Proceedings of the 2008 Computer Measurement Group (CMG) Conference, Las Vegas, NV, Dec. 8-12, 2008.

Computing Missing Service Demand Parameters for Performance Models

Daniel A. Menascé
Dept. of Computer Science, MS 4A5
The Volgenau School of IT & Engineering
George Mason University
Fairfax, VA 22030, USA
menasce@gmu.edu

Abstract

One of the challenges in building analytic performance models such as queuing network models is obtaining service demands for the various workloads and various devices. While some of these parameters can be easily measured, some may not be easy to obtain due to the complexity that the measurements may entail or because it may not be possible to stop the operation of a production system to collect measurements. This paper discusses a black-box approach for computing unknown service demand parameters in queuing network models. The paper addresses the problem of finding a subset of the service demand values given the known values and given the values of the response times for all workloads. A unique closed form solution is given for the case of a single missing parameter and a process for obtaining a feasible solution for the case of multiple missing parameters is discussed. Numerical examples illustrate the approach. An online service demand estimator that successively computes better estimates for the service demands is described. The experiments carried out with this estimator show a relatively low relative error in the predicted response times when the estimated service demands are used.

1 Introduction

Significant research has been done in the development of analytic models that can be used to predict the performance of computer systems given a set of parameters [1, 2, 3, 8]. These models, called Queuing Network (QN) models, have been shown to be quite successful and robust and have been incorporated in many commercial performance prediction and capacity planning products. The typical problem solved by these formulations is depicted in Fig. 1, which shows an analytic model of a computer system receiving a set of input parameters and generating a set of performance metrics.

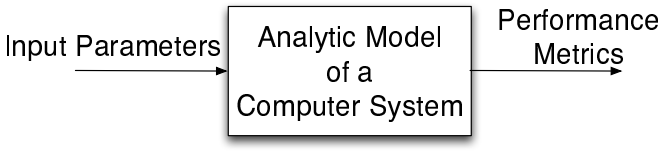

Figure 1: Typical Use of Analytic Performance Models.

Sometimes, we do not have all input parameters for a performance model but can easily measure the response of a system for which we want to build a model. So, the problem we address in this paper is that depicted in Fig. 2 which consists of computing the values of the missing input parameters from the known input parameters and from the measured values of the performance metrics. This is a kind of "inverse" problem from that illustrated in Fig. 1 and requires knowledge of the analytic performance model plus some type of numeric multi-equation solver, as will be discussed later in the paper.

The rest of the paper is organized as follows. Section 2 discusses basic concepts and notation used throughout the rest of the paper. The next section provides a closed form solution to the problem of finding a single unknown parameter. Section 4 discusses the sensitivity of the response time values to the value of an input parameter. This result is important because it can be used by numeric multi-equation solvers. Section 5 discusses the problem of computing more than one missing parameter. The next section discusses an online service demand estimator that successively computes better estimates for the service demands and shows the predictive power of the estimated service demands. Finally, section 7 presents a summary of the contributions of this paper.

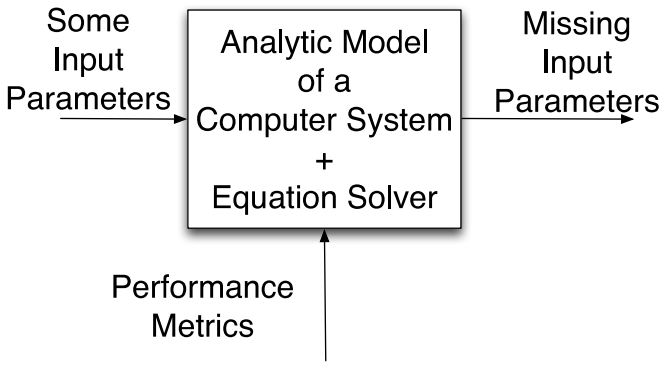

Figure 2: Novel Use of Analytic Performance Models.

2 Basic Concepts and Notation

Analytic queuing network (QN) models are very useful in predicting the performance of computer systems [1, 5, 8]. A QN has a certain number of queues that each represents physical devices in the computer system (e.g., processors, I/O devices, and networks) or logical devices such as software resources (e.g., threads, database locks, or critical sections) [7]. These models may be used to represent various classes of transactions or requests when there is a need to capture workloads that vary in intensity and/or use of resources. Such models are called *multiclass models*.

A QN model has two categories of parameters:

- Workload intensity parameters. They measure how many requests of each workload are present in the system (i.e., the concurrency level) or the rate at which requests of a workload arrive to the system per unit time. When workload intensity parameters are specified as arrival rates, the model is called an *open model*. This paper focuses on these models.
- Service demands. The service demand of a request of a given workload at a given device measures the total service time (no queuing involved) of that request at that device. Since service demands are the sum of all service times at a device during the execution of a request, they do not depend on the system load.

The solution of open multiclass QN models is well-known and is simple to obtain from the input parameters [5]. Let,

- R: number of classes of the QN model,
- K: number of devices in the QN model,
- λ_r: arrival rate (in tps) of class r requests,
- $D_{i,r}$: service demand (in sec) of class r requests at device i, and
- T_r: average response time of class r requests measured at the real system.

Then, T_r can be written as

$$T_r = \sum_{i=1}^{K} \frac{D_{i,r}}{1 - \sum_{v=1}^{R} \lambda_v \times D_{i,v}}. \quad (1)$$

For example, consider a two-class model with the parameters given in Table 1. Using Eq. (1), the response time for classes 1 and 2 are 0.20 sec and 0.24 sec, respectively.

	Class	
	1	2
	Arrival Rates (in tps)	
	5.5	6.0
	Service Demands (in sec)	
CPU	0.030	0.045
Disk 1	0.025	0.038
Disk 2	0.050	0.045

Table 1: Parameters for a 2-class open QN model.

The paper considers the problem of computing unknown service demand parameters for a queuing network (QN) analytical model given that we know the response time and the workload intensity for all classes and some but not all of the service demands. For example, consider that the service demand $D_{CPU,1}$ in the previous example is not known, but we know all other service demands and the response time for classes 1 and 2. The question is whether we can compute $D_{CPU,1}$ from the other parameters and from the response times.

Let us first consider why this is an important problem to solve. We start by first examining how service demands are typically measured. Using the Service Demand Law [5], we can write the service demand $D_{i,r}$ as

$$D_{i,r} = \frac{U_{i,r}}{\lambda_r} \quad (2)$$

where $U_{i,r}$ is the utilization of device i by class r requests and λ_r is the arrival rate of class r requests. The total utilization, U_i, of a device i can be easily obtained from the OS performance monitor. However, a direct measurement of the values of $U_{i,r}$ is not always straightforward because the OS is not aware of the workload characterization determined by the user. There are some techniques for apportioning the total utilization into its class components (see [5]). Thus, one of the reasons for addressing the problem stated above is that it may be sometimes difficult to measure the service demand for a specific device and for a specific class.

Another reason for being able to solve this problem, is that some autonomic computing [4] techniques for designing self-optimizing and self-configuring systems are based

on the system being able to dynamically build an analytic performance model of the system and estimate its service demands [6].

3 Computing a Single Service Demand

In this section, we assume that only one out of the $K \times R$ service demands is unknown. In particular, assume without loss of generality that we want to compute $D_{i,r}$ as a function of T_r $(r = 1, \cdots, R)$, λ_r $(r = 1, \cdots, R)$, and $D_{j,v}$ $(j = 1, \cdots, K; j \neq i; v = 1, \cdots, R; v \neq r)$.

We can re-write Eq. (1) as

$$T_r = \frac{D_{i,r}}{1 - \left(\sum_{v=1, v\neq r}^{R} \lambda_v D_{i,v}\right) - \lambda_r D_{i,r}} + \sum_{j=1, j\neq i}^{K} \frac{D_{j,r}}{1 - \sum_{v=1}^{R} \lambda_v D_{j,v}}. \quad (3)$$

The second term of Eq. (3) (call it α) does not depend on $D_{i,r}$. The summation in the denominator of the first term (call it β) also does not depend on $D_{i,r}$. Thus, Eq. (3) can be written as

$$T_r = \frac{D_{i,r}}{1 - \beta - \lambda_r D_{i,r}} + \alpha \quad (4)$$

Eq. (4) can be easily solved for $D_{i,r}$ and the solution is:

$$D_{i,r} = \frac{(T_r - \alpha)\ (1 - \beta)}{1 + \lambda_r\ (T_r - \alpha)} \quad (5)$$

where

$$\alpha = \sum_{j=1, j\neq i}^{K} \frac{D_{j,r}}{1 - \sum_{v=1}^{R} \lambda_v D_{j,v}} \quad (6)$$

and

$$\beta = \sum_{v=1, v\neq r}^{R} \lambda_v D_{i,v}. \quad (7)$$

The above result indicates that $D_{i,r}$ can be computed using a closed form expression as a function of the response time of class r and no other class, and as a function of the arrival rates of all classes and of all known service demands.

Going back to the example of Table 1, we can compute $D_{\mathrm{CPU},1}$ as a function of the arrival rates λ_1, λ_2, the response time of class 1 (equal to 0.20 sec), and all other service demands. The result is 0.030 sec as expected.

It is important to note that service demand parameters are load independent. Thus, once response times are measured for a given workload intensity level, a missing service demand can be computed and used to predict response times for any future value of the workload intensity. For example, suppose that the arrival rate λ_1 is equal to 6.5 tps instead of 5.5 tps. Then, using Eq. (1), the response time for class 1 becomes 0.22 sec, but $D_{\mathrm{CPU},1}$ is still 0.030 as computed using the method above.

4 Analyzing the Rate of Change of the Response Time with the Service Demands

It is obvious that as a given service demand value increases, all response times increase. But, the question is what is the rate of change of the response times of all classes with respect to the variation of a given service demand. Being able to answer this question is important to decide which device should be upgraded in order to obtain the largest reduction in response time.

To answer the question above, we need to compute the partial derivative of the response time of a given class with respect to the value of a service demand. We consider two cases:

Case 1: Compute the partial derivative of T_r with respect to the service demand $D_{i,s}$, where $s \neq r$. To compute this derivative, we re-write Eq. (1) as

$$T_r = \frac{D_{i,r}}{1 - \lambda_s D_{i,s} - \sum_{v=1, v\neq s}^{R} \lambda_v D_{i,v}} + \sum_{j=1, j\neq i}^{K} \frac{D_{j,r}}{1 - \sum_{v=1}^{R} \lambda_v D_{j,v}}. \quad (8)$$

The derivative of the second term in Eq. (8) with respect to $D_{i,s}$ is zero since it is independent of $D_{i,s}$. Thus,

$$\frac{\partial\ T_r}{\partial\ D_{i,s}} = \frac{\lambda_s D_{i,r}}{\left(1 - \sum_{v=1}^{R} \lambda_v D_{i,v}\right)^2}. \quad (9)$$

The summation inside the parentheses in the denominator of Eq. (9) is simply the utilization U_i of device i. Thus,

$$\frac{\partial\ T_r}{\partial\ D_{i,s}} = \frac{\lambda_s D_{i,r}}{(1 - U_i)^2}. \quad (10)$$

Case 2: Compute the partial derivative of T_r with respect to the service demand $D_{i,s}$, where $s = r$. To compute this derivative, we re-write Eq. (1) as

$$T_r = \frac{D_{i,r}}{1 - \lambda_r D_{i,r} - \sum_{v=1, v\neq r}^{R} \lambda_v D_{i,v}} + \sum_{j=1, j\neq i}^{K} \frac{D_{j,s}}{1 - \sum_{v=1}^{R} \lambda_v D_{j,v}}. \quad (11)$$

The derivative of the second term in Eq. (11) is zero since it is independent of $D_{i,r}$. Thus,

$$\frac{\partial\ T_r}{\partial\ D_{i,r}} = \frac{1 - \sum_{v=1, v\neq r}^{R} \lambda_v D_{i,v}}{\left(1 - \sum_{v=1}^{R} \lambda_v D_{i,v}\right)^2} = \frac{1 - \sum_{v=1, v\neq r}^{R} \lambda_v D_{i,v}}{(1 - U_i)^2}. \quad (12)$$

From Eqs. (9) and (12) we draw the following conclusions:

- The partial derivative of the response time with respect to a given service demand is always positive, as expected. Eq. (9) is clearly positive. It is easy to see that Eq. (12) is also always positive since the denominator is always positive since the numerator is positive given that $\sum_{v=1,v\neq r}^{R} \lambda_v D_{i,v}$ is less than the utilization of device i, which has to be less than one for the system to be stable.

- The partial derivative of the response time with respect to the service demand at a given device i depends on the service demands for device i for all classes but does not depend on any service demand for any other device.

- The partial derivative of the response time with respect to a given service demand of a device i is inversely proportional to the square of one minus the utilization of that device.

Applying the results of Eqs. (9) and (12) to the example of Section 2, we obtain the derivatives shown in Table 2. The table shows that the response time for class 1 increases at a rate of 3.53 seconds per second increase in $D_{\text{Disk2},1}$ when the value of $D_{\text{Disk2},1} = 0.050$. That is the highest rate of increase among all other rates. The rate of increase of the response time for class 2 with respect to $D_{\text{Disk2},2}$ is almost as high. That confirms that an upgrade in the speed of disk 2 will have a very high impact on reducing the response time of both classes.

	Derivatives of T_1	
	1	2
CPU	2.29	0.56
Disk 1	1.92	0.37
Disk 2	3.53	1.45
	Derivatives of T_2	
	1	2
CPU	0.78	2.62
Disk 1	0.52	2.14
Disk 2	1.20	3.50

Table 2: Derivatives of the Response Times for Classes 1 and 2 with Respect to the Service Demands.

5 Computing More than One Service Demand

In the case where more than one service demand is unknown, there is no unique solution, i.e., no single set of values for the missing service demands that gives the same response time values. For example, consider the parameters in Table 3. The only difference with respect to the parameters in Table 1 is in the values of $D_{\text{CPU},1}$ and $D_{\text{Disk2},1}$. However, the response times for classes 1 and 2 are exactly the same as in the example of Table 1.

	Class	
	1	2
	Arrival Rates (in tps)	
	5.5	6.0
	Service Demands (in sec)	
CPU	0.041	0.045
Disk 1	0.025	0.038
Disk 2	0.041	0.045

Table 3: Different Parameters for a 2-class Open QN model.

The problem can now be cast as:

- Given the response times T_r for all classes $r = 1, \cdots, R$. Let us call these response time goals.

- Given a subset $\mathcal{S}$ of the service demand values $D_{i,r}$, for $i = 1, \cdots, K$ and $r = 1, \cdots, R$.

- Given the arrival rates λ_r for all classes $r = 1, \cdots, R$.

- Find a set of values for the service demands not included in $\mathcal{S}$ such that the resulting response times computed using Eq. (1) are the same as the response time goals.

The solution to this problem can be obtained by solving the following non-linear constrained optimization problem:

Minimize

$$\sum_{r=1}^{R} \left(T_r - \sum_{i=1}^{K} \frac{D_{i,r}}{1 - \sum_{v=1}^{R} \lambda_v \times D_{i,v}} \right)^2 \qquad (13)$$

Subject to

$D_{i,r} \geq 0 \quad \forall\, i, r$

$\sum_{r=1}^{R} \lambda_r \; D_{i,r} < 1 \;\; \forall\, i.$

Note that the term inside the parentheses in Eq. (13) has to be equal to zero for each class r when the solution to the problem satisfies Eq. (1). The first constraint simply says that service demands cannot be negative and the second constraint indicates that the utilization of any device has to be less than one.

There are several packages that can be used to solve this type of problem. Besides standalone packages, companies such as Frontline Systems also offer solver engines that can be used in C++, .Net, and Java programs (see http://www.solver.com/). Microsoft's Excel has a Solver available from the Tools menu. This solver uses the Generalized Reduced Gradient Algorithm, which is an iterative numerical method.

These iterative solution methods start from an initial solution, i.e., an estimated matrix of service demands that

has known service demands as well as some guessed values of service demands that have to satisfy the utilization constraint $U_i = \sum_{r=1}^{R} \lambda_r \ D_{i,r} < 1 \ \ \forall \ i$. If U_i is known (and in fact it can be easily measured in most cases), it is even easier to come up with some initial estimates for the unknown service demands based on the known utilization value. If the initial value for the matrix of service demands satisfies Eq. (1), then all solutions to the minimization problem stated above provide the same result for any values of arrival rates and their corresponding response times.

Consider the example problem shown in Table 4, which requires that three service demand values be computed: $D_{\mathrm{CPU},1}$, $D_{\mathrm{Disk1},2}$, and $D_{\mathrm{Disk2},1}$. These missing values are indicated by question marks in Table 4.

	Class	
	1	2
	Arrival Rates (in tps)	
	5.5	6.0
	Response Time Goals (in sec))	
	0.4	0.5
	Service Demands (in sec)	
CPU	?	0.045
Disk 1	0.025	?
Disk 2	?	0.045

Table 4: Example Problem.

Using a numerical solver, such as Microsoft's Excel, we obtain the complete set of values for the service demands as shown in Table 5. When using Eq. (1) to compute the response times we obtain the response time goals in Table 4.

	Class	
	1	2
	Arrival Rates (in tps)	
	5.5	6.0
	Response Time Goals (in sec))	
	0.4	0.5
	Service Demands (in sec)	
CPU	0.078	0.045
Disk 1	0.025	0.088
Disk 2	0.035	0.045

Table 5: Results for Example Problem.

6 Online Estimation of Service Demands

Figure 3 illustrates an online estimator of service demands that collects regular measurements on transaction arrival rates for each class and measures their corresponding response times in order to estimate service demands based on the formulation described in the previous section.

The online estimator starts from an initial estimate D_0 of the matrix of service demands and computes a new estimate using the process in the previous section using as inputs the measured values for arrival rates and response times. The process is repeated several times as new arrival rate and response time measurements are obtained. At each step k, the starting point of the estimation process is the matrix of service demands obtained in the previous step. More precisely, the process used by the online estimator can be described as follows:

$$D_k = f(\vec{\lambda}_k, \vec{T}_k, D_{k-1}) \tag{14}$$

where $\vec{\lambda}_k$ is the set of arrival rates of all classes measured at interval k, $\vec{T}_k$ is the set of response times of all classes measured at interval k, and D_{k-1} is the estimated matrix of service demands obtained at step $k-1$. This matrix would typically contain some known service demands and some estimated service demands.

The final estimate of the matrix of service demands is obtained as the average, over all steps, of the matrices obtained at all steps.

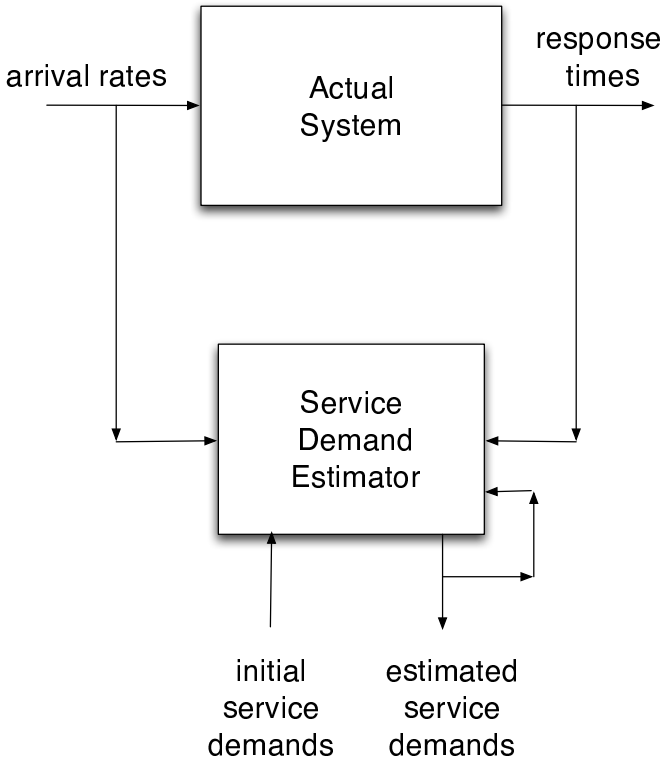

Figure 3: Online Estimation of Service Demands.

We now provide an example of online service demand estimation and show the predictive value of the estimated demands. Assume that the actual system has service de-

mands as specified in Table 5. However, the service demands marked with a "?" in Table 4 are unknown. Table 6 shows values of service demands for $D_{\mathrm{CPU},1}$, $D_{\mathrm{Disk1},2}$, and $D_{\mathrm{Disk2},1}$ obtained at steps 1–8. The column "Load Factor" is a multiplicative factor of the arrival rates in Table 5. The last row of Table 6 shows the average service demands after eight steps.

Step	Load Factor	$D_{\mathrm{CPU},1}$	$D_{\mathrm{Disk1},2}$	$D_{\mathrm{Disk2},1}$
0		0.032	0.090	0.035
1	0.95	0.057	0.088	0.066
2	1.00	0.060	0.088	0.065
3	1.05	0.061	0.088	0.066
4	1.10	0.062	0.088	0.067
5	1.15	0.060	0.088	0.070
6	1.20	0.064	0.088	0.069
7	1.25	0.063	0.088	0.072
8	1.28	0.056	0.088	0.070
Avg.		0.061	0.088	0.068

Table 6: Several steps of the Online Service Demand Estimator.

We then used the average values of the service demands shown in Table 6 as input to an analytic queuing model solver and obtained the response times for each load factor. We also compared these response times with those obtained for the same load factors but assuming the matrix of service demands given in Table 5 (the actual system demands).

The graph of Fig. 4 shows the absolute value of the relative error between the response time for the actual system and that predicted with the estimated service demands. The figure shows that the relative error varies from zero to 7.6% for workload 2 and from 0.8% to 18% for workload 1, a reasonable value given that 50% of the service demands are being estimated. The predictive power of the estimated service demands is better for workload 2 because it has less unknown service demands. It can also be seen that the relative predictive error increases with the workload intensity.

7 Concluding Remarks

This paper discussed the problem of finding unknown values for a subset of the service demands that satisfy given values of the response times in a multiclass open queuing network model. In addition, the paper established the rate of change of the response time of a given class with respect to the value of a given service demand.

The contributions and observations of this paper can be summarized as follows:

- When a single service demand $D_{i,r}$ is unknown, its value can be computed using a closed form expression as a function of the response time of class r and no other class, and as a function of the arrival rates of all classes and of all known service demands.

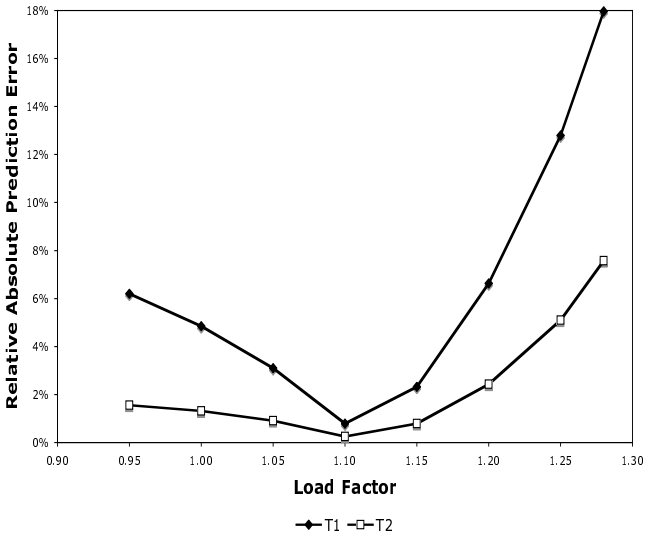

Figure 4: Absolute Relative Error on the Predictive Value of the Estimated Service Demands.

- The rate of variation of the response time with respect to a given service demand is always positive.
- The rate of variation of the response time with respect to the service demand at a given device i depends on the service demands for device i for all classes but does not depend on any service demand for any other device.
- The rate of variation of the response time with respect to a given service demand of a device i is inversely proportional to the square of one minus the utilization of that device.
- In the case where more than one service demand is unknown, there is no unique solution, i.e., no single set of values for the missing service demands that gives the same response time values.
- In the case where more than one service demand is unknown, a feasible set of unknown values can be obtained by solving a constrained non-linear minimization problem using any of the existing numerical solvers.
- An online service demand estimator is described. The experiments carried out with this estimator show a relatively low relative error in the predicted response times when the estimated service demands are used.

Acknowledgments

The work of Daniel Menascé is partially supported by award no. CCF-0820060 from the National Science Foundation.

References

[1] Baskett, F., K. M. Chandy, R. R. Muntz and F.G. Palacios, "Open, closed and mixed networks of queues with different classes of customers," *Journal of the ACM*, 1975, 22: 248260.

[2] P.J. Denning and J.P. Buzen, "The Operational Analysis of Queueing Network Models," *ACM Computing Surveys*, 10, 3 (Sep. 1978), pp. 225–261.

[3] W.J. Gordon and G.F. Newell, "Closed queueing systems with exponential servers," *Operations Research*, 15, 1976, pp. 254–265.

[4] D.A. Menascé and J. Kephart, "Guest Editor's Introduction," Special Issue on Autonomic Computing, *IEEE Internet Computing*, January 2007.

[5] D.A. Menascé, V.A.F. Almeida, and L.W. Dowdy, *Performance by Design: Computer Capacity Planning by Example*, Prentice Hall, Upper Saddle River, 2004.

[6] D.A. Menascé and M.N. Bennani, "On the Use of Performance Models to Design Self-Managing Computer Systems," *Proc. 2003 Computer Measurement Group Conference*, Dallas, TX, Dec. 7-12, 2003.

[7] D.A. Menascé, "Two-level Iterative Queuing Modeling of Software Contention," *Proc. Tenth IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2002)*,Fort Worth, TX, Oct. 12-16, 2002.

[8] M. Reiser and S.S. Lavenberg, "Mean Value Analysis of Closed Multichain Queueing Networks," Journal of the ACM, Vol. 27, No. 2, Apr. 1980.