PM
12:00 AM 27-Sep-99
4:00 AM
8:00 AM
12:00 PM
4:00 PM
8:00 PM
12:00 AM 28-Sep-99
4:00 AM
8:00 AM
12:00 PM 28-Sep-99
Last 6 Days, Last Updated 28-Sep-99 5:07:34 PM EDT
http://192.170.43.201/scripzone/page1.htm ID [Trans US10 Group] from 17-Sep-99
http://192.170.43.203/scripzone/page1.htm ID [Trans US108 Group] from 21-Sep-99
Keynote Systems, Inc.
WebQoS
NowebQoS

More WebQoS sessions succeed

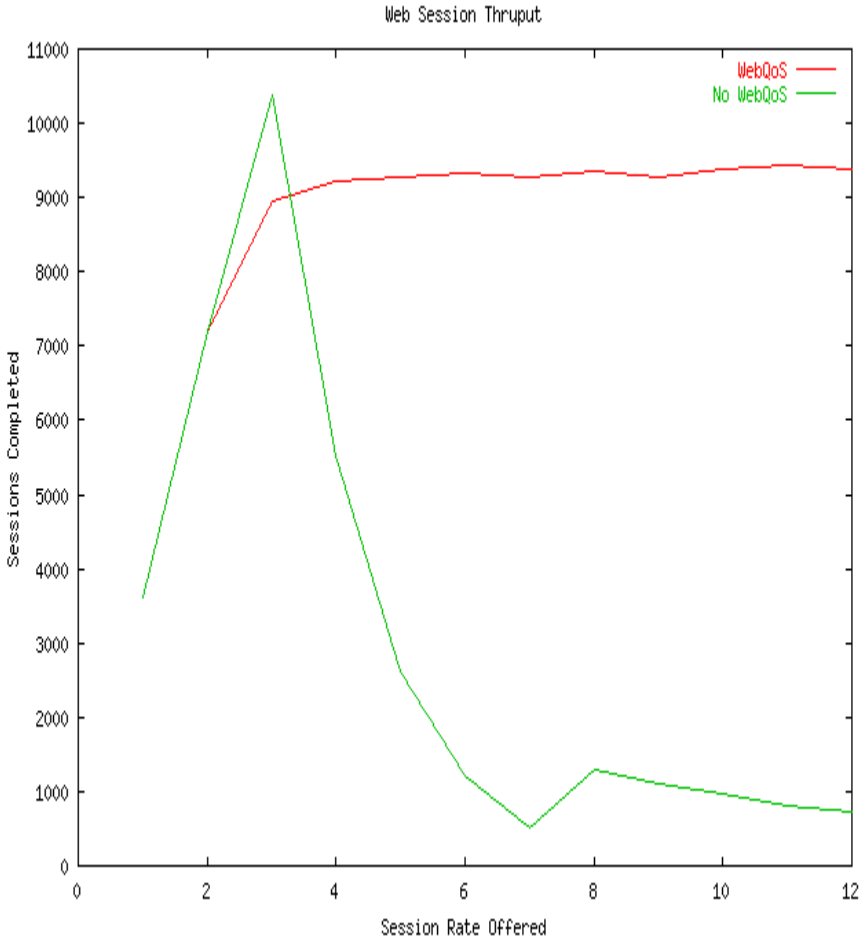
WebQoS - 1182 sessions succeeded
97 sessions failed



NoWebQoS -624 sessions succeeded
663 sessions failed



Web session throughput is higher



QoS session throughput is higher:

- WebQoS has higher sustained throughput when the server is busy.
- No noticeable impact on throughput when the server is not busy

Non-WebQoS throughput drops even faster when the session length is increased!

Notes:

- session is 3 pages long
- each page contains 14 images
- pages 1-3 use 50, 100, and 50ms cpu
- 2 seconds think time between pages

WebQoS benefits: Summary

- Higher session session throughput when the server is overloaded
- Better session completion time when the server is overloaded
- Minimal impact on response time and throughput when the server is not overloaded.