Citation for the published paper:
M. Andersson, M. Kihl, A. Robertsson, B. Wittenmark, 2004,
"Admission Control of the Apache Web Server",
*Seventeenth Nordic Teletraffic Seminar, NTS 17, Fornebu, Norway,
25-27 August 2004*.
ISBN: 82-423-0595-1. Publisher: Fornebu : Telenor.

Admission Control of the Apache Web Server

M. Andersson, M. Kihl, A. Robertsson and B. Wittenmark.

Abstract

Web sites are exposed to high rates of incoming requests. The servers may become overloaded during temporary traffic peaks when more requests arrive than the server is designed for. An admission control mechanism rejects some requests whenever the arriving traffic is too high and thereby maintains an acceptable load in the system. This paper presents how admission control mechanisms can be designed with a combination of queueing theory and control theory. In this paper we model an Apache web server as a G/G/1-system and then design a PI-controller, commonly used in automatic control, for the server. The controller has been implemented as a module inside the Apache source code. Measurements from the laboratory setup show how robust the implemented controller is, and how it correspond to the results from the theoretical analysis.

I. Introduction

One problem with web servers is that they are sensitive to overload. The servers may become overloaded during temporary traffic peaks when more requests arrive than the server is designed for. Because overload usually occurs rather seldom, it is not economical to overprovision the servers for these traffic peaks, instead admission control mechanisms can be implemented in the servers. The admission control mechanism rejects some requests whenever the arriving traffic is too high and thereby maintains an acceptable load in the system. Traditionally, server utilization or queue lengths have been the variables mostly used in admission control schemes. For web servers, the main objective of the control scheme is to protect it from overload. As long as the average server utilization or queue length is below a certain level, the response times are low. One well-known controller in automatic control is the PID-controller, which enables a stable control for many types of systems (see, for example Åström, [1]). In order to get the system to behave well it is necessary to decide proper control parameters. Therefore, before designing the PID-controller, the system must be analyzed so that its dynamics during overload are known. This means that the system must be described with a control theoretic method. If the model is linear, it is easily analyzed with linear control theoretic methods. However, a queueing system is both nonlinear and stochastic. The main problem is that nonlinear models are much harder to analyze with control theoretic methods. Very few papers have investigated admission control mechanisms for server systems with control theoretic methods. Abdelzaher ([2], [3]) modeled the web server as a static gain to find optimal controller parameters for a PI-controller. A scheduling algorithm for an Apache [4] web server was designed using system identification methods and linear control theory by Lu et al [5]. Bhatti [6] developed a queue length control with

Maria Kihl and Mikael Andersson are with the Department of Communication System, Lund Institute of Technology, Box 118, SE-221 00 Lund, Sweden, {maria|mike}@telecom.lth.se, Anders Robertsson and Björn Wittenmark are with the Department of Automatic Control, Lund Institute of Technology, Box 118, SE-221 00 Lund, Sweden, {andersro|bjorn}@control.lth.se

priorities. By optimizing a reward function, a static control was found by Carlström [7]. An on-off load control mechanism regulating the admittance of client sessions was developed by Cherkasova [8]. Bhoj [10] used a PI-controller in an admission control mechanism for a web server. However, no analysis is presented on how to design the controller parameters. Papers analyzing queueing systems with control theoretic methods usually describe the system with linear deterministic models. Stidham Jr [11] argues that deterministic models cannot be used when analyzing queueing systems. Until now, no papers have designed admission control mechanisms for server systems using nonlinear control theory. In this paper we implement an admission control mechanism for the Apache web server. Measurements in the laboratory setup show how robust the implemented controller is, and that it corresponds to the results from the theoretical analysis. Section 2 shows how this can be applied on a web server. In section 3, we describe a nonlinear control theoretic model of an admission control mechanism for a web server. We give an analysis of the closed loop system in section 4. The control theoretic model is used to design and implement an admission control mechanism for the Apache web server. The measurements are shown in section 5, section 6 discusses the results and section 7 concludes the work.

II. Investigated System

The system we have investigated in this work, is a web server with an admission control mechanism. The web server is Apache, described below.

A. Web servers

A web server like Apache, contains software that offers access to documents stored on the server. Clients can browse the documents in a web browser. The documents can be for example static Hypertext Markup Language (HTML) files, image files or various script files, such as Common Gateway Interface (CGI), Java scripts or Perl files. The communication between clients and server is based on HTTP [12]. An HTTP transaction consists of three steps: TCP connection setup, HTTP layer processing and network processing. The TCP connection setup is performed through a threeway handshake, where the client and the server exchange TCP SYN, TCP SYN/ACK and TCP ACK messages. Once the connection has been established, a document request can be issued with an HTTP GET message to the server. The server then replies with an HTTP GET REPLY message. Finally, the TCP connection is closed by TCP FIN and TCP ACK messages in both directions. Apache, which is a well-known web server and widely used, is multi-threaded. This means that a request is handled by its own thread or process throughout the life cycle of the request.

B. Admission control

Since continuous control is not possible in computer systems, time is divided into control intervals of length h seconds. At the end of interval k, that is when the time is kh, the controller calculates the desired admittance rate for interval $[kh, kh + h]$, denoted $u(kh)$, from the measured average server utilization during the interval, $\rho(kh)$, and the reference value ρ_{ref}. There are three main parts in our admission control architecture, see Figure 1:

Gate. The Gate module gets notified whenever the web server gets an incoming request. It takes a decision whether to admit the request and then notifies the web server of the

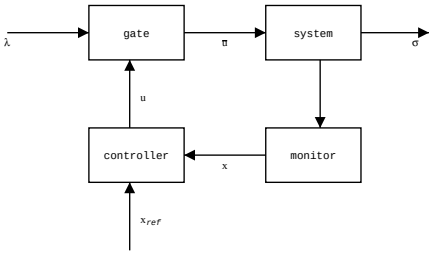

Fig. 1. An admission control mechanism

result. In this paper we use the token bucket algorithm to reject those requests that cannot be admitted. New tokens are generated at a rate of $u(kh)$ tokens per second during time interval $[kh, kh + h]$. If there is an available token upon the arrival of a request, the request consumes the token and enters the web server. If there are no available tokens, the request is rejected. When a request is rejected, the TCP socket to the client is closed. Rejected requests are assumed to leave the system without retrials.

Monitor. The Monitor thread constantly samples the server utilization every control interval. The server utilization is calculated as one minus the fraction of time an idle process has been able to run during the last control interval. The idle process' priority level is set to the lowest possible, which means that it only runs whenever there is no request requiring CPU work. This way of measuring the load on the CPU results in a quantization effect in server utilization. The reason to this is that the operating system where the admission control mechanism runs has a certain time resolution in function calls regarding process uptimes. This means that the control interval cannot be chosen arbitrary. It has to be long enough not to be affected by the time resolution effects, and short enough so that the controller responds quickly.

Controller. The Controller is a PI-controller. The Controller's output is forwarded to the Gate module. The Controller design is discussed more extensively in section 4.

III. Control Theoretic Model

We use the discrete-time control theoretic model of web server developed by Kihl et al. [14]. We assume that the system can be modeled as a GI/G/1-system with an admission control mechanism. Kihl et al. showed that the queueing model is a good model for admission control purposes. The input to the system is the actual admittance rate, $\bar{u}$, whereas the output is the server utilization, ρ. The model is a flow or liquid model in discrete-time. The model is an averaging model in the sense that we are not considering the specific timing of different events, arrivals, or departures from the queue. We assume that the sampling period, h, is sufficiently long to guarantee that the quantization effects around the sampling times are small. The model is shown in Figure 2. The system consists of an arrival generator, a departure generator, a controller, a queue and a monitor.

There are two stochastic traffic generators in the model. The *arrival generator* feeds the system with new requests. The number of new requests during interval kh is denoted

$\alpha(kh)$. $\alpha(kh)$ is an integrated stochastic process over one sampling period with a distribution obtained from the underlying interarrival time distribution. If, for example, the arrival process is Poisson with mean λ, then $\alpha(kh)$ is Poisson distributed with mean λh. The *departure generator* decides the maximum number of departures during interval kh, denoted $\sigma_{max}(kh)$. $\sigma_{max}(kh)$ is also a stochastic process with a distribution given by the underlying service time distribution. If, for example, the service times are exponentially distributed with mean $1/\mu$, then $\sigma_{max}(kh)$ is Poisson distributed with mean μh. It is assumed that $\alpha(kh)$ and $\sigma_{max}(kh)$ are independent from between sampling instants and uncorrelated to each other. The *gate* is constructed as a saturation block that limits $u(kh)$ to be

$$\bar{u}(kh) = \begin{cases} 0 & u(kh) < 0 \\ u(kh) & 0 \leq u(kh) \leq \alpha(kh) \\ \alpha(kh) & u(kh) > \alpha(kh) \end{cases}$$

The *queue* is represented by its state $x(kh)$, which corresponds to the number of requests in the system at the end of interval kh. The difference equation for the queue is given by

$$x(kh+h) = f(x(kh) + \bar{u}(kh) - \sigma_{max}(kh))$$

where the limit function, $f(w)$, equals zero if w <0 and w otherwise. The limit function assures that $x(kh+h) \geq 0$. When the limit function is disregarded then the queue is a discrete-time integrator.

The *monitor* must estimate the server utilization since this is not directly measurable in the model. The server utilization during interval kh, $\rho(kh)$, is estimated as

$$\rho(kh) = min(\frac{\bar{u}(kh) + x(kh)}{\sigma_{max}(kh)}, 1)$$

The objective of the *controller* is to minimize the difference between the server utilization during interval kh, $\rho(kh)$, and the reference value, ρ_{ref}. The control law is given by the transfer function, $Gc(z)$.

IV. Stability analysis of closed loop system

In this section we will consider the stability properties of the controlled server node, when using a PI-controller for admission control. First we will consider an approach based on a linear queue model and compare with the admission control parameters derived from nonlinear analysis. The analysis is based on the Tsypkin/Jury-Lee stability criterion (discrete-time versions of the Popov criterion) [15]. In the analysis only the dominating 'queue-limitation' φ will be considered. See Section VI for comments on the saturation.

A. Linear design (neglecting saturations)

Neglecting the nonlinearities in Figure 2 (assuming $\varphi(z) = z$, i.e., linear and no saturation) and using a standard PI-controller $G_c(z) = K(1 + \frac{1}{T_i} \cdot \frac{h}{z-1})$ will result in the closed loop dynamics

$$\begin{aligned} G_c &= \frac{G_c(1+G_q)G_m}{1 + G_c(1+G_q)G_m} \\ &= \frac{z \cdot K/\sigma\,(z - 1 + h/T_i)}{z \cdot (z^2 + (K/\sigma - 2)z + (1 - K/\sigma + Kh/(\sigma T_i))} \end{aligned} \tag{1}$$

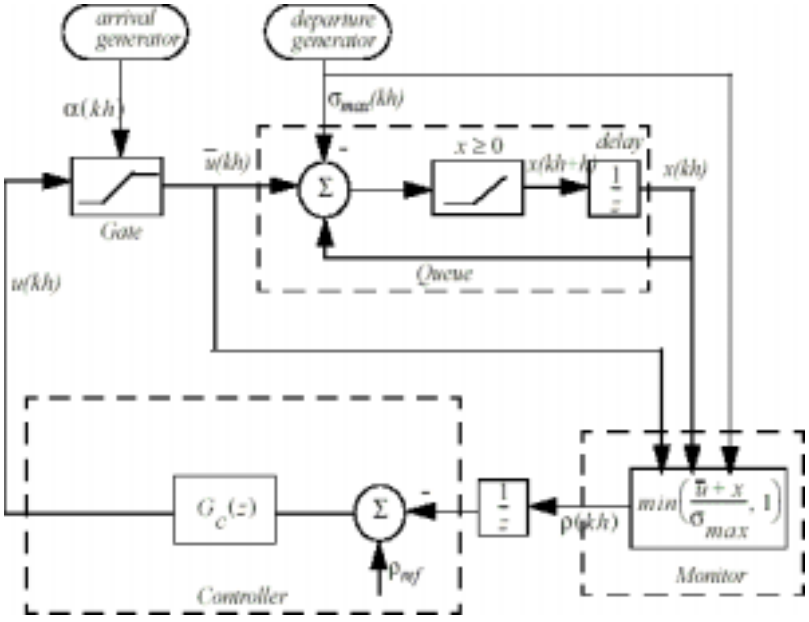

Fig. 2. Discrete-time model with controller saturation and saturation φ for positive queue lengths.

where G_q and G_m represent the queue and monitor dynamics, respectively. To match the characteristic polynomial

$$z \cdot (z^2 + (K/\sigma - 2)z + (1 - K/\sigma + Kh/(\sigma T_i)) \tag{2}$$

with a desired characteristic polynomial

$$z \cdot (z^2 + a_1 z + a_2) \tag{3}$$

we get the control parameters

$$K = (2 + a_1)\,\sigma, \quad T_i = h\,(2 + a_1)/(1 + a_1 + a_2)$$

Using the parameters of the PI-controller it is thus possible to make an arbitrary pole-placement, except for the pole $z = 0$, which corresponds to a time delay. A simplified linear analysis will thus predict stability for the closed loop for all coefficients $\{a_1,\, a_2\}$ belonging to the stability triangle

$$\{\; a_2 < 1, \quad a_2 > 1 + a_1, \quad a_2 > 1 - a_1 \;\}, \tag{4}$$

see [1].

B. Model with queue limitation

Consider the admission control scheme in Figure 3 where we have introduced the states $\{x_1,\, x_2,\, x_3\}$ corresponding to the queue length, the (delayed) utilization ρ and the integrator state in the PI-controller, respectively.

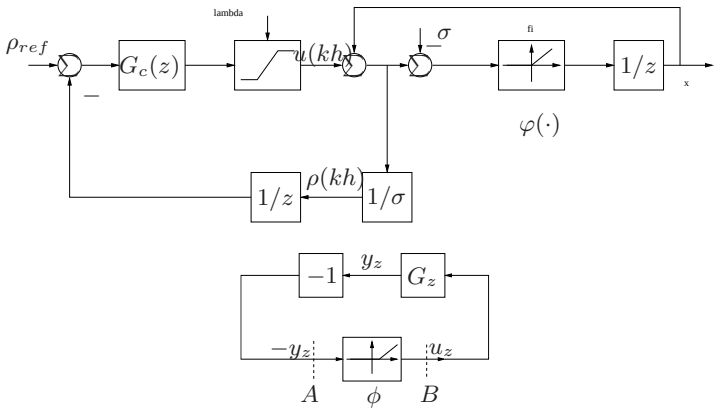

Fig. 3. Decomposition into a linear block (G_z) and a nonlinear block (ϕ) under negative feedback.

The state space model will be

$$\begin{aligned} x_1(kh+h) &= \varphi\left(u + x_1(kh) - \sigma\right) \\ x_2(kh+h) &= \frac{1}{\sigma}(u + x_1(kh)) \\ x_3(kh+h) &= Kh/T_i(\rho_{ref} - x_2(kh)) + x_3(kh) \end{aligned} \tag{5}$$

where $u = K(\rho_{ref} - x_2) + x_3$ and $\varphi(\cdot)$ is the saturation function in Figure 3. By introducing the *forward shift operator* and leaving out the time arguments, we get

$$q\,x_1 = \varphi\left(K(\rho_{ref} - x_2) + x_3 + x_1 - \sigma\right) \tag{6}$$

$$q\,x_2 = \frac{1}{\sigma}\left(K(\rho_{ref} - x_2) + x_3 + x_1\right) \tag{7}$$

$$q\,x_3 = Kh/T_i\left(\rho_{ref} - x_2\right) + x_3 \tag{8}$$

The equilibrium for the system (6–8) satisfies $qx = x$.
From (8) we get

$$x_3 = Kh/T_i\left(\rho_{ref} - x_2\right) + x_3 \quad \Rightarrow \quad x_2^o = \rho_{ref}$$

Inserting this in (6) and (7) we get

$$\begin{aligned} x_1^o &= \varphi(x_3^o + x_1^o - \sigma) \\ x_2^o &= \rho_{ref} = \frac{1}{\sigma}\left(x_3^o + x_1^o\right) \\ &\Rightarrow \\ x_1^o &= \varphi\left(\sigma(\rho_{ref} - 1)\right) \end{aligned} \tag{9}$$

As $\rho_{ref} \in [0, 1]$ and using the fact that $\varphi(z) = 0$, $\forall z \leq 0$ we get

$$\begin{cases} x_1^o = 0 \\ x_2^o = \rho_{ref} \\ x_3^o = \sigma x_2^o = \sigma\rho_{ref} \end{cases} \tag{10}$$

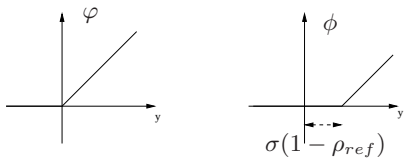

Fig. 4. $\phi(y) = \varphi(y - \sigma(1 - \rho_{ref}))$ where $\sigma > 0$ and $\rho_{ref} \in [0, 1]$.

By introducing the change of variables

$$\begin{cases} z_1 = x_1 - 0 & \text{or} \quad x_1 = z_1 \\ z_2 = x_2 - \rho_{ref} & \text{or} \quad x_2 = z_2 + \rho_{ref} \\ z_3 = x_3 - \sigma\rho_{ref} & \text{or} \quad x_3 = z_3 + \sigma\rho_{ref} \end{cases}$$

we get

$$\begin{aligned} q\, z_1 = q\, x_1 - 0 \quad &= \varphi\left(-K z_2 + z_3 + z_1 - \sigma\right) \\ q\, z_2 = q\, x_2 - \rho_{ref} \quad &= \frac{1}{\sigma}\left(-K z_2 + z_3 + \sigma\rho_{ref} + z_1\right) - \rho_{ref} \\ q\, z_3 = q\, x_3 - \sigma\rho_{ref} \quad &= -\, Kh/T_i\, z_2 + z_3 + \sigma\rho_{ref} - \sigma\rho_{ref} \end{aligned}$$

Rewriting this as a linear system in negative feedback with the nonlinear function $\phi : y \to \varphi(y - \sigma(1 - \rho_{ref}))$, we get

$$\begin{aligned} qz &= A_z z + B_z u_z = A_z z + B_z \phi(-y) \\ y &= C_z z \end{aligned}$$

$$q \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1/\sigma & -K/\sigma & 1/\sigma \\ 0 & -Kh/T_i & 1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \phi(-y)$$

$$y = \begin{bmatrix} -1 & K & -1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}$$

Note that for $\rho_{ref} \in [0, 1]$ the function $\phi(\cdot)$ will belong to the same cone as $\varphi(\cdot)$, namely $[\alpha,\, \beta] = [0, 1]$, see Figure 4. The incremental variation will also have the same maximal value (=1).

The transfer function $G_z = G_{u_z \to y_z}(z)$ from cut B to cut A in Figure 3 will be

$$\begin{aligned} G_z &= C_z (zI - A_z)^{-1} B_z \\ &= \frac{-z \cdot (z - 1)}{z \cdot \left(z^2 + (-1 + K/\sigma)\, z + K\left(h - T_i\right)/(\sigma T_i)\right)} \end{aligned} \tag{11}$$

For the forthcoming stability analysis we determine for which control parameters the linear subsystem G_z is stable.

The poles of (11) are stable for the area depicted in Figure 5 for the normalized parameters K/σ and h/T_i.

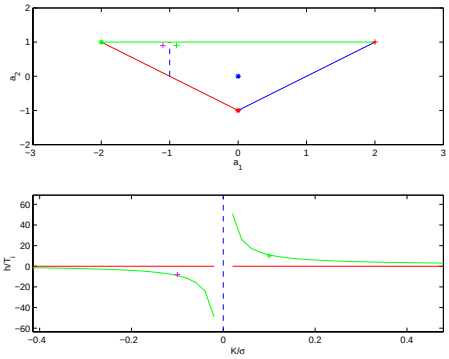

Fig. 5. *(Upper:)* The internal of the triangle corresponds to the stability area for the characteristic polynomial $z \cdot (z^2 + a_1 z + a_2)$ of G_z. *(Lower:)* Corresponding stabilizing control parameters $\{K/\sigma,\ h/T_i\}$.

C. Stability analysis for discrete-time nonlinear system

To determine the stability for the nonlinear system in (11) we can use the Tsypkin criterion or the Jury-Lee criterion which are the discrete-time counterparts of the Popov criterion for continuous time systems [16].

Sufficient conditions for stability are that G_z has all its poles within the unit circle $|z| < 1$ and that there exists a (positive) constant η such that

$$\mathrm{Re}[(1 + \eta(1 - z^{-1}))G_z(z)] + \frac{1}{k} \geq 0 \text{ for } z = e^{i\omega},\ \omega \geq 0 \tag{12}$$

where the nonlinearity ϕ belongs to the cone $[0, k = 1]$.

In the upper left plot of Figure 6 we have *the stability triangle* for the characteristic polynomial of Eq.(2). By choosing coefficients for the characteristic polynomial (2) in the upper left triangle (A1) we will get controller parameters $\{K,\ T_i\}$ which also will give a stable transfer function G_z. The corresponding poles are plotted in the lower diagram of Figure 6. Figure 6 shows a graphical representation of the Tsypkin condition (12) for this set of control parameters. The dashed non-intersecting line in Figure 6 corresponds to the existence of a positive parameter η satisfying Eq.(12). Thus, absolute stability for the nonlinear system also is guaranteed for this choice of parameters.

Remark: The Tsypkin criterion guarantees stability for any cone bounded nonlinearity in $[0, 1]$ and we can thus expect to have some robustness in addition to stability in our case.

V. Experiments

The admission control mechanism was implemented in the Apache web server. Apache is made up of a core package and several modules that handle different operations, such as Common Gateway Interface (CGI) execution, logging, caching etc. A new module was created that contains the admission control mechanisms. The new module was then hooked into the core of Apache, so that it was called every time a request was made to the web server. The module could then either reject or admit the request according to the control mechanism. The admission control mechanism was written in C and tested on a Windows platform. We tested the system by running tests on it and collecting performance

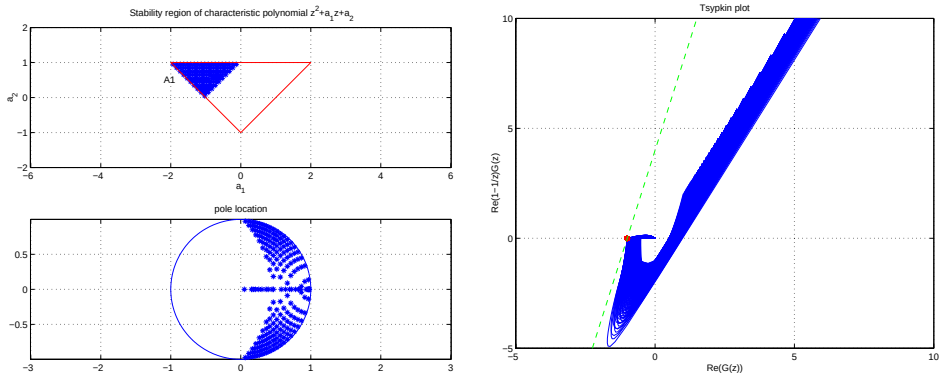

Fig. 6. *(Upper left:)* The large triangle is the stability area a linear model would predict. However, from this parameter set, nonlinear analysis guarantees only stability for parameters $\{a_1, a_2\}$ in the upper left triangle (A_1, $'*'$). *(Lower left:)* Pole location corresponding to $\{a_1, a_2\} \in A1$. *(Right:)* Set of Tsypkin plots which all satisfy the frequency condition (12) for $G_z = G_z(K, T_i)$, where (K, T_i) correspond to pole locations to the left

metrics such as the server utilization distribution and step responses. We also compared the measurements with simulations. The queueing model was represented by a discrete-event simulation program implemented in C, and the control theoretic models were implemented with the Matlab Simulink package. The traffic generators in the discrete-time model were built as Matlab programs. They generate arrivals and departures according to the given statistical distributions.

A. Setup

Our measurements used one server computer and one computer representing the clients connected through a 100 Mbits/s Ethernet switch. The server was a PC Pentium III 1700 MHz with 512 MB RAM running Windows 2000 as operating system. The computer representing the clients was a PC Pentium II 400 MHz with 256 MB RAM running RedHat Linux 7.3. Apache 2.0.45 was installed in the server. We used the default configuration of Apache. The client computer was installed with an HTTP load generator, which was a modified version of S-Client [17]. We modified the S-Client code to use Poissonian arrivals instead of the original deterministic ones. The client program was programmed to request dynamically generated HTML files from the server. The CGI script was written in Perl. It generates a number of random numbers, adds them together and returns the summation. The average request rate was set to 100 requests per second in all experiments except for the measurements in Figure 7. In all experiments, the control interval was set to one second.

B. Validation of the Model

We have validated that the open system, that is without control feedback, is accurate in terms of average server utilization. The average server utilization for varying arrival rates are shown in Figure 7. For a single-server queue, the server utilization is proportional to the arrival rate, and the slope of the server utilization curve is given by the average service time. The measurements in Figure 7 gives an estimation of the average service time in the web server, $1/\mu$=0.0225.

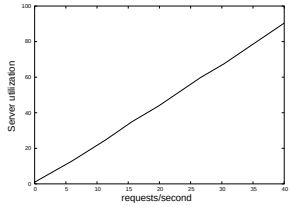

Fig. 7. Average server utilization for the open system.

C. Controller parameters

Control parameters for the PI-controller are chosen from the stability area A1 in Figure 6. In the simulations and experiments below we use $\{K, T_i\}$={20, 2.8}. The parameter setting is also compared to $\{K, T_i\}$={20, 0.1}, found outside the stability area.

D. Performance metrics

An admission control mechanism has two control objectives. First, it should keep the control variable at a reference value, i.e. the error, $e = y_{ref} - y$, should be as small as possible. Second, it should react rapidly to changes in the system, i.e. the so-called settling time should be short. Therefore, we test the mechanism in two ways. First, we show the steady-state distribution of the control variable, by plottting the estimated distribution function. The distribution function is estimated from measurements during 1000 seconds with the specific parameter setting. Second, we plot the step response during 60 seconds when starting with an empty system.

E. Distribution function

Figure 8 shows the estimated distribution function for the PI-controller. Both good and bad parameter settings were used. An ideal admission control mechanism would show a distribution function that is zero until the wanted load, and is one thereafter. In this case, the load was kept at 0.8, and the parameter setting, {K, Ti}={20, 2.8}, chosen from results in a controller that behaves very well in this sense. The parameter setting, $\{K, T_i\}$={20, 0.1}, as can be seen, perform worse. Also, as comparison, results from simulations of the M/D/1 system and the M/M/1 system are given in Figure 8, when using $\{K, Ti\}$={20, 2.8}.

F. Step response

Figure 9 shows the behaviour of the web server during the transient period. The measurements were made on an empty system that was exposed to 100 requests per second. The good parameter setting, $\{K, Ti\}$={20, 2.8}, exhibits a short settling time with a relatively steady server utilization. The bad parameter setting, $\{K, Ti\}$={20, 0.1} has its poles outside the unit circle and behaves badly, the load oscillates and is never stable. Comparisons to M/D/1 and M/M/1 simulations, also in Figure 9, show that the model is accurate.

VI. Discussion

The analysis in Section IV-C gives sufficient conditions and a region for control parameters which guarantee stability of the nonlinear closed loop as well as for the simplified linear model. We are of course not restricted to choose parameters from only this region as the main objective is that the nonlinear system should be stable. However, we can conclude that

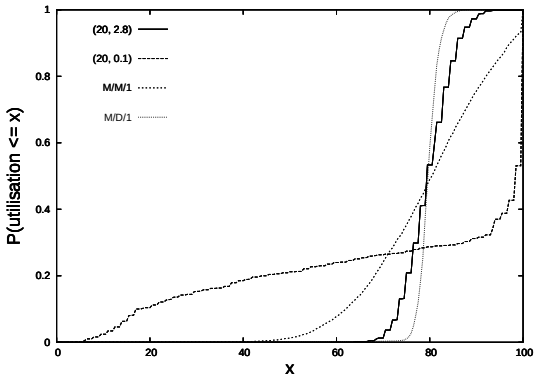

Fig. 8. Server utilization distribution of measurements from the real system together with simulations of the M/M/1 and the M/D/1 system.

- Pole-placement based on a linear model is OK in a restricted area (region $A1$ in Figure 6).
- There are choices of parameters that gives stable closed loop poles, but where the linear analysis would indicate an unstable closed loop systems.

During simulation studies the dominant nonlinear effect has come from the queue nonlinearity φ. The saturation due to limited arrival rate can be handled with a standard implementation of an anti-reset windup scheme, see [18].

VII. Conclusion

Traditionally, queuing theory has been used when investigating server systems. However, within queuing theory there are few mathematical tools for design and stability analysis of, for instance, admission control mechanisms. Therefore, these mechanisms have mostly been developed with empirical methods. In this paper, we have designed load control mechanisms for a web server system with control theoretic methods and analyzed its stability properties. The controller structure considered is a PI-controller and a region for stabilizing control parameters is presented.

The designs have been experimentally verified with simulations and experiments on an Apache web server system.

VIII. Acknowledgments

This work has partially been supported by the Swedish Research Council through the Multi Project Grant 621-2001-3020 and contract 621-2001-3053.

References

[1] K. Åström and B. Wittenmark, *Computer-controlled systems, theory and design*. Prentice Hall Internation Editions, 1997, third Edition.

[2] T. Abdelzaher and C. Lu, "Modeling and performance control of internet servers," in *Proceedings of the 39th IEEE Conference on Decision and Control*, 2000, pp. 2234–2239.

[3] K. S. T.F. Abdelzaher and N. Bhatti, "Performance guarantees for web server end-systems: a control theoretic approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 1, pp. 80–96, January 2002.

[4] "Apache web server," http://www.apache.org.

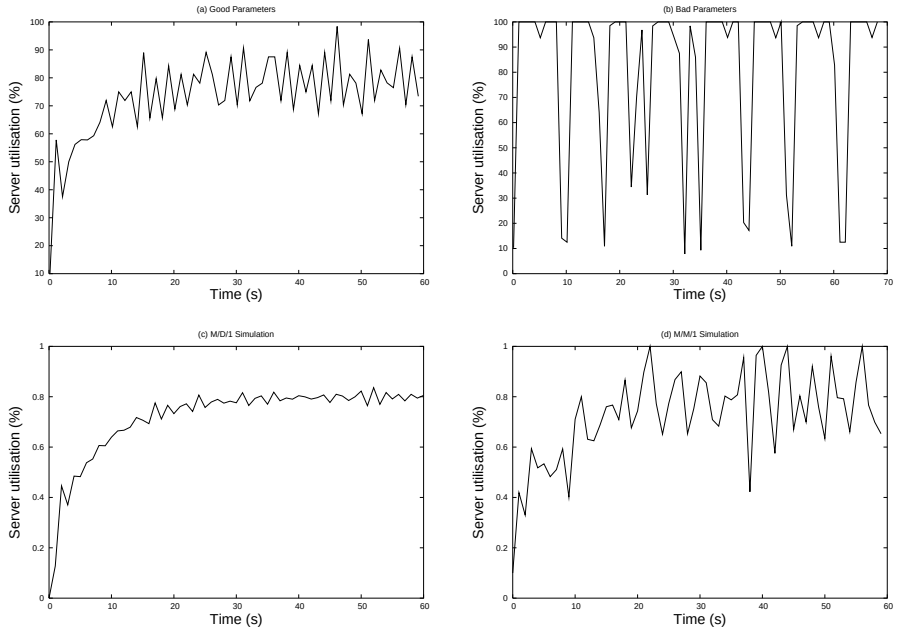

Fig. 9. (a) Example of a realisation with good parameters. (b) Example of a realisation with bad parameters. (c) Simulation of M/D/1-system with good parameters. (d) Simulation of M/M/1-system with good parameters.

[5] J. S. C. Lu, T.F. Abdelzaher and S. So, "A feedback control approach for guaranteeing relative delays in web servers," in *Proceedings of the 7th IEEE Real-Time Technology and Applications Symposium*, 2001, pp. 51–62.

[6] N. Bhatti and R. Friedrich, "Web server support for tiered services," *IEEE Network*, pp. 64–71, Sept/Oct 1999.

[7] R. R. J. Carlström, "Application-aware admission control and scheduling in web servers," in *Proc. Infocom*, 2002.

[8] P. P. L. Cherkasova, "Predictive admission control strategy for overloaded commerical web servers," in *Proc. 8th International IEEE Symposium on modeling, analysis and simulation of computer and telecommunication systems*, 2000, pp. 500–507.

[9] P. G. T. Voigt, "Adaptive resource-based web server admission control," in *Proc. 7th International Symposium on Computers and Communications*, 2002.

[10] S. R. P. Bhoj and S. Singhal, "Web2k: Bringing qos to web servers," *HP Labs Technical report, HPL-2000-61*, 2000.

[11] S. S. Jr., "Optimal control of admission to a queueing system," *IEEE Transactions on Automatic Control*, vol. 30, no. 8, pp. 705–713, August 1985.

[12] W. Stallings, *Data & Computer Communications*. Prentice Hall, 2000, sixth Edition.

[13] T. Voigt, "Overload behaviour and protection of event-driven web servers," in *In proceedings of the International Workshop on Web Engineering*, May 2002, pisa, Italy.

[14] B. W. M. Kihl, A. Robertsson, "Performance modelling and control of server systems using non-linear control theory," in *Proc. 18th International Teletraffic Congress*, 2003.

[15] Y. Z. Tsypkin, "Frequency criteria for the absolute stability of nonlinear sampled-data systems," *Automation and Remote Control*, vol. 25, no. 3, pp. 261–267, 1964.

[16] M. Larsen and P. V. Kokotovic, "A brief look at the tsypkin criterion: from analysis to design," *Int. J. of Adaptive Control and Signal Processing*, vol. 15, no. 2, pp. 121–128, 2001.

[17] G. Banga and P. Druschel, "Measuring the capacity of a web server," in *USENIX Symposium on Internet Technologies and Systems*, December 1997, pp. 61–71.

[18] M. K. A. Robertsson, B. Wittenmark, "Analysis and design of admission control in web-server systems," in *Proc. of ACC03*, 2003.