Performability: An e-Utility Imperative

John F. Meyer
EECS Department
University of Michigan
Ann Arbor, MI 48109
Ph. +1-616-335-3337
Fax +1-734-763-4617
jfm@umich.edu

Lisa F. Spainhower
Server Group
International Business Machines, Inc.
Poughkeepsie, NY 12601
Ph. +1-845-435-6485
Fax +1-845-432-9413
lisa@us.ibm.com

Abstract

An e-business infrastructure is emerging to support a new generation of server-hosted, network-delivered applications and services: the e-utility. An e-utility is implemented by scalable clusters of small servers which, collectively, deliver Web content and services for multiple businesses. Demands for such services, as experienced by an entire utility, typically consist of concurrent and independent read-only requests, resulting in a randomly varying workload whose peak rate far exceeds the average request-rate. Hence, significant overprovisioning of capacity is currently quite common. This situation is undesirable for the utility provider and differential quality of service (QoS) is emerging as a possible remedy. Over time, the allocation of computing resources may change from one business to another, depending on prior contractual agreements (e.g.,workload prioritization, resource limits) and the availability of other servers (as randomly affected by faults, local workloads, etc.). Hence, dynamics of both an e-utility and its operational environment can substantially affect QoS perceived by users.

To evaluate e-utility QoS, we contend that the use of model-based performability evaluation techniques and tools is imperative. Our discussion begins with some relevant background regarding both e-utilities and the concept of performability. The above contention is then supported with a more general argument based on definitions of QoS provided by the telecom and Internet communities. Specifically, we show that performability measures specialize to QoS measures and, moreover, provide an effective means of expressing the "collective effect" of lower level measures. A performability approach to evaluating e-utility QoS is then discussed, where the main steps concern QoS measure specification, workload characterization, system model construction, measure formulation (in terms of model behavior), and finally measure evaluation.

Index Terms: e-commerce, e-utility, performability, quality of service, Web servers, Web services.

1. Introduction

Similar to the provisioning of electricity, it is becoming increasingly apparent that Web services can be more effectively and efficiently delivered by a central utility. Due to the rapid growth of demands for such services, a utility this type, referred to simply as an *e-utility*, will soon consist of thousands of servers. The driving forces behind e-utilities are the Internet, the increasing complexity of technology, and the need to pool IT skills [6,8]. The Internet is the prime driver since it permits sharing people and technology over an open, standards-based infrastructure. The rapid rate of deployment of new e-commerce applications calls for frequent changes in existing hardware and software, thus increasing complexity. And the centralized nature of an e-utility permits the integration of various IT skills within a single facility. .

In the discussion that follows, we are concerned with how the behavior of an e-utility can be evaluated in terms that convey the quality of service (QoS) perceived by its users (clients who access Websites supported by the utility). In particular, we seek QoS measures which are suited to model-based evaluation. Although design issues are not dealt with directly, just as evaluations of performance measures such as throughput and response time are employed in the "capacity planning" of computer systems, QoS evaluations can likewise be applied to the design and control of an e-utility. Indeed, from a practical standpoint, such applications constitute a principal reason for developing effective means of model-based QoS evaluation.

As suggested by the title of the paper, we contend that the use of (model-based) performability evaluation techniques and tools is imperative in the above regard. This is due to the fact that user-perceived QoS is affected by consequences of fault occurrences and subsequent fault recovery actions. Accordingly, QoS evaluations should account for factors affecting dependability as well as performance. However, most of the work to date concerning QoS in a Web service context has dealt with performance aspects only.

We begin in Section 2 with some relevant background regarding both e-utilities and the concept of performability. Section 3 then supports the above contention with a more general argument based on definitions of QoS provided by the telecom and Internet communities. Specifically, we show that performability measures specialize to QoS measures and, moreover, provide an effective means of expressing the "collective effect" of lower level measures. Model-based e-utility QoS evaluation is then addressed in Section 4, where we describe a 5-step approach for this purpose.

2. Background

2.1 e-Utilities

Existing e-utilities are Web-hosting centers, and of the 1000 new data centers to be built by 2003, 80% will be of this type [21]. A characteristic of their workloads is a high degree of unpredictability with peak to average ratios of 5:1 or more [4]. Insurance of acceptable response times typically requires server utilization to remain below 70% [2,23]. Consequently, "rules of thumb" are applied that result in 2-5x average capacity over-provisioning [19,20]. To operate more efficiently, providers of e-utilities are developing offerings which eliminate excessive over-provisioning. Planning for the offerings needs to include evaluation of capacity loss due to failures as well as the effects of high demand.

Enforcement of differential QoS is a method of reducing the level of over-provisioning while still providing acceptable response time to the most business-critical users [1,22]. The notion is to divide requests into two or more classes, which can occur at different levels of granularity. At a coarse level, an e-utility may offer each business the possibility of subscribing to one of several quality choices. Classes can be distinguished by average or maximum response time for some guaranteed percent of requests, time of day, denied requests, etc. A more fine-grained level would permit the business itself to offer different user classes. For instance an on-line brokerage may give best response time to established customers and deny access to quote browsers when established customer usage is high.

Even more granular, a user may change classes based on the nature of usage. For example, when a catalog browser adds an item to a shopping cart, that user may be moved into a more privileged class.

When the total request rate is average, all requests get essentially equivalent response times. However when the request rate increases such that some response times are exceeding that guaranteed by the highest priority class of service, the highest priority get preferential treatment. Methodologies for determining and managing differential QoS are the subject of considerable research and beyond the scope of this paper (see [1,17,22], for example). Providers of e-utilities must additionally consider the frequency of failure of servers and other necessary equipment to assure that contractual agreements can be achieved.

Another approach to limiting extreme over-provisioning is provide each business (i.e., the Website thereof) adequate capacity for current usage plus some headroom and then have a pool of undedicated servers which are assigned to the business based on demand and policy [3]. This is most effective if the peak periods of the supported Websites are unlikely to overlap. Design also must consider the incremental capacity of each server together with factors concerning its physical integrity (e.g., failure rates, repair times). Conceptually similar is the use of virtual servers but business assignment is not on physical server boundaries; that is, servers are shared. It permits consolidation onto fewer physical entities and resource reallocation can be done more dynamically in real-time response to request rate changes, but introduces issues of security and privacy that to date have been inhibitors to widespread adoption. Consolidation of many workloads on one or a few servers is very attractive from a system management perspective. However, accounting for effects of faults becomes especially critical in delivering guaranteed QoS with virtual servers.

These methods for improving utilization have common elements. There is adequate capacity to respond to all requests most of the time; there is not enough capacity to respond to all Websites peak rate at the same time; policy has determined which requests will be delayed or refused when necessary; failures can affect the policy-based service delivery. All of these factors need to be accounted for when formulating the QoS of an e-utility. As argued in Section 3, the concept of a performability measure is well suited for this purpose. The subsection that follows provides a brief review of this notion along with reasons for its development.

2.2 Performability

In the context of system evaluation, the notion of *performance* traditionally refers to "how well a system performs, provided it is correct." On the other hand, per terminology established by IFIP Working Group 10.4 [9], *dependability* is that property of a system which allows "reliance to be justifiably placed on the service it delivers." Such service is *proper* if it is delivered as specified; otherwise it is *improper*. System *failure* is identified with a transition from proper service to improper service.

This basic distinction between performance and dependability has been particularly useful in developing measures and evaluation techniques suited to each concept. As a consequence, both performance evaluation and dependability evaluation have evolved as important technical disciplines. However, if separate evaluations of system performance and system dependability are to suffice, one must assume a binary view of the system's operational state (either up or down) and equate proper service (see above) with that delivered when the system is up. In this case, performance measures quantify the quality of proper service and dependability measures quantify the system's ability to deliver service of that quality. Accordingly, results of each type of evaluation, when taken together, can provide a rather complete assessment of user-perceived quality.

Complex information systems, on the other hand, are typically *degradable* in the sense that fault-caused variations in system structure, internal state, and environment can alter the quality of delivered service, even though that service, according to failure criteria, remains proper. For such systems, a binary correct-incorrect (up-down) classification of operational status is too coarse. These limitations were recognized in the mid-1970s, motivating the definition of a unified performance-dependability concept called *performability* [13]. Formally, a performability measure is a probability measure (generally the probability distribution function but often the expected value) of a random variable Y_T that is defined with respect to a specified *utilization period T* and takes values in a designated set *A* of possible *performance outcomes*. Depending on *A*, the performability variable Y_T may be either discrete or continuous, ranging from the type of binary-valued variables (success or failure) considered in dependability evaluation to continuous-valued variables of the type addressed in traditional performance evaluation. Beginning

with this basic notion, the development and application of theory, techniques, and tools for performability modeling and evaluation has enjoyed significant progress over the past 20 years (see surveys [14,15], for example, along with a recent book on the subject [7]).

3. QoS and Performability

QoS is a term that originated within the telecommunication industry, being defined as the collective effect of *service performances* which determine the degree of satisfaction of a user of the service [5]. As elaborated in the reference just cited, the service performances are of four types, namely

i) *service support performance*,
ii) *service operability performance*,
iii) *serveability performance*, and
iv) *service integrity*.

where i) relates to the user-administration interface and ii)-iv) to the user-network interface. Of greatest relevance to the discussion that follows is serveability performance, which consists of *service accessibility performance* and *service retainability performance*. In turn, each of these is determined by various *item performances* that quantify either performance (in the strict sense) or dependability.

As noted in the introduction, QoS is now an emerging as concern of Internet service providers and users, where the *Internet QoS Forum* views it as a collective measure of the level of service delivered to the customer [18]. In this context, it is characterized by several basic performance criteria, including availability (low downtime), error performance, response time and throughput, lost calls or transmissions due to network congestion, connection set-up time, and speed of fault detection and correction.

Although the telecom and Internet definitions differ in their details, both emphasize the "collective" nature of QoS, where the items being collected are lower level aspects of system performance and dependability. However, just how such a collection might be accomplished is not well specified. One possibility is to literally place several lower level measures side by side, i.e., consider a vector-valued measure whose coordinates are the values of the individual measures . However, due to one or both of the following reasons, this approach will generally fail to convey the QoS perceived by a user.

1. User-perceived quality is a unified effect of various aspects of system behavior which are quantified by the lower level measures. Unless there is but one such measure, a side-by-side listing will typically not convey this effect.

2. Individual measures are often formulated and evaluated under differing assumptions. For example, (strict) performance measures assume a fault-free system and dependability measures assume the existence or occurrence of faults, but convey service quality only to the extent of proper (satisfactory) delivery. Assuming at least one measure of each type, if a system is degradable then, as noted in Section 2.2, such a listing will not suffice.

If the lower level measures are evaluated under common assumptions (thus excluding Reason 2), one means of obtaining a unified QoS measure is to express the latter as some function of the underlying measures. For example, this approach has been recently used by Menascé et al. [10,11] to formulate QoS for an e-commerce Website. The lower level measures in this case are server-side average response time R, the site-side throughput X_0 , i.e., the number of requests per second that complete execution from the site, and the probability P_{rej} that an arriving request will be rejected. With respect to specified bounds (lower level QoS requirements) $R_{\max}$, $X_0^{\min}$, and $P_{\text{rej}}^{\max}$ on these respective measures, normalized *QoS deviations* are then defined for each measure. For example, the QoS deviation for the average response time R is

$$\Delta QoS\, R = (R_{\max} - R) / R_{\max} .$$

The deviations $\Delta QoS\ X_0$ and $\Delta QoS\ P_{\text{rej}}$ are similarly defined. These are then unified via weighting factors w_R, w_X, and w_P that sum to 1 and reflect the relative importance of the respective deviations, yielding the measure

$$QoS = w_R \times \Delta QoS\ R + w_X \times \Delta QoS\ X_0 + w_P \times \Delta QoS\ P_{\text{rej}}. \qquad (1)$$

Note, however, that the normalized deviations so combined relate to performance only, i.e., it is tacitly assumed that the site so evaluated is fault free, thereby avoiding Reason 2. Indeed, this assumption is crucial to subsequent developments in [10,11] where queueing models are employed for the purpose of dynamic QoS control. However, as emphasized in the introduction, effects of faults (e.g., server failures) can alter the QoS of an e-utility and, hence, that of a Website. Moreover, for Reason 2, a simple side-by-side combination with measures that account for such effects (e.g., reliability, availability) will not suffice.

Generally, the problem of expressing the combined effects of various aspects of degradable system behavior is just what the concept of performability aims to solve. In particular, if the performability variable Y_T is QoS then its corresponding measure is indeed a measure in the probability-theoretic sense. Moreover, the formulation of Y_T, whether model-based or measurement-based, is an expression of the so-called "collective" effect referred to in the telecom and Internet definitions of QoS. This is not to say that the "collective" issue is avoided through the use of performability evaluation; instead, it is treated as an inherent part of the evaluation process.

4. Model-Based Evaluation of e-Utility QoS

For the reasons cited above, we employ performability measures to express the quality of service provided by an e-utility, where the performability variable Y_T is either QoS, per se, or relates to it in some meaningful way. For example, with respect to specified QoS requirements, Y_T could be the fraction of a utilization period T during which such requirements are satisfied (or, alternatively, violated).

Moreover, for design and planning purposes, we contend that the evaluation of such measures should be based primarily on models as opposed to actual measurements. This is due, in part, to the fact that measurement of fault effects must rely either on field data collected over a long period of time or on fault injection techniques. Although considerable effort has been devoted to the latter, progress in this regard has been disappointing. On the other hand, evaluation results based on models can be obtained much more quickly, thereby expediting design and planning decisions. Moreover, as in [10,11], it is possible to incorporate model-based evaluation as an integral part of an e-utility for the purpose of QoS control.

Specifically, we propose the following 5-step approach to effective QoS-related performability evaluation of an e-utility.

1. Specification of the QoS-related performability variable(s) Y_T whose measures are to be evaluated.
2. Characterization of e-utility workload.
3. Construction of a stochastic process model of the total system (e-utility and its workload) that supports Steps 4 and 5.
4. Formulation of the variable(s) Y_T in terms of the state behavior of the system model.
5. Evaluation (solution) of designated measures of the variables Y_T.

Although choosing what is to be evaluated (Step 1) is often delayed until after a system model is constructed, tailoring the model to fit what is measured can often result in reduced model complexity. As argued in Section 3, what we seek here are measures of user-perceived QoS. The unified measure defined in [10,11] (see equation (1)) might be considered, provided that the lower level measures R (server-side average response time), X_0 (throughput), and P_{rej} (probability of request rejection) are treated as performability measures before being combined in this manner. However, even when fault effects are not accounted for, there are dependencies among

these measures, e.g., a lower rejection probability P_{rej} increases the average response time R for requests that are accepted. Hence, choosing the values of w_R , w_X , and w_P in equation (1) requires a great deal of judgement. Moreover, throughput X_0 is a site-wide measure of interest to e-utility providers, but is perceived by users only through its influence on R and P_{rej} .

Individually, both R and P_{rej} reflect user-perceived quality and each suggests candidates for Y_T . As opposed to considering long-term averages (as in the definitions R and P_{rej}), let T be some bounded period of e-utility use (e.g., a 24-hour day). Suppose further that request rejection (service denial) is the quality in question (response time can be treated in a similar fashion). To express this in performability terms, let $u(t) = 1$ if the e-utility cannot accept a service request at time t; else $u(t) = 0$. Then the performability variable

$$Y_T = \int u(t)dt / T$$

(the integral is over the period T) is the fraction of period T during which service requests are rejected. In particular, note that Y_T is influenced by aspects affecting both performance (e.g., congestion that fills an input queue) and dependability (e.g. faulty servers). As T approaches infinity, Y_T approaches the steady-state probability of rejecting a service request. Although the latter is much easier to analyze (see Chapter 11.7.3 of [12], for example), a bounded period T requires evaluation in a transient domain which, with respect to fault effects, is much more realistic.

Regarding step 2) and following usual practice, this involves the determination of model parameters concerning both workload intensity (the frequency of service requests) and the amount of work required to process a service request. With regard to workload intensity, the number of Website hits per hour is an important initial consideration. As revealed in Figure 1, such hit rates vary with time and, moreover, differ according to the nature of an e-business. Specifically, Figure 1 displays the variation in Website loads for three different sites that employ a common e-utility, where this is actual data collected over a 24-hour period.

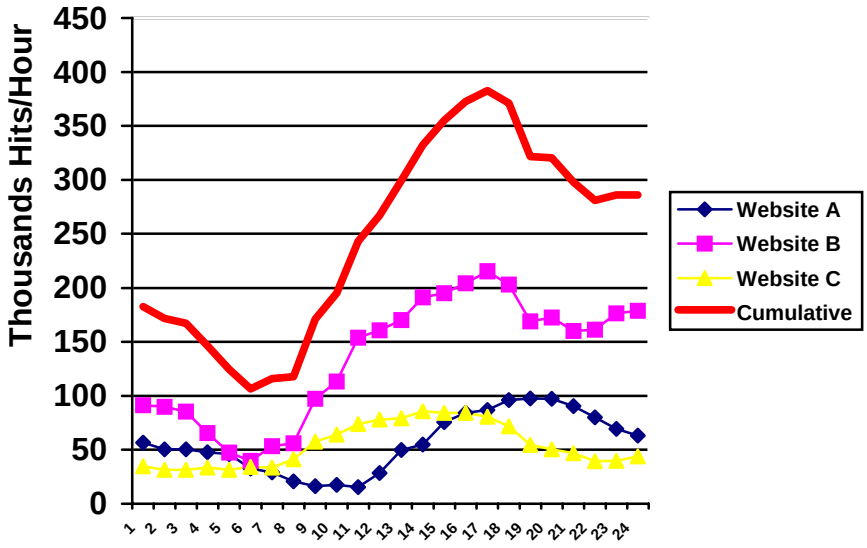

Figure 1: Variations in Website loads over a 24-hour Period.

Unless the hit rates are averaged, which masks important information concerning peak rates, such variations with time need to be accounted for by parameters that characterize the "burstiness" of Web traffic.

If Websites supported by an e-utility are not distinguished with respect to QoS guarantees, it suffices to consider the cumulative load due to the users of all the sites. This is also displayed in Figure 1 (assuming sites A, B, and C are the only ones supported by the utility). However, as discussed in Section 2.1, it may still be the case that a given Website offers different choices of service quality to its users. If so then service requests need to be classified accordingly, resulting in intensity parameters for each class. Assuming that a service request results from an HTTP

request (as seen by an e-utility at the network-utility interface), the amount of work required to process a request depends on the nature of the requested action (e.g., GET, HEAD, etc.) and possibly other information such as the type of document requested. Such work requirements are translated into service times when constructing the system model.

Steps 3,4, and 5 can be accomplished using known techniques and tools for performability model construction and solution (see [7], for example). In view of the above discussion, construction of a stochastic model of the total system (Step 3) obviously relies on both the workload characterization and the performability variable(s) Y_T. Although detailed formulations of the Y_T (Step 4) must await model construction, as noted earlier, their specification (Step 1) serves to drive the modeling process.

In conclusion, although we have yet to apply this approach to a specific e-utility, a recent evaluation study of a similar nature [16] attests to its feasibility.

References

[1] T. Abdelzaher and C. Lu: Modeling and performance control of Internet servers, in *39th IEEE Conference on Decision and Control*, December 2000.

[2] T. Abdelzaher, K. Shin, and N. Bhatti: Performance guarantees for Web server end-systems: A Control-theoretical approach, in *IEEE Trans. on Parallel and Distributed Systems* (to appear).

[3] K. Appleby, S. Fakhouri, L. Fong, G. Goldszmidt, et al.: Oceano – SLA Based Management of a Computing Utility. In *Proc. 2001 IEEE/IFIP Int'l Symp. on Integrated Network Management (IM 2001*), Seattle, WA,May 2001

[4] M. Aron, S. Iyer, and P. Druschel: A resource management framework for predictable quality of service in Web servers (submitted for publication).

[5] *CCITT Blue Book, Fascicle III.1*, International Telecommunication Union, Geneva, 1989.

[6] Fixing the Outsourcing Problem, *Forrester Research*, October, 2000

[7] B. R. Haverkort, R. Marie, G. Rubino, and K. Trivedi, editors, *Performability Modelling: Techniques and Tools*, Wiley, 2001.

[8] C. Houlton: *Utility Computer Providers: Infrastructure, Management and Functionality Providers.* Dain Rauscher Wessels Institiutional Research, October 12, 2000.

[9] J.-C. Laprie, editor: *Dependability: Basic Concepts and Terminology*, volume 5 of *Dependable Computing and Fault-Tolerant Systems*, Springer-Verlag, 1992.

[10] D. Menascé: Using performance models to dynamically control e-business performance, in *Proc. 11th GI/ITG Conference on Measuring, Modelling, and Evaluation of Computer and Communication Systems*, Aachen, Germany, September 2001.

[11] D. Menascé, D. Barbará, and R. Dodge: Preserving QoS of e-commerce sites through self-tuning: A performance model approach, in *Proc. 2001 ACM Conference on E-commerce*, Tampa, FL, October 2001.

[12] D. Menascé and V. Almeida: *Capacity Planning for Web Services: Metrics, Models, and Methods*, Prentice Hall, 2001.

[13] J. F. Meyer: On evaluating the performability of degradable computing systems, in *Proc. 8 th Int'l Symp. on Fault-Tolerant Computing*, pp. 44-49, Toulouse, France, June 1978.

[14] J. F. Meyer: Performability: A retrospective and some pointers to the future, *Performance Evaluation*, vol. 14, no. 3-4, pp. 139-156, February 1992.

[15] J. F. Meyer: Performability evaluation: Where it is and what lies ahead, in *Proc. 1995 IEEE Int'l Computer Performance and Dependability Symposium*, pp. 334-343, Erlangen, Germany, April 1995.

[16] J. F. Meyer: Performability of an algorithm for connection admission control, *IEEE Trans. Comput*., pp. 724-733, July 2001.

[17] S. Parekh, N. Gandhi, J. L. Hellerstein, D. Tilbury, T. . Jayram, and J. Bigus: Using control theory to achieve service level objectives in performance management, in *Proc. IFIP/IEEE International Symposium on Integrated Network Management*, 2001.

[18] Quality of service-Glossary of terms, www.qosforum.com/white-papers/qos-glossary-v4.

[19]S. Schechter, M. Krishnan, and M.D. Smith: Using path profiles to predict HTTP requests. *Proceedings of the Seventh International World Wide Web Conference* Brisbane, Australia, 1998

[20] Solaris resource manager. Sun Microsystems, Inc., 1999.

[21] D. Tapper and D. Chandler: The Computing utility market: Should IBM adopt the Exodus model, *International Data Corporation Bulletin*, 2001.

[22] Y. Teo, R. Ayani: Comparison of load balancing strategies on cluster-based Web servers, *Transactions of the Society for Modeling and Simulation*, 2001.

[23] UpScale Case Study: Telecom online sales and service, Ejasent, Inc., July, 2001.