TANDEM

Performance Management and SURVEYOR

Helaine Horwitz
Stephen Shugh
Scott Sitler

Technical Report 89.2
December 1988
Part Number: 15308

TANDEM

Performance Management and SURVEYOR

Helaine Horwitz
Stephen Shugh
Scott Sitler

Technical Report 89.2
Part Number 15308
December 1988

Performance Management and SURVEYOR

Copyright Notice

Document History

Part Number:	15308
Product Version:	SURVEYOR C10
Operating System Version:	C10
Date:	September 1988

Abstract

This document describes how SURVEYOR fits into performance management functions. While this document does not cover all aspects of performance management, some important aspects of performance management are discussed.

This document is intended for Tandem systems analysts and Customer systems analysts familiar with general performance management concepts. Readers should be familiar with SURVEYOR, having used the program, read the User's Guide, and/or attended the SURVEYOR Seminar (Tandem education class number 14060).

For additional information on SURVEYOR, please refer to the *SURVEYOR User's Guide* (Tandem part number 84153) and the *SURVEYOR Reference Manual* (Tandem part number 85154).

Authors: Helaine Horwitz, Scott Sitler, & Stephen Shugh; Customer Support Organization, Randy Baker, Vice President.

Performance Management and SURVEYOR

TABLE OF CONTENTS

Section 1: Performance Management Overview

Performance Management is the process of assuring adequate computing service to individuals using the existing computer resources. A user expects a system to work (execute applications) in a timely and reliable fashion. The available resources must be managed to meet users' needs while maintaining a high level of system performance.

Performance management activities include:

- studying and understanding the computing environment
- defining system requirements and service objectives
- monitoring and measuring system performance and workloads
- maintaining historical performance data
- adjusting and tuning the systems
- predicting future performance requirements
- recommending hardware and software changes to improve performance

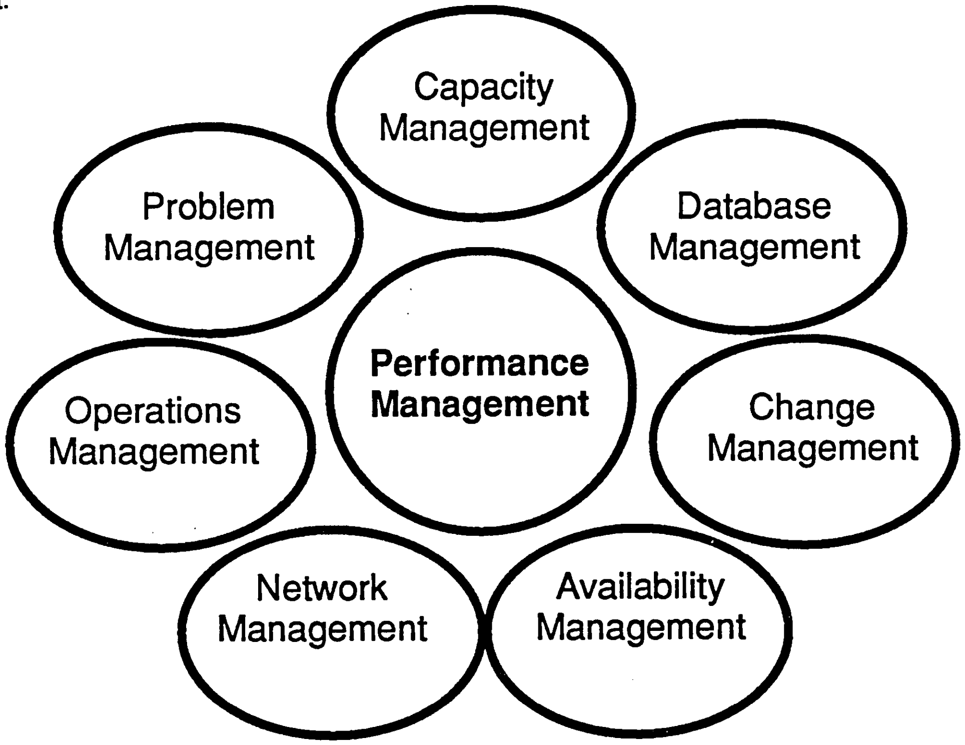
In most cases, performance management is part of a comprehensive computer installation/facilities management organization. Figure 1 shows the typical functions included.

As shown in Figure 1, each function is inter-related. For example, capacity management is heavily dependent on performance management for performance information. If system performance begins to degrade, capacity management should be alerted so that additional resources can be ordered. The Performance Management organization must work with Network Management to ensure network delays do not cause a deterioration in service to users. Performance management depends on operations management to provide controlled operational access to the system(s). Figure 1 provides a perspective on where performance management might fit into a computer installation/facilities management organization. In this perspective, capacity planning is not part of performance management. In other situations, capacity planning is addressed within the performance management organization and is not a separate organization. Other definitions place performance analysis and tuning within the capacity management organization. Even though there are different organizational perspectives, the need for performance management and capacity management is well defined.

The rest of this paper focuses on some of the important aspects of performance management and how SURVEYOR, a performance database manager, can assist in performance management.

In the following sections, "performance management" refers to the group of people performing the performance management functions. "Users" refers to any person or group of people using resources on the system.

Figure 1.
Installation/Facilities Management.



CHANGE MANAGEMENT
- Evaluation
- Planning
- Testing
- Tracking

PERFORMANCE MANAGEMENT
- System Measurement
- Analysis
- Prediction
- Tuning

DATABASE MANAGEMENT
- Design
- Administrative Control
- Operations & Performance
- Application Support

PROBLEM MANAGEMENT
- Detection & Collection
- Analysis & Resolution
- Tracking & Control

CAPACITY MANAGEMENT
- Workload Definition
- Forecasting
- Analysis & Reporting
- Planning

NETWORK MANAGEMENT
- Design
- Testing & Installation
- Operation
- Training

OPERATIONS MANAGEMENT
- Planning & Scheduling
- Operation & Control
- Analysis & Reporting

AVAILABILITY MANAGEMENT
- Evaluation
- Software Design
- Hardware Configuration
- Tracking & Control

For more details on the installation/facilities management perspective shown in Figure 1, please see *Capacity Planning Implementation*, an IBM Technical Bulletin (# GG22-9015-00, January 1979).

This page was intentionally left blank

Section 2:
The Role Of Surveyor In Performance Management

Surveyor plays a key role in four important performance management functions:

System Measurement
- Collecting and storing workload and system resources information

Performance Monitoring
- Monitoring service level objectives and other performance indicators

Performance Analysis
- Reducing and summarizing performance information
- Providing input to performance models

Performance Prediction
- Trending and forecasting via a historical performance database
- Providing input to capacity planning models

Defining Performance: System Measurement

System performance refers to how the resources of a system respond as users run applications. The resources involved are processors, disks, communications lines, memory, etc. Before performance can be managed, the definition of performance must be established. The first step in defining performance is to understand current resource consumption and the performance of applications using system resources. By measuring and monitoring system resources, insights into resource consumption are acquired.

To help understand resource consumption and the performance characteristics of a system, some basic issues must be addressed:

- What are the system's average and peak processor utilizations ?
- What are average and peak disk utilization ?
- Is the system communication bandwidth fully utilized ?
- Is the system swapping too much due to insufficient memory ?
- Are long queues building up at any one resource ?
- Are any resources under-utilized ?
- What is the service time of transaction T ?
- What impact (on system performance) is caused by workload X ?
- Is the throughput of workload Z at an acceptable level ?
- What is the average response time for users of application A ?

Answers to these questions will provide some clues to the performance characteristics of the system under study. This information may indicate which system resources to measure and monitor.

Performance can be defined from two perspectives: the "service" or the "user" perspective. The questions above define performance from the service perspective. Users might have a different definition of performance and adequate service. So, defining exactly what "performance" and "adequate" service means is the next step.

Service Level Objectives

The user community and the performance management organization must work together to define "adequate service." Users know intuitively what they need to get their job accomplished. For example, the user might view his computing needs in terms of orders per day. Performance management must understand this requirement in terms of system resources in order meet this need.

The performance management organization must translate adequate service definitions into service level objectives (SLOs). SLOs represent the meaning of "adequate service" to performance management. They define values or ranges of values for performance indicators considered acceptable for the users. In most cases, SLOs address response time, availability, and/or throughput issues.

In the orders per day example above, an SLO may be a target RECEIVE-RATE for the "order" servers in the order processing application. This may be established by collecting and storing information on the "order" servers. For this example, the sum of the RECEIVE-RATE for all the order server processes provides the orders per day measurement to compare against the defined SLO.

Typically, SLOs are developed from either "rules of thumb" or from calculated expected values. Rules of thumb are general guidelines adopted to gauge performance such as: "run all processors at xx% utilization," or, "never let the queue length get above *n*." Though useful, these general guidelines should be used with caution. They are general to all environments, but not necessarily accurate for one particular environment.

Service level objectives should be based more on calculated expected values than on general rules of thumb. Calculated expected values may initially be derived from these general guidelines. But specific business volume and growth information should be factored in to calculate expected values. The calculated expected values usually develop over time as the users and performance management better understand the environment. Queueing network modeling theory and operational analysis can also help to define the expected values. Queueing network modeling is a methodology for the analysis of computer systems. (For more information on queueing network models, please read *Quantitative System Performance - Computer System Analysis Using Queueing Network Models* by Edward Lazowska, published by Prentice-Hall, 1984.)

An SLO must be tangible to both the user and performance management. In order for an SLO to be effective, it must be an objective, measureable goal. If the SLO isn't being measured or can't be measured, then it can't be managed. Both the users and performance management must agree on the definition and the method used to measure each SLO.

Service Level Agreements

The SLOs form the foundation of the service level agreement between the users and system management. A service level agreement (SLA) documents the service level objectives and provides performance management with a mechanism to measure their performance. With a service level agreement in place, systems management has better control over the computing environment. As new applications are installed or as the user work requirements change, the SLA is redefined and renegotiated.

Once the users and performance management have documented a service level agreement, SURVEYOR can track some of the SLOs. Tracking performance indicators with SURVEYOR is discussed later in this document.

Workload Definition

To meet and measure SLOs, performance management must understand the work imposed on system resources by the various user applications and requests. In short, they must understand the users' work to manage resource consumption and system performance.

Resource consumption is frequently stated in terms of workloads. A workload is a unit of computer work performed on behalf of users. It is usually defined from the users' perspective. A transaction or group of transactions executed via some application is also considered a workload. When an application is run, the work imposed on the system resources is categorized as an "application workload." The concept of a workload allows system resource usage to be apportioned among the applications.

Workload characterization is the process of identifying and quantifying the consumption of system resources into logical groups. Workloads are generally organized around functional groups. For example, the accounting organization of a business may have workloads defined along functional lines. All work associated with one function such as payroll, would be characterized as one workload. Resource consumption for the payroll workload is logically grouped together.

High-Level vs Low-Level Characterization

Workloads can be characterized from a "high level" global perspective or a "low-level" detailed perspective. In some cases, an individual transaction is established as one workload. This is an example of a low-level perspective. In other cases, it is more important to characterize high level workloads. Instead of characterizing a workload as a single transaction, the entire application is defined as a workload. The work done by all transactions executed through the application is grouped into one logical work unit.

Many pieces of information are used to characterize a workload. The choice of high-level or low-level characterization depends on the amount of detail needed about the particular workload or resource.

First, the business function and the expected end result of that function must be defined. Using the accounting example above, accounts receivable can be characterized as one business function. The end result of the accounts receivable function is to make sure all money expected is received and credited to the business. This defines the bounds of what could be part of the accounts receivable workload.

Once the bounds are established, the next level of detail involves understanding the type of work required by the system to achieve the expected end result. A user may submit many different types of small transactions; all of these transaction types could be grouped into one workload. This workload would provide performance information about the entire accounts receivable function as one logical unit. However, for more accurate accounting and increased control of the system resources, smaller individual units of work should be used instead. Each transaction type of the accounts receivable workload should be defined as one workload.

SURVEYOR Workloads

Workloads provide a simple way to account for resource consumption. Instead of tracking multiple performance data fields for many different entities, workloads provide logically grouped information on resource consumption.

In order to accurately measure a workload, performance management needs as much detail about the workload as possible. Some of the key pieces of information used to characterize workloads are shown below.

- How many users submit each type of transaction ?
- How often ? Are there peak periods ?
- What individual transactions make up this business function ?
- What system resources are used by each transactions ? ...processor, disk, communication line, memory, etc?
- What is the path that each transaction takes to do its work ? From terminal, over communications line, to communications controller, to communications line handler, to terminal control program, to requester process, to server process, to disk process, to disk controller, to disk, and then back to the terminal.
- What work does each transaction do at each step of its path ? computations, i/os, etc.
- Knowing the transaction path, how much time is spent in the processor, on the communication line, on the disk, etc.
- What percentage of shared resources (disk processes, terminal control programs, etc.) need to be allocated to this transaction ?
- What priorities are involved as the transaction moves through the system ?

With this information documented, SURVEYOR can be configured to help monitor user workloads and system resource usage.

SURVEYOR provides a grouping function for monitoring and tracking workloads. In cases where the entire entity (disk volume, process, communication line, etc.) is associated with the workload, SURVEYOR will logically group the entire entity with the workload.

SURVEYOR also allows the apportioning of the entities within a workload. Once you define the workloads and understand how individual resources are shared, SURVEYOR can track those workloads with shared resources.

The Tandem TCP (a multi-threaded terminal control process) is one process that is usually shared among multiple transaction types. Therefore, if a workload exists for each transaction using the TCP, then each workload must be defined such that a portion of the TCP is allocated to each workload.

In the above TCP example, apportioning could be based on information obtained from the programmers. By understanding what type and how many SCOBOL (Screen COBOL) verbs are executed by each transaction, an apportioning scheme can be determined. Another form of apportioning is "message based apportioning." That is, the TCP resource is apportioned between each workload based on the number of messages sent/received by key process(es) in the transaction path. The volume and frequency of each transaction flowing through the TCP plays a role in apportioning the TCP among the workloads.

A variety of ways exist to do resource and entity apportioning. These are some examples of apportioning a shared resource. Use SURVEYOR to apportion individual entities within a workload that use shared resources.

Information Collection

With a service level agreement in place, SLOs clearly defined and quantified, and workloads characterized appropriately, SURVEYOR can be used to help manage the performance of the system and help performance management deliver "adequate service" to its user community. The information collected by SURVEYOR can tell performance management whether they are meeting the established SLOs, where a potential performance problem exists, which resources are most heavily used, and which workloads impose the greatest strain on system resources.

When deciding what performance information to collect and store, think about the commonly asked questions about system and workload performance. Use those questions to help decide what performance information to collect.

SURVEYOR can collect and store information about:

- resource utilizations,
- transaction rates,
- service times,
- message and byte rates,
- resource queueing, and
- within-the-system residence times.

Where to start

At a minimum, performance data should be collected for the processors, processes within the processors, disks and communication lines. A good starting point is to use the default configuration for MEAS-ATTR and MASK-ATTR provided by SURVEYOR. The CPU, DISC, PROCESS and LINE entities should be selected in the SURVEYOR MEAS-ATTR configuration section. The default entity selection in SURVEYOR provides the basic configuration needs for performance management.

If the initial performance information does not help answer the performance related question, then more information needs to be collected. The SURVEYOR MEAS-ATTR and MASK-ATTR configurations should be adjusted to collect additional performance information. Over time, a better understanding of the computing environment will allow performance management to be selective in what performance information is collected and stored. A more complex and detailed configuration using SURVEYOR can be achieved when a thorough understanding of the computing environment exists.

SURVEYOR Data Fields

SURVEYOR provides a comprehensive set of performance data fields associated with each entity. Only those data fields required for a specific environment need be saved. For each entity, unless configured otherwise, SURVEYOR uses its default set of data fields to specify what is collected and stored for each entity.

CPU Entity

For the SURVEYOR CPU entity, the AVERAGE-UTIL or the TOTAL-UTIL data field shows the utilization for the processor(s). The AVERAGE-QUEUE data field provides information on how many processes are waiting in the queue to use the processor. The SWAP-RATE tells how often memory needed to be swapped in or out. The DISPATCH-RATE is an indicator of how many processes are serviced by the processor. The DISC-IO-RATE shows the amount of disk activity attributed to the processor. The USER-PROC-UTIL and SYS-PROC-UTIL data fields show how much time the processor spent executing user processes and system processes. INTERRUPT-UTIL tells the amount of time the processor was executing interrupt handler processes. This data field is useful when trying to apportion system interrupt activity among multiple applications or workloads.

DISK Entity

Use the SURVEYOR DISC entity to measure consumption of disk resources on the system. DISC-UTIL and AVERAGE-QUEUE data fields provide information on how busy the disk is. By collecting performance information on logical and physical I/Os, performance management can understand what type of work the disk is doing. READ-IO-RATE and WRITE IO RATE provide information on the type of work being requested of a particular disk. The comparison of PHYSICAL-IO-RATE to LOGICAL-IO-RATE provides information on how many physical disk I/Os are needed for a logical or application I/O. Cache hit and miss rates are useful in determining necessary adjustments to cache memory size and activity on the disk. Cx-HIT- RATE and Cx-MISS-RATE could be monitored to further understand how the applications use the disk.

PROCESS Entity

PROCESS-CPU-UTIL of the SURVEYOR PROCESS entity indicates how much processor time a particular process used. The READY-TIME divided by PROCESS-CPU-UTIL shows the ratio of the time waiting for processor time to the time actually doing some work. The SEND-RATE and RECEIVE-RATE are measured to help understand the frequency of data passed into and out of a process. The PAGE-FAULT-RATE is monitored to know who is causing memory paging/swapping. The AVERAGE-QUEUE data field can indicate that a process is not getting enough processor time to complete all the work requested of it.

LINE Entity

The LOGICAL-IO-RATE, READ-RATE, WRITE-RATE and TOTAL-BYTE-RATE of the SURVEYOR LINE entity provide performance information about communication line rates and utilizations.

Default Entities

Most of the data fields identified above are defaults set up within the MASK-ATTR configuration section of SURVEYOR. The default MEAS-ATTRs and MASK-ATTRs are recommended as a starting point for collecting and storing some basic performance information. In many cases, other data fields for specific entities are selected to provide additional information. For example, cache hit/miss information is useful performance information. The cache hit data fields are not defaults in SURVEYOR and must be manually configured. Please refer to the SURVEYOR reference material for more information on default entities and their values.

Modeling

A model is a representation of some existing or planned object. An equation, rules of thumb, a collection of atomic numbers, and a graph can each be considered a model. Modeling is often used to predict future values or the outcome of specific events and is based on empirical analysis or observed reactions. Combining historical values with the model can result in a prediction of the future. How reliable this prediction is depends on the both the validity of the model and the integrity of the data or values used.

Performance management is often the focal point for information used in modeling. The modeling function itself is usually performed by the Capacity Planning organization. When planning capacity, historical information about system performance is critical. Growth of an on-line system can usually be expressed in terms of growth in either transactions for an application or additional applications. That growth information combined with baseline performance information forms a model to help a capacity planner project future resource needs.

SURVEYOR plays a key role in modeling. SURVEYOR not only collects, stores, and summarizes the historical information needed for modeling, it also provides certain modeling functions. Workloads provide the logical grouping needed to simplify and organize the resource consumption data into manageable units. Aggregation schemes are the SURVEYOR mechanism to provide data summarization. An aggregation scheme is configured to specify what, when, and how summarization is to occur.

System Modeling

In system modeling, the existing or planned object is a computer system, the applications within the system, or individual resources used by the applications. The values are historical data about the entity being modeled. For example, a model can be a simple relationship between server message receive rates and average system utilization. Based on past performance, a linear relationship can exist between two performance indicators - throughput and utilization. As system throughput increases, the average system utilization increases proportionally.

Typically, models are used to predict performance, cost, or size of a system, application, or a resource. Three frequently used types of models are: linear regression, consumption, and costing. Each model provides specific information to logically represent an aspect of an object. A linear regression model could use performance indicators to represent the overall system performance in terms of utilization. Workload performance information could be used as input to a consumption model. A combination of user statistics and system hardware/software pricing structures could be combined to represent the cost of a system.

Linear Regression

The forecast command of SURVEYOR provides linear regression modeling. The SURVEYOR forecast command uses historical data to predict future values. Linear regression is a technique that takes a set of data points and fits a straight line to the data by minimizing the distance between the data points and the line.

For SURVEYOR's linear regression to be "accurate," the historical data should have a linear shape. Not all systems have activity that grows in a linear fashion. Many businesses have seasonal trends. If the historical data does not have a linear "look" to it, then linear regression may not be appropriate. An example of a linear, a non-linear, and a seasonal data set is shown in Figure 2 on page 13.

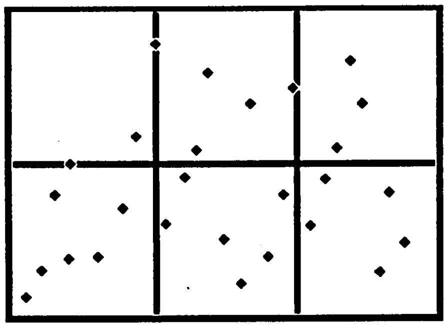


Figure 2a. Nonlinear data set.

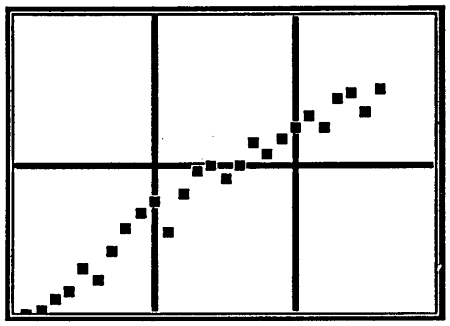


Figure 2b. Linear data set.

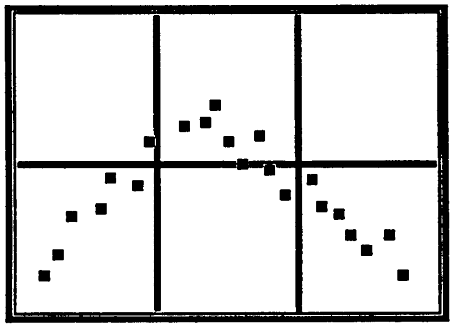


Figure 2c. Seasonal data set.

While linear regression is the forecasting technique used by SURVEYOR, that should not imply that the technique is applicable to all computer systems. Other techniques may be better suited to model a particular system that performs in a non-linear fashion. Queueing network models (mentioned earlier in this document) and consumption models (an example provide later in this document) are options that could be used.

When linear regression modeling is appropriate for a particular system or application, performance management must select the basis of the model. That is, a linear regression model uses historical data to predict future values. A particular type of value (specifically a SURVEYOR performance data field) must be selected on which to collect historical data and then predict the future values.

The AVERAGE-UTIL or the TOTAL-UTIL (of the CPU entity) are used in some cases as the basis for the model. Another indicator used to forecast is the RECEIVE-RATE of a key process of the application. Which performance data field is used as the basis of the linear regression model depends on the environment. The model provided by SURVEYOR can be based on any performance data field stored in the performance database.

As historical data is collected and stored about a particular data field, SURVEYOR uses this information to predict future values. SURVEYOR provides a high and a low value for each expected value presented. The high and low values form the 90% confidence range when using the model as a predictive tool.

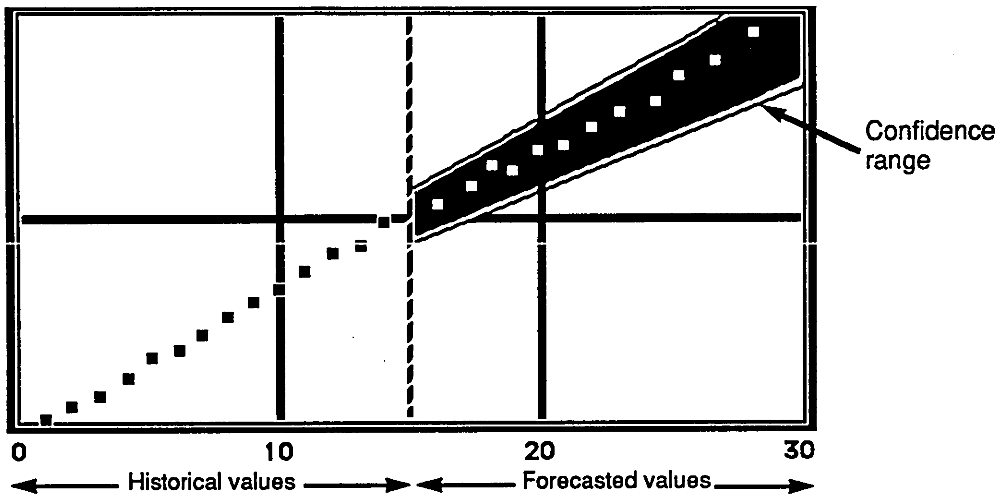


Figure 3. Example of forecast with confidence range.

Consumption Modeling

Consumption modeling is another type of modeling. SURVEYOR can provide the inputs to a consumption model. Consumption modeling is an analytical modeling mechanism. This type of model examines combinations of processes that use system resources to complete a transaction. Later in this document, what information SURVEYOR can provide and how to use that information to produce a consumption model is discussed.

Performance Monitoring

Monitoring system and workload performance is a key part of performance management. Using SURVEYOR, performance data is regularly collected and stored. Depending on the amount of performance information collected, monitoring system performance can be extremely time consuming and labor intensive. Performance monitoring looks only at key performance indicators. When irregularities are identified, performance management must be alerted. Typically, alerts are sent to operations consoles and other on-line monitoring devices. Exception reports can also be generated for performance management to review. The concept of performance monitoring is to identify and resolve potential problems before the user community is negatively affected.

Ideally, automated mechanisms should be used monitor system and workload performance. The automated mechanism should produce the alerts and exception reports for performance management.

Some tools exist to provide on-line monitoring of system performance. VIEWSYS, NSS and Enlighten are examples of such monitoring tools. SURVEYOR provides an additional monitoring tool. SURVEYOR's threshold feature and exception reporting feature helps performance management monitor system and workload performance.

High, low, and display threshold values are established by performance management. Thresholds define ranges of values that indicate a potential problem to performance management. These thresholds can be adjusted as needed. Thresholds can be established for any performance data field collected and stored by SURVEYOR. Utilizations, queue lengths, send/receive message rates and swapping rates are typical performance indicators with exception reporting thresholds defined. Usually, performance data is collected and stored on a daily basis. SURVEYOR examines the days' performance indicators looking for performance irregularities. SURVEYOR can initiate exception reports to alert performance management about indicators that fall outside the expected ranges.

The actual value or range of values defined as the threshold is dependent on a particular environment. If a particular system has no excess resource capacity, then thresholds values are set to alert performance management as soon as a capacity increase occurs. If excess resource capacity does exist, then the threshold value can be set to alert performance management only when some of the excess capacity gets used up. SURVEYOR allows performance management to establish threshold values for the performance indicators peculiar to a particular environment.

All of the standard performance reports supplied by SURVEYOR can be turned into exception reports. With thresholds defined, SURVEYOR provides reports that show only the exception data. Performance management can better utilize its time by examining exception reports instead of looking for performance anomalies in detailed performance data.

A section with examples of using SURVEYOR to monitor system performance appears later in this document. Examples are given for monitoring processor and disk activity.

Performance Database Management

Performance management must collect performance data on a regular basis. That performance data must be stored over time for trending purposes. Reporting on current as well as historical data is a requirement. Because of the volumes of performance data involved, performance management must reduce the detailed performance data to a manageable amount. Performance management needs a performance database (PDB) and a performance database manager to help with these functions. SURVEYOR provides all the necessary PBD management functions.

SURVEYOR is an important tool used by performance management. SURVEYOR provides the following features to help manage performance information:

- automated or manual data collection, reduction, and storage
- long term storage of performance data
- database maintenance of files/tables/indexes
- archiving and retrieving PDB data
- standard and customized threshold and exception reporting
- statistical functions for summarization
- workload tracking
- data exporting to PCs

The SURVEYOR performance database is a benefit to performance management. All the performance information, historical as well as current, is centrally located for easy access by performance management and capacity planners. Beyond providing a PDB, SURVEYOR automates some key functions involved in performance management. The following sections illustrate two of those key functions; measuring system performance and capacity planning.

Section 3: Monitoring System Performance with SURVEYOR - An Example

SURVEYOR can automate the every day, time consuming process of monitoring system performance. When used to monitor system performance, SURVEYOR combines the best of MEASURE and ENFORM plus other new and useful functions. Using SURVEYOR, the user can automatically recognize system performance anomalies and then use MEASURE to conduct more detailed performance analysis.

Based on the system's particular characteristics, the user can configure reports that will be generated only if certain exceptions are met. For example, a processor report can be configured so a report will be generated only if processor swaps exceed one per second. Another could be configured for disk and generate a report only when disk busy exceeds 35%. Exception reports can be configured based on any data field (counter) stored in the PDB.

In this section, "performance monitoring" is defined as the process of **recognizing** system performance anomalies. "Performance analysis and tuning" is the process of **identifying the cause** of these system performance anomalies.

This section is not meant to be a guide for performance analysis and tuning. Every system has it's own unique characteristics just as every analyst has their own "favorite" performance counters. The purpose of this section is to demonstrate how to use these counters and to encourage other ideas for customizing SURVEYOR.

Table 4-1 (Significant Counters and their Maximum Values) in the *MEASURE User's Guide* (part number 84157) is a good starting point for selecting performance counters for use with SURVEYOR. These guidelines may or may not be applicable to your specific environment.

System Description

The system used to generate and collect measurement data was a four processor TXP system with four mirrored volumes. Each of the four processors was a primary for one mirrored volume. The PATHWAY system was driven by terminal simulators and consisted of four different types of transactions.

The following examples discuss exception reports and how they can be used to monitor processor and disk performance. The processor example shows how common conditions can be detected with exception reports. The conditions are processor utilization imbalance, over-utilization, and memory pressure (swapping). These conditions were caused by transient activity during the steady state measurement period. The next set of examples deal with disk and show how imbalances in the disk subsystem can be detected with exception reports. In this example, one volume had it's cache set incorrectly.

Monitoring Processor Performance

The default CPU-STATS report is very useful for monitoring processor performance. As a first step, the report allows evaluation of processor resource "consumption" - resource consumption including, at a minimum, both utilization and memory usage. By specifying the workload ^ALL-CPUS, a CPU-STATS report can be produced reflecting a total system view of processor performance. The ^ALL-CPUS specification combines all processor data fields into one data unit.

Because all data from the last PDB update is to be included in the example, the FROM and TO parameters of the report object are not set. Data for all intervals of the last PDB update are included when FROM and TO are omitted. A CPU-STATS report of ^ALL-CPUS is produced as follows (the test system had four processors):

```
assume report
set detailed on
print cpu-stats ^all-cpus

SURV 1024 Warning. Since FROM, TO and FOR are all
undefined only the data produced by the last operation
will be used as the target data for the following report(s)
```

```
                  Processor   Average  Total    Average Swap
Date        Time  No. Type    Util     Util     Queue   Rate
1988-03-09  7:55  ^ALL-CPUS   17.530    70.123  0.874   2.876
1988-03-09  7:58  ^ALL-CPUS   22.208    88.832  1.130   1.920
1988-03-09  8:01  ^ALL-CPUS   14.152    56.611  0.747   1.765
1988-03-09  8:04  ^ALL-CPUS   33.448   133.792  1.947   0.621
1988-03-09  8:07  ^ALL-CPUS   36.203   144.815  2.149   0.388
1988-03-09  8:10  ^ALL-CPUS   44.956   179.826  3.670   1.150
1988-03-09  8:13  ^ALL-CPUS   61.133   244.532  5.117   0.000
1988-03-09  8:16  ^ALL-CPUS   62.314   249.256  5.478   0.400
1988-03-09  8:19  ^ALL-CPUS   61.415   245.660  5.233   0.000
1988-03-09  8:22  ^ALL-CPUS   63.257   253.030  5.633   2.533
1988-03-09  8:25  ^ALL-CPUS   67.597   270.388  6.547   5.138
1988-03-09  8:28  ^ALL-CPUS   61.173   244.695  5.172   0.038
1988-03-09  8:31  ^ALL-CPUS   61.695   246.782  5.170   0.005
1988-03-09  8:34  ^ALL-CPUS   61.022   244.088  5.178   0.000
```

The measurement period was from March 9th at 7:55 until 8:34 with a measurement interval of 3 minutes. It seems the system was in a steady state beginning with the 8:13 interval. The AVERAGE-UTIL column is TOTAL-UTIL divided by the number of processors in the system. AVERAGE-QUEUE and SWAP-RATE are system totals - not per processor. The AVERAGE-UTIL data field is useful when trying to determine how well balanced processors are. This usage is discussed later in the section.

Processor Reports - Analyzing Individual Processor Activity

Specifying processor numbers rather than ^ALL-CPUS, the CPU-STATS report lists data fields for each individual processor. This allows a more detailed look at processor performance. Analysis of the steady state environment is desired so FROM and TO are specified. Note the processor specification of (0,1,2,3) rather than ^ALL-CPUS ("*" could be used also. It includes interval data for ^ALL-CPUS and individual processors).

CPU-STATS (0,1,2,3) lists each data field on a per processor basis. Therefore, AVERAGE-UTIL is equal to TOTAL-UTIL. Excluding the intervals of 8:22 and 8:25, processor utilization is fairly well balanced though processors 2 and 3 are slightly out of line with processors 0 and 1. Processors 0 and 1 are very near the CPU-STATS^ALL-CPUS reported AVERAGE-UTIL of approximately 61%. Processors 2 and 3 are slightly lower and higher, respectively, than this average. From the CPU-STATS ^ALL-CPUS report presented earlier, a system wide increase in SWAP-RATE and TOTAL-UTIL occurred at 8:22 and 8:25. CPU-STATS (0,1,2,3) shows that this increased activity took place in processor 1. Processor 0 may have been affected also because of it's slight increase in utilization compared with other intervals.

```
assume report
set detailed on
set from 1988-03-09 8:13
set to *
print cpu-stats (0,1,2,3)
```

```
                     Processor Average  Total   Average Swap
Date        Time     No.Type   Util     Util    Queue   Rate
1988-03-09  8:13  0  TXP   60.349   60.349   1.248   0.000
1988-03-09  8:13  1  TXP   62.839   62.839   1.360   0.000
1988-03-09  8:13  2  TXP   56.855   56.855   1.143   0.000
1988-03-09  8:13  3  TXP   64.489   64.489   1.366   0.000
1988-03-09  8:16  0  TXP   61.971   61.971   1.348   0.000
1988-03-09  8:16  1  TXP   62.590   62.590   1.332   0.000
1988-03-09  8:16  2  TXP   56.661   56.661   1.128   0.000
1988-03-09  8:16  3  TXP   68.034   68.034   1.670   0.400
1988-03-09  8:19  0  TXP   61.135   61.135   1.305   0.000
1988-03-09  8:19  1  TXP   62.581   62.581   1.290   0.000
1988-03-09  8:19  2  TXP   57.268   57.268   1.176   0.000
1988-03-09  8:19  3  TXP   64.676   64.676   1.462   0.000
1988-03-09  8:22  0  TXP   63.314   63.314   1.360   0.000
1988-03-09  8:22  1  TXP   69.155   69.155   1.796   2.533
1988-03-09  8:22  2  TXP   56.419   56.419   1.088   0.000
1988-03-09  8:22  3  TXP   64.142   64.142   1.389   0.000
1988-03-09  8:25  0  TXP   68.880   68.880   1.631   0.000
1988-03-09  8:25  1  TXP   77.446   77.446   2.222   5.138
1988-03-09  8:25  2  TXP   58.369   58.369   1.215   0.000
1988-03-09  8:25  3  TXP   65.693   65.693   1.479   0.000
1988-03-09  8:28  0  TXP   60.491   60.491   1.243   0.000
1988-03-09  8:28  1  TXP   61.972   61.972   1.311   0.027
1988-03-09  8:28  2  TXP   56.268   56.268   1.120   0.000
1988-03-09  8:28  3  TXP   65.964   65.964   1.498   0.011
1988-03-09  8:31  0  TXP   60.149   60.149   1.198   0.000
1988-03-09  8:31  1  TXP   63.177   63.177   1.354   0.005
1988-03-09  8:31  2  TXP   58.081   58.081   1.181   0.000
1988-03-09  8:31  3  TXP   65.375   65.375   1.437   0.000
1988-03-09  8:34  0  TXP   58.976   58.976   1.202   0.000
1988-03-09  8:34  1  TXP   63.068   63.068   1.360   0.000
1988-03-09  8:34  2  TXP   56.216   56.216   1.113   0.000
1988-03-09  8:34  3  TXP   65.828   65.828   1.503   0.000
```

Exception Reports - Processor Utilization

To best interpret and apply exception reports requires a familiarity with the system under analysis. Familiarity means knowing typical levels of utilization of the system's components or, in the case of memory, knowing what memory pressure exists, if any. Once these areas and their typical values are identified, exception reports can be configured that produce data only when a certain data field exceeds a specified value. In the following examples, the exact threshold values selected are unimportant. What is important is the concept behind exception report configuration. To take full advantage of exception reports, the reader must be familiar with the characteristics of the system under analysis.

The CPU-STATS report is a good starting foundation for building a set of exception reports. A general guideline is that a TXP processor not exceed 60% to 70% busy. A CPU-STATS ^ALL-CPUS report can be configured alerting staff that the guideline had been exceeded. The exception would be based on TOTAL-UTIL. A four processor example would require the TOTAL-UTIL exception be set to 280% (4 x 70%). If based on AVERAGE-UTIL, a CPU-STATS report can help identify CPU load imbalances. This example is discussed below.

The CPU-STATS ^ALL-CPUS report indicated that peak processor utilization occurred between 8:13 and 8:34. Average CPU utilization ranged from a low of 61.022 to a high of 67.597. Already mentioned was the curious rise in processor consumption between 8:22 and 8:25 (utilization and swaps). In this example, the AVERAGE-UTILs for these two intervals will be ignored because some unexpected event occurred as indicated by SWAP-RATE. In this test environment, swaps are not typical events so they will be investigated later. Excluding 8:22 and 8:25, the range of AVERAGE-UTIL during peak steady state load was 61.022 to 62.314. This is not to suggest ignoring certain intervals - they were ignored here so the following examples can best demonstrate how unusual intervals can be recognized.

Based on the "normal" range of AVERAGE-UTIL (61.022 to 62.314), a threshold of 65% TOTAL-UTIL is specified in the following example. CPU utilization imbalances can be recognized this way. Specifying a high TOTAL-UTIL threshold of 65% results in the exception report including only those processors more than 65% busy. True production systems might need a larger "margin" or another data field used as the exception.

When configuring customized reports, the order report fields are defined (via SET REPORT FIELD) determines, from left to right, the order they are displayed. Because CPU utilization is the data field of interest, it will be SET first.

A CPU-STATS exception report for the steady state period of March 9, 1988 from 8:13 through 8:34 is configured as follows:

```
Assume report
Set Title      "CPU Utilization Exception Report"
Set Detailed   On
Set Field      Cpu TOTAL-UTIL
Set Field      Cpu AVERAGE-QUEUE
Set Field      Cpu SWAP-RATE
Set Field      Cpu DISC-IO-RATE
Set High       Cpu TOTAL-UTIL 65
Set From       1988-03-09 8:13
Set To         1988-03-09 8:34
Print          Cpu-stats (0,1,2,3)
```

```
             CPU Utilization Exception Report

                 Processor  Total   Average  Swap   Disc IO
Date        Time No.Type    Util    Queue    Rate   Rate
1988-03-09  8:16  3  TXP    68.034  1.670    0.400  22.144
1988-03-09  8:22  1  TXP    69.155  1.796    2.533  21.527
1988-03-09  8:25  0  TXP    68.880  1.631    0.000  28.122
1988-03-09  8:25  1  TXP    77.446  2.222    5.138  21.272
1988-03-09  8:25  3  TXP    65.693  1.479    0.000  21.205
1988-03-09  8:28  3  TXP    65.964  1.498    0.011  21.444
1988-03-09  8:31  3  TXP    65.375  1.437    0.000  21.372
1988-03-09  8:34  3  TXP    65.828  1.503    0.000  21.738
```

As configured, this exception report reveals several anomalies. The first is processor utilization. It may be necessary to review the previous CPU-STATS reports to see the connection, but this system seems to have two distinct processor utilization "exceptions." Most significant is the rise in processor 1's utilization during the intervals for 8:22 and 8:25. There seems to be a relationship with swap activity during this time because SWAP-RATE increases significantly with the increase in utilization. Though processor 0 has but one entry in this exception report, review of the CPU-STATS (0,1,2,3) report shows a corresponding increase, though less marked, in utilization for processor 1. Perhaps a connection exists. Processor 3 is another matter. Relative to the other processors (excepting the intervals for 8:22 and 8:25), processor 3 is consistently busier than the other CPUs. Conversely, processor 2 is consistently less utilized. In the real world, the degree of imbalance between processors 2 and 3 is usually insignificant.

Exception Reports - Recognizing Memory Pressure

The previous example demonstrated an exception report based on utilization. The example that follows demonstrates how other data fields can be used as exception qualifiers for report generation. Note the SET order and the corresponding order of the report columns. There are many possibilities for exception reports. Exactly what is configured depends on what needs monitoring on the system.

```
Assume report
Set Title      "CPU Swaps Exception Report"
Set Detailed   On
Set Field      Cpu SWAP-RATE
Set Field      Cpu MEMORY-PAGES
Set Field      Cpu AVG-MEM-QUEUE
Set Field      Cpu TOTAL-UTIL
Set High       Cpu SWAP-RATE .5
Set From       1988-03-09 8:13
Set To         1988-03-09 8:34
Print          cpu-stats (0,1,2,3)
```

```
                CPU Swaps Exception Report

                Processor  Swap   Memory     Avg        Total
Date       Time No. Type   Rate   Pages      Mem Queue  Util
1988-03-09 8:22  1  TXP    2.533  4096.000   0.040      69.155
1988-03-09 8:25  1  TXP    5.138  4096.000   0.075      77.446
```

Memory Pressure - Finding the Cause

The two previous exception reports indicated unusually high SWAP-RATEs during the intervals for 8:22 and 8:25. There are two principal reasons why swaps occur - the first being a true shortage of memory the second being transient process activity. The data suggests transient activity because swapping was not evident during all interval periods. A true shortage of memory usually exhibits itself as continuous swap activity for all intervals and for all processes, including non-transients.

The following exception report is based on the PROCESS-STATS report. The PAGE-FAULT-RATE and AVERAGE-DURATION fields are selected because they will tell who is causing the swaps and whether the swaps are due to transient process creations. Specifying a HIGH exception of 0.001 will display only process entities that experienced swapping, however small the amount.

```
Assume report
Set Detailed  On
Set Field     Process PAGE-FAULT-RATE
Set Field     Process AVERAGE-DURATION
Set High      Process PAGE-FAULT-RATE .001
Set From      1988-03-09 8:13
Set To        *
Print         process-stats *
```

```
                Process                              Page Fault Average
   Date     Time Name    Program Filename                Rate    Duration
1988-03-09 8:13  $TPB1   $SYSTEM.SYSTEM.PATHTCP2        0.038    179.973
1988-03-09 8:13  $TPC1   $SYSTEM.SYSTEM.PATHTCP2        0.177    179.940
1988-03-09 8:13  $TPD1   $SYSTEM.SYSTEM.PATHTCP2        0.022    180.050
1988-03-09 8:16          $SYSTEM.SYSTEM.EDIT            0.111      5.961
1988-03-09 8:16$DELAY    $SYSTEM.SYSTEM.DELAY           0.011     70.724
1988-03-09 8:16  $REQ0   $SYSTEM.SYSTEM.PATHTCP2        0.005    180.074
1988-03-09 8:16  $TPA1   $SYSTEM.SYSTEM.PATHTCP2        0.005    179.812
1988-03-09 8:16  $TPC1   $SYSTEM.SYSTEM.PATHTCP2        0.005    180.096
1988-03-09 8:16  $TPD1   $SYSTEM.SYSTEM.PATHTCP2        0.005    180.060
1988-03-09 8:16  ^MISC                                  0.333    151.736
1988-03-09 8:22  $TPB    $SYSTEM.SYSTEM.PATHTCP2        0.005    179.825
1988-03-09 8:22  ^MISC                                  2.677     83.386
1988-03-09 8:25  $NCP    $SYSTEM.SYS00.OSIMAGE          0.011    179.933
1988-03-09 8:25  ^MISC                                  5.588     56.534
1988-03-09 8:28$CONSOL$SYSTEM.SYS00.OSIMAGE             0.005    179.366
1988-03-09 8:28  $DELAY $SYSTEM.SYSTEM.DELAY            0.011    170.628
1988-03-09 8:28  ^MISC                                  0.011    160.287
1988-03-09 8:31  $REQ0$SYSTEM.SYSTEM.PATHTCP2           0.005    180.045
```

To improve readability SET ORDER-BY PROCESS PAGE-FAULT-RATE could have been specified but since only a small set of entities satisfy the exception, ordering by time (the default) doesn't make the report difficult to read. As expected, the significant intervals are 8:22 and 8:25. SWAP-RATEs for all other intervals are insignificant.

The entity causing the swaps to occur is shown as ^MISC. ^MISC is a default workload that includes many common system utilities such as FUP, PUP, BACKUP, RESTORE, etc. This example illustrates why it is a good idea not to delete the MEASURE data file after it is updated into the PDB. Using MEASURE and "windowing" into the correct interval, a set of reports could be produced for each program file included in the ^MISC workload and the actual cause of the swapping determined.

The usefulness of grouping processes into workloads cannot be over-emphasized. The value of this capability is demonstrated in the capacity planning section. Though in most cases not recommended, the following example shows what happens when the ^MISC workload is deleted. Deleting ^MISC is done as follows:

```
Assume configuration
Ungroup process ^misc
Alter configuration workload
```

After deleting the workload, the data for the period being analyzed will have to be deleted and re-updated (MEASURE data file needed). Once this is done, print the report again (notice the report configuration is identical to the previous example):

```
Assume report
Set Detailed  On
Set Field     Process PAGE-FAULT-RATE
Set Field     Process AVERAGE-DURATION
Set High      Process PAGE-FAULT-RATE .001
Set From      1988-03-09 8:13
Set To        *
Print         process-stats *
```

```
                Process                             Page Fault  Average
Date        Time Name    Program Filename              Rate     Duration
1988-03-09  8:13 $TPB1   $SYSTEM.SYSTEM.PATHTCP2      0.038     179.973
1988-03-09  8:13 $TPC1   $SYSTEM.SYSTEM.PATHTCP2      0.177     179.940
1988-03-09  8:13 $TPD1   $SYSTEM.SYSTEM.PATHTCP2      0.022     180.050
1988-03-09  8:16         $SYSTEM.SYS00.COMINT         0.061       2.403
1988-03-09  8:16         $SYSTEM.SYS00.FUP            0.116       2.857
1988-03-09  8:16         $SYSTEM.SYS00.PUP            0.150       8.401
1988-03-09  8:16         $SYSTEM.SYSTEM.EDIT          0.111       5.961
1988-03-09  8:16$DELAY   $SYSTEM.SYSTEM.DELAY         0.011      70.724
1988-03-09  8:16 $REQ0   $SYSTEM.SYSTEM.PATHTCP2      0.005     180.074
1988-03-09  8:16 $TCL5   $SYSTEM.SYS00.TACL           0.005     179.500
1988-03-09  8:16 $TPA1   $SYSTEM.SYSTEM.PATHTCP2      0.005      79.812
1988-03-09  8:16 $TPC1   $SYSTEM.SYSTEM.PATHTCP2      0.005     180.096
1988-03-09  8:16 $TPD1   $SYSTEM.SYSTEM.PATHTCP2      0.005     180.060
1988-03-09  8:22         $SYSTEM.SYS00.COMINT         0.072      82.783
1988-03-09  8:22         $SYSTEM.SYS00.PUP            2.605       2.285
1988-03-09  8:22 $TPB1   $SYSTEM.SYSTEM.PATHTCP2      0.005     179.825
1988-03-09  8:25         $SYSTEM.SYS00.FUP            0.105       7.832
1988-03-09  8:25         $SYSTEM.SYS00.PUP            5.483       2.458
1988-03-09  8:25 $NCP    $SYSTEM.SYS00.OSIMAGE        0.011     179.933
1988-03-09  8:28         $SYSTEM.SYS00.COMINT         0.005       7.109
1988-03-09  8:28         $SYSTEM.SYS00.FUP            0.005       6.483
1988-03-09  8:28$CONSOL$SYSTEM.SYS00.OSIMAGE          0.005     179.366
1988-03-09  8:28 DELAY   $SYSTEM.SYSTEM.DELAY         0.011     170.628
1988-03-09  8:31 $REQ0   $SYSTEM.SYSTEM.PATHTCP2      0.005     180.045
```

The PAGE-FAULT-RATEs for $SYSTEM.SYS00.PUP speak for themselves. There is an obvious correlation between the processor SWAP-RATE and the process PAGE-FAULT-RATE for 8:22 and 8:25. In this example, the interval for 8:22 has a single entry for PUP where, in reality, 16 separate copies were run. During the 8:25 interval 40 copies were run. Uniquely naming each copy of PUP would have resulted in 56 separate entries (notice the entries for the TCPs). We can be fairly certain that a true shortage of memory doesn't exist because the AVERAGE-DURATION for the PUP processes is only a few seconds. That, plus the fact that no other processes were swapping make it a safe bet.

Deleting the ^MISC workload does not eliminate the need for future MEASURE analysis. As mentioned, a total of 56 separate copies of PUP were run. The exception report does not show what individual process fault activity was. Deleting ^MISC does narrow the program file set down to $SYSTEM.SYS00.PUP, however. Measure produces the reports necessary to show the unique memory consumption on a per process basis.

Other Processor Exception Reports

The following exception reports are based on the suggestions in Table 4-1 of the MEASURE User's Guide. The SET HIGH represents the data field the exception report is based on. What other data fields are included in the report are entirely up to the user.

For queue lengths:

```
Assume report
Set Detailed On
Set Field  Cpu AVERAGE-QUEUE
Set Field  Cpu TOTAL-UTIL
Set Field  Cpu SWAP-RATE
Set Field  Cpu DISC-IO-RATE
Set Field  Cpu CACHE-HIT-RATE
Set High   Cpu AVERAGE-QUEUE 1.0
Print      cpu-stats (0,1,2,3)
```

For dispatches:

```
Assume report
Set Detailed On
Set Field  Cpu DISPATCH-RATE
Set Field  Cpu TOTAL-UTIL
Set Field  Cpu USER-PROC-UTIL
Set Field  Cpu SYS-PROC-UTIL
Set Field  Cpu INTERRUPT-UTIL
Set Field  Cpu PROC-OVHD-UTIL
Set Field  Cpu SEND-UTIL
Set High   Cpu DISPATCH-RATE 300   -TXP VALUE
Print      cpu-stats (0,1,2,3)
```

Monitoring Disk Subsystem Performance

Just as exception reports were useful for monitoring processor performance, exception reports are equally useful for monitoring disk subsystem performance. As did the preceding processor examples, the following examples will demonstrate how exception reports can aid performance management of the disk subsystem.

Due to the nature of mirrored volumes, some data fields are reported on a logical volume basis rather than a per physical spindle basis. See the SURVEYOR Reference manual for further details.

A different MEASURE data file was used in the following disk analysis. The steady state period began at 7:21 and was determined in the same manner as it was in the processor examples. All the subsequent examples are for the steady state period.

Disk Reports - Analyzing Individual Disk Spindle Activity

The following example is a customized version of the DISC-UTIL-SUMMARY report. To best report physical spindle activity, DISC-UTIL was omitted, PRIMARY-DISC-IOS and MIRROR-DISC-IOS added. This permits better correlation with the guidelines in the *MEASURE User's Guide* (Table 4-1). PRIMARY and MIRROR-DISC-IOS include seeks and therefore do not represent just read and write counts. Because seeks are included in these data fields, do not try to "match" processor DISC-IO-RATEs with those of the disk. The processor DISC-IO-RATE data field does not include seek activity. This is not unlike MEASURE.

The data fields PRIMARY-UTIL, MIRROR-UTIL, PRIMARY-DISC-IOS, and MIRROR-DISC-IOS provide the necessary data to configure reports based on Table 4-1 in the *MEASURE User's Guide* (TXP disk busy not to exceed 35%, TXP disk rate not to exceed 25).

```
Assume report
Set Detailed On
Set Field    Disc PRIMARY-UTIL
Set Field    Disc MIRROR-UTIL
Set Field    Disc PRIMARY-DISC-IOS
Set Field    Disc MIRROR-DISC-IOS
Set From     1988-03-09 7:21
Set To       *
Print        disc-util-summary ($d1, $d2, $data, $system)
```

```
                Disc     Primary Mirror Primary  Mirror
Date       Time Name     Util    Util   Disc IOs Disc IOs
1988-03-09 7:21 $D1      49.376  55.287 35.354   42.416
1988-03-09 7:21 $D2      26.557  31.506 20.126   31.326
1988-03-09 7:21 $DATA    14.578  18.159 10.160   15.110
1988-03-09 7:21 $SYSTEM  26.206  31.815 18.393   29.338
1988-03-09 7:24 $D1      50.885   6.607 36.154   42.848
1988-03-09 7:24 $D2      24.502  29.896 18.810   30.032
1988-03-09 7:24 $DATA    15.719  18.903 10.627   16.315
1988-03-09 7:24 $SYSTEM  29.423  34.793 20.126   30.399
1988-03-09 7:27 $D1      49.265  54.631 35.288   42.820
1988-03-09 7:27 $D2      24.542  29.707 18.455   28.927
1988-03-09 7:27 $DATA    13.666  16.910  9.549   14.538
1988-03-09 7:27 $SYSTEM  29.573  34.244 20.960   30.866
1988-03-09 7:30 $D1      50.342  56.223 36.104   43.337
1988-03-09 7:30 $D2      24.799  29.932 18.976   30.254
1988-03-09 7:30 $DATA    14.393  16.872  9.716   14.160
1988-03-09 7:30 $SYSTEM  29.018  34.095 20.293   30.226
1988-03-09 7:33 $D1      46.947  53.543 33.620   41.605
1988-03-09 7:33 $D2      24.606  29.964 18.971   29.410
1988-03-09 7:33 $DATA    14.231  16.820  9.860   14.920
1988-03-09 7:33 $SYSTEM  28.051  33.150 19.522   30.094
1988-03-09 7:36 $D1      47.822  54.966 34.609   42.483
1988-03-09 7:36 $D2      26.389  31.790 20.121   32.055
1988-03-09 7:36 $DATA    15.338  18.388 10.338   15.715
1988-03-09 7:36 $SYSTEM  27.363  32.720 18.977   29.909
1988-03-09 7:39 $D1      49.633  56.412 35.688   43.144
1988-03-09 7:39 $D2      25.112  30.139 19.177   30.588
1988-03-09 7:39 $DATA    15.352  18.221 10.493   15.670
1988-03-09 7:39 $SYSTEM  26.143  31.820 18.305   29.605
```

Clearly, $D1 is more utilized than the other volumes. Utilization for both primary and mirror is nearly double that of other busy volumes. I/O rates (seeks included) are much higher than those for other volumes. This degree of imbalance could result in a significant performance degradation.

Exception Reports - Disk Utilization

There are many options when configuring disk exception reports. One option is to configure the exception report based on either PRIMARY-UTIL or MIRROR-UTIL.

Usually, only in read intensive environments do PRIMARY-UTIL and MIRROR-UTIL differ significantly. Exception reports based on DISC-UTIL potentially can mask an overly busy physical spindle. In these environments other exception reports based on different data fields can close such "gaps" and compliment one another. Complimentary exception reports may be as simple as having one based on PRIMARY-UTIL and another based on MIRROR-UTIL. Use what best fits the characteristics of the system under measurement. The following exception report is based on PRIMARY-UTIL exceeding 35%. This report makes $D1's over-utilization much more obvious.

```
Assume report
Set Detailed On
Set Field   Disc PRIMARY-UTIL
Set Field   Disc MIRROR-UTIL
Set Field   Disc PRIMARY-DISC-IOS
Set Field   Disc MIRROR-DISC-IOS
Set High    Disc PRIMARY-UTIL 35
Set From    1988-03-09 7:21
Set To      *
Print       disc-util-summary ($d1, $d2, $data, $system)

                Disc  Primary  Mirror  Primary  Mirror
Date      Time  Name  Util     Util    Disc IOs Disc IOs
1988-03-09 7:21 $D1   49.376   55.287  35.354   42.416
1988-03-09 7:24 $D1   50.885   56.607  36.154   42.848
1988-03-09 7:27 $D1   49.265   54.631  35.288   42.820
1988-03-09 7:30 $D1   50.342   56.223  36.104   43.337
1988-03-09 7:33 $D1   46.947   53.543  33.620   41.605
1988-03-09 7:36 $D1   47.822   54.966  34.609   42.483
1988-03-09 7:39 $D1   49.633   56.412  35.688   43.144
```

For all intervals, only $D1 exceeds the threshold of 35% PRIMARY-UTIL.

Disk Reports - Imbalances in Disk Subsystem Utilization

Like MEASURE, the Surveyor DISCOPEN report shows the "physical" view of activity on a disk. Unlike the FILE report which doesn't report on secondary partitions, DISCOPEN reports all file activity on a disk. Secondary partitions, as well as primary partitions, are listed in the DISCOPEN report. Additionally, DISCOPEN reports I/O and cache hit rates. These data fields correspond to the CPU-STATS DISC-IO-RATE and CACHE-HIT-RATE making a disk file's impact on disk easy to recognize. For these reasons, DISCOPEN is usually the most useful report for identifying disk consumers. DISCOPEN reports make it easy to identify which file should be moved to another volume.

Every file opened on the volume is listed but combined into one entry per file opened. For example, there were 52 openers of ACCOUNT, BRANCH, and TELLER. The interval entries for each of these files is the sum of all data records for each open.

To verify this, add up IO-RATE and CACHE-HIT-RATE for an interval from the DISCOPEN report. It will very closely correspond to the DISC-IO-RATE and CACHE-HIT-RATE of the controlling processor and to the READ-RATE, WRITE-RATE, and CACHE-HIT-RATEs of the physical disk. Any differences are due to rounding or omission of data records from the MEASURE data file due to storage thresholds.

The DISCOPEN-STATS report is generated as follows:

```
Assume report
Set Detailed On
Set From    1988-03-09 7:21
Set To      *
Print       discopen-stats $d1.*.*
```

Date	Time	File Name	IO Rate	Cache Hit Rate
1988-03-09	7:21	$D1.CUNION.DBASE	11.260	2.433
1988-03-09	7:21	$D1.CUNION.LOG01	.227	0.000
1988-03-09	7:21	$D1.CUNION.LOG02	1.521	0.000
1988-03-09	7:21	$D1.TESTDB.ACCOUNT	12.516	3.105
1988-03-09	7:21	$D1.TESTDB.BRANCH	6.533	4.688
1988-03-09	7:21	$D1.TESTDB.TELLER	7.811	4.083
1988-03-09	7:24	$D1.CUNION.DBASE	10.555	1.966
1988-03-09	7:24	$D1.CUNION.LOG01	0.288	0.000
1988-03-09	7:24	$D1.CUNION.LOG02	1.532	0.000
1988-03-09	7:24	$D1.TESTDB.ACCOUNT	13.799	3.288
1988-03-09	7:24	$D1.TESTDB.BRANCH	6.766	4.855
1988-03-09	7:24	$D1.TESTDB.TELLER	8.605	4.600
1988-03-09	7:27	$D1.CUNION.DBASE	10.666	2.283
1988-03-09	7:27	$D1.CUNION.LOG01	0.282	0.000
1988-03-09	7:27	$D1.CUNION.LOG02	1.599	0.005
1988-03-09	7:27	$D1.TESTDB.ACCOUNT	13.094	3.461
1988-03-09	7:27	$D1.TESTDB.BRANCH	7.099	5.122
1988-03-09	7:27	$D1.TESTDB.TELLER	8.649	4.627
1988-03-09	7:30	$D1.CUNION.DBASE	11.399	2.444
1988-03-09	7:30	$D1.CUNION.LOG01	0.260	0.000
1988-03-09	7:30	$D1.CUNION.LOG02	1.593	0.000
1988-03-09	7:30	$D1.ST1.ST1	0.050	0.016
1988-03-09	7:30	$D1.TESTDB.ACCOUNT	12.999	3.183
1988-03-09	7:30	$D1.TESTDB.BRANCH	6.588	4.755
1988-03-09	7:30	$D1.TESTDB.TELLER	9.022	4.755
1988-03-09	7:33	$D1.CUNION.DBASE	10.949	2.344
1988-03-09	7:33	$D1.CUNION.LOG01	0.255	0.000
1988-03-09	7:33	$D1.CUNION.LOG02	1.604	0.005
1988-03-09	7:33	$D1.TESTDB.ACCOUNT	12.388	3.194
1988-03-09	7:33	$D1.TESTDB.BRANCH	6.811	4.888
1988-03-09	7:33	$D1.TESTDB.TELLER	8.244	4.377
1988-03-09	7:36	$D1.CUNION.DBASE	10.844	2.211
1988-03-09	7:36	$D1.CUNION.LOG01	0.299	0.000
1988-03-09	7:36	$D1.CUNION.LOG02	1.671	0.000
1988-03-09	7:36	$D1.TESTDB.ACCOUNT	12.705	3.272
1988-03-09	7:36	$D1.TESTDB.BRANCH	6.444	4.666
1988-03-09	7:36	$D1.TESTDB.TELLER	8.466	4.388
1988-03-09	7:39	$D1.CUNION.DBASE	11.749	2.644
1988-03-09	7:39	$D1.CUNION.LOG01	0.327	0.000
1988-03-09	7:39	$D1.CUNION.LOG02	1.449	0.005
1988-03-09	7:39	$D1.TESTDB.ACCOUNT	13.294	3.350
1988-03-09	7:39	$D1.TESTDB.BRANCH	6.110	4.400
1988-03-09	7:39	$D1.TESTDB.TELLER	8.743	4.583

The following CPU-STATS (0,1,2,3) report provides meaningful insight into the problem with $D1 if the system configuration and workload is clearly understood. The system under test had one disk primaried in each of the four processors. Volumes $D2, $DATA, and $SYSTEM all have CACHE-HIT-RATEs well over 50%. These three volumes were primaried in processors 3, 2, and 0, respectively. $D1 was primaried out of processor 1. $D1's cache hit rate is far less than this though the total system I/O rate (CACHE-HIT-RATE plus DISC-IO-RATE) is nearly the same as that for $D2 and $SYSTEM. In this particular test, $D1's cache was "under-configured" which resulted in the poor cache hit rate and corresponding increase in physical I/O activity. Admittedly, the following CPU-STATS report may seem awkward for this sort of analysis. However, if expected physical I/O rates and/or cache hit rates were known, exception reports would better highlight the problem. PUP, FUP, and MEASURE could also be used to conduct cache and file analysis.

A CPU-STATS (0,1,2,3) report customized to emphasize the I/O system is configured as follows:

```
Assume report
Set Detailed  On
Set Field     Cpu DISC-IO-RATE
Set Field     Cpu CACHE-HIT-RATE
Set Field     Cpu SWAP-RATE
Set Field     Cpu SYS-PROC-UTIL
Set From      1988-03-09 7:21
Set To        *
Print         cpu-stats (0,1,2,3)
```

```
                 Processor  Disc IO  Cache Hit  Swap  Sys Proc
Date       Time  No. Type   Rate     Rate       Rate  Util
1988-03-09 7:21  0   TXP    21.294   26.744   0.000   20.266
1988-03-09 7:21  1   TXP    36.744   15.294   0.083   26.807
1988-03-09 7:21  2   TXP    13.361   20.150   0.127   16.728
1988-03-09 7:21  3   TXP    22.361   29.461   0.050   21.291
1988-03-09 7:24  0   TXP    23.261   27.583   0.000   21.436
1988-03-09 7:24  1   TXP    37.850   16.022   0.000   27.826
1988-03-09 7:24  2   TXP    14.261   20.527   0.000   16.863
1988-03-09 7:24  3   TXP    21.255   27.611   0.383   20.604
1988-03-09 7:27  0   TXP    23.250   29.350   0.000   21.960
1988-03-09 7:27  1   TXP    36.816   16.361   0.000   27.353
1988-03-09 7:27  2   TXP    12.738   19.505   0.000   15.630
1988-03-09 7:27  3   TXP    20.683   26.883   0.011   19.691
1988-03-09 7:30  0   TXP    23.033   28.288   0.000   21.492
1988-03-09 7:30  1   TXP    37.761   16.000   0.250   28.198
1988-03-09 7:30  2   TXP    12.938   19.822   0.000   16.011
1988-03-09 7:30  3   TXP    21.411   28.016   0.011   20.411
1988-03-09 7:33  0   TXP    22.100   28.500   0.000   21.012
1988-03-09 7:33  1   TXP    35.711   15.922   0.000   26.547
1988-03-09 7:33  2   TXP    13.105   20.061   0.000   15.941
1988-03-09 7:33  3   TXP    20.866   27.377   0.000   20.013
1988-03-09 7:36  0   TXP    21.694   27.588   0.000   20.749
1988-03-09 7:36  1   TXP    36.472   15.750   0.000   26.839
1988-03-09 7:36  2   TXP    13.577   20.472   0.000   16.712
1988-03-09 7:36  3   TXP    22.522   29.300   0.000   21.171
1988-03-09 7:39  0   TXP    21.350   27.094   0.000   20.344
1988-03-09 7:39  1   TXP    37.472   15.705   0.000   27.164
1988-03-09 7:39  2   TXP    13.794   19.794   0.000   16.511
1988-03-09 7:39  3   TXP    21.416   28.133   0.000   20.434
```

The preceding examples demonstrate the strength of SURVEYOR when used for performance monitoring. With properly configured exception reports, SURVEYOR can provide valuable information about a system's "health."

However, performance monitoring is only one part of a complete system management strategy. Capacity planning is just as crucial as performance monitoring.

Section 4: SURVEYOR as an Aid for Capacity Planning - An Example

Consumption Modeling

There are a variety of methods of modeling computer systems that can use the information contained in the Performance Data Base. The SURVEYOR product contains linear regression techniques for trend analysis. The *SURVEYOR User's Guide* explains how to use these built in features. Linear Regression is a good technique to use for estimating future growth in a system when there is good reason to believe that growth in the system will continue as it has been in the past; i.e. no new major applications being added or changes in transaction proportions (percentage of one type of transaction to another).

Another modeling technique which is commonly used is Consumption Modeling. This approach is applicable when there is some knowledge outside the system that the transaction proportions will change. This could be due to changes in user population or business priorities. This technique uses the data within a SURVEYOR performance data base and builds an analytical model.

Consumption Modeling views a Tandem system as a combination of processes that consume resources on behalf of a given transaction or application. Some of these processes are unique to a given transaction type and some are shared between multiple transaction types. For the processes that are shared, consumption modeling provides a method of apportioning the work they do amongst the transaction types that share it.

The model results in a demand per transaction at each physical service center. A physical service center is a set of similar devices that provide a particular service. For example, a node of CPUs is a physical service center providing CPU processing time. The DISK drives attached to a system of CPUs are a physical service center providing physical I/O and storage. For a given transaction, consumption modeling defines the demand the transaction has for each physical service center.

CPU Planning

CPU planning for a Tandem system requires the understanding and the analysis of many different system entities and their constraints. For example, there must be enough ports for the terminal population. There also must be enough CPU cycles to get the necessary work done. Consumption modeling determines how many seconds of CPU time are needed for each transaction type. The information from the model can then be used to determine how many CPU seconds will be needed for a particular transaction rate.

The most difficult part of consumption modeling is determining the subsystems within the CPU that each particular transaction uses. The next hardest part is determining what portion of each subsystem is used by each transaction. Most of this work needs to be done outside of the SURVEYOR environment. SURVEYOR can help in the latter task, if the system being analyzed is well designed and/or well instrumented. The first task, that of determining the pieces of the system each transaction/application uses, can be accomplished by talking with the software developers and system management personnel.

The information about what processes a transaction uses is represented in a transaction flow diagram. The transaction flow diagram is a pictorial representation of a transaction's flow through a system. An example of a transaction flow diagram is shown in Figure 4.

Disk Planning

Disk planning must satisfy two constraints: sufficient space on all the disk drives to hold the necessary information for a system and enough disk "power" to handle the number of requests for that information. Because SURVEYOR can help in the latter, it is the focus of this discussion.

An application issues a "logical" request to the disk process. This logical request results in a certain number of "system" requests. These system requests consist of index level I/Os plus data level I/Os that are a direct result of the one logical request. Some of these system requests are satisfied from the disk cache. Others result in a request to the physical disk. These requests to the physical disk are referred to as "physical" I/Os. Using general terminology, here are the definitions for these different I/Os:

Logical I/Os	=	Requests from application to read, write, update, or delete a record. (Receive Rate on the Disk Processes)
System I/Os	=	Cache Hits + Physical Reads + Physical Writes
Physical I/Os	=	Physical Reads + Physical Writes

When planning how many disk spindles are required for a system, it is necessary to know how many physical I/Os are likely to occur in a given time interval. This, of course depends on the cache hit rate in the system. The higher the cache hit rate, the lower number of physical requests that occur. A consumption model can be built either by associating logical requests to physical requests or by associating logical requests to system requests and then making the association between system requests and physical requests. In the second case, increases in cache hit rate can be investigated.

The consumption model building demonstrated in this example assumes all logical disk requests are equal. There is no differentiation between requests to different types of file structures (key-sequenced, entry-sequenced and unstructured). The example also considers READ, WRITE, UPDATE, or DELETE operations as equivalent. While these assumptions might not be accurate for all systems, they are valid for the following example and simplifies the necessary analysis.

SURVEYOR Consumption Modeling

SURVEYOR's ability to group processes into workloads and to summarize information about these workloads over time is particularly helpful for capacity planning. (It is assumed that you have read the *SURVEYOR User's Guide* and are familiar with both the GROUP command for creating workloads and the AGGREGATE object for summarizing information. If not, please review the appropriate sections in the User's Guide before continuing.)

The Tandem system is a message-based system. That means processes "talk" to each other by sending and receiving messages to and from each other. By tracing the messages of a given transaction through the system, it is possible to estimate what resources the transaction is utilizing. Messages received and messages replied to by individual processes are recorded by MEASURE and can be maintained in the SURVEYOR Performance data base (PDB). These performance metrics are essential to building a consumption model of a system. For a particular process, the messages received per second are in the field RECEIVE-RATE and the number of messages sent are in the field SEND-RATE. Even though these are in the default SURVEYOR configuration, make sure that in any customized configuration the following two statements are included :

```
XX$  Select Process SEND-RATE
XX$  Select Process RECEIVE-RATE
```

Workloads

Setting up the workloads in the system is a very important task for accurate consumption modeling. The workloads should represent either the transactions or applications that you wish to model. Care needs to be taken in deciding what processes can be grouped with others and which need to be grouped separately.

Before establishing the GROUPs in SURVEYOR, identify the transaction types or the applications that are going to be used in the model of the system. Once these particular groupings are defined, the Tandem processes or programs that are executed by each of these transactions need to be identified. Those processes/programs that are shared by transactions need to be distinguished from those that are unique to a particular application (an example of a shared process is the TCP processes in a PATHWAY environment). Some processes may only be shared by two "groups" while others may be used by every transaction or application in the system.

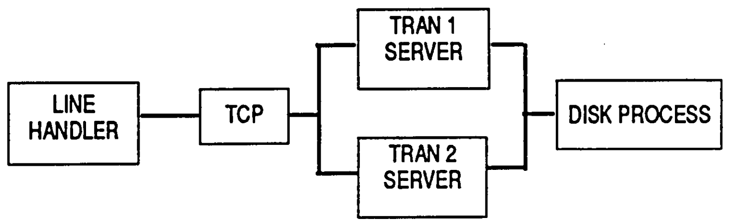


Figure 4. Sample Transaction Flow Diagram.

The transaction flows for all the transactions being modeled need to be understood. This includes determining the point of entry into the system (usually a line handler) and each and every process executed on its behalf. The transaction flow also identifies those processes within the path that make logical I/O requests. In essence, the desired final outcome is a flowchart of the transactions being modeled. A simple PATHWAY application with two transactions, is shown in Figure 4.

From this diagram, it is easy to see that the messages sent from the TRAN 1 SERVER and the TRAN 2 SERVER are being received by the disk process. The MESSAGES-SENT field in the PROCESS entity for these two server types would represent the number of logical I/Os performed per second. If the processes TRAN 1 SERVER and TRAN 2 SERVER were called once to process a single transaction of their type, then the RECEIVE-RATE field in the PROCESS entity for each of these two server types would represent the number of transactions per second for the two different transaction types.

In this example, there are two workloads (Transaction Type 1 and Transaction Type 2). The Line Handler, TCP and Disk Processes need to be apportioned among the two transactions. The Line Handler and the TCP are divided equally between both transaction types; that is, each transaction going through each of these processes will use the same amount of that resource. This assumption is made because there is no way to measure how much of each resource a particular transaction type uses.

The disk process is divided equally between every logical I/O in the system, each of which can be viewed as a "transaction" to the disk process. To model this system, six SURVEYOR groups are created. They are:

One containing the Line Handler processes
One containing the TCP processes
One containing Tran 1 Servers
One containing Tran 2 Servers
One containing Disk Processes
One containing All processes running

When building a model, all work in the system must be accounted for. The purpose of having a workload that contains all the processes running in the system is to determine the amount of "OTHER" activity that is being performed and is contributing to CPU consumption. It is important that

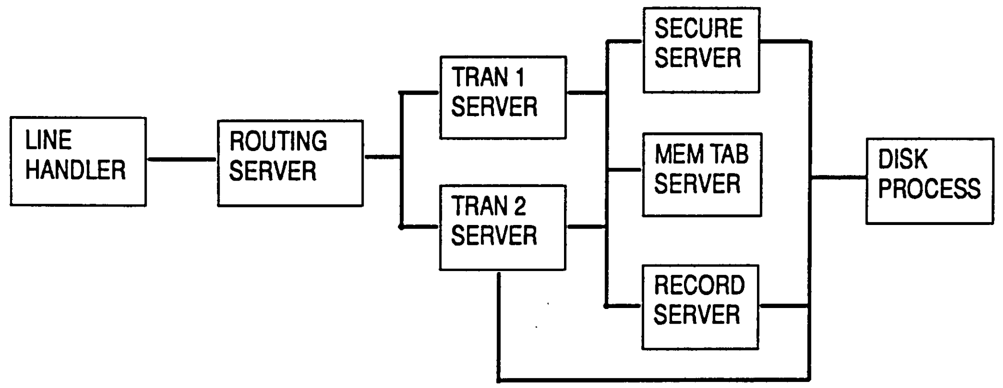


Figure 5. Transaction flow diagram for a complex system.

the thresholds for storing and displaying information are set up to insure all information needed is present.

In a more complex environment, there may be many servers talking to each other as well as the disk process. This complicates the flowchart as well as the analysis. An example of a flowchart for this type of system is shown in Figure 5.

In this example, every process except for TRAN 1 SERVER and TRAN 2 SERVER are shared by the two transaction types. Does that mean they should all be put in a GROUP together? No. To analyze the transaction flows of this system requires that almost every "process" on the chart be put into its own GROUP.

The LINE HANDLER and the ROUTING SERVER can be put into the same GROUP. They are both apportioned by the total number of transactions being processed.

The TRAN 1 SERVER and the TRAN 2 SERVER need to be in their own GROUPs so that the transaction rates for each transaction type can be determined as well as the service times.

All disk process activity needs to be GROUPed together. This allows the model to view the disk as a single resource. Also, this type of GROUPing is necessary to help identify the miscellaneous process activity (by process of elimination - the ^all-processes GROUP minus all the other defined GROUPS equals the miscellaneous process activity).

The SECURE SERVER, MEM TAB SERVER and RECORD SERVER are put into individual GROUPs. There is no easy way to figure out how many messages out of the TRAN 2 SERVER went to the MEM TAB SERVER or how many went to the SECURE SERVER. By looking at the RECEIVE-RATE on the MEM TAB SERVER, you determine the number of messages from both the TRAN 1 and TRAN 2 SERVERs that went to the MEM TAB SERVER. Without the use of user defined counters within this server, the service time within the MEM TAB SERVER for a TRAN 1 request versus a TRAN 2 request cannot be determined. An average service time for a request to the MEM TAB SERVER is the best that can be found. The same holds true for the SECURE and RECORD servers.

The TRAN 2 SERVER sends messages to the disk process as well as to the three shared servers (SECURE, MEM TAB and RECORD). It is necessary to determine how many messages out of the TRAN 2 SERVER were logical I/O requests. To do this, add up the number of MESSAGES-RECEIVED by the three servers (SECURE, MEM TAB, and RECORD) and subtract that number from the total number of MESSAGES-SENT by the TRAN 1 SERVER and the TRAN 2 SERVER. The result will be the number of messages sent to the disk process.

There are eight GROUPs required to build the model. They are:

One for the Line Handler and ROUTING SERVER
One for the TRAN 1 SERVER
One for the TRAN 2 SERVER
One for the SECURE SERVER
One for the MEM TAB SERVER
One for the RECORD SERVER
One for the DISK PROCESSES
One for All processes running

The point of this discussion is that there is not a one-to-one relationship between SURVEYOR "groups" or workloads and the workloads in a consumption model.

Aggregation Schemes

The aggregation feature in SURVEYOR summarizes data as requested by the user. This summarization could result in information summarized on any user-defined timeframe (e.g. daily, weekly, monthly). When using SURVEYOR data to build the consumption model, it is recommended that a separate AGGREGATE be created for this purpose. By doing so, changes to the consumption model will not impact the ongoing data gathering used for performance monitoring.

The time period which is used for the SELECT statement of the AGGREGATE object is based upon knowledge of system usage and the actual measurement that is being taken. The most important criteria for the time selection is making sure that all the transactions that the model will be based on have been exercised in the system. If, for example, it is known that all transactions in the model are entered between 1:00 and 1:30 pm, then an AGGREGATE can be set up as follows:

```
-- Aggregate to be used in building
-- Consumption Model of System
XX$ Set Aggregate Title "Consumption Model Input"
XX$ Set Aggregate Records (cpu,disc,process)
XX$ Set Aggregate Stats (avg,std,p90)
XX$ Set Aggregate Span Daily
XX$ Set Aggregate Summarization Automatic
XX$ Set Aggregate Select * 1:00 To 1:30
XX$ Add Aggregate Model-Input
```

This will create the average value, the standard deviation and 90th percentile values for data values between 1:00 and 1:30p.m. The standard deviation is a measure of the dispersion of the data points from the average. For example, if the measurement contained 10 data points between 1:00 and 1:30 and the average data point value was 12 with a standard deviation of 8, then it is known that those 10 points were not all close to the value of 12. If the standard deviation was 2, then all 10 data points were close to value of 12.

For modeling, the standard deviation should be relatively small in comparison to the average value being used. If it is not, then it is recommended that the statistic P90 be used in place of the average. The P90 statistic gives the value at which 90 percent of the data points between 1:00 and 1:30 fall below. This is more conservative than using an average value. Whether the average or the P90 statistic is used, the steps for creating the model are the same.

Extracting Information From SURVEYOR

After determining the aggregation schemes, you must specify the information to extract from SURVEYOR. The pieces of information needed for building a model are listed below.

TOTAL CPU BUSY
TOTAL PROCESS BUSY
TOTAL INTERRUPT BUSY
TOTAL DISK BUSY

For each SHARED Process:	TOTAL PROCESS BUSY REQUEST RATE MADE TO IT
For each UNIQUE Process for a transaction type :	TOTAL PROCESS BUSY REQUEST RATE MADE TO IT REQUEST RATE TO SHARED WORKLOADS
For the DISK ACTIVITY :	CACHE HIT RATE READ and WRITE RATE

The Performance metrics from SURVEYOR that are used to obtain that information are:

CPU object	INTERRUPT-UTIL AVERAGE-UTIL DISC-IO-RATE
DISC object	PRIMARY-UTIL MIRROR-UTIL LOGICAL-IO-RATE
PROCESS object	PROCESS-CPU-UTIL RECEIVE-RATE SEND-RATE

If the model building is being done for a "mixed" system, meaning there is a combination of TNSII's, TXP's and/or VLX's, then the model building would use the "normalized" fields of TNSII-UTIL in place of AVERAGE-UTIL and PROCESS-CPU-UTIL in the CPU and PROCESS objects respectively. (See the *SURVEYOR Reference Manual,* Section 4 for a description of these fields). This is necessary because a process executing on a TXP will run faster on a VLX. Normalization is the process of using a common baseline for different representations of values. The normalization process takes the number of TXP (or VLX) seconds a process uses and computes the number of TNSII seconds that process would use.

Model Building–An Example

The easiest way to describe consumption model building is to present a complete example. A description of the system to be modeled and a step by step description of each stage of the model building process follows.

System Description and Objective

The example is a PATHWAY system consisting of four major transactions which comprise the majority of work being done. They are TRAN1, TRAN2, TRAN3, and TRAN4. Each transaction passes through a line handler, a TCP, a server process, and the disk process. There is other work being done in the system during the day; this work will continue in the future. All the transaction types are entered between 12:20 and 12:40 each day.

The objective is to determine the effect of a 20% increase in TRAN4 transaction rate on CPU and disk utilizations.

Step 1 - Create the Transaction Flow Diagram

The flow diagram in Figure 6 depicts the system described above.

Step 2 - Set Up SURVEYOR Configuration

The SURVEYOR configuration for this system requires eight GROUPs to be set up. They are:

One for the Line Handlers
One for the TCPs
One for each of the four Transaction Servers
One for the Disk Processes
One for all processes running

The Line Handlers in this system are processes executing program files $DATA.BOTH.USIM and $DATA.BOTH.SSIM (Do not be concerned that these are not true "line handlers.")

The TCPs are processes executing program file $SYSTEM.SYSTEM.PATHTCP2.

This system is unique in that three of the four transactions are executing the same program file. Therefore, to distinguish them from each other, the processes have been named to uniquely identify each transaction type. Those processes with process names starting with $B and followed by three numbers are transaction type TRAN1. Those processes with process names starting with $I and followed by three numbers are transaction type TRAN2.

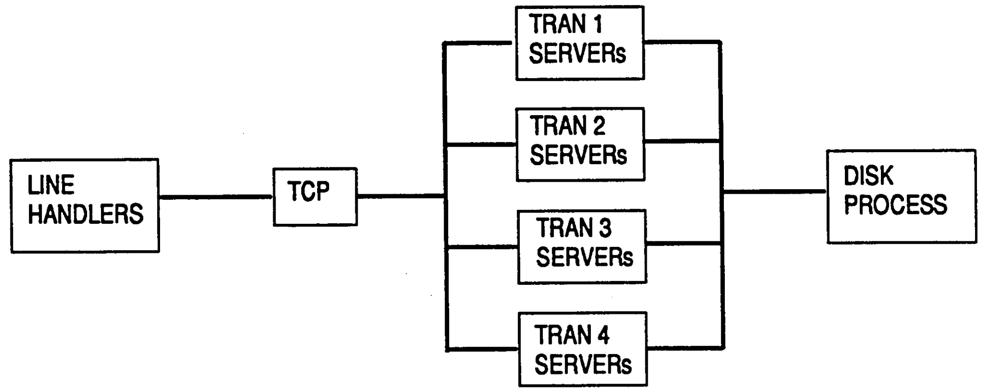


Figure 6. PATHWAY system flow diagram.

Those processes with process names starting with $U and followed by three numbers are transaction type TRAN3.

TRAN4 are processes executing program file $DATA.ST1.SERVERO.

The Disk Processes in this system are identified by their device names $SYSTEM, $DATA, $D1 and $D2.

All other processes in the system are identified using wildcards for volume, subvolume and program file name. This field gives a summary of all user and system processes run.

The required input to SURVEYOR to configure these workloads is:

```
-
- redefine thresholds
-
   set threshold entity device, field average-util
   delete threshold store
   set threshold entity disc, field disc-util
   delete threshold store
   set threshold entity process, field process-cpu-util
   delete threshold store
-
- now set up the appropriate configuration
-
 assume configuration
 Add Cpu *
 Add Process *
 Add Disc $*
-
group process ^line-hand = $data.both.usim + &
$data.both.ssim
-
group process ^tcp = $system.system.pathtcp2
-
group process ^tran1 = $b000 + $b001 + $b002 + &
$b003 + $b004 + $b005 &+ $b006 + $b007 + $b008 + &
$b009 + $b010 + $b011 + $b012 + $b013 + $b014 + &
$b015
-
group process ^tran1 = $b016 + $b017 + $b018 + &
$b019 + $b020 + $b021 + $b022 + $b023 + $b024 + &
$b025 + $b026 + $b027 + $b028 + $b029 + $b030 + &
$b031 + $b032
-
group process ^tran2 = $l000 + $l001 + $l002 + $l003 &
+ $l004 + $l005 + $l006 + $l007 + $l008 + $l009 + $l010 &
+ $l011 + $l012 + $l013 +  $l014 + $l015
group process ^tran2 = $l016 + $l017 + $l018 + $l019 &
+ $l020 + $l021 + $l022 + $l023 + $l024 + $l025 + $l026 &
+ $l027 + $l028 + $l029 + $l030 + $l031 + $l032
-
group process ^tran3 = $u000 + $u001 + $u002 + &
$u003  + $u004 + $u005 + $u006 + $u007 + $u008 + &
$u009 + $u010 + $u011 + $u012 + $u013 + $u014 + $u015
group process ^tran3 = $u016 + $u017 + $u018 + &
$u019 + $u020 + $u021 + $u022 + $u023 + $u024 + &
$u025 + $u026 + $u027 + $u028 + $u029 + $u030 + &
$u031 + $u032
-
group process ^tran4 = $data.st1.servero
-
group process ^disk-procs  = $system + $data + $d1 &
+ $d2
-
group process ^all-processes = $*.*.*
-
replace configuration
```

Note: Some of the GROUP definitions contain the same group name. This results in the second group of processes being concatenated to the first group of processes. This is necessary because they cannot all fit in one command.

Notice that this configuration deletes certain threshold values. When building a consumption model the goal is to account for every last CPU second (or millisecond!) a transaction is responsible for. The defaults for these threshold values are set up to ignore entries with very low values (<0.05%). For consumption modeling, these entries are needed. Discussions of thresholds can be found in the *SURVEYOR User's Guide.*

The final element to add to the configuration is the aggregate for model building. It is identical to the aggregate defined for model building except the SELECT statement is customized for the particular system being modeled.

```
-- Aggregate to be used in building
-- Consumption Model of System
Set Aggregate Title "Consumption Model Input"
Set Aggregate Records (cpu,disc,process)
Set Aggregate Stats (avg,std,p90)
Set Aggregate Span Daily
Set Aggregate Summarization Automatic
Set Aggregate Select * 12:20 to 12:40
Add Aggregate Model-Input
```

The customization of the SURVEYOR configuration is now complete so that the necessary information from the data base for consumption modeling can be extracted.

Step 3 - Loading the SURVEYOR Data Base

Loading the SURVEYOR data base is described in the *SURVEYOR User's Guide*. The aggregate Model-Input is summarized automatically every day for the prior day's information.

Step 4 - Creating the Customized Reports

Below is a sample set of reports that have been customized for this example. There are three reports; one for the CPU entity, one for the DISC entity, and one for the PROCESS entity. Each report appears in three sections; one for each statistic specified in the AGGREGATE object (avg, standard deviation and P90). Each report is shown with the corresponding SURVEYOR commands that were in the OBEY file used for printing the report.

CPU Report

The CPU Report shows the average CPU utilization and the number of physical reads and writes performed per second.

Here is the input to SURVEYOR to create the CPU report:

```
-
assume report
Reset report *
Set Report Title       "Model CPU Report"
Set Report summary model-input
Set Report Field       cpu average-util
Set Report Field       cpu disc-io-rate
Set Report Field       cpu cache-hit-rate
Set Report From        1988-01-04 00:00
Set Report To          1988-01-04 23:59
Set Report Order-by Time
Set Report export      off
Set Report Display     brief
Set Report Width       132
print /out $s.#cpuinfo / cpu-util *
-
```

The following formatted report is the result of this OBEY file. Note that three separate reports are created, one for each statistic in the aggregate definition.

```
                   Model CPU Report
                         Avg

                  Processor   Average  Disc IO  Cache Hit
Date         Time  No.  Type     Util     Rate       Rate
1988-01-04  0:00   0   TXP    65.728   25.452    32.100
1988-01-04  0:00   1   TXP    66.357   24.482    31.580
1988-01-04  0:00   2   TXP    67.852   12.777    22.108
1988-01-04  0:00   3   TXP    65.661   23.949    32.314
1988-01-04  0:00^ALL-CPUS     66.399   86.662   118.103

                   Model CPU Report
                         Std

                  Processor   Average  Disc IO  Cache Hit
Date         Time  No.  Type     Util     Rate       Rate
1988-01-04  0:00   0   TXP     0.571    0.739     1.245
1988-01-04  0:00   1   TXP     1.125    0.825     1.222
1988-01-04  0:00   2   TXP     0.794    0.371     0.564
1988-01-04  0:00   3   TXP     0.755    0.651     0.803
1988-01-04  0:00^ALL-CPUS      0.598    0.969     1.284
```

Model CPU Report
P90

Date	Time	Processor No.	Type	Average Util	Disc IO Rate	Cache Hit Rate
1988-01-04	0:00	0	TXP	66.133	26.261	33.266
1988-01-04	0:00	1	TXP	67.464	25.294	32.883
1988-01-04	0:00	2	TXP	68.606	13.138	22.472
1988-01-04	0:00	3	TXP	66.482	24.722	33.316
1988-01-04	0:00	^ALL-CPUS		67.171	87.859	119.781

Disk Report

The disk report contains the information on the number of logical I/O requests that were performed per second. It also contains the utilizations for the primary and the mirror disks as well as the processor utilization for the disk processes handling the drive.

```
-
Reset report *
Set Report Title      "Model DISC I/O Report"
Set Report summary model-input
Set Report Field      Disc  logical-io-rate
Set Report Field      Disc  primary-util
Set Report Field      Disc  mirror-util
Set Report Field      Disc  processor-util
Set Report From       1988-01-04 00:00
Set Report To         1988-01-04 23:59
Set Report Order-by Time
Set Report export     off
Set Report Display    brief
Set Report Width      132
print /out drinfo/ disc-rate-detailed
($system,$data,$d1,$d2)
-
```

The formatted report from this OBEY file is shown below. As before, there are three separate sections, one for each statistic.

Model DISC I/O Report
Avg

Date	Time	Disc Name	Logical IO Rate	Primary Util	Mirror Util	Processor Util
1988-01-04	0:00	$D1	20.978	30.915	37.525	23.829
1988-01-04	0:00	$D2	21.833	28.798	33.955	23.737
1988-01-04	0:00	$DATA	19.088	13.979	16.918	15.865
1988-01-04	0:00	$SYSTEM	21.413	32.053	37.782	23.658

Model DISC I/O Report
Std

Date	Time	Disc Name	Logical IO Rate	Primary Util	Mirror Util	Processor Util
1988-01-04	0:00	$D1	0.767	1.372	1.531	0.764
1988-01-04	0:00	$D2	0.429	1.160	0.604	0.546
1988-01-04	0:00	$DATA	0.191	0.562	0.681	0.240
1988-01-04	0:00	$SYSTEM	0.674	0.739	1.168	0.717

Model DISC I/O Report
P90

Date	Time	Disc Name	Logical IO Rate	Primary Util	Mirror Util	Processor Util
1988-01-04	0:00	$D1	21.800	32.359	38.990	24.560
1988-01-04	0:00	$D2	22.355	30.296	34.624	24.435
1988-01-04	0:00	$DATA	19.355	14.592	17.532	16.181
1988-01-04	0:00	$SYSTEM	22.044	32.957	38.999	24.331

Process Report

The third report includes information about the GROUPs or workloads that were defined in the configuration. This report gives the utilizations for each group as well as the send and receive rates. The average duration is also reported so that transient activity can be identified.

Here is the SURVEYOR input for the customized report:

```
-
Reset report *
Set Report Title  "Model Category Process Report"
Set Report summary model-input
Set Report Field      Process  PROCESS-CPU-UTIL
Set Report Field      Process  RECEIVE-RATE
Set Report Field      Process  SEND-RATE
Set Report Field      Process  AVERAGE-DURATION
Set Report From       1988-01-04 00:00
Set Report To         1988-01-04 23:59
Set Report Order-by Time
Set Report export     off
Set Report Display    brief
Set Report Width      132
print /out procinf/ process-stats  (^all-processes, &
^tran1, ^tran2, ^tran3,^tran4,^disk-procs,&
^line-hand,^tcp)
```

The output from this report request appears below:

Model Process Report
Avg

Date	Time	Process Name	Process CPU Util	Receive Rate	Send Rate	Average Duration
1988-01-04	0:00	^ALL-PROCESSES	222.844	230.604	229.337	179.949
1988-01-04	0:00	^DISK-PROCS	87.091	163.421	79.935	179.901
1988-01-04	0:00	^LINE-HAND	14.200	17.180	0.477	179.948
1988-01-04	0:00	^TCP	83.219	30.619	65.033	179.966
1988-01-04	0:00	^TRAN1	3.627	1.141	6.935	179.967
1988-01-04	0:00	^TRAN2	1.153	1.883	1.883	179.985
1988-01-04	0:00	^TRAN3	14.233	6.434	19.600	179.997
1988-01-04	0:00	^TRAN4	18.387	7.738	54.144	179.994

Model Process Report
Std

Date	Time	Process Name	Process CPU Util	Receive Rate	Send Rate	Average Duration
1988-01-04	0:00	^ALL-PROCESSES	1.623	1.824	1.687	0.229
1988-01-04	0:00	^DISK-PROCS	0.833	1.329	0.740	0.458
1988-01-04	0:00	^LINE-HAND	0.123	0.105	0.237	0.219
1988-01-04	0:00	^TCP	0.535	0.299	0.484	0.088
1988-01-04	0:00	^TRAN1	0.208	0.069	0.385	0.068
1988-01-04	0:00	^TRAN2	0.033	0.047	0.047	0.062
1988-01-04	0:00	^TRAN3	0.262	0.123	0.397	0.003
1988-01-04	0:00	^TRAN4	0.273	0.112	0.726	0.017

Model Process Report
P90

Date	Time	Process Name	Process CPU Util	Receive Rate	Send Rate	Average Duration
1988-01-04	0:00	^ALL-PROCESSES	225.271	233.211	231.827	180.211
1988-01-04	0:00	^DISK-PROCS	88.323	165.360	81.016	180.452
1988-01-04	0:00	^LINE-HAND	14.359	17.322	0.833	180.192
1988-01-04	0:00	^TCP	83.970	31.000	65.688	180.084
1988-01-04	0:00	^TRAN1	3.885	1.233	7.433	180.045
1988-01-04	0:00	^TRAN2	1.199	1.950	1.950	180.037
1988-01-04	0:00	^TRAN3	14.502	6.577	20.077	180.001
1988-01-04	0:00	^TRAN4	18.780	7.900	55.177	180.019

Step 5 - Building The Model

With the information in a concise form, the mathematical modeling can begin. Refer back to the section titled Extracting Information from SURVEYOR, for the list of items needed for model building. The flow diagram shown in Figure 6 will be used in gathering the pieces for every transaction. The two additional work-loads, INTERRUPT and OTHER will be discussed after the individual transaction workloads have been modeled.

SURVEYOR reports "utilization;" this paper discusses "busy." Utilization is "busy-ness" per second. Since SURVEYOR reports everything as rates, it reports busy-ness as utilization. To calculate the percent busy, divide the utilization rate by 100. For example, if a process has a utilization of 15.00, it was busy 0.15 (15.00/100) out of 1.00, or 15 percent of the time.

The first step in modeling is to determine whether to use the average values or the 90th percentile values. By comparing the standard deviations of the values for the PROCESS entities (found in the Std Report) to the average values (found in the Avg Report), it is evident that the system is in a steady state. For example, the average Process CPU Util for the group ^ALL-PROCESSES is 222.844 and the standard deviation is 1.623. The standard deviation is small compared to the average (1/137th). It is reasonable to use the average value for this system.

CPU Consumption Model

This section uses assumptions and laws of Operational Analysis. Operational Analysis, developed by Jeff Buzen and Peter Denning, is "a pragmatic philosophy of computer system analysis using queueing network models."[1] The definitions for the laws we will be using are:

***Flow Balance Assumption*:**	The number of completions within a given interval is equal to the number of arrivals during the same interval.
***Utilization Law*:**	Utilization = Transaction rate x transaction demand

For each process that a transaction executes, we can determine the demand using the utilization law. Then, by summing the individual demands at each process (for a given transaction type), the total process demand for a transaction type is found. Finally, we add to that a portion of the interrupt processing in the system to arrive at the CPU demand for the transaction.

The steps involved at each process is to determine the number of requests that are executed within the second. This is where the flow balance assumption is used. We assume that the number of completions by a particular process is equal to the number of requests made to it. In most cases the number of requests is equal to the RECEIVE-RATE as reported by SURVEYOR; for some, such as the TCP, this does not hold true.

There are four transaction types in the system to be modeled: TRAN1, TRAN2, TRAN3, and TRAN4. The first step of the modeling process is to determine the rate at which these four transaction types enter the system. From the transaction flow diagram in Figure 6 (page 36) it can be seen that while the transaction follows the flow, it's transaction type cannot be determined until the point where it either goes to TRAN1 server, TRAN2 server, TRAN3 server, or TRAN4 server. The RECEIVE RATE at each of these servers then represents the number of that type of transaction requesting service. The Process Report (Avg) shows the values for the four transaction types as:

TRAN1:	1.141 transactions per second
TRAN2:	1.883 transactions per second
TRAN3:	6.434 transactions per second
TRAN4:	7.738 transactions per second

The total number of transactions per second is 17.196.

1) Lazowska, et. al. 1984. *Quantitative System Performance*, , Prentice Hall. pg. xii. For a complete description of the laws, please refer to this work.

Table 1. Consumption Model information.

Tran Type	Line Handler	TCP	Server	Disk Process	TOTAL
TRAN1	0.008				
TRAN2	0.008				
TRAN3	0.008				
TRAN4	0.008				

All data is stated in seconds

Using the transaction flow diagram shown in Figure 6, the analysis starts at the left (line handler), and proceeds through each process in the path: TCP, servers, and disk processes. The table above will be filled in at each step in the model building exercise.

Line Handler

The line handler processes requests from all four transaction types. The RECEIVE-RATE on the Process Report (page 40) for the group ^LINE-HAND represents the transaction rate to the line handler processes. In this example, the RECEIVE-RATE on the line handlers should equal the transaction rate in the system since the line handlers are only used by the application being modeled.

The CPU Utilization for the line handler processes is found in the Process Report (page 40) under the heading PROCESS CPU UTIL for the group ^LINE-HAND. The value is 14.20%. Using the Utilization Law, the demand per transaction at the TCP is:

$$\text{Demand per transaction} = \frac{\text{utilization}}{\text{transaction rate}}$$

$$= \frac{14.20\%}{17.196} = 0.008 \text{ sec/transaction}$$

(Note: the answer is rounded to milliseconds)

TCP Processes

TCP processes are responsible for function key operations and application transactions. The utilization of the TCP reflects both of these activities. When apportioning TCP activity over the application transactions, the function key activity must also be accounted for. Therefore, the RECEIVE-RATE on the TCP servers is not used as the transaction rate; the total application transaction rate is used. This apportioning forces the function key and screen activity to be equally divided between all the transactions.

The CPU Utilization for the TCP processes is found in the Process Report under the heading PROCESS CPU UTIL for the group ^TCP. The value is 83.219%. Using the Utilization Law, the demand per transaction at the line handler is:

$$\text{Demand per transaction} = \frac{\text{utilization}}{\text{transaction rate}}$$

$$= \frac{83.219\%}{17.196} = 0.048 \text{ sec/transaction}$$

Table 2.
TCP data.

Tran Type	Line Handler	TCP	Server	Disk Process	TOTAL
TRAN1	0.008	0.048			
TRAN2	0.008	0.048			
TRAN3	0.008	0.048			
TRAN4	0.008	0.048			

All data is stated in seconds

TRAN1, TRAN2, TRAN3, and TRAN4 Servers

Each one of these servers is responsible for a single transaction type. To determine the server demand for each transaction type, divide the PROCESS CPU UTIL for the server GROUP in the PROCESS REPORT by the RECEIVE RATE for the server GROUP.

TRAN1 $\frac{3.627\%}{1.141} = 0.032$ seconds/transaction

TRAN2 $\frac{1.153\%}{1.883} = 0.006$ seconds/transaction

TRAN3 $\frac{14.233\%}{6.434} = 0.022$ seconds/transaction

TRAN4 $\frac{18.387\%}{7.738} = 0.024$ seconds/transaction

The third column of the table is now complete.

Disk Processes

The disk processes service logical I/O requests. The transaction flow diagram in Step 1 indicates that the TRAN1, TRAN2, TRAN3, and TRAN4 servers make requests to the disk process. Other processes (MEASURE) are making requests also.

By determining the utilization per logical I/O and the number of logical I/Os per transaction, the disk process utilization per transaction can be determined.

The LOGICAL I/O RATE for the system is found by adding up the individual LOGICAL I/O RATEs for each disk in the DISK I/O Report (page 39).

$$20.978 + 21.833 + 19.088 + 21.413 = 83.312$$

The combined utilizations of the disk processes is found on the PROCESS REPORT in the PROCESS CPU UTIL field for the group ^DISK-PROCS. It is 87.091%.

Tran Type	Line Handler	TCP	Server	Disk Process	TOTAL
TRAN1	0.008	0.048	0.032		
TRAN2	0.008	0.048	0.006		
TRAN3	0.008	0.048	0.022		
TRAN4	0.008	0.048	0.024		

All data is stated in seconds

Table 3.
Server data.

To determine the demand at the disk process for each logical I/O, divide the Disk Processes' utilization by LOGICAL I/O RATE for the system.

$$\frac{87.091\%}{83.312} = 0.010 \text{ seconds/transaction}$$

Logical I/Os per Transaction
Looking at the transaction flow diagram, the only "group" receiving messages that are sent out of the transaction processes is the disk process group. This makes it simple to determine the number of logical I/Os per transaction. The receive rate (from the process report) is the number of transactions processed per second (transaction rate) and the send rate (also from the process report) is the number of messages sent from the transaction process to the disk process. Divide the transaction rate of the process into the rate of messages sent to the disk process to obtain the logical I/O's per transaction.

Using the RECEIVE RATE and SEND RATE values found in the PROCESS Report for the transaction groups, the Logical I/Os per transaction are calculated as follows.

$$\text{Logical I/Os per TRAN1 transaction} = \frac{\text{SEND RATE}}{\text{RECEIVE RATE}} = \frac{6.935}{1.141} = 6.08$$

$$\text{Logical I/Os per TRAN2 transaction} = \frac{\text{SEND RATE}}{\text{RECEIVE RATE}} = \frac{1.883}{1.883} = 1.00$$

$$\text{Logical I/Os per TRAN3 transaction} = \frac{\text{SEND RATE}}{\text{RECEIVE RATE}} = \frac{19.600}{6.434} = 3.05$$

$$\text{Logical I/Os per TRAN4 transaction} = \frac{\text{SEND RATE}}{\text{RECEIVE RATE}} = \frac{54.144}{7.738} = 7.00$$

Disk Process demand
The total disk process demand can now be calculated for each transaction type by multiplying the number of logical I/Os per transaction by the demand for a single logical I/O found previously.

Disk Process demand for TRAN1
= 6.08 x 0.010 sec = 0.061 sec/transaction

Disk Process demand for TRAN2
= 1.00 x 0.010 sec = 0.010 sec/transaction

Disk Process demand for TRAN3
= 3.05 x 0.010 sec = 0.031 sec/transaction

Disk Process demand for TRAN4
= 7.00 x 0.010 sec = 0.070 sec/transaction

Column four, Disk Process, is now filled in.

Tran Type	Line Handler	TCP	Server	Disk Process	TOTAL
TRAN1	0.008	0.048	0.032	0.061	▨
TRAN2	0.008	0.048	0.006	0.010	▨
TRAN3	0.008	0.048	0.022	0.031	▨
TRAN4	0.008	0.048	0.024	0.070	▨

All data is stated in seconds

Table 4.
Disk Process data.

Total Consumption per Transaction

By adding up the demands for each process a transaction executes, the total consumption (demand) by a particular transaction type can be determined. The table is now completed by filling in the TOTAL column. (See Table 5.)

Other Work

There are two other workloads in the system that need to be taken into consideration: interrupt handling and other work (i.e. MEASURE, FUP).

In this system, the only other work going on is MEASURE processing. The MEASURE subsystem uses both the CPU and the Disk. In most environments, details of all the other work being done in the system will not be known. The amount of demand these other processes create on the system can be identified. It is the difference between the PROCESS work that has been accounted for in the GROUPs set up for the workloads in the model and the category ^ALL-PROCESSES which contains everything that was running in the system. Add up all the GROUP's PROCESS CPU UTIL and subtract it from the PROCESS CPU UTIL for ^ALL-PROCESSES to get the value for the processing demand for all other work.

222.844 - (87.091 + 14.200 + 83.219 + 3.627 + 1.153 + 14.233 + 18.387)

= 0.934% or 0.009 seconds/second

The model must also include the disk process activity for all logical I/Os in the system other than those accounted for by transactions. The number can be determined by subtracting the sum of the logical I/Os for all the transactions (remember, its the SEND RATE from the Process Report-avg (page 40) for the TRAN1, TRAN2, TRAN3, and TRAN4 servers) from the system logical I/O rate.

Table 5.
Completed data table with totals.

Tran Type	Line Handler	TCP	Server	Disk Process	TOTAL
TRAN1	0.008	0.048	0.032	0.061	**0.149**
TRAN2	0.008	0.048	0.006	0.010	**0.072**
TRAN3	0.008	0.048	0.022	0.031	**0.109**
TRAN4	0.008	0.048	0.024	0.070	**0.150**

All data is stated in seconds

Other Logical I/Os

```
           TRAN1 TRAN2  TRAN3   TRAN4
= 83.312 - (6.935 + 1.883 + 19.60 + 54.144)
= 0.75
```

The CPU demand for these logical I/Os is found, as before, by multiplying this rate by the demand per logical I/O.

```
0.75 x .010 =  0.008 seconds/second
               disk process demand
```

The total CPU demand for other work is:

```
0.008 + 0.009 =  0.017 seconds/second
```

Interrupt processing is the last consumer of CPU seconds that needs to be accounted for. Interrupt processing is the difference between total CPU utilization and total process utilization.

Total CPU utilization is calculated by adding up the average utilization for each processor (field AVERAGE UTIL on the CPU report, page 38).

```
65.728 + 66.357 + 67.852 + 65.661 = 265.60%
```

Total process utilization is found in the PROCESS CPU UTIL field of the PROCESS report (page 40) for the group ^ALL-PROCESSES. This is the utilization for any process executing any object file on the system.

```
Total process utilization = 222.84%
```

Total interrupt utilization is the difference between total CPU utilization and total process utilization. This interrupt utilization value should be close to the INTERRUPT UTIL field available from SURVEYOR. (This value is not shown on the CPU Report on page 38, but the value is 43.019%.). Rounding error accounts for the difference between the two interrupt utilization values.

```
265.60 - 222.84 =  42.76% or .4276 seconds/second
(as compared to 43.019% interrupt utilization value )
```

To check whether all the CPU utilization has been accounted for, multiply the transaction rates for each of the transaction types by their respective demands to get their total consumption. Add these figures to the consumption for other work and interrupt handling. Compare this result to the total CPU utilization reported by SURVEYOR (265.60%).

```
Tran1 Consumption = 1.141  x  0.149 =  0.1700
Tran2 Consumption = 1.883  x  0.072 =  0.1356
Tran3 Consumption = 6.434  x  0.109 =  0.7013
Tran4 Consumption = 7.738  x  0.150 =  1.1607
                              OTHER =  0.0170
                          INTERRUPT =  0.4276
                        Total=  2.6122 or 261.22 %
```

The difference is due to rounding errors.

Disk Consumption Modeling

Disk consumption modeling can be quite complex when all the different activities on different file structures are considered. In the simplest form of DISK modeling, all READS and WRITES to disk are considered to be processed equally. This allows for an association to be made between the logical I/Os occurring and the physical activity occurring on behalf of those logical I/Os.

Just as in the case of the CPUs, this looks upon all the disk drives as being one server (primaries and mirrors).

The modeling approach is similar to the approach used to determine the disk process demand per transaction. First a physical disk demand per logical I/O is found by taking the total physical disk utilization reported in SURVEYOR dividing it by the number of logical I/Os processed. For each transaction type, the physical disk demand is calculated by multiplying the number of logical I/Os per transaction by the physical disk demand for a logical I/O.

The total disk utilization is found by adding up the PRIMARY UTIL and MIRROR UTIL fields from the DISC I/O Report. Although SURVEYOR has a field DISC UTIL, the value in this field is not the sum of the PRIMARY UTIL and the MIRROR UTIL and should not be used. (See the *SURVEYOR Reference Manual* for the definition of DISC UTIL.)

Disk Util =

30.915 + 28.798 + 13.979 + 32.053 + 37.525 + 33.955 + 16.918 + 37.782

= 231.925 %

The Logical I/O Rate was 83.312, therefore, the physical disk demand per logical I/O is:

$$\frac{\text{Total Physical Disk Utilization}}{\text{Logical I/O Rate}}$$

$$= \frac{231.925\ \%}{83.312} = 0.0278 \text{ seconds per logical I/O}$$

To find the physical disk demand per transaction, multiply the number of logical I/Os per transaction by the demand per logical I/O.

Physical Disk Demand for TRAN1 = 6.08 x 0.0278 secs.
= 0.0169 secs/transactions

Physical Disk Demand for TRAN2 = 1.00 x 0.0278 secs.
= 0.0278 secs/transactions

Physical Disk Demand for TRAN3 = 3.05 x 0.0278 secs.
= 0.0848 secs/transactions

Physical Disk Demand for TRAN4 = 7.00 x 0.0278 secs.
= 0.1946 secs/transactions

Physical Disk Demand for Other work = 0.75 x 0.0278 secs.
= 0.0209 secs/group

A simple accuracy check, similar to the CPU check can be made here. Multiply the transaction rates by their respective physical disk demand per transaction and then sum the products. Compare the result to the total physical disk activity (231.925%) to see if all the disk activity has been accounted for.

Disk Consumption for:

Tran1	=	1.141	x	0.1690	=	0.1928
Tran2	=	1.883	x	0.0278	=	0.0523
Tran3	=	6.434	x	0.0848	=	0.5456
Tran4	=	7.738	x	0.1946	=	1.5058
Other					=	0.0209
Total :						**2.3174**
						or 231.74%

It appears that almost all the disk utilization (except rounding errors) has been accounted for.

Forecasting Using the Consumption Model

Remember the question to be answered? It is: What effect will a 20% increase of TRAN4 transactions have on the CPU and disk utilizations?

With the information from the consumption model, this question is easy to answer. The first step is to determine the new transaction rate for TRAN4. Then a "new" CPU and DISK model of the system can be created. The only tricky part is determining the new INTERRUPT workload.

The new transaction rates will be:

Tran1: 1.141 transactions per second
Tran2: 1.883 transactions per second
Tran3: 6.434 transactions per second
Tran4: 7.738 x 1.2 = 9.286 transactions per second

The new CPU consumption for each transaction type will be:

Tran1:	1.141	x	0.149	=	0.1700 seconds/second
Tran2:	1.883	x	0.072	=	0.1356 seconds/second
Tran3:	6.434	x	0.109	=	0.7013 seconds/second
Tran4:	9.286	x	0.150	=	1.3929 seconds/second

The activity of OTHER work remains constant at 0.017 seconds/second.

The INTERRUPT handling grows proportionally as the workload increases, i.e. if the INTERRUPT handling added 15% more processing to the system than the processes being run by the users, then when the load increases the interrupt handling will be 15% additional to the total processing for the new user processing.

To estimate the new interrupt handling, first calculate the percentage of current user processing activity that INTERRUPT handling is.

$$\frac{\text{INTERRUPT BUSY}}{\text{CPU-UTIL for ALL\^{}PROCESSES}} = \frac{42.76}{222.84}$$

$= 0.1919$ or 19.19%

If all the GROUP's consumptions (transactions and OTHER) are added together and an additional 19.19% is added for interrupt processing, the result is an estimate for the expected total CPU utilization with the increased transaction rate for TRAN4.

(0.1700 + 0.1356 +0 .7013 + 1.3929 + 0.017)
x 0.1919 = 2.8806 or 288.06%

The current system has four CPUs, so the new average utilization will be (288.06/4) = 72.02% versus the average of 66.40% previously.

To determine the new physical disk demand required for the new transaction rates, multiply the new transaction rates by their respective physical disk demand and add them together.

Tran1:	1.141	x	0.1690	=	0.1928 seconds/second
Tran2:	1.883	x	0.0278	=	0.0523 seconds/second
Tran3:	6.434	x	0.0848	=	0.5456 seconds/second
Tran4:	9.286	x	0.1946	=	1.8070 seconds/second
OTHER work remains constant at:					0.0209 seconds/second
Total:					2.6186 seconds/second

The total physical disk demand is 2.6186 seconds/second or 261.86%. Since there are eight spindles (four groups of a primary and mirror disk), the new expected average disk utilization would be:

$$\frac{261.86}{8} = 32.73\% \text{ new expected utilization}$$

vs.

$$\frac{231.925}{8} = 28.99\% \text{ for the current system}$$

Summary

Without taking into account the effect on response time that the increased workload would create, each CPU in the current system will be about 5.6% busier and each disk spindle will be about 3.75% busier.

This model assumes that everything else in the system such as the cache hit ratio remains the same. There is also the assumption that no new bottlenecks will appear given the increased workload. Are these assumptions realistic? For low growth estimates the assumptions are more valid than for large growth estimates. That is why it is important to constantly monitor the performance of the system.

During the time period when the system is growing, the model should be validated and if necessary, corrected. The most important concept to remember in computer model building is that the the environment being modeled is always changing and the model needs to change with it as well. This is true whether the model being used is the linear regression technique used in the SURVEYOR FORECAST command or its a consumption model built from the SURVEYOR output.

The first step in any modeling exercise is to get a good insight into the system and the data that is to be modeled. This requires months of historical data that can be analyzed. Only after analyzing the information can one pick a "good" modeling approach. SURVEYOR has great functions for manipulating the data in order to view it in many different ways. It also provides an easy interface to many software packages on a PC that can graph and analyze the information to a greater extent. These facilities will help in determining the "shape" of the data which, in turn, dictates the modeling approach to take.

It cannot be stressed enough how important it is to look at the historical information and get a feel for the trends in that data. Then, the determination of what to forecast can be made. Whether to forecast for the peaks or the averages is a business decision. If forecasting peak loads is desired then an aggregate must be created which will capture the peak values of interest (be it daily peak or weekly peak). The peak can be found by using the MAX statistic in an aggregate definition.

Conclusion

SURVEYOR is an important tool for use in performance management of Tandem systems. This document described the major areas of performance management where SURVEYOR plays a key role.

SURVEYOR provides functions for performance data:

- collection,
- reduction,
- summarization,
- reporting,
- and management.

SURVEYOR can be the foundation tool for performance management modeling needs. A linear regression model is provided in SURVEYOR. A SURVEYOR PDB can provide the input data for other performance and capacity planning models.

The combination of SURVEYOR with the proper staff and methodology, will help performance management organizations insure that adequate computing services are provided to users.

Distributed by
TANDEM
Corporate Information Center
10400 N. Tantau Ave., LOC 248-07
Cupertino, CA 95014-0708