Open Source Development Labs

Carrier Grade Linux Requirements Definition Overview

Version 3.2

Prepared by the Carrier Grade Linux Working Group

Open Source Development Labs, Inc.
12725 SW Millikan Way, Suite 400
Beaverton, OR 97005
USA

Phone: +1-503-626-2455

Comments on the contents of this document should be sent to cgl_discussion@osdl.org .

1 Carrier Grade Linux

A transformation is taking place in the telecommunication industry to meet the demands of new voice and data technologies. These technologies include Voice over IP (VoIP), the packet-switched alternative to the circuit-switched telephony of the past. To enable VoIP traffic, application servers must provide carrier-grade reliability that guarantees high service availability (up-time of 99.999% or better). In addition, these systems must scale to handle hundreds of thousands of calls and provide predictable performance, and high speech quality.

The telecommunications industry is undergoing enormous change as equipment providers migrate from proprietary platform architectures to open software environments and commercial off-the-shelf (COTS) platform architectures. Open software and COTS hardware are seen as the means for rapid deployment of new voice and data services, while reducing capital expenses and operating costs, enabling equipment providers to stay competitive and profitable.

Carrier Grade Linux (CGL) stands at the center of the move to open architectures. About three years ago, a group of industry representatives from platform vendors, Linux distribution suppliers, and network equipment providers set out to define how "Carrier Grade Linux" could enable environments with higher availability, serviceability, and scalability requirements, so the Open Source Development Lab (OSDL) CGL working group was formed. Since its formation, the working group has produced two versions of a specification to define these required capabilities. In response, Linux distribution suppliers are now demonstrating that they can meet the emerging needs of telecommunications by registering (disclosing publicly) how their products address the requirements defined in the *Carrier Grade Linux Requirements Definition – Version 2.0.*

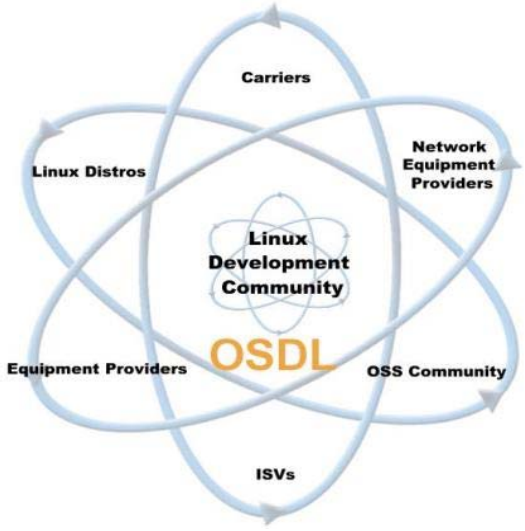

Illustration 1. Carrier Grade Linux Ecosystem

Today, the CGL working group has grown to include over three dozen representatives from platform vendors, Linux distribution suppliers, network equipment providers, carriers, and development community members worldwide. See Illustration 1. This expanded group has released this third version of the CGL requirements.

As CGL capabilities become available in mainstream implementations and distributions, Linux not only becomes more attractive for telecommunications applications, but the entire Linux community benefits from a highly available, scalable, high-performance, and manageable Linux environment.

High availability middleware components and service availability middleware that run on CGL systems are addressed by organizations such as the Distributed Management Task Force (DMTF), the Object Management Group (OMG), and the Service Availability

Forum (SAF). High availability hardware platforms underlying CGL are addressed by organizations such as the PCI Industrial Computer Manufacturers Group (PICMG) and the Intelligent Platform Management Interface (IPMI). See Illustration 2.

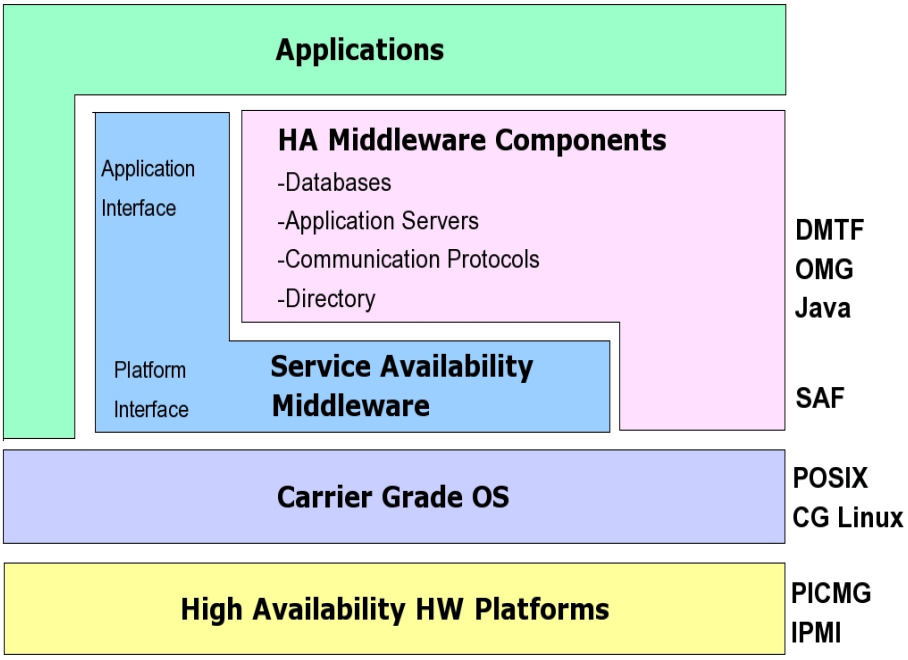

Illustration 2. Full Carrier Grade Application Stack

CGL 3.x specifications are an upward compatible superset of the CGL 2.0.2 specification. In 2003 and 2004, member companies have been producing communications products based on the CGL 1.1 specifications. In the latter half of 2004, Linux distributors began to announce Linux offerings based on the CGL 2.0.2 specification. Products based on the CGL 2.0.2 spec will begin to appear in 2005. A smooth transition is expected for carriers and equipment providers as Linux distribution suppliers incorporate CGL 3.x capabilities in 2005 and 2006.

Development is underway on many of the CGL capabilities that do not appear in mainline distributions. While the CGL requirements are specified for Linux-based platforms in the communications industry, a high availability, high performance, scalable system is viewed as beneficial to the entire Linux user community.

2 Document Organization

For clarity and ease of use, the specification has been split by topic into the following documents:

Requirements Definition Overview
Provides an overview of Carrier Grade Linux and the specifications included in CGL 3.2.

Availability Requirements Definition Version 3.2
Describes useful and necessary functionality for single node availability and recovery.

Clusters Requirements Definition Version 3.2
Describes useful and necessary components to build a clustered set of individual systems. The key target is clustering for high availability, although load balancing and performance are secondary aims. It is recognized that “one size fits all” is not achievable, so not all features will always be used together.

Serviceability Requirements Definition Version 3.2
Describes useful and necessary features for servicing and maintaining a system and coverage of tools that support serviceability.

Performance Requirements Definition Version 3.2
Describes useful and necessary features that contribute to adequate performance of a system, such as real-time requirements. Also describes base operating system components for supporting performance tools (requirements for the tools themselves are not addressed).

Standards Requirements Definition Version 3.2
Provides references to useful and necessary APIs, specifications, and standards, such as POSIX, IETF, and SA Forum standards.

Hardware Requirements Definition Version 3.2
Describes useful and necessary hardware-specific support that relates to a carrier operating environment.

Security Requirements Definition Version 3.2
Describes useful and necessary features for building secure systems. It is recognized that “one size fits all” is not achievable, so not all features will always be used together.

These documents can be found at:
http://www.osdl.org/lab_activities/carrier_grade_linux/documents.html/document_view

3 Availability Requirements Definition Version 3.2

Telecommunication customers expect their voice and data services to always be available. System availability is dependent on the availability of individual components in the system. To help ensure 24/7 service, it must be possible to perform system maintenance and system expansion on running telecommunication networks and servers without disrupting the services they implement. Systems must be able to withstand component failures, making redundancy of components such as power supplies, fans, network adapters, storage, and storage paths essential. Software failures can also significantly impact the availability of a compute node, so robust application software, middleware, and operating system software is required for single node availability.

The *CGL Availability Requirements Definition – V3.2* is a collection of requirements that address the robustness of a single computing node. Availability is further enhanced by clustering individual computing nodes so that a node cannot represent a single point of failure. The single node requirements in the Availability section can be categorized as:

- On-line operations
- Redundancy
- Monitoring
- Robust software

On-line Operations

On-line operations enable the system to continue to provide a service while the software or the hardware is replaced or upgraded on the system. For instance, when a file system needs repair, repair procedures may require rebooting the system. However, CGL requires that it be possible to forcibly unmount a file system, allowing repair and remounting without rebooting. The ability to replace or upgrade hardware such as disks, processors, memory, or even entire processor/memory blades without bringing down that node or the network contributes significantly to continuous service availability.

Redundancy

A highly available system must be composed of redundant components and must be able to take advantage of redundant hardware such that the system continues to function when a component fails. Ideally, designs can eliminate all single points of failure from a system. Using redundant communication paths, such as redundant network ports and host adapters, together with network fail-over software capabilities, such as Ethernet bonding, improve network availability. Redundant storage paths, such as redundant fibre channel ports and host adapters used with multipath I/O, improve storage availability. Redundancy of memory components may not be possible, but error detection and correction can be used to mask memory cell failures; CGL requires software Error Correction Code (ECC) support. Single bit errors are reported when they are detected in the hardware and logged by the kernel. The kernel invokes a panic routine whenever uncorrectable multi-bit errors are detected.

Monitoring

Rapid detection of hardware or software failures requires health monitoring. Health monitoring is also needed to check for hardware or software that is beginning to fail, such as ECC memory checking, predictive analysis for disks, and processes that do not respond in a predicted way. Examples of CGL monitoring requirements include *Non-Intrusive Monitoring of Processes* and *Memory Overcommit Actions*. The *Non-Intrusive Monitoring of Processes* requirement detects abnormal behavior by a process, such as process death, and initiates an action, such as the creation of a new process. The *Memory Overcommit Actions* requirement monitors system memory usage and controls process activity when memory usage exceeds specified thresholds.

Robust Software

Robust software not only implies high quality levels for operating system software, middleware, and application software, but includes capabilities for maintaining and upgrading software without bringing the system down. In many cases, continuous service availability can be maintained. A *Live Patching* requirement enables process modification without process termination. An E*xcessive CPU Cycle Detection* requirement enables the detection of abnormal process behavior by setting CPU usage thresholds at various points in the process, catching problems such as infinite loops or thrashing, and initiates actions such as restarting the process.

To maximize system uptime, it is important to minimize the time that a system is in an off-line state, such as shutting down or booting. The *Fast Linux Restart Bypassing* BIOS requirement addresses the time it takes to reboot a system by specifying an ability to bypass the BIOS firmware. The *Boot Image Fallback Mechanism* requirement defines a mechanism that enables a system to fallback to a previous known good boot image in the event of a catastrophic boot failure.

The requirements defined in the *CGL Availability Requirements Definition – Version 3* are common functions on proprietary carrier grade systems and address the gap between Linux, which has been developed with a desktop focus, and carrier grade systems, in which service availability is critically important.

4 Clusters Requirements Definition Version 3.2

The CGL working group conducted a clusters usage model study from which they learned that no single clustering model meets the needs of all carrier applications. So CGL takes a more general approach to defining clustering requirements. CGL defines the functional components of a carrier grade High Availability Cluster (HAC). The requirements for other cluster models, such as a scalability cluster, a server consolidation cluster, and a High Performance Computing (HPC) cluster, have been treated as secondary to requirements for the HAC cluster model. See Illustration 3.

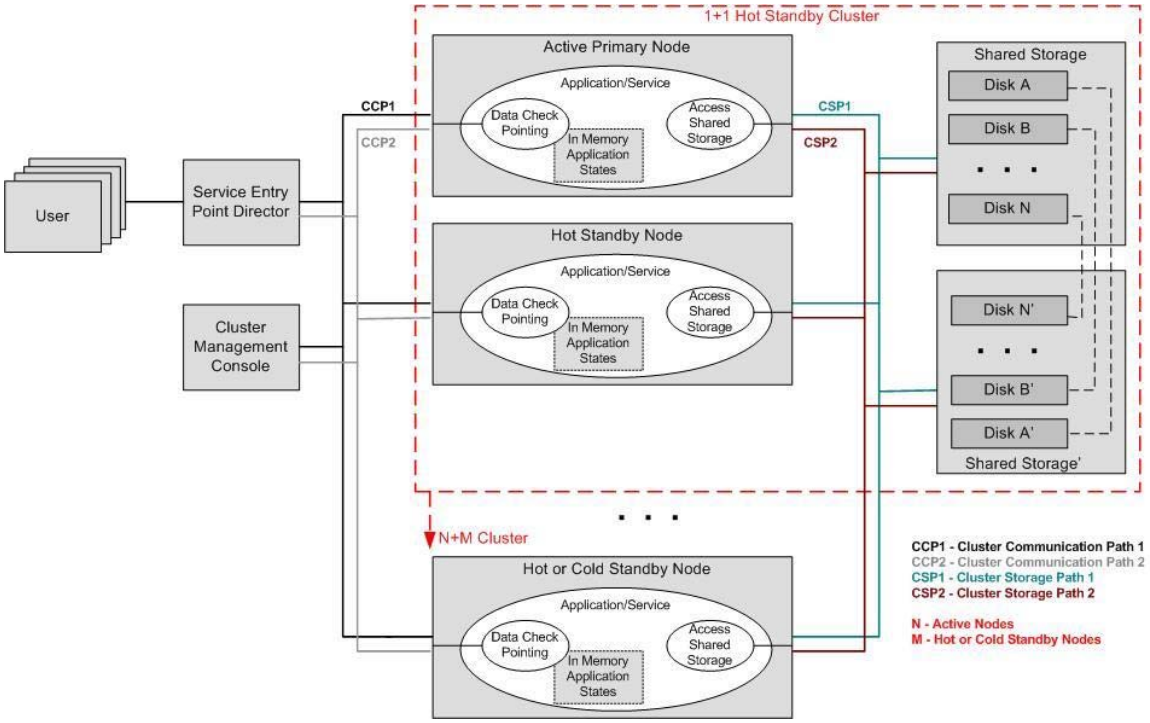

Illustration 3. CGL Cluster Model

A CGL high availability cluster is characterized by a set of two or more computing nodes between which an application or workload can migrate depending on a policy-based failover mechanism. Essentially, the cluster nodes can “cover” for each other. Carrier grade services must maintain an uptime of 5 nines (99.999%) or better and, quite often, a failing service must restart in sub-second time frames to maintain continuous operation.

A loosely coupled cluster model with no shared storage is a basic clustering technique that is suitable for many types of telecommunications applications servers. This model eliminates the possibility of a failed shared component affecting the availability of the service or the availability of system.

Whether shared storage is implied or not, a cluster provides the following advantages:

- Prevents a node from being a single point of failure. With hardware faults, the failing node can be replaced or repaired without affecting the service uptime (no unscheduled downtime)

- Allows a software or kernel upgrade to be completed on each node separately without affecting the availability of the service
- Isolates failing nodes from the cluster and enables service to continue using the remaining healthy nodes
- Allows hardware upgrades on each node separately without affecting service availability
- Enables increased capacity to meet load/traffic increases

CGL clustering functional requirements include support for redundancy (no single point of failure), not only at the cluster node level, but at the hardware level as well, including fans, power supplies, memory ECC, communication paths, and storage paths. To support continuous operation of carrier grade services, requirements are defined for node failure detection and various forms of service failover, such as application, node address, and connections failovers.

The CGL clustering requirements are framed around industry standard programming interfaces. The Service Availability Forum (SA Forum) has developed an Application Interface Specification (AIS) that defines service interfaces for clustered applications. The specification is OS-independent and is being used in both proprietary and open source cluster developments. The SA Forum AIS specifies a membership service API, a checkpoint service API, an event service API, a message service API, and a lock service API. AIS also specifies an availability management framework (AMF) that provides resource management and application failover policy in the cluster.

5 Serviceability Requirements Definition Version 3.2

The *CGL Serviceability Requirements Definition – Version 3.2* specifies a set of useful and necessary features for servicing and maintaining a system. Telecommunication systems such as management servers, signaling servers, and gateways must have the capability to be managed and monitored remotely, have robust software package management for installations and upgrades, and have mechanisms for capturing and analyzing failure information. A single point of control is required for applications, software, hardware, and data for functions such as data movement, security, backup, and recovery.

CGL systems will support remote management standards such as Simple Network Management Protocol (SNMP), Common Information Model (CIM), and Web-Based Enterprise Management (WBEM). Local management standards include IPMI and the Service Availability Forum's Hardware Platform Interface (HPI).

Debuggers, application and kernel dumpers, watchdog triggers, and error analysis tools are needed to debug and isolate failures in a system. Diagnostic monitoring of temperature controls, fans, power supplies, storage media, the network, CPUs, and memory are needed for quick failure detection and failure diagnosis.

6 Performance Requirements Definition Version 3.2

The *CGL Performance Requirements Definition – V3.2* is a collection of requirements for the Linux operating system that describe the performance and scalability requirements of typical communications systems. Key requirements include a system's ability to meet service deadlines; to scale to take advantage of symmetric multiprocessing (SMP), Hyper-Threading technology, and large memory systems; and to provide efficient, low latency communication.

Without predictable scheduling latencies, it is possible that service deadlines would not be met, resulting in dropped calls, unreasonable call-response characteristics, or even dropping the entire service from active operation. Soft real-time scheduling provides predictable scheduling latencies within defined loads. Latency and scheduling parameters are required to be configurable at runtime, including the scheduling quantum being configurable to 1ms or less. Protection against priority inversion is also required to maintain predictable scheduling.

To take advantage of scalable hardware architectures, CGL specifies support for SMP and Hyper-Threading technologies, which includes process affinity and interrupt affinity capabilities. Large memory systems of more than 4GB of physical memory are needed to handle the memory demands of scalable communication applications.

Protocol stacks are required to be prioritized so certain protocols may take scheduling priority over less important network protocols. To improve latency and reduce CPU usage in network communications, zero-copy network protocols may be needed. IPv6 forwarding tables are required to be compact and use a small amount of memory. Support in the Linux Kernel for a 9000 byte Maximum Transfer Unit (MTU) is required.

7 Standards Requirements Definition Version 3.2

One goal of the CGL effort to achieve high reliability, availability, and serviceability (RAS), and application portability is to leverage mature and well-established industry standards that are common and relevant to the carrier-grade environment and include them as part of the CGL requirements.

Open standards are important because they are freely available for anyone or any organization to use and because open standards can evolve with wide community feedback and validation. The CGL WG is actively working with recognized standard bodies, such as the Linux Standard Base (LSB – a workgroup of the Free Standards Group) and the Service Availability Forum (SA Forum). These organizations are producing standards and specifications that address the RAS and application portability gaps between Linux as it exists today and where it needs to be to support highly available communications applications.

The first requirement in the *CGL Standards Definition – Version 3.2* shows the CGL working group's desire to work alongside recognized standards bodies:

> *CGL 3.2 specifies the need for compliance to the Linux Standard Base (LSB) version 2.0.1 to ensure a CGL 3.2 distribution will have the support for the same level of the application binary compatibility as is required by the LSB standard.*

CGL 3.2 requires implementation of the latest interface specifications from the SA Forum to provide a common set of standards and building blocks for high availability architectures and platform management. The SA Forum provides standards specifications that define interfaces for cluster-aware applications (*Application Interface Specification - AIS version B.01.01*) and for platform management applications (*Hardware Platform Interface - HPI version B.01.01*). See the SA Forum site (www.saforum.org) for the B.01.01 versions of the AIS and HPI specifications.

Continuing from previous versions of the CGL specifications, the *CGL Standards Definition – Version 3.2* adds more POSIX compliance requirements based on IEEE Std 1003.1-2001. These additional areas of POSIX compliance are intended to bridge the application portability gaps as mainstream communications applications are ported to Linux application environments.

A variety of other standards requirements are included in the *CGL Standards Definition – Version 3.2* to address the networking, communications, and platform needs of carrier environments. Standards requirements such as Stream Control Transfer Protocol (SCTP), Internet Protocols (Ipv4/IPv6), Mobile Internet Protocol (MIPv6), Simple Network Management Protocol (SNMP), Intelligent Platform Management Interface (IPMI), IEEE 801.Q (virtual LAN), Diameter, Common Information Model (CIM), Web-Based Enterprise Management (WBEM), Advanced Configuration and Power Interface (ACPI), PCI Express, and Trusted Platform Module (TPM) are included.

More open industry standards will become mature and recognized over time. The CGL working group will evaluate them for consideration in future versions of the CGL requirements. The CGL working group believes that the adoption of open standards in mainline Linux offerings will benefit application developers and solution providers and

will carry Linux to the next level of popularity in the communications industry as well as the general Linux user community.

8 Hardware Requirements Definition Version 3.2

To stay competitive and profitable in the telecommunication industry, standards-based, modular, commercial-off-the-shelf (COTS) hardware components are being used along with open software, including operating systems, middleware, and applications. A goal of the CGL working group is to promote the migration of the telecommunication industry from the proprietary hardware platforms to COTS hardware by insuring that the Linux environment provides adequate support for these COTS platforms. The *CGL Hardware Requirements Definition – Version 3.2* identifies a set of widely-used industry hardware platforms and defines the support that is needed in the operating system for these platforms. The scope of these hardware requirements applies to the Linux kernel, kernel interfaces (APIs and libraries), system software, and tools.

The *CGL Hardware Requirements Definition – Version 3.2* specifies a set of generic requirements that are common across platform types. It includes support for blade servers, for hardware management interfaces, and for blade hot swap events. To address the need to manage highly available carrier grade systems through hardware out-of-band mechanisms, management capabilities such as those found in the Intelligent Platform Management Interface (IPMI) are also described.

Carrier-grade systems require high performance and high throughput interconnections within a system and between system nodes. Hardware-related requirements, such as PCI Express support, Message Signal Interrupt, and PCI Express Device Hot Plug, are included. Other hardware related requirements such as a CPU throttle mechanism, a “suspend to disk and resume” capability, trusted platform module (TPM) support, and boot-loader integrity check are also specified.

Considering the diversity of hardware platforms used in a carrier grade environment, the *CGL Hardware Requirements Definition - Version 3.2* does not define requirements for just one type of industry platform. Instead it defines generic platform requirements and then provides an “Industry Platforms” section to provide implementation guidelines for specific architectures. Examples of such industry platforms include AdvancedTCA, BladeCenter, CompactPCI and rack mount types of servers.

9 Security Requirements Definition Version 3.2

CGL Security Requirememnts Definition – Version 3.2 is the first release of security requirements in the *Version 3.x* train.

The telecommunications environment is different from a general-purpose computing environment. The most salient differences to consider in developing a CGL threat model are:

- CGL systems do not have many user accounts.
- User accounts do not reflect individual users.
- CGL systems are configured through custom user interfaces.
- CGL systems are typically configured without shell access.
- Administrators are trusted and competent.

The major threat to the telecommunications environment is, therefore, unauthorized access to management and control interfaces by outsiders. These outsiders can gain access by subverting the operating system or one of the applications it is running.

A severe potential security threat arises when applications need to touch multiple security planes. Many telecommunication services can be provisioned remotely by the end-user. Many ISPs that offer domain hosting allow customers to create new mailboxes or route incoming calls to 5-digit work extensions to any telephone number in the world with just a few clicks on a web page. Facilities like these create a new set of risks:

- Unauthorized rerouting of email and telephone calls by disgruntled associates or unscrupulous competitors.
- Exploitation of vulnerabilities in software to “jump” from one security plane to another, which can lead to many types of risks.

Mitigating these risks will require some forethought such that users of these systems are properly authenticated and authorized and that information traveling between planes passes through narrowly defined interfaces that protect against unauthorized access.

An OSDL special interest group for security (security SIG) is generating the CGL security profile and the architectural approach for security on communications servers.

10 References

Background information useful to readers of this document can be found in the following places:

Open Source Development Labs (OSDL) home page: http://www.osdl.org

The Carrier Grade Linux web page on the OSDL Web site: http://www.osdl.org/lab_activities/carrier_grade_linux

The OSDL “Carrier Grade Linux Requirements Definition, Version 2.0.2”: http://www.osdl.org/lab_activities/carrier_grade_linux/documents.html/document_view

OSDL “Carrier Grade Linux Requirements Specification, Version 1.1”: http://www.osdl.org/lab_activities/carrier_grade_linux/documents.html/document_view