# Modeling and Predicting End-to-End Response Times in Multi-tier Internet Applications

Sandjai Bhulai, Swaminathan Sivasubramanian, Rob van der Mei, and Maarten van Steen

Vrije Universiteit Amsterdam
Faculty of Sciences
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
{sbhulai,swami,mei,steen}@few.vu.nl

**Abstract.** Many Internet applications employ multi-tier software architectures. The performance of such multi-tier Internet applications is typically measured by the end-to-end response times. Most of the earlier works in modeling the response times of such systems have limited their study to modeling the mean. However, since the user-perceived performance is highly influenced by the variability in response times, the variance of the response times is important as well.

We first develop a simple model for the end-to-end response times for multi-tiered Internet applications. We validate the model by real data from two large-scale applications that are widely deployed on the Internet. Second, we derive exact and approximate expressions for the mean and the variance, respectively, of the end-to-end response times. Extensive numerical validation shows that the approximations match very well with simulations. These observations make the results presented highly useful for capacity planning and performance prediction of large-scale multi-tiered Internet applications.

**Keywords:** end-to-end response times, Internet services, multi-tier architectures, variance approximation.

## 1 Introduction

The past few years have witnessed a tremendous growth in Internet services and applications. Modern Web sites such as amazon.com, yahoo.com, and ebay.com do not simply deliver static pages but generate content on the fly using multi-tiered applications, so that the pages can be customized for each user. For example, a single request to amazon.com's home page is served by hundreds of internal applications [1]. These enterprises build their software systems out of many Web applications, usually known as services. Typically, such a service performs certain business logic that generates queries to its associated database and requests to other services. They are usually exposed through well-defined client interfaces accessible over the network. Examples of services include online

L. Mason, T. Drwiega, and J. Yan (Eds.): ITC 2007, LNCS 4516, pp. 519–532, 2007.
© Springer-Verlag Berlin Heidelberg 2007

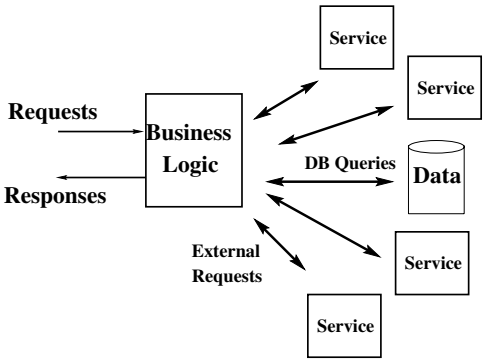



**Fig. 1.** Application Model of an Internet Service

entertainment services, order processing services, PC banking, online shopping. The application model of one typical service is shown in Figure 1.

For online services, providing a good client experience is one of the primary concerns. Human computer interaction studies have shown that frequent users prefer response times of less than a second for most tasks, and that human productivity improves more than linearly as computer system response times fall in the sub-second range [2]. Hence, for such online services, providing low response times to their clients is a crucial business requirement.

A key factor for providing good client response times is the ability to predict and control the performance in terms of the end-to-end response times. For the user-perceived quality, both the mean and the variance of the response times are key metrics. Therefore, we first focus on deriving a simple analytical model for such multi-tier Internet services. Second, with the use of such a model we aim to estimate the end-to-end response time of a service and also its variance. Such a model is important for several reasons. First, it allows service administrators to provision their system with enough resources at the right tiers so that the service response times are within the expected limits. In addition, it allows them to employ appropriate request policing so that excess requests can be rejected during overload scenarios. Finally, it allows service designers to predict the performance of their applications for different load conditions.

In the past few years, several groups have looked at modeling Internet applications. Most of them focus on modeling single-tier applications such as Web servers [3,4,5,6] and databases [7]. However, very few have studied models for multi-tier applications, like those given in Figure 1, which are more commonplace in the Web. Applying simple single-tier models to multi-tiered systems leads to poor solutions as different tiers have different characteristics. Therefore, there is a need to model multi-tier systems directly such that performance analysis is tractable.

Closest to our approach is recent work by Urgaonkar et al. [8]. They use mean-value analysis based techniques for estimating the mean response time of applications modeled as a network of queues. However, they cannot provide any

bounds on variances. In practice, most e-commerce organizations measure client experience based on percentiles and not on means [1]. Hence, obtaining such bounds, even if approximate, is highly beneficial.

Our contribution in this paper is that we model a multi-tiered Internet service within a queueing-theoretical framework. Our model consists of $N+1$ nodes with a single entry node which receives the requests and sends requests to the other $N$ nodes in a deterministic order. The request arrival distribution is assumed to be a Poisson process, which is usually true for arrivals from a large number of independent sources and shown to be realistic for Internet systems [9]. The entry node can be equated to the business logic (shown in Figure 1) that receives the service requests, and the service nodes correspond to databases/other services that are queried upon for serving the request. Secondly, using such a model, we derive expressions for the mean end-to-end response time and an approximation to its variance. We show the effectiveness of our variance approximations in two steps. In the first step, we compare the analytical results with simulations. In the second step, we validate our complete model using real Web services for different numbers of tiers. In our experiments, we find that our model is very effective in predicting the mean end-to-end response time and its variances (with an error margin less than 10%).

The rest of the paper is organized as follows. Section 2 presents the related work and Section 3 presents the system model. Section 4 presents the estimations for the mean response time and its variances. Section 5 validates the approximations we derived for variances using numerical simulations. Section 6 presents the validation experiments, where we validate the model with two Web service applications and Section 7 concludes the paper.

## 2 Related Work

### 2.1 Modeling Internet Systems

Various groups in the past have studied the problem of modeling Internet systems. Typical models include modeling Web servers, database servers, and application servers [3,4,7,5]. For example, in [3], the authors use a queueing model for predicting the performance of Web servers by explicitly modeling the CPU, memory, and disk bandwidth in addition to using the distribution of file popularity. Bennani and Menasce [10] present an algorithm for allocating resources to a single-tiered application by using simple analytical models. Villela et al. [9] use an M/G/1/PS queueing model for business logic tiers and to provision the capacity of application servers. A G/G/1 based queueing model for modeling replicated Web servers is proposed in [11], which is to perform admission control during overloads. In contrast to these queueing-based approaches, a feedback control based model was proposed in [6]. In this work, the authors demonstrate that by determining the right handles for sensors and actuators, the Web servers can provide stable performance and be resilient to unpredictable traffic. A novel approach to modeling and predicting the performance of a database is proposed in [7]. In this work, the authors employ machine learning and use an $K$-nearest

neighbor algorithm to predict the performance of database servers during different workloads. However, the algorithm requires substantial input during the training period to perform effective prediction.

All the aforementioned research efforts have been applied only to single-tiered applications (Web servers, databases, or batch applications) and do not study complex multi-tiered applications which is the focus of this paper. Some recent works have focused on modeling multi-tier systems. In [5], the authors model a multi-tiered application as a single queue to predict the performance of a 3-tiered Web site. As mentioned, Urgaonkar et al. [8] model multi-tier applications as a network of queues and assume the request flows between queues to be independent. This assumption enables them to assume a product-form network so that they can apply a mean value analysis (MVA) to obtain the mean response time to process a request in the system. Although this approach can be very effective, MVA approaches can be limiting in nature as they do not allow us to get variances which are also of crucial importance in large scale enterprises [1].

### 2.2 Performance Analysis

There are few works that study the performance of Internet systems in the context of multi-tier applications. Although the response time was made explicit for the first time in [12], a lot of research on response times has already been done in other systems. Results for the business logic, modeled as a processor sharing (PS) node, are given in [13], where the Laplace-Stieltjes Transform (LST) is obtained for the M/M/1/PS node. In [14] an integral representation for this distribution is derived, and in [15] the distribution is derived for the more general M/G/1/PS system (in case a representation of the service times is given). The individual services, behind the business logic, are usually modeled as first-come-first-served (FCFS) queueing systems for which results are given in [16].

The first important results for calculating response times in a queueing network are given in [17], in which product-form networks are introduced. A multi-tier system modeled as a queueing network is of product-form when the following three conditions are met. First, the arrival process is a Poisson process and the arrival rate is independent of the number of requests in the network. Second, the duration of the services (behind the business logic) should be exponentially distributed, and is not allowed to depend on the number of requests present at that service. Finally, the sequence in which the services are visited is not allowed to depend on the state of the system except for the state of the node at which the request resides. Multi-tier systems that satisfy these properties fall within the class of so-called Jackson networks and have nice properties.

In [18], the authors give an overview of results on response times in queueing networks. In particular, they give expressions for the LST of the joint probability distribution for nodes that are traversed by requests in a product-form network according to a pre-defined path. Response times in a two-node network with feedback (such as at the business logic) were studied in [19]. The authors propose some solid approximations for the response times with iterative requests. They show that the approximations perform very well for the first moment of the

response times. In [20], a single PS node is studied with several multi-server FCFS nodes. The authors derive exact results for the mean response time as well as estimates for the variance. The performance analysis in this paper is an extension of their work.

## 3 System Model

In this section we develop a model for multi-tier Internet services in the context of a queueing-theoretical framework. For this purpose, consider a queueing network with $N+1$ nodes. Requests, that are initiated by an end-user, arrive according to a Poisson process with rate $\lambda$ to a dedicated entry node. This node can be identified with the business logic depicted in Figure 1. The other $N$ nodes in the queueing network represent the existing basic services delivered by the service provider.

A request traverses the network by first visiting the entry node. This node serves requests in a processor-sharing fashion with service durations that are drawn from a general probability distribution having a mean service time of $\beta_{\text{ps}}$ time units. After service completion, the request is routed to each of the $N$ service nodes in sequence. At service node $i$, requests receive service with a duration that is exponentially distributed with a mean of $\beta_{\text{fcfs},i}$ time units. Requests are served on a first-come first-served (FCFS) basis by one of $c_i$ servers dedicated to node $i$. When no idle servers are available upon arrival, the request joins an infinite buffer in front of node $i$, and waits for its turn to receive service. After having received service at node $i$, the results are sent back for processing to the entry node (which takes place with the same parameters as upon first entry). Thus, every request visits the entry node $N+1$ times, and finally leaves the system after having visited all $N$ service nodes.

The mean response time of the service is modeled as the sojourn time of the request in the system. Let $S_{\text{ps}}^{(k)}$ and $S_{\text{fcfs},i}$ be the sojourn times of the $k$-th visit to the entry node and the sojourn time of the visit to service node $i$, respectively. Then the expected sojourn time $\mathbb{E}S$ of an arbitrary request arriving to the system is given by

$$\mathbb{E}S = \mathbb{E}\Bigg[\sum_{k=1}^{N+1} S_{\text{ps}}^{(k)} + \sum_{i=1}^{N} S_{\text{fcfs},i}\Bigg].$$

Note that the system is modeled such that it satisfies the conditions of a product-form network. First, the arrivals occur according to a Poisson process with a rate that is not state dependent. This is not unrealistic in practice, since arrivals from a large number of independent sources do satisfy the properties of a Poisson process. Second, the service times follow an exponential distribution which do not depend on the number of requests at that node. Even though this modeling assumption may seem unrealistic, in our validation results we will show that it describes the performance of real systems very well. Finally, since the sequence in which the service nodes are visited is fixed, and thus does not

depend on the state of the system, the network is of product-form. Also note that when $c_i$ is large (i.e., hardly any queueing occurs at the basic services) the basic services resemble $\cdot/\mathrm{G}/\infty$ queues for which insensitivity with respect to the service times is known to hold. In cases where queueing is significant, we loose the product form which results in more complex models (see [19]).

## 4 The Mean Response Time and Its Variance

In the previous section we have seen that the queueing network is of product-form. Consequently, when $L_{\mathrm{ps}}$ and $L_{\mathrm{fcfs,i}}$ denote the stationary number of requests at the entry node and service node $i$ for $i = 1, \ldots, N$, respectively, we have

$$\mathbb{P}(L_{\mathrm{ps}} = l_0; L_{\mathrm{fcfs},1} = l_1, \ldots, L_{\mathrm{fcfs,N}} = l_N) = \mathbb{P}(L_{\mathrm{ps}} = l_0) \prod_{i=1}^{N} \mathbb{P}(L_{\mathrm{fcfs,i}} = l_i),$$

with $l_i = 0, 1, \ldots$ for $i = 0, \ldots, N$. From this expression, the expected sojourn time at the entry node and the service nodes can be determined. First, define the load on the entry node by $\rho_{\mathrm{ps}} = (N+1)\lambda\beta_{\mathrm{ps}}$ and the load on service node $i$ by $\rho_{\mathrm{fcfs},i} = [\lambda \cdot \beta_{\mathrm{fcfs},i}]/c_i$. Then, by Little's Law, the sojourn time at the entry node for the $k$-th visit is given by

$$\mathbb{E}S_{\mathrm{ps}}^{(k)} = \frac{\beta_{\mathrm{ps}}}{1 - \rho_{\mathrm{ps}}},$$

for $k = 1, \ldots, N+1$. The expected sojourn time at service node $i$ is given by $\beta_{\mathrm{fcfs},i}$ if upon arrival the request finds a server idle. However, when no idle server is available, the request also has to wait for the requests in front of him to be served. The probability $\pi_i$ that this occurs, can be calculated by modeling the service node $i$ as a birth-death process with birth rate $\lambda$ and death rate $\min\{c_i, l_i\}/\beta_i$ when $l_i$ requests are present at the node and in the queue. From the equilibrium equations $\lambda\mathbb{P}(L_{\mathrm{fcfs},i} = l_i - 1) = \min\{c_i, l_i\}\mathbb{P}(L_{\mathrm{fcfs},i} = l_i)/\beta_i$, the probability of delay is given by

$$\pi_i = \frac{(c_i \cdot \rho_{\mathrm{fcfs},i})^{c_i}}{c_i!} \left[ (1 - \rho_{\mathrm{fcfs},i}) \sum_{l=0}^{c_i - 1} \frac{(c_i \cdot \rho_{\mathrm{fcfs},i})^l}{l!} + \frac{(c_i \cdot \rho_{\mathrm{fcfs},i})^{c_i}}{c_i!} \right]^{-1}.$$

Given that a request has to wait, the expected waiting time is equal to the expected waiting time in an FCFS queueing system with one server and service time $\beta_i/c_i$. This expression is given by $\beta_{\mathrm{fcfs},i}/(1 - \rho_{\mathrm{fcfs},i})c_i$. Finally, combining all the expressions for the expected sojourn time at each node in the network, we derive that the expected response time is given by

$$\begin{aligned} \mathbb{E}S &= \mathbb{E}\left[ \sum_{k=1}^{N+1} S_{\mathrm{ps}}^{(k)} + \sum_{i=1}^{N} S_{\mathrm{fcfs},i} \right] = (N+1)\mathbb{E}S_{\mathrm{ps}}^{(1)} + \sum_{i=1}^{N} \mathbb{E}S_{\mathrm{fcfs},i}. \\ &= \frac{(N+1)\beta_{\mathrm{ps}}}{1 - \rho_{\mathrm{ps}}} + \sum_{i=1}^{N} \left[ \frac{\beta_{\mathrm{fcfs},i}}{(1 - \rho_{\mathrm{fcfs},i}) \cdot c_i} \pi_i + \beta_{\mathrm{fcfs},i} \right]. \end{aligned}$$

Let us now focus our attention to the variance of the sojourn time. It is notoriously hard to obtain exact results for the variance. Therefore, we approximate the total sojourn time of a request at the entry node by the sum of $N+1$ independent identically distributed sojourn times. Moreover, we approximate the variance by imposing the assumption that the sojourn times at the entry node and the sojourn times at the service nodes are uncorrelated. In that case, we have

$$\mathbb{V}\text{ar } S = \mathbb{V}\text{ar } \left[ \sum_{k=1}^{N+1} S_{\text{ps}}^{(k)} + \sum_{i=1}^{N} S_{\text{fcfs},i} \right] = \mathbb{V}\text{ar } \left[ \sum_{k=1}^{N+1} S_{\text{ps}}^{(k)} \right] + \mathbb{V}\text{ar } \left[ \sum_{i=1}^{N} S_{\text{fcfs},i} \right].$$

To approximate the variance of the sojourn times at the entry node, we use the linear interpolation of Van den Berg and Boxma in [21] to obtain the second moment of the sojourn time of an M/G/1/PS node. We adapt the expression by considering the $N+1$ visits together as one visit with a service time that is a convolution of $N+1$ service times. Therefore, we have that the second moment of the sojourn time is given by

$$\begin{aligned}\mathbb{E}S_{\text{ps}}^2 \approx (N+1)c_{\text{ps}}^2 \left[1 + \frac{2+\rho_{\text{ps}}}{2-\rho_{\text{ps}}}\right] \left[\frac{\beta_{\text{ps}}}{1-\rho_{\text{ps}}}\right]^2 + \\ \left((N+1)^2 - (N+1)c_{\text{ps}}^2\right) \left[\frac{2\beta_{\text{ps}}^2}{(1-\rho_{\text{ps}})^2} - \frac{2\beta_{\text{ps}}^2}{\rho_{\text{ps}}^2(1-\rho_{\text{ps}})}(e^{\rho_{\text{ps}}} - 1 - \rho_{\text{ps}})\right],\end{aligned}$$

where $c_{\text{ps}}^2$ is the squared coefficient of variation of the service time distribution at the entry node, which should be derived from real data. The variance at the entry node is therefore given by

$$\mathbb{V}\text{ar } \left[ \sum_{k=1}^{N+1} S_{\text{ps}}^{(k)} \right] = \mathbb{E}S_{\text{ps}}^2 - \left((N+1)\mathbb{E}S_{\text{ps}}^{(1)}\right)^2.$$

Let $W_{\text{fcfs},i}$ and $c_{\text{fcfs},i}^2$ denote the waiting time and the coefficient of variation of the service distribution at service node $i$, respectively. Then, the variance of the total sojourn times at the service nodes can be expressed as follows

$$\begin{aligned}\mathbb{V}\text{ar } \left[ \sum_{i=1}^{N} S_{\text{fcfs},i} \right] &= \sum_{i=1}^{N} \mathbb{V}\text{ar } S_{\text{fcfs},i} + 2\sum_{i=1}^{N}\sum_{j=i+1}^{N} \mathbb{C}\text{ov } [S_{\text{fcfs},i}, S_{\text{fcfs},j}] \\ &\approx \sum_{i=1}^{N} \left(\mathbb{E}W_{\text{fcfs},i}^2 - [\mathbb{E}W_{\text{fcfs},i}]^2 + \beta_{\text{fcfs},i}^2\right) \\ &= \sum_{i=1}^{N} \left[\pi_i \frac{2\beta_{\text{fcfs},i}^2}{c_{\text{fcfs},i}^2(1-\rho_{\text{fcfs},i})^2} - \pi_i^2 \frac{\beta_{\text{fcfs},i}^2}{c_{\text{fcfs},i}^2(1-\rho_{\text{fcfs},i})^2} + \beta_{\text{fcfs},i}^2\right] \\ &= \sum_{i=1}^{N} \left[\frac{\pi_i(2-\pi_i)\beta_{\text{fcfs},i}^2}{c_{\text{fcfs},i}^2(1-\rho_{\text{fcfs},i})^2} + \beta_{\text{fcfs},i}^2\right].\end{aligned}$$

Finally, by combining all the expressions for the variances of the sojourn time at each node in the network, we derive that the variance of the response time is given by

$$\begin{aligned}\mathbb{V}\mathrm{ar}\ S \approx (N+1)c_{\mathrm{ps}}^2\left[1+\frac{2+\rho_{\mathrm{ps}}}{2-\rho_{\mathrm{ps}}}\right]\left[\frac{\beta_{\mathrm{ps}}}{1-\rho_{\mathrm{ps}}}\right]^2 + \\ \left((N+1)^2-(N+1)c_{\mathrm{ps}}^2\right)\left[\frac{2\beta_{\mathrm{ps}}^2}{(1-\rho_{\mathrm{ps}})^2}-\frac{2\beta_{\mathrm{ps}}^2}{\rho_{\mathrm{ps}}^2(1-\rho_{\mathrm{ps}})}(e^{\rho_{\mathrm{ps}}}-1-\rho_{\mathrm{ps}})\right]- \\ \left[\frac{(N+1)\beta_{\mathrm{ps}}}{1-\rho_{\mathrm{ps}}}\right]^2+\sum_{i=1}^{N}\left[\beta_{\mathrm{fcfs},i}^2+\frac{\pi_i(2-\pi_i)\beta_{\mathrm{fcfs},i}^2}{c_{\mathrm{fcfs},i}^2(1-\rho_{\mathrm{fcfs},i})^2}\right].\end{aligned}$$

## 5 Numerical Experiments

In this section we assess the quality of the expressions of the mean response time and the variance that was derived in the previous section. First, we perform some numerical experiments to test validity of the expressions against a simulated system. In this case, the mean response time does not need to be validated, because the results are exact due to [17]. Therefore, we can restrict our attention to validating the variance only. Then, we validate the expressions by using real data from Web services. Note that real systems are notoriously complex, while the model we use is simple. Therefore, the validation results should be judged from that perspective.

### 5.1 Accuracy of the Variance Approximation

We have performed numerous numerical experiments and checked the accuracy of the variance approximation for many parameter combinations. This was achieved by varying the arrival rate, the service time distributions, the asymmetry in the loads of the nodes, and the number of servers at the service nodes. We calculated the relative error by $\Delta\mathbb{V}\mathrm{ar}\ \% = 100\% \cdot \big(\mathbb{V}\mathrm{ar}\ S - \mathbb{V}\mathrm{ar}_s\ S\big)/\mathbb{V}\mathrm{ar}_s\ S$, where $\mathbb{V}\mathrm{ar}_s\ S$ is the variance based on the simulations.

We have considered many test cases. We started with a queueing network with exponential service times at the entry node and two service nodes. In the cases where the service nodes were equally loaded and asymmetrically loaded, we observed that the relative error was smaller than 3% and 6%, respectively. We also validated our approximation for a network with five single-server service nodes. The results demonstrate that the approximation is still accurate even for very highly loaded systems. Based on these results, we expect that the approximation will be accurate for an arbitrary number of service nodes. The reason is that cross-correlations between different nodes in the network disappear as the number of nodes increases. Since the cross-correlation terms have not been included in the approximation (because of our initial assumptions), we expect the approximation to have good performance in those cases as well.

**Table 1.** Response time variances of a queueing network with general service times at the entry node and two asymmetrically loaded single-server service nodes

| $\beta_{\text{ps}}$ | $c^2_{\text{ps}}$ | $\beta_{\text{fcfs}}$ | | $\mathbb{V}\text{ar}_s\, S$ | $\mathbb{V}\text{ar}\ S$ | $\Delta\mathbb{V}\text{ar}$ % |
|---|---|---|---|---|---|---|
| 0.1 | 0 | 0.1 | 0.9 | 82.51 | 81.05 | −1.33 |
| 0.3 | 0 | 0.8 | 0.5 | 85.89 | 86.81 | 1.06 |
| 0.1 | 0 | 0.5 | 0.3 | 1.25 | 1.22 | −1.76 |
| 0.3 | 0 | 0.9 | 0.1 | 147.49 | 150.82 | 2.25 |
| 0.1 | 4 | 0.1 | 0.9 | 80.83 | 81.33 | 0.62 |
| 0.3 | 4 | 0.8 | 0.5 | 274.00 | 278.46 | 1.63 |
| 0.1 | 4 | 0.5 | 0.3 | 1.54 | 1.50 | −2.68 |
| 0.3 | 4 | 0.9 | 0.1 | 331.30 | 342.47 | 3.37 |
| 0.1 | 16 | 0.1 | 0.9 | 81.55 | 82.16 | 0.75 |
| 0.3 | 16 | 0.8 | 0.5 | 831.15 | 853.41 | 2.68 |
| 0.1 | 16 | 0.5 | 0.3 | 2.37 | 2.33 | −1.86 |
| 0.3 | 16 | 0.9 | 0.1 | 871.49 | 917.42 | 5.27 |

**Table 2.** Response time variances of a queueing network with general service times at the entry node and two asymmetrically loaded multi-server service nodes

| $\beta_{\text{ps}}$ | $c^2_{\text{ps}}$ | $\beta_{\text{fcfs}}$ | | $\mathbb{V}\text{ar}_s\, S$ | $\mathbb{V}\text{ar}\ S$ | $\Delta\mathbb{V}\text{ar}$ % |
|---|---|---|---|---|---|---|
| 0.1 | 0 | 0.2 | 2.7 | 80.82 | 85.66 | 5.99 |
| 0.3 | 0 | 1.6 | 1.5 | 88.82 | 89.70 | 0.99 |
| 0.1 | 0 | 1.0 | 0.9 | 2.43 | 2.43 | −0.02 |
| 0.3 | 0 | 1.8 | 0.3 | 149.31 | 152.38 | 2.05 |
| 0.1 | 4 | 0.2 | 2.7 | 88.50 | 85.94 | −2.90 |
| 0.3 | 4 | 1.6 | 1.5 | 272.00 | 281.35 | 3.44 |
| 0.1 | 4 | 1.0 | 0.9 | 2.71 | 2.71 | −0.23 |
| 0.3 | 4 | 1.8 | 0.3 | 330.14 | 344.03 | 4.21 |
| 0.1 | 16 | 0.2 | 2.7 | 89.60 | 86.77 | −3.16 |
| 0.3 | 16 | 1.6 | 1.5 | 820.26 | 856.30 | 4.39 |
| 0.1 | 16 | 1.0 | 0.9 | 3.57 | 3.57 | −0.79 |
| 0.3 | 16 | 1.8 | 0.3 | 920.45 | 918.98 | −0.16 |

Since the approximation for different configurations with single-server service nodes turned out to be good, we turned our attention to multi-server service nodes. We carried out the previous experiments with symmetric and asymmetric loads on the multi-server service nodes while keeping the service times at the entry nodes exponential. Both cases yielded relative errors smaller than 6%. Finally, we changed the service distribution at the entry node. Table 1 shows the results for a variety of parameters, where the coefficient of variation for the service times at the entry nodes is varied between 0 (deterministic), 4 and 16 (Gamma distribution). These results are extended in Table 2 with multi-server service nodes. If we look at the results, we see that the approximation is accurate in all cases. To conclude, the approximation covers a wide range of

different configurations and is therefore reliable enough to obtain the variance of the response time.

## 6 Model Validation

In the previous section, we showed using simulation results that the approximation we derived for the variance was effective. In this section, we validate our model with two Web applications: a Web service application and a popular bulletin board Web application benchmark. We present our experimental setup followed by the validation results.

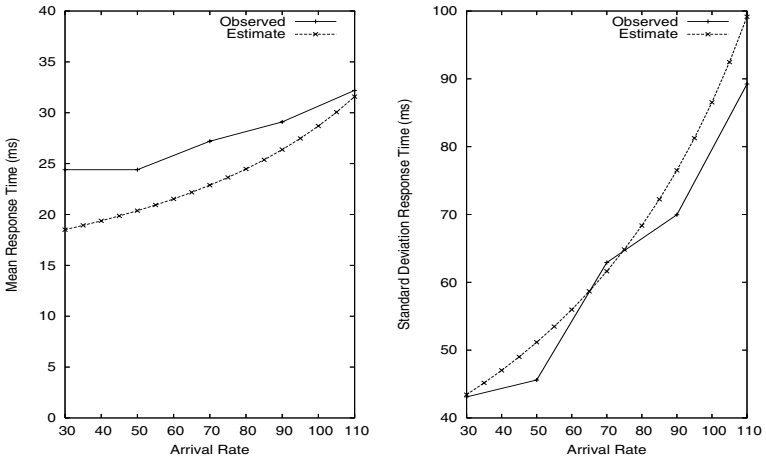
### 6.1 Experimental Setup

For validating our model, we experimented with two types of applications: a promotional Web service and the RUBBoS benchmark. The choice of these two applications was motivated by their differences in their behavior. We hosted the business logic of these applications in the Apache Tomcat/Axis platform and used MySQL 3.23 for database servers. We ran our experiments on Pentium *III* machines with a 900 Mhz CPU and 2 GB memory running a Linux 2.4 kernel. These servers belonged to the same cluster and network latency between the clusters was less than a millisecond. We implemented a simple client request generator that generated Web service requests as a Poisson arrival process. In our experiments, we varied the arrival rate and measured the mean end-to-end response and its variance and compared them with the values predicted by the model.

In our model, an application (or a service) is characterized by the parameters $\beta_{\text{ps}}$ and $\beta_{\text{fcfs}}$. Therefore, to accurately estimate the mean and the variance of the response times for a given application, we first need to obtain these values. Note that all measurements of execution times should be realized during low loads to avoid measuring the queueing latency in addition to the service times. This method has been successfully employed in similar problems [8].

### 6.2 Experiment 1: A Promotional Service

The first type of service we experimented with is a *promotional service* modeled after the "*Recommended for you*" service in amazon.com. This service recommends products based on the user's previous activity. The service maintains two database tables: (i) the item table contains details about each product (title, author, price, and stock); (ii) the customer table contains information regarding customers and the list of previous items bought by each customer. The business logic of this service executes two steps to generate a response. First, it queries the customer table to find the list of product identifiers to which a customer has shown prior interest. Second, it queries the item table to generate the list of items related to these products and their information. The database tables are stored in different database servers running at different machines. This makes




(a) Mean Response Time (b) Standard Deviation

**Fig. 2.** Comparison between the observed and the predicted values for the mean and the standard deviation of the response times for the promotional service

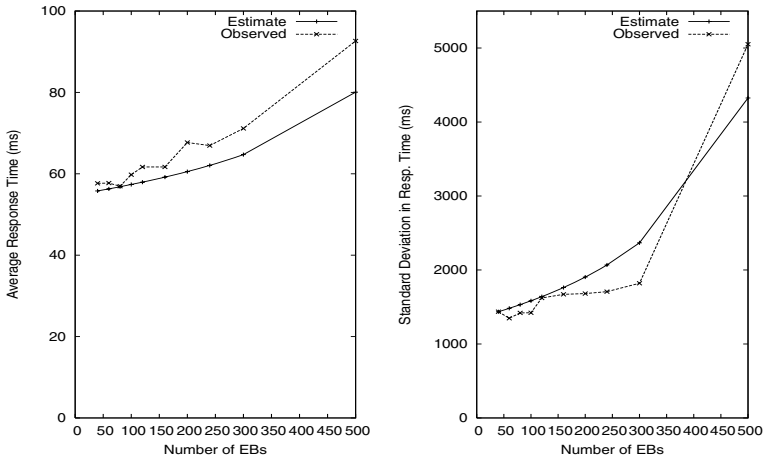
the application a 3-node system with $N = 2$. The business logic is exposed as a Web service and receives the requests in XML format (over HTTP) and returns responses also in XML. In our experiments, we populated the database server with $500,000$ product records and $200,000$ customer records. The appropriate indices were created to enhance the query performance. We restricted ourselves to read-only workloads in our experiments[1].

The results of our experiments are given in Figure 2. As seen in the figure, the model does a reasonable job in predicting the mean response times. The margin of error is less than 5% in most of the cases. In Figure 2(b), we can see that the model does a commendable job in getting reasonable approximations to the standard deviation. This is especially remarkable considering the fact that the service times of the database do not strictly adhere to any specific statistical distribution. This observation also explains the deviation of the predicated mean from the observed mean at high loads. At higher loads, considerable queueing occurs in our model so that the results become more sensitive to the service distribution. Moreover, in practice, the service nodes are multi-threading systems that are best modeled by processor sharing queues.

### 6.3 Experiment 2: RUBBoS – A Bulletin Board Web Application

In our next set of experiments, we experimented with the RUBBoS benchmark, a bulletin board application that models `slashdot.org`. The RUBBoS application models a popular news Web site. RUBBoS's database consists of five tables,

[1] The read and write query characteristics of databases vary a lot and is highly temporal in nature. By far, there are few models that have modeled database systems with complex workloads. Hence, in this study we restrict ourselves to read-only workloads.




(a) RUBBoS Mean Response Time (b) RUBBoS Standard Deviation

**Fig. 3.** Comparison between the observed and the predicted values for the mean and the standard deviation of the response times for RUBBoS benchmark

storing information regarding users, stories, comments, submissions, and moderator activities. Each request to the Web site is handled by the bulletin board application which in turn issues one or more requests to the underlying database to generate the HTML responses. In our experiments, we filled the database with information on 500,000 users and 200,000 comments on the stories. Our experiments were performed with the browse-only workload mix. For our experiments, we chose the open source implementation of these benchmarks[2]. In this implementation, the application is written in Java and runs on the Apache Tomcat servlet engine.

The client workload for the benchmark is generated by Emulated Browsers (EBs). The run-time behavior of an EB models a single active client session. Starting from the home page of the site, each EB uses a Customer Behavior Graph Model (a Markov chain with various interactions with the Web pages in a Web site as nodes and transition probabilities as edges) to navigate among Web pages, performing a sequence of Web interactions [22]. The behavior model also incorporates a think time parameter that controls the amount of time an EB waits between receiving a response and issuing the next request, thereby modeling a human user more accurately. The mean think time between subsequent requests of a client session is set to 7 seconds.

In our experiments, we varied the number of EBs and measured the end-to-end response time for the requests. We compared the measurements against the estimated response time from our model and the results are given in Figure 3. As in the previous experiment, the results in Figure 3 show that the model reasonably predicts both the mean and the variance. The margin of error is less

[2] `http://jmob.objectweb.org/rubbos.html`

than 10% in most of the cases. Given that the RuBBoS application does not conform to all of the assumptions of our model, this is a commendable result.

## 7 Conclusion

In this paper, we presented a simple analytical model for multi-tiered Internet applications based on a queueing-theoretical framework. Based on this model, we can estimate the mean response time of a service and also provide a reasonable approximation to its variance. We verified the effectiveness of our approximations using numerical simulations and validated the complete model with real data using two large-scale Web service applications that are widely deployed on the Internet. Our validation experiments suggest that the model provides high accuracy in estimating not just the mean response time but also its standard deviation. We believe this work serves as a good starting point in modeling complex multi-tiered Internet applications.

There are many interesting avenues for further research on the modeling side. We plan to derive accurate approximations when deterministic routing of requests to the service nodes is replaced by a mixture of Markovian and deterministic routing. This will enlarge the scope of applicability of our model considerably as we can take into effect various caching tiers that are typically used as performance optimization tiers. Moreover, we also plan to enhance the model by modeling the service nodes as processor sharing nodes, which we believe will model servers more accurately. In addition to these extensions, we plan to investigate web admission control models that will allow us to determine the best strategy to handle overloads.

## References

1. Vogels, W.: Learning from the Amazon technology platform. ACM Queue 4(4) (2006)
2. Shneiderman, B.: Response time and display rate in human performance with computers. ACM Comput. Surv. 16(3), 265–285 (1984)
3. Menasce, D.A.: Web server software architectures. IEEE Internet Computing 7(6), 78–81 (2003)
4. Doyle, R., Chase, J., Asad, O., Jin, W., Vahdat, A.: Web server software architectures. In: Proc. of USENIX Symp. on Internet Technologies and Systems (2003)
5. Kamra, A., Misra, V., Nahum, E.: Yaksha: A controller for managing the performance of 3-tiered websites. In: Proceedings of the 12th IWQoS (2004)
6. Abdelzaher, T.F., Shin, K.G., Bhatti, N.: Performance guarantees for web server end-systems: A control-theoretical approach. IEEE Trans. Parallel Distrib. Syst. 13(1), 80–96 (2002)
7. Chen, J., Soundararajan, G., Amza, C.: Autonomic provisioning of backend databases in dynamic content web servers. In: Proceedings of the 3rd IEEE International Conference on Autonomic Computing (ICAC 2006) (2006)
8. Urgaonkar, B., Pacifici, G., Shenoy, P., Spreitzer, M., Tantawi, A.: An analytical model for multi-tier internet services and its applications. In: Proc. of the ACM SIGMETRICS conference, pp. 291–302 (2005)

9. Villela, D., Pradhan, P., Rubenstein, D.: Provisioning servers in the application tier for e-commerce systems. In: Proceedings of the Twelfth IEEE International Workshop on Quality of Service (IWQoS 2004), Montreal, Canada (2004)
10. Bennani, M.N., Menasce, D.A.: Resource allocation for autonomic data centers using analytic performance models. In: ICAC '05: Proc. of the Second Int. Conf. on Automatic Computing, pp. 229–240, Washington, DC, USA (2005)
11. Urgaonkar, B., Shenoy, P.: Cataclysm: policing extreme overloads in internet applications. In: WWW '05: Proceedings of the 14th international conference on World Wide Web, pp. 740–749. ACM Press, New York, USA (2005)
12. van der Mei, R., Meeuwissen, H.: Modelling end-to-end quality-of-service for transaction-based services in a multi-domain environment. In: Proceedings IEEE International Conference on Web Services ICWS, Chicago, USA (2006)
13. Coffman, E., Muntz, R., Trotter, H.: Waiting time distributions for processor-sharing systems. Journal of the ACM 17(1), 123–130 (1970)
14. Morrison, J.: Response-time distribution for a processor-sharing system. SIAM Journal on Applied Mathematics 45(1), 152–167 (1985)
15. Ott, T.: The sojourn time distribution in the M/G/1 queue with processor sharing. Journal of Applied Probability 21, 360–378 (1984)
16. Cooper, R.: Introduction to Queueing Theory. North Holland (1981)
17. Jackson, J.: Networks of waiting lines. Operations Research 5, 518–521 (1957)
18. Boxma, O., Daduna, H.: Sojourn times in queueing networks. Stochastic Analysis of Computer and Communication Systems, pp. 401–450 (1990)
19. Boxma, O., van der Mei, R., Resing, J., van Wingerden, K.: Sojourn time approximations in a two-node queueing network. In: Proc. of ITC 19, pp. 112–1133 (2005)
20. van der Mei, R., Gijsen, B., Engelberts, P., van den Berg, J., van Wingerden, K.: Response times in queueing networks with feedback. Performance Evaluation 64 (2006)
21. van den Berg, J., Boxma, O.: The M/G/1 queue with processor sharing and its relation to a feedback queue. Queueing Syst. Theory Appl. 9(4), 365–402 (1991)
22. Smith, W.: (TPC-W: Benchmarking an e-commerce solution), `http://www.tpc.org/tpcw/tpcw_ex.asp`