Much of the progress in high availability applications these days is taking place in the Linux world where the nature of the open-source development paradigm provides a fertile proving ground for many new ideas. Interest in Carrier Grade Linux, in particular, is increasing, and acceptance of the Linux OS in telecommunications equipment designs that require high availability was one of the key areas of growth in 2002. One of the main reasons for the increase in acceptance is the fact that Linux is being optimized to meet the carrier grade requirements of modern telecommunications carriers.

Linux in telecommunications

Three main categories of applications for which equipment providers are targeting Linux include gateways, signaling servers, and management servers.

Most applications fall into one of the following categories:

- Gateways are bridges between two different technologies or administration domains. For example, a media gateway performs the critical function of converting voice traffic from a native telecommunications time-division-multiplexed (TDM) network to an Internet Protocol (IP) packet-switched network. A gateway maintains a large number of connections in real time over a large number of interfaces without losing a frame or packet. Gateways are implemented on dedicated platforms with replicated (rather than clustered) systems for redundancy.

- Signaling servers handle call control, session control, and radio recourse control. A signaling server handles routing and maintains the status of calls over the network. It takes the requests of user agents that want to connect to other user agents and routes these requests with the appropriate signaling. Signaling servers require soft real-time response capabilities of less than 80 msecs and may manage tens of thousands of simultaneous connections. Due to requirements for quick switching and a capacity to manage large numbers of connections, a signaling server application is context-switch and memory intensive.

- Management servers handle traditional network management operations as well as service and customer management. These servers provide services such as home location and visitor location registers for wireless networks, or customer information such as personal preferences or authorized features the customer can use. Typically, management applications are data and communication intensive. Their response time requirements are less stringent by several orders of magnitude compared to those of signaling and gateway applications.

Carrier Grade Linux standards

Open standards are a key reason why equipment providers are moving toward Linux-based solutions. Creating platforms based on open standards ensures interoperability with...
third-party software and makes maintenance and application development much easier. Therefore, utilizing the standard Linux kernel and adhering to key Linux standards, including the Linux Standards Base (LSB), is essential.

There are many standards-related activities in the industry to define hardware and software high availability:

- The PICMG group is defining them for high availability hardware.
- The Service Availability Forum (SA Forum) is focusing on APIs for hardware platform management and for application failover in the application API.
- The Open Source Development Lab (OSDL) is defining specifications for Carrier Grade Linux.

One of the assumptions in this Carrier Grade Linux specification is that Linux is capable of being an embedded real-time Linux. This is a key reason why Carrier Grade Linux, rather than a typical workstation or server Linux, is the preferred choice for telecommunications solutions.

Linux Standards Base

The Linux Standards Base (LSB) defines a system interface for compiled applications and a minimal environment for support of installation scripts. Its purpose is to enable a uniform, industry-standard environment for high-volume applications conforming to the LSB.

The LSB defines source and binary interfaces for application programs compiled and packaged for LSB-conforming implementations on many different hardware architectures. Since a binary specification must include information specific to the computer processor architecture for which it is intended, it is not possible for a single document to specify the interface for all possible LSB-conforming implementations. Therefore, the LSB is a family of specifications, rather than a single specification.

The LSB is composed of two basic parts: A common specification, generic LSB or gLSB, describing those parts of the interface that remain constant across all implementations of the LSB, and an architecture-specific specification, archLSB, describing the parts of the interface that vary by processor architecture. Together, the generic LSB and the architecture-specific supplement for a single hardware architecture provide a complete interface specification for compiled application programs on systems that share a common hardware architecture.

Service availability platform interface specification

The Service Availability Forum is a consortium of communications industry leaders and innovative startups dedicated to producing standards to enable development of carrier-grade communications systems from off-the-shelf hardware platforms and middleware. The key characteristic of a carrier grade system is its ability to provide uninterrupted user access to the services it is designed to deliver, with no loss to the continuity of those services. To meet this goal, the Service Availability Forum is developing two layers of standard, carrier-grade interfaces: an application interface and a platform interface.

The Service Availability Forum Application Interface provides access to a standard set of tools for application software to use to distribute processing over multiple computing elements, and to respond to failures of those elements without loss of service delivery or continuity to any user. Management middleware that conforms to the Service Availability Forum Application Interface provides these tools. Specification middleware that conforms to this specification is called service availability middleware.

Service availability middleware provides these tools to application software in part by monitoring and controlling the physical components of a high-availability computing platform via the Service Availability Forum Platform Interface. By using a standard interface to manage the physical platform, developers can write the service availability middleware independently of any particular hardware. This independence, in turn, allows application developers to choose the best hardware platform and the best service availability middleware to fit their needs.

Carrier Grade Linux Specification

The OSDL Carrier Grade Linux Working Group formed in January 2002. It consists of several equipment providers including Nokia, Alcatel, Cisco, Ericsson, and NEC, and platform providers such as IBM, HP, and Intel. MontaVista Software, SuSE, and Red Hat are Linux distributors that are also part of the Carrier Grade Working Group. These companies are committed to developing and promoting a common standard definition for Carrier Grade Linux. The specification promotes portability, ease of programming, and software availability for telecom developers looking to implement Linux in an equipment design. This standard focuses on high availability, performance, and adopting new leading edge technologies that promote high availability and service ability. Additionally, the working group hopes to develop standards that make it easier to avoid problems in coding, and thus improve the reliability of systems. These standards will ensure that companies have choices for Carrier Grade Linux but that all of them will meet the specification, will support a rich set of high availability features, and will have consistent interfaces and functionality.

“...Carrier Grade Linux, rather than a typical workstation or server Linux, is the preferred choice for telecommunications solutions.”

Some of the standards on which the OSDL Working Group is focusing are LSB compliance, IPv6 (including IPSECv6 and MIPv6) compliance, SNMP support, and POSIX interface compliance in the areas of timers, signals, message queues, semaphores, event logging, and threads.

High availability

One important characteristic of highly available systems is redundancy of key subsystems. A highly available system should include redundant Ethernet to ensure constant networking connections and disk mirroring to ensure high levels of data reliability. Carrier Grade Linux also includes an enhanced kernel with hardened device drivers and fault response behav-
ors, such as panic handler improvements, that ensure the application logs appropriate messages and sends notifications before a kernel panic.

**Hardware redundancy**

Hardware redundancy is commonly achieved by deploying CompactPCI chassis that support failover and hot-insertion and removal of field replaceable units. CompactPCI and its successor, AdvancedTCA, are preferred platforms for carrier grade solutions. CompactPCI offers the most flexibility in terms of processor choice and overall architecture. From an application software perspective, the choice of form factor is often transparent. The Linux OS kernel, drivers, and interfaces encapsulate support for CompactPCI. Linux is very much at home on these architectures, and solid commercial ports are available for the embedded architectures represented thereon, specifically IA-32/ x86, PowerPC, and sometimes MIPS. The same cannot be said of Windows and most RTOS platforms. Carrier Grade Linux requires full support of PICMG 2.12 and full device enumeration.

When defining the high availability platform, the main requirement is to support hot swap, remote boot, diskless operation, and *headless* operation (no console). The Carrier Grade Linux definition of hot swap also includes the concept of hot insert (adding cards to the system not originally in place at boot time), hot remove (not replacing cards that are removed), and identity maintenance (maintaining device identities across hot swaps and system boots). Today, different implementations of embedded Linux support these features.

**Redundant Ethernet**

Redundant Ethernet refers to a driver, the bonding driver, which permits grouping of LAN connections to provide for link failover, link throughput aggregation, or both. All links in a bond have the same Ethernet and MAC addresses. Link failover refers to the process by which a failure of one component of a redundant link can be transparently masked by having its traffic routed to another link. The bonding driver accomplishes this task by continuously monitoring all links available to it and removing any links from service that are no longer operative. With the proper sort of switch, the system can aggregate the redundant links to provide better throughput, but it can offer failover operation with any class of switch.

**Other availability features**

Some other high availability challenges Carrier Grade Linux has met are:

- Flexible options for booting compressed and remotely hosted kernel images
- Support of compressed r/w and read-only Flash file systems (JFFS and CramFS)
- Accelerated boot and daemon start times from several minutes to seconds, speeding shutdown and eliminating costly file system operations with journaling file systems

Carrier Grade Linux supports these requirements with:

- Boot from multiple media – from local Flash, via Ethernet, or over the CompactPCI backplane
- Flexible boot image – how much local Flash and RAM is available (Linux is smaller than you imagine!)
- Root file system – without local rotating media (disk), Linux/bin, and other key directories can mount with NFS or on local RAM or local non-volatile memory
- Rapid reboot and fast boot time – reducing boot time of CPU boards/blades after hot insertion
- Reduced halt time – during a hot removal, reduce the time that the system needs to shut down, sync file systems, etc., before enabling the Blue LED
- Journaling file systems for faster reboot

**Performance and scalability**

Carrier Grade Linux, with its requirement for soft, real-time performance, is the ideal operating system for telecommunications systems. Some key features include a preemptible kernel with millisecond-level real time (less than 5 msecs worst-case latencies), RAID 0 (striping), application pre-loading, and a scaling analysis and report to identify bottlenecks.

**Real time**

Real-time performance is critical to many carrier grade applications. The ability to respond quickly and predictably to external events is a key feature of availability.

The preferred method for implementing soft real-time support in embedded Linux is with a preemptible kernel, which redefines the Symmetric Multiprocessing (SMP) spin-lock protection regions as non-preemptible regions in the kernel. Since concurrent use protection issues for SMP and uniprocessor kernel preemption are the analogous, drivers or kernel modules that are SMP safe are also safe for use with a preemptible kernel.

This technology has the advantage that any time the flow of execution is not in a protected region in the kernel, the current context becomes immediately preemptible. In addition, work by the Linux community to reduce the number and length of spin-lock protected regions, so-called *big locks*, helps to reduce the length of non-preemptible regions in the kernel, and so reduce scheduling latency.

**High resolution timers**

System time in standard Linux is kept in terms of *jiffies*. A jiffy is the period of the system clock, usually 10 msecs, although it can be as small as 1 msec on some architectures and is configurable on others. For timer or alarm resolution of less than 2 jiffies, an alternative to standard Linux timer mechanisms is required, since timer accuracy is by definition ±1 jiffy. Some embedded Linux solutions provide higher-resolution timers via alternative time bases – timers with microsecond resolution. This solution also allows for an increased maximum
number of timers in the system by reducing the amount of overhead timers placed on the system from native timer interrupts and jiffy calculation.

**Service ability**
Features in embedded Linux that support service ability include tools for resource monitoring, kernel crash dump and analysis features, and online application patching and debugging. Other kernel-level features include structured kernel messages, dynamic kernel probing, hardware error logging, and remote access to the event log. Figure 1 depicts the flow of a typical resource monitoring subsystem.

**Resource monitoring**
The Carrier Grade Linux specification will provide a publish-and-subscribe resource for monitoring API usage and tracking kernel gauges. Example kernel gauges include Ethernet traffic, free memory pages, processes created, number of zombies (terminated but still present processes and threads), and current kernel load. New gauges are easy to add to a system for tracking other information, and gauges may be self-discovered by user-level software by using the unique gauge-string identifier.

Monitor options include high, low, and leaky bucket monitors. Using an API, system developers can keep track of system resources to help diagnose problems and provide preemptive remedial action before the system becomes unusable.

**Dynamic probes**
Dynamic kernel probing allows designers to examine data structures and establish execution break points without the requirement of compiling debug code into the kernel itself. Kernel probing allows developers to observe system behavior easily and quickly from a debugger in a running system, without having to recompile modules under observation with debug capabilities beforehand, or indeed even having to decide in advance which modules may need to be probed. Dynamic kernel probing also allows system maintainers to examine a deployed system in situ without needing to bring it down.

**Application patching and debugging**
Some Carrier Grade Linux implementations offer a Field Safe Application Debugger (FSAD). The FSAD lets developers debug an application while it is running on a live system in the field. Non-intrusive, FSAD does not cause significant runtime performance degradation in the application, and thus the application may safely run with FSAD while under load and performing normal operations. Many bugs that occur when an application runs in the field cannot be easily reproduced in a lab setting. FSAD allows debugging of a deployed application, letting the user set trace points (which are like break points, but only record information and do not stop the application) in code and record execution information.

"Carrier Grade Linux, with its requirement for soft, real-time performance, is the ideal operating system for telecommunications systems."

**Crash dump and analysis**
Kernel dump and go is required for many carrier grade systems. The dump must provide as much information as is practical, and then the system must be brought back into service as soon as possible. Some embedded Linux platforms provide a suite of tools that allows the developer to configure the scope of state information in a kernel dump, up to the full contents of registers and all of memory. Dumps go to a reserved location on the disk. This solution also provides a crash analysis tool to help a post-mortem analysis of the dump to find the cause quickly.

**High-availability telecommunications solution**
Carrier Grade Linux is the preferred operating system for today’s high-availability telecommunications solution, due to its rich feature set, rapid rate of technology innovation, and the large Carrier Grade Linux developer community. Embedded Linux provides support for real-time, as
well as excellent support for the common hardware platforms deployed in high availability solutions. Some of the key applications deploying embedded Linux today are gateways, signaling servers, and management servers. These all have requirements that are met by a highly available, carrier-class Linux system.

The OSDL is already defining the carrier grade architecture and its requirements, and implementation work is ongoing. Telecommunications equipment providers are seriously looking at this more robust Linux for next-generation wireless and IP networking solutions. At least one Linux distributor has already released a carrier-grade class product.

Enhancements are being made to the Linux kernel and open source software stacks to support Carrier Grade Linux. These upgrades are in several key areas, including standards, high availability, performance, scalability, and service ability enhancements.

Glenn E. Seiler is a high-availability architect for MontaVista Software and is responsible for managing MontaVista’s first carrier-grade quality product, MontaVista Linux Carrier Grade Edition. Before joining MontaVista, Glenn was responsible for managing high-availability clustering solutions at Santa Cruz Operation. Glenn has more than 15 years experience managing Unix and Linux operating systems, including previous work with Texas Instruments, SCO, BSDi, and MontaVista. Contact Glenn at: gseiler@mvista.com.

MontaVista Software, Inc. is a global supplier of systems software for intelligent connected devices and associated infrastructure. The company offers a family of products under the MontaVista Linux umbrella that address broad-based software developer needs encompassing applications ranging from communications infrastructure to consumer electronics.

For further information contact:

MontaVista Software, Inc.
1237 East Arques Ave.
Sunnyvale, CA 94085
Tel: 408-328-9200
Fax: 408-328-9204
Web site: www.mvista.com
Availability is technically defined as:

\[
\text{Availability} = \frac{\text{MTBF}}{\text{MTBF} + \text{MTTR}}
\]

MTBF means Mean Time Between Failures, and MTTR means Mean Time To Repair.

So, if a system were to offer MTBF of 20,000 hours with an average MTTR of 2.5 hours, then its availability would be 99.9875 percent.

The term 5-nines, or 99.999 percent uptime, literally implies exactly that – 365 days, 23 hours, 54 minutes, and 45 seconds of operational time during a year (or 5 minutes and 15 seconds downtime). However, in practice, that is not a useful number when defining high-availability software because these nines only apply to an overall solution that involves integrated high-availability hardware, software (OS and middleware), and the application.

A complete high-availability solution that demonstrates 5-nines requires close integration of high-availability hardware such as CompactPCI or AdvancedTCA, a robust high-availability Linux solution such as Carrier Grade Linux, high-availability middleware, and application software that can cause failover to redundant systems. Figure 2 illustrates a complete high-availability solution using Carrier Grade Linux.