Capacity Planning for Web Services
Techniques and Methodology

Virgilio A.F. Almeida

Department of Computer Science,
Federal University of Minas Gerais,
31270-010 Belo Horizonte, Brazil
virgilio@dcc.ufmg.br

Abstract. Capacity planning is a powerful tool for managing quality of service on the Web. This tutorial presents a capacity planning methodology for Web-based environments, where the main steps are: understanding the environment, characterizing the workload, modeling the workload, validating and calibrating the models, forecasting the workload, predicting the performance, analyzing the cost-performance plans, and suggesting actions. The main steps are based on two models: a workload model and a performance model. The first model results from understanding and characterizing the workload and the second from a quantitative description of the system behavior. Instead of relying on intuition, ad hoc procedures and rules of thumb to understand and analyze the behavior of Web services, this tutorial emphasizes the role of models, as a uniform and formal way of dealing with capacity planning problems.

1 Introduction

Performance, around-the-clock availability, and security are the most common indicators of quality of service on the Internet. Management faces a twofold challenge. On the one hand, it has to meet customer expectations in terms of quality of service. On the other hand, companies have to keep IT costs under control to stay competitive. Many possible alternative architectures can be used to implement a Web service; one has to be able to determine the most cost-effective architecture and system. This is where the quantitative approach and capacity planning techniques come into play. This tutorial introduces capacity planning [19,1] as an essential tool for managing quality of service on the Web and presents a methodology, where the main steps are: understanding the environment, characterizing the workload, modeling the workload, validating and calibrating the models, predicting the performance, analyzing the cost-performance plans, and suggesting actions. It provides a framework for planning the capacity of Web services and understanding their behavior. The tutorial also discusses a state transition graph called Customer Behavior Model Graph (CBMG), that is used to describe the behavior of groups of users who exhibit similar navigational patterns. The rest of the paper is organized as follows. Section two presents the main steps of the capacity planning methodology. Section three discusses the role of models in capacity planning. Section four describes workload models. Next section discusses issues related to performance models. Finally, section six presents concluding remarks.

2 Capacity Planning as a Management Tool

Planning the capacity of Web services requires that a series of steps be followed in a systematic way. Figure 1 gives an overview of the main steps of the quantitative approach to analyze Web services. The starting point of the process is the business model and its measurable objectives, which are used to establish service level goals and to find out the applications that are central to the goals. Once the business model and its quantitative objectives have been understood, one is able to go through the quantitative analysis cycle. We now cover the various steps of the capacity planning process.

2.1 Understand the Environment

The first step entails obtaining an in-depth understanding of the service architecture. This means answering questions such as: What are the system requirements of the business model? What is the configuration of the site in terms of servers and internal connectivity? How many internal layers are there in the site? What types of servers (i.e., HTTP, database, authentication, streaming media) is the site running? What type of software (i.e., operating system, HTTP server software, transaction monitor, DBMS) is used in each server machine? How reliable and scalable is the architecture? This step should yield a systematic description of the Web environment, its components, and services. This initial phase of the process consists of learning what kind of hardware and software resources, network connectivity, and network protocols, are present in the environment. It also involves the identification of peak usage periods, management structures, and service-level agreements. This information is gathered by various means including user group meetings, audits, questionnaires, help desk records, planning documents, interviews, and other information-gathering techniques [14].

Table 1 summarizes the main elements of a system that must be catalogued and understood before the remaining steps of the methodology can be taken.

2.2 Characterize Workload

Workload characterization is the process of precisely describing the systems's global workload in terms of its main components. Each workload component is further decomposed into basic components. The basic components are then characterized by workload intensity (e.g., transaction arrival rates) and service demand parameters at each resource.

Capacity planning procedures have been used to assure that users receive adequate quality of service as they navigate through the site. A key step of any

Table 1. Elements in Understanding the Environment

Element	Description
Web Server	Quantity, type, configuration, and function.
Application Server	Quantity, type, configuration, and function.
Database Server	Quantity, type, configuration, and function.
Middleware	Type (e.g., TP monitors and DBMS).
Application	Main applications.
Network connectivity	Network connectivity diagram showing LANs, WANs, routers, servers, etc.
Network protocols	List of protocols used.
Service-level agreements	Existing SLAs per application or service.
User Community	Number of potential users, geographic location, etc.
Procurement procedures	Elements of the procurement process, expenditure limits.

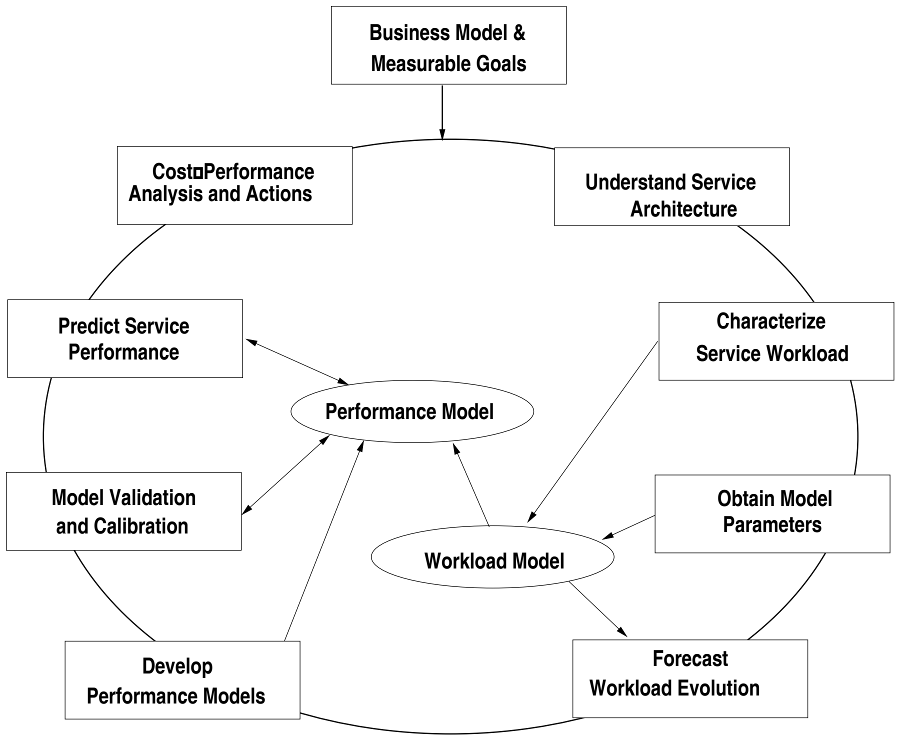
Business Model &
Measurable Goals
Understand Service Architecture
Characterize Service Workload
Obtain Model Parameters
Forecast Workload Evolution
Develop Performance Models
Model Validation and Calibration
Predict Service Performance
Cost Performance Analysis and Actions
Performance Model
Workload Model

Fig. 1. Capacity Planning Process

performance evaluation and capacity planning study is workload characterization [4,18,3]. Thus, the second step of the methodology aims at characterizing the workload of a Web service. In Web-based environments [3], users interact with the site through a series of consecutive and related requests, called sessions. A session is a sequence of requests to execute e-business functions made by a single customer during a single visit to a Web service. Different navigational patterns can be observed for different groups of users. Examples of e-business functions requested by an online shopper include browse the catalog, search for products or services based on keywords, select products to obtain more detailed information, add to the shopping cart, user registration, and checkout. A customer of an online brokerage site would request different functions, such as enter a stock order, research a mutual fund history, obtain real-time quotes, retrieve company profiles, and compute earning estimates. Web workload is unique in its characteristics and some studies [2,5] identified workload properties and invariants, such as the heavy-tailed distributions (e.g., Pareto distribution) of file sizes in the Web. It has been also observed that Web traffic is bursty in several time scales [20,8].

2.3 Obtain Parameters for the Workload Model

The third step consists of obtaining values for the parameters of the workload models. This step also involves monitoring and measuring the performance of a Web service. It is a key step in the process of guaranteeing quality of service and preventing problems. Performance measurements should be collected from different reference points, carefully chosen to observe and monitor the environment under study. For example, logs of transactions and accesses to servers are the main source of information. Further information, such as page download times from different points in the network may help to track the service level perceived by customers. The information collected should help us answer questions such as: What is the number of user visits per day? What is the average and peak traffic to the site? What characterizes the shoppers of a particular set of products? What are the demands generated by the main requests on the resources (e.g., processors, disks, and networks) of the IT infrastructure? Steps 2 and 3 generate the workload model, which is a synthetic and compact representation of the workload seen by a Web service.

The parameters for a basic component are seldom directly obtained from measurements. In most cases, they must be derived from other parameters that are measured directly. Table 2 shows an example of two basic components, along with examples of parameters that can be measured for each. The last column indicates the type of basic component parameter—workload intensity (WI) or service demand (SD). Values must be obtained or estimated for these parameters, preferably through measurements with performance monitors and accounting systems. Measurements must be made during peak workload periods and for an appropriate monitoring interval.

2.4 Forecast Workload Evolution

The fourth step forecasts the expected workload intensity for a Web service. Forecasting is the art and science of predicting future events. It has been extensively used in many areas, such as the financial market, climate studies and

Table 2. Example of Basic Component Parameters and Types

Basic Component and Parameters	Parameter Type
Order transaction	
Number of transactions submitted per customer	WI
Number of registered customers	WI
Total number of IOs to the Sales DB	SD
CPU utilization at the DB server	SD
Average message size sent/received by the DB server	SD
Web-based training	
Average number of training sessions/day	WI
Average size of image files retrieved	SD
Average size of http documents retrieved	SD
Average number of image files retrieved/session	SD
Average number of documents retrieved/session	SD
Average CPU utilization of the httpd server	SD

production and operations management [12]. For example, one could forecast number and type of employees, volume and type of production, product demand, volume and destination of products. In the Internet, demand forecasting is essential for guaranteeing quality of service. It is critical for the operation of Web services. Let us consider the following scenario [16]. Unprecedented demand for the newest product slows Web servers to a crawl. The company servers were overwhelmed on Tuesday as a wave of customers attempted to download the company's new software product. Web services, in terms of responsiveness and speed, started degrading as more and more customers tried to access the service. And it is clear that many frustrated customers simply stopped trying. This undesirable scenario emphasizes the importance of good forecasting and planning for Web environments.

A good forecast is more than just a single number; it is a set of scenarios and assumptions. Time plays a key role in the forecasting process. The longer the time horizon, the less accurate the forecast will be. Forecasting horizons can be grouped into the classes: short term (e.g., less than three months), intermediate term (e.g., from three months to one year) and long term (e.g., more than 2 years). Demand forecasting in the Web can be illustrated by typical questions that come up very often during the course of capacity planning projects. Can we forecast the number of visitors to the company's Web site in order to plan the adequate capacity to support the load? What is the expected workload for the credit card authorization service during the Christmas season? How will the number of messages processed by the e-mail servers vary over the next year? What will be the number of simultaneous users for the streaming media services six months from now? Implementation of Web services should rely on a careful planning process, a planning process that pays attention to performance and capacity right from the beginning. The goal of this step is to use existing forecasting methods and techniques to predict future workload for Web services.

The literature [12,7] describes several forecasting techniques. In selecting one, some factors need to be considered. The first one is the availability and reliability of historical data. The degree of accuracy and the planning horizon are also factors that determine the forecasting technique. The pattern found in historical data has a strong influence on the choice of the technique. The nature of historical data may be determined through visual inspection of a plot of the data as a function of time. Three patterns of historical data are commomly identified: random, trend, and seasonal. While the trend pattern reflects a workload that tends to increase (or decrease, in some cases), seasonal patterns show the presence of fluctuations. The underlying hypothesis of forecasting techniques is that the information to be forecast is somehow directly related to historical data; this emphasizes the importance of knowing the pattern of historical data. There are many commercial packages (e.g., Matlab, S-PLUS, MS-EXCEL [11]) that perform various methods of forecasting techniques.

2.5 Develop Performance Model

In the fifth step, quantitative techniques and analytical models based on queuing network theory are used to develop performance models of Web services. Performance models are used to predict performance when any aspect of the workload or the site architecture is changed. Two types of models may be used: simulation models and analytical models. Analytical models [6] specify the interactions between the various components of a Web system via formulas. Simulation models [10] mimic the behavior of the actual system by running a simulation program. After model construction and parameterization, the model is solved. That is, the model parameters are manipulated in some fashion to yield performance measures (e.g., throughput, utilization, response time). Many solution techniques have been suggested [13].

Performance models have to consider contention for resources and the queues that arise at each system resource—processors, disks, routers, and network links. Queues also arise for software resources—threads, database locks, and protocol ports. The various queues that represent a distributed system are interconnected, giving rise to a network of queues, called a queuing network (QN). The level of detail at which resources are depicted in the QN depends on the reasons to build the model and the availability of detailed information about the operation and availability of detailed parameters of specific resources.

The input parameters for queuing network models describe the resources of the system, the software, and the workload of the system under study. These parameters include four groups of information:

– servers or components
– workload classes
– workload intensity
– service demands

In order to increase the model's representativeness, workloads are partitioned into classes of somehow similar components. Programs that are alike concerning the resource usage may be grouped into workload classes. Depending on the way a given class is processed by a system, it may be classified as one of two types: *open*, or *closed*.

Servers or service centers, are components of performance models intended to represent the resources of a system. The first step in specifying a model is the definition of the servers that make up the model. The scope of the capacity planning project helps to select which servers are relevant to the performance model. Consider the case of a Web site composed of Web servers, application and database servers connected via a LAN. The capacity planner wants to examine the impact caused on the system by the estimated growth of sales transactions. The specific focus of the project may be used to define the components of a performance model. For example, the system under study could be well represented by an open queueing network model consisting of queues, which correspond to the servers of the site. A different performance model, with other queues, would be required if the planner were interested in studying the effect of a proxy cache on the performance of the system.

2.6 Validate Performance Model

Once the model has been constructed, parameterized, and solved, it should be validated. That is, the performance measures found by solving the model should be compared against actual measurement data of the system being modeled. A performance model is said to be valid if the performance metrics (e.g., response time, resource utilizations, and throughputs) calculated by the model match the measurements of the actual system within a certain acceptable margin of error. For instance, the actual server utilizations should be compared against the server utilizations found by solving the model. This comparison check will be judged to be either acceptable or unacceptable. The choice of what determines acceptable versus unacceptable is left to the modeler. As a rule of thumb, device utilizations within 10%, system throughput within 10%, and response time within 20% are considered acceptable [13]. If the comparison is unacceptable, a series of questions must be addressed to determine the source of the errors.

Errors are possible within each capacity planning step. During workload characterization, measurements are taken for service demands, workload intensity, and for performance metrics such as response time, throughput, and device utilization. The same measures are computed by means of the performance model. If the computed values do not match the measured values within an acceptable level, the model must be calibrated. Even though one can hypothesize the source of possible errors, it is often difficult to pinpoint them and correct them. Therefore, it is normal to iterate among the steps of the methodology until an acceptable model is found. This changing of the model to force it to match the actual system is referred to as the calibration procedure. A calibration is a change to some target parameter of the analytic model. A detailed discussion of calibration techniques is given in [13]. When the model is considered valid it can be used for performance prediction. The sixth step aims at validating the models used to represent performance and workload.

2.7 Predict Service Performance

Prediction is key to capacity planning because one needs to be able to determine how a Web service will react when changes in load levels and customer behavior occur or when new business models are developed. This determination requires predictive models and not experimentation. So, in the seventh step, one uses performance models to predict the performance of Web services under many different scenarios [9].

Performance models aim at representing the behavior of real systems in terms of their performance. In order to use performance models for predicting future scenarios, one needs to obtain the input parameters to feed the model. The input parameters for performance models describe the hardware configuration, the software environment, and the workload of the system under study. The representativeness of a model depends directly on the quality of input parameters. Therefore, a key issue to conduct practical performance prediction is the determination of input parameters for performance models. Two practical questions naturally arise when one thinks of modeling a real system:

- What are the information sources for determining input parameters?
- What techniques are used to calculate input parameters?

The main source of information is the set of performance measurements collected from the observation of the real system under study. Further information can also be obtained from benchmarks and from product specifications provided by manufacturers. However, typical measurement data do not coincide with the kind of information required as input by performance models. For modeling purposes, typical measurement data need to be reworked in order to become useful.

2.8 Analyze Future Scenarios

In the eighth step of the cycle, many possible candidate architectures are analyzed in order to determine the most cost-effective one. Future scenarios should take into consideration the expected workload, the site cost, and the quality of service perceived by customers. Finally, this step should indicate to management what actions should be taken to guarantee that the Web services will meet the business goals set for the future.

The performance model and cost models can be used to assess various scenarios and configurations. Some example scenarios are, "Should we use CDN services to serve images?" " Should we use Web hosting services? " "Should we mirror the site to balance the load, cut down on network traffic and improve global performance?" For each scenario, we can predict what the performance of each system component will be and what the costs are for the scenario. The comparison of the various scenarios yields a configuration plan, an investment plan, and a personnel plan. The configuration plan specifies which upgrades should be made to existing hardware and software resources. The performance model is built and solved and a cost model developed, various analyses can be made regarding cost-performance tradeoffs. The investment plan specifies a timeline

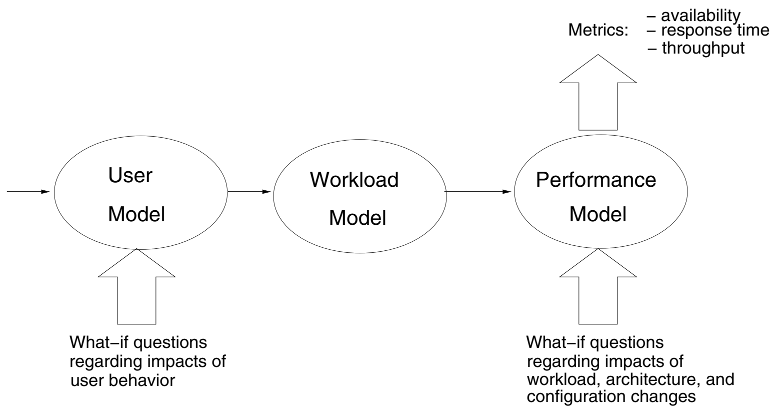
Metrics:
– availability
– response time
– throughput
User Model
Workload Model
Performance Model
What–if questions regarding impacts of user behavior
What–if questions regarding impacts of workload, architecture, and configuration changes

Fig. 2. Customer, Workload, and Resource Models.

for investing in the necessary upgrades. The personnel plan determines what changes in the support personnel size and structure must be made in order to accommodate changes in the system.

3 Models for Capacity Planning

Models play a central role in capacity planning. In the methodology discussed here, we consider two types of models: performance model and workload model. In Web environments, user models are important to provide information to workload models. Figure 2 shows the relationship between user model, workload model, and performance model.

Each Web service request (e.g., a credit card authorization or a search) may exercise the site's resources in different manners. Some services may use large amount of processing time from the application server while others may concentrate on the database server. Other service may demand high network bandwidth, such as requests for streaming media services. Different users exhibit different navigational patterns and, as a consequence, invoke services in different ways with different frequencies. For instance, in an e-business service, some customers may be considered as heavy buyers while others, considered occasional buyers, would spend most of their time browsing and searching the site. Understanding the customer behavior is critical for characterizing the workload as well as to an adequate sizing of the site's resources. Models of user behavior can be quite useful. In addition to characterizing navigational patterns within sessions, one needs to characterize the rate at which sessions of different types are started. This gives us an indication of the workload intensity. Workload models provide input parameters for performance models, that predict the system behavior for that specific workload.

Customer (i.e. user) models capture elements of user behavior in terms of navigational patterns, e-business functions used, frequency of access to the various e-business functions, and times between access to the various services offered by the site. A customer model can be useful for navigational and workload prediction.

- Model User Navigational Patterns for Predictive Purposes. By building models, one can answer what-if questions regarding the effects on user behavior due to site layout changes or content redesign.
- Capture Workload Parameters. If the only purpose of a customer model is to generate a workload model to be used as input to a resource model, then it is not necessary to use a detailed model.

Workload models describe the workload of an Web service in terms of workload intensity (e.g., transaction arrival rates) and service demands on the various resources (e.g., processors, I/O subsystems, networks) that make up the site. The workload model can be derived from the customer model as shown in [16]. Performance models represent the various resources of the site and captures the effects of the workload model on these resources. A performance model can be used for predictive purposes to answer what-if questions regarding performance impacts due to changes in configuration, software and hardware architecture, and other parameters. A performance model is used to compute the values of metrics such as response time, throughput, and business-oriented metrics such as revenue throughput.

4 Workload Models

A workload model is a representation that mimics the real workload under study. Although each system may require a specific approach to characterize and generate a workload model, there are some general guidelines that apply well to all types of systems [4]. The common steps to be followed by any workload characterization include: (1) specification of a point of view from which the workload will be analyzed, (2) choice of the set of parameters that captures the most relevant characteristics of the workload for the purpose of capacity planning, (3) monitoring the system to obtain the raw performance data, (4) analysis and reduction of performance data, (5) construction of a workload model, and (6) verification that the model does capture all the important performance information.

Graphs are also used to represent workloads. For example, a graph-based model can be used to characterize Web sessions and generate information for constructing workload models. This section concentrates on models that represent the behavior of users (i.e, customers). User models capture elements of user behavior in terms of navigational patterns, Web service functions used, frequency of access to the various functions, and times between access to the various services offered by the site. Two different types of models are commonly used in the capacity planning methodology.

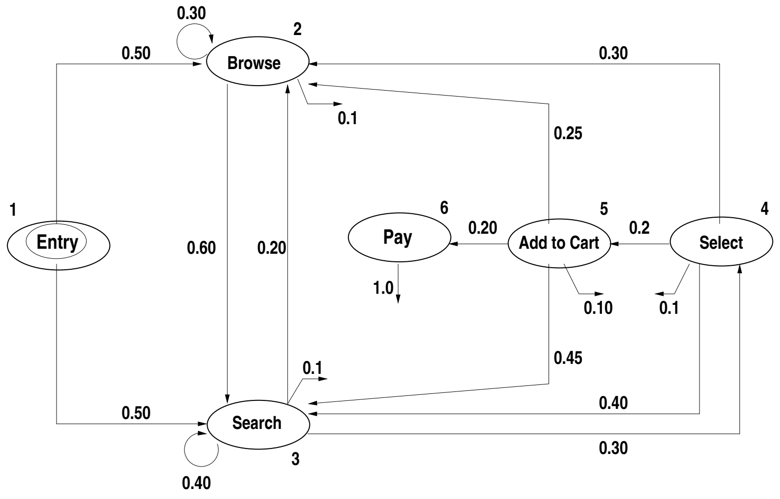

Fig. 3. The Customer Behavior Model Graph

4.1 Customer Graph Behavior Model

In Web-based environments, users interact with the site through a series of consecutive and related requests, called *sessions*. It has been observed that different customers exhibit different navigational patterns. The Customer Behavior Graph Model (CBMG), introduced in [15,17], can be used to capture the navigational pattern of a customer through an e-business site. This pattern includes two aspects: a transitional and a temporal one. The former determines how a customer moves from one state (i.e., an e-business function) to the next. This is represented by the matrix of transition probabilities. The temporal aspect has to do with the time it takes for a customer to move from one state to the next. This time is measured from the server's perspective and is called *server-perceived think time* or just think time. This is defined as the average time elapsed since the server completes a request for a customer until it receives the next request from the same customer during the same session. A think time can be associated with each transition in the CBMG.

So, a CBMG can be defined by a pair (P, Z) where $P = [p_{i,j}]$ is an $n \times n$ matrix of transition probabilities between the n states of the CBMG and $Z = [z_{i,j}]$ is an $n \times n$ matrix that represents the average think times between the states of the CBMG. Recall that state 1 is the Entry state and n is the Exit state.

Consider the CBMG of Figure 3. This CBMG has seven states; the Exit state, state seven, is not explicitly represented in the figure. Let V_j be the average number of times that state j of the CBMG is visited for each visit to the e-business site, i.e., for each visit to the state Entry. Consider the Add to Cart state. We can see that the average number of visits (V_{Add}) to this state is equal to the average number of visits to the state Select (V_{Select}) multiplied by the probability (0.2) that a customer will go from Select to Add Cart. We can then write the relationship

$$V_{\text{Add}} = V_{\text{Select}} \times 0.2. \tag{1}$$

Consider now the Browse state. The average number of visits (V_{Browse}) to this state is equal to the average number of visits to state Search (V_{Search}) multiplied by the probability (0.2) that a customer will go from Search to Browse, plus the average number of visits to state Select (V_{Select}) multiplied by the probability (0.30) that a customer will go from Select to Browse, plus the average number of visits to the state Add to Cart (V_{Add}) multiplied by the probability (0.25) that a customer will go from Add to Cart to Browse, plus the average number of visits to the state Browse (V_{Browse}) multiplied by the probability (0.30) that a Customer will remain in the Browse state, plus the number of visits to the Entry state multiplied by the probability (0.5) of going from the Entry state to the Browse state. Hence,

$$\begin{aligned} V_{\text{Browse}} = V_{\text{Search}} \times 0.20 + V_{\text{Select}} \times 0.30 + V_{\text{Add}} \times 0.25 + \\ V_{\text{Browse}} \times 0.30 + V_{\text{Entry}} \times 0.5. \end{aligned} \tag{2}$$

So, in general, the average number of visits to a state j of the CBMG is equal to the sum of the number of visits to all states of the CBMG multiplied by the transition probability from each of the other states to state j. Thus, for any state j ($j = 2, \cdots, n-1$) of the CBMG, one can write the equation

$$V_j = \sum_{k=1}^{n-1} V_k \times p_{k,j}, \tag{3}$$

where $p_{k,j}$ is the probability that a customer makes a transition from state k to state j. Note that the summation in Eq. (3) does not include state n (the Exit state) since there are no possible transitions from this state to any other state. Since $V_1 = 1$ (because state 1 is the Entry state), we can find the average number of visits V_j by solving the system of linear equations

$$V_1 = 1 \tag{4}$$

$$V_j = \sum_{k=1}^{n-1} V_k \times p_{k,j} \quad j = 2, \cdots, n-1. \tag{5}$$

Note that $V_n = 1$ since, by definition, the Exit state is only visited once per session.

Useful metrics can be obtained from the CBMG. Once we have the average number of visits (V_j) to each state of the CBMG, we can obtain the average session length as

$$\text{AverageSessionLength} = \sum_{j=2}^{n-1} V_j. \tag{6}$$

Table 3. Using the CVM to Characterize a Session

	Session 1	Session 2	Session 3
Home	1	2	3
Browse	4	8	4
Search	5	5	3
Login	0	1	1
Pay	0	0	1
Register	0	0	1
Add to Cart	0	2	1
Select	3	3	2

For the the visit ratios of Fig. 3, the average session length is

$$\begin{aligned}\text{AverageSessionLength} &= V_{\text{Browse}} + V_{\text{Search}} + V_{\text{Select}} + V_{\text{Add}} + V_{\text{Pay}} \\ &= 2.498 + 4.413 + 1.324 + 0.265 + 0.053 \\ &= 8.552. \end{aligned} \tag{7}$$

The buy to visit Ratio is simply given by V_{Pay}.

Each customer session can be represented by a CBMG, that can be derived from HTTP logs. "Similar" sessions can be clustered to represent each group by a single CBMG. The goal is to characterize the workload by a relatively small and representative number of CBMGs as opposed to having to deal with thousands or even hundreds of thousands of CBMGs. Procedures and algorithms for clustering CBMGs are described in details in [15]

4.2 The Customer Visit Model (CVM)

An alternate and less detailed representation of a session would entail representing a session as a vector of visit ratios to each state of the CBMG. The visit ratio is the number of visits to a state during a session.

Table 3 shows an example of three sessions described by the number of visits to each state of the CBMG. Note that states Entry and Exit are not represented in the CVM since the number of visits to these states is always one. Session 1 in the table represents a session of a customer who browsed through the site, did a few searches, but did not login or buy anything. In Session 2, the customer logged in but did not need to register because it was an already registered customer. This customer abandoned the site before paying even though two items had been added to the shopping cart. Finally, Session 3 represents a new customer who registers with the site, adds one item to the shopping cart, and pays for it.

The CVM is then a set of vectors (columns in Table 3) that indicate the number of times each of the functions supplied by the e-business site are executed. For example, Session 1 would be represented by the vector (1, 4, 5, 0, 0, 0, 0, 3) and Session 2 by the vector (2, 8, 5, 1, 0, 0, 2, 3).

The CBMG is a state transition graph, in which the nodes correspond to states in the session (e.g., browsing, searching, selecting, checking out, and paying) and the arcs correspond to transitions between states. Probabilities are associated with transitions as in a Markov Chain. A Customer Visit Model (CVM) represents sessions as a collection of session vectors, one per session. A session vector $V_j = (v_1, v_2, \cdots v_m)$ for the j^{th} session indicates the number of times, that each of the different functions (e.g., search, browse, add to cart, etc) were invoked during the session. To be able to perform capacity planning studies of a Web service, one needs to map each CBMG or CVM resulting from the workload characterization process described above to IT resources. In other words, one has to associate service demands at the various components (e.g., processors disks and network) with the execution of the functions [16].

5 Performance Models

Performance models represent the way system's resources are used by the workload and capture the main factors determining system performance. These models use information provided by workload models and system architecture description. Performance models are used to compute both traditional performance metrics such as response time, throughput, utilization, and mean queue length as well as innovative business-oriented performance metrics, such as revenue throughput or lost-revenue throughput. Basically, performance models can be grouped into two categories: analytic and simulation models. Performance models help us understand the quantitative behavior of complex systems, such as electronic business applications, e-government, and entertainment. Performance models have been used for multiple purposes in systems.

- In the infrastructure design of Web-based applications, various issues call for the use of models to evaluate system alternatives. For example, a distributed Web server system is any architecture consisting of multiple Web server hosts distributed on a LAN, with some sort of mechanism to distribute incoming requests among the servers. So, for a specific type of workload, what is the most effective scheme for load balancing in a certain distributed Web server system? Models are also useful for analyzing document replacement policies in caching proxies. Bandwidth capacity of certain network links can also be estimated by performance models. In summary, performance models are an essential tool for studying resource allocation problems in the context of Web services.
- Most Web-based applications operate in multi-tiered environments. Models can be used to analyze performance of distributed applications running on three-tiered architectures, composed of Web servers, application servers and database servers.
- Performance tuning of complex applications is a huge territory. When a Web-based application presents performance problems, a mandatory step to solve them is to tune the underlying system. This means to measure the system and try to identify the sources of performance problems: application

design, lack of capacity, excess of load, or problems in the infrastructure (i.e., network, servers, ISP). Performance models can help find performance problems by answering what-if questions as opposed to making changes in the production environment.

Parameters for queuing network (QN) models are divided into the following categories. (1) System parameters specify the characteristics of a system that affect performance. Examples include load-balancing disciplines for Web server mirroring, network protocols, maximum number of connections supported by a Web server, and maximum number of threads supported by the database management system. (2) Resource parameters describe the intrinsic features of a resource that affect performance. Examples include disk seek times, latency and transfer rates, network bandwidth, router latency, and processor speed ratings. (3) Workload parameters that are derived from workload characterization and are divided into types: workload intensity and service demand. Workload intensity parameters provide a measure of the load placed on the system, indicated by the number of units of work that contend for system resources. Examples include the number of requests/sec submitted to the database server and number of sales transactions submitted per second to the credit card service. Workload service demand parameters specify the total amount of service time required by each basic component at each resource. Examples include the processor time of transactions at the database server, the total transmission time of replies from the database server and the total I/O time at the streaming media server.

6 Concluding Remarks

Capacity planning techniques are needed to avoid the pitfalls of inadequate capacity and to meet users' performance expectations in a cost-effective manner. This tutorial provides the foundations required to carry out capacity planning studies. Planning the capacity of Web services requires that a series of steps be followed in a systematic way. This paper gives an overview of the main steps of the quantitative approach to analyze Web services. The main steps are based on two models: a workload model and a performance model. The two models can be used in capacity planning projects to answer typical what-if questions, frequently faced by managers of Web services.

References

1. V. Almeida and D. Menascé, "Capacity Planning: an essential tool for managing Web services", IEEE IT Pro, Vol. 4, Issue 4, July-August, 2002.
2. M. Arlitt and C. Williamson, "Internet Web Servers: workload characterization and performance implication", in *IEEE/ACM Trans. on Networking*, October 1997.
3. M. Arlitt, D. Krishnamurthy, and J. Rolia, "Workload Characterization and Performance Scalability of a Large Web-based Shopping System", in *ACM Transactions on Internet Technologies*, Vol.1, No. 1, Aug. 2001.

4. M. Calzarossa and G. Serazzi, "Workload Characterization: A Survey," *Proceedings of the IEEE*, Vol. 81, No. 8, August 1993.
5. M. Crovella and A. Bestravos, " Self-Similarity in the World Wide Web: evidence possible causes", in *IEEE/ACM Transactions on Networking*, 5(6):835–846, December 1997.
6. P. Denning and J. Buzen, "The operational analysis of queuing network models", *Computing Surveys*, Vol. 10, No. 3 , September 1978, pp. 225-261.
7. R. Jain, *The Art of Computer Systems Performance Analysis*. New York: Wiley, 1991.
8. K. Kant and Y. Won "Server Capacity Planning for Web Traffic Workload", in *IEEE Trans. on Knowledge and Data Engineering*, September 1999.
9. D. Krishnamurthy and J. Rolia, "Predicting the Performance of an E-Commerce Server: Those Mean Percentiles," in *Proc. First Workshop on Internet Server Performance*, ACM SIGMETRICS 98, June 1998.
10. A. Law and W. Kelton, *Simulation Modeling and Techniques*. 2nd ed. New York: McGraw-Hill, 1990.
11. D. Levine, P. Ramsey, R. Smidt, *Applied Statistics for Engineers and Scientists: Using Microsoft Excel & MINITAB*, Upper Saddle River, Prentice Hall, 2001,
12. J. Martinich, *Production and Operations Management : An Applied Modern Approach*, John Wiley & Sons, 1996.
13. D. A. Menascé, V. A. F. Almeida, and L. W. Dowdy, *Capacity Planning and Performance Modeling: From Mainframes to Client-Server Systems*. Upper Saddle River, NJ: Prentice Hall, 1994.
14. D. A. Menascé, D. Dregits, R. Rossin, and D. Gantz, A federation-oriented capacity management methodology for LAN environments, *Proc. 1995 Conf. Comput. Measurement Group*, Nashville, TN, Dec. 3–8, 1995,
15. D. A. Menascé, V. Almeida, R. Fonseca, and M. Mendes, "A Methodology for Workload Characterization for E-Commerce Servers", *Proc. 1999 ACM Conference in Electronic Commerce*, Denver, 1999.
16. D. A. Menascé and V. A. F. Almeida, *Scaling for E-Business: technologies, models, performance and capacity planning*, Prentice Hall, Upper Saddle River, 2000.
17. D. A. Menascé, V. A. F. Almeida, R. Fonseca, and M. A. Mendes, "Business-oriented Resource Management Policies for E-Commerce Servers," *Performance Evaluation*, September 2000.
18. D. A. Menascé, V. Almeida, R. Fonseca, R. Riedi, F. Ribeiro, and W. Meira Jr., "In Search of Invariants for E-Business Workloads ", *Proc. 2000 ACM Conference in Electronic Commerce*, Minneapolis, 2000.
19. D. A. Menascé and V. A. F. Almeida, *Capacity Planning for Web Services: metrics, models and methods*, Prentice Hall, Upper Saddle River, 2002.
20. V. Paxson and S. Floyd, "Wide area traffic: The failure of Poisson modeling," *IEEE/ACM Transactions on Networking* **3**, pp. 226–244, 1995.