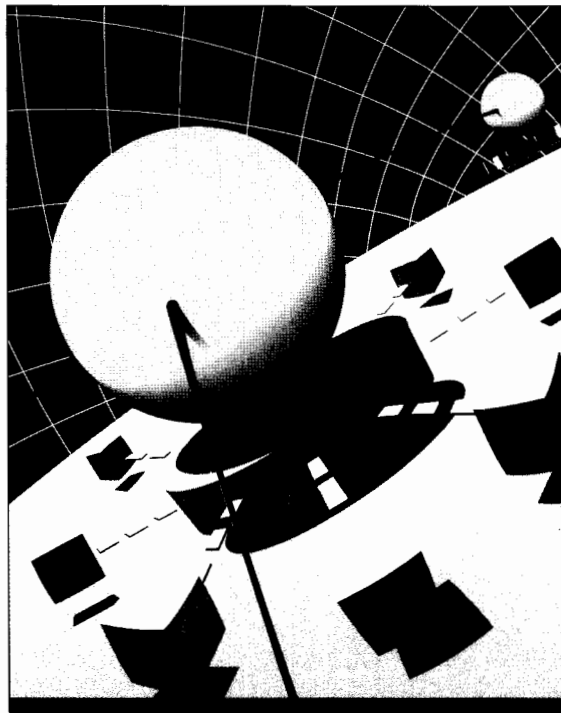


T A N D E M

SYSTEMS REVIEW

VOLUME 9, NUMBER 3

SUMMER 1993



Graphical User Interfaces

Expand Performance

Application Profile

Technical Information and Education

Product Update

T A N D E M
SYSTEMS REVIEW

VOLUME 9, NUMBER 3

SUMMER 1993





Editor's Note

This issue of the *Tandem Systems Review* presents two aspects of client/server application development and implementation. The first article in the issue, "Designing and Implementing a Graphical User Interface," discusses the stages involved in developing a client/server user interface and emphasizes the importance of iterative testing, reevaluating, and modifying the GUI design. The article, which examines the GUI design stage in depth, provides design guidelines to help developers create simple and consistent user interfaces.

The second article presents a practical implementation of client/server computing. "Application Profile: Storing Macintosh Graphics on the Tandem 5200 Optical Storage Facility" describes how Val-Pak Direct Marketing Services, Inc., developed a way to link their Macintosh network to the Tandem OSF. With this client/server implementation, Val-Pak MIS professionals were able to efficiently store large graphics files on the OSF and provide quick access to those files for their Macintosh graphics application.

When upgrading to new Tandem systems, users should consider their networking requirements early in migration planning. The third article in this issue, "Expand High-Performance Solutions," focuses on the performance capabilities of the Expand-over-LAN network access subsystem. The article shows that Expand-over-LAN is a suitable solution for many applications requiring high performance.

At the back of this issue you will find our Reader Survey. So that we are kept informed of your technical information needs, please fill in the questionnaire and return it to us at your convenience.

—AL

EDITOR
Anne Lewis

ASSOCIATE EDITORS
David Gordon, Steven Kahn,
Mark Peters

PRODUCTION MANAGER
Anne Lewis

ILLUSTRATION AND LAYOUT
Donna Caldwell, Christine Kawashima

COVER ART: David Thompson

SUBSCRIPTIONS: Elaine Vaza-Kaczynski

ADVISORY BOARD
Mark Anderton, Jim Collins,
Terrye Kocher, Randy Mattran,
Mike Noonan

Tandem Systems Review is published quarterly by Tandem Computers Incorporated. All correspondence and subscriptions should be addressed to *Tandem Systems Review*, 10400 Ridgeview Court, Loc 208-65, Cupertino, CA 95014.

Subscriptions: \$75.00 per year; single copies are \$20.00. Detailed subscription information is provided on the subscription order form at the end of this book.

Tandem Computers Incorporated assumes no responsibility for errors or omissions that may occur in this publication.

Copyright ©1993 Tandem Computers Incorporated. All rights reserved. No part of this document may be reproduced in any form, including photocopy or translation to another language, without the prior written consent of Tandem Computers Incorporated.

CLX, CLX/R, CLX 2000, Cyclone, Cyclone/R, FOX, Expand, Guardian, Guardian 90, Multilan, NDX, NonStop, PSX, SNAX, Tandem, the Tandem logo, TMF, TXP, and VLX are trademarks and service marks of Tandem Computers Incorporated, protected through use and/or registration in the United States and many foreign countries.

NET/MASTER is a trademark of Systems Center, Inc.

UNIX is a registered trademark of UNIX Systems Laboratories, Inc. in the USA and other countries.

All brand names and product names are trademarks or registered trademarks of their respective companies.



CLIENT / SERVER

- 6** Designing and Implementing a Graphical User Interface
Susan Wolfe

- 34** Application Profile: Storing Macintosh Graphics on
the Tandem 5200 Optical Storage Facility
Diane Broyles



FEATURE

- 22** Expand High-Performance Solutions
Donald Smith

DEPARTMENTS



- 2** Product Update
- 42** Technical Information and Education
- 44** Index of Articles

Guardian 90 Based Software

Syshealth Toolkit

April 1993

The Syshealth toolkit for Tandem NonStop systems replaces and enhances some of the functions of the existing diagnostic system, such as event logging, log viewing, event distribution, and problem notification. The toolkit consists of software that continuously monitors system resources and notifies the local operator and, optionally, a designated remote site whenever a problem with a system resource exists.

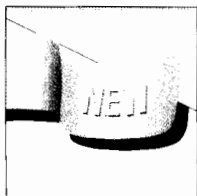
The Syshealth toolkit employs a block-mode type of user interface that allows the operator to manipulate the program through a series of screens, dialog boxes, and menus. It provides a context-sensitive help facility that allows operators to have immediate access to needed information.

Tandem Failure Data System

March 1993

The Tandem Failure Data System (TFDS) is software that provides automatic CPU-failure data collection, diagnosis, and recovery services. TFDS monitors Tandem CPUs in NonStop systems and, in the event of a failure, automatically initiates a CPU dump, analyzes the failure data, and takes action based on the type of defect discovered.

TFDS ensures that sufficient data is collected the first time a problem occurs, uniquely identifies the problem, and then suppresses data collection if the problem recurs. TFDS services can be tailored to meet the needs of the user. Operations personnel are given the ability to control a variety of options, including automatic tape backup and CPU dumping.



Client/Server Computing Products

CLX/R and Cyclone/R Transaction Servers

May 1993

The CLX/R and Cyclone/R Transaction Servers are new hardware-and-software packages for client/server computing. They are based on the CLX/R and Cyclone/R systems respectively.

In addition to the base system hardware, each transaction server is configured with the following: Guardian 90XL operating system, Remote Server Call (host component), Pathway Open Environment Toolkit (POET), Tandem Dynamic Data Exchange (DDE) Gateway, Tandem SQL Server Gateway, Data Access Language (DAL) Server, TandemTalk, COBOL85 run-time library, Tandem LAN Access Method (TLAM) software, and two Ethernet LAN controllers.

Tandem Client/Server Online Transaction Processing Toolkit

March 1993

The Tandem Client/Server Online Transaction Processing (OLTP) Toolkit simplifies interoperability and extends business-critical transaction processing to many desktop platforms in a client/server environment. The products in the Client/Server OLTP Toolkit minimize programming complexity between client and server platforms in an OLTP application, and make Tandem NonStop servers available to users at the desktop.

The Client/Server OLTP Toolkit consists of the Pathway Open Environment Toolkit (POET), Tandem Dynamic Data Exchange (DDE) Gateway, and an enhanced version of Remote Server Call (RSC). POET and DDE Gateway accelerate development of OLTP applications with Microsoft Windows based clients. RSC is an application-programming interface that reliably delivers messages between the server and clients that run Windows.

Tandem Workflow Management

May 1993

Tandem Workflow Management provides an open-architecture solution for departments in which large numbers of related documents must be routed and processed. It is a complete client/server system for reengineering business processes to take full advantage of up-to-date information processing technology. Tandem Workflow Management allows users to scan documents into digital images, then route them over a local area network to predefined destinations for processing. Several users can update a document simultaneously, and images can be automatically routed for signature approval. Through imaging, the entire history of a transaction can be made instantly viewable.

Tandem Workflow Management incorporates Tandem PSX and NDX platforms, ViewStar imaging software, Novell NetWare, and other components selected from leading vendors. All products have been certified by Tandem and ViewStar to be compatible with each other.

Communications and Networking Products

MultiLAN Access Device and MultiLAN Access Device Kit

April 1993

Tandem now offers new versions of the MultiLAN Access Device (MLAD) and the MLAD Kit. These two products provide the same functionality as the ones they replace, but reflect the use of newer technology. The new MLAD Kit includes a 16-bit Ethernet LAN adapter, cables, and connectors for attachment to the Ethernet port on a Tandem Guardian host system. The software includes both host and user interface code, and has been updated to include the latest interface drivers. The MLAD Kit now provides the software on both 3.5-inch and 5.25-inch diskettes.

The new MLAD product consists of the new MLAD Kit preinstalled in a Tandem PSX EP386SX/33 workstation. The workstation includes 2 megabytes of memory, a 3.5-inch diskette drive, and a 40-megabyte IDE hard drive.

NetWare Boot PROMs

March 1993

Tandem now offers two NetWare boot PROMs, for 8-bit and 16-bit bus configurations, respectively. These PROMs can be added to existing UUTP8 and UUTP16A LAN adapter cards to allow booting from a NetWare server. This eliminates the need for a floppy or hard drive in the local workstation.

Storage Products

4510 Disk Subsystem

May 1993

The 4510 disk subsystem is a high-capacity, cost-effective disk storage device designed for high-performance online transaction processing (OLTP) and large database applications. With 2 gigabytes of formatted data in each disk drive, the 4510 disk subsystem delivers a price-per-megabyte improvement of up to 38 percent over the existing 4500 disk subsystems.

The 4510 disk subsystem offers 72 gigabytes of formatted capacity in a footprint of 6.3 square feet (0.6 square meters). It can be connected to any NonStop Cyclone, Cyclone/R, or CLX system through the Tandem 3128 disk controller. Using advanced fiber-optic cabling, the 3128 controller makes it possible for disk subsystems to be located up to 6580 feet (2 kilometers) away from the host. The 4510 is designed to operate in a normal office environment and can be serviced online without interrupting the operation of the rest of the system.

4250 Disk Drive

May 1993

The 4250 disk drive is a new high-capacity, high-performance internal disk storage device for the NonStop Cyclone/R and CLX systems. The 4250 has a formatted capacity of 2 gigabytes and is contained in a standard customer-replaceable unit (CRU).

The 4250 disk drive has the lowest price per megabyte and highest capacity of any of the internal disk storage devices for the NonStop Cyclone/R and CLX systems. Using the 4250 disk storage device, a fully configured 16-processor Cyclone/R system (with expansion cabinets and a total of 96 drives) can provide up to 192 gigabytes of formatted internal storage.

StorageTek 4400 Automated Cartridge System

May 1993

Beginning with the D10 release of the Guardian 90 operating system, Tandem is providing support for the StorageTek (STK) 4400 Automated Cartridge System (ACS) on the following NonStop systems: Cyclone, Cyclone/R, CLX 2000, and CLX 800. The 4400 ACS offers automated handling of up to 19.2 terabytes of tape data. With the Tandem 3217-1 Data Path Adapter (DPA) and SF02 Tandem Client Software Component (TCSC), it provides unattended access to this data from NonStop systems 7 days a week, 24 hours a day. Users can thus fully automate Tandem's Transaction Monitoring Facility (TMF) software, backup and restore programs, and any other applications that use tape as input or output.

The 3217-1 DPA hardware transfers data between a NonStop system and the tape controller of the STK 4400 ACS. The SF02 TCSC software provides the control path to completely automate tape handling for the NonStop system. Multiple host systems can be attached to a single 4400 ACS. A DPA unit can be mounted in a standard 19-inch rack, or as many as four DPA units can be stacked in a freestanding configuration. The fiber-optic cable between the NonStop system and the DPA can be up to 1.24 miles (2 kilometers) long, thus making it possible to centralize the DPAs for an entire campus of NonStop systems.

Workstation and Terminal Products

PSX SP-Series Workstations

April 1993

The PSX SP-series workstations are high-performance, expandable desktop computers using the EISA I/O bus standard. The SP-series workstations feature the most powerful Intel processors available today. In addition, this series provides support for an Intel Pentium processor card and second-level cache up to 512 kilobytes. The SP-series workstations incorporate 5 EISA-bus expansion slots, 5 drive bays, and a connector for adding external I/O devices. In addition, users can add up to 128 megabytes of RAM on the system board. Also available as options are the 170-megabyte and 340-megabyte IDE hard drives.

Standard features of the PSX SP graphics subsystem include 1 megabyte of video memory, a local-bus interface, 64-bit video memory access, Bit Block Transfer (BitBLT), and extended resolution modes supporting up to 1280 x 1024 with 256 colors. The PSX SP-series also incorporates a number of new features for security, including user and system administrator software passwords, floppy drive write protect, I/O port locking, boot drive select, asset management, chassis lock, and AST Walk-n-Lock.

Memory Option for PSX EP-Series Workstations

February 1993

Tandem has added a 4-megabyte memory option for its PSX EP-series workstations. With the addition of the 4-megabyte option, two new memory configurations, 6-megabyte and 12-megabyte, are now available for EP systems.

Memory Expansion Card for PSX Workstations

February 1993

A new memory expansion card is now available for all Tandem PSX systems employing CUPID-32 architecture. The new card is expandable to provide up to 32-megabytes of memory. This one expansion card services all CUPID-based systems and replaces the two previous memory expansion card offerings.

MS-DOS 6.0

April 1993

Tandem is now including MS-DOS 6.0 in all new shipments of PSX systems. In addition, Tandem is offering DOS 6.0 upgrades to its existing users. In the U.S., DOS 6.0 upgrades are available in both 3.5-inch and 5.25-inch media. In Europe, USASCII, French, and German versions are available in 3.5-inch media only.

Product Programs

Tandem Client/Server Ready Program

April 1993

To make the move to client/server computing easier for users, Tandem is now offering the Client/Server Ready Program. The program makes available three preconfigured personal computer (PC) packages that include industry-standard PCs and software to update or create new applications for client/server computing. Each of the three PC packages, high-end, mid-range, and low-end, includes the following standard components: Tandem Terminal Emulators for Windows software; Microsoft Windows, version 3.1; DOS, version 6.0; a mouse and mouse pad; a 16-function-key keyboard; and a current-loop adapter.

As part of the program, Tandem is also offering several educational courses and specialized services to help users learn more about client/server computing and implement it in their environment. A trade-in credit for old terminals and PCs, as well as credit to apply toward the services and courses, completes this special offer.

Designing and Implementing a Graphical User Interface

A graphical user interface (GUI) that provides windows, menus, and icons and uses a mouse to select and manipulate objects is often touted as the only way to go in making complex applications easy to use. A GUI of this type is supposed to decrease the amount of time and expense needed for training users, reduce user errors, and increase productivity. These benefits, coupled with the proliferation of workstations and client/server computing, have encouraged many development teams to consider implementing GUIs for applications that run on Tandem™ computer systems.

However, simply providing a GUI front end does not of itself make an application easy to use. A poorly designed GUI can confuse or mislead users. Instead of simplifying an application, it can intimidate users. An effective GUI requires a careful process of design and implementation.

This article provides a brief overview of the stages involved in developing a GUI and then devotes separate sections to each of the primary stages:

- Analysis of users and tasks.
- Design of the GUI and development of a prototype.
- Final implementation.

Most of the discussion is devoted to the section on GUI design, which includes a list of guidelines for interface designers.

The article does not assume more than casual familiarity with the use of a GUI on a Macintosh, PC, or other system. It is intended both for the reader with a general interest in GUIs and for developers who may be working on applications with a graphical user interface. It should be noted that many of the guidelines and procedures described in the article apply equally well to both graphical and character-based interfaces.

Stages in the Development of a GUI

Developing a GUI is a three-stage process. The stages and their subcomponents are outlined in the sidebar.

The first stage in developing a GUI is to analyze the people who will use the GUI and the tasks they need to accomplish. In the second stage, the GUI itself is designed and prototyped. Design of the interface should usually be guided by specialists who work closely with the software developers responsible for the design of the application. First, the interface designer maps tasks and subtasks to windows. Following the mapping of tasks to windows, the designer lays out the contents of individual windows according to principles of design (see "Design Guidelines," later in this article) and aesthetics. Next, prototype versions of the windows are created and reviewed for usability and effectiveness. Based on the review, the design may be modified and reevaluated. This is usually a highly iterative process with repeated cycles of design, test, evaluate, and redesign. The initial design almost always needs to be modified to achieve optimal usability.

In the final stage, the design is implemented online as part of the overall application. New usability tests are conducted and, if necessary, the implementation is refined.

Each stage in the development of a GUI is important in its own right. Sufficient time for all three stages should be factored into the development schedule from the start in order to ensure an easy-to-use GUI.

Stages in the Development of a Graphical User Interface

I. Analysis of Users and Tasks

A. Analyze the users.

- Who are they?
- What do they do?
- Why do they do it?
- What do they know?
- What additional knowledge do they require?

B. Analyze the tasks.

- What do they accomplish?
- How are they currently done?
- How can they be broken down into subtasks?
- What is the optimal sequence of subtasks and tasks?

II. Interface Design

A. Based on the earlier task analysis, assign tasks and subtasks to windows.

B. Lay out the contents of each window according to design guidelines (listed later in this article).

C. Devise representations of the windows (prototypes) that allow the interface to be reviewed for:

- Assignment of tasks to windows.
- Placement and design of features within windows.
- Sequencing and flow from window to window.

D. Conduct usability reviews.

E. If required by usability reviews, return to Step IIA or IIB and modify the design.

III. Implementation

A. Within the constraints of the technology available, implement the interface.

B. Conduct usability tests.

C. If necessitated by the test findings, return to Step IIIA and refine the implementation.

Analysis of Users and Tasks

The first stage in the development of a GUI is an analysis of who will be using the application and the tasks they need to accomplish. The following sections describe the analysis of users and tasks separately, although in practice they are intertwined.

Figure 1

Experience with other applications	More	Assumes new application will work like others	May confuse this application with others	Wants multiple ways of achieving goals
	Less	Needs flow to be very obvious	May need to relearn application each time it is used	Wants shortcuts to facilitate completing the task
		Novice user	Intermittent user	Frequent user
		Experience with this application		

Figure 1.

Characteristics of users with different levels of computer experience.

Analyzing Users

Factors such as educational background, occupation, experience with computers, familiarity with the current application or similar applications, and knowledge of the subject matter can significantly influence the way users approach an application. As a result, in developing a GUI it is important to know the characteristics of the different user groups that may be carrying out tasks within an application. This information can be obtained by developing profiles of user groups through such means as questionnaires, surveys, focus groups, and direct observation of the work place.

Effect of Previous Computer Experience on User Needs and Expectations. In working with a given application, a user's needs and expectations are influenced by previous experience with similar applications and with computers in general. Figure 1 shows some of the common effects of different levels of user experience.

One of the implications of Figure 1 is that if a given task is likely to be carried out by users with different experience levels, the GUI for the task should provide a combination of features. For example, if there are multiple ways to navigate through a task, novice or intermittent users will require that at least one path be explicit and obvious. On the other hand, frequent users will want shortcuts and multiple ways of carrying out a task.

Another implication of Figure 1 is that different design features may need to be emphasized according to the type of user that carries out a task. For example, if users only carry out a particular task intermittently, the design considerations may be different than for a frequently executed task only carried out by highly experienced personnel.

Characteristics Common to Users in General.

Certain generalizations are likely to apply to all user groups:

- Users want to be able to apply what they already know to what they are learning.
- Users do not necessarily follow directions or read manuals.
- Users tend to learn and use only those parts of an application that are necessary for completing their specific tasks.

Users do not want to learn everything from scratch. They want to be able to extrapolate from what they already know to what they are learning. For example, they want to be able to transfer knowledge of other applications to the current application. This is one of the important reasons for maintaining interface standards that provide consistency across different applications.

Rather than looking at manuals or instructions, users often try to anticipate what they should do to carry out an operation or sequence of steps. This makes it important to understand the user's perception of task flow and design the interface so that it effectively leads the user through a task. This also means that users should be able to undo actions and retrace the steps that led them to a dead end or unwanted action.

Users generally do not like to deal with material extraneous to their tasks, and in many cases, the tasks performed by a user require only a small subset of the functionality contained in an application. As a result, the interface should be designed, as far as possible, to provide distinct paths for different users or user groups. Each path should put the most frequent or important operations for its users in the foreground. Less frequent or secondary tasks should remain accessible, but further in the background.

Analyzing Tasks

Tasks define the functionality of an application. Each task is defined by a starting point, an action, and a goal. Subtasks are tasks contained within a larger task. Frequently, tasks and subtasks must be carried out in a fixed order. In conducting a task analysis, it is important to directly observe the activities users currently carry out and note the way and sequence in which they perform them.

As an example, suppose the starting point of an electronic mail application is logging on. This mandates the need for a logon function immediately upon invoking the application. One goal of the application is to allow users to send messages. This involves several actions, and hence several subtasks: creating the contents of a message, addressing it, and

sending it. In terms of sending the message, the order of subtasks is important. Users should not be allowed to send a message before it is created and addressed. In other regards, however, the order of subtasks is irrelevant. It does not matter whether the contents are created first or the message is addressed first, as long as both are completed before the message is sent.

The task analysis should also make clear what information or activities are necessary for carrying out a task and what is extraneous or optional. For example, in an electronic mail application, users typically want the system to send their mail messages immediately upon completion, but they may also want the option of specifying a delayed send time for a message. Once a message is sent, users may have no interest in the precise route it takes to get to the recipient or the length of time it takes to cross the network.

Using such information, the designer can make immediate delivery of mail messages the default procedure, initiated with a single mouse click or keystroke, and provide delayed messaging as an option invoked through a push button and dialog box. Information about message routing or elapsed delivery time would not be provided.

Figure 2

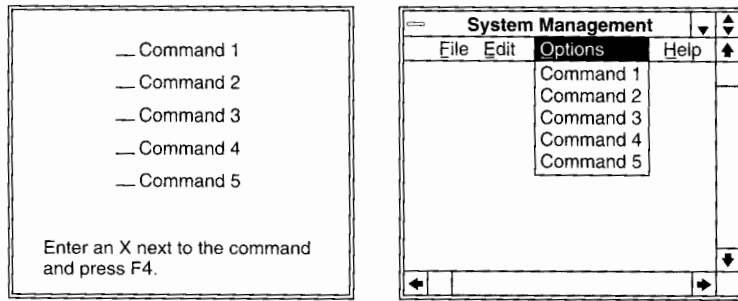


Figure 2.

Executing the same commands from a character-based screen and a GUI menu.

The Design Phase

Once users and tasks have been analyzed, formal GUI design can begin. In truth, informal GUI design is usually quite far advanced by this time, since the designer typically hypothesizes possible interface features and mechanisms as a part of analyzing users and tasks. However, when formal design begins, a more rigorous evaluation of design options can begin. Tasks can be mapped to windows, the layout of GUI features within windows can be undertaken, and navigation from one window to another can be defined. Finally, prototypes of the windows can be created, tested, and refined.

Mapping Tasks to Windows

Windows are the fundamental building blocks of GUIs. Users perform their tasks by working with a set of windows on the screen. After launching an application, users communicate with it by viewing, typing, manipulating, and navigating through the contents of windows. A given window may allow a user to carry out some tasks or, more likely, some subtasks, and then lead the user to another window which contains the next tasks in a sequence. Tasks must be organized and mapped onto windows in such a way that users can do their jobs quickly and easily.

It is important to emphasize that the organization of windows in a GUI should not be a one-to-one mapping of operations from character-based screens in an earlier application. If it is, the inherent limitations of a character-based interface will simply be replicated in the GUI. The organization of GUI windows should reflect a reasoned grouping of tasks based on a prior analysis of users and their activities.

For example, suppose a screen in a character-based interface lists five commands. To invoke one of the commands, the user must enter an X next to the command name and press a function key. As illustrated in Figure 2, replacing the original screen with an empty GUI window that simply lists the same commands under a menu is no improvement and may even be a disservice.

In Figure 2, the GUI menu listing commands is shown in an open position. A user would first see a completely empty window, with no direct indication of what to do or where to look for commands 1 through 5. A more effective GUI might use the window surface to display data or entities manipulated by the application. The user could click on one of these objects to select it and then execute a command against the object by choosing it from the menu.

Using Storyboards to Illustrate the Mapping of Tasks to Windows. Once a designer has planned the mapping of tasks to windows, storyboards provide a simple way of representing windows so that the mapping can be evaluated well before coding begins. In storyboarding (which started in the motion picture industry as a means of reviewing the organization and flow of a film), static pictures are used to represent a sequence of scenes or activities. Using this method, a designer can create a sequence of storyboards to illustrate the major functions of an application and the way tasks flow from one window to another. For example, in an electronic mail application, one set of storyboards might illustrate the series of windows used in creating a message, another might show the sequence for sending a message, and a third might represent the use of windows for selecting and reading an incoming message.

The intended users of an application can review storyboard sequences to ensure that the assignment of functionality to windows and the order of presentation are satisfactory. This can reveal any major flaws in the functionality of an application long before prototyping begins or developers begin writing code. It provides a useful, and economically beneficial, sanity check on the general model of the application.

Planning Online Help. Online help is an integral part of a GUI application and should be included early in the design stage. Planning for online help includes a consideration of the types of help to be provided and the mechanisms for presenting them. To be most effective, help should be context-sensitive and to the point. Research has found that users do not avail themselves of online help if they are forced to read or search through extended amounts of text.

Help can be provided at several levels. At the most general level, it can offer information about the application as a whole and its main features and uses. At the window level, it can describe the type of task carried out through the window. Within a window, help can be associated with the individual objects displayed, such as icons, push buttons, graphs, text-entry fields, and any other visual or text items. In some cases, help may display a list of topics and allow the user to select those of interest.

There are a number of mechanisms for making help available. Three common mechanisms are help menus, designated keyboard keys, and push buttons. A help menu allows the user to select different types of help or help topics. A help key, which may be any function key or control-key combination designated by the application, can provide context-sensitive help: the user selects an item in the window and presses the help key to get information about it. A push button can provide information specific to the window or dialog box that contains the button.

Figure 3

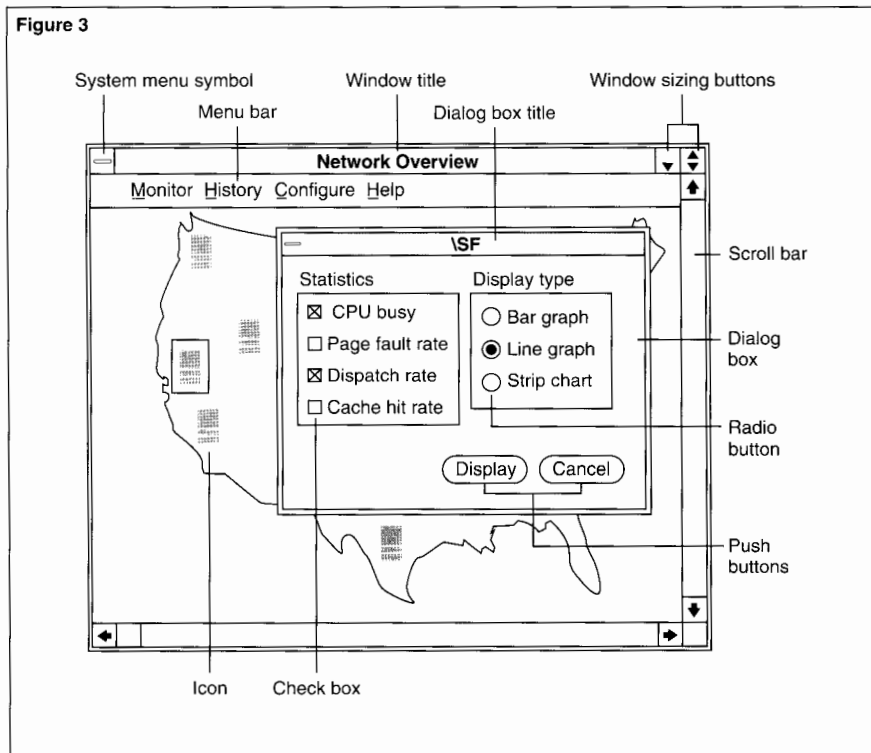


Figure 3.

Window showing use of terms for GUI features and objects.

In making online help available, it is important that the development schedule provide adequate time for technical writers to document the help, integrate it into the system, test its usability, and revise it accordingly. Further, if the schedule allows writers to carry out such activities at the same time that software developers design, test, and revise code, the ultimate result will be better online help and a more usable application.

Laying Out Window Content

Once tasks have been assigned to windows, the designer can begin laying out the contents of individual windows. Figure 3 illustrates a window containing common GUI objects. For each window belonging to an application, the designer must make decisions about (1) the type of window used, for example, resizable with an action bar or fixed size with no action bar (i.e., a dialog box); (2) the placement of items within the window, including the use of text, icons, graphical representations of objects, and the use of color; and (3) the ways in which commands are executed, such as direct manipulation of an icon in the window, selection from a menu, use of a push button, or keystrokes on the keyboard.

At this point, decisions should also be made about the interface features of online help and the use of graphics for displaying data and messages to the user. A GUI interface that merely consists of text in windows and dialog boxes does not take advantage of the power of a graphical workstation to enhance the users' understanding of information (see, for example, Guideline 11 under "Design Guidelines").

Appropriate Use of Graphics. In laying out window content, the designer must consider the specific use of graphical representations for invoking commands and displaying data. In a GUI, icons are typically used to represent the objects and entities on which an application operates. For example, in a file-management application, one expects to find icons for files and directories, and possibly disk volumes. In a network-management application, one would expect to have icons for representing the individual nodes in a network.

In some cases, direct manipulation of a graphical object is an effective way to execute a command. For example, to print a file, one might have the user drag an icon representing a file to an icon for a printer. In an electronic mail application, the user might send a message by dragging an icon for a letter to the icon for a mailbox.

In many cases, a graphical representation can make it easier to analyze quantitative data or to indicate the relative status of an event. This is illustrated in Figures 4 and 5.

Figure 4 shows the same data presented in a table and in a graph. If a user needs to compare the activity levels of CPUs, it is much easier to evaluate a graphical display than a strictly tabular presentation of the data. Figure 5 shows an animated status bar that makes the processing status of an event immediately apparent.

In some cases, graphical representations are clearly not appropriate. For example, on a menu, it is better to list commands as text items than as icons. A one-word or two-word text item is recognized as rapidly as an icon and is probably more directly interpretable. This is illustrated by the two Edit menus in Figure 6. As another example, in a word-processing application it would be comically inappropriate to display a full keyboard online and ask the user to enter all text by clicking on characters with the mouse rather than using the real keyboard.

User-Interface Standards. For ease of use, it is of critical importance to adhere to existing user-interface standards in designing a GUI and laying out the contents of windows. GUI standards assure consistency both within and across applications and make it possible for the user to transfer skills from one application to another.

There are several industry-recognized interface standards available to assist the designer, including standards for PC-based Windows and OS/2 applications, Macintosh applications, and applications for several flavors of UNIX. These standards (1) define the preferred use of the window-management system, for example, the types of windows available, their specific

Figure 4

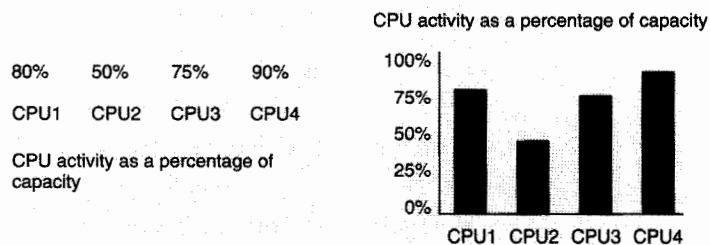


Figure 5

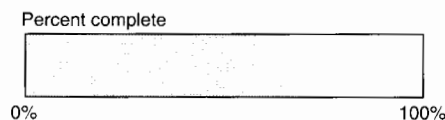


Figure 4.

Comparison of tabular and graphical displays of data.

Figure 5.

Graphic showing status of an operation or event.

Figure 6

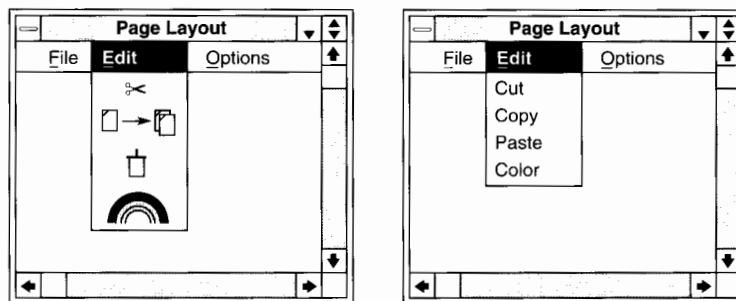


Figure 6.

Comparison of a menu using icons with a menu using text items.

attributes, and the way users interact with them to use menus or other features; (2) identify the elements that can be used in windows, such as radio buttons, check boxes, and push buttons; and (3) describe methods for moving from one window to another.

In addition to current standards, Tandem has developed its own guidelines for graphical user interfaces. These guidelines will be available to users and third-party developers in the forthcoming *Tandem Style Guide for Graphical User Interfaces*. The guide will cover

- Graphical user interfaces, in general, and Tandem GUIs in particular.
- The elements of a good graphical user interface.
- The process of creating a graphical user interface.
- The production of GUIs consistent with the Tandem standard.

Tandem's GUI guidelines support interface standards developed for diverse systems, but provide additional specifications that ensure consistency from one Tandem application to another. For example, with respect to text, the guidelines specify conventions for the use of fonts, terminology, and abbreviations. The guidelines also provide general specifications for the way graphics and text are to be used within a window and the way in which navigation from one window to another should be carried out. Specifications are designed so that they can always be carried out in accord with the GUI standards that apply to a given platform.

Tandem's guidelines for designing a GUI are based on sound human factors principles. When combined with GUI standards for individual platforms, they provide consistency and ease of use and can reduce the cost of user training and support.

Design Guidelines

A GUI design should be simple and consistent from window to window. Its primary goals are ease of use and user productivity. Guidelines for achieving these goals follow.

1. Organize and sequence windows in terms of the user's goals and needs. The organization and sequence of windows should lead users directly through major tasks. Application logic or the structural characteristics of a database should not influence the way in which windows are designed or presented. For example, if a given user task requires data from two tables in a database, only ease of use and the user's goals, not the database, should determine whether the data is displayed in one window or in two or more windows.

2. Be consistent in the way functions are carried out and objects are represented. Make similar functions look and act the same way throughout the application. Conversely, make dissimilar functions look and act differently throughout the application. Use terms, fonts, colors, and graphics consistently throughout the application.

3. Group information in functionally related units or modules. For example, in a word processing application, users should specify the number of copies to print at the same time other attributes of the print request are specified.

4. Use analogy and metaphor to make visual features and text immediately accessible. For example, in an electronic mail application, dragging an icon representing a letter to a graphic representation of a mailbox would provide a visual analogy for sending a message.

5. Make the most important and most frequently used functions the most prominent and easy to use. For example, in many cases an important operation can be activated simply by clicking an icon or labeled push button (as in Figure 3). Another alternative is to place the operation at the top of an appropriately labeled menu.

6. Keep window layout clear and simple.

- Use space judiciously; do not overcrowd windows.
- Label windows, objects, and fields so that their purposes are evident.
- Layer content so that important information or activities are readily apparent, but secondary or optional information is not visible unless explicitly requested by the user.

7. Provide time-saving options and actions.

- Where possible, offer choices that can be selected with a simple click of the mouse or other action, rather than requiring the user to enter full text items from the keyboard.
- If two or more operations are always (or almost always) carried out together, allow the user to carry out both operations with a single action. For example, most word processing applications automatically open a new document when invoked, rather than requiring the user to first start the application and then open a document.
- If the system already has required information, the user should not have to enter it unnecessarily. For example, if the user has already entered a name and address at one point, the system can automatically supply it at other points.

8. Provide feedback to the user through status messages and visual indicators. Indicate progress or the current status of long-running operations. Confirm the completion of meaningful operations. If a process takes longer than 2 seconds, indicate progress (for example, through a moving second hand on a watch). If the process is going to take longer than 20 seconds to complete, allow the user to back out of the action while it is in progress. If the process is likely to take longer than 2 minutes, inform the user that this is the case and require the user to confirm the operation before proceeding.

9. Make allowances for user error. Where possible, allow users to undo actions. Before allowing an unrecoverable, destructive action, use a dialog or message box that requires the user to confirm the operation. Avoid unrecoverable deletions that occur as a side effect of another action.

10. Give users control over the way they accomplish their goals. Where possible, allow users to set their own defaults and to customize procedures. Let users determine the pace at which they carry out operations. Provide redundant controls (for example, both a button and a menu item) and shortcuts (for example, both a menu item and a keystroke).

11. Where appropriate, provide graphs or other visual representations of quantitative data.

As discussed earlier (see “Appropriate Use of Graphics”), in many cases a graphical representation may make it easier to analyze quantitative data or to indicate the relative status of an event.

12. Make online help context-sensitive; make help texts short and to the point. Research indicates that online help is not used unless it quickly and concisely provides the information the user is seeking.

13. Be sensitive to cultural and educational factors. Consider the possibility that members of other cultures may use an application and that users may have very different backgrounds in terms of education and experience. Avoid use of jargon, abbreviations, uncommon technical terms, and highly idiomatic or culture-bound terms. Make sure that the choice of metaphor and analogy is appropriate in expected cross-cultural contexts.

14. Where possible, follow existing GUI standards. Adhering to standards provides consistency across applications and allows users to transfer skills from one application to another. New users can learn an application faster; users in general are less likely to make errors.

Prototyping

Mapping tasks to windows and laying out the contents of windows results in an initial GUI design. The next step is to develop a prototype version of the interface that allows others to review and evaluate the design of individual windows and the way the windows work in sequence.

The Prototyping Process. Designing a GUI is an iterative process in which an initial design is prototyped, evaluation of the prototype leads to changes in the design, and a new prototype is created to evaluate the modified design. The initial prototype versions of an interface are usually static representations of windows that can be used in storyboarding. The representations need not be anything more than paper and pencil drawings. At this stage, the goal is to reveal any major flaws in overall design and to detect obvious problems in the planning of window sequences or the design of individual windows.

Once the interface has been sufficiently tested and refined, static prototypes can be replaced with more realistic software prototypes that have been developed online. Such prototypes are dynamic. They allow users to select and move objects within a window and to move between windows in a way similar to interaction with a functioning application.

A software prototype typically represents only the front end, or client portion, of a client/server application, and uses dummy data to represent interaction with another system. It is usually written in a programming environment specifically selected for ease of prototype development. Often, this is not the environment used for final coding of the GUI on the front end as part of the overall client/server application.

Prototyping Tools and Environments. Many types of tools and environments can be used in prototyping. These include paper and pencil, applications for creating graphics (such as MacDraw), programming languages, and applications specifically designed for building interfaces. Often, a number of different tools will be used in the course of prototyping a GUI.

The choice of prototyping environments does not depend on the platform that the application will ultimately run on. Any GUI can be prototyped on any platform and later implemented on the platform used for application development. Decisions about the prototyping environment to be used at a given point should be based on ease of use and speed.

The programming environment used for final implementation of the GUI can also be used to produce prototypes. However, in this case, any prototype code carried into the final implementation should be carefully examined to ensure that it is of production-level quality.

Testing the GUI for Usability. Prototypes make it possible to carry out usability tests in which typical users perform tasks representative of the application. Among the design attributes evaluated through usability testing are:

- *Effectiveness.* Is the GUI appropriate for the tasks users carry out? Does the GUI lead users through tasks and subtasks in a way that matches their conceptual models of the tasks? Do the operations in the GUI help users carry out their tasks efficiently and productively?

- *Ease of learning.* Is it easy for users to learn how to use the application? Is it easy to install or initiate? Once learned, how much effort is required to relearn use of the application after an extended interval?

- *User acceptance.* What are users' attitudes toward the application? Do the design's aesthetics make users like or dislike using the application? Are colors overused or misused? Are the windows too busy? Do users find they like using the application because they are able to use it successfully? Do they get frustrated and dislike it because they are unsuccessful at using it?

Designing a GUI is an iterative process.

Usability testing can provide qualitative data through direct observation and interviews. It can provide quantitative data through scaled user ratings and counts or measures of such factors as user error rates and the length of time it takes to complete a task. Based on the results of such testing, the designer can modify the GUI and subsequent prototypes in accordance with user perceptions and performance.

Implementation

When prototyping and usability testing have resulted in a stable design, software developers can start implementing the GUI in the final application code. This, like prototyping, is an iterative process. The GUI is given an initial implementation within the client portion of a client/server application and then tested against the server portion. Based on the results of testing, some aspect of the GUI or its implementation may need to be modified and tested again.

The boundary between the implementation stage and prototyping in the design stage is not always clear, particularly when prototypes are coded in the same language as the final implementation. The GUI is subject to modification at both stages. However, prototyping focuses on design of the GUI for a front-end system, often independent of its interaction with software on a back-end system. Implementation of the GUI is concerned with the way the software on a client system interacts with a back-end system.

During the implementation stage, there are two major reasons for making changes in the GUI:

- Testing reveals flaws in the design of the GUI.
- Aspects of performance need to be addressed through modifications in the GUI.

Flaws in the Design of the GUI

When the client and server portions of an application are first used together, defects in the GUI that were not apparent during earlier prototyping and usability testing are likely to appear. Since a prototype primarily represents the client side of an application, testing it independently cannot present the user with the same range of events and experiences possible under the full implementation. However, if the prototyping process has been thorough, design defects identified during the implementation stage should be minor.

Performance Considerations Requiring Modification of the GUI

In some cases, the GUI may need to be modified not because of defects in the design of the interface itself, but because of the way the application performs when both client and server portions are tested together. For example, during testing it may become apparent that a particular operation takes a long time and that the GUI should give users information about the status or duration of the operation (see Guideline 8 under "Design Guidelines"). In addition to providing added information, the GUI might also be modified to offer users the option of halting an operation based on its status or duration.

In the preceding example, changing the GUI probably would not require changing the design of a feature so much as using existing features to provide added information or to offer a new choice of actions. In other cases, a modification in the design of a feature or the way it is implemented may be necessary.

For example, suppose users initially found a particular GUI feature to be highly useful during prototyping, but then found performance to be unacceptably slow when the feature was coded in the final programming language and tested with the server portion of the application. If poor performance resulted because the feature required that an excessive number of messages be sent between client and server, the problem would not be the fault of either the design or the server but of the interaction between the two. The problem might be dealt with by making a change in the design of the GUI, in the implementation of the GUI (if this reduced the number of messages), or in the server (if this were feasible and could improve message handling).

As a further example, suppose a GUI is designed to go with an electronic mail application that already has a character-based interface. The GUI allows users to scroll through mail messages a page at a time or line by line. The character-based interface only allows users to display messages a screenful at a time. Each time a user pages to a new screen, display text is provided by a server that always sends a screenful of text, unless it is at the end of a message. If the GUI is implemented and tested against the original server for the character-based interface, line-by-line scrolling is likely not to work at all or to result in erratic or burstlike displays of text.

Implementation begins when prototyping and usability testing have resulted in a stable GUI design.

Possible solutions to the problem would be to change the way line-by-line scrolling is implemented on the client system, to make modifications in both the server and the implementation of scrolling, or to change the design of the GUI and not allow line-by-line scrolling. The third option, not providing line-by-line scrolling, would be the least desirable, since line-by-line scrolling for continuous text is an expected feature of most GUIs.

Coordinating Final Implementation of the GUI and Back-End Software

If feasible, coding on the back end should not be closed until implementation of the GUI has been fully tested. Success of the GUI may depend on changes in either front-end or back-end software during the implementation stage.

The preceding section gave two examples of GUI features that were highly desirable, but could only be preserved if there were changes in the implementation of the GUI or in the server on the back end. In the example of line-by-line scrolling for an electronic mail application, it could turn out that changing the way scrolling was implemented would still not provide adequate scrolling. In this case, it would be highly desirable to make changes in the server or to write a new server, rather than drop line-by-line scrolling. Similarly, in the example of a GUI feature that sent too many messages to servers, it might be important to preserve the feature by making whatever changes were necessary, whether in the implementation of the GUI, the server, or possibly both.

In sum, if users find a GUI to be difficult or unpleasant to use, they will avoid using it as much as they can and will be more error-prone when they do use it. Consequently, as much as possible, developers should be prepared to make changes in either back-end or front-end software for the benefit of the GUI. A perfect application on the back end with an unmanageable GUI in front is of little value.

Conclusion

Simply providing a client/server application with a graphical user interface does not make the application easier to use or the user more productive. Developing an effective GUI is a three-stage process of analysis, design, and implementation. The first stage analyzes the tasks that need to be carried out through the GUI and the types of users who will be performing them. The second stage uses the analyses of tasks and users to make decisions about the design of the GUI. A number of guidelines are available for carrying out the design process. During the design stage, there is a repeated pattern of refinement in which a prototype GUI is evaluated, the design for the GUI revised, and a new prototype generated and evaluated. In the third stage, implementation, there is a further process of GUI refinement as the client and server portions of an application are repeatedly tested together, revised, and retested.

Reference

Tandem Style Guide for Graphical User Interfaces. Tandem Computers Incorporated. Part no. 96318. Forthcoming.

Acknowledgments

The author wishes to thank Ellen Chiri, Rita Danielsen, and Brady Farrand for all of their contributions to the *Tandem Style Guide for Graphical User Interfaces*, from which much of this article is derived. The author also thanks David Gordon for his perspicacity and patience in editing this article.

Susan Wolfe is manager of Tandem's Human Factors Department, which is responsible for the development of user interface standards, user interface design, and usability evaluations. Susan joined Tandem in 1987 after several years of experience at other companies as a human factors engineer.

Expand High-Performance Solutions

Expand™ networking software provides several alternatives for connecting a distributed network of Tandem systems. The Expand-over-LAN network access subsystem and FOX™ and FOXII fiber-optic extensions are the connection options best suited for high-performance applications.

Users planning to upgrade to Tandem systems that use new technology need to consider the impact these connection options may have on their proposed networks. These users should consider their networking requirements early in migration planning.

Many Tandem users have upgraded successfully from TXP™ and VLX™ systems to Tandem NonStop™ Series/RISC (TNS/R) systems. A significant number of TXP and VLX users configure the networks at their central sites with FOX rings. Current TNS/R systems, including the Cyclone/R™ and the CLX™ 1200 series, do not offer FOX but have many other advantages that can offset the absence of FOX.

Expand-over-LAN can be a high-speed alternative to FOX for users who plan to upgrade from TXP or VLX systems to TNS/R systems, or who plan to connect a TNS/R system to an existing FOX network. However, these users must consider several performance-related issues during their migration planning. When choosing a new CPU architecture, these users should determine whether the proposed networking subsystem can support their requirements for application throughput, application response time, and the use of CPU resources.

The performance results presented in this article can help MIS managers and planners decide whether Expand-over-LAN is a viable alternative to FOX for their high-performance, intersystem network applications. The current test results show that Expand-over-LAN performance has improved significantly over the past two years. These improvements make Expand-over-LAN a suitable solution for many applications requiring high performance.

This article compares the architectures of Expand-over-LAN and FOX, discusses the types of applications that use Expand networks, and describes several tests that examine key performance capabilities of Expand-over-LAN. It also briefly summarizes FOX performance capabilities. The tests were performed using the C30.09 release of Expand and the Tandem LAN Access Method (TLAM) subsystem. The article assumes that readers are familiar with Expand, basic application design, and the fundamentals of Tandem system architecture.

Expand-over-LAN Background

Expand-over-LAN was initially released for the C10 release of the Guardian™ 90 operating system. Many users expected Expand to use a high percentage of the Ethernet protocol's 10-megabit-per-second bandwidth. Experience has shown that the designs of Ethernet, the Tandem Ethernet controllers, and off-the-shelf applications, as well as Guardian 90 processor power, limit the Ethernet bandwidth utilized. Some users of TXP or VLX systems connected by FOX who wanted to upgrade to RISC-based CLX or Cyclone/R systems considered these limitations to be an obstacle to migration.

Over the past two years, Tandem developers enhanced and optimized Expand-over-LAN to improve its throughput and reduce its CPU cost. Expand-over-LAN can now achieve more than 10 times the throughput, for the same CPU cost, that it could when it was initially released. The Tandem 3615 TLAM controller, available on the D10 release of Guardian 90, should improve response time and provide higher potential throughput for each controller.

Expand Architecture

Expand software is an extension of the Guardian 90 operating system that provides a fault-tolerant network for intersystem communication. Users designing an application with modular requesters and servers can easily distribute the application across an Expand network.

An Expand network can include up to 255 nodes, and each node can support up to 63 Expand paths to other nodes in the network. Each node must have a unique name and number. Expand dynamically determines the best route from one system to another and automatically recognizes new systems as they are attached to the network.

An Expand path is a logical data path between two nodes; it is controlled by an Expand process at each end. A simple path consists of one line, which may be a FOX fiber-optic connection, a network connection, or a dedicated data communications line. A complex multiple-line path can consist of up to eight network connections or communication lines in parallel, but cannot include FOX fiber-optic connections. There are four connection options for Expand lines:

- FOX.
- Expand NetDirect (Expand direct connection).
- Expand-over-X.25, which uses Expand NetNAM (Network Access Method) over X.25 Access Method (X25AM).
- Expand-over-LAN, which uses Expand NetNAM over TLAM.

Of these choices, FOX provides the most efficient connections because it allows data to move across the interprocessor bus (IPB) instead of the I/O channel. The Guardian 90 message system and Expand combine to enable application and system processes to send data directly across dedicated optical fibers, bypassing Expand for all data transfers.

Expand NetDirect operates directly over bit-synchronous communications controllers that support bandwidths of up to 256 kilobits per second per line. Expand NetDirect uses industry-standard wide area network (WAN) protocols such as high-level data-link control (HDLC) to provide a reliable and efficient connection between systems.

Expand NetNAM over X25AM or over TLAM does not directly access a communications controller. Instead, Expand sends messages to either X25AM or TLAM using the Network Access Method (NAM) protocol. Expand NetNAM is designed so that Expand can access standards-based networks (such as an IEEE 802.3 LAN or an X.25 network) without implementing the necessary protocols within the Expand process.

For a comparison of all Expand connectivity options, see the article titled "TLAM: A Connectivity Option for Expand" in the April 1991 issue of the *Tandem Systems Review* (MacKenzie, 1991). For more information about Expand-over-LAN, refer to the white paper by Terrill (1991).

Architectural Comparison of Expand-over-LAN and FOX

When planning a migration, users should understand that Expand-over-LAN and FOX employ very different architectures. From a network design perspective, they both use high-bandwidth media, but they use the media in different ways. FOX uses a proprietary protocol over dedicated fiber-optic connections that are limited, for FOXII, to 4 kilometers between systems. Expand-over-LAN can operate over shared, industry-standard Ethernet networks and can be bridged to extend connections over WANs, thus connecting systems over wide geographic distances.

From a system design perspective, FOX and Expand-over-LAN differ significantly in the way they take advantage of system hardware resources.

Figure 1

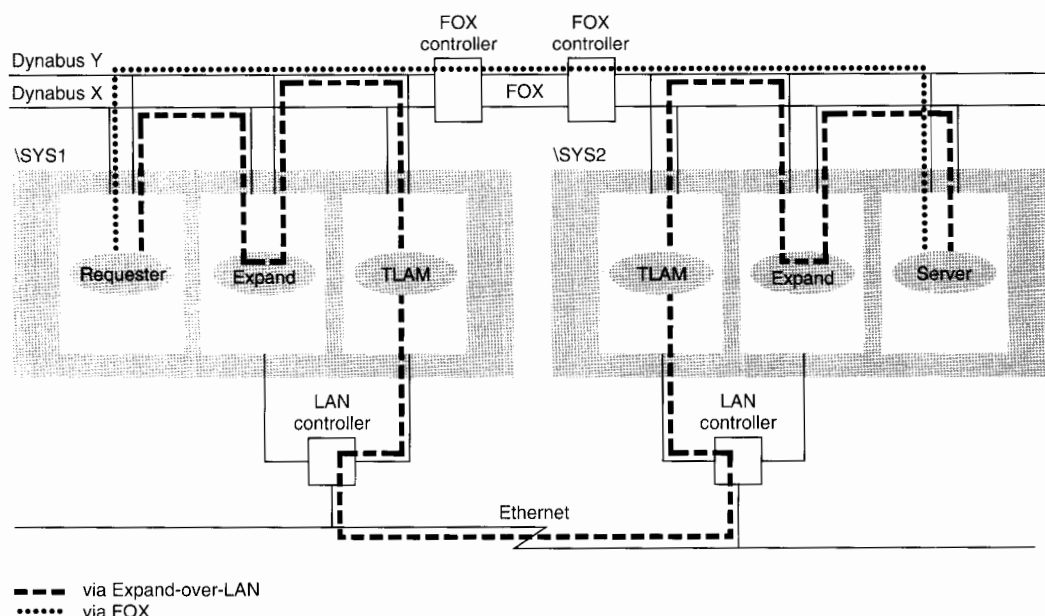


Figure 1.

A comparison of the message pathlengths through Expand-over-LAN and FOX.

Hardware Versus Software

FOX hardware is an extension of the IPB. Processes in CPUs in a FOX ring can address messages directly to CPUs in other nodes in the FOX ring. Because messages are sent out on the IPB, only a few additional lines of message-system code are required to send an intersystem message via FOX on the IPB.

Once the message is on the IPB, FOX hardware routes it directly to the destination node and CPU across the fault-tolerant FOX subsystem. Figure 1 shows how the physical FOX connection between IPBs in different systems is used to send a message directly to the destination CPU.

Messages sent through Expand-over-LAN may travel twice across the IPB before they leave the system. The first IPB hop is from the requester process to Expand; the second is from Expand to TLAM. Both Expand and TLAM must copy the message into a data buffer and process it. The pathlength through Expand and TLAM is longer at both ends than the pathlength through FOX, as illustrated in Figure 1.

Linear Expandability

FOX facilities are embedded in each CPU in a system. As the number of CPUs in the system increases, or the power of each CPU increases, the potential throughput of FOX increases linearly.

Expand-over-LAN supports one Expand path between two systems. One Expand path can include up to eight TLAM processes and eight Ethernet controllers and LANs. However, all data must be processed by Expand no matter how many communication lines are in the path. Expand, and the processing power of the CPU where it is located, limit the potential throughput and can provide limited linear expandability.

Figure 2

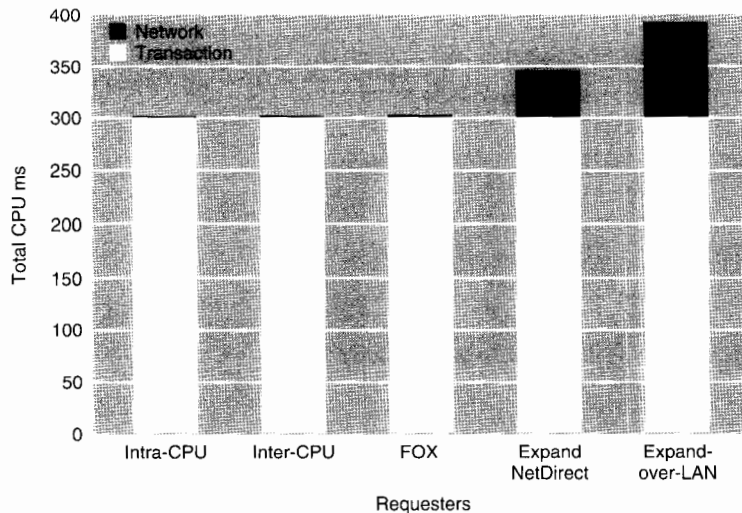


Figure 2.
Combined transaction and network costs of a hypothetical 300-ms transaction.

Network Application Design Issues

Some users with systems connected by FOX have taken advantage of the geographical independence within a FOX ring by configuring peripherals on any system in the ring. While this configuration is possible in a FOX ring, it is not considered good distributed system design. Network access costs are a component

of distributed system design and cannot be overlooked. System designers should reconsider deployment of applications and peripherals when network performance can affect application performance.

When comparing the impact of FOX and Expand-over-LAN on system performance, designers should remember that network delays and Expand costs are only two components that contribute to transaction response time and affect system sizing. Other components such as disk-process and I/O costs may contribute much more to the total response time and the use of system resources.

CPU Cost per Request

Figure 2 shows the incremental network cost when one sends a typical transaction between processes on Tandem systems. The transaction cost is the same in all cases and consists of disk-processing and application-processing costs.

Sending a message within the same CPU, the most efficient method, adds little to the total transaction cost. When one uses inter-CPU communication or FOX, the additional CPU cost is also minimal. When one uses Expand, a greater incremental network cost is added to the total transaction cost.

Transactions and Network Requests

Most Tandem applications and systems are designed to process a given number of transactions per second while maintaining a target response time. Each transaction must be handled by several components and may traverse a network at some points. Some transactions include a sequence of requests (messages and responses) between components, as shown in Figure 3.

The sample transaction in Figure 3 includes three network requests. Therefore, one must add the network delay three times to the total response time per transaction. To determine the CPU costs for this transaction, one must add the costs for the network access subsystem at both ends of the connection.

Application Types

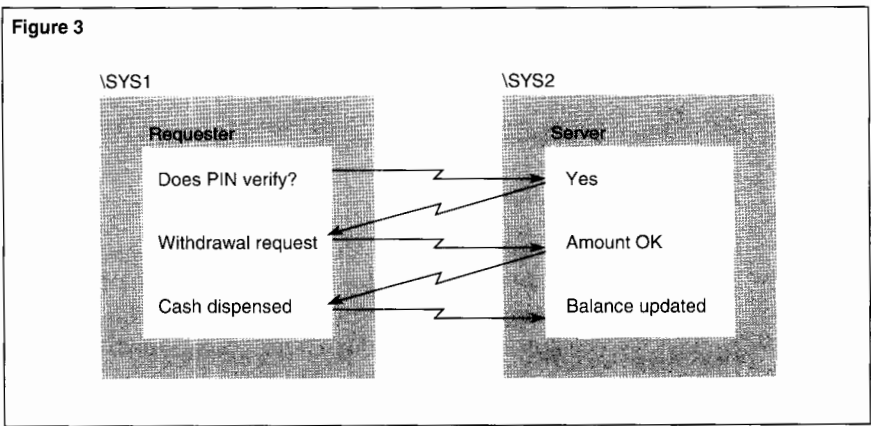
There are two broad categories of applications that often use network resources: online transaction processing (OLTP) and bulk transfer applications. The two are at opposite ends of the spectrum in their demands on network subsystem software and hardware.

OLTP Considerations

Most Tandem installations are designed to process high volumes of small transactions quickly. A transaction may include several small network requests, as shown in Figure 3. During peak periods of the business day, many transactions appear on the network simultaneously. Response time is most critical during these periods and is the key performance factor for OLTP applications. Throughput is not usually a limiting factor unless the network provides low bandwidth between systems.

Bulk Transfer Considerations

Applications requiring large network requests belong in this category. Typical bulk transfer applications include activities such as file transfers and remote-tape archiving operations. Bulk



transfer applications often read large blocks of data from disk. (A block can be almost 32 kilobytes in the Guardian 90 environment.) The application then sends the block in one large request to the network subsystem, which breaks it down into packets for transfer.

At the receiving end, the network subsystem must reassemble the entire block and forward it to the destination device or process. During the transfer, the sender usually waits for completion before preparing another large block.

Figure 3.
Three network requests and replies make up one completed transaction.

Table 1.
Maximum limits.

Connection	OLTP (50-byte message) requests/sec	Bulk (30-kilobyte message) Mb/sec
FOX	500/CPU	20.0
Expand-over-LAN	290/system	3.3

Bulk transfers demand high bandwidth. The network subsystem can process bulk transfers more efficiently (use fewer CPU resources per byte) than it can OLTP requests. Given unlimited network-subsystem CPU resources, bandwidth is the limiting factor. Given unlimited bandwidth, bulk demand tends to exhaust all available network-subsystem CPU resources.

Maximum Limits for OLTP and Bulk Transfer Applications

Table 1 compares the capabilities of Expand-over-LAN and FOX for OLTP and bulk transfer applications. The limits shown were achieved in controlled experiments with no disk or other system activity. Table 1 rates OLTP in requests per second. It rates bulk transfer in megabits-per-second potential throughput.

In this article, a request includes both a message and a response. Bulk megabits-per-second rates refer to user data and do not include additional message overhead added by networking protocols.

The test results cited in this article are drawn from a variety of tests performed during the past two years. Table 2 lists the configurations used to generate the test results shown in Table 1.

Users should consider some of the results shown in Table 1 within the perspective of other limitations. For example, if a system is designed to process 20 OLTP transactions per second, and each transaction includes three network requests, the maximum network request rate will be about 60 requests per second. Both Expand-over-LAN and FOX can support many times this network request rate.

Users should also view the bulk transfer results in relation to disk or tape throughput, or to how fast the applications can read or write to disk or tape. For example, if a tape-backup operation averages about 250 kilobytes per second, both Expand-over-LAN and FOX can support the required 2 megabits-per-second throughput.

Expand-over-LAN Performance Capabilities

It is important to understand the capabilities and limitations of any network environment before deciding whether it is a viable solution. The test results cited in this article should help users understand the performance capabilities of Expand-over-LAN.

Key Performance Components

Performance for networking and communications subsystems comprises three key components:

- Throughput.
- Response time.
- CPU efficiency.

The perfect network would provide unlimited throughput with zero response time at no CPU cost. Most network subsystems cannot achieve optimum performance for all three components at the same time. For example, response time is best when throughput is low because requests do not need to wait for the network to become available.

Maximum Throughput of Expand-over-LAN

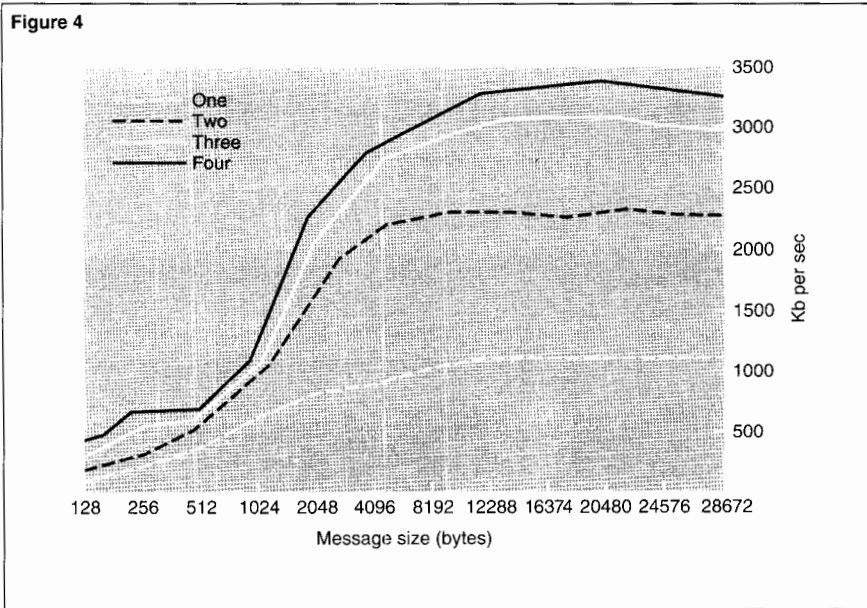
Figure 4 shows maximum throughputs obtained over an Expand multiline path with from one to four TLAM subsystems. (The test used Cyclone™ systems and Tandem 3613 controllers at both ends.) As shown, Expand-over-LAN throughput increases as more TLAM controllers are added until CPU and controller limits are reached. Throughput also increases as average message size increases, reflecting a change in application type from small OLTP transactions to bulk transfers.

The test set the FRAMESIZE parameter to the default value of 132 words and the PATHBLOCKBYTES parameter to 1480 bytes. Users can install this combination on any Expand path in a network without changing the network-wide value for the FRAMESIZE parameter. With Expand-over-LAN, throughput

Table 2. Configurations that generated the maximum limits shown in Table 1.*

Test	CPUs	CPU type	Message size (bytes)	Operation	Requesters
OLTP FOX	16	VLX	50	WRITEREAD	32
OLTP Expand	4	Cyclone	50	WRITEREAD	16
Bulk FOX	4	Cyclone	32K	WRITE	32
Bulk Expand	4	Cyclone	28K	WRITEREAD	16

*Expand FRAMESIZE 132, PATHBLOCKBYTES 1480



is relatively limited for OLTP applications. It is constrained mostly because Expand-over-LAN supports a limited request rate when average message sizes are small.

Figure 4. Maximum throughput for Expand-over-LAN, including from one to four LANs, on a Cyclone system.

Figure 5.
Cyclone/R OLTP request rates for Expand-over-LAN.

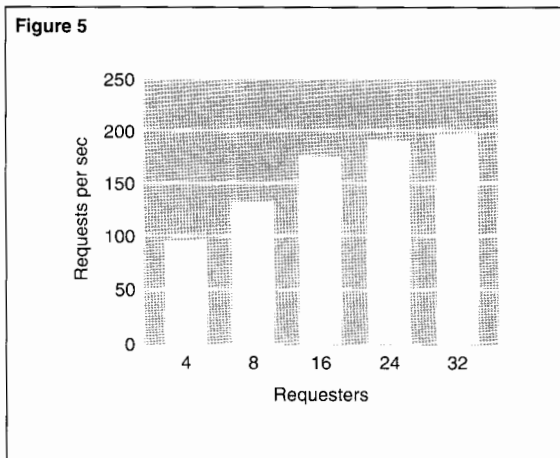
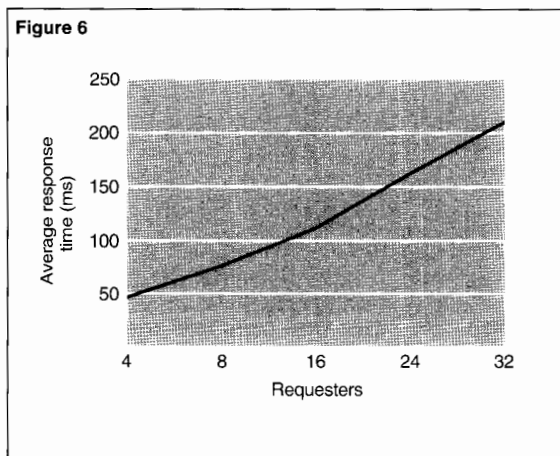


Figure 6.
Average CLX800 response times for 50-byte messages using Expand-over-LAN.



Users can employ Figure 4 to help determine if Expand-over-LAN can satisfy specific throughput requirements. For example, given an average message size of 512 bytes, one can achieve a potential throughput of 500 kilobits per second with multiple requesters.

For bulk message sizes, if a user's throughput requirements are higher than the achievable bandwidth shown in Figure 4, increasing the Expand FRAMESIZE parameter may provide higher throughput.

Request Rate

The Expand request rate is crucial for networked OLTP applications. Figure 5 illustrates the maximum requests per second supported by Expand-over-LAN. The peak request rate is limited by the design of Expand and varies depending on the CPU type. For example, with 64 concurrent requesters on a CLX800 CPU, Expand can process about 160 requests per second (rps); on a Cyclone/R CPU it can process about 210 rps; and on a Cyclone CPU it can process 290 rps. Typically, Expand reaches the maximum request rate even if the utilization of the Expand CPU is less than 80 percent.

Expand becomes more efficient when it can process multiple concurrent small requests. The Expand multipacket frame feature saves CPU cycles by grouping up to 32 small requests into each frame sent to TLAM. Expand becomes more efficient as the demand increases and uses much less CPU per request. For more information on the Expand multipacket frame feature, refer to Section 2 of the *System Generation Manual for Expand* (1993).

Figure 5 can help users determine if Expand-over-LAN can support an OLTP application. Figure 5 shows the rates Expand-over-LAN achieved on a Cyclone/R CPU. To determine the required request rate, one should multiply the transaction rate by the number of network requests per transaction. One should also consider the average number of concurrent requesters if it is fewer than 16.

OLTP Response Time

Response time requirements are critical to OLTP applications. Figure 6 shows the average network response times with Expand-over-LAN for small (50-byte) requests on a CLX800 system. (CPU type does not greatly affect response time.) Response time increases predictably as the number of requesters increases from 4 to 32. These results reflect the behavior of a classic single-server queue with a fairly long service time. All network requests to a given system must use the same Expand path (the same queue). Imagine, for example, a bank with one teller available to service all requests.

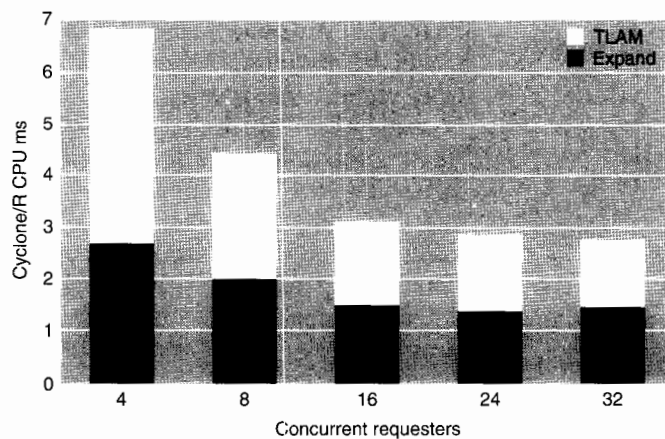
Configuring Expand and TLAM Processes

Tests performed on a CLX800 system show that Expand and TLAM can require up to 14 CPU milliseconds per OLTP request when there are few concurrent requests to process. The same tests performed on a Cyclone/R CPU show a maximum of 6.7 CPU milliseconds per request, as illustrated in Figure 7.

Expand and TLAM become more efficient as the demand increases. The CPU cost per request drops in half (to about 3 milliseconds) as the number of concurrent requesters (level of concurrency) increases to 16. Additional efficiency occurs with more than 16 requesters, but TLAM shows the most significant improvement between 4 and 16 requesters.

The CPU costs per request shown in Figure 7 are derived from tests performed with Expand and TLAM residing in the same CPU. Users should expect some additional CPU cost when Expand and TLAM are configured in separate CPUs. The least efficient case occurs when there are no concurrent requests; then the cost

Figure 7



per request can double. However, as the demand increases, Expand (and TLAM) become more efficient.

When designing a system for Expand-over-LAN, users can isolate Expand in one CPU and spread TLAM processes across other CPUs. As a rule of thumb, the CPU cost for Expand is slightly less than the total CPU cost for all TLAM processes added together.

Figure 7.
*Cyclone/R CPU costs of
Expand-over-LAN per
50-byte request.*

Users can employ Figure 7 to make a rough estimate of CPU costs for Expand and TLAM. First, one estimates the peak network request rate based on transactions per second and network requests per transaction. Assume, for example, that there are 80 requests per second. Next, one estimates the level of concurrency by dividing the per-second request rate by 10 ($80/10 = 8$). This calculation provides a rough estimate; the actual concurrency level can vary depending on application design.

One can approximate the level of concurrency with the number of requesters shown in Figure 7. Using the number of requesters as a guide, one can locate in Figure 7 the approximate CPU cost per request. Thus, if there are 8 concurrent requesters, each request would cost about 4 CPU milliseconds on a Cyclone/R CPU. The total CPU cost for 80 requests per second is about 320 CPU milliseconds, or 32 percent utilization of a Cyclone/R CPU.

At this level of utilization, it would be safe to locate both Expand and TLAM in the same CPU, and Expand and TLAM would operate at their peak efficiency. If the CPU cost were over 50 percent, it would be advisable to configure Expand and TLAM in separate CPUs, and some extra inter-CPU costs would apply.

FOX Performance Capabilities

FOX is often ignored in system design considerations because it provides more throughput than CPUs can utilize at no perceptible CPU cost. Like Expand-over-LAN, FOX throughput is limited by a maximum request rate that varies depending on CPU type. With FOX, however, one must multiply the peak rate by the number of CPUs in the system. Actual applications on a VLX system, or even a Cyclone system, are unlikely to approach the maximum potential request rate for FOX.

FOX response time is optimal when requesters (or servers) are distributed across all available CPUs. FOX behaves like a multiple-server queuing model. (Imagine, for example, a bank with up to 16 efficient tellers working concurrently.) Service time in each CPU is not adversely affected by FOX usage in other CPUs.

Adding FOX to a system effectively adds dedicated processing power to each CPU. The application uses the FOX subsystem exclusively for intersystem requests. Because the system can direct intersystem requests through FOX, it frees up processing power for other applications within the system.

Conclusion

When migrating from a VLX or TXP system connected by FOX, users should consider the network technology to be used before they finalize a technology-upgrade migration path. Users should examine the application's performance requirements as well as the CPU cost of supporting the network technology.

If current FOX network performance data is not available, users should analyze current applications that use FOX. Tandem performance specialists can help users monitor system performance to identify the peak message rates and average messages sizes sent through the FOX subsystem. Current performance data, together with the test results presented in this article, will contribute to a smooth migration to new technology.

Recent performance improvements make Expand-over-LAN a viable alternative to FOX for many types of applications. Expand-over-LAN can provide more throughput than most applications demand, and it can satisfy requirements for response time, request rate, and CPU cost in many situations.

References

MacKenzie, K. 1991. TLAM: A Connectivity Option for Expand. *Tandem Systems Review*, Vol. 7, No. 1. Tandem Computers Incorporated. Part no. 46988.

System Generation Manual for Expand. 1993. Tandem Computers Incorporated. Part no. 98644.

Terrill, W. 1991. *Expand over LAN: High Speed System Integration*. Tandem Computers Incorporated. Part no. 105021.

Acknowledgment

The author would like to thank Mike Noonan for his extensive contributions, which made this article possible.

Donald W. Smith has been at Tandem for over 10 years and has been involved with Expand support in a variety of roles since 1987. He provides Expand performance information to Tandem analysts and users.

**Application Profile: Storing Macintosh Graphics
on the Tandem 5200 Optical Storage Facility**

A *direct marketing services firm found that physical storage of its merchants' coupons was usurping valuable office floor space and that its manual coupon update service would soon be unable to keep up with the increase in requested changes. To help solve these problems, the company sought a reliable graphics image storage device. The MIS manager chose the Tandem™ 5200 Optical Storage Facility and worked with Tandem to find a way to link the device to a network of Macintosh computers.*

In 1985, graphic designers at Val-Pak Direct Marketing Services, Inc., used knives to manually cut and paste information and design changes on more than a million marketing coupons per year. Furthermore, the company's MIS professionals found that they had strained the capabilities of their existing IBM system to run the ordering, accounting, and purchasing functions of their expanding business. To remain competitive, Val-Pak had to make some drastic changes.

Val-Pak initially solved its back-office problems by replacing its IBM system with a Tandem NonStop™ II system. To improve the coupon updating process, analysts linked Macintosh computers through local area networks (LANs) and then contemplated a previously untried integration of the Tandem 5200 Optical Storage Facility (OSF) with Macintosh graphics.

This article describes how Val-Pak professionals first networked their Macintosh computers without using the OSF and how they later built a more reliable system by integrating the Macs with the OSF.

Val-Pak's Business Before Using the OSF

The direct marketing franchise business involves selling a service to merchants who want to advertise by sending coupons to selected prospects. Located in Largo, Florida, Val-Pak differentiates itself from similar direct marketing franchises by using its demographic data to target geographic zones that most closely match the merchant's requirements. Val-Pak also creates the coupons for the merchant and mails them to the prospective buyers.

An example of this targeted direct mail service might involve a merchant whose business is selling Mercedes-Benz hubcaps. The merchant might ask Val-Pak to select from its demographic data all persons who own Mercedes-Benz cars and who live in areas where the roads are bumpy and the people may thus be more apt to need new hubcaps. Val-Pak's database would provide information that tells when the next mailing to these people is scheduled. To these prospects, Val-Pak would send coupons for the merchant's Mercedes-Benz hubcaps.

Val-Pak's business is franchised, which means they have dealers across the country who have salesmen calling on merchants. Val-Pak charges back to the franchisee a dollar amount, or a percentage per coupon sent, depending on the number of mailings. The business generates major expenses in labor and materials, has little margin, and is highly competitive. In 1985, Val-Pak's mailings were increasing at a rate of 3,000 per week.

In 1985, Val-Pak replaced their overburdened IBM system with a Tandem NonStop II system. This purchase provided them more processing power and solved the computer overload in their back-office business systems. However, it did not solve the bottleneck their designers were experiencing in their daily coupon updating operations.

Val-Pak's hallways were lined with multiple rows of file cabinets loaded with merchants' hard-copy coupons representing three years of data. Simply changing an expiration date on a coupon involved finding the cabinet where the folder with the artwork resided, taking the folder to a work area, and then making up new hard copy by cutting and pasting portions of each coupon. The work was then placed on a layout board and the revised coupon photographed. Using an etching process, a plate was made from the photographic negative. The plate was then installed on a printing press, which put the image on paper. The coupon artwork was returned to the file cabinet where it was originally stored until the merchant requested another change.

Thirty-five percent of Val-Pak's volume was repeat business. Often, the requested change was only the expiration date of the coupon, but designers still had to do the tedious cutting and pasting. Because of Val-Pak's small profit margin and its growing business, Val-Pak managers had to reduce labor costs. They had to find a more efficient method for updating and archiving their graphics.

Choosing a Desktop Computer

In 1989, Val-Pak managers considered whether to use Macintosh computers or IBM PCs with DOS operating systems. They compared the response time to do graphics on a PC with the response time on a Mac and found that the Mac performance was better. The reason is that the Mac applications, graphics, and networking are all integrated into the workstation; on the PC, these products are layered and the integration of the products becomes the task of the user. Val-Pak managers found that the Mac also had a better WYSIWYG (what-you-see-is-what-you-get) interface.

The First Graphics Solution: Networked Macintoshes

Although the OSF could have provided up to 83.9 gigabytes of compact storage, Val-Pak managers felt it was too expensive an option. Instead, the managers wanted to network their Macs. Also, at that time, neither Tandem nor Apple Computer had announced a way for the routers on the LANs to understand the Transmission Control Protocol/Internet Protocol (TCP/IP) used by the Tandem system. Thus, Val-Pak lacked a means for the Macs to communicate with the Tandem system.

The company purchased 15 1-gigabyte disk drives, and MIS professionals attached them to Macintosh computers in which Ethernet cards were installed. They used one LAN to link the Macs through an AppleTalk network. Using Quark Xpress software, they put all of their graphics on the Macs. Apple's AppleShare product provided the server code for this system.

The server volumes quickly reached their data limit, and the 15 gigabytes of data storage were filled to capacity. Val-Pak's only alternatives were to add more disk drives or find a disk drive with larger storage capacity. The company needed a data-safe storage system that was expandable and also cost-effective.

Upgrading the Back-Office System

In 1989, Tandem offered Val-Pak a deal on an upgrade from their NonStop II system to a Tandem CLX™ 700 computer. Although Val-Pak was happy with the performance of their Tandem products, they also wanted to look for a host system that would interface with their Macs. At the time, no such interface was available. They researched computers from Digital Equipment Corporation, Sun Microsystems, and National Cash Register, all of whom proposed the use of servers. Since Val-Pak managers were pleased with the fault-tolerance and data integrity that their current Tandem system provided, they decided on the Tandem CLX 700 system with the hope that they could eventually interface it with their Macintosh computers.

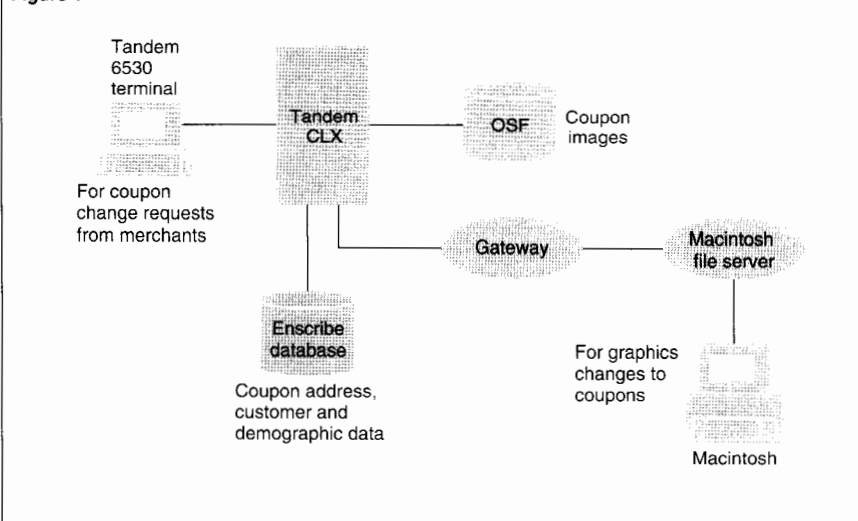
The Tandem-Apple Solution

Representatives from Tandem and Val-Pak met with those from Apple, and, with the help of a Val-Pak programmer, eventually solved the Tandem-Macintosh integration problem. The result was the plan for the system shown in Figure 1.

Creating New Coupons

When a merchant places an order for a new coupon, the Val-Pak designer and the merchant lay out the coupon together. If the merchant already has a coupon design, the merchant faxes the coupon to the designer and the designer uses an optical scanner to enter the coupon onto a Macintosh server. Then the coupon image must pass from the Mac server to the OSF.

Figure 1



Providing Macintosh-to-Tandem Communication

To provide communication between the Macs and the OSF, MIS professionals at Val-Pak installed an Ethernet card in each Mac in the network. The Ethernet cards provided high-speed connectivity and access to the TCP/IP transport mechanism, which allowed the Macs to communicate with the Tandem system.

To ensure that the coupon image will be found when needed, a host process stores its OSF location in a database on the Tandem system. The address is accessible by a host application program that fetches the coupon image on request.

Figure 1.

Using the Macintosh, the Tandem CLX, and the OSF for updating coupons.

An Intervening Postal Rate Increase

In November 1990, while Val-Pak MIS professionals were implementing the use of the OSF to reduce costs, the U.S. Postal Service announced a mailing rate increase. With planned mailings of 4.5 million envelopes per week, even a minimal increase would substantially affect Val-Pak's profits.

The increase was to take effect in February 1991, but the final specifications for the increase weren't available until January 1991. In effect, Val-Pak officials had less than 45 days to respond to the increase. Val-Pak managers appealed to postal service officials and were promised discounts if they could meet certain criteria.

If Val-Pak's personnel could minimize the postal workers' handling by stacking the mailing envelopes on trays and placing the trays on palettes loaded onto sealed trailers, the postal service would give them a discount. Val-Pak would receive an additional discount if its personnel sorted the palettes in the order that the postal workers needed them. A third discount was offered if Val-Pak would sort the trays in the sequence that the mail carrier would deliver them.

In addition, if Val-Pak could guarantee 70 percent saturation in a delivery area, known as a carrier route, the postal service would give another discount. To prove the saturation rate, Val-Pak had to provide names, addresses, and demographic data. Also, the post office required that the envelopes be sorted so that the mail carrier could pick the envelopes from the mail bag and, without looking at them, put them in the proper mail boxes.

Postal service officials offered to have a postmaster present to seal the trailers at Val-Pak's loading dock if Val-Pak would provide a consolidated statement on a Mac at the loading dock. If the postmaster could bill and settle the account at the loading dock, Val-Pak would receive still another discount.

Val-Pak managers knew that the massive sorts required to meet these requests, the queries to the demographic data, and the consolidation of bills would have to be designed and implemented on the Tandem computer (the front end of the system) at the same time that designers were implementing the Macintosh graphics output on the back end of the system.

Managers estimated the savings at \$200,000 per month or \$2.4 million a year if they could meet all the requirements for all of the postal discounts. The result was that Val-Pak was one of only two companies in the United States that was able to respond to the deadline in time to receive the discounts.

Val-Pak attributes their success to a talented staff and the suitable technology that Tandem provided them. To perform the sorts and queries necessary to meet the postal service requirements, Val-Pak's programmer loaded data from the Tandem Enscribe database into Tandem NonStop SQL tables. Because of their knowledge of Tandem systems and their programming skills, Val-Pak's MIS professionals were able to use Tandem technology to satisfy a complicated set of demands in a short time.

The Coupon Change Process on the Tandem System

During the order entry process, a merchant places an order for a change to a coupon, and a clerk enters the order using a Tandem 6530 terminal. Information about customers and change instructions for the coupons are stored in a database on the Tandem host. Another application stores demographic data here too.

A job scheduling process on the Tandem system keeps track of the jobs that the Val-Pak designers must do. Each morning the job scheduler places work that needs to be done on each designer's Mac desktop. When work on a coupon is finished, the designer sends the coupon image back to the job scheduler. The job scheduler then moves it to the OSF and to a Mac disk, from which it is transferred to the printing presses.

Storage on the Optical Storage Facility

The coupons are graphics images stored on the OSF as binary large objects (BLOBs). The BLOBs are uncompressed files of 250,000 bytes or more and represent four-part color. Each additional color exponentially increases the disk space that the BLOB occupies.

How the OSF Works

The OSF operates in a way similar to an old-fashioned juke box. The Tandem model 5200 OSF contains a stack of 32 optical disk platters. This storage facility is a write-once-read-many (WORM) device suited for archival systems in which data is never deleted from the disk. The OSF physically retrieves the correct platter, just as a juke box would, and the read-write heads read the platter, just as the arm in the juke box would play a record.

At Val-Pak, data usually resides on the OSF for 14 to 18 months. It remains on the OSF until the platter with the data is physically removed from the OSF. The removed platter is stored offline and a new blank platter is inserted in its place. Once a platter has been removed, its corresponding data in the Tandem database is purged.

Accessing Coupon Data With the Macintosh

When a Val-Pak designer requests the graphics data at a Mac, a host application, different from the order entry application, moves the appropriate file from the OSF through the gateway that translates the TCP/IP protocol to LocalTalk, the AppleTalk protocol.

From the gateway, the data is transferred to a 10-gigabyte server, where it is stored as a folder. At this point the file is out of the Tandem system and has entered the Macintosh part of the configuration. From the server, the folder moves to the designer's desktop.

To save time, a designer can batch requests to the OSF. For example, to work on four jobs that have been written to the OSF two months earlier, the designer sends one batch request to the OSF and a Val-Pak custom program notifies the designer when the data is available. This program searches the index on the Tandem database to learn the OSF locations of all four files. Then the program issues a command to the OSF and a Tandem process notifies the designer on the Mac that the files have been retrieved.

Once the files from the OSF are on the designer's desktop, the designer uses Quark Xpress graphics software to update their contents. This process completely replaces the manual cut-and-paste process. When the coupon changes are completed, the files are ready to be stored back on the OSF.

The Macintosh system accepts file and folder names of up to 32 characters, but the Tandem system allows only files (no folders) with names having a maximum of 8 characters. An in-house Val-Pak program converts the longer Mac names into unique 8-character names for the Tandem computer. Once the names are converted, the files can be stored on the OSF.

However, since the OSF is a WORM device, the designer cannot update the OSF. Instead, a new file is created. The old file remains archived on the OSF until its platter is physically removed.

When a coupon is ready to be printed, production personnel produce a plate, as in the old system, but they no longer have to take a photo to make a negative. Instead, the image is created on the Macintosh and is printed on several four-color 1200-dot-per-inch laser presses.

Improving Throughput

The first benchmark tests run on this system showed that the throughput using TCP/IP was 40 kilobytes per second. The Val-Pak application, however, needed an even higher bandwidth for optical data transmission. The use of User Datagram Protocol (UDP) instead of TCP/IP appeared at first to be a solution because it doubled the throughput. On the negative side, UDP leaves all error correction to the application and data integrity is not ensured during transmission. This was not acceptable.

Instead, Val-Pak managers increased the transmission speed by taking advantage of the performance capabilities of the Tandem CLX 800 system. By upgrading from a CLX 700 system to a CLX 800 system, they achieved a 30 percent increase in throughput.

Handling Message Collisions

On a large LAN, collisions between messages can be a network performance concern. On other networks the same size as Val-Pak's, the message collision rate is usually 13 to 15 percent, but the rate on Val-Pak's system is less than 2 percent. Val-Pak managers attribute this success to the fact that the system is not highly interactive. A long period of time transpires while the designer copies a coupon from a server, works on it, and then copies it back to the server. Thus, not many collisions are bound to take place.

A second reason for the low collision rate is that the Mac file servers have been allocated by the type of work they process. Each server is on its own Ethernet segment of the LAN. When a designer sends a request to download a coupon, the Tandem process knows where on the LAN to put it. If all servers were placed on one common Ethernet LAN, a collision would be more likely to occur.

Benefits of Using the OSF

Among the benefits Val-Pak received from using the OSF is a cost savings. The expense of magnetic storage on a large disk was from 10 to 14 dollars per megabyte. On the OSF, the cost is only 2 cents per megabyte.

By changing the hard-copy coupons in the file cabinets to images on the OSF, Val-Pak gained 2,000 square feet of storage space. The company completely eliminated hard-copy cutting and pasting. By replacing the unreliable disk drives on the Mac network with a reliable OSF, designers and production personnel had constant access to their coupon images. The result was an expedited printing process.

Val-Pak's maintenance problems were reduced too, for they moved from several large Mac disks to one OSF. Their current network consists of one LAN with 20 gigabytes of storage and the same number of servers as they had before they installed the OSF. Without the OSF, this configuration would have grown substantially by now, as would the disk maintenance. In addition, only 4, instead of 18, months of data would have been available online.

Acknowledgments

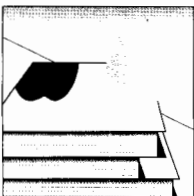
The author thanks David Hatala and Mark Gurley for providing almost all of the information for this article.

The author also thanks Steve Keteltas, Hugh Myers, Bob Unanski, and Joe Schrengohst for their availability for questions and Jon Harms of Val-Pak for his enthusiasm and cooperation.

Diane Broyles is a writing consultant and programmer who began working with Tandem's Technical Marketing Publications in September 1992.

Conclusion

After trying a Mac-only network solution, Val-Pak personnel proved the Tandem 5200 Optical Storage Facility to be ideal for their online storage, archival, and image processing needs. It provided a safe, fault-tolerant medium for storing graphic images. Val-Pak professionals segmented their LANs to minimize message collisions, and they improved the speed of the system by upgrading from a Tandem CLX 700 computer to a CLX 800 computer.



Tandem Education

The following paragraphs provide highlights of the latest education courses offered by Tandem. To sign up for a class or to order an independent study program (ISP), users should call 1-800-621-9198. Full descriptions of all the available courses and ISPs appear in the *Tandem Education Course Catalog* and on InfoWay.

Frame Relay Fundamentals

This independent study program provides a comprehensive introduction to frame relay technology and products. The study guide traces the events contributing to the outgrowth of the frame relay network interface. Students learn how frame relay technology evolved from the Integrated

Services Digital Network (ISDN) and X.25 standards to deal with the increasing number of LAN-interconnected personal computers and workstations in business, government, education, and research organizations. The study guide also describes and illustrates frame relay function and protocols in network operations.

Integrity Systems for Analysts

This lecture-and-lab course focuses on Tandem enhancements that differentiate the Integrity systems from other available UNIX systems. The five-day course currently deals with System V, Release 3. Installation and administrative procedures are taught within the broader context of the architecture and design of the various subsystems. Procedures are restricted to customer-replaceable unit (CRU) management. No field-replaceable unit (FRU) procedures are covered.

ISDN Fundamentals

This independent study program provides a comprehensive tutorial on Integrated Services Digital Network (ISDN) architecture and products. The study guide explores the history of digital transmission facilities in the predivestiture United States telecommunications network, which was owned and operated by AT&T. It also explains the evolution of T-1 encoding and framing techniques. Students gain a solid foundation in the ISDN architecture and the related CCITT standards, as well as in ISDN's functional layers and their respective protocols.

The Technical Information and Education department is an annotated list of new Tandem education courses and consulting and information services, as well as other technical information of interest to Tandem users.

NetWare Fundamentals

This independent study program provides a comprehensive guide to Novell's NetWare network operating system. The function and operation of services such as print and file sharing, messaging, and distributed processing are discussed. With this self-study guide, the student gains an understanding of the NetWare product line, as well as related client/server and network management applications. In addition, this ISP discusses NetWare's support of local area network protocols, wide area network connectivity, and telecommunications technologies.

NonStop NET/MASTER Rule Management Services (RMS)

In this classroom-and-lab course, students acquire the skills needed to use Rule Management Services (RMS) in the NonStop NET/MASTER environment. Students can thus achieve the benefits of automation using RMS. The three-day course includes hands-on exercises using the NonStop NET/MASTER RMS application in an operational environment.

OSI Fundamentals

This independent study program provides a comprehensive introduction to Open Systems Interconnection (OSI) concepts. The study guide begins by giving the student an understanding of the reasons that OSI came into being and the organizations involved in the development of OSI standards. The guide then covers subjects such as the OSI Reference Model and OSI functions, protocols, data structures, and applications. In addition, the ISP discusses the OSI Profiles (MAP, TOP, and GOSIP) and the issues of management and security.

Problem Solving for Tandem Operators

This five-day course (a replacement for the previous Operator Training II offering) provides an overview of basic Tandem problem-solving tools and introduces a systematic problem solving methodology. It also teaches new operators how to (1) monitor, start up, shut down, and change the state of basic system components, and (2) how to deal with the most common problems these components present.

SNAX/APC API and Transaction Design

This lecture-and-lab course teaches the student how to develop peer-to-peer transactions using Release 3 of the SNAX Advanced Program Communication (SNAX/APC) software package. The five-day course provides thorough coverage of the design and implementation of LU6.2-based transactions on the Tandem system for use in an Advanced Peer-to-Peer (APPC) environment.

SNAX/HLS Programming

This lecture-and-lab course teaches the student how to develop transactions communicating with Systems Network Architecture (SNA) devices or applications using the SNAX High-Level Support (SNAX/HLS) application programming interface. The course lasts five days.

Tandem Networking for Cooperative Processing

This independent study program provides technical managers with an introduction to the communications software and functionality offered by Tandem that support cooperative processing applications.

TCP/IP Fundamentals

This independent study program provides a comprehensive introduction to Transmission Control Protocol/Internet Protocol (TCP/IP). In this ISP, the student learns what TCP/IP is and how it works. TCP/IP functional layers, data structures, applications, and related protocols and services, such as UDP, SNMAP, and ICMP, are covered.

X.25 Fundamentals

This independent study program provides a comprehensive introduction to the CCITT Recommendation X.25 interface and related products. The study guide traces the events contributing to the popularity of public and private packet-switched X.25 networks. It also describes and illustrates the three levels of the X.25 interface, including the physical, data link, and packet levels. The student will see where X.25 fits into the OSI Reference Model and how it relates to X.3, X.28, X.29, and X.75.

TandemSystemsReviewIndex

The *Tandem Journal* became the *Tandem Systems Review* in February 1985. Four issues of the *Tandem Journal* were published:

Volume 1, No. 1	Fall 1983
Volume 2, No. 1	Winter 1984
Volume 2, No. 2	Spring 1984
Volume 2, No. 3	Summer 1984

As of this issue, 22 issues of the *Tandem Systems Review* have been published:

Volume 1, No. 1	Feb. 1985	Volume 6, No. 1	March 1990
Volume 1, No. 2	June 1985	Volume 6, No. 2	Oct. 1990
Volume 2, No. 1	Feb. 1986	Volume 7, No. 1	April 1991
Volume 2, No. 2	June 1986	Volume 7, No. 2	Oct. 1991
Volume 2, No. 3	Dec. 1986	Volume 8, No. 1	Spring 1992
Volume 3, No. 1	March 1987	Volume 8, No. 2	Summer 1992
Volume 3, No. 2	Aug. 1987	Volume 8, No. 3	Fall 1992
Volume 4, No. 1	Feb. 1988	Volume 9, No. 1	Winter 1993
Volume 4, No. 2	July 1988	Volume 9, No. 2	Spring 1993
Volume 4, No. 3	Oct. 1988	Volume 9, No. 3	Summer 1993
Volume 5, No. 1	April 1989		
Volume 5, No. 2	Sept. 1989		

The articles published in all 26 issues are arranged by subject below. (*Tandem Journal* is abbreviated as TJ and *Tandem Systems Review* as TSR.) A second index, arranged by product, is also provided.

Index by Subject

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
APPLICATION DEVELOPMENT AND LANGUAGES					
Ada: Tandem's Newest Compiler and Programming Environment	R. Vnuk	TSR	3,2	Aug. 1987	83940
A New Design for the PATHWAY TCP	R. Wong	TJ	2,2	Spring 1984	83932
An Overview of Client/Server Computing on Tandem Systems	H. Cooperstein	TSR	8,3	Fall 1992	89803
An Introduction to Tandem EXTENDED BASIC	J. Meyerson	TJ	2,2	Spring 1984	83932
Application Code Conversion for D-Series Systems	K. Liu	TSR	9,2	Spring 1993	89805
Application Profile: Storing Macintosh Graphics on the Tandem 5200 Optical Storage Facility	D. Broyles	TSR	9,3	Summer 1993	89806
Debugging TACL Code	L. Palmer	TSR	4,2	July 1988	13693
Designing and Implementing a Graphical User Interface	S. Wolfe	TSR	9,3	Summer 1993	89806
Designing Client/Server Applications for OLTP on Guardian 90 Systems	W. Culman	TSR	8,3	Fall 1992	89803
Implementing Client/Server Using RSC	M. Iem, T. Kocher	TSR	8,3	Fall 1992	89803
Instrumenting Applications for Effective Event Management	J. Dagenais	TSR	7,2	Oct. 1991	65248
New TAL Features	C. Lu, J. Murayama	TSR	2,2	June 1986	83837
PATHFINDER—An Aid for Application Development	S. Benett	TJ	1,1	Fall 1983	83930

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
APPLICATION DEVELOPMENT AND LANGUAGES <i>(cont.)</i>					
PATHWAY IDS: A Message-level Interface to Devices and Processes	M. Anderton, M. Noonan	TSR	2,2	June 1986	83937
The RESPOND OLTP Business Management System for Manufacturing	H. Bolling, W. Bronson	TSR	9,1	Winter 1993	89804
State-of-the-Art C Compiler	E. Kit	TSR	2,2	June 1986	83937
TACL, Tandem's New Extensible Command Language	J. Campbell, R. Glascock	TSR	2,1	Feb. 1986	83936
Tandem's New COBOL85	D. Nelson	TSR	2,1	Feb. 1986	83936
The DAL Server: Client/Server Access to Tandem Databases	W. Schlansky, J. Schrengohst	TSR	9,1	Winter 1993	89804
The ENABLE Program Generator for Multifile Applications	B. Chapman, J. Zimmerman	TSR	1,1	Feb. 1985	83934
TMF and the Multi-Threaded Requester	T. Lemberger	TJ	1,1	Fall 1983	83930
Writing a Command Interpreter	D. Wong	TSR	1,2	June 1985	83935
CLIENT/SERVER					
An Overview of Client/Server Computing on Tandem Systems	H. Cooperstein	TSR	8,3	Fall 1992	89803
Application Profile: Storing Macintosh Graphics on the Tandem 5200 Optical Storage Facility	D. Broyles	TSR	9,3	Summer 1993	89806
Designing and Implementing a Graphical User Interface	S. Wolfe	TSR	9,3	Summer 1993	89806
Designing Client/Server Applications for OLTP on Guardian 90 Systems	W. Culman	TSR	8,3	Fall 1992	89803
Gateways to NonStop SQL	D. Slutz	TSR	6,2	Oct. 1990	46987
Implementing Client/Server Using RSC	M. Iem, T. Kocher	TSR	8,3	Fall 1992	89803
The DAL Server: Client/Server Access to Tandem Databases	W. Schlansky, J. Schrengohst	TSR	9,1	Winter 1993	89804
DATA COMMUNICATIONS					
An Overview of SNAX/CDF	M. Turner	TSR	5,2	Sept. 1989	28152
A SNAX Passthrough Tutorial	D. Kirk	TJ	2,2	Spring 1984	83932
Changes in FOX	N. Donde	TSR	1,2	June 1985	83935
Connecting Terminals and Workstations to Guardian 90 Systems	E. Siegel	TSR	8,2	Summer 1992	69848
Expand High-Performance Solutions	D. Smith	TSR	9,3	Summer 1993	89806
Introduction to MULTILAN	A. Coyle	TSR	4,1	Feb. 1988	11078
Overview of the MULTILAN Server	A. Rowe	TSR	4,1	Feb. 1988	11078
SNAX/APC: Tandem's New SNA Software for Distributed Processing	B. Grantham	TSR	3,1	March 1987	83939
SNAX/HLS: An Overview	S. Saltwick	TSR	1,2	June 1985	83935
TLAM: A Connectivity Option for Expand	K. MacKenzie	TSR	7,1	April 1991	46988
Using the MULTILAN Application Interfaces	M. Berg, A. Rowe	TSR	4,1	Feb. 1988	11078

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
DATA MANAGEMENT					
A Comparison of the B00 DP1 and DP2 Disc Processes	T. Schachter	TSR	1,2	June 1985	83935
An Overview of NonStop SQL Release 2	M. Pong	TSR	6,2	Oct. 1990	46987
Batch Processing in Online Enterprise Computing	T. Keefauver	TSR	6,2	Oct. 1990	46987
Concurrency Control Aspects of Transaction Design	W. Senf	TSR	6,1	March 1990	32968
Converting Database Files from ENSCRIBE to NonStop SQL	W. Weikel	TSR	6,1	March 1990	32986
DP1-DP2 File Conversion: An Overview	J. Tate	TSR	2,1	Feb. 1986	83936
Determining FCP Conversion Time	J. Tate	TSR	2,1	Feb. 1986	83936
DP2's Efficient Use of Cache	T. Schachter	TSR	1,2	June 1985	83935
DP2 Highlights	K. Carlyle, L. McGowan	TSR	1,2	June 1985	83935
DP2 Key-sequenced Files	T. Schachter	TSR	1,2	June 1985	83935
Gateways to NonStop SQL	D. Slutz	TSR	6,2	Oct. 1990	46987
High-Performance SQL Through Low-Level System Integration	A. Borr	TSR	4,2	July 1988	13693
Improvements in TMF	T. Lemberger	TSR	1,2	June 1985	83935
NetBatch: Managing Batch Processing on Tandem Systems	D. Wakashige	TSR	5,1	April 1989	18662
NetBatch-Plus: Structuring the Batch Environment	G. Earle, D. Wakashige	TSR	6,1	March 1990	32986
NonStop SQL: The Single Database Solution	J. Cassidy, T. Kocher	TSR	5,2	Sept. 1989	28152
NonStop SQL Data Dictionary	R. Holbrook, D. Tsou	TSR	4,2	July 1988	13693
NonStop SQL Optimizer: Basic Concepts	M. Pong	TSR	4,2	July 1988	13693
NonStop SQL Optimizer: Query Optimization and User Influence	M. Pong	TSR	4,2	July 1988	13693
NonStop SQL Reliability	C. Fenner	TSR	4,2	July 1988	13693
Online Information Processing	J. Viescas	TSR	9,1	Winter 1993	89804
Online Reorganization of Key-Sequenced Tables and Files	G. Smith	TSR	6,2	Oct. 1990	46987
Optimizing Batch Performance	T. Keefauver	TSR	5,2	Sept. 1989	28152
Overview of NonStop SQL	H. Cohen	TSR	4,2	July 1988	13693
Parallelism in NonStop SQL Release 2	M. Moore, A. Sodhi	TSR	6,2	Oct. 1990	46987
The NonStop SQL Release 2 Benchmark	S. Englert, J. Gray, T. Kocher, P. Shah	TSR	6,2	Oct. 1990	46987
The Outer Join in NonStop SQL	J. Vaishnav	TSR	6,2	Oct. 1990	46987
The Relational Data Base Management Solution	G. Ow	TJ	2,1	Winter 1984	83931
Tandem's NonStop SQL Benchmark	Tandem Performance Group	TSR	4,1	Feb. 1988	11078
The TRANSFER Delivery System for Distributed Applications	S. Van Pelt	TJ	2,2	Spring 1984	83932
TMF Autorollback: A New Recovery Feature	M. Pong	TSR	1,1	Feb. 1985	83934

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
OPERATING SYSTEMS					
Application Code Conversion for D-Series Systems	K. Liu	TSR	9,2	Spring 1993	89805
Highlights of the B00 Software Release	K. Coughlin, R. Montevaldo	TSR	1,2	June 1985	83935
Increased Code Space	A. Jordan	TSR	1,2	June 1985	83935
Managing System Time Under GUARDIAN 90	E. Nellen	TSR	2,1	Feb. 1986	83936
Migration Planning for D-Series Systems	S. Kuukka	TSR	9,2	Spring 1993	89805
New GUARDIAN 90 Time-keeping Facilities	E. Nellen	TSR	1,2	June 1985	83935
New Process-timing Features	S. Sharma	TSR	1,2	June 1985	83935
NonStop II Memory Organization and Extended Addressing	D. Thomas	TJ	1,1	Fall 1983	83930
Overview of the C00 Release	L. Marks	TSR	4,1	Feb. 1988	11078
Overview of the D-Series Guardian 90 Operating System	W. Bartlett	TSR	9,2	Spring 1993	89805
Overview of the NonStop-UX Operating System for the Integrity S2	P. Norwood	TSR	7,1	April 1991	46988
Robustness to Crash in a Distributed Data Base: A Nonshared-memory Approach	A. Borr	TSR	1,2	June 1985	83935
The GUARDIAN Message System and How to Design for It	M. Chandra	TSR	1,1	Feb. 1985	83935
The Tandem Global Update Protocol	R. Carr	TSR	1,2	June 1985	83935
PERFORMANCE AND CAPACITY PLANNING					
A Performance Retrospective	P. Oleinick, P. Shah	TSR	2,3	Dec. 1986	83938
Buffering for Better Application Performance	R. Mattran	TSR	2,1	Feb. 1986	83936
Capacity Planning Concepts	R. Evans	TSR	2,3	Dec. 1986	83938
Capacity Planning With TCM	W. Highleyman	TSR	7,2	Oct. 1991	65248
C00 TMDs Performance	J. Mead	TSR	4,1	Feb. 1988	11078
Credit-authorizing Benchmark for High Performance and Linear Growth	T. Chmiel, T. Houy	TSR	2,1	Feb. 1986	83936
Debugging Accelerated Programs on TNS/R Systems	D. Cressler	TSR	8,1	Spring 1992	65250
DP2 Performance	J. Enright	TSR	1,2	June 1985	83935
Estimating Host Response Time in a Tandem System	H. Horwitz	TSR	4,3	Oct. 1988	15748
Expand High-Performance Solutions	D. Smith	TSR	9,3	Summer 1993	89806
FASTSORT: An External Sort Using Parallel Processing	J. Gray, M. Stewart, A. Tsukerman, S. Uren, B. Vaughan	TSR	2,3	Dec. 1986	83938
Getting Optimum Performance from Tandem Tape Systems	A. Khatri	TSR	2,3	Dec. 1986	83938
How to Set Up a Performance Data Base with MEASURE and ENFORM	M. King	TSR	2,3	Dec. 1986	83938
Improved Performance for BACKUP2 and RESTORE2	A. Khatri, M. McCline	TSR	1,2	June 1985	83935
Improving Performance on TNS/R Systems With the Accelerator	M. Blanchet	TSR	8,1	Spring 1992	65250
MEASURE: Tandem's New Performance Measurement Tool	D. Dennison	TSR	2,3	Dec. 1986	83938
Measuring DSM Event Management Performance	M. Stockton	TSR	8,1	Spring 1992	65250
Message System Performance Enhancements	D. Kinkade	TSR	2,3	Dec. 1986	83938
Message System Performance Tests	S. Uren	TSR	2,3	Dec. 1986	83938
Network Design Considerations	J. Evjen	TSR	5,2	Sept. 1989	28152
NonStop VLX Performance	J. Enright	TSR	2,3	Dec. 1986	83938
Optimizing Sequential Processing on the Tandem System	R. Welsh	TJ	2,3	Summer 1984	83933
Pathway TCP Enhancements for Application Run-Time Support	R. Vannucci	TSR	7,1	April 1991	46988

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
PERFORMANCE AND CAPACITY PLANNING (cont.)					
Performance Benefits of Parallel Query Execution and Mixed Workload Support in NonStop SQL Release 2	S. Englert, J. Gray	TSR	6,2	Oct. 1990	46987
Performance Considerations for Application Processes	R. Glasstone	TSR	2,3	Dec. 1986	83938
Performance Measurements of an ATM Network Application	N. Cabell, D. Mackie	TSR	2,3	Dec. 1986	83938
Predicting Response Time in On-line Transaction Processing Systems	A. Khatri	TSR	2,2	June 1986	83937
The 6600 and TCC6820 Communications Controllers: A Performance Comparison	P. Beadles	TSR	2,3	Dec. 1986	83938
The ENCORE Stress Test Generator for On-line Transaction Processing Applications	S. Kosinski	TJ	2,1	Winter 1984	83931
The PATHWAY TCP: Performance and Tuning	J. Vatz	TSR	1,1	Feb. 1985	83934
The Performance Characteristics of Tandem NonStop Systems	J. Day	TJ	1,1	Fall 1983	83930
Sizing Cache for Applications that Use B-series DP1 and TMF	P. Shah	TSR	2,2	June 1986	83937
Sizing the Spooler Collector Data File	H. Norman	TSR	4,1	Feb. 1988	11978
Tandem's 5200 Optical Storage Facility: Performance and Optimization Considerations	S. Coleman	TSR	5,1	April 1989	18662
Tandem's Approach to Fault Tolerance	B. Ball, W. Bartlett, S. Thompson	TSR	4,1	Feb. 1988	11078
Understanding PATHWAY Statistics	R. Wong	TJ	2,2	Spring 1984	83932
PERIPHERALS					
5120 Tape Subsystem Recording Technology	W. Phillips	TSR	3,2	Aug. 1987	83940
An Introduction to DYNAMITE Workstation Host Integration	S. Kosinski	TSR	1,2	June 1985	83935
Application Profile: Storing Macintosh Graphics on the Tandem 5200 Optical Storage Facility	D. Broyles	TSR	9,3	Summer 1993	89806
Data-Encoding Technology Used in the XL8 Storage Facility	D. S. Ng	TSR	2,2	June 1986	83937
Data-Window Phase-Margin Analysis	A. Painter, H. Pham, H. Thomas	TSR	2,2	June 1986	83937
Introducing the 3207 Tape Controller	S. Chandran	TSR	1,2	June 1985	83935
Peripheral Device Interfaces	J. Blakkan	TSR	3,2	Aug. 1987	83940
Plated Media Technology Used in the XL8 Storage Facility	D.S. Ng	TSR	2,2	June 1986	83937
Streaming Tape Drives	J. Blakkan	TSR	3,2	Aug. 1987	83940
Terminal Selection	E. Siegel	TSR	8,2	Summer 1992	69848
The 5200 Optical Storage Facility: A Hardware Perspective	A. Patel	TSR	5,1	April 1989	18662
The 6100 Communications Subsystem: A New Architecture	R. Smith	TJ	2,1	Winter 1984	83931
The 6600 and TCC6820 Communications Controllers: A Performance Comparison	P. Beadles	TSR	2,3	Dec. 1986	83938
The DYNAMITE Workstation: An Overview	G. Smith	TSR	1,2	June 1985	83935
The Model 6VI Voice Input Option: Its Design and Implementation	B. Huggett	TJ	2,3	Summer 1984	83933
The Role of Optical Storage in Information Processing	L. Sabaroff	TSR	3,2	Aug. 1987	83940
The V8 Disc Storage Facility: Setting a New Standard for On-line Disc Storage	M. Whiteman	TSR	1,2	June 1985	83935

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
PROCESSORS					
Fault Tolerance in the NonStop Cyclone System	S. Chan, R. Jardine	TSR	7,1	April 1991	46988
NonStop CLX: Optimized for Distributed On-Line Transaction Processing	D. Lenoski	TSR	5,1	April 1989	18662
NonStop VLX Hardware Design	M. Brown	TSR	2,3	Dec. 1986	83938
Overview of Tandem NonStop Series/RISC Systems	L. Faby, R. Mateosian	TSR	8,1	Spring 1992	65250
The High-Performance NonStop TXP Processor Transaction Processing	W. Bartlett, T. Houy, D. Meyer	TJ	2,1	Winter 1984	83931
The NonStop TXP Processor: A Powerful Design for On-line Transaction Processing	P. Oleinick	TJ	2,3	Summer 1984	83933
The VLX: A Design for Serviceability	J. Allen, R. Boyle	TSR	3,1	March 1987	83939
SECURITY					
Dial-In Security Considerations	P. Grainger	TSR	7,2	Oct. 1991	65248
Distributed Protection with SAFEGUARD	T. Chou	TSR	2,2	June 1986	83937
Enhancing System Security With Safeguard	C. Gaydos	TSR	7,1	April 1991	46988
SYSTEM CONNECTIVITY					
Building Open Systems Interconnection with OSI/AS and OSI/TS	R. Smith	TSR	6,1	March 1990	32986
Connecting Terminals and Workstations to Guardian 90 Systems	E. Siegel	TSR	8,2	Summer 1992	69848
Implementing Client/Server Using RSC	M. Iem, T. Kocher	TSR	8,3	Fall 1992	89803
Network Design Considerations	J. Evjen	TSR	5,2	Sept. 1989	28152
Terminal Connection Alternatives for Tandem Systems	J. Simonds	TSR	5,1	April 1989	18662
Terminal Selection	E. Siegel	TSR	8,2	Summer 1992	69848
The OSI Model: Overview, Status, and Current Issues	A. Dunn	TSR	5,1	April 1989	18662
SYSTEM MANAGEMENT					
Configuring Tandem Disk Subsystems	S. Sittler	TSR	2,3	Dec. 1986	83938
Data Replication in Tandem's Distributed Name Service	T. Eastep	TSR	4,3	Oct. 1988	15748
Enhancements to TMDS	L. White	TSR	3,2	Aug. 1987	83940
Event Management Service Design and Implementation	H. Jordan, R. McKee, R. Schuet	TSR	4,3	Oct. 1988	15748
Introducing TMDS, Tandem's New On-line Diagnostic System	J. Troisi	TSR	1,2	June 1985	83935
Instrumenting Applications for Effective Event Management	J. Dagenais	TSR	7,2	Oct. 1991	65248
Measuring DSM Event Management Performance	M. Stockton	TSR	8,1	Spring 1992	65250
Network Statistics System	M. Miller	TSR	4,3	Oct. 1988	15748
Overview of DSM	P. Homan, B. Malizia, E. Reisner	TSR	4,3	Oct. 1988	15748
SCP and SCF: A General Purpose Implementation of the Subsystem Programmatic Interface	T. Lawson	TSR	4,3	Oct. 1988	15748
RDF: An Overview	J. Guerrero	TSR	7,2	Oct. 1991	65248
RDF Synchronization	F. Jongma, W. Senf	TSR	8,2	Summer 1992	69848

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
SYSTEM MANAGEMENT <i>(cont.)</i>					
Tandem's Subsystem Programmatic Interface	G. Tom	TSR	4,3	Oct. 1988	15748
Using FOX to Move a Fault-tolerant Application	C. Breighner	TSR	1,1	Feb. 1985	83934
Using the Subsystem Programmatic Interface and Event Management Services	K. Stobie	TSR	4,3	Oct. 1988	15748
VIEWPOINT Operations Console Facility	R. Hansen, G. Stewart	TSR	4,3	Oct. 1988	15748
VIEWSYS: An On-line System-resource Monitor	D. Montgomery	TSR	1,2	June 1985	83935
Writing Rules for Automated Operations	J. Collins	TSR	7,2	Oct. 1991	65248
UTILITIES					
Enhancements to PS MAIL	R. Funk	TSR	3,1	March 1987	83939

Index by Product

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
3207 TAPE CONTROLLER					
Introducing the 3207 Tape Controller	S. Chandran	TSR	1,2	June 1985	83935
5120 TAPE SUBSYSTEM					
5120 Tape Subsystem Recording Technology	W. Phillips	TSR	3,2	Aug. 1987	83940
5200 OPTICAL STORAGE					
Application Profile: Storing Macintosh Graphics on the Tandem 5200 Optical Storage Facility	D. Broyles	TSR	9,3	Summer 1993	89806
Tandem's 5200 Optical Storage Facility: Performance and Optimization Considerations	S. Coleman	TSR	5,1	April 1989	18662
The 5200 Optical Storage Facility: A Hardware Perspective	A. Patel	TSR	5,1	April 1989	18662
The Role of Optical Storage in Information Processing	L. Sabaroff	TSR	4,1	Feb. 1988	11078
6100 COMMUNICATIONS SUBSYSTEM					
The 6100 Communications Subsystem: A New Architecture	R. Smith	TJ	2,1	Winter 1984	83931
6530 TERMINAL					
The Model 6VI Voice Input Option: Its Design and Implementation	B. Huggett	TJ	2,3	Summer 1984	83933
6600 AND TCC6820 COMMUNICATIONS CONTROLLERS					
The 6600 and TCC6820 Communications Controllers: A Performance Comparison	P. Beadles	TSR	2,3	Dec. 1986	83938
Ada					
Ada: Tandem's Newest Compiler and Programming Environment	R. Vnuk	TSR	3,2	Aug. 1987	83940
BASIC					
An Introduction to Tandem EXTENDED BASIC	J. Meyerson	TJ	2,2	Spring 1984	83932
C					
State-of-the-art C Compiler	E. Kit	TSR	2,2	June 1986	83937
CIS					
Customer Information Service	J. Massucco	TSR	3,1	March 1987	83939
CLX					
NonStop CLX: Optimized for Distributed On-Line Transaction Processing	D. Lenoski	TSR	5,1	April 1989	18662
COBOL85					
Tandem's New COBOL85	D. Nelson	TSR	2,1	Feb. 1986	83936
COMINT (CI)					
Writing a Command Interpreter	D. Wong	TSR	1,2	June 1985	83935
CYCLONE					
Fault Tolerance in the NonStop Cyclone System	S. Chan, R. Jardine	TSR	7,1	April 1991	46988
DAL SERVER					
The DAL Server: Client/Server Access to Tandem Databases	W. Schlansky, J. Schrengohst	TSR	9,1	Winter 1993	89804

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
DP1 AND DP2					
A Comparison of the B00 DP1 and DP2 Disc Processes	T. Schachter	TSR	1,2	June 1985	83935
Determining FCP Conversion Time	J. Tate	TSR	2,1	Feb. 1986	83936
DP1-DP2 File Conversion: An Overview	J. Tate	TSR	2,1	Feb. 1986	83936
DP2 Highlights	K. Carlyle, L. McGowan	TSR	1,2	June 1985	83935
DP2 Key-sequenced Files	T. Schachter	TSR	1,2	June 1985	83935
DP2 Performance	J. Enright	TSR	1,2	June 1985	83935
DP2's Efficient Use of Cache	T. Schachter	TSR	1,2	June 1985	83935
Sizing Cache for Applications that Use B-series DP1 and TMF	P. Shah	TSR	2,2	June 1986	83937
DSM					
Data Replication in Tandem's Distributed Name Service	T. Eastep	TSR	4,3	Oct. 1988	15748
Event Management Service Design and Implementation	H. Jordan, R. McKee, R. Schuet	TSR	4,3	Oct. 1988	15748
Instrumenting Applications for Effective Event Management	J. Dagenais	TSR	7,2	Oct. 1991	65248
Measuring DSM Event Management Performance	M. Stockton	TSR	8,1	Spring 1992	65250
Network Statistics System	M. Miller	TSR	4,3	Oct. 1988	15748
Overview of DSM	P. Homan, B. Malizia, E. Reisner	TSR	4,3	Oct. 1988	15748
SCP and SCF: A General Purpose Implementation of the Subsystem Programmatic Interface	T. Lawson	TSR	4,3	Oct. 1988	15748
Tandem's Subsystem Programmatic Interface	G. Tom	TSR	4,3	Oct. 1988	15748
Using the Subsystem Programmatic Interface and Event Management Services	K. Stobie	TSR	4,3	Oct. 1988	15748
VIEWPOINT Operations Console Facility	R. Hansen, G. Stewart	TSR	4,3	Oct. 1988	15748
Writing Rules for Automated Operations	J. Collins	TSR	7,2	Oct. 1991	65248
DYNAMITE					
An Introduction to DYNAMITE Workstation Host Integration	S. Kosinski	TSR	1,2	June 1985	83935
The DYNAMITE Workstation: An Overview	G. Smith	TSR	1,2	June 1985	83935
ENABLE					
The ENABLE Program Generator for Multifile Applications	B. Chapman, J. Zimmerman	TSR	1,1	Feb. 1985	83934
ENCOMPASS					
The Relational Data Base Management Solution	G. Ow	TJ	2,1	Winter 1984	83931
ENCORE					
The ENCORE Stress Test Generator for On-line Transaction Processing Applications	S. Kosinski	TJ	2,1	Winter 1984	83931
ENSCRIBE					
Converting Database Files from ENSCRIBE to NonStop SQL	W. Weikel	TSR	6,1	March 1990	32986
EXPAND					
Expand High-Performance Solutions	D. Smith	TSR	9,3	Summer 1993	89806
FASTSORT					
FASTSORT: An External Sort Using Parallel Processing	J. Gray, M. Stewart, A. Tsukerman, S. Uren, B. Vaughan	TSR	2,3	Dec. 1986	83938

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
FOX					
Changes in FOX	N. Donde	TSR	1,2	June 1985	83935
Using FOX to Move a Fault-tolerant Application	C. Breighner	TSR	1,1	Feb. 1985	83934
FUP					
Online Reorganization of Key-Sequenced Tables and Files	G. Smith	TSR	6,2	Oct. 1990	46987
GUARDIAN 90					
Application Code Conversion for D-Series Systems	K. Liu	TSR	9,2	Spring 1993	89805
B00 Software Manuals	S. Olds	TSR	1,2	June 1985	83935
C00 Software Manuals	E. Levi	TSR	4,1	Feb. 1988	11078
Highlights of the B00 Software Release	K. Coughlin, R. Montevaldo	TSR	1,2	June 1985	83935
Improved Performance for BACKUP2 and RESTORE2	A. Khatri, M. McCline	TSR	1,2	June 1985	83935
Increased Code Space	A. Jordan	TSR	1,2	June 1985	83935
Managing System Time Under GUARDIAN 90	E. Nellen	TSR	2,1	Feb. 1986	83936
Message System Performance Enhancements	D. Kinkade	TSR	2,3	Dec. 1986	83938
Message System Performance Tests	S. Uren	TSR	2,3	Dec. 1986	83938
Migration Planning for D-Series Systems	S. Kuukka	TSR	9,2	Spring 1993	89805
New GUARDIAN 90 Time-keeping Facilities	E. Nellen	TSR	1,2	June 1985	83935
New Process-timing Features	S. Sharma	TSR	1,2	June 1985	83935
NonStop II Memory Organization and Extended Addressing	D. Thomas	TJ	1,1	Fall 1983	83930
Overview of the C00 Release	L. Marks	TSR	4,1	Feb. 1988	11078
Overview of the D-Series Guardian 90 Operating System	W. Bartlett	TSR	9,2	Spring 1993	89805
Robustness to Crash in a Distributed Data Base: A Nonshared-memory Multiprocessor Approach	A. Borr	TSR	1,2	June 1985	83935
Tandem's Approach to Fault Tolerance	B. Ball, W. Bartlett, S. Thompson	TSR	4,1	Feb. 1988	11078
The GUARDIAN Message System and How to Design for It	M. Chandra	TSR	1,1	Feb. 1985	83934
The Tandem Global Update Protocol	R. Carr	TSR	1,2	June 1985	83935
INTEGRITY S2					
Overview of the NonStop-UX Operating System for the Integrity S2	P. Norwood	TSR	7,1	April 1991	46988
MEASURE					
How to Set Up a Performance Data Base with MEASURE and ENFORM	M. King	TSR	2,3	Dec. 1986	83938
MEASURE: Tandem's New Performance Measurement Tool	D. Dennison	TSR	2,3	Dec. 1986	83938
MULTILAN					
Introduction to MULTILAN	A. Coyle	TSR	4,1	Feb. 1988	11078
Overview of the MULTILAN Server	A. Rowe	TSR	4,1	Feb. 1988	11078
Using the MULTILAN Application Interfaces	M. Berg, A. Rowe	TSR	4,1	Feb. 1988	11078

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
NETBATCH-PLUS					
NetBatch: Managing Batch Processing on Tandem Systems	D. Wakashige	TSR	5,1	April 1989	18662
NetBatch-Plus: Structuring the Batch Environment	G. Earle, D. Wakashige	TSR	6,1	March 1990	32986
NONSTOP SQL					
An Overview of NonStop SQL Release 2	M. Pong	TSR	6,2	Oct. 1990	46987
Concurrency Control Aspects of Transaction Design	W. Senf	TSR	6,1	March 1990	32986
Converting Database Files from ENSCRIBE to NonStop SQL	W. Weikel	TSR	6,1	March 1990	32986
Gateways to NonStop SQL	D. Slutz	TSR	6,2	Oct. 1990	46987
High-Performance SQL Through Low-Level System Integration	A. Borr	TSR	4,2	July 1988	13693
NonStop SQL Data Dictionary	R. Holbrook, D. Tsou	TSR	4,2	July 1988	13693
NonStop SQL: The Single Database Solution	J. Cassidy, T. Kocher	TSR	5,2	Sept. 1989	28152
NonStop SQL Optimizer: Basic Concepts	M. Pong	TSR	4,2	July 1988	13693
NonStop SQL Optimizer: Query Optimization and User Influence	M. Pong	TSR	4,2	July 1988	13693
NonStop SQL Reliability	C. Fenner	TSR	4,2	July 1988	13693
Overview of NonStop SQL	H. Cohen	TSR	4,2	July 1988	13693
Parallelism in NonStop SQL Release 2	M. Moore, A. Sodhi	TSR	6,2	Oct. 1990	46987
Performance Benefits of Parallel Query Execution and Mixed Workload Support in NonStop SQL Release 2	S. Englert, J. Gray	TSR	6,2	Oct. 1990	46987
Tandem's NonStop SQL Benchmark	Tandem Performance Group	TSR	4,1	Feb. 1988	11078
The NonStop SQL Release 2 Benchmark	S. Englert, J. Gray, T. Kocher, P. Shah	TSR	6,2	Oct. 1990	46987
The Outer Join in NonStop SQL	J. Vaishnav	TSR	6,2	Oct. 1990	46987
OSI					
Building Open Systems Interconnection with OSI/AS and OSI/TS	R. Smith	TSR	6,1	March 1990	32986
The OSI Model: Overview, Status, and Current Issues	A. Dunn	TSR	5,1	April 1989	18662
PATHFINDER					
PATHFINDER—An Aid for Application Development	S. Benett	TJ	1,1	Fall 1983	83930
PATHWAY					
A New Design for the PATHWAY TCP	R. Wong	TJ	2,2	Spring 1984	83932
PATHWAY IDS: A Message-level Interface to Devices and Processes	M. Anderton, M. Noonan	TSR	2,2	June 1986	83937
Pathway TCP Enhancements for Application Run-Time Support	R. Vannucci	TSR	7,1	April 1991	46988
The PATHWAY TCP: Performance and Tuning	J. Vatz	TSR	1,1	Feb. 1985	83934
Understanding PATHWAY Statistics	R. Wong	TJ	2,2	Spring 1984	83932
POET					
Designing Client/Server Applications for OLTP on Guardian 90 Systems	W. Culman	TSR	8,3	Fall 1992	89803
PS MAIL					
Enhancements to PS MAIL	R. Funk	TSR	3,1	March 1987	83939

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
RDF					
RDF: An Overview	J. Guerrero	TSR	7,2	Oct. 1991	65248
RDF Synchronization	F. Jongma, W. Senf	TSR	8,2	Summer 1992	69848
RESPOND					
The RESPOND OLTP Business Management System for Manufacturing	H. Bolling, W. Bronson	TSR	9,1	Winter 1993	89804
RSC					
Implementing Client/Server Using RSC	M. Iem, T. Kocher	TSR	8,3	Fall 1992	89803
SAFEGUARD					
Dial-In Security Considerations	P. Grainger	TSR	7,2	Oct. 1991	65248
Distributed Protection with SAFEGUARD	T. Chou	TSR	2,2	June 1986	83937
Enhancing System Security With Safeguard	C. Gaydos	TSR	7,1	April 1991	46988
SNAX					
An Overview of SNAX/CDF	M. Turner	TSR	5,2	Sept. 1989	28152
A SNAX Passthrough Tutorial	D. Kirk	TJ	2,2	Spring 1984	83932
SNAX/APC: Tandem's New SNA Software for Distributed Processing	B. Grantham	TSR	3,1	March 1987	83939
SNAX/HLS: An Overview	S. Saltwick	TSR	1,2	June 1985	83935
SPOOLER					
Sizing the Spooler Collector Data File	H. Norman	TSR	4,1	Feb. 1988	11078
TACL					
Debugging TACL Code	L. Palmer	TSR	4,2	July 1988	13693
TACL, Tandem's New Extensible Command Language	J. Campbell, R. Glascock	TSR	2,1	Feb. 1986	83936
TAL					
New TAL Features	C. Lu, J. Murayama	TSR	2,2	June 1986	83837
TCM					
Capacity Planning With TCM	W. Highleyman	TSR	7,2	Oct. 1991	65248
TLAM					
TLAM: A Connectivity Option for Expand	K. MacKenzie	TSR	7,1	April 1991	46988
TMDS					
C00 TMDS Performance	J. Mead	TSR	4,1	Feb. 1988	11078
Enhancements to TMDS	L. White	TSR	3,2	Aug. 1987	83940
Introducing TMDS, Tandem's New On-line Diagnostic System	J. Troisi	TSR	1,2	June 1985	83935
TMF					
Improvements in TMF	T. Lemberger	TSR	1,2	June 1985	83935
TMF and the Multi-Threaded Requester	T. Lemberger	TJ	1,1	Fall 1983	83930
TMF Autorollback: A New Recovery Feature	M. Pong	TSR	1,1	Feb. 1985	83934
TNS/R					
Debugging Accelerated Programs on TNS/R Systems	D. Cressler	TSR	8,1	Spring 1992	65250
Improving Performance on TNS/R Systems With the Accelerator	M. Blanchet	TSR	8,1	Spring 1992	65250
Overview of Tandem NonStop Series/RISC Systems	L. Faby, R. Mateosian	TSR	8,1	Spring 1992	65250

Article title	Author(s)	Publication	Volume, Issue	Publication date	Part number
TRANSFER					
The TRANSFER Delivery System for Distributed Applications	S. Van Pelt	TJ	2,2	Spring 1984	83932
TXP					
The High-Performance NonStop TXP Processor	W. Bartlett, T. Houy, D. Meyer	TJ	2,1	Winter 1984	83931
The NonStop TXP Processor: A Powerful Design for On-line Transaction Processing	P. Oleinick	TJ	2,3	Summer 1984	83933
V8					
The V8 Disc Storage Facility: Setting a New Standard for On-line Disc Storage	M. Whiteman	TSR	1,2	June 1985	83935
VIEWSYS					
VIEWSYS: An On-line System-resource Monitor	D. Montgomery	TSR	1,2	June 1985	83935
VLX					
NonStop VLX Hardware Design	M. Brown	TSR	2,3	Dec. 1986	83938
NonStop VLX Performance	J. Enright	TSR	2,3	Dec. 1986	83938
The VLX: A Design for Serviceability	J. Allen, R. Boyle	TSR	3,1	March 1987	83939
XL8					
Data-encoding Technology Used in the XL8 Storage Facility	D. S. Ng	TSR	2,2	June 1986	83937
Plated Media Technology Used in the XL8 Storage Facility	D. S. Ng	TSR	2,2	June 1986	83937

Tandem Systems Review Order Form

Use this form to order new subscriptions, change subscription information, and order back issues.

- ☐ I am a Tandem customer. My Tandem sales representative is _____.
- ☐ I am not a Tandem customer and am enclosing a check or money order for the requests indicated on this form. (Subscriptions are \$75 per year and each back issue is \$20. Make checks payable to Tandem Computers Incorporated.)

Subscription Information

- ☐ New subscription
- ☐ Update to subscription information
Subscription number: _____

Your subscription number is in the upper right corner of the mailing label.

COMPANY

NAME

JOB TITLE

DIVISION

ADDRESS

COUNTRY

TELEPHONE NUMBER (include all codes for U.S. dialing)

Title or position:

- ☐ President/CEO
- ☐ Director/VP information services
- ☐ MIS/DP manager
- ☐ Software development manager
- ☐ Programmer/analyst
- ☐ System operator
- ☐ End user
- ☐ Other: _____

Your association with Tandem:

- ☐ Tandem customer
- ☐ Third-party vendor
- ☐ Consultant
- ☐ Other: _____

Back Issue Requests

Number of copies **Tandem Systems Review**

- | | |
|---------------------------------|----------------------------------|
| _____ Vol. 1, No. 1, Feb. 1985 | _____ Vol. 6, No. 1, March 1990 |
| _____ Vol. 1, No. 2, June 1985 | _____ Vol. 6, No. 2, Oct. 1990 |
| _____ Vol. 2, No. 1, Feb. 1986 | _____ Vol. 7, No. 1, April 1991 |
| _____ Vol. 2, No. 2, June 1986 | _____ Vol. 7, No. 2, Oct. 1991 |
| _____ Vol. 2, No. 3, Dec. 1986 | _____ Vol. 8, No. 1, Spring 1992 |
| _____ Vol. 3, No. 1, March 1987 | _____ Vol. 8, No. 2, Summer 1992 |
| _____ Vol. 3, No. 2, Aug. 1987 | _____ Vol. 8, No. 3, Fall 1992 |
| _____ Vol. 4, No. 1, Feb. 1988 | _____ Vol. 9, No. 1, Winter 1993 |
| _____ Vol. 4, No. 2, July 1988 | _____ Vol. 9, No. 2, Spring 1993 |
| _____ Vol. 4, No. 3, Oct. 1988 | _____ Vol. 9, No. 3, Summer 1993 |
| _____ Vol. 5, No. 1, April 1989 | |
| _____ Vol. 5, No. 2, Sept. 1989 | |

Tandem Journal

- | | |
|----------------------------------|----------------------------------|
| _____ Vol. 1, No. 1, Fall 1983 | _____ Vol. 2, No. 2, Spring 1984 |
| _____ Vol. 2, No. 1, Winter 1984 | _____ Vol. 2, No. 3, Summer 1984 |

For questions or ordering information, call
800-473-5868 in the U.S. and Canada or
+1-408-285-0665 in other countries.

Send this form to:

Tandem Computers Incorporated
Tandem Systems Review, Loc 208-65
10400 Ridgeview Court
Cupertino, CA 95014-0723
FAX: +1-408-285-0840

Tandem employees must order their subscriptions and back issues through Courier.

Menu sequence: Marketing Information→
Literature Orders→ Technical Marketing
Pubs (TSR)

▲ FOLD



▲ FOLD

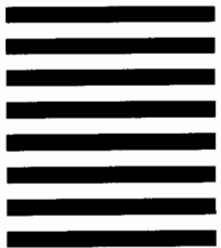
BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 482 CUPERTINO, CA U.S.A.

POSTAGE WILL BE PAID BY ADDRESSEE

TANDEM SYSTEMS REVIEW
LOC 208-65
TANDEM COMPUTERS INCORPORATED
19333 VALLCO PARKWAY
CUPERTINO, CA 95014-9862

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



▼ FOLD

▼ FOLD

TandemSystemsReviewReaderSurvey

The purpose of this questionnaire is to help the *Tandem Systems Review* staff select topics for publication. Postage is prepaid when mailed in the United States. Readers outside the U.S. should send their replies to their nearest Tandem sales office.

1. How useful is each article in this issue?

Product Update

01 ☐ Indispensable 02 ☐ Very 03 ☐ Somewhat 04 ☐ Not at all

Designing and Implementing a Graphical User Interface

05 ☐ Indispensable 06 ☐ Very 07 ☐ Somewhat 08 ☐ Not at all

Expand High-Performance Solutions

09 ☐ Indispensable 10 ☐ Very 11 ☐ Somewhat 12 ☐ Not at all

Application Profile: Storing Macintosh Graphics on the Tandem 5200 Optical Storage Facility

13 ☐ Indispensable 14 ☐ Very 15 ☐ Somewhat 16 ☐ Not at all

Technical Information and Education

17 ☐ Indispensable 18 ☐ Very 19 ☐ Somewhat 20 ☐ Not at all

2. I specifically would like to see more articles on (select one):

- 21 ☐ Overview discussions of new products and enhancements 22 ☐ Performance and tuning information
23 ☐ High-level overviews on Tandem's approach to solutions 24 ☐ Application design and customer profiles
25 ☐ Technical discussions of product internals
26 ☐ Other _____

3. Your title or position:

- 27 ☐ President, VP, Director 28 ☐ Systems analyst 29 ☐ System operator
30 ☐ MIS manager 31 ☐ Software developer 32 ☐ End user
33 ☐ Other _____

4. Your association with Tandem:

- 34 ☐ Tandem customer 35 ☐ Tandem employee 36 ☐ Third-party vendor 37 ☐ Consultant
38 ☐ Other _____

5. Comments

NAME

COMPANY NAME

ADDRESS

▲ FOLD



▲ FOLD

BUSINESS REPLY MAIL

FIRST CLASS

PERMIT NO. 482

CUPERTINO, CA U.S.A.

POSTAGE WILL BE PAID BY ADDRESSEE

TANDEM SYSTEMS REVIEW

LOC 208-65

TANDEM COMPUTERS INCORPORATED

19333 VALLCO PARKWAY

CUPERTINO, CA 95014-9862

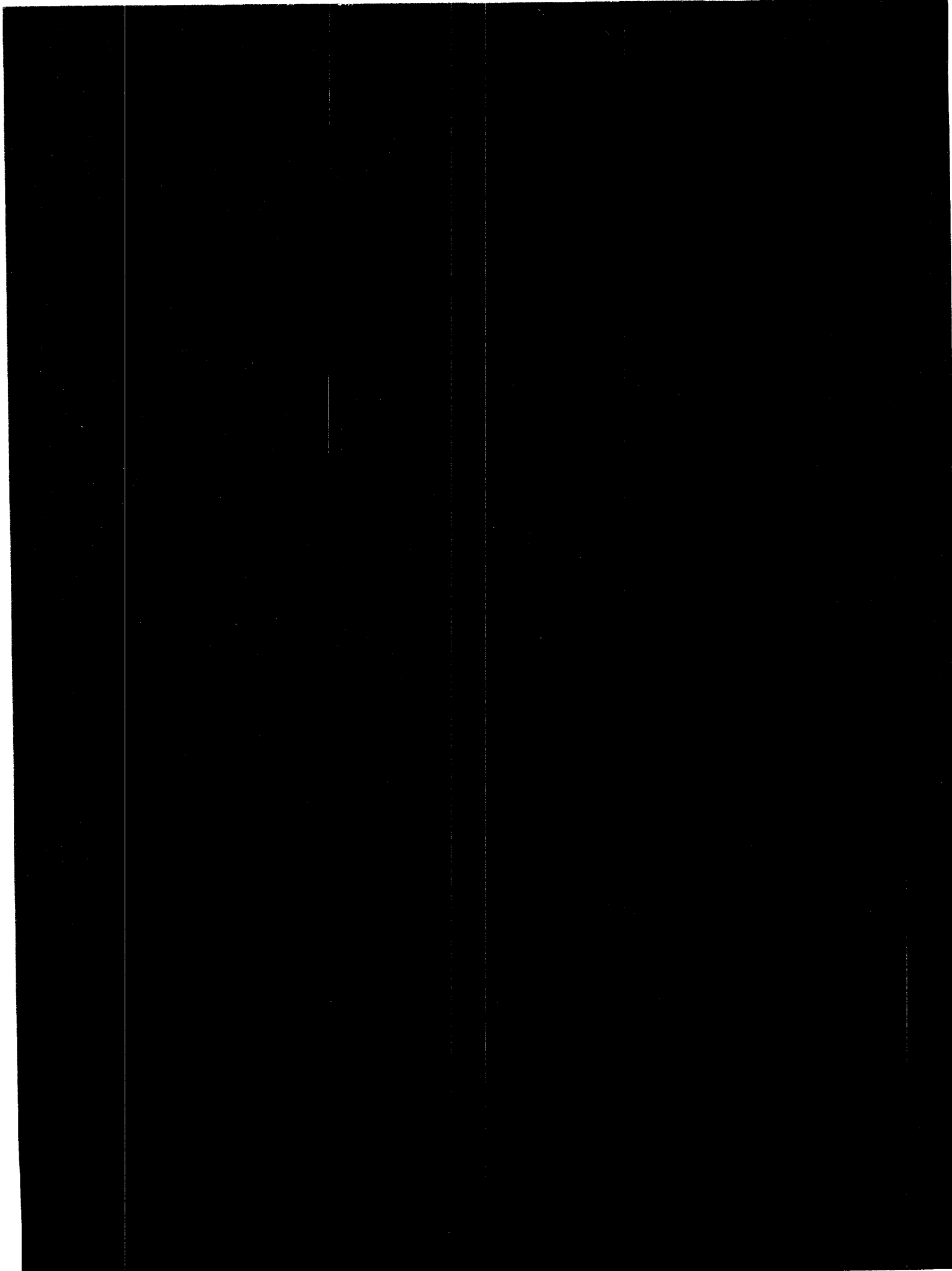
NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



▼ FOLD



▼ FOLD





Tandem Computers Incorporated
19333 Vallco Parkway
Cupertino, CA 95014-2599

MARC BRANDIFINO
LOC NUM 56-00
NEW YORK NY DOWNTOWN DISTRICT