OPERATIONS SYSTEMS TECHNOLOGY FOR NEW AT&T NETWORK AND SERVICE CAPABILITIES

Robert Merski and D. Mark Parrish

Robert Merski *and* ***D. Mark Parrish*** *are with AT&T Bell Laboratories (Red Hill) in Middletown, New Jersey. Mr. Merski is a distinguished member of technical staff in the Network Operations Systems Project Management and System Test Department and is the project engineer for version 1A of TOPAS (Testing, Operations, Provisioning, Administration System). He joined the company in 1977 and has a B.A. and an M.S. in mathematics from The Catholic University of America. Mr. Parrish is supervisor of the Network Provisioning Features Group in the Network Operations Systems Software Development Department. He joined the company in 1981 and has a B.S. in applied mathematics from Columbia University and an M.S. in computer science from the Polytechnic Institute of New York.*

Operations systems (OSs) have assumed an expanded role in the network and are instrumental in providing increasingly complex services. Therefore, a strategy was needed to be able to develop a new high-quality OS or upgrade an existing one in a cost-effective and timely way. This strategy includes a standard AT&T operating environment and a modular software architecture that emphasizes large-scale module reuse and layered virtual machines. With this base, operations systems will be able to respond effectively to future network needs. This article discusses the architecture of TOPAS, which is a primary example of an advanced operations system. Because of its modular design, considerable TOPAS software was reused in TMAS, another operations system.

Perspective

Computer-based operations systems (OSs) provide links between the rapidly evolving network, the personnel who operate and maintain the network, and the network's customers. The traditional role of an OS has been to provide cost-effective network operations. Typical OS applications include electronic record management, centralization of operations personnel, automated circuit testing, and planning tools. These applications have been successfully providing high-quality AT&T telecommunication services.

Yet, new and expanded roles for operations systems have emerged over the last few years. These new roles are being driven by the need to provide new customer services in increasingly shorter intervals. In addition, OSs must be able to respond to the increasing complexity of services and to new and diverse network technologies.

Operations systems make possible features like customer-initiated testing and access-line monitoring. Such features may be available in several customer service offerings; e.g., Megacom® 800 telecommunications service and SDN (Software Defined Network),

which is based on the public switched network with customer-defined capabilities that reside in databases at network control points (NCPs). The ISDN (Integrated Services Digital Network) offering likewise relies on OS support for functions such as provisioning the network-element database, testing circuits, and monitoring or correlating alarms.

Advances in OS design and development will result in rapid deployment to meet new network and sophisticated service needs. Major OS productivity enhancers are standard run-time and development environments based on the UNIX® operating system. Beyond the UNIX System V standard, however, there are other *key* components and trends for the new generation of operations systems. These include:

- Modular software architectures that effectively reuse system requirements, designs, and code
- Standardized communications protocols
- Well-structured and well-articulated distributed database strategies.

Application of these concepts is evident in two OSs that are being developed to support the AT&T network and service operations. One is the Testing, Operations, Provisioning, Administration System (TOPAS) and the other is the Transport Maintenance Administration System (TMAS). These systems support switched circuit and facility networks through a wide range of provisioning, monitoring, and maintenance functions.

The software architectural principles of *modular design, information hiding (black box technique)* and *layering to achieve insulation (separation of concerns)* have been discussed in the literature for years. The strategic decisions in implementing these principles are the main focus in this paper. Emphasis is on large-scale, reusable modules and the layered or *tiered* virtual machine and database strategy. (In general software terms, a *virtual machine* is a technique that emulates all the privileged facilities of a real machine, so that a program appears to be executing with all those facilities when it really is not.[1]) The extensive subject of protocol standards will be briefly addressed in terms of an OS strategy.

Development of TOPAS was started before TMAS and is further along. Therefore, it serves as the model to illustrate the points of strategy, while TMAS provides a model for large-scale module reuse.

More Motivation for the Strategy

In the current AT&T network environment, hundreds of different types of OSs are in use, and many additional types are evident in the local exchange carrier networks. Cumulative hardware and software development, maintenance, and research costs are in the millions.

Often, OSs are database-intensive with a large portion of development dollars spent on database design. Yet populating a single instance of a database may represent an end-user investment of several million dollars. Many of these systems were developed to support a specific type of equipment, function, or service. To perform a job, a person in a typical work center may require access to the terminals of four or five operations systems. The databases for these systems often are autonomous entities, sometimes disjoint from each other, sometimes overlapping in data content.

A local area network that consolidates multiple work center terminals onto a single physical video screen, with possibly multiple windows, is not a panacea. The problem remains for a human to deal with many possible screen and data item formats, mentally or manually associating data items from diverse screens or systems. Such operations are prone to human error. Proliferation of additional systems to automate the task of consolidating OS information introduces cost and complexity that rivals those of the network itself. Additional systems only exacerbate the problem of synchronizing and reconciling databases.

Figure 1 illustrates the interconnection of a few network elements and a fleet of current OSs involved in providing the AT&T primary-rate ISDN offering. The enormity of the planning and coordination efforts for such a configuration is apparent. A significant number of inter-

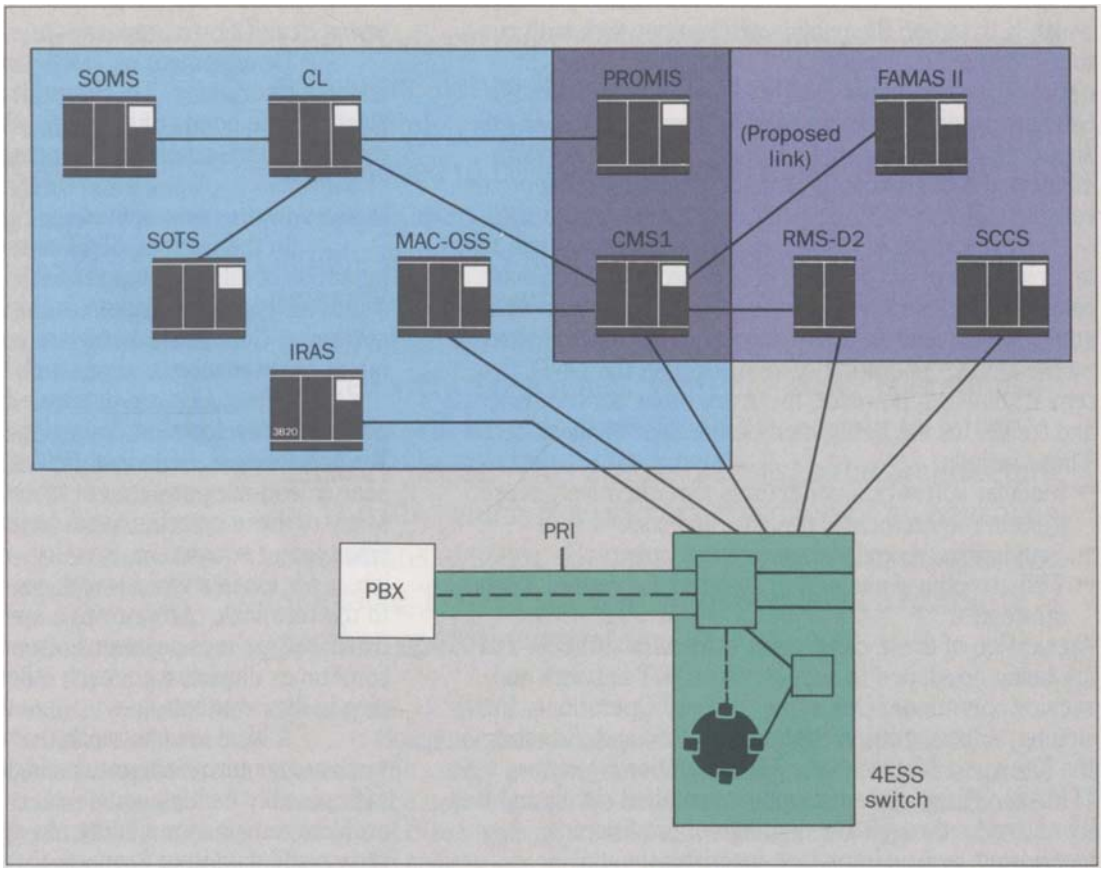

Figure 1. Operations systems planning, an AT&T ISDN example. CL = circuit layout; CMS 1C = Circuit Maintenance System 1C; FAMAS II = Facility Management Administration System; IRAS = Integrated Routing Assignment System; MAC-OSS = Machine Administration Center—Operations Support System; PROMIS = Provisioning and Maintenance Information System; RMS-D3 = Remote Maintenance System D3; SCCS = Software Change Control System; SOMS = Service Order Mechanization System; SOTS = Service Order Tracking System.

faces, both machine-to-machine and human-to-OS-machine, require modifications. For the most part, the interfaces are individually customized and involve customized software. With so many independently developed systems involved, the problem of assigning functions to each and coordinating the development and maintenance of the functions becomes intractable.

Furthermore, understanding and controlling the performance of such a system (network elements and OSs), as shown in Figure 1, become extremely difficult. The high number of interfaces to the 4ESS™ switch gives it the responsibility of handling or throttling the OS-imposed load. This is unfortunate because circuit switching is the paramount function of the 4ESS switch. Another critical consideration is the need for skilled, highly experienced planners and engineers who can comprehend the overall picture and detailed subtleties of such complex architectures as the one in Figure 1.

A significant number of OSs have a life expectancy of more than 10 years. Because migrating the hardware, software, and database components of an embedded system to new components is typically a monumental task, the

desire for a long-term OS architectural strategy is apparent. The desire—from a business, operational, and development viewpoint—is for an integrated architecture that is simple, flexible, and extensible.

Structural OS Evolution

Many older OS products still in operation today have a high percentage of customized software that runs on the OS processor. The software is intimately tied to the hardware, and usually requires software rewriting and major system retesting to accommodate hardware updates.

Recent standard OS products often use UNIX System V to help insulate the customized application software from the OS hardware. This type of configuration demonstrates the principles of separation of concerns and information hiding. UNIX System V helps separate or insulate the applications from hardware changes. It hides detailed knowledge about OS hardware device structures from the applications software.

The next generation OS goes beyond UNIX System V by providing additional standardized components, or *core*, built on the UNIX system. These components provide the advantages of concern separation and information hiding at a higher level. Figure 2 illustrates this evolutionary OS technical trend and a perspective of the layering structures.

Layering, high-level programming languages, and standardized high-level components have definite advantages but not without a price. For example, marked improvements in software quality, maintainability, and productivity must be measured against system performance considerations. Fortunately, increasing computer processing power and decreasing relative hardware costs can allay some of the drawbacks of the new software technologies. Also, understanding the performance of software components and how to model and analyze system performance is important if system performance tuning is required.

A Kit of Core Modules

During the initial system requirements phase, the TOPAS applications were examined for functionality that was determined to be generic to operations systems. This functionality was the basis for the collection of well-tested, large-scale modules called *TOPAS core*. The core is designed to reduce development costs, speed feature development, and help shield the application software from changes in interfaces to network elements or other OSs. There was large-scale software reuse in the creation of the core. In turn, the core provides a collection of components that will be useful in other operations or transactions systems.

UNIX System V provides the basic operating system environment. Another essential core component is a UNIX Transaction Management System, which is built on top of UNIX System V. Although both of these components are evolving products, they have been available for use since the inception of the TOPAS core collection. Other reused core components include the AT&T X.25 Network Interface software package and the Database Manager (DM) package.

Transaction Management System. Because it provides enhancements to and a software framework for UNIX System V, the Transaction Management System makes a more suitable vehicle for transactions systems. More important, it did not require changes to the standard UNIX operating system.

The Transaction Management System enhancements can be summarized as follows:

1. *DUX*—a database system for UNIX System V. DUX is a transaction-oriented file system that provides concurrency control and fallback and recovery.
2. *TUX*—transaction executive for UNIX System V. TUX provides the framework for the software architecture. It includes a mask-oriented input/output (I/O) subsystem and an interprocess communications mechanism, which provides logical routing of messages to application services independent of their packaging into processes.

AT&T X.25 Network Interface. This network interface is a powerful data transport product that is based on X.25

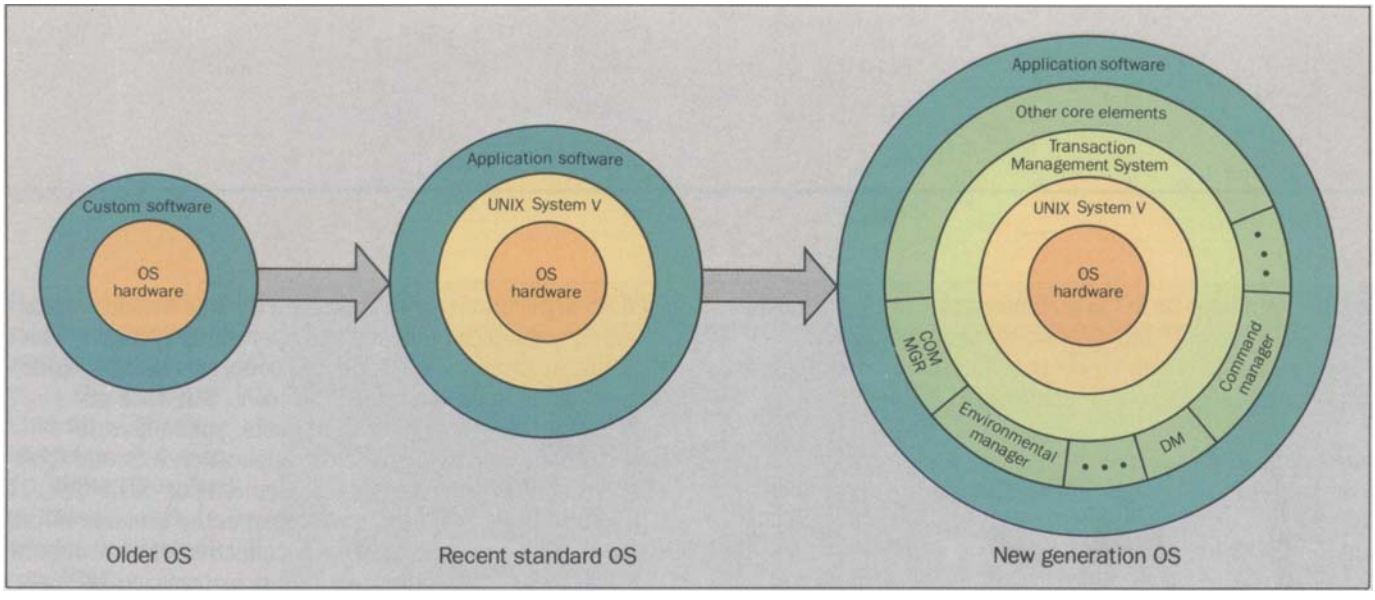

Figure 2. Structural operations system (OS) evolution. COM MGR = communications manager; DM = Database Manager.

packet-switching standards adopted by CCITT (International Telegraph and Telephone Consultative Committee). The CCITT standards include those recommended by the International Standards Organization (ISO) subcommittee that developed the standard model for Open Systems Interconnection (OSI) and data communications between networked systems.

Database Manager. The Database Manager package, which was developed for the CommView® medical-diagnostic system, runs on top of the Transaction Management System DUX record manager. It provides application software with a relational-like view of the database.

Core Components for TOPAS

The TOPAS core software carries the Transaction Management System enhancements a level higher by adding to the Transaction Management System software the functionality needed for an operations system. The added functions are generally applicable to other transaction-oriented systems. The TOPAS enhancements are summarized below.

Terminal Management. The Transaction Management System mask I/O subsystem has been enhanced to provide command language support, menu selection by cursor position, and command-level security checks.

The Transaction Management System terminal management package provides data entry and display through forms. But TOPAS offers a command language as an alternative, so experienced users can execute a command directly without first displaying a form. Both modes of entry are always available.

Another way to execute a command is available. Users can position the terminal cursor on a menu entry, and the form that corresponds to the command is displayed and opened for data entry. Access to each command and form can be restricted based on the user's privileges.

Environment Management. At the center of the TOPAS core architecture is the environment manager that has three general functions. It coordinates command activity, provides a context area for application data, and supports communication between application processes.

In addition, the environment manager helps support the implementation of a TOPAS virtual-machine concept. This is discussed further later.

Communications Management. Communications managers have been added to allow TOPAS applications software to communicate with external systems. The communications managers are designed according to the TOPAS software architecture philosophy that the applications software receives the same form of input, regardless of the source. In this way, the TOPAS communications managers allow applications software to communicate with an external system in the same way as with a terminal or another application. Currently, interfaces to the 5ESS™

switch, Remote Maintenance System D3 (RMS-D3), and Circuit Maintenance System 1C (CMS-1C) are supported.

The CMS-1C interface uses what is known as the NETCOM (network communications) manager. NETCOM provides a session and packet-layer interface to all X.25-based interfaces that use the Flexible Computer Interface Format (FCIF) syntax. [The FCIF parsing tools were taken from the SARTS (Switched Access Remote Test System) project.] In addition, NETCOM provides data transformation between FCIF and the uniform internal format that all applications software uses. All these communications managers run on top of the AT&T X.25 Network Interface core component.

As the seven-level OSI model gradually evolves as an international protocol standard, the core communications managers will undergo change. Eventually, the AT&T X.25 Network Interface component may be upgraded to a new component. The core absorbs the shock of the protocol changes and eliminates the need to modify the many lines of application code that are built on it.

System Configuration Management. The System Configuration Control supports the system operator. It provides a startup and shutdown mechanism and way for users to configure a TOPAS system to their environments.

Other Library Functions. Additional TOPAS core library functions include:

- *Application dispatcher*—supports the scheduling and execution of processes in background mode.
- *State manager*—provides a uniform mechanism for managing item work flow.
- *Error manager*—provides a uniform mechanism for reporting and recording errors throughout a software system.

Virtual Machine and Database Strategy

TOPAS is being planned to provide total operational support for all switched network circuits. The early phases include interfaces to 4ESS and 5ESS switches and support of circuits that terminate on these switches. Plans are to have TOPAS consolidate the functions of several currently deployed OSs.

Somewhat independently of the layered core structure, the TOPAS architectural strategy again invokes the principles of information hiding and separation of concerns in its relationship to the network that it supports and to the physical network elements with which it interfaces. Fundamental to the strategy is the notion of two distinct types of virtual (software) machines, and both types may reside within a TOPAS processor (a physical machine). One is called an Equipment Interface Tier (EIT) machine; and the other is called a Network Support Tier (NST) machine. Each type will have its own database.

In an EIT machine, the database is dedicated to a specific network element (NE) equipment type. For example, there might be a 4ESS switch EIT and a separate 5ESS switch EIT. EIT databases have detailed knowledge about the physical properties of their respective equipment types. The NST machines, on the other hand, have databases that do not concern themselves with equipment specifics. They deal with a higher-level or more abstracted view of the network and network-provided services.

As the network evolves and new equipment types and technology are introduced, much of the NST software can remain intact and be reused—an important advantage of the tiered virtual-machine strategy. Of course, new equipment types require new EIT modules, but OS software changes remain as confined as possible to the EIT level. A salient feature of an EIT is the potential to minimize a network element's interfaces from external systems (e.g., interfaces to the 4ESS switch in Figure 1) and to throttle external system loads on network elements.

Because the NST database views the network as an abstract mathematical graph (network), even major shifts in network technology should cause minimal perturbation to NST software and logical data structures. For example, the NST is not concerned that classical boundaries between switching and transmission are becoming blurred. It was conceived to accommodate today's circuit, wideband-packet, or hybrid switching. Even the concept of

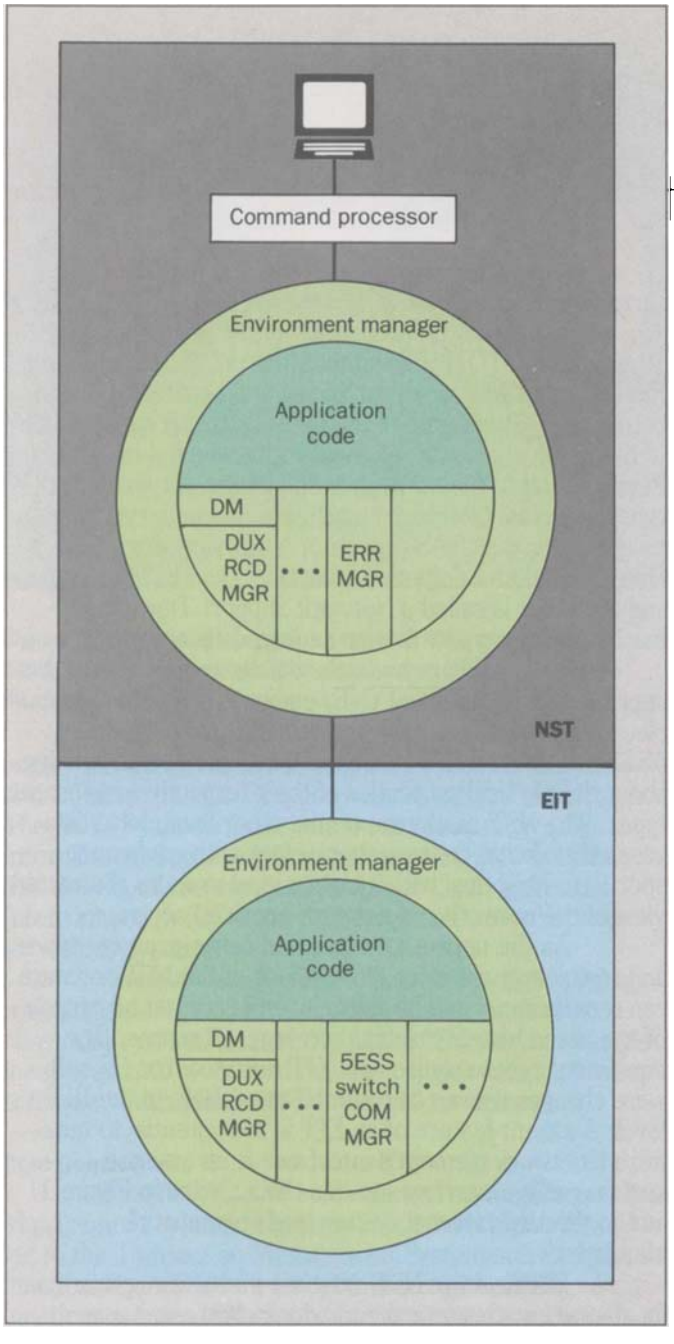

Figure 3. Structural relationship of core, applications, and NST (Network Service Tier) and EIT (Equipment Service Tier) virtual machines. COM MGR = communications managers. DM = Database Manager; DUX RCD MGR = record manager for database system for UNIX System V; ERR MGR = Error Manager.

a "transwitch" does not necessarily require NST redesign. The evolutionary merger of current *special service* circuits into the message network should not cause perturbation of the NST architecture either, because the NST database has been designed to support the classical two-ended trunk, as well as more complex multipoint circuits.

Another of the advantages of the tiered virtual-machine concept is that, if necessary, a virtual machine can be moved to a different physical processor with relative ease. However, such movement is often not trivial; not having modularized virtual machines of TOPAS would render the database movement task nearly impossible. Reasons for moving virtual machines include, among others, network element reconfiguration, service or work center consolidation, and OS processor load balancing.

One extremely important point of strategy is the use of international Common Language names to identify network elements and as linking fields between the EIT and NST databases. NST modules use Common Language key fields—along with other generic parameters that are not specific to any equipment—to send their requests for interaction with network equipment to the EIT modules that are associated with the desired equipment. Similarly, to enable network equipment to tell the appropriate NST module about changes in network status, the EIT modules use Common Language fields along with generic parameters (e.g., loss, noise, and bit-error rate).

Figure 3 illustrates the relationships between the TOPAS core, applications code, and the NST and EIT virtual machines.

Core modules execute in both the EIT and NST virtual machines. Core communication modules shield the EIT applications software from the protocol and even the application-level syntax of the computer-to-computer interface to the network elements. The job of the EIT applications software is to hide the equipment-specific data type and value from the NST modules. The EIT applications software and database also maps Common Language and generic fields that are not equipment specific into the fields and format that the specific equipment requires.

The core environment manager supports both

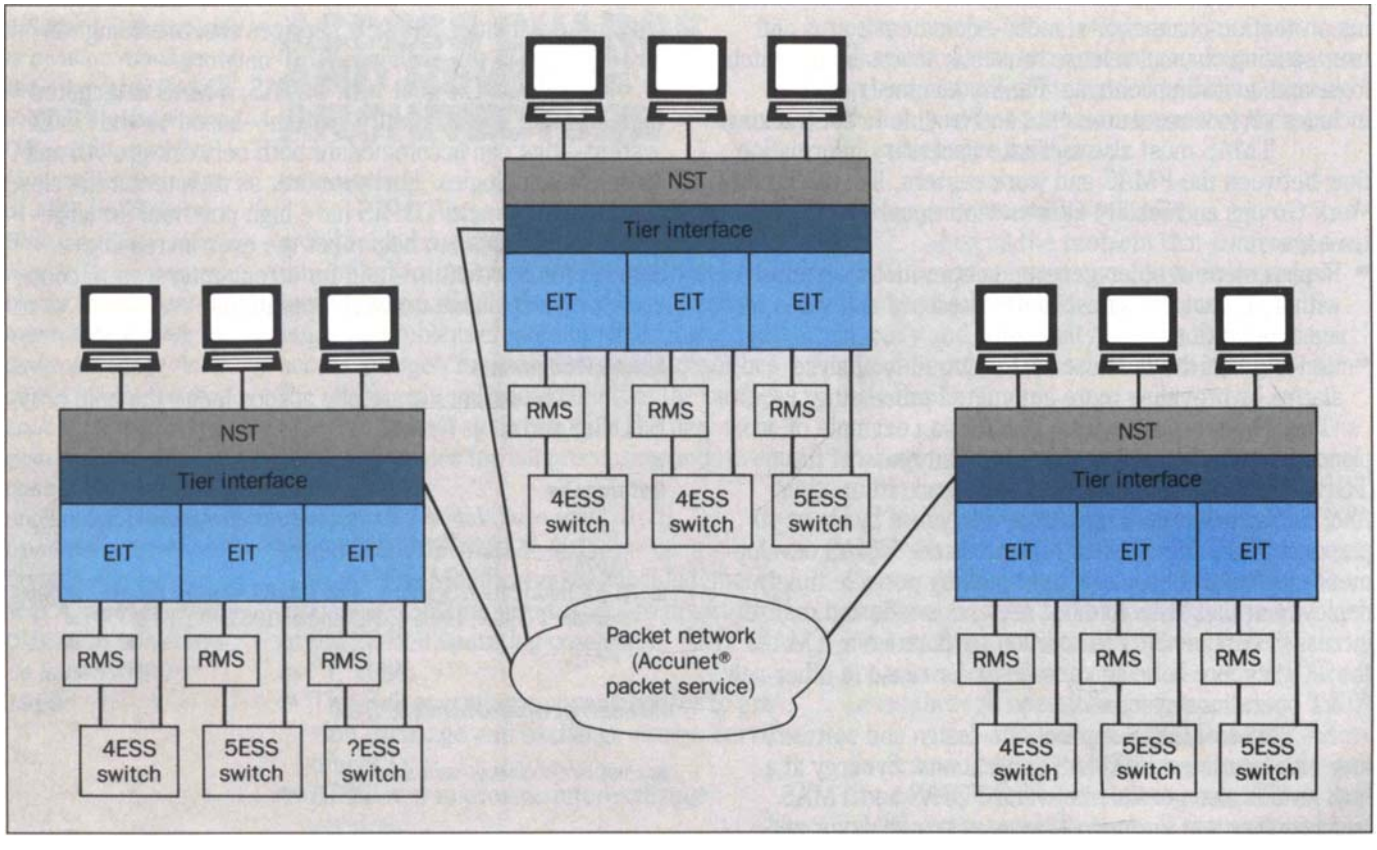

Figure 4. Design for easy remote access. EIT = Equipment Interface Tier; NST = Network Support Tier; RMS = Remote Measurement System.

intratier and intertier communications for the virtual machines. It also enables NST or EIT applications functions, which execute within a TOPAS processor, to communicate easily with NST or EIT functions in other physically remote processors. Figure 4 illustrates how this design allows network operations personnel and automated functions to access network elements that are diverse in type and geographic location. A person in any work center can have remote access to equipment at any location.

Core Reuse and Other Synergism

The goal of TMAS is to support centralized facility operations and management. A Facility Maintenance and Administration Center (FMAC) that relies on the operations system requires, among other things, automated support for facility implementation, maintenance, restoration, and administration.

Facility implementation includes database provisioning associated with new facility-route establishment and ongoing facility reconfiguration. Maintenance includes alarm surveillance and control, as well as performance monitoring and fault locating. Restoration involves monitor-

ing protection-channel or standby-equipment status and use, sending channel release requests, maintaining switch logs, and operating controls. Facility administration includes service measurements and trouble ticket tracking.

TMAS must also facilitate necessary information flow between the FMAC and work centers, like the Facility Work Groups and Facility Offices. Additional TMAS objectives are:

- Replacement of older-generation operations systems with a product that consolidates database and video terminal operations
- Reduction of the human effort required to analyze alarms by providing more automated processing.

The development of TMAS is an example of planned large-scale software reuse. The reuse of the TOPAS core, along with reuse of software from other AT&T OSs, will reduce TMAS development by about 40 percent. Thus, the effect is to shorten the TMAS development interval by about one year, making possible timely deployment of TMAS to meet network needs and realize increased cost benefit. In addition to its reuse in TMAS, the TOPAS core is being considered for reuse in other new AT&T operations systems.

Some TOPAS applications design and software may also be reused in TMAS applications. Synergy at a high level is also possible between TOPAS and TMAS because they use Common Language key fields for network equipment reference and will have symmetrical alarm report designs. For example, the DS-1 level (1.544 Mb/s, 24 channels) facility alarms that TOPAS receives from switches could be correlated with similar alarms detected by transmission equipment that interfaces with TMAS.

Conclusion

The ability to reuse software among various operations systems is essential for keeping quality high and costs low, and for reducing delivery intervals. Modular software architectures and standard AT&T operating environments provide the base to realize those benefits and deploy OSs in concert with the wide range of advanced network capabilities and new services that are being rapidly deployed in the evolving AT&T network.

As was the goal with TOPAS, TMAS is targeted to be a single, high-capacity system—based on the UNIX system—that can accommodate both network growth and diverse technologies. Furthermore, as architecturally close cousins, TMAS and TOPAS have high potential for large-scale module reuse to help meet the ever-increasing demand for new features and for strong intersystem cooperation to streamline network operations.

Acknowledgments

The authors gratefully acknowledge the help of D. E. Diller and G. J. Dome.

References

1. H. Stone et al., *Introduction to Computer Architecture*, Second Edition, Science Research Associates, Inc., Chicago, Illinois, 1975.
2. AT&T Information Systems, *AT&T X.25 Network Interface Administrator's Guide*, Issue 1, November 1985.

(Manuscript received March 5, 1987)