Proc. 11th GI/ITG Conf. Measuring, Modelling, and Evaluation of Computer and Communication Systems, Aache, Germany, Sept. 11-14, 2001, invited keynote paper.

Using Performance Models to Dynamically Control E-Business Performance

Daniel A. Menascé
Dept. of Computer Science, MS 4A5
E-Center for E-Business
George Mason University
4400 University Dr.,
Fairfax, VA 22030
menasce@cs.gmu.edu

Abstract

E-business workloads are quite complex as demonstrated by the hierarchical workload characterization discussed here. While these features may pose challenges to performance model builders, it is possible to use queuing network models that closely track the performance metric trends to design and build dynamic QoS controllers as explained in this paper.

1 Introduction

E-business has added a new dimension to the Web by allowing consumers and businesses to interact and carry on commercial transactions over the Internet. These interactions give rise to workload patterns that are unique to these applications and bring their own challenges to the people involved in managing the performance and planning the capacity of the IT infrastructures that support e-business sites.

This paper summarizes the results obtained by the author in characterizing the workload of e-businesses. As shown here, these workloads have interesting and complex features, which pose challenges to performance analysts. One of the goals of performance related studies of Web sites is to improve QoS levels by some form of control (see [2]). We present here a technique that uses queuing models of an e-commerce site to dynamically control their QoS levels.

2 Understanding E-Business Workloads

Understanding the nature and characteristics of e-business workloads is a crucial step to improve the quality of service offered to customers in electronic business environments. However, the variety and complexity of the interactions between customers and sites make the characterization of e-business workloads a challenging problem.

E-business workloads are composed of sessions. A *session* is a sequence of requests of different types made by a single customer during a single visit to a site. During a session, a customer requests the execution of various e-business functions such as browse, search,

select, add to the shopping cart, register, and pay. A request to execute an e-business function may generate many HTTP requests to the site. For example, several images may have to be retrieved to display the page that contains the results of the execution of an e-business function.

A hierarchical approach to understanding e-business workloads was used in [6]. There, the analysis was done in three layers: session layer, function layer, and HTTP request layer. This study examined logs of actual e-commerce sites—an online bookstore and an auction site—and obtained several interesting results that we summarize here.

2.1 Session Layer Characterization

One of the findings is related to session duration. A de facto industry-standard has been that thirty minutes (i.e., 1,800 sec) should be used to delimit sessions. In other words, after thirty minutes of inactivity by a user, the session can be declared to be terminated and any resources associated with the session may be released. Analysis of the data showed that if one varies the session duration threshold and plot the number of active sessions initiated vs. time, one finds that for thresholds larger than 1,000 sec, the number of active sessions varies very little for different threshold values, which indicates that most sessions last less than 1,000 sec.

Another finding of [6] is that if we measure the session length by the number of e-business functions requested by a customer, we see that i) a large majority of the sessions (88%) have less than ten requests and ii) the session length is heavy tailed, especially for sites subject to requests generated by robots [1].

2.2 Function Layer Characterization

Functions were divided in [6] into four categories: static, product selection, purchase, and other. Static functions comprise the home and informational pages about the store. Product selection includes all functions that allow a customer to find and verify a product they are looking for: browse, search, and view. Purchase functions indicate a desire to buy, either by selecting a product for later acquisition (e.g., add to cart) or by ordering it (e.g., pay). One interesting invariant in the logs analyzed in [6], is that more than 70% of the functions performed are product selection functions.

The work in [6] also performed a multi-scale time analysis of the workload. It was observed that very frequent e-business functions (e.g., search at the bookstore) have a pattern similar to the HTTP request process at multiple time scales. The same is not true for less frequent functions such as pay, which show clear bursts and a very different behavior from the HTTP request process.

A very interesting result of [6] is that Zipf's law[10] seems to hold for the terms used in search functions. In other words, the popularity of search terms and their rank follows a Zipf's Law over an extremely wide range of popularity.

2.3 Request Layer Characterization

This level examines the workload as a sequence of HTTP requests and studies the characteristics of the arrival process at several time scales. It was found in [6] that there is a very strong correlation in the arrival process at the request level. This correlation is given by a Hurst parameter value of 0.9273

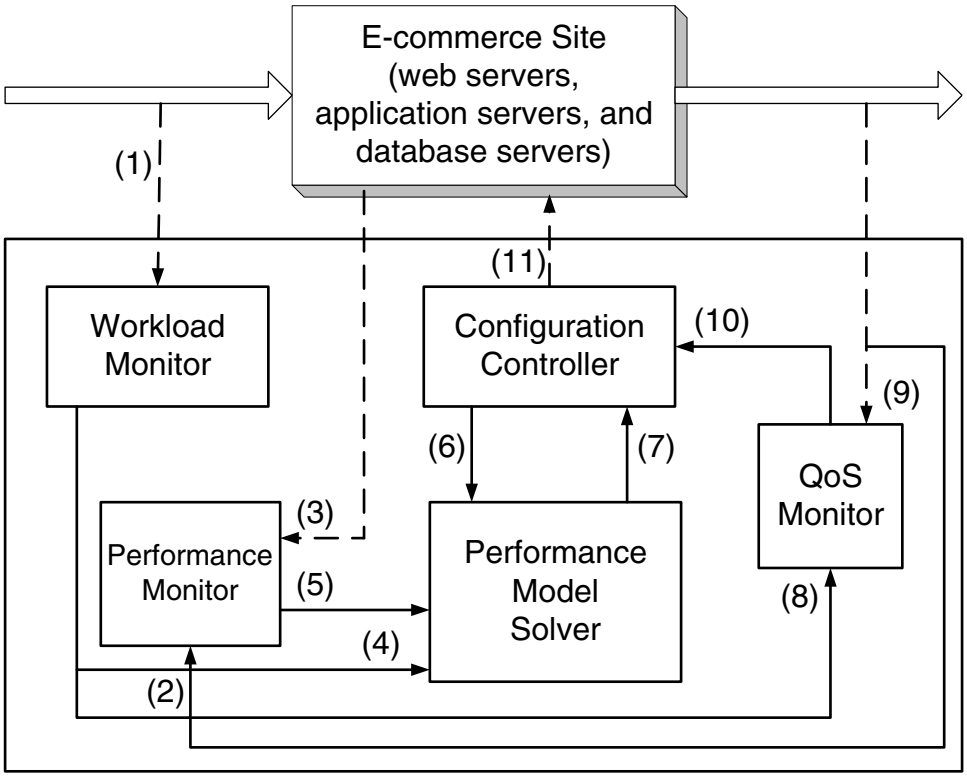

Figure 1: Architecture of the QoS controller.

3 Controlling E-commerce Sites

The complex nature of E-business workloads as described in the previous section, the complexity of the multi-layer architectures of the IT infrastructures that support E-Business sites, and the short-term fluctuations of the workload pose several challenges to performance management and modeling. Many may question if queuing based models [3, 5] can be safely used to address performance issues in E-business environments. We have been investigating the use of these models in the design and implementation of dynamic Quality of Service (QoS) controllers for e-commerce sites [4].

Our approach can be summarized in Figure 1, which shows the architecture of a QoS controller and its relationship to an e-commerce site. The main modules of the controller are the Workload Monitor, Performance Monitor, Configuration Controller, Performance Model Solver, and the QoS Monitor. The Workload Monitor collects information about the workload intensity levels observed in previous intervals. The Performance Monitor collects the utilization of various resources (e.g., processors and disks) needed to compute service demand parameters for a queuing network model for the site. The Configuration Controller uses an optimization technique based on a hill-climbing method guided by a queuing network model of the site to determine the best values for various configuration parameters for the site. Examples of configuration parameters include but are not limited to maximum number of connections per server, number of threads per server, and load-balancer parameters. The Performance Model Solver uses a queuing model of the site with the workload intensity parameters obtained from the Workload Monitor and service demand parameters obtained by the Performance Monitor to provide values of QoS metrics to the Configuration Controller.

The efficiency of this approach was validated in practice in an e-commerce site built

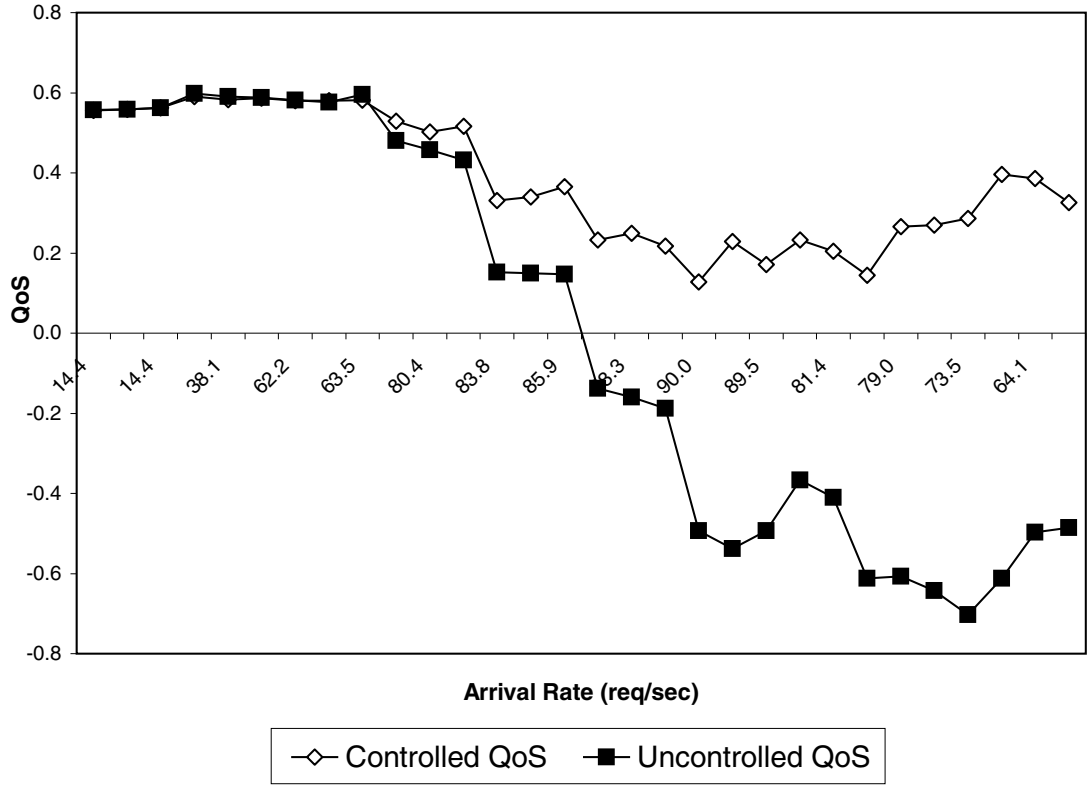

Figure 2: QoS Values With and Without Control.

according to the Transaction Processing Council (TPC) specifications for its e-commerce benchmark TPC-W [9]. We ran many experiments in which the workload was increased to high values and observed the QoS level delivered by the site with the controller enabled and disabled.

We define a QoS function that combines the normalized deviations of the response time ($\Delta QoSR$), throughput ($\Delta QoSX_0$), and probability of rejection ($\Delta QoSP_{\text{rej}}$), respectively, with respect to the desired QoS goals as

$$QoS = w_R \times \Delta QoSR + w_X \times \Delta QoSX_0 + w_P \times \Delta QoSP_{\text{rej}}, \tag{1}$$

where w_R, w_X, and w_P are weights assigned by site management to each QoS deviation. These weights must sum to one and reflect the importance given by management to each QoS metric.

Figure 2 shows that at the beginning, when the arrival rate is in its increasing phase, there is virtually no difference in the QoS levels between the controlled and uncontrolled systems. As the arrival rate reaches its peak value, the QoS value of the uncontrolled system starts to decrease and enters negative territory, indicating violation of one or more of the QoS levels. The QoS for the controlled system manages to stay positive throughout the entire experiment.

4 Concluding Remarks

Despite the complexity of the workload of e-business sites, one can make use of approximate queuing models to dynamically change the parameters of the various components of an e-commerce site to continuously improve its QoS levels. Even in the absence of exact queuing

models for complex computer systems, approximate models that closely track the trends of performance metrics can be very valuable in designing dynamic controllers. The technique described here was successfully applied to a complex e-commerce site, compliant with TPC-W, and implemented in a multi-tiered architecture. This technique is quite general and can be applied to computer systems in general. Other recently examined approaches to QoS control include the design of controllers based on control theory [8].

Acknowledgements

The work of Daniel A. Menascé was partially funded by grant no. INF-00-022 from Virginia's Center for Innovative Technology (CIT) and by the TRW Foundation. The work reported here was done in collaboration with V. Almeida, D. Barbará, R. Dodge, R. Fonseca, W. Meira Jr., F. Pelegrinelli, and R. Riedi.

References

[1] V. A. F. Almeida, D. Menascé, R. Riedi, F. Pelegrinelli, R. Fonseca, and W. Meira, Jr. Analyzing Web Robots and their Impact on Caching, *Proc. Sixth Workshop on Web Caching and Content Distribution*, Boston, MA, June 20-22, 2001.

[2] L. Cherkasova and P. Phaal. Session Based Admission Control: A Mechanism for Improving the Performance of an Overloaded Web Server,HPL-98-119, HP Labs Technical Reports, 1998.

[3] D. A. Menascé and V. A. F. Almeida, *Capacity Planning for Web Performance: metrics, models, and methods*, 2ed., 2001, Prentice Hall, Upper Saddle River, New Jersey.

[4] D. A. Menascé, D. Barbará, and R. Dodge. Preserving QoS of E-commerce Sites Through Self-Tuning: A Performance Model Approach, *Proc. 2001 ACM Conference on E-commerce*, Tampa, FL, October 14-17, 2001.

[5] D. A. Menascé and V. A. F. Almeida. *Scaling for E-business: technologies, models, performance, and capacity planning*, 2000, Prentice Hall, Upper Saddle River, NJ.

[6] D. A. Menascé, V. A. F. Almeida, R. Riedi, F. Pelegrinelli, R. Fonseca, and W. Meira Jr. In Search of Invariants for E-Business Workloads, Proc. Second ACM Conference on Electronic Commerce, Minneapolis, MN, October 17-20, 2000.

[7] D. A. Menascé, V. A. F. Almeida, R. Fonseca, and M. A. Mendes. A Methodology for Workload Characterization of E-commerce Sites, *Proc. 1999 ACM Conference on Electronic Commerce*, Denver, CO, November, 1999.

[8] S. Parekh, N. Gandhi, J. Hellerstein, D. Tilbury, T. Jayram, and J. Bigus. Using Control Theory to Achieve Service-level Objectives in Performance Management, *Proc. Int'l Symp. Integrated Network Management*, May 2001.

[9] Transaction Processing Council. TPC-W: A Transactional Web E-commerce Benchmark, www.tpc.org/tpcw

[10] G. Zipf. *Human Behavior and the Principle of Least Effort*, Addison-Wesley, Cambridge, MA, 1949.