How to use COTS technology to build telecom/datacom systems with five-9s availability

By Bruce Rostowfske

The rapid advance of computer technology, increasing competition, and the growth of the Internet are forcing telecom network equipment providers to adopt a new design paradigm. Today's telecom networks were built using proprietary circuit-switching technology, but today's data networks are packet-based, and use Commercial-Off-The-Shelf (COTS) technology wherever possible. Voice and data network convergence will require a new generation of equipment that successfully merges these disparate network paradigms. Thus the race is on to develop and deploy the next generation of softswitches, network controllers, and gateways. In this article Bruce discusses the problems with the current approaches to designing and deploying COTS-based next generation equipment, and describes a solution to these problems.

Historically, telecoms have used the class four and class five switches developed by North American companies like AT&T (now Lucent), Nortel, and several European and Asian manufacturers. Although these switches achieve high availability, they are also highly proprietary, with custom hardware and software. While they are good at what they were designed to do, they are also expensive, difficult to maintain, and do not easily scale. Perhaps most significantly, these switches were built to provide circuit switching, and cannot handle the packet traffic of integrated voice/data networks. In other words, they do not support the next generation of network services.

Because of the high cost (and long delays) of developing proprietary microprocessors and operating systems for dedicated telco applications, developers of next generation switches are turning to COTS hardware and its associated off-the-shelf system software, such as operating systems. Sun SPARC and Motorola PowerPC processors have been the architectures of choice to date, but Intel is beginning to penetrate the market with the growth of Linux. The use of COTS technology leverages the product development and volume production for the enterprise sector (principally in IT applications), thus lowering the cost and speeding up deployment. However, as compelling as the benefits are, COTS technology has several drawbacks that must be considered when using it for telco, rather than enterprise applications.

The problems with using COTS technology

Telecom customers have come to expect nearly continuous availability of service – the target is 99.999% ("five-9s") availability. [1][2] This is equivalent to five minutes a year of downtime. This downtime includes both scheduled downtime (to repair faults, load software, upgrade hardware, perform periodic maintenance) or unscheduled downtime (due to a failure).

However, the majority of COTS hardware and software has been designed for enterprise applications, which typically provide no more than two-9s uptime, and were never intended to provide five-9s availability. IT departments have routine periodic maintenance and repair schedules, and it is common for their equipment to be out of service for two hours or more per week, to allow for scheduled maintenance. A single two-hour outage per year degrades availability to 99.977% – a factor of 40x worse than true five-9s. COTS high-availability software also requires highly specialized system administrators to keep the system running at even a two-9s level of availability.

COTS hardware lifecycles are also typically 18 months or less. This means that a product deployed today might be obsolete, no longer available, or even unsupported by the vendor 18 months from now. That can cause problems in the telco market where equipment life cycles are typically 10 years or more, and where replacing a piece of hardware with a "newer, faster" version every 18 months is simply not a viable option. Therefore, COTS high-availability systems must be modified:

- to provide less than five minutes total downtime per year
- to gracefully accommodate newer system elements

One common approach to achieving high availability with COTS products is with a "failover" mechanism. However, while the failover paradigm has been used successfully for certain enterprise applications, it is difficult (and expensive) to use when trying to achieve five-9s availability.

The limitations of computer industry high-availability implementations

The failover approach (which has been widely deployed for mission-critical applications in the commercial world) is conceptually simple. It typically employs two systems:

- an active system
- a standby system

If any failure or outage occurs in the active system, the backup system takes over within a few seconds to a few minutes, resulting in a minimal (but non-zero) interruption of service. The failover approach provides some level of protection against failure with few or no modifications to existing applications. However, the actual implementation of such an approach is typically very complicated. There are also several drawbacks when using the failover approach for telco applications:

- There is a high initial investment.
- Identical redundant hardware is required.
- The architecture is hard to scale.
- The monitoring and switchover process is complex.
- The finite switchover time means temporary loss of service.

The high initial investment

Failover systems typically use a 1+1 configuration, with one standby node for each active node. While this provides redundancy in the event of a failure, it also requires an investment in duplicate resources that are unproductive most of the time. An alternative configuration – called N+1 (in which there is 1 backup node for N active nodes) lowers the system cost, in exchange for a more complex failover management scheme – and a somewhat higher risk of system failure.

Identical redundant hardware requirements

From a practical standpoint, all nodes in a failover system must be identical, in order to communicate and function properly. The standby node must have:

- at least the same throughput as the node it is backing up
- the same data structure (and typically the same physical memory map) in order to properly execute the data/system state replication and switchover

This can be a problem due to relatively short COTS product life cycles, which are typically about 18 months. An exact replacement might not be available from the vendor if a failure occurs after 24 months of service. The user has two choices:

- hold inventories of discontinued products over the entire field life of the equipment, for use as spares
- replace all the nodes of a system when a failed node cannot be replaced, at considerable expense and risk

Difficulties with scaling failover systems

The use of a failover system also makes scaling difficult and costly (see Figure 1). In a 1+1 configuration (1 active + 1 standby node) all the work is handled by a single system, so the maximum performance is limited. The use of an N+1 (N active + 1 standby node) configuration (with multiple processors) does raise the performance ceiling. However, the operating system overhead involved in managing the multiple processors (combined with the additional overhead required to support failover) limits the effective number of processors to four (or sometimes eight) per system. Conventional multiprocessing techniques also greatly complicate the task of replicating the data and the system state information, which must be done to a duplicate a failed system.

Monitoring and switchover is complex

The failover model requires mechanisms to:

- monitor the health of the system nodes
- determine when a failure has occurred
- manage the switchover of all client and server processes from the active node to the standby node when a failure does occur

These mechanisms add overhead and additional complexity to the system. The fastest switchover occurs when the backup node is running "hot" (i.e., simultaneously processing the same transactions as the active node). However, this adds additional overhead for constant data and system state replication.

In addition, while the failover mechanism can be hidden from the application software, failover schemes are fastest and most effective when the application software is "failover-aware" and manages some of the failover activities directly. In practice, therefore, the application software is somewhat dependent on the implementation of the failover mechanism, and must be modified if obsolete hardware must be replaced.

Finite switchover time means lost work

It is difficult to achieve sub-second failover times – particularly when "offsite" redundancy is used to protect against a disaster (such as an earthquake) that might cause the failure of an entire physical unit – including any local redundant nodes. Traditional failover systems require the switchover of all server, application, and database activities. Thus, each failover event can cause a significant disruption, resulting in hundreds (or even thousands) of lost calls.

Parallel processing: An alternative to failover

As described above, the failover techniques used in commercial applications

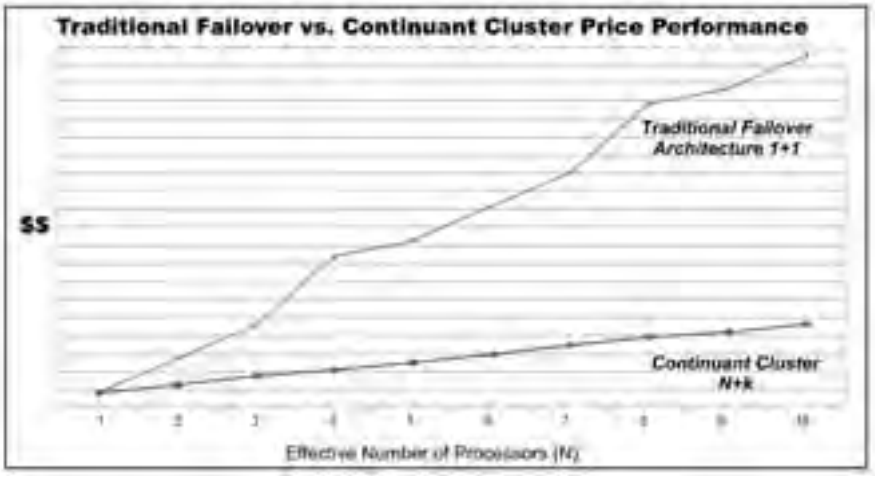

Figure 1

impose limitations when used to support telco services. However, most of these limitations (as well as the limitations of using COTS technology) can be overcome through the use of a parallel processing architecture.

In the commercial world, parallel processing has not been widely used because many commercial applications involve working on one (or a few) large processing tasks at a time, and are therefore adequately handled by sequential processing. [3] However, most telecom and datacom applications involve the concurrent processing of voice and/or data streams from hundreds (or even thousands) of separate, independent sessions. Because of this, telecom applications are better suited to parallel processing.

The Continuant Cluster Suite solution

GNP has developed a parallel processing architecture with ultra-high availability extensions which is called Continuant Cluster Suite (see Figure 2). It consists of a set of products that, collectively, provide a "true carrier class" platform with system management for telco applications. It is based on an N+k cluster configuration, where all nodes are active.

The cluster may be heterogeneous, consisting of different generations of processor chips and operating systems, or even different processor architectures (such as Sun, Motorola, and Intel) all working together in a single geographically localized cluster, or from geographically separated locations.

GNP's Continuant Cluster Suite provides a total system approach to achieving 99.999% availability by addressing the collective effects of:

- hardware
- software
- operations

The standard industry term for this is OAM&P (Operations, Administration, Maintenance, and Provisioning). For simplicity, we will use the term operations instead of OAM&P. Other approaches that include only one or two of the three areas (e.g., hardware, or hardware and software) fall short of "true carrier class" results, even if they deliver five-9s availability for those areas that they do address.

Continuant Cluster Suite includes:

- *Continuant Cluster*, which is middleware that provides a continuous-availability infrastructure used COTS technology. It also provides seamless (or nearly seamless) scalability, and allows geographically separated nodes to function as a single cluster.
- *Continuant System Manager*, which provides a single, comprehensive view of the entire system, allowing easy system monitoring and management.
- *Continuant Watchdog*, which is hardware that provides a standardized interface for out-of-band monitoring, as well as management of individual system elements.

While these three elements are designed to provide a comprehensive solution when working together, they can also be used independently. Each is described in further detail below.

The Continuant Cluster

The Continuant Cluster uses a 20-year-old parallel processing approach known as *Linda*. Simply put, it uses a *pool-of-work* (a.k.a. *a-bag-of-tasks*) model in which data (the *work* or the *tasks*) is placed into a virtual shared memory space (called the *bag* or *pool*). Each processor in the cluster pulls data out of the shared memory, performs the required unit of work, and (if necessary) puts the result back into shared memory. It then repeats this process.

This approach is simple and robust, since interprocessor communication is asynchronous and anonymous (there is no need for processes or processors to identify each other for interaction). It has been successfully applied in a wide variety of specialized applications such as processing high energy physics data, and real-time pricing of financial instruments, such as options.

However, when the bag-of-tasks approach was originally conceived it was optimized for high throughput and computational accuracy – not for high availability. Therefore it is not ideally suited for use in telco applications. GNP has designed and implemented extensions to Linda that optimize availability in a telecom environment. These patent-pending extensions, are the basis for GNP's Natural Clustering Technology (NCT), and form the heart of Continuant Cluster.

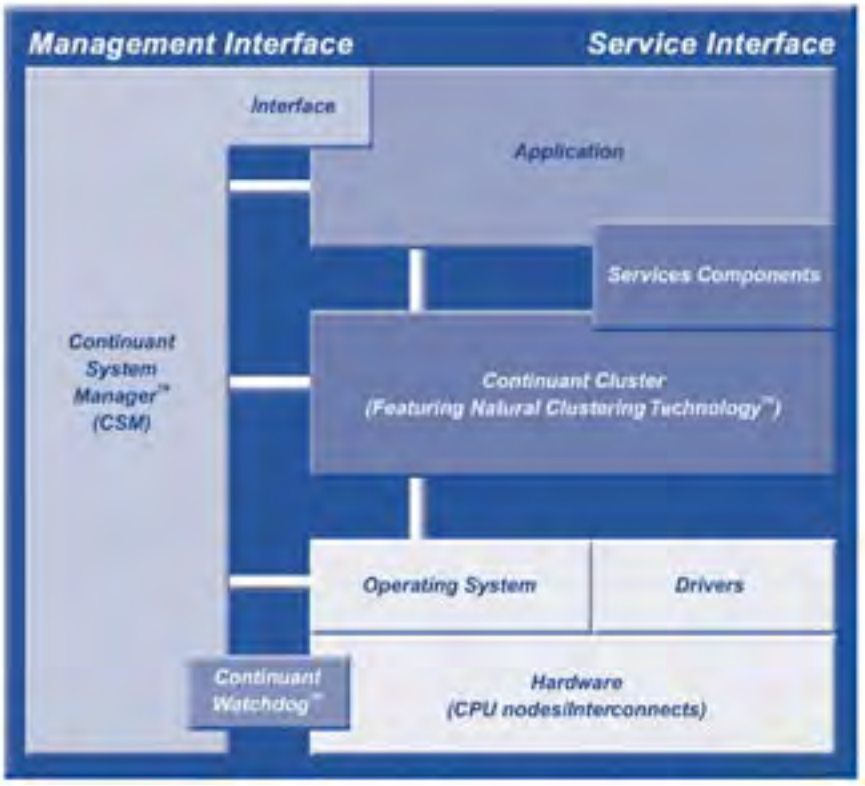

Figure 2

How Natural Clustering Technology works

Natural Clustering Technology allocates a portion of each processor's memory to a virtual shared memory space, as shown in Figure 3. This virtual shared memory is generally referred to as "tuple space." Each "tuple" (that is, each ordered set of data fields) is a single piece of work that contains a sequence of data elements that might include:

- data values
- information about the values, including type tags such as int32
- information about the operations to be performed on that data

Processes and applications access this tuple space using five basic functions, which can be called from C, C++, or Java:

ts_out(tuple)	write
ts_in(tuple)	destructive blocking read
ts_inp(tuple)	destructive non-blocking read
ts_rd(tuple)	non-destructive blocking read
ts_rdp(tuple)	non-destructive non-blocking read

Message processing applications (such as those based on SS7, H.323, SIP, or MGCP) are decomposed into a stream of *transactions* that consist of:

- input data
- a unit of work
- resulting output data

Each transaction within this stream can then be processed by any of the nodes in the cluster. At any given time, tuple space contains:

- data associated with the activation of a new application
- intermediate products of applications already in process

Any processor can read a tuple from tuple space, process it (i.e., perform the work associated with that data), and return the results. Any available processor can then read the result, process it, write that result back into tuple space, etc. The tuple space therefore represents a "pool of work" that is available to any processor. "Executing an application" consists of progressively processing and updating the tuples.

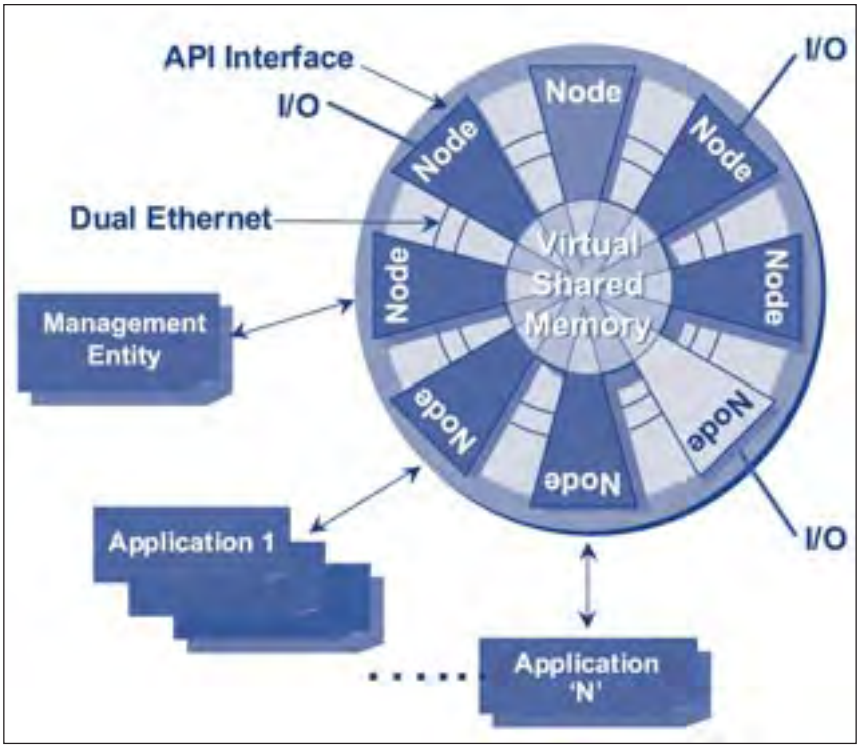

Figure 3

All tuple space transactions have degree-two ACID properties, which are:

- Atomicity, meaning that grouped operations either all occur or do not occur, as a unit.
- Consistency, meaning that completion of a transaction leaves the system in a consistent state.
- Isolation, which means that concurrent transactions do not affect each other.
- Durability, which means that the result of a transaction is at least as persistent as the entity being operated upon.

Depending on the cluster configuration, the tuple space resides on one or more processors, and a copy is also maintained on one or more backup servers. NCT provides a *Transaction API* that (in the event of a node failure) performs a rollback of a failed transaction, and then allows another node to process the information.

NCT uses a main memory database to provide tuple space. In the event that data needs to be persistent, NCT provides a *Persistent* attribute. This creates a RAID 1 equivalent image of the persistent tuples in main memory, in addition to the images on the local disks of multiple computers within the cluster. In addition, for permanent or near-permanent data (such as billing information or lookup tables) NCT will interface with external commercial databases, such as Oracle.

The nodes communicate among each other using standard Ethernet connections. Ethernet is reliable, inexpensive, and fast enough to support most cluster configurations. As they become available, NCT will support faster interprocessor communications technologies including Gigabit Ethernet as well as the new switched system interconnects (cPSB, Infiniband, RapidIO, and StarFabric). NCT also allows synchronization across a WAN for purposes of geographic redundancy.

The advantages of Natural Clustering Technology

The architecture and implementation of NCT provides many inherent advantages for telecommunications applications:

- There is less overhead for task management.

- The application software does not need to support failover.
- Data processing is done in a parallel manner.
- Heterogeneous environments are supported.
- Scalability is seamless.
- Software and hardware maintenance are facilitated.
- Less code is required for scheduling and load balancing.

Reduced overhead for task management

With NCT, each node pulls its own work from tuple space. This eliminates the need for a management process (or layer of software) to assign each piece of work to each node.

The application software does not need to support failover

With NCT there is no need for a failover procedure when a node fails [4] because every node in the cluster is active and is continuously pulling its own work. If one of the nodes fails, it simply stops taking work and the workflow continues across the other nodes. One failed node therefore does not impact the system as a whole.

The amount of work handled by the other nodes might rise. However, because each takes work from the pool at its own pace, there is a natural balancing of work across the cluster. As long as there is sufficient reserve capacity in the cluster, each of the processor loads increase according to their capacity. (Consider also the fact that in NCT the processing power of all nodes is available while, in an N+k failover configuration, only N nodes are active at any given time.)

Data processing is done in a parallel manner

NCT inherently provides parallel processing. This fits well with most telecom applications, which require the simultaneous processing of many independent voice and data streams.

Support for a heterogeneous environment

Using NCT, any processor with idle capacity can pull work from tuple space using the NCT API. The processors in the cluster do not need to be identical. They might be different generations of a particular architecture (such as a 270 MHz UltraSPARC IIi running Solaris 7, or a 500 MHz UltraSPARC IIe running Solaris 8) or even different architectures altogether (such as SPARC nodes, PowerPC nodes, and Intel nodes).

Since additional processors (and the associated system software required for expansion or replacement) need not be identical to the original equipment, the problem of matching short COTS product life cycles to the long deployed life cycles of telecom equipment is resolved. For example, if a Sun SPARC processor breaks-down after two years, and if Sun no longer supports that particular version of the processor, there is no need to replace all of the cluster processors with the latest version. A newer version of the processor can be incorporated into the cluster of older processors.

Contrast this with a failover system. If some processor in the cluster fails, and if all of the processors need to be upgraded, the codependency of the hardware and software might dictate an upgrade to the operating system, as well as rewrites of any application software that participates in the failover process. With NCT, only the processor that failed is replaced. The operating system might need to be upgraded. However, the application software can remain the same.

Seamless scalability

Since the "pool of work" algorithm requires no administrative overhead, additional processors simply expand the pool of resources available to draw work from tuple space. NCT scales easily and gracefully and (in theory) can expand to an infinite number of processors. In practice, a practical limit is imposed by the bandwidth of the interprocessor communication. Also, if only certain nodes handle the I/O, the I/O might become a bottleneck.

Facilitation of software and hardware maintenance

NCT's ability to scale in a heterogeneous environment is a tremendous advantage when performing maintenance and upgrades. Just as nodes can easily be added when scaling, one or more of the nodes can be removed from service temporarily for upgrades and/or maintenance with minimal impact on the remainder of the cluster.

NCT's support of heterogeneous environments also allows online upgrading of operating systems or other system software. Even application software can be upgraded, as long as it does not change the data structures used within the system. In the event the data structures must be changed from one version of the application to the next, an online upgrade is still possible, but it requires special coordination to ensure zero downtime.

Less code is required for scheduling and load balancing

The simplicity of NCT significantly reduces the amount of code needed to achieve 99.999% availability. NCT's pool-of-work concept (which is supported by tuple space) enables the nodes in a cluster to share the workload without the need for any extra code for scheduling or load balancing. This has significant implications for the system developer, since code normally developed to handle these functions can be completely eliminated. Thus development costs and time-to-market are dramatically reduced, while the quality and the reliability of the application are improved.

The Continuant System Manager

The Continuant System Manager is the second component of the Continuant Cluster Suite. It is a suite of tools designed to manage the Continuant Cluster Suite and any associated components. It provides a complete operations framework designed to eliminate non-hardware-based failures within the system, and to automate most of the system administrator's routine tasks. It features a high-level, unified system management view across all nodes (and across all geographically dispersed sites) enabling a user to monitor the entire network from anywhere in the world – regardless how many nodes exist, and at how many sites.

Most importantly, the Continuant System Manager supports successful recovery from the inevitable craft or administrator induced crashes. Continuant System Manager is "craft-friendly" in that it prevents (or at least facilitates graceful recovery from) errors that may be made by field personnel, such as:

- typing in the wrong commands
- pulling out the wrong node
- improper shut down

If a node goes offline, the cluster continues to function using the pool-of-work model, and the system does not crash. Continuant System Manager can be thought of as an embedded systems administrator that works 24/7 on the network, eliminating the need for dedicated human system administrators.

Continuant System Manager consists of the following logical elements:

- a recovery facility for UNIX
- system and application data administration
- a process monitor
- a logging and engineering manager
- a fault recovery/reconfiguration manager
- a display manager
- a diagnostic manager
- an update manager

The recovery facility for UNIX

This facility protects a stock version of UNIX (e.g., Solaris or Linux) against improper shutdown. In the event of improper shutdown (such as the craft improperly removing an active processor) the node reboots to a standard, known state, after which it is automatically configured (based on the needs of the cluster) and then reintroduced into the cluster. This eliminates the need for manual system repair following an unexpected event.

System and application data administration

This facility monitors both the system and the application data, detecting any changes that might affect the system. It also provides a set of tools for managing the data, and logs any changes that are made.

The process monitor

The process monitor tracks the health of cluster processes, as well as non-cluster applications that are important to maintaining service. In addition, it also manages heartbeats and failover for applications that are not part of the cluster processing.

The logging and engineering manager

The logging and engineering manager provides the capability for logging system-wide information, collating it, and presenting it in a unified system view, as well as tracking the actual performance of the cluster and application.

The fault recovery/reconfiguration manager

The fault/reconfiguration manager uses a customizable set of rules and instructions to detect faults and to optimize the system configuration after changes. It also monitors the network to ensure compliance within a set of preset conditions and restrictions, and it has the intelligence to dynamically reroute around failures, and to integrate new hardware as it is added. For example, if one of the nodes goes offline the fault/reconfiguration manager will attempt to restart it. Once restarted, the node will be automatically reconfigured, and then (if it is functioning properly) it will be re-introduced into the cluster.

The display manager

The display manager is a set of basic tools and reference designs for building graphical and command line interfaces for the cluster, and for cluster applications. Figure 4 shows a sample of a GUI designed with this toolset.

The diagnostic manager

The diagnostic manager uses a set of intelligent agents that are capable of isolating faults within the system, and then works with the fault recovery/reconfiguration manager and local craft to repair the failures.

The update manager

The update manager handles the roll-out/roll-back for software, operating system patches, and system policies for the entire cluster.

The Continuant Watchdog

The final piece of Continuant Cluster Suite is called the Continuant Watchdog. It is a PC board that provides an out-of-band remote control and maintenance framework that is connected to all hardware components in the cluster. It can be configured to manage any COTS component.

The Continuant Watchdog pairs an out-of-band hardware component with each component in the system. In doing so, it provides the necessary interfaces to allow a monitoring system (or a user) to completely manage a networked computer system. Through a single-port terminal server to each console port, the Continuant Watchdog allows remote monitoring, configuration, and provisioning capabilities. It can even perform a hard reset on an unresponsive (or worse yet, a "babbling") node. The Continuant Watchdog also interfaces with existing Central Office alarming schemes.

Conclusion

In the race to build the next-generation voice and data communications infrastructure, COTS technology will enable telecoms to quickly (and cost effectively) develop and deploy the latest soft-switches, network controllers, and gateways. However, COTS technology alone will not allow telecoms to achieve 99.999% availability. Traditional failover techniques offer some degree of high availability, but as telecoms continually expand their infrastructures, they will find themselves burdened by the limitations and the overhead associated with conventional failover models.

The Continuant Cluster Suite provides a new approach to building a "true carrier class" infrastructure – one that is tailored to the five-9s availability and the expansion needs of the teledatacom market. The Continuant Cluster Suite combines the simplicity, the scalability and the robust nature of parallel processing with GNP's extensions, which are designed to optimize availability in a telecom environment.

The Continuant Cluster Suite's management component (Continuant System Manager) and the service component (Continuant Cluster) together with Continuant Watchdog, provide a robust solution with a single point of management for the entire network. Although designed as a comprehensive solution, each of the Continuant Cluster Suite elements can be used independently, and integrated not only with COTS elements, but also with any telecom's proprietary hardware and/or software.

The Continuant Cluster Suite plays a critical role in providing telecom network equipment providers with the 99.999% availability required to compete in the race for next generation data and voice convergence.

Notes:

[1] This article uses the phrases "99.999% availability," "five-9s," and "carrier class" interchangeably. Availability here refers specifically to "application" availability, which is dependent on hardware, software, as well as operations. Therefore, it is a more restrictive definition than "system" (i.e. hardware, or hardware plus software) availability.

[2] The term "availability" here refers specifically to "application" availability, which is dependent on hardware, software, as well as operations. Therefore, it is a more restrictive definition than "system" (i.e. hardware, or hardware plus software) availability. [1] The phrases "99.999% availability," "five-9s," and "carrier class" will be used interchangeably in this article.

[3] Most commercial applications could be written to support parallel processing more extensively. Parallel processing has been underutilized due to the conventional approach to programming, which results in sequential execution of applications.

[4] There is a failover mechanism applied to server-based functions within NCT, the primary function being management of tuple space. However, since much of the application is executed as client processes, for which there is no failover, an NCT failover event is less disruptive than that within a traditional "failover" system, in which all client as well as server processes must be switched. In NCT, the TSS failover time is a parameter that can be tuned to account for client applications' characteristics (e.g., geographic separation, network traffic, etc.)

Bruce Rostowfske *is the Chief Technical Officer at GNP, where he helps ensure that the company continually develops and upgrades leading edge solutions that keep GNP at the forefront of the developing convergent communications industry. Rostowfske is the lead designer for GNP's Continuant Cluster Suite, a multi-node computer operating environment for use in telecommunications switching applications. During his six years at GNP, Rostowfske has also developed controller solutions for embedded network applications in wireless and Internet markets, for such companies as Lucent, Nortel, QUALCOMM, and BellSouth. Prior to joining GNP, Rostowfske served at AT&T Bell Labs as a Switching System Architect. In this role, he supported major Central Office switching system design activities. During his 10 years at AT&T, he headed the development work on the company's 4ESS switch, supported the planning and development of major hardware and software upgrades, and helped to introduce SS7 control technology for AT&T and RBOC networks. Rostowfske was also a member of AT&T's Root Cause Analysis team, which was responsible for investigating network failures, and participated in studies of PSTN/Packet network integration. Rostowfske served in the United States Air Force for five years. He received a BS degree in Computer Science from the University of Wisconsin, and an MS in Computer Science from George Washington University.*

If you would like further information about the NCT API or programming with NCT, you can request the *Natural Clustering Technology Developer's Manual* from GNP at:

GNP Computers
555 E. Huntington Dr.
Monrovia, CA 91016
Tel: 626-305-8484
Fax: 626-305-9177
Web site: www.gnp.com

Figure 4