NIPPON STEEL TECHNICAL REPORT N0. 76 MARCH 1998

UDC 681 . 3

Performance Evaluation for Distributed Systems

Takashi OSHIRO*[1] Masaaki NAKAMURA*[1]
Yasuhiro HATAKEYAMA *[1]

Abstract:

There is an increasing need for constructing distributed systems that can freely cope with changing business conditions. However, as many cases show, it is evident that the construction of the system is not only difficult, but there are many special performance problems such as increasing response times at terminals. Frequently, the performance problems are revealed only at the last stage of system development, and consequently the cost to solve such problems is increased, or redoing the development itself may be involved. The technology of performance evaluation has the objective of avoiding in advance such performance problems as those extending through the life cycle of the system and of solving the problems that occur. Performance evaluation technology is very important to SI vendors who contract for system integrating work, since it plays the role of risk management for developing the system. This paper introduces how the Systems Research and Development Center of Nippon Steel buckles down to the performance problems of distributed systems.

1. Introduction

Distributed systems, represented by client/server systems, offer the advantages that first, they can be widely selected to use advanced technology at lower cost, and second, they are easy to expand for optimum system architecture. The system structure, however, is not always easy, as can be seen in many cases. A study of these failures shows that many performance problems are reported where the response time at terminals is slow and batch jobs do not complete within criterion times. Frequently, the performance problems are only revealed at the last stage of system development such as during integration testing, and they are, if found, too difficult to deal with at that stage. Consequently, the cost of solving the problems is greater, or at the worst, redoing of the development itself may be involved.

The reason why the performance problems of distributed systems are difficult to deal with is explained by the following. First, standard hardware, OS, database, and network, etc. can be selected and constructed due to the trend of open and multi-vender applications. But this easy selection causes numerous combinations for which it is very difficult to evaluate performance after combination. And second, a quantitative approach to performance has seldom if ever been applied, and there are many estimates of software development, which brings an attitude of "leaving things to chance" based on hunches and experience. Unlike a single-vendor development system controlled by the collecting of detailed information, it is difficult to collect necessary information and the conventional main frames are likely to be useless.

In these situations, many SI vendors are striving to establish performance evaluation technology for the distributed system as a software developing method by their own corporation to solve the problems.

In this paper, the authors introduce how Systems Research and Development Center in Electronics & Information Systems Division of Nippon Steel Corp. (hereinafter referred to as the Center) addresses performance problems of distributed systems.

2. Performance Evaluation of Distributed System

Performance evaluation is intended to avoid beforehand performance problems during the period from development to operation of the system and to specify and solve problems that occur.

*[1] Electronics & Information Systems Div.

For SI vendors contracting for the system architecture, it is risk management for development, and it is important for proposing and constructing a system that will be suitable for customer's needs (right-sizing). Introduction of SLA (service level agreement)[*1] defining system performance as one of the contract requirements is increasingly raising the significance of performance evaluation.

For discussion of performance, a scale for measuring performance (performance metrics) is important. For this purpose, responsiveness, utilization, and throughput are usually used. Responsiveness means response time from completion of input to return of response. This is an important performance metric for users and directly affects the user's business efficiency. Utilization is an operating ratio for system resources such as CPU, disk, and network. This is a very important scale for the system operation management divisions from the cost management viewpoint. Throughput is work volume which can be processed per unit time. For example, job throughput (the number of job processes per unit time) is a typical unit of volume.

These indices are related to each other, and it is the purpose of performance evaluation to design performance itself so as to obtain a combination of optimum values in consideration of user's demand, costs, and other requirements. In response to development phases, contents required for performance evaluation are changed. At the initial stage of development, the optimum system architecture should be studied. At the final testing stage, detection of performance problems and tuning or performance validation are important. Performance evaluation with a quantitative method depending on development phases is necessary.

2.1 Outline of performance evaluation

Performance evaluation is roughly classified into three steps: (1) preparation of performance evaluation model, (2) execution of performance evaluation method, and (3) analysis of statistical values.

A performance evaluation model is a modeled target system that takes into consideration the purpose and necessary costs for evaluation. It consists of a hardware model including a system constructing computer, disks, and networks; a software model including application programs, middleware structure, and database structure; and a load (workload) applied to the target system. In addition, values to be changed (processing capacity of a server machine and network transmitting capacity, etc.) depending on measuring location of performance metrics and evaluation purpose should be determined at the same time. Performance characteristics for the target system are analyzed by changing this value. Upon system evaluation, a "what-if" is very important; it is necessary for performing multilateral evaluation from various conditions.

A workload is more likely to be neglected than a hardware model or a software model but it has much influence on evaluation results. The workload is a load applied to the system and should be determined not only for the load size from multiple work loads but also for its frequency of occurrence. Frequently, the desired performance satisfied during the unit testing is not obtained during the multi-workload testing caused by queue due to potential competition between workloads. If the workload is insufficiently modeled, the result may be meaningless.

Evaluation methods are described later. Resultant data obtained by applying each evaluation method are important not only for an average but also for a variance, maximum value and minimum value. In particular, special attention should be given to results when the variance is large. For example, even if average CPU utilization is small, in the case of large variance, a high load process may exist and extremely deteriorated performance can occur at a certain point. From the viewpoint of performance assurance, so called 90-percentile value becomes an important index.

2.2 Performance evaluation method

A performance evaluation method is usually discussed in three classified categories: (1) performance prediction, (2) performance measurement, and (3) performance monitoring.

In performance prediction, performance of the target system is predicted by an analytical method or a simulation method, which is characterized by requiring no actual system (execution environment and program) to be evaluated.

The analytical method typified by a queuing theory utilizes a mathematical method to determine mean system waiting time using mainly mean arrival rate and mean service time of the system. This method can analyze data quickly but can only obtain an averaged value. Another defect of this method is that the analysis is very difficult for complicated systems.

The simulation method makes a model of the target system by discrete event simulation (simulation language) to determine performance metrics. It can determine not only an average but also its distribution and can change the load dynamically for flexible evaluation. However, modeling requires sophistication, and construction cost cannot be neglected. Model construction suitable for the required accuracy is important.

In performance measurement, performance metrics such as response time and throughput are measured by imposing the specified workload on the entire or partial system. It can be divided into single workload testing and multi-workload testing as ways of applying the workload. For performance validation, both tests are often employed where results of the single workload testing are optimized first individually then verified for the entire system by multi-workload testing. For generation of multi-workload, a terminal emulation is often employed to produce multiple users by one machine.

For performance comparison, standard benchmark tests such as SPEC[*2] benchmark test and TPC (transaction processing performance council) benchmark test create a series of tests suitable for system purposes for evaluating performance by actual measurement. As most of the vendors publish benchmark results, those results are convenient when benchmark results similar to the target system have been reported. It is also often used as a performance evaluation metric for a single computer.

On the other hand, the corporation-oriented information system proposed by SI vendors is mainly based on database, and the system performance is greatly affected by the database. Results of standard benchmark tests for the database are published. They are useful as a guide for rough estimation; however, it is difficult to decide whether the database performance satisfies the system specification by those results alone. Essential is an application specific benchmark (ASB)[*3] reflecting characteristics of database structure, data distribution and workload, which differ in individual application programs.

Performance measurement, unlike performance prediction,

[*1] SLA (service level agreement): System service quality such as response time to assure for service using method or its requirement.

[*2] SPEC is a trade mark of The Standard Performance Evaluation Corporation.

requires an environment operating with a certain scale and test data, even if it is a prototype. In particular, ASB needs high grade skills and costs more (in work load and work time). This requires ingenious efforts to easily construct a testing environment.

In performance monitoring, operating conditions and performance of the system after the startup are monitored to evaluate whether predicted performance is achieved. It is important not only to monitor performance metrics supplied by the system but also to compare system workloads after the startup with those defined by SLA. In particular, when the workload is larger than the defined value or tends to increase, it becomes basic information for judging the scale and timing of the system extension. Generally, an operating system has a tendency to increase at all times; in particular, the data volume of the database often directly affects performance. Besides performance, special attention should be paid to changes in accumulated data, which is closely involved in capacity management.

3. Performance Prediction by Simulation

In this chapter, performance prediction by simulation is described, with its methods and examples.

The first case where simulation technique is applied is where there is no specific target system. For example, simulation is an effective method for the risk management of performance at the initial development stage to predict specifications and the required number of server machines based on the estimated workload, prediction of utilization and response time of hardware, and workload forecasting for the existing system when expanded.

The second case is where measurement needs exceed costs and a test environment is difficult to construct, or where there are plenty of conditions for comparative investigations. The simulation can evaluate an entire large-scale and complicated system and is available for repeated evaluation by changing parameters of the created model when it determines the system structure for its optimization.

3.1 Outline of simulation

Simulation can be divided into three stages: (1) model preparation, (2) execution and results analysis, and (3) model validation.

Components of the system hardware and software, such as CPU, disk, network, OS and database, are prepared in advance as parts models, and these parts are combined into a system model. Owing to the reuse of parts models, even when the target model is altered, the system model can be constructed efficiently. Thus, for diverse alteration of conditions such as changes in the specification of the hardware or the arrangement of the distributed software, the system model construction can flexibly accommodate them. Since the accuracy of the system model depends on the accuracy of performance metrics used for the simulation, it is important to select parts models with an appropriate elaboration.

In order to propose a system, estimated responsiveness and utilization of the entire system are required. Because there is little information available at this point, an entire modeled system is prepared irrespective of details for qualitative prediction of the utilization situation of server machines or the tendency of performance for the number of users.

[*3] Application specific benchmark (ASB): Benchmark performed based on the testing requirements reflecting characteristics of developing system (application). Unlike standard benchmark test for which each organization decided how to conduct benchmark, ASB's result has no general versatility but is useful to grasp performance characteristics of developing system in detail.

At the design stage, with many selections for realizing the system, trade-offs between these realizing methods should be clarified. A quantitative basis for determining the design specification of the system is obtained by repeating simulations with altered model parameters. In addition, clarifying characteristics of the selected realizing method is effective for the early finding and solving of problems which might occur in the later developing steps.

At the later stage of development and the operating stage, performance evaluation based on measurement is also available. However, this takes too much time for preparation and execution, so that the simulation is applied as prior evaluation for the test specification of measurement. Linkage of measurement and simulation is important for the preparation of a measurement test specification in addition to the model validation.

3.2 Simulation example for selecting server machine

An example of simulation for selecting a server machine at proposal is described below. The server machine is usually selected by a spreadsheet performance model in which the load of a supposed process and its frequency are integrated and the results are multiplied by a safety factor. However, this method generates queues due to competition of system resources such as CPU and disk, so that it cannot predict a server load exactly.

The simulation gives a result, shown in **Fig. 1**, from the same information as used for the spreadsheet performance model. In this case, CPU utilization of the server machine was predicted when its throughput expressed at the value of tpmC[*4] was changed. Utilization of CPU is generally reported to be limited to 60 to 70%, so that level of significance was obtained from an average and variance of utilization. The result shows that the required specification is expected to be more than 4,200 (tpmC) in this case.

Furthermore, the simulation is effective for simulating many what-if cases including prediction with background workload of the system, prediction for increased users and prediction of the effect for increased number of server machines.

3.3 Simulation example for workload forecasting of the server machine

Described below is an example of the simulation to evaluate validity of the basic system design at the stage where the server machine is selected and the application specification is clarified to some extent. With processing time to be estimated from the application specification, CPU utilization of the selected server machine is predicted for judging the design validity. In order to

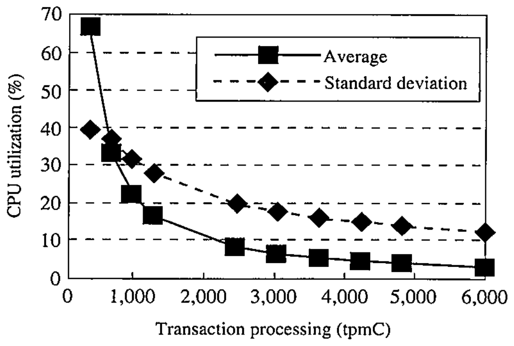


Fig. 1 CPU utilization to processing capacity of server machine

[*4] Value of tpmC: The number of processes of specified transaction per minute at measurement satisfying a transaction ratio of TPC-C and response time requirement.

determine CPU utilization of the server machine, an entire network structure was not modeled; only the server machine and the network segment directly connected to the server machine were modeled.

Fig. 2 shows CPU utilization obtained by the simulation. A solid line represents the ratio of frequency to CPU utilization (a ratio of total time processed by the utilization). The broken line indicates the accumulated ratio of the number of processes treated by CPU utilization. The average value of CPU utilization is 21% with satisfactory variance, which suggests that the load on the selected server machine is considered within the safety region. **Fig. 2** also shows that 90% of the process is treated at a CPU utilization rate of about 40% or less. This suggests that there is no need to review the application specification.

In this example, only CPU utilization is focused on and analyzed for the overall result. Since utilization of disk and network for each process can also be determined in this simulation, it is possible to specify the place and to clarify the cause of performance problems.

3.4 Simulation example for determining the structure of middleware

This is an example of simulation for determining resource assignment to each process when processes differ in character; for example, on-line batch process requiring comparably long process time and on-line transaction[*5] process requiring short process time are operated by the same system. Resource assignment is managed by a database access middleware, but on earlier occasions, it was difficult to determine an appropriate configuration for the middleware because of its large degree of freedom.

In order to set an optimum structure for the middleware, simulation was repeated by changing parameters to predict the response time. **Fig. 3** shows predicted values and ratios of the required

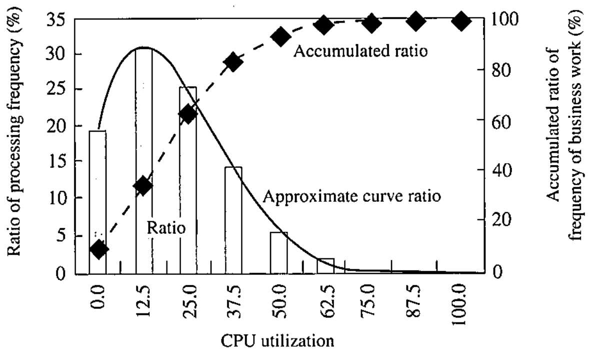


Fig. 2 Ratio of processing frequency to CPU utilization

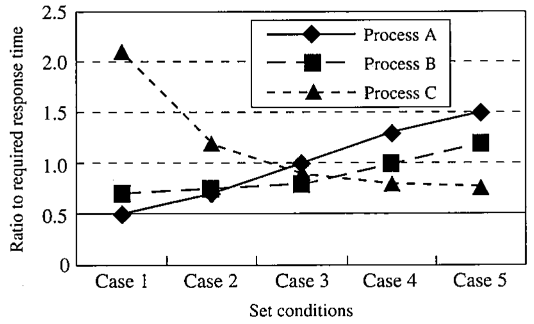


Fig. 3 Change in responsiveness to set conditions

response time for five parameters set for each process. Only Case 3 satisfied performance requirements for all processes.

In this example, only the peak load was employed as the workload, but performance evaluations at varied workloads is important. Changes in responsiveness to the variation of workload can be predicted by simulation.

4. Performance Validation of Distributed System by Actual Measurement

In this chapter, the authors introduce an implementation of performance validation on the basis of measurement utilizing BenchWorks™[*6] as an example of implementation of Application Specific Benchmark (ASB).

BenchWorks™ is a performance evaluation test support tool for distributed systems including not only a database server but also a network of a system constructed using middleware such as database access tools. BenchWorks™ is an output of R&D in this Center and is applied for performance validation in the system integration project within and outside Nippon Steel Corp.

4.1 Outline of BenchWorks™

BenchWorks™ supports performance evaluation tests for an entire development cycle from initial stage to completion stage as a test environment structure support tool of ASB. Major functions includes: (1) database modeling, (2) workload modeling, and (3) automatic implementation and resultant collection/analysis of workload. BenchWorks™ can construct database model and workload model while enhancing accuracy step by step.

A problem which might occur upon ASB operation is that a test environment may not be well arranged for measurement. Accuracy tends to fall with ambiguous evaluation unless measurement is conducted with the same environment as the actual system or with an equivalent hardware (HW) environment. Besides HW environment, database configuration and application architecture also present similar conditions. It is necessary to find closely how the test environment is approximated to actual conditions for meaningful results, and to construct a test environment that will be suitable for validation purposes.

At the initial stage of development, another problem arises: information of the system and process required for test is not well filed and is not well prepared for the test specification. At this stage, where neither HW environment nor application are established, HW should be prepared as a test environment on which the database and workload should be constructed. BenchWorks™ supplies a function for preparing of the test database and preparing and running the workload.

At the later stage of development, HW environment is prepared and the application is established to some extent. The test should be done involving this environment. For performance evaluation of existing systems, the test environment and application satisfactorily prepared should be utilized for the test. BenchWorks™ can provide a test meeting such requirements.

BenchWorks™ can automatically operate the workload defined for the prepared database to calculate system throughput as a measured result, as well as graphical display and various statistical data of response time.

4.2 Preparation of database model

During preparation of the database modeling, specification of database objects such as a table space, table, cluster, index, and

[*5] Transaction: A series of processes not divided anymore.

[*6] BenchWorks™: A trade mark of Nippon Steel Corp.

constraint, etc. is arranged to define the database using GUI (graphical user interface) of BenchWorks™. BenchWorks™ can define not only the logical structure of the database but also such physical information as file arrangement. BenchWorks™ automatically produces DDL (data definition language)[*7] from the database definition information and transmits it to DBMS (database management system) for performing data construction.

At the same time, column attributes are arranged and defined using BenchWorks™. For each column, column attributes of data type, data distribution, data range and step (for numerical data), string data length and variation (for character data), and frequency of NULL data are defined through a dialogue window for definition. BenchWorks™ can easily deal with these processes using a dialogue window (see **Fig. 4**).

Definition of column attributes greatly affects results of the performance evaluation test. In particular, data distribution of the column largely affects performance, so that it should be defined in consideration of hit ratios or hit records of each transaction specified by the workload model. For definition of column attributes, it is important to reflect the actual system as exactly as possible.

BenchWorks™ automatically generates table data based on the definition information of the column attributes and transmits the DDL statement mentioned above to insert data into the defined database and automatically produces the database for testing. BenchWorks™ automatically carries out intricate test data generation for easy construction of performance evaluation test environment.

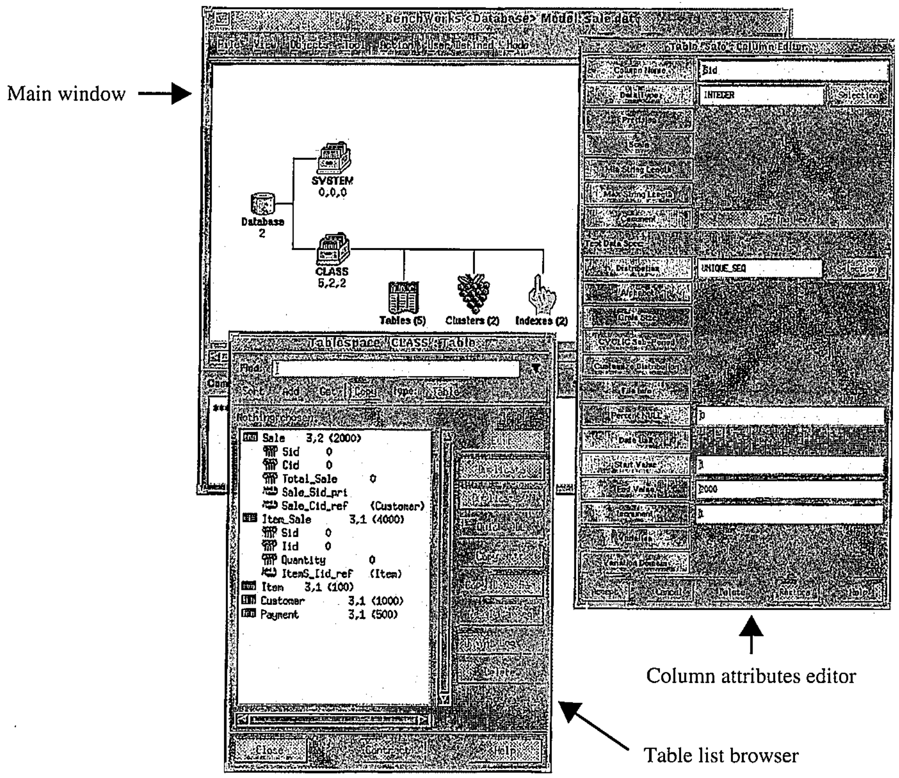
4.3 Preparation of workload model

Upon preparing the workload model, it arranges information of processing contents of the transaction and trasaction generation pattern (transaction mixed, occurence, transaction flow, the number of concurrent users, etc.) and the executing environment of performance evaluation test, and the workload is defined based on them.

For processing contents of the transaction, it is necessary to have information regarding what to access in which table of the created database model, how to process and what constraints (business tool) are to be followed at processing. Other information is also useful including inquiry conditions and hit ratio as well as the number of hit records under its inquiry conditions.

BenchWorks™ supplies two methods for creating the workload model. One is modeling prepared by an exclusive simplified language that extends SQL (structured query language)[*8] for defining the workload of BenchWorks™. Another method is modeling by C language using BenchWorks™ C library for defining the workload. Modeling prepared by the simplified language is effective for creating and testing the prototype system at the initial stage of development to easily describe the workload model. C language using BenchWorks™ C library can describe a model in detail to create the workload model using middleware of the database access and existing applications. It is effective for performance evaluation testing at the latter stage of development or for an existing system.

The execution environment of a performance evaluation test defines whether to start and operate a workstation (WS) executing the client process and some of the processes operating under each



```
Sale      3,2 (2000)
  Sid       0
  Cid       0
  Total_Sale        0
  Sale_Sid_pri
  Sale_Cid_ref      (Customer)
Item_Sale       3,1 (4000)
  Sid       0
  Iid       0
  Quantity          0
  ItemS_Iid_ref     (Item)
Item      3,1 (100)
Customer        3,1 (1000)
Payment         3,1 (500)
```

Fig. 4 Example of database modeling screen

[*7] DDL (data definition language): Definition language of relational database schemer.

[*8] SQL: Operating language of relational database

WS. **Fig. 5** shows the case where the database server and two WS are connected to the network to execute "m" and "n" sets of processes at each WS. In this case, (m + n) number of clients are implemented using a terminal emulation function of BenchWorks™. This function constructs the test environment with limited resources at the initial stage of development without sufficient HW to implement the test. This is one of the great advantages of BenchWorks™.

4.4 Execution and analysis of evaluation test results

BenchWorks™ automatically executes the test based on the definition of the workload. It activates the specified number of client processes for processing according to definition using the terminal emulation function. A required time for all transactions activated from the client process is recorded as a test result. From the result, BenchWorks™ calculates statistical values such as the number of processed transactions, system throughput and response time to create changes of the throughput with elapsed time and histograms of the response time (see **Fig. 6**. These graphs are drawn by a graphic software Xmgr[*9]).

System performance was repeatedly evaluated by testings in which parameters were changed for the database model and workload model. It is important not only to measure the response time of the system but also to check breakpoint of the system and to verify performance characteristics of the system.

BenchWorks™ supplies a function for easily changing the database model such as table allocation and table size, and the workload model such as the number of access users and generation pattern of transactions. This reduces the workload during test environment construction to shorten the required time for performance evaluation testing.

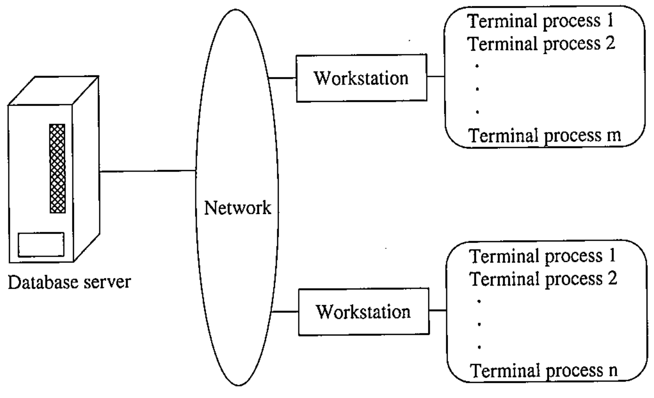


Fig. 5 Example of execution environment of performance evaluation test

5. Conclusions

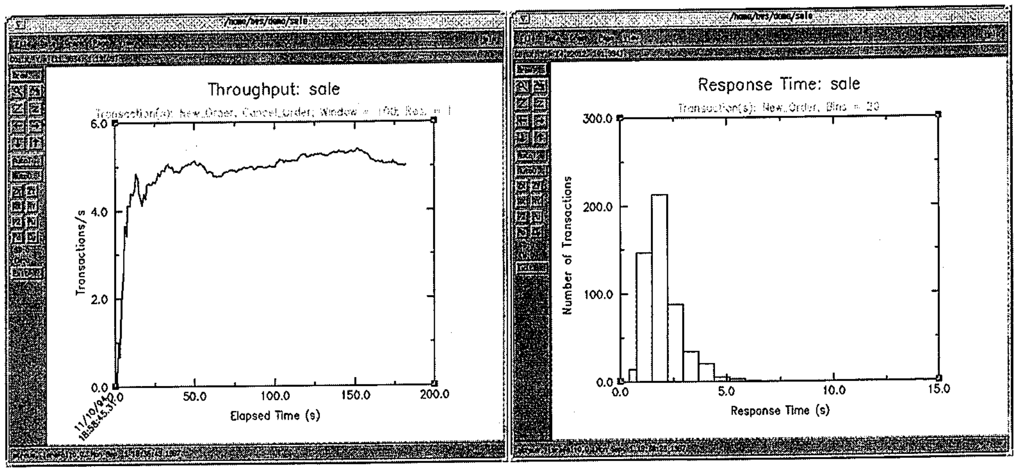
We introduced the stance of the Center engaged in performance evaluation of distributed systems, the necessity of which has rapidly increased recently.

Performance evaluation technology itself may not necessarily be new technology, but it requires, for actual evaluation, a wide range of knowledge of information on the target system and technical ability and negotiation skills with the clients as well as each elemental technology.

The Center has researched and developed the three element technologies of prediction, measurement, and monitoring as core technologies on what performance evaluation should be during the entire phase of development. This activity is linked to actual projects including those for the steelworks inside the corporation. In July 1997, the Center established the "Benchmark & Consultation Center" as a new key operating station associated with Systems Research and Development Center in Electronics & Information Systems Division of Nippon Steel Corp.

Reference

1) Jain, R.: The Art of Computer Systems Performance Analysis. 1st ed. John Wiley & Sons. Inc. 1991, 685p.



(a) Charge of throughput with elapsed times

(b) Histogram of response time

Fig. 6 Graphic display of test results

*9 Xmgr: Copyright 1991-1996 by Paul J. Turner. Portland. OR All Rights Reserved.