CTOS Revealed

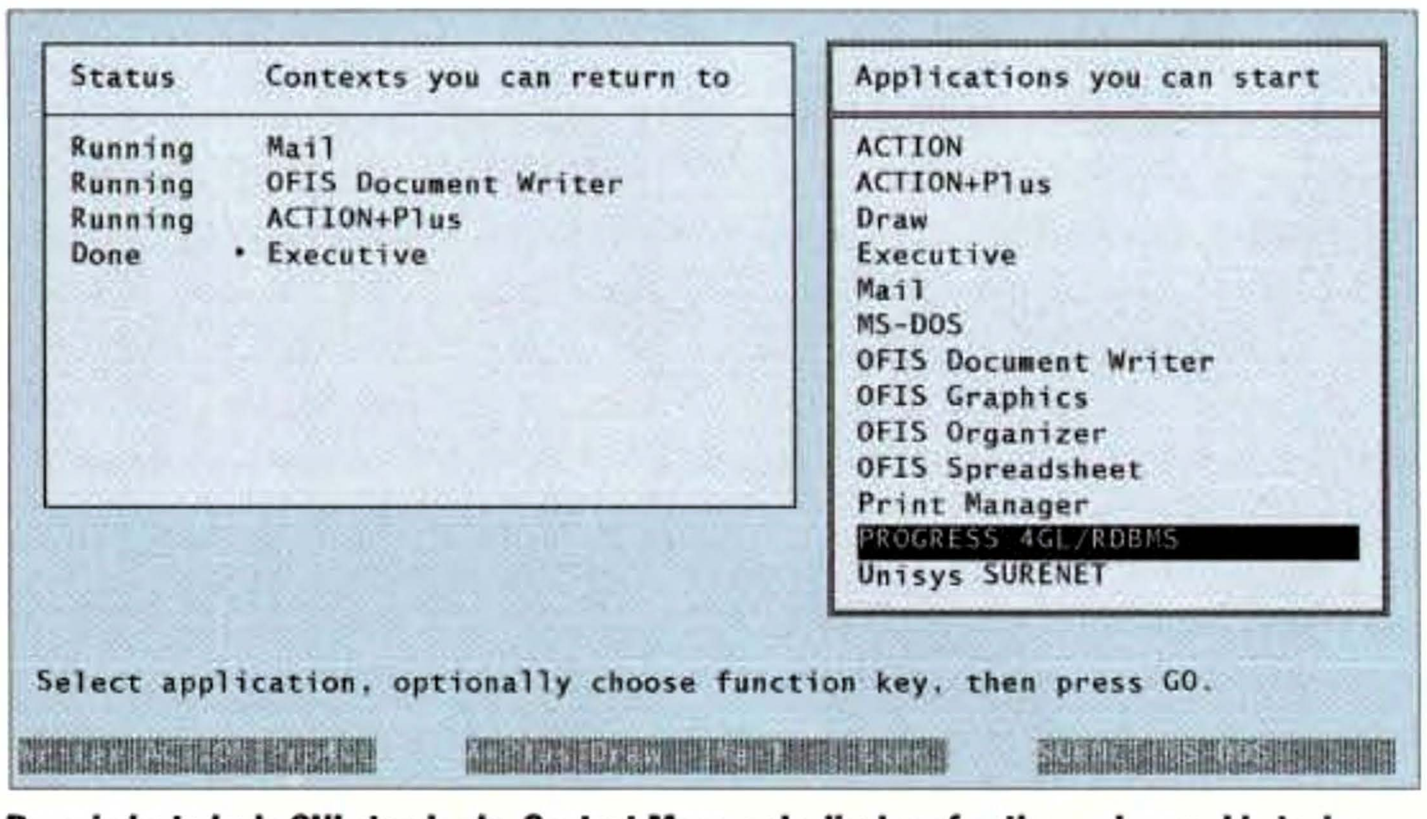
Unisys' best-kept secret is an operating system built for distributed business applications

DIRK S. FAEGRE AND JON UDELL

Operating systems all seem to be headed in the same direction. OS/2, Windows NT, Solaris, NextStep, and the Mac OS, among others, are converging on a microkernel-based, message-passing architecture. Operating systems lacking these or other desired features—including preemptive multitasking, modularity, virtual memory, long file names, built-in networking, and memory protection—are scrambling to provide them.

It's ironic, then, that CTOS (Convergent Technologies Operating System), which does all this and more, remains virtually unknown. If you've never heard of it, don't be surprised. CTOS is the invisible operating system. Its vendor, Unisys, doesn't advertise CTOS and does little to market it. But when you want to field business applications that are distributed and yet manageable, CTOS is hard to beat, as the U.S. Coast Guard, Nationwide Insurance, U-Haul, and Hong Kong Bank will attest.

CTOS was conceived and designed by a small cadre of engineers from Intel and Xerox PARC (Palo Alto Research Center). The group formed Convergent Technologies in 1979 (a time before PCs and LANs, when multitasking was the sole province of Unix and mainframes). Part of the team looked to the future with a mainframe slant. The engineers in this group gave their product the ability to multitask, address large memory, spool printed output, and dynamically recognize system resources. The other half of the team, those with Intel's vision, made sure that the system would be networked and exploit emerging microprocessor technology. Together these groups developed a product that, when it shipped in 1980 on Convergent's x86-based clustered workstations, was years ahead of its time.



```
Status     Contexts you can return to        Applications you can start

Running    Mail                              ACTION
Running    OFIS Document Writer              ACTION+Plus
Running    ACTION+Plus                       Draw
Done     • Executive                         Executive
                                             Mail
                                             MS-DOS
                                             OFIS Document Writer
                                             OFIS Graphics
                                             OFIS Organizer
                                             OFIS Spreadsheet
                                             Print Manager
                                             PROGRESS 4GL/RDBMS
                                             Unisys SURENET

Select application, optionally choose function key, then press GO.
```

Prosaic by today's GUI standards, Context Manager's display of active and runnable tasks serves the needs of the vertical markets in which CTOS remains strong.

The CTOS Message-Passing Microkernel

The tiny 4-Kb CTOS microkernel deals only with process scheduling and dispatch, as well as message-based IPC (interprocess communications). Because no other services are bound into the microkernel, it passes messages quickly—on a real-time basis, in effect. Other system and user processes invoke each others' services by passing requests (specially formatted messages) through the microkernel. The kernel reads a message, identifies its source, interprets the request, and forwards it to a resource that can respond to it. The response can come from the user's own workstation, from a local server, or from a remote server located across a WAN. In all these cases, the user is unaware of which machine responds or how it does so, because CTOS routes and responds without assistance. Likewise, the programmer doesn't need to know how to accomplish message-based IPC—it's just intrinsic to CTOS. When CTOS returns a response, the microkernel channels it to the program or utility that originally made the request.

Each CTOS workstation manages a resource table. When a program makes a disk request, for example, the microkernel checks the resource table to determine if it has a disk. If so, the local disk service handles the I/O request. But if it is a diskless client (as cluster workstations often are), the microkernel forwards the request to the cluster server for a response. In the case of a communications gateway, the server may in turn forward the request to another server in the WAN. This simple, elegant design is the foundation of a networking architecture that works, is simple to manage, and is built in. It also enables CTOS to exploit multiprocessor hardware. On a dual-processor machine, for example, CTOS can dedicate a database server or a communications gateway to one CPU, freeing the second CPU to handle all remaining chores.

CTOS System Services

Services external to the microkernel do most of the work in CTOS. This approach ensures that unused services don't eat up RAM, that each workstation can run an appropriate mix of services, and that CTOS can cleanly integrate new technologies. Services added to CTOS over the years include Posix, NFS (Network File System), SNA (Systems Network Architecture), TCP/IP, Token Ring, Ethernet, IPX/SPX, LAN Manager, NetWare, 3270 gateway, and more.

A CTOS system service, such as a GUI event loop, waits for and responds to messages. Scheduling of services is also event-driven in the sense that a waiting service moves onto the run queue only when it receives a message. Its position in the run queue depends on its priority. CTOS itself owns the highest range of priorities, followed by layered system services, and, finally, applications. Note that while CTOS does time-slice applications, the event-driven scheduling of system services means that they can yield the CPU only to same- or lower-priority processes when waiting for messages. In other words, CTOS system services multitask cooperatively. As with NetWare and its NLMs (NetWare loadable modules), it's the programmer's job to make sure that a CTOS system service uses the CPU responsibly.

Filters are special system services that can replace, monitor, or modify the behavior of existing system services. Filters that monitor existing services typically do so to collect statistics for network management or auditing purposes. Filters that modify existing services can preserve most of their functionality while altering only specific behaviors. This form of inheritance is one of the keys to the powerful extensibility of CTOS. It lends itself particularly well to various kinds of redirection. Filters can be used by a remote-access program to redirect screen writes and keyboard reads or by a router to strip the node name from an address and redirect a network message to a cluster.

CTOS Networking

CTOS first appeared and for years ran only on proprietary x86–based workstation clusters connected with twisted-pair wire. Small clusters were simply daisychained; larger clusters communicated through a hub called TeleCluster. The wire-level protocol, operating at 3.68 Mbps, resembles SNA's multidrop poll-select scheme. It's highly efficient for individual clusters, although, as with SNA, the polling can create problems on WANs.

For departmental computing, this arrangement can be a highly convenient and effective alternative to the two dominant approaches: LANs and multiuser systems. It distributes processing power as does a LAN, yet it centralizes administration as would an AS/400, Unix, or other departmental multiuser system. More recently, CTOS has shed the proprietary label. Unisys' SuperGen workstations, first shipped in 1993, are standard PCs that use ISA adapters to connect to CTOS clusters.

While a CTOS cluster looks and feels much like a server-based LAN (the workstations boot from the server, load programs from it, and share files stored on it), the entire cluster is potentially just one node in a multicluster CTOS network. Cluster servers can connect to form such networks over a variety of media, including Ethernet, Token Ring, and X.25. Only in this larger network environment must users and applications use node names to address resources. Within a cluster, the server's resources are automatically available to all workstations. Utilities for sharing workstation resources in a peer-to-peer fashion are also available.

When NetWare LANs spring up around existing CTOS clusters, as often happens nowadays, the CTOS users invariably wonder what all the fuss is about. On a PC network, misconfiguration of the server or of any workstation can cause failure, and installation of new or updated software can be a daunting task. CTOS works much more simply, in part because all workstations synchronize on a common configuration file.

Applications use CTOS's networking strengths automatically, with little or no user or developer intervention. Consider Progress, the client/server database that the Concord Group (Concord, NH) deploys in small insurance offices. On CTOS, the Progress engine runs on the cluster server as a system service. To the Progress client, which loads from the server and runs on workstations as an application, the Progress engine is indistinguishable from any other standard CTOS system service. Connectivity between the client and server pieces of a distributed application, often a stumbling block in PC LAN environments, is automatic in the CTOS cluster environment.

CTOS at Work

The solution that the Concord Group offers to the independent insurance agent running his or her own business is typically a cluster of about eight CTOS workstations. The application mix includes a CTOS-based Progress application that handles accounting, claims and policy tracking, forms printing, and marketing, as well as Unisys' CTOS-based mail and word processing programs. Because agents use a variety of rating applications from other companies, several concurrent DOS or Windows sessions may also be running—something the CTOS emulator handles reliably. There is also typically a heavy communications load—perhaps a 3270 session and one or more asynchronous communications sessions.

Nowadays, 8-MB 486 boxes are typical, with 12-MB 486s or Pentiums as servers (much less RAM would be needed to support CTOS applications only; DOS and Windows exact a substantial penalty). These systems multitask the required mix of CTOS, DOS, and Windows applications with ease. Equally important, CTOS clusters run virtually trouble-free once installed. Unisys offers a powerful management tool called CTOS InControl that enables a central site to monitor and manage branch offices. It uses CTOS messaging to relay alerts to the central office, which can then correct problems at the branch offices by remote control. However, the Concord Group has never found a need to use InControl, because CTOS installations by and large just work, and keep on working.

Unisys has ported Presentation Manager and XVT Software's XVT to CTOS, enabling development of native CTOS GUI applications. In the markets where CTOS is strong, however, character-mode applications remain dominant. While such GUI applications as CorelDraw and Wingz were ported to CTOS, Unisys' PM and XVT initiatives failed to generate much interest among CTOS developers and users who, for the most part, think that character mode is an appropriate technology and can be forgiven for seeing the Windows 95 Taskbar as a reinterpretation of the decade-old CTOS Context Manager (see the screen on page 213).

Whither CTOS? Unisys has announced a plan to integrate CTOS with NT at the server, running NT on one or more main CPUs and CTOS on a bus-mastering I/O processor board. The idea is to protect investment in distributed CTOS applications while embracing the scalable power and broad appeal of NT. Unisys calls this coexistence, but many CTOS faithful worry that it implies migration and will be watching closely to see what happens. They know how simply and reliably CTOS can work, and they don't want to abandon ship. ■

ACKNOWLEDGMENT

Thanks to Tony Lavinio, a longtime CTOS and Progress developer, for supplying valuable technical information.

Dirk S. Faegre directs microcomputer functions for the Concord Group (Concord, NH), which deploys CTOS clusters at 40 sites in Maine, Vermont, and New Hampshire. He can be reached on CompuServe at 72322,3545 or on BIX c/o "editors." Jon Udell is a BYTE senior technical editor at large. He can be reached on the Internet or BIX at judell@bix.com.