Web Server Software Architectures

Author: Daniel A. Menascé
Presenter: Noshaba Bakht

Web Site performance and scalability

- 1.workload characteristics.
- 2.security mechanisms.
- 3. Web cluster architectures. AND
- 4. Web server software architecture.

Software Architecture Schemes

- Two dimensions generally characterize the architecture:
- (1). The processing model.

It describes the type of process or threading model used to support a Web server operation;

- (2). Pool-size behavior.
- Pool-size behavior specifies how the size of the
- pool of processes or threads varies over time with workload intensity.

Processing Models

- The main options for a processing model are
- 1.process-based,
- 2. thread-based,
- 3. a hybrid of the two.

process-based servers.

- Process-based servers consist of multiple single-threaded processes, each of which handles one request at a time.

The implementation of this server model also called prefork in Apache Unix version and in Apache version 2.0 MPM prefork module.

http://portalux.com/networking/servers/http-servers/

http://www.apache.org/dist/httpd/Announcement.html

Thread based architecture.

- The server consists of a single multithreaded server, each thread handles one request at a time.
- Implementation of this model is seen in Windows NT's version of Apache 1.3.
- http://httpd.apache.org/docs/windows.html
- http://httpd.apache.org/docs2.0/platform/windows.html

Hybrid Model.

- The model consists of multiple multithreaded processes, with each thread of any process handling one request at a time.
- *This Multi-Processing Module (MPM) implements a hybrid multi-process multi-threaded server. By using threads to serve requests, it is able to serve a large number of requests with less system resources than a process-based server. Yet it retains much of the stability of a process-based server by keeping multiple processes available, each with many threads.*
- http://httpd.apache.org/docs-2.0/mod/worker.html

Advantage's / Drawbacks

- The process based architecture is stable, the crash of any process generally does not effect the others so the system continues to operate.
- Creating and killing of processes overloads the web server, due to the address –space management.
- In thread based architecture a single malfunctioning thread can bring the entire web server down resulting in great instability as all the threads share the same address space.
- The memory requirements are much smaller in this model, it is more efficient to spawn threads of the same process due to sharing of the same address space, than to fork a process, various threads easily share data structures as caches.

Hybrid architecture.

- Combines the advantages of both models and reduce their disadvantages.
- Example
- # of processes = p.
- # of threads per process = n.
- Total # of requests handled by the server at a time = n * p.
- If a thread/process crashes # of requests handled at a time = (p -1) * n
- Less memory required by the system to handle the requests and the system is more stable.

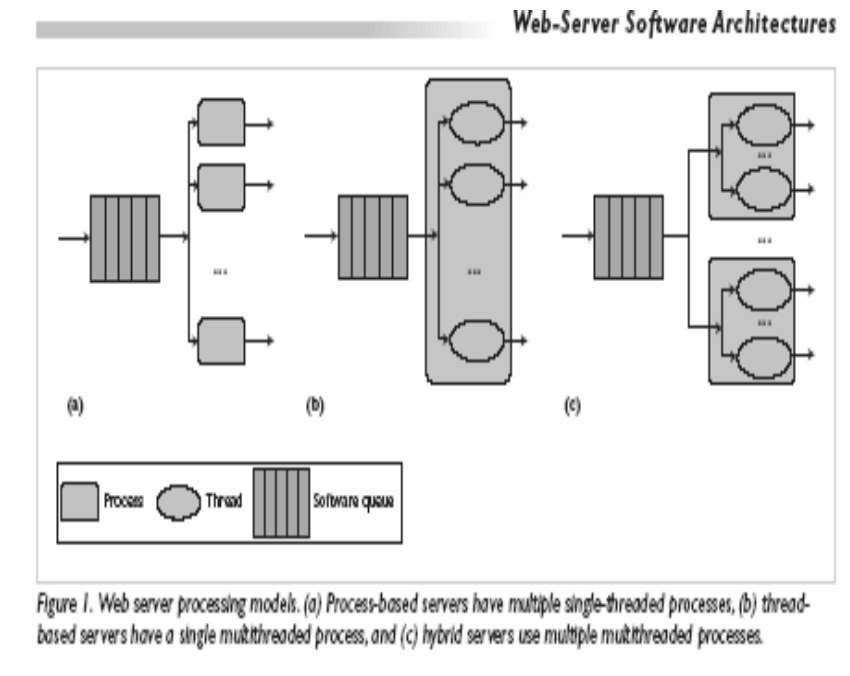


Figure 1. Web server processing models. (a) Process-based servers have multiple single-threaded processes, (b) thread-based servers have a single multithreaded process, and (c) hybrid servers use multiple multithreaded processes.

POOL SIZE BEHAVIOR

- Two approaches are possible.
- 1. Static Approach.
- The Web server creates a fixed number of processes or threads at the startup time. Rush hours:::If the # of requests exceed the number of the p/t the request waits in the queue. Server load is low::::idle p/t.
- The internet Information Server to view HTML pages by windows
- http://www.microsoft.com/windowsserver2003/iis/default.mspx

- 2. Dynamic Approach.
- The Web server creates the number of processes and threads depending on the load, as the load increases so does the pool, letting more requests process concurrently and the queue size is reduced.

Dynamic Pool-size Implementation.

- Web server parent process establishes
- Configuration parameters.

1.Initial configuration parameter is the number p of child processes created to handle the requests correctly.

2.Minimum # m of idle processes in another configuration parameter.

As the load of the request increases the processes are used up and the m # decreases below its sets level, thus telling the parent process to fork and created more child processes to handle the requests efficiently. When the load requests is decreased the min # of the processes increase telling the parent process to kill the extra processes.

3. Another configuration parameter is based on the life of the process or the number of requests it handles before it dies.

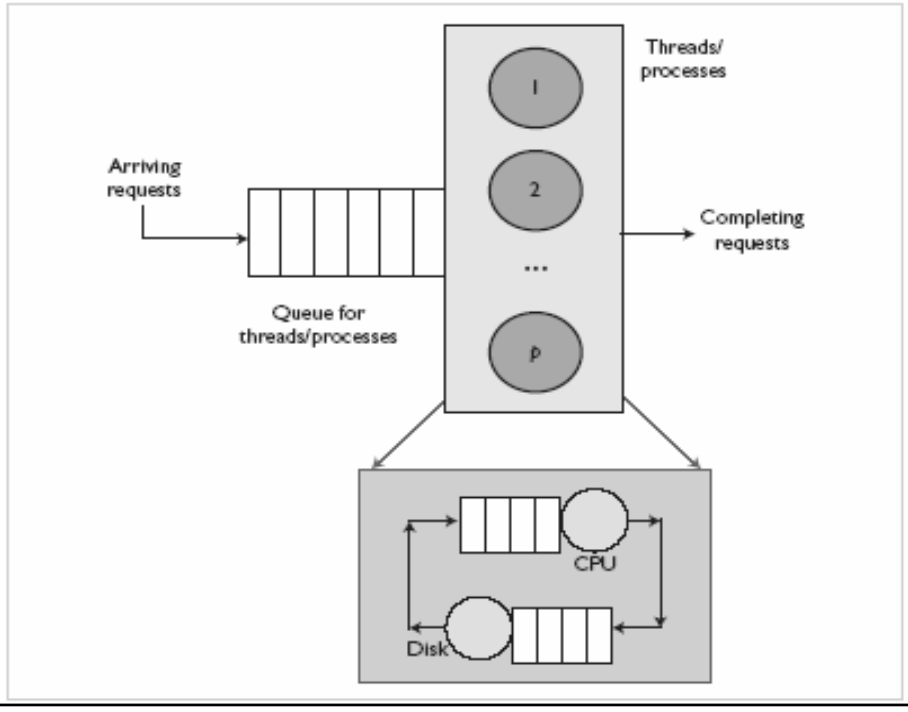
It can be set to zero and the process will have infinite life.

It can be set to a number value.

This life limit reduces the memory loads and leaks.

Performance Considerations

- The Web serve request's total response time is divided into following components.
- 1.Service Time. The total time a request spends on the physical resource level i.e disk and cpu.
- Physical contention. The time the process spends waiting for the physical resources of the web server.
- Software contention. The total time a server request spends waiting in the queue for a process/thread to be available to handle the request.



Architecture Performance

- Queuing Network Model.

It models the physical resources of the Web server and calculate the throughput.

- Markov Chain Model.

It represent the software architecture

Analytic Performance Models

- Queuing Network Model (QN).

A queuing network model represents a system as a set of service centers that process customers (also known as jobs, transactions, or arrivals). Service centers can be any active resource such as a CPU, disk ,printer, network link. Queuing network models can predict the latency, throughput and utilization of a system and its components. It models the Web server's physical resources and computes it throughput.

Distributions of the service and arrival time is the key.

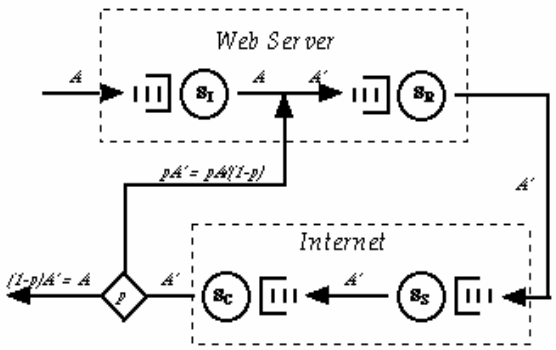


Figure 2. Queueing Network Model of a Web Server

Markov model

- A Markov model is a random process in which the *transition probability* to the next state depends solely on the previous state
- **Transition probability** is a conditional probability for the system to go to a particular new state, given the current state of the system.
- Memoryless model.

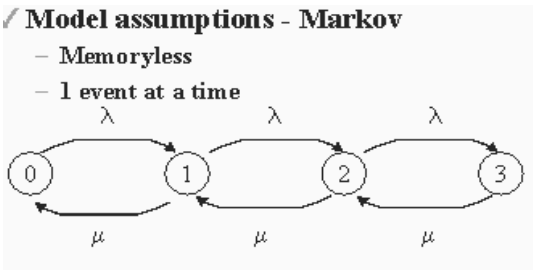
$$p(S) = p(s_1 \ldots s_n) = p(s_1) \prod_{i=2}^{n} p(s_i | s_{i-1})$$

Markov chain

- The characteristic property of this sort of process is that it retains ***no memory*** of where it has been in the past. This means that only the current state of the process can influence where it goes next....
- The process can assume only a finite or countable set of states, when it is usual to refer to it as a ***Markov chain***.
- "Markov chains are the simplest mathematical models for random phenomena evolving in time.
- Indeed, the whole of the mathematical study of random processes can be regarded as a generalization in one way or another of the theory of Markov chains."
- "Chains exist both in discrete time... and continuous time..."

Analytic Performance of the Models

- **Arrival rate (λ)(inverse of mean inter-arrival time)**
- **Service rate (μ) (inverse of mean service time)**



✓ Identify all Probabilities in terms of P_0

$$P_1 = P_0 * \left(\frac{\lambda}{\mu}\right)$$

$$P_0 * \lambda + P_2 * \mu = P_1 * \lambda + P_1 * \mu \Longrightarrow$$

$$P_0 * \lambda + P_2 * \mu = P_0 * \left(\frac{\lambda}{\mu}\right) * \lambda + P_0 * \left(\frac{\lambda}{\mu}\right) * \mu \Longrightarrow$$

$$P_2 * \mu = P_0 * \left(\frac{\lambda^2}{\mu}\right) \Longrightarrow P_2 = P_0 \left(\frac{\lambda}{\mu}\right)^2$$

$$P_1 = P_0 * (\lambda / \mu)$$

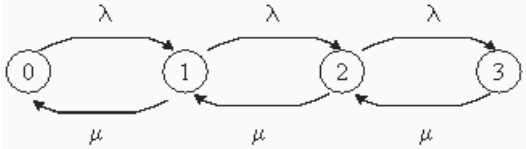
$$P_2 = P_0 * (\lambda / \mu)^2$$

$$P_3 = P_0 * (\lambda / \mu)^3$$

$$P_0 + P_1 + P_2 + P_3 = 1$$

✓ **Model assumptions - Markov**

- Memoryless
- 1 event at a time



$$P_1 = P_0 * (\lambda / \mu)$$

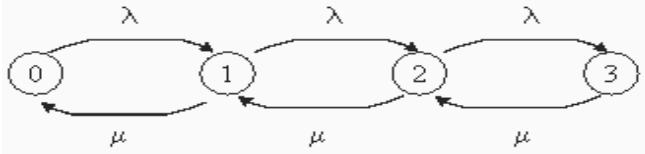
$$P_2 = P_0 * (\lambda / \mu)^2$$

$$P_3 = P_0 * (\lambda / \mu)^3$$

$$P_0 + P_1 + P_2 + P_3 = 1$$

$$P_0 = 1/(1 + (\lambda / \mu) + (\lambda / \mu)^2 + (\lambda / \mu)^3$$

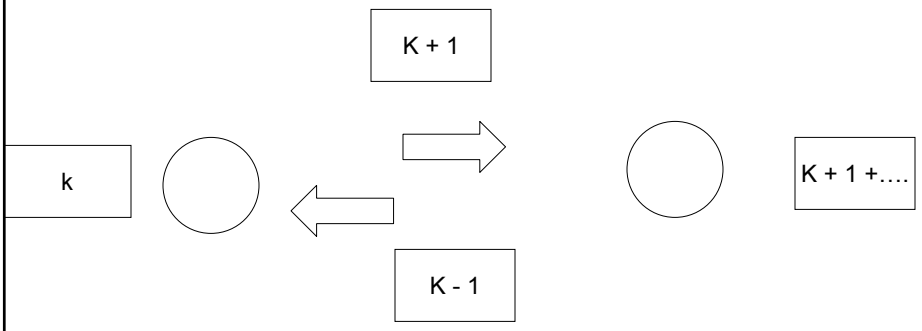
- **Model assumptions - Markov**
  - Memoryless
  - 1 event at a time

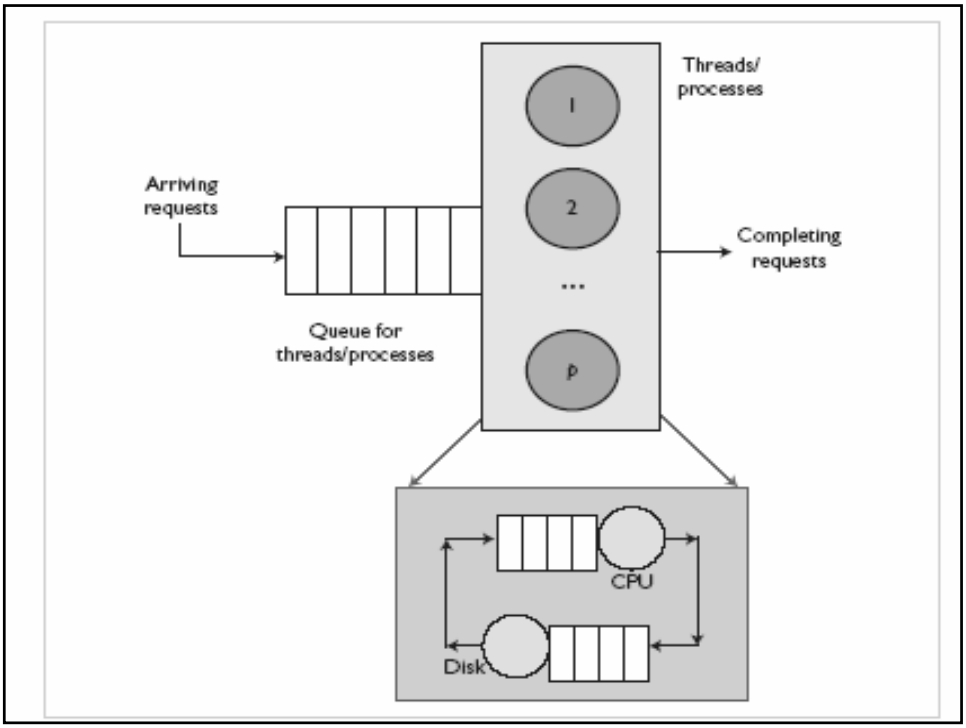


- Idle => $P_0 = 8/15$
- Utilization = busy => $1 - P_0 = 7/15$
- Throughput => $\mu * (1 - P_0) = 14/15$
- **Average queue size** => $1 * P_1 + 2*P_2 + 3*P_3 = 1(4/15) + 2(2/15) + 3(1/15)$ => **11/15**

The state k represent # of the request in the system (waiting or being handled).

Transition of k to k+1 occur at a rate at which new requests arrive at the server. The state k -1 indicates the completion of the request.





Idle Processes Pool

- The average number of idle processes is a function of the average arrival rate of requests for three different values of the active processes p.
- For very light loads.

idle processes ~ total # of processes.

- When the load increases the # of idle processes decrease slightly in the beginning and precipitously as the arrival rate reaches to its maximum possible value.
- This maximum value which is 5 req/sec in this case
- Represent that the web server becomes unstable at this point and the queue for the processes grows without bound.

Table 1. Scenarios of different numbers of idle processes versus arrival rate variations.

Scenario	Number of processes	Arrival rate (requests per second)	Average number of idle processes
1	20	4.5	10.3
2	20	4.8	5.3
3	30	4.8	11.3
4	30	4.9	6.6
5	40	4.9	11.4

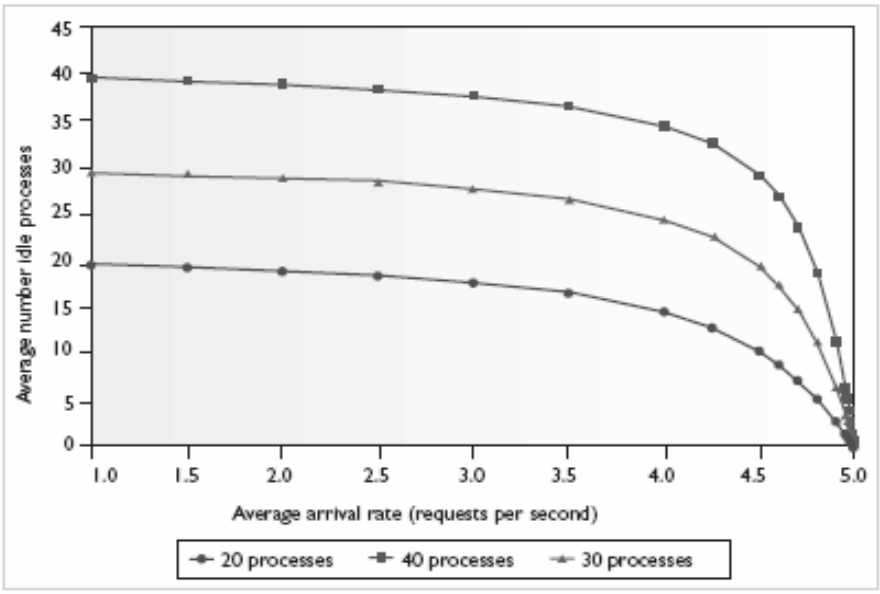


Figure 3. Average number of idle processes versus the average arrival rate of requests. At low loads, almost all processes tend to be idle. At high loads, the number of idle processes falls to zero very quickly.

Web Server @ Dynamic Pool Size

- Maintain an average of idle processes to be 10 – 11.

Final Remarks

- Software connection and architecture effects the web server performance.
- Computer system web server can adjust dynamically the pool of the processes with the help and efficient use of the analytic performance models.

Reference Articles.

- Scaling The Web DanielA.Menasce

Response-Time Analysis of Composite Web Services.
(Submit the query to various search engines in parallel)
Fork and join model.

- http://cs.gmu.edu/~menasce/papers/IEEE-IC-RespTimeWS-January2004.pdf

Scaling Web Sites through Caching

- http://cs.gmu.edu/~menasce/papers/IEEE-IC-ScalingCaching-July-2003.pdf
- Scaling your E-business

Traffic spikes and its control via the building of scalable infrastructure.

- http://www.softwaremag.com/archive/2001feb/ScalingE-business.html

- Sun Secure Web Server Reference Architecture.
- http://www.sun.com/smi/Press/sunflash/2004-04/sunflash.20040419.5.html
- Dynamic Load balancing on Web-Server Systems

Distributed web server systems, Client based , server architecture based approach.
The bandwidth of the network and its effect on the web traffic.

- http://tele1.dee.fct.unl.pt/td_2002_2003/teo/web_replicated.pdf

- ANALYSIS OF COMPUTER PERFORMANE TE*CHNIQUES*
- http://www.frontrunnercpc.com/info/infomain-fs.htm

Performance Analysis Techniques for the models.
http://www.frontrunnercpc.com/info/techniques.htm

Three Critical Questions.

- For the author the average arrival time of the requests is important as it determines the number of the idle processes and which is the control switch for the configuration parameter in the dynamic size pool behavior of the web server architecture, what is the impact of complexity of the software architecture and the arrival time of the requests?
- Web based services are offered to potentially tens of millions of users by hundreds of thousands of servers. The web servers using the analytic performance models as a guide to dynamically adjust configuration parameters to handle the maximum request with minimum time delay will definitely improve the availability and scalability of the e-business but what is its impact on the cost-effectiveness of the web services?
- Some e-business sites may become popular very quickly. How fast can the site architecture be scaled up? What components of the site should be upgraded? Database server? Web servers? Application servers? Or the network link bandwidth?
- Capacity planning is the process of predicting when future loads levels will saturate the system, and of determining the most cost-effictive way of delaying the system saturation as much as possible. Prediction is key to capacity planning because an organization needs to be able to determine how an e-business site will react when changes in load levels and customer behavior occur, or when new business models are developed. This determination requires predictive models and not the experimentation. Is the analytic performance models are a better fit than the simulation models in capacity planning ?