Information Knowledge Systems Management 9 (2010) 99–116
DOI 10.3233/IKS-2010-0160
IOS Press

Models of dynamic relations among service activities, system state and service quality on computer and network systems[1]

Nong Ye[a,*], Steve Yau[a], Dazhi Huang[a], Mustafa Baydogan[a], Billibaldo M. Aranda[a], Auttawut Roontiva[a] and Patrick Hurley[b]
[a]*School of Computing, Informatics, Decision Systems Engineering, Arizona State University, Tempe, AZ 85287-8809, USA*
[b]*U.S. Air Force Research Laboratory/RIGA, Rome, NY 13441-4505, USA*

Abstract: Service quality on computer and network systems has become increasingly important as many conventional service transactions are moved online. Service quality of computer and network services can be measured by the performance of the service process in throughput, delay, and so on. On a computer and network system, competing service requests of users and associated service activities change the state of limited system resources which in turn affects the achieved service quality. Modeling dynamic relations of service activities, system state and service quality is required to determine if users' service requests and requirements of service quality can be satisfied by the system with limited resources and how the system and service configuration can be adapted to meet service quality requirements. This paper presents our empirical study to establish activity-state-quality models for a voice communication service. We run experiments to collect system dynamics data under various service conditions and use statistical techniques to analyze experimental data and build activity-state-quality models. The results reveal four major types of dynamic relations among service activity parameters, system resource state, and the network throughput – a measure of achieved service quality for the voice communication service. Although delay-related measures are also important for voice data communication, they are not collected in this study. Five system state variables concerning the memory, CPU, process and IP resources are uncovered to be affected by service activity parameters significantly and be associated with the achieved service quality closely. We also obtain an insight about increasing the size of the buffer which holds voice data before transmission over the network to alleviate the workload on system resources and maintain the network throughout when the number of client requests and the client requirement in voice quality increase.

Keywords: Computer and network service, achieved service quality, service and system configuration, voice communication service, statistical analysis and modeling

1. Introduction

Service quality has been essential for conventional, off-line service transactions, and has become increasingly important as many conventional service transactions are moved online with computer and network systems providing services. Service quality of computer and network services can be measured by the performance of the service process in throughput, delay, and so on. Chen et al. [4] describe service quality requirements of various network applications for online services, e.g., web browsing, email, file transfer, audio and video broadcasting, audio and video on demand, audio and video conferencing,

[1]Approved for Public Release; Distribution Unlimited: 88ABW-2010-5243 dated 28 Sep 2010.
*Corresponding author. E-mail: nongye@asu.edu.

voice over IP, etc. Service quality requirements for those online services are specified using metrics on timeliness (e.g., response time, delay and jitter), precision (e.g., bandwidth and loss rate) and accuracy of services (e.g., error rate).

When competing service requests with specific service quality requirements come to a computer network system providing services, the system must determine if its limited system resources can satisfy these service requests at the required level of service quality and furthermore what service configuration, resource configuration and service-resource binding should be used for the achieved service quality [8, 15,17,25,26,28,31]. The competing service requests and resulting service activities change the state of limited system resources which in turn affects the achieved service quality [27]. Service quality is considered as a subset of service performance measures. For example, the increasing number of clients for a voice communication service is expected to produce more service activities which consume more system resources such as CPU, memory and network bandwidth and make these resources less available to each client, consequently leading to a decrease in the voice data throughput as a key measure of service quality for each client. Such dynamic relations of service activities, system state and service quality are the basis for determining and adapting service and system configurations to meet service quality requirements.

However, models of such activity-state-quality relations are not readily available from the design of computer and network systems which provides mostly algorithm-based operational models. Activity-state-performance dynamic models from existing studies focus mostly on individual resources (e.g., router, CPU, memory, and hard disk) and limited aspects of dynamic models. For examples, there are studies by

- Vazhkudai and Schopf [22] on regression models for relations between disk load variation and file transfer time,
- Kapadia et al. [9] on resource usage models in a computational grid environment,
- Ravindran and Hegazy [16] on regression models between the timeliness performance of periodic tasks and external and internal load parameters such as CPU utilization,
- Shivam et al. [19] on regression models of relations between varying assignments of computing, network and storage resources and application completion time, and
- Sun and Ifeachor [20] on models of relations between the playout buffer control and the quality of voice over IP.

In literature we did not find models of activity-state-quality dynamics during realistic operations of computer and network systems at a more comprehensive, system scale which account for a wide range of hardware and software resources (including CPU, memory, physical disk, caches, buffers, network interface, IP, TCP, UDP, Terminal Service, etc.), their interactions, their state changes with service activities, and their effects on the achieved service quality. Service and system configuration for service quality satisfaction should not simply consider the service effect on an individual resource or change the configuration of an individual resource because certain system resources (e.g., CPU and memory) often interact with and place constraints on one another. The achieved service quality depends on activity-state-quality dynamics at the system scale that takes into account effects of service activities on all system resources and service quality of all competing service processes/threads.

Due to the lack of models for activity-state-quality dynamics at a more comprehensive, system scale, existing studies on service and system configuration for service quality often bypass the issue of establishing models of activity-state-quality dynamic relations. Those studies focus only on the evaluation of service quality according to user-defined weights of service quality attributes without

getting into models of realistic activity-state-quality dynamics to account for how the performance level of various service quality attributes change dynamically with competing, dynamic service demands and the varying state, constraints and interactions of limited system resources. For example, Sha et al. [18] presents a quality-based web services selection model that selects the best web service based on a set of service quality attributes that are related to performance, price, availability and latency. The model first selects a set of candidate web services that match functional requirements of user web service requests. Each candidate web service then receives a utility score of service quality based on a weighted sum of the normalized attribute values of service quality for the selection of the best web service. However, the study does not include information about how the attribute values of service quality on performance, availability and latency are obtained in real time and how the attribute values of service quality change dynamically with competing service requests and with state and constraints of system resources. There are other studies [2,3,5,7,10–12,21,23,24,29,30] on service and system configuration for meeting service quality requirements, all without the specification of activity-state-quality models of system dynamics or the basis for determining if and how service requests can be satisfied with the given state of limited system resources to achieve the required service quality.

The lack of models for realistic activity-state-quality dynamic relations at the system scale produces a significant gap in bringing existing work on service configuration and adaptation to real-world applications. This paper presents our study to establish models of activity-state-quality dynamics models for a voice communication service as an illustration of how such models can be established empirically. We collect system dynamics data of service activities, resource state and achieved service quality with services running on a real computer and network system under various service conditions. We then analyze experimental data to uncover activity-state-quality relations and build activity-state-quality models.

In Section 2, we define cause-effect dynamics of service activities, resource state and service quality, which provides a framework for establishing activity-state-quality models. In Section 3, we describe the experiment to collect system dynamics data under various service conditions and system configurations for a voice communication service. In Section 4, we present the methodology to analyze the experimental data, uncover activity-state-quality relations, and build activity-state-quality models. In Section 5, we discuss the analytical and modeling results. Section 6 concludes the paper.

2. Cause-effect dynamics of service activities, resource state and service quality

Figure 1 illustrates the cause-effect dynamics of service activities, resource state, and service quality in a typical user-process-resource interaction on computer and network systems [28]. A service request from a user with service quality requirements calls for a service process which utilizes certain system resources and consequently changes the state of these resources. Changes of the resource state in turn affect the achieved service quality of the service process. We refer to models capturing such cause-effect dynamics of service activities (A), resource state (S) and service quality (Q) as Activity-State-Quality (ASQ) models.

3. An experiment to collect system dynamics data for a voice communication service

Computers with a Windows operating system are used in our experiment. In the experiment, a voice communication service is set up in an online radio broadcasting context in which multiple clients

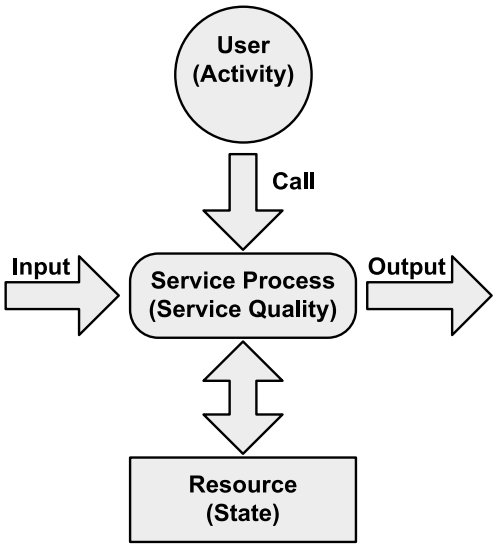


Fig. 1. The cause-effect dynamics of service activities, resource state and service quality in the user-process-resource interaction.

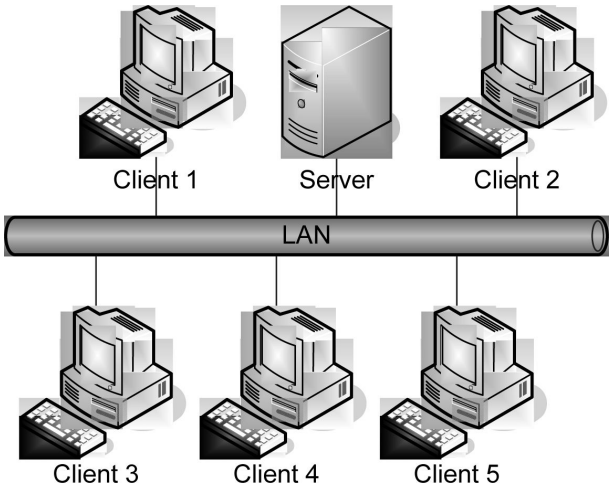


Fig. 2. The computer and network setup for the experiment.

simultaneously request and receive the voice communication service from a server which sends out real-time voice data streams of various quality levels controlled by the data sampling rate. Multiple clients run on their own computers with only one client on one computer. Figure 2 shows the computer and network setup with one server and five clients. Each computer has 1 GB memory and Intel Pentium4 2.2 GHz. Both client and server computers are running Windows XP with SP2. The voice communication service is running on Internet Information Service 6.0. The voice communication service is developed by converting an open-source Video Conferencing software package [1] into Web Services using C# in .NET. The computer network system for the experiment stands alone without connections with any other computer and network system to avoid interferences.

Windows operating systems provide Windows performance objects (Microsoft, 2003) which cover various aspects of process activities, resource state and service performance for many objects of resources

Table 1
Service and system parameters and their levels in the experiment

	Sampling rate (Sa)	Number of clients (C)	Buffer size (B)
Level 1	44,100 Hz	1	16 Kbytes
Level 2	88,200 Hz	2	24 Kbytes
Level 3	132,300 Hz	3	32 Kbytes
Level 4	176,400 Hz	4	40 Kbytes
Level 5	220,500 Hz	5	48 Kbytes

and processes. Some examples of performance objects are Physical Disk, Memory, System, Process, Processor, IP, UDP, and TCP. Each object has a number of counters which reflect various aspects of activity, state and performance of the object. For example, the Memory object has a counter, *Available Bytes*, to show one aspect of the memory state. The IP object has a counter, *Fragmented Datagrams/sec*, to reflect the data transmission performance at the IP level. We use Windows performance objects to collect system dynamics data of various resources and processes which allow us to look into computer and network resources, their interactions, and their effects on service quality at the system scale.

To collect system dynamics data reflecting varying activity-state-quality dynamics for the voice communication service, we introduce various levels of service activities and system configurations in our experiment. The voice communication service is a communication-intensive application. Service quality attributes for the voice communication service concern mainly the throughput and delay of voice data transmission. Hence, we want to select parameters of service activities and system configurations that are expected to affect the achieved service quality of the voice data communication service. Two parameters of service activities are selected for the experiment: 1) the sampling rate (Sa) which is used to record the voice data stream and thus determines the quality of voice data, and 2) the number of clients (C). These two parameters are used to reflect various levels of service quality demands and service activities in this voice communication service and to create competing service requests with service quality requirements that are expected to affect the achieved service quality of the voice communication service. The parameter of system configuration, the size of the buffer (B) on the server for storing the voice data before transmitting the data to a client over the network, is selected because the buffer size directly affects the throughput and delay of voice data transmission.

As shown in Table 1, each parameter has five levels in the experiment such that we collect data with sufficient granularity for data analysis to obtain ASQ relations and models. Hence, we have 125 ($5 \times 5 \times 5$) experimental conditions. The five levels of the sampling rate are denoted by Sa1, Sa2, Sa3, Sa4, and Sa5. The five levels of the number of clients are denoted by C1, C2, C3, C4, and C5. The five levels of the buffer size are denoted by B1, B2, B3, B4, and B5.

ASQ models represent cause-effect relations among service activities (A), state of resources (S), and service quality (Q) of service processes. The two parameters of the sampling rate and the number of clients directly drive service activities and are considered as A variables in the ASQ models. The parameter of the buffer size affects the state of the buffer. With complex interactions of the buffer with other system resources during the process of the voice communication service, the parameter of the buffer size is also expected to affect the state of other system resources. Hence, the parameter of the buffer size is also considered as an A variable that is expected to affect the state of system resources and thus the achieved service quality.

System dynamics data of resource state and service quality of the voice communication service for S and Q variables in ASQ models are collected using eight Windows performance objects, including Physical Disk, Memory, System, Process, IP, UDP, TCP, and Server. These objects are selected because they are expected to be involved in the voice communication service. A number of variables collected

through the eight Windows performance objects are related to the throughput and delay of voice data transmission – the service quality attributes of interest for the voice communication service. For example, *Fragmented Datagrams/sec_IP*, *Datagrams Sent/sec_IP* and *Datagrams Sent/sec_UDP* are all related to the throughput of voice data transmission from the server to the clients. The variable used to measure the throughput of voice data transmission will be selected in Section 5 based on the analytical results on effects of the activity variables on the collected state and QoS variables. The delay of voice data transmission includes 1) the time of generating and sending voice data on the server which is closely related to the throughput of voice data transmission on the server, and 2) the transmission time of voice data over the network from the server to a client which is not directly measured and collected in our experiment. Hence, the delay of voice data transmission is not considered in this study. In our future research, we may also include additional parameters and variables such as the size of the receiving buffer, network delay and its variability, and a larger number of clients.

The experimental run under each of the 125 experimental conditions includes one minute of the voice communication service for a given level of the three service parameters. Sixty data observations under each experimental condition are collected with the sampling rate of 1 observation per second. The data is recorded in log files. Experimental data are collected on the server since the server data reflects the effect of multiple clients requesting the service. The analysis of the data is performed on the server data to obtain ASQ relations and models.

Some variables from Windows performance objects give accumulated values of system state or performance that increase over time, or change their values depending on the operation history. As a result, those variables are affected by the time when various service conditions are run. One example of such variables is the *System Up Time* counter of the System object whose values increase over time. Hence, the time when a service condition is run has a confounding effect on those variables with various service conditions in the experiment. Such variables need to be removed from further data analysis that determines the effect of service conditions. To identify and remove such variables, we run a small-scale experiment involving only 3 of the 125 experimental conditions. The three experimental conditions in the small-scale experiment are the service condition with Sa1, C1 and B1, the service condition with Sa3, C3 and B3, and the service condition with Sa5, C5 and B5. The small-scale experiment includes two runs, each of which has three service conditions and two no-service conditions, as follows:

- Run 1: 1) no service at the beginning, 2) Sa1C1B1, 3) Sa3C3B3, 4) Sa5C5B5, and 5) no service at the end;
- Run 2 with the reversed order to that in Run 1: 1) no service, 2) Sa5C5B5, 3) Sa3C3B3, 4) Sa1C1B1, and 5) no service at the end.

The experimental data from these two runs with different orders of service conditions and no-service conditions allow us to analyze and examine possible confounding effects of service conditions and times when they are run on some variables and thus remove those variables. The remaining variables are kept for further data analysis to determine the effect of various service conditions using the data from the large-scale experiment with the full set of 125 service conditions.

4. Methodology of data analysis and modeling

In this section, we first describe the steps of analyzing the data from the small-scale experiment to identify and remove variables whose values depend on the time when a service condition is run. Then we present the methodology of analyzing the data from the large-scale experiment to obtain the ASQ relations and models. The data modeling technique to build ASQ models is also described.

Table 2
The recording of the Mann-Whitney test results for data screening using the data of the small-scale experiment

		Service condition		
Run	In Comparison with	Sa1C1B1	Sa3C3B3	Sa5C5B5
1	No-service at the beginning			
	No-service at the end			
2	No-service at the beginning			
	No-service at the end			

Table 3
Tukey's test results for the effect of the sampling rate on *%Committed Bytes in Use_Memory*

The level of the	The average value of *%committed bytes*	Group				
sampling rate	*in use_memory*	1	2	3	4	5
Sa1	11.77	***				
Sa2	12.09		***			
Sa3	12.40			***		
Sa4	12.64				***	
Sa5	12.91					***

4.1. Data Screening Using the Data of the Small-Scale Experiment

The small-scale experiment includes two runs with different orders of three service conditions and two no-service conditions. For each run and each variable of the Windows performance objects data, we perform a Mann-Whitney test on the experimental data to compare each of the three service conditions (Sa1C1B1, Sa3C3B3, and Sa5C5B5) with each of the two no-service conditions (at the beginning and at the end). The Mann-Whitney test [13,28] is selected because the test uses a nonparametric statistic based on ranks and thus depends little on the probability density distribution of the data. Generally the Mann-Whitney test is as powerful as its typical parametric counterpart, the two-sample t test. The statistical software, Statistica7, is used to perform the Mann-Whitney test. Each Mann-Whitney test determines whether or not the effect of the service condition is significantly different from that of the no-service condition on a given variable. If there is a statistically significant difference, the specific difference (increase or decrease) of the service condition from the no-service condition is noted. Table 2 shows how the Mann-Whitney test results for each variable are recorded. Each cell in Table 2 has one of the three values: increase, decrease, or no significant difference.

For each variable, if two rows of values for each run as recorded in Table 3 are not the same, the variable is removed. This is based on the consideration that the effect of each service condition on the variable should be the same when compared with the two no-service conditions at the beginning and at the end if the variable is not affected by the time and order of running a service condition and a no-service condition. If two rows of values for each run are the same but they are different from two rows for another run, the variable is also removed because the difference indicates that the variable is affected by different orders of running service conditions or the time of running a given service condition. Only the remaining variables after the data screening are considered for further data analysis using the data of the large-scale experiment with the full set of 125 service conditions. Hence, the state and quality variables of the Windows performance objects in the next section refer to only the remaining variables after the data screening described in this section.

4.2. Data analysis and modeling using the data of the large-scale experiment

The data from the large-scale experiment with the full set of 125 service conditions is used to uncover the relations of the service activity parameters with the resource state variables and service quality variables collected from the Windows performance objects. The following statistical data analyses are carried out.

1. A-SQ relation discovery and categorization. For each state or quality variable, the Analysis of Variance (ANOVA) in Statistica7 is performed with the three activity parameters of the sampling rate, the number of clients, and the buffer size (A's) as the independent variables and the state or quality variable (S or Q) as the dependent variable to determine the effects of A's on S or Q. For example, the increasing level of the sampling rate may cause an increase in the used memory. If ANOVA results reveal a significant effect of one or more A variables on a S or Q variable, the Tukey's honest significant difference (HSD) test in Statistica7 is performed to determine how different levels of one or more A variables affect the S or Q variable. Then similar A-S or A-Q relations are grouped together and categorized into a certain type of A-SQ relations.
2. Development of the ASQ relation map. The representative A-S relations for each category of A-SQ relations from Step 1 are selected to construct the ASQ relation map which includes the three A variables, the selected S variables, and the Q variable measuring the network throughput of the voice communication service. In the ASQ relation map, each A, S, or Q variable is represented as a node. There is a directed link between an A variable and a Q variable to represent the A-S relation between this A variable and this Q variable. There is also a directed link between each S variable with the Q variable. For example, there may be a directed link between the sampling rate (an A variable) and the used memory (a S variable) and a directed link between the used memory (a S variable) and the network throughput variable (a Q variable).
3. ASQ modeling. For each S variable in the ASQ relation map, a regression model is built to give the quantitative relation of the S variable with one or more A variables. A regression model is also built to give the quantitative relation of the Q variable with the related S variables. We use the multiple regression procedure in Statistica to build a linear regression model if a linear regression model fits the data well; otherwise, a nonlinear regression technique such as the multivariate Adaptive Regression Splines (MARS) technique [6] in the *earth* package of the R software (http://cran.r-project.org/web/packages/earth/earth.pdf) to build a nonlinear regression model.

5. Results and discussions

This section describes and discusses the results of data screening and further data analysis and modeling. In the following text, a state or quality variable is denoted by *CounterName_ObjectName*.

5.1. Data screening results

The data screening steps described in Section 4.1 produce the 106 remaining state and quality variables whose values are not affected by the time of running service conditions.

Table 4
Five categories of A-SQ relations and the state and quality variables in each category

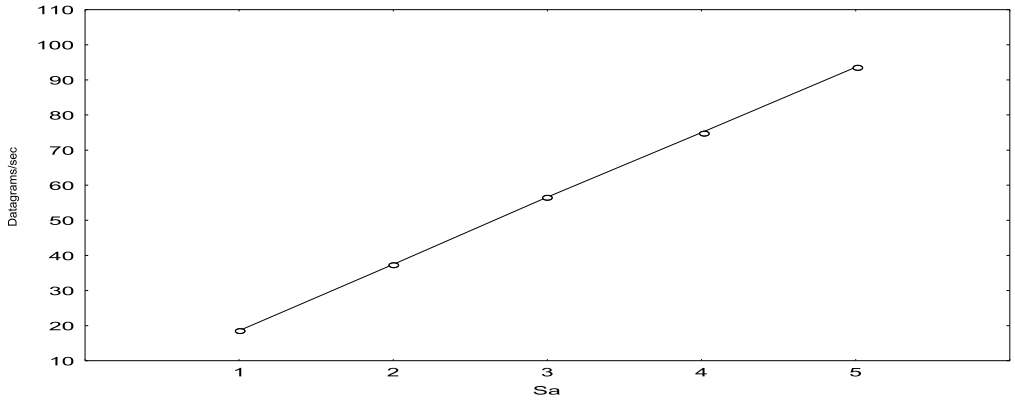
Category	State and quality variables
1. Increase with Sa and C and B	% Committed Bytes In Use_Memory, Committed Bytes_Memory
2. Increase with Sa and C, decrease with B	% Privileged Time_Process, % Processor Time_Process, % User Time_Process, Context Switches/sec_System, Datagrams/sec_UDP, Datagrams/sec_IP, Datagrams Sent/sec_UDP, Datagrams Sent/sec_IP, File Control Operations/sec_System, File Control Bytes/sec_System, Fragmented Datagrams/sec_IP, Fragments Created/sec_IP, IO Other Operations/sec_Process, IO Other Bytes/sec_Process
3. Increase with C, stable with Sa except at one end, inverse-U change with B	Thread Count_Process, Page Faults/sec_Memory
4. Decrease with Sa, C and B	Available Bytes_Memory, Available KBytes_Memory, Available MBytes_Memory
5. Inconsistent change with Sa, C and B and sometimes strong interaction of Sa, C and B	% Registry Quota In Use_System, Avg. Disk sec/Transfer_Physical Disk, Datagrams Received Delivered/sec_IP, Processor Queue Length_System, Datagrams Received/sec_IP, Datagrams Received/sec_UDP, Page Faults/sec_Process

5.2. Results of A-SQ relations

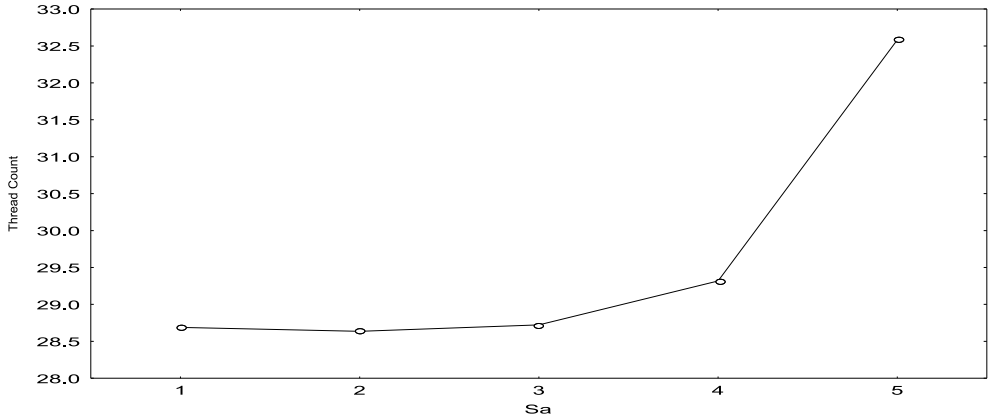
ANOVA for each of the remaining state and QoS variables from Section 5.1 reveal 28 S and Q variables that have a significant effect (with the p-value less than 0.05) of one or more A variables. For each significant effect on each S or Q variable, the Tukey's HSD test is performed to determine how different levels of one or more A variables affect the S or Q variable. For example, all the three A variables (the sampling rate, the number of clients, and the buffer size) have significant effects on the S variable, *%Committed Bytes in Use_Memory*, based on the ANOVA results. Table 3 gives the Tukey's test results for the effect of the sampling rate on*%Committed Bytes in Use_Memory.* Table 3 shows the groups of levels in the sampling rate that are significantly different from each other in their effects on *%Committed Bytes in Use_Memory*. If two levels marked by "***" fall into two different groups, there is a statistically significant difference between these two levels. For example, the Sa1 level of the sampling rate falls into group 1, whereas the Sa2 level of the sampling rate falls into group 2, indicating that the difference between Sa1 and Sa2 produces the statistically significant different effects on *%Committed Bytes in Use_Memory*. That is, the increase of the sampling rate from Sa1 to Sa2 significantly increases *%Committed Bytes in Use_Memory*. Table 3 indicates that *%Committed Bytes in Use_Memory* increases as the sampling rate increases. Tukey's test results for all the effects on *%Committed Bytes in Use_Memory* show that *%Committed Bytes in Use_Memory* increases as the sampling rate, the number of clients and the buffer size increase. Hence, *%Committed Bytes in Use_Memory* is categorized into a group of the S and Q variables with the A-SQ relations characterized as increasing with Sa, C and B (see Category 1 in Table 4).

Based on the Tukey's test results, the 28 S and Q variables, which show significant effects of one or more A variables based on the ANOVA results, are grouped into five categories of the A-SQ relations as shown in Table 4.

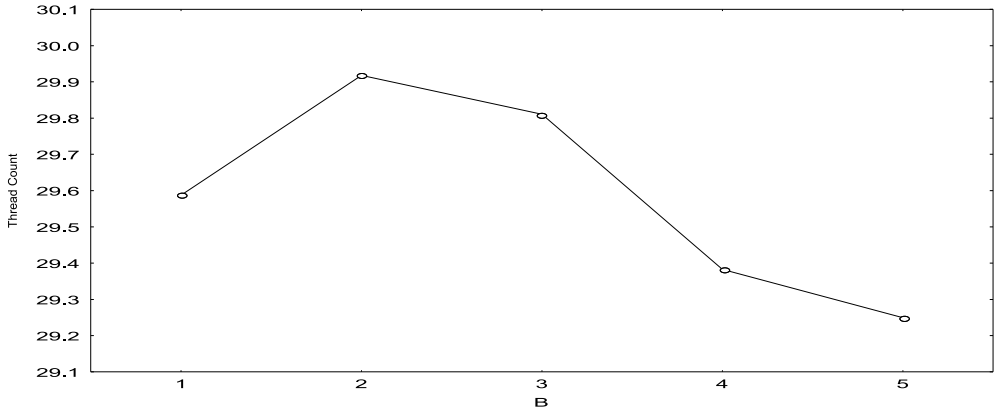
Category 1 has 2 state variables from the Memory object: *% Committed Bytes In Use_Memory* and *Committed Bytes_Memory*. These two state variables reflect the state of the memory concerning the memory consumption. The values of each S variable in this category increase as the sampling rate, the number of clients and the buffer size increase as shown in Fig. 3a. Figure 3a plots the average values of the state variable, *Datagrams/sec_IP*, for the five levels of the sampling rate. The A-S relation of the state variables increasing with the three service parameters (the A variables) in this category indicates that the increasing level of the sampling rate, the number of clients and the buffer size in the voice



a. "Increase with Sa" of *Datagrams/sec_IP* for 5 levels of the sampling rate.

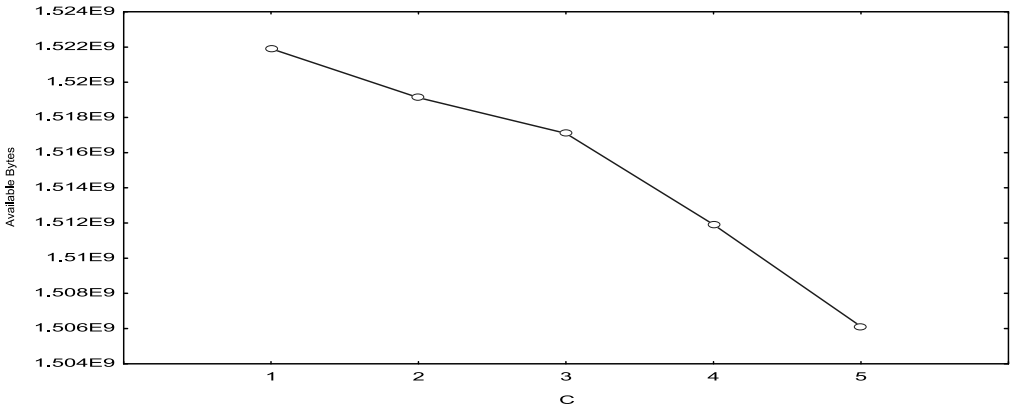


b. "Stable with Sa except at one end" of *Thread Count_Process* for 5 levels of the sampling rate.

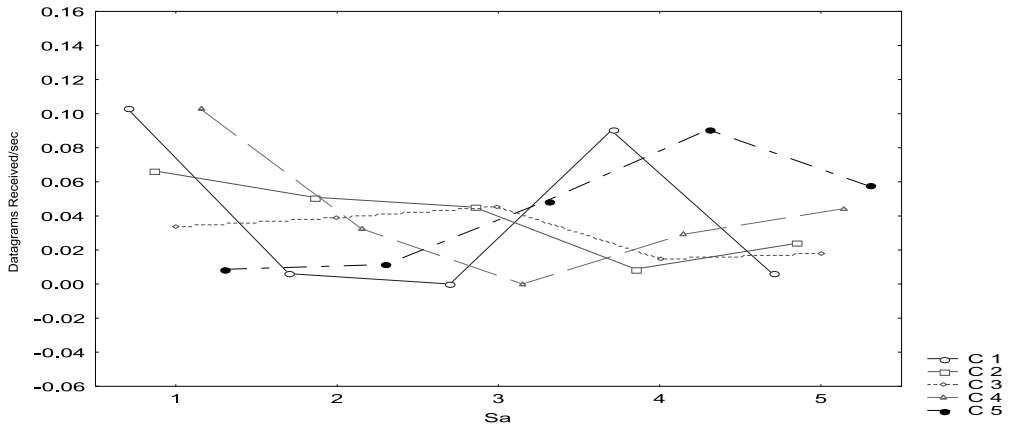


c. "Inverse-U change with B" of *Thread Count_Process* for 5 levels of the buffer size.

Fig. 3. An illustration of relationships between activity variables and state-quality variables in the five categories.



d. "Decrease with C" of *Available Bytes_Memory* for 5 levels of the number of clients.



e. "Inconsistent change or strong interaction" of *Datagrams Received/sec_IP* for 25 levels of interactions between the sampling rate and the number of clients.

Fig. 3. continued.

communication service demands for more system memory. This A-S relation can be used to guide the adaptation of service and system configuration for the achieved service quality. When the higher level of service quality (e.g., a larger sampling rate for a better voice quality) is needed or more clients need to be served, the memory resource can be adapted by increasing the reserved memory, allocating more memory to the voice communication service, or through other means. The adaptation can be done through APIs, such as the *SetProcessWorkingSetSize* or *SetInformationJobObject* functions in Windows.

Category 2 includes 14 variables that reflect the state and/or performance of the Process, System, and network-related software objects concerning the use of the CPU and network interface resources at the hardware level. Specifically, the two variables from the UDP object (*Datagrams/sec_UDP* and *Datagrams Sent/sec_UDP*) and four variables from IP object (*Datagrams/sec_IP*, *Datagrams Sent/sec_IP*, *Fragmented Datagrams/sec_IP* and *Fragments Created/sec_IP*) reflect the amount of network communication generated mainly by the service process of voice communication. Three variables from the Process object (*%Privileged Time_IP*, *%Processor Time_IP*, and *%User Time_IP*) reflect the CPU consumption by the service process of voice communication. The other two variables from the Process object (*IO Other Operations/sec_Process* and *IO Other Bytes/sec_Process*) reflect IO control operations

by the service process of voice communication. The three variables from the System object (*Context Switches/sec_System*, *File Control Operations/sec_System* and *File Control Bytes/sec_System*) reflect the operating system (OS) operations for process scheduling and file system.

The values of each S or Q variable in category 2 increase as the sampling rate and the number of clients increase, but decrease as the buffer size increases as shown in Fig. 3b. Hence, a higher level of service quality (i.e., a higher level of the sampling rate) and more user requests (i.e., a larger number of clients) demand for more CPU, communication bandwidth, and system IO and file resources. However, increasing the size of the communication buffer (B) can effectively reduce the number of datagrams sent by the voice communication service even with the similar amount of network data being sent, and subsequently reduce the workload on the CPU, system IO and file system. Hence, this category of A-SQ relations suggests a useful strategy for adapting service and system configuration by increasing the size of the communication buffer to maintain the workload level on CPU, system IO and file system resources and thus not to stress out these resources even when higher QoS requirements with more user requests (more clients) and better voice quality (higher sampling rate) are present.

Category 3 has two variables reflecting the state of the process and the memory, *Thread Count_Process* and *Page Faults/sec_Memory*. The values of each S variable in this category increase as the number of clients increases, show little change (stable) as the sampling rate increases except for the lowest or highest level (one end) of the sampling rate, and have an inverse-U change as the buffer size increases with higher values in the middle range of the buffer size (that is, the values of each variable increase and then decrease as the buffer size increases) as shown in Fig. 3c. *Thread Count_Process* is determined by two factors: the number of clients requesting voice data and the number of buffers waiting to be sent. As the number of clients increases, more threads need to be created to send voice data to the clients, resulting in the increase of the thread count. As the buffer size increases, the number of buffers filled up by the sampling thread (*NBF*) during a period of time decreases due to the larger buffer size. However, the number of buffers sent out by a sender thread (*NBS*) during a period of time also decreases due to the longer delay for sending the larger size of data. When *NBF* is larger than *NBS*, more sender threads need to be created to send out the voice data. When *NBF* is smaller than *NBS*, only one sender thread is needed for each client. Hence, as the buffer size increases, the difference between *NBF* and *NBS* first becomes larger causing the increase of *Thread Count_Process*, and then becomes smaller causing the decrease of *Thread Count_Process*. For *Page Faults/sec_Memory*, the increase of *Page Faults/sec_Memory* as the number of clients increases indicates that the competition on the system memory resource by the voice communication service and other applications/services in the system becomes more intensive when the number of clients increases. As the buffer size increases, more memory is consumed (as explained earlier for Category 1), which also leads to more intensive competition on the system memory resource. However, the number of memory accesses is reduced as the buffer size increases due to the reduced workload on the CPU and system IO (as explained earlier for Category 2), which leads to a smaller number of page faults. Hence, the value of *Page Faults/sec_Memory* first increases and then decreases as the buffer size increases. Since page faults cause slower memory accesses, it is always desirable to have less page faults. The A-S relation of *Thread Count_Process* and *Page Faults/sec_Memory* indicates that selecting a proper initial value of the buffer size is important for tuning the system workload imposed by the voice communication service.

Category 4 includes the three state variables from the Memory object measuring the same state of the memory in different units: *Available Bytes_Memory*, *Available Kbytes_Memory*, and *Available Mbytes_Memory*. The values of each variable in this category decrease as the sampling rate, the number of clients and the buffer size increase as shown in Figure 3d. The variables in this category measure the

available memory, whereas the variables in Category 1 measure the used memory consumption. Hence, the A-SQ relation in Category 4 conveys similar information to that in Category 1.

Category 5 has 7 state variables concerning the System, Physical Disk, IP, UDP, and Process objects. These variables measure the state and performance of those system and network resources which are not directly linked to the voice communication service. For example, the voice communication service generates mostly data sent out from the server rather than data received by the server as measured by *Datagrams Received Delivered/sec_IP*, *Datagrams Received/sec_IP*, and *Datagrams Received/sec_UDP* in Category 5. Each S variable in this category does not show a consistent change as the sampling rate, the number of clients and the buffer size increase. For example, the change of *Datagrams Received/sec* IP with the sampling rate is different for different numbers of clients as shown in Figure 3e. The variables in this category are likely affected by not only the voice communication service but also system routine activities which together produce the inconsistent change pattern of these variables with the service activity parameters of the voice communication service. Hence, the variables in this category should not be considered as accurate measures of system state and QoS performance that are directly or solely linked to the voice communication service.

In summary, the 21 state and QoS variables in Categories 1-4 are directly related to the voice communication service, and show four major categories of the A-SQ relations.

5.3. The ASQ relation map

Among the 21 state and quality variables from Section 5.2, some variables present similar information. While summarizing the ASQ relations into an ASQ relation map with each node representing a variable and a link representing a relation between two variables, we can keep only one variable among a group of variables that present similar information.

The two variables in Category 1 have the following relationship:

% Committed Bytes In Use_Memory = Committed Bytes_Memory / Memory$_{total}$,

where *Memory*$_{total}$ is the total size of system memory. Only *Committed Bytes_Memory* is kept in the ASQ relation map.

In category 2, *Datagrams/sec_IP*, *Datagrams/sec_UDP*, *Datagrams Sent/sec_IP*, *Datagrams Sent/sec_UDP*, *Fragmented Datagrams/sec_IP* and *Fragments Created/sec_IP* present similar information about the amount of network traffic created and sent out by the voice communication service at UDP and IP layers of the network protocols. Note that there is little incoming traffic to the voice communication server in our experiment. Since the sizes of datagrams depend on the communication buffer size used in our experiment, *Datagrams Sent/sec_IP* instead of *Fragments Created/sec_IP* closely reflects network throughput. Hence, *Fragments Created/sec_IP* is taken as the service quality measure of the network throughput for the voice communication service, and is the only variable among the six variables from the IP and UDP objects that is kept in the ASQ relation map.

Also in category 2, *%Processor Time_Process*, *%User Time_Process* and *%Privileged Time_Process* has the following relation:

%Processor Time_Process = %User Time_Process + %Privileged Time_Process,

Only *%Processor Time_Process* is kept in the ASQ relation map. In category 2, *IO Other Operations/sec_Process* and *IO Other Bytes/sec_Process* of the Process Object presents similar information about the workload of IO devices or how busy the IO devices are. Only *IO Other Operations/sec_Process* is kept in the ASQ relation map.

In category 2, *File Control Operations/sec_System* and *File Control Bytes/sec_System* of the System Object measure the control operations of the file system. Since IO devices are treated as special device

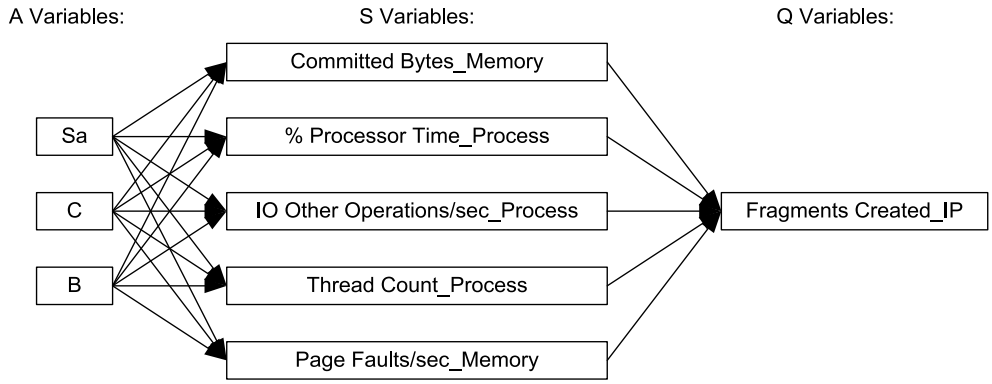


Fig. 4. The ASQ relation map.

files in Windows in order to simplify the system design, *File Control Operations/sec_System* includes both accesses to IO devices (already measured by *IO Other Operations/sec_System*) and regular files (mainly the log files in our experiments). In other words, *File Control Operations/sec_System* and *File Control Bytes/sec_System* are affected not only by the voice communication service activities by also the monitoring activities. Hence, these two variables are excluded in the ASQ relation map.

In category 3, both *Threat Count_Process* and *Page Faults_Memory* are kept since they give the state information for different system resources.

In category 4, *Available Bytes_Memory*, *Available Kbytes_Memory* and *Available Mbytes_Memory* of the Memory object present similar information to that by *Committed Bytes_Memory* of the Memory object which is kept in the ASQ relation map. Hence, the three variables in category 4 are excluded in the ASQ relation map.

In summary, the following six state and Q variables:

- *Committed Bytes_Memory*
- *% Processor Time_Process*
- *IO Other Operations/sec_Process*
- *Thread Count_Process*
- *Page Faults/sec_Memory*, and the following QoS variable:
- *Fragments Created/sec_IP*

are kept in the ASQ relation map. The results in Section 5.2 reveal the relations of the three service activity parameters, Sa, C and B, with these five state variables. The A-S relations are directly represented in the ASQ relation map as shown in Fig. 4. As described in Section 2, we consider that the service activity parameters first act on the system resources and change the state of the system resources which in turn causes the change of process performance and thus the service quality variable. The voice communication service continuously samples voice data from the sound card of the system and transmits the sampled voice data to the clients. These operations require access to IO devices (mainly sound card and network interface) and creation of multiple threads, and consume CPU time and system memory. The states of CPU, memory, IO devices, and operating system have significant impact on the network throughput of the voice communication service, which is measured by *Fragments Created/sec_IP*. Hence, in the ASQ relation map shown in Fig. 4, the relations of the five state variables with the service quality variable are represented.

Table 5
Linear regression models for A-S and S-Q relations

S or Q variable	Regression model	R^2
Committed Bytes_Memory (S_1)	$S_1 = 429097992 + 11746708(Sa) + 15190725(C) + 426122(B)$	0.9939
%Processor Time_Process (S_2)	$S_2 = -2.30198 + 1.02864(Sa) + 0.87906(C) - 0.33787(B)$	0.7569
IO Other Operations/sec_Process (S_3)	$S_3 = -19.7548 + 37.5533(Sa) + 37.4458(C) - 30.9270(B)$	0.7986
Thread Count_Process (S_4)	$S_4 = 14.79020 + 0.84680(Sa) + 4.20747(C) - 0.12113(B)$	0.9595
Page Faults/sec_Memory (S_5)	$S_{15} = -119.014 + 24.065(Sa) + 199.893(C) + 4.758(B)$	0.5160
Fragments Created/sec_IP (Q)	$Q = -7544.65 + 0(S_1) + 174.76(S_2) - 7.86(S_3) - 0.06(S_4)$	0.9419

5.4. ASQ models

For each state variable in the ASQ relation map, we obtain a linear regression model that represents the quantitative A-S relation of the three service activity parameters with the state variable. For the variable of the achieved service quality, we build a linear regression model to represent the quantitative S-Q relation of the five state variables with the variable of the achieved service quality. These regression models are presented in Table 5. In these regression models:

C = 1, 2, 3, 4, 5 for one to five clients, respectively,

Sa = 1, 2, 3, 4, or 5 for 44100, 88200, 132300, 176400, 220500 Hz, respectively,

B = 1, 2, 3, 4, or 5 for 16384, 24576, 32768, 40960, 49152 Kbytes, respectively.

Since the difference between two successive levels of Sa (e.g., 88200–44100 = 132300–88200) and B (24675–16384 = 32768–24576) is the same, S = 1, 2, 3, 4 or 5 and C = 1, 2, 3, 4, or 5 in the regression models is simply a scaled value for the original value of S and B. The R-square value for each regression model gives the goodness-of-fit of the regression model to the data with a larger value indicating a better fit. For *Page Faults/sec_Memory*, the linear regression model does not fit the data well. Hence, we use the MARS technique to build a nonlinear regression model with the R^2 value of 0.8160. The MARS model is not presented in the paper due to its complex form.

6. Conclusions

This paper presents our methodology of data collection, data analysis and data modeling to establish activity-state-quality models which can be used to enable the configuration and adaptation of services and system resources for meeting service quality requiments. We illustrate the methodology using the voice communication service. For the voice communication service, we uncover a number of state and service quality variables that are significantly affected by the three service activity parameters of the sampling rate, the number of clients and the buffer size, including *Committed Bytes_Memory*, *% Processor Time_Process*, *IO Other Operations/sec_Process*, *Thread Count_Process*, *Page Faults/sec_Memory*, *Fragments Created/sec_IP*, and others. We also reveal four major categories of the ASQ relations: 1) a state or quality variable increases its values as the three service parameters of the sampling rate, the number of clients and the buffer size; 2) a state or quality variable increases its values as the sampling rate and the number of clients increases but decreases its values as the buffer size increases; 3) a state or quality variable increases its values as the number of clients increases with little effect of the sampling rate except at one low or high level, and first increases and then decreases its values as the buffer size increases, and 4) a state or quality variable decreases its values as the three service parameters increase. The ASQ relations provide us with insights into meeting higher demands of service quality with a higher sampling rate of voice data and more client requests by properly increasing the size of the buffer holding

communication data before transmission over the network. The ASQ relations and the regression models defining the quantitative ASQ relations can be useful in predicting the change of the achieved service quality when initially configuring and later adapting the service activity parameters, the resource capacity and the service-resource binding to meet the service quality requirements. Although this study focuses on a communication-intensive voice communication service, the experimental and analytical methodology is applicable to investigating and developing dynamics models of service activity, system state and service quality cause-effect dynamics for other computer and network services. For a different service, empirical data for that service needs to be collected and analyze to obtain ASQ relations and models. The ASQ relations and models established in this study involve only three service and system parameters of the sampling rate, the number of clients, and the size of the buffering holding the sampled voice data. In our future research, we may work on developing more comprehensive ASQ models including additional modeling parameters such as the size of the receiving buffer, network delay and its variability, and a larger number of clients.

Acknowledgments

This work is sponsored in part by the National Science Foundation (NSF) under grant number CCF-0725340 and in part by the Air Force Research Laboratory (AFRL) under award number FA8750-08-2-0155. The U.S. government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of, NSF, AFRL, or the U.S. Government. We would like to thank Professor Hessam Sarjoughian for his constructive comments throughput this research. We would like to thank Suseon Yang for developing the regression models presented in this study.

References

[1] F.M. Abdel-qader, *Examples to create your conferencing system in .NET, C# VOIP & video conferencing systems using H.323 and TAPI 3*. Retrieved 03/16, 2009, from http://www.codeproject.com/KB/IP/Video_Voice_Conferencing.aspx?fid=373359&df=90&mpp=25&noise=3&sort=Position&view=Quick&select=2645686, 2007.

[2] N. Artaiam and T. Senivongse, Enhancing Service-Side QoS Monitoring for Web Services, 9th *ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing* (2008), 765–770.

[3] J. Cao, J. Huang G. Wang and J. Gu, QoS and Preference based Web Service Evaluation Approach, 8th *International Conference on Grid and Cooperative Computing* (2009), 420–426.

[4] Y. Chen, T. Farley and N. Ye, QoS requirements of network applications on the internet, *Inf Knowl Syst Manag* **4**(1) (2004), 55–76.

[5] X. Deng and C.A. Xing, QoS-oriented Optimization Model for Web Service Group. 8th *IEEE/ACIS Conference on Computer and Information Science* (2009), 903–909.

[6] J.H. Friedman, Multivariate adaptive regression splines, *The Annals of Statistics* **19**(1) (1991), 1–67.

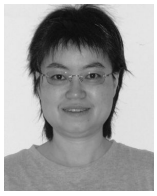
[7] M. Grah and P. Radcliffe, Dynamic QoS and Network Control for Commercial VoIP Systems in Future Heterogeneous Networks. 10th *International Symposium on Multimedia* (2008), 356–383.

[8] C. Jiang, H. Hu, K. Cai, D. Huang and S.S. Yau, An intelligent control architecture for adaptive service-based software systems with workflow patterns, *Computer Software and Applications Conference, Annual International* (2008), 824–829.

[9] H.N. Kapadia, A.B.J. Fortes and C.E. Brodley, Predictive Application-Performance Modeling in a Computational Grid Environment, *Proceedings of the 8th IEEE International Symposium on High Performance Distributed Computing* (1999).

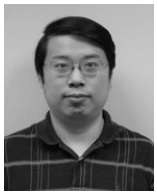
[10] K. Kritikos and D. Plexousakis, Requirements for QoS-based Web Service Description and Discovery. 31st *Annual International Computer Software and Applications Conference* (*COMPSAC*) (2007), 467–472.

[11] L. Kulnarattana and S. Rongviriyapanish, A Client Perceived QoS Model for Web Services Selection. 6th *International Conference on Electrical Engineers/Electronics, Computer, Telecommunications and Information Technology* (2009), 731–734.
[12] T. Lv, Research on Workflow QoS, *International Joint Conference on Artificial Intelligence* (2009), 219–221.
[13] H.B. Mann and D.R. Whitney, On a test of whether one of two random variables is stochastically larger than the other, *Annals of Mathematical Statistics* **18**(1) (1947), 50–60.
[14] Microsoft, *Window server 2003 performance counters reference*. Retrieved 03/01, 2009, from http://technet2.microsoft.com.ezproxy1.lib.asu.edu/windowsserver/en/library/3fb01419-b1ab-4f52-a9f8-09d5ebeb9ef21033.mspx, 2003.
[15] X. Peng, T. Telkamp, V. Fineberg, C. Cheng and L.M. Ni, A practical approach for providing QoS in the internet backbone, *IEEE Communications Magazine* **40**(12) (2002), 56–62.
[16] B. Ravindran and T. Hegazy, A Predictive Algorithm for Adaptive Resource Management of Periodic Tasks in Asynchronous Real-Time Distributed Systems in *International Parallel and Distributed Processing Symposium*, 2001.
[17] M. Reisslein, K.W. Ross and S. Rajagopal, A framework for guaranteeing statistical QoS, *Networking, IEEE/ACM Transactions on* **10**(1) (2002), 27–42.
[18] L. Sha, G. Shaozhong, C. Xin and L. Mingjing, A QoS based Web Service Selection model, *2009 International Forum on Information Technology and Applications* (2009), 353–356.
[19] P. Shivam, S. Babu and J.S. Chase, Learning Application Models for Utility Resource Planning, Autonomic Computing, *ICAC '06. IEEE International Conference on* (2006), 255–264.
[20] L. Sun and E.C. Ifeachor, Voice quality prediction models and their application in VoIP networks, *IEEE Transactions on Multimedia* **8**(4) (2006), 809–820.
[21] V.X. Tran, H. Tsuji and R. Masuda, A new QoS ontology and its QoS-based ranking algorithm for Web Services, *Simulation Modelling Practice and Theory* **17** (2009), 1378–1379.
[22] S. Vazhkudai and J.M. Schopf, Using Disk Throughput Data in Predictions of End-to-End Grid Data Transfers, *Grid Computing, LNCS* (2002), 291–304.
[23] X. Wang, T. Vitvar, M. Kerrigan and I. Toma, A QoS-aware Selection Model for Semantic Web Services. 4th *International Conference on Service Oriented Computing* (2006), 390–401.
[24] B. Wu, C. Chi and S. Xu, Service Selection Model based on QoS Reference Vector, *2007 IEEE Congress on Service* (2007), 270–277.
[25] Z. Yang, N. Ye and Y.C. Lai, QoS model of a router with feedback control, *Quality and Reliability Engineering Int'l* **22**(4) (2008), 429–444.
[26] S.S. Yau, D. Huang, L. Zhu and K. Cai, A software cybernetics approach to deploying and scheduling, *FTDCS '07: Proceedings of the 11th IEEE International Workshop on Future Trends of Distributed Computing Systems* (2007), 149–156.
[27] N. Ye, QoS-centric stateful resource management in information systems, *Information Systems Frontiers* **4**(2) (2002), 149–160.
[28] N. Ye, *Secure computer and network systems: Modeling, analysis and desig*n Wiley Publishing: London, UK, 2008.
[29] B. Zhang, Y. Shi and Y. Chen, A Policy-based Adaptation Method for Service Composition. 1st *International Symposium on Pervasive Composition* (2006), 619–624.
[30] G. Zhang, H. Zhang and Z. Wang, A QoS-based web services selection method for dynamic web service composition. *First International Workshop on Education Technology and Computer Science* (2009), 832–835.
[31] J. Zhou, K. Cooper, I. Yen and R. Paul, Rule-base technique for component adaptation to support QoS-based reconfiguration, *Object-Oriented Real-Time Distributed Computing, IEEE International Symposium* (2005), 426–433.



Nong Ye is a Professor of Industrial Engineering at the School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe. She holds a Ph.D. degree in industrial engineering from Purdue University, a M.S. degree from the Chinese Academy of Sciences, and a B.S. degree from Peking University, both in computer science. Her research interests are in information assurance, quality of service, and data mining.

Stephen S. Yau is a life fellow of the IEEE and a fellow of the American Association for the Advancement of Science. He received the PhD degree in electrical engirieering from the University of Illinois, Urbana. He is the director of the Information Assurance Center and a Professor of Computer Science and Engineering at Arizona State University, Tempe. He was previously with the University of Florida, Gainesville, and Northwestern University, Evanston, Illinois. He served as the president of the IEEE Computer Society and the editor-in-chief of *Computer*. His current research is in distributed and service-oriented computing, adaptive middleware, software engineering, trustworthy computing, and data privacy.



Dazhi Huang is a Ph.D. student in computer science at Arizona State University, Tempe. He received the BS in computer science from Tsinghua University, China. His research interests include middleware, mobile and ubiquitous computing, and workflow systems in service-oriented computing environments.

Mustafa Gökçe Baydogan received the MS degree in industrial engineering from Middle East Technical University, Turkey, in 2008. He is a PhD student in industrial engineering, Arizona State University, Tempe. His research interests are data mining, heuristic search and simulation.

Billibaldo Martinez Aranda received the BS degree in Industrial and Systems Engineering from University of Sonora, Mexico, in 2004 and the MS degree in Industrial Engineering from Technological Institute of Hermosillo, Mexico, in 2006. He is currently a PhD student in Industrial Engineering at Arizona State University, Tempe. His research interests include data mining and statistics with applications in fields of information systems, bioengineering and manufacturing.

Auttawut Roontiva received the BS degree from Chulalongkorn University in 2000 and the MS degree from Arizona State University, Tempe in 2004, both in mechanical engineering. He is a Ph.D. student in industrial engineering at ASU. His research interests include information system security and assurance, adaptive service based software systems, and data mining.

Patrick Hurley is a Computer Engineer with the Air Force Research Laboratory's Information Directorate, Rome, NY, USA. He holds a Bachelor of Science in Computer Science from the State University of New York at Oswego and a Master's degree in Computer Science from the State University of New York Institute of Technology (SUNYIT). His research interests are in quality of service, cyber defense, survivability and recovery.