Introduction to Carrier Grade Linux

A Paper for the Seminar on
High Availability and Timeliness in Linux
in Spring 2003

by Kimmo Hämäläinen (kphamala@cs.helsinki.fi)

5th February 2003

1 Introduction

The telecommunications industry today is changing because of new requirements. One motivator for the change is increasing competition, which forces telephone equipment manufacturers (TEMs) to operate in a cost effective way and constantly invent new attractive services. The competition has increased because many countries have deregulated the industry, and the deregulation has given an opportunity for new competitors to enter the market, thus pressing the prices down. The increasing use of the Internet and other data services have made the transition from circuit-switched to packet-switched traffic increasingly reasonable. The transition requires integrating voice with data traffic to save network capacity. A natural choice for a packet switching protocol to use in the new packet-switched networks is the Internet protocol (IP). Conversion of old circuit-switched networks to new IP-based networks is very expensive. The high cost is partly because of development costs of a new IP-aware operating system (OS) and hardware for the switches. Third generation mobile phones and future multimedia data services have been anticipated to require significantly greater bandwidth, which should be economically provided by the new networks, as well. Old telephone networks also need changes because of increasing mobility, brought by the growing population of mobile users, in telecommunications traffic [2].

The apparent inevitableness of this change in telecommunications networks has naturally led the industry to seek solutions for decreasing expenses and risks involved in the change. The risks and costs are usually smaller if development time of the required software can be shortened. A way to reduce the development work is to utilise work done by others. This can mean using commercial off-the-shelf (COTS) products as part of the new systems. Unfortunately, ready-made solutions do not usually exist because of lack of a common standard for carrier grade platform. A *carrier grade platform* refers to the hardware and operating system needed for a network server to be suitable for use in a telecommunications network.

An open, common standard for such a platform would benefit TEMs as well as makers of COTS products, and getting the carrier grade platform almost for free and ready-made would likewise be beneficial.

The open-source Linux operating system is the fastest growing server operating system today. The openness, extensibility, and stability of Linux makes it an attractive choice for the operating system of the carrier grade platform. The Open Source Development Lab (OSDL, `http://www.osdl.org/`) is an independent, non-profit, and vendor-neutral organisation dedicated to provide guidance and resources for open source developers to build carrier grade and data center (not covered in this paper) enhancements into Linux. The OSDL has formed the Carrier Grade Linux (CGL) Working Group (WG) to organise development of the carrier grade enhancements into Linux and its software stack. The CGL WG will cooperate with the industry and the open source community to accomplish its task. The goal is to develop a platform, named as Carrier Grade Linux, of high availability, robustness, and suitableness for development of state-of-the-art communications services.

1.1 What is Carrier Grade?

Reliability and availability requirements for a telephone network are very strict. The transition from old circuit-switched networks to packet-switched networks must not compromise the reliability or availability, which are usually required by the law. A telephone network is commonly required to provide "six nines" (99.9999%) availability, which means the network is allowed to be down only about a minute in a year [5]. In practice, the network must recover from hardware and software failures automatically, and maintenance operations must be done so that the network remains operational during maintenance. These requirements imply that the network's switches, and their platforms, are often required to be equally reliable and available. A platform or an operating system having the required characteristics is said to be *carrier grade.*

A carrier grade platform must provide soft real-time response times (of few tens of milliseconds), reliable transmission, and high capacity. High capacity means support for a large number of simultaneous connections (10,000 or more) and wide bandwidth. In order to enable billing, the platform has to be secure, be able to authenticate subscribers, and provide call logging. It must be possible to scale the platform up and down by adding/removing hardware, according to capacity requirements, without interrupting the operation. It should be possible to monitor the platform's performance remotely and manage it remotely. The platform must recover automatically from hardware and software failures, and it should provide means for remote problem solving.

2 Motives Behind the Use of Open Standards and Linux

Making an open standard of a carrier grade platform means that everybody has access to workings of the platform and can develop software components which are compatible with the platform. This means that independent software and hardware developers can make COTS products to be used with the platform. TEMs could then save money, shorten development time of new features, and reduce time to market by using ready-made third party software and hardware. This saving of time would reduce risks of new development projects because they could then begin to generate profit earlier. COTS developers would benefit from the standard through a large customer base as the standard's popularity grows. An open standard can help to make the platform "more" carrier grade, as design of the platform is transparent to the watchful eye of the public. Furthermore, operating system vendors (e.g. Linux distributors) can benefit from the standard and CGL functionality in marketing their systems.

The Linux operating system does not generate costs in form of run-time royalties, because it is free. A great advantage is the fact that Linux has a stable code base, which is tested in practice, on which to build enhancements and additional features to make Linux carrier grade. The use of Linux and a CGL standard makes the costly development and maintenance of proprietary operating systems no longer necessary and integration of third party components easier. Linux is the result of a large open source community of developers, who will develop it further for free and thus appear as some kind of free workforce in the industry's perspective.

3 The Carrier Grade Linux Working Group

The Carrier Grade Linux WG was established in January 2002 by the OSDL to promote use of Linux and open standards based software components in carrier grade platforms. The initial members of the working group include Alcatel, Cisco Systems, Hewlett-Packard Company, IBM, Intel, MontaVista Software, Nokia, Red Hat, and SuSE, which all are sponsors of the OSDL. The sponsors provide hardware, knowledge, and leadership for the WG, but products of the WG will be non-commercial and open source. The WG will provide a roadmap for Linux improvements, define requirements and architecture for CGL, and ensure consistency across Linux distributions. In addition, the CGL WG aims to reduce development risks and improve the flexibility of Carrier Grade Linux for the users.

The working group operates under the OSDL board of directors. The board of directors is a group of representatives of OSDL members. The WG consists of a steering committee, a roadmap coordinator, technical subgroups, and marketing subgroups. The steering committee is the highest organ in the WG, and it will approve the roadmap and the architecture, which both are approved after the committee by the OSDL board of directors. The roadmap coordinator is a person who manages development of the roadmap and project lists. The technical subgroups work under the roadmap coordinator and do the actual development work.

The vision of the CGL WG is "Next-generation and multimedia communication services can be delivered using Linux based open standards platforms for carrier grade infrastructure equipment." The working group tries to achieve the vision by, in addition to the means mentioned above, promoting a stable platform on which commercial components and services can be deployed, working closely with Linux distribution vendors to ensure that CGL features will be adopted in their distributions, and launching open source projects to implement CGL components when there is no suitable existing projects.

The working group contains three [1] technical subgroups: Proof of Concept, Specifications, and Validation. (In [5], the Specifications subgroup is missing and there is two additional subgroups, Architecture and Requirements.) The subgroups are open for new developers who are genuinely willing to help and dedicated enough. The Proof of Concept subgroup is responsible for making documents about the CGL design, leading implementation work of CGL features, ensuring testing of integrated features, and establishing and leading an open source project to coordinate the implementation work. The Specifications subgroup defines a set of requirements for CGL and prioritizes the requirements. Furthermore, the subgroup interacts with other standards defining bodies, open source communities, developers, and Linux distributions to ensure the enhancements can be adopted into the base Linux operating system. The Validation subgroup defines a standard test environment for CGL, coordinates designing of tests, and leads an open source project which develops the tests.

3.1 Focus of the Working Group

The Carrier Grade Linux WG has specified three initial application areas for CGL: gateways, signaling servers, and management servers [3]. Gateways are servers which mediate traffic between two networks. An example of a gateway is a server which receives voice traffic from a TDM (Time Division Multiplexed) network and transmits it as IP packets onto an IP network. A gateway platform needs to handle a large number of simultaneous connections and network interfaces in real-time without losing a packet. Signaling servers handle the signaling and routing required to establish calls and maintain status of calls. A signaling server can maintain tens of thousands of simultaneous connections and has to provide response times less than 80 ms. Because of the large number of connections, the server's efficiency depends much on the swiftness of context switching. Management servers manage networks, customers and services. Customer management includes authorisation of customers for a particular service and keeping track of the billing and customer information. A management server has to support different databases and intensive communication, but its response time requirements are lesser than with gateway and signaling applications.

The CGL WG addresses at least the following aspects of CGL in its work: high availability, security, scalability, turn-around times, and development environment and services. High availability refers to a system's ability to recover from software and hardware errors and at the same time remain operational. Security here includes authentication of users and applications, authorisation of user access to resources, logging for billing, and management of prepaid services. Scalability here relates to a system's ability to allow hardware additions to accommodate increased workload while the system is operational. Addressing turn-around times means that response times of the system are ensured to be in acceptable limits. Development environment and services refers to development of tools which facilitate development of carrier grade telecommunications applications. Requirements for a CGL system are described in section 5.

4 Carrier Grade Linux Architecture

The overall architecture of CGL is depicted in Figure 1. It shows the scope of the CGL WG does not include everything the platform consists of, but rather only the carrier grade enhanced OS and software development tools. The LSB in the figure stands for Linux Standard Base (`http://www.linuxbase.org/`), which is an effort to develop a common standard for Linux to facilitate interoperation of different Linux products. The carrier grade enhancements include additional code to improve availability and scalability of the Linux OS. The enhancements include improving the device drivers to tolerate some hardware failures, adding more support for redundant hardware components, and adding support for switching to redundant hardware into the kernel. The CGL kernel enhancements include improving the Linux scheduler, making the kernel pre-emptible, and adding support for kernel level detection of hardware and OS failures. In addition, a CGL enhanced Linux OS provides hardware configuration and management interfaces for certain classes of hardware, co-processor interfaces for communication with external processors such as network processors, and interfaces for monitoring resources and services. CGL will also provide some support for platform replication and management of different platform software versions.

The development tools are made to facilitate debugging the kernel and carrier grade applications, for kernel profiling such as call counting and call graphing, and for tracing kernel events such as interrupts and system calls. The high availability (HA) components shown in the figure include mechanisms for process monitoring, monitoring system parameters, and managing clusters; interfaces to enable these are provided by the carrier grade enhancements. Middleware components provide services such as databases, object managers, and Java runtime environments; they are needed to support carrier grade applications, and they will use only standard kernel APIs such as POSIX. The applications in the figure provide TEMs ways to differentiate their product from other vendors' products; TEMs can use applications to improve high availability of the system or provide additional services. The high availability hardware platforms in the figure represent the hardware part of the system; it is able to recover from a hardware failure of any piece of hardware by replication and allowing the failed hardware to be replaced without interrupting the operation of the system.

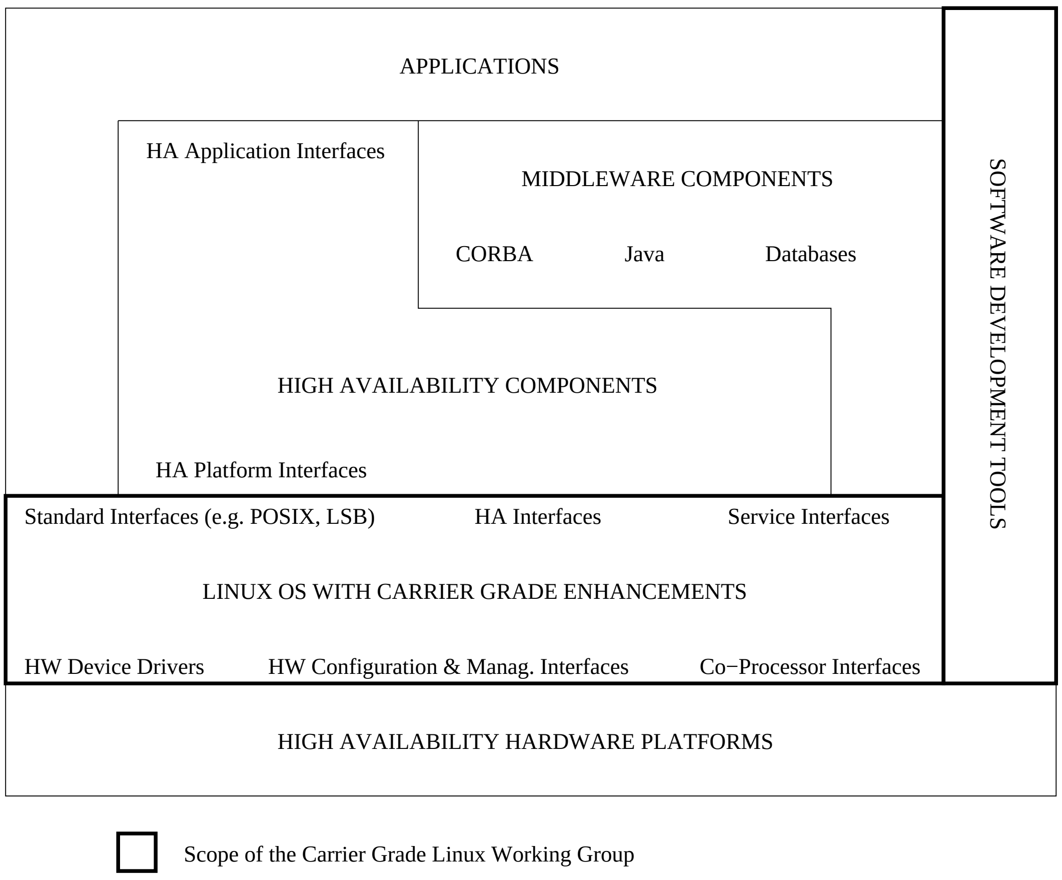
APPLICATIONS
HA Application Interfaces
MIDDLEWARE COMPONENTS
CORBA
Java
Databases
HIGH AVAILABILITY COMPONENTS
HA Platform Interfaces
SOFTWARE DEVELOPMENT TOOLS
Standard Interfaces (e.g. POSIX, LSB)
HA Interfaces
Service Interfaces
LINUX OS WITH CARRIER GRADE ENHANCEMENTS
HW Device Drivers
HW Configuration & Manag. Interfaces
Co−Processor Interfaces
HIGH AVAILABILITY HARDWARE PLATFORMS
Scope of the Carrier Grade Linux Working Group

Figure 1: Carrier Grade Linux Architecture

5 Carrier Grade Linux Requirements

The OSDL CGL WG has divided the CGL requirements specification in eight categories of requirements [6]: standards, platform, availability, serviceability, tools, performance, security, and scalability. Each of these categories has been further divided in three priority levels: level 1 defines requirements for the first release of the specification, level 2 defines requirements which are desirable for the first release and required for the second release, and level 3 defines requirements for future releases. The following paragraphs describe briefly some of the requirements in the categories as they are found in documents [4] and [6].

Standards category The standards category includes measures to make Linux compliant with various standards and develop new standards to improve reliability of CGL. First priority requirements in the category include Linux Standard Base compliance, some IPv6 compliance (including Mobile IPv6 and IPSec for IPv6), SNMP (Simple Network Management Protocol) support (for SNMPv1, SNMPv2, and SNMPv3), and extending POSIX compliance. The POSIX compliance is improved in the following areas: timers, signals, message queues, semaphores, event logging, and threads. Second priority tasks add support for SCTP (Stream Control Transmission Protocol, RFCs 2960 and 1112) and extend IPv6 compliance. Lowest (level 3) priority requirements will include requirements for a standard API such that applications can work in different distributions; those requirements are anticipated to utilise (future) specification work done by the Service Availability Forum (`http://www.saforum.org/`).

Platform category The platform category rates support for hot swap, remote boot, boot cycle detection, diskless operation, and no console operation as the most important requirements. Hot swap in CGL is defined to include hot insert, hot remove, and hot device identities. Hot insert means that hardware can be added to the system without interrupting the service provided by the system, and hot remove means about the opposite. Hot device identities makes it possible for a single device to keep its identity over system boots and hot swaps. Boot cycle detection refers to the system's ability to detect frequent reboots and go offline if that happens. Diskless operation includes the ability to boot without a disk and load all applications remotely, with and without a ramdisk. The second priority adds the ability to have the system console on a remote system, to boot an alternate kernel if it detects that the system is rebooting frequently, and hyperthreading of CPUs. There are no level 3 requirements in the platform category.

Availability category In the availability category, on the priority level 1 there are device driver hardening and support for watchdog timers, application heartbeat monitors, Ethernet bonding and aggregation, RAID 1 (disk mirroring), journaling file system (such as ext3 and ReiserFS), and HA disk and volume management. Driver hardening means making davice drivers more reliable by reviewing their code, adding checking code, and adding failure recovery code. The purpose of the device hardening is to prevent the kernel from panicing (i.e. stopping) in case of hardware failure and extend uptime of the system. Hardening is required in network, disk access, and logical storage device (or logical volume manager, LVM) drivers. Watchdog timers are a way to ensure that a software component is still running. They are hardware clocks which must be reset regularly or they reboot the system. Heartbeat monitors are programs which monitor applications in similar way as watchdog timers, but they can restart only the monitored application instead of the whole system. Ethernet bonding allows multiple Ethernet devices to be bound in a single logical device, which makes it possible to re-route traffic through another Ethernet link if some link bound to the device goes down.

Serviceability category The serviceability category includes requirements for resource monitoring, kernel crash dump and analysis features, structured kernel messages, dynamic kernel probing, hardware error logging, and remote access to error logs. The priority 2 adds requirements for fast boot, consistent system device enumeration, online diagnostics, and forced unmount of file systems. The level 3 includes kernel panic handler enhancements and kernel dump generation of a running system.

Tools category The first priority requirements in the tools category include debuggers for threaded programs and kernel, and a tool for analysing kernel crash dumps. The second priority adds support for fault injection testing, kernel tracing, and kernel profiling. The level 3 improves debugging tools, so that debugging can proceed from a parent process to a child process.

Performance category The level 1 in the performance category includes pre-emptible kernel, RAID 0, application pre-loading, and support for scaling analysis. The second level adds virtual memory improvements, support for multiple scheduler policies, and SMP (Symmetric MultiProcessor) enhancements. The lowest priority brings support for a self-resizing file system.

Security category The version 1.1 of the CGL requirements document [6] does not list any requirements in the security category, but security is addressed in the CGL WG white paper [3], which says that the CGL WG will ensure secure operation of the kernel and provide guidelines for correct use and installation of CGL.

Scalability category The version 1.1 of the CGL requirements document [6] does not list any requirements in the scalability category, but scalability issues have been clearly taken into account in other categories, e.g. in the performance category. The white paper identifies the following aspects which must be scalable in CGL:

- Number of processes, threads, and timers
- Number of network connections
- Network bandwidth
- Number of I/O devices
- Number of other I/O related entities (such as disks, partitions, files)
- I/O bandwidth
- Memory size
- Disk size

- Number of processors

Scalability here means that there must not be artificial limits for the resources and increasing the number of resources will not degrade the performance significantly. Also, for example adding more processors must increase the processing power almost linearly.

References

[1] The Web site of the Carrier Grade Linux Working Group. http://www.osdl.org/projects/cgl/technical.html [2nd Feb 2003], 2002.

[2] OSDL Carrier Grade Linux Working Group, Charter version 2.0. http://www.osdl.org/projects/cgl/charter2_0.pdf [2nd Feb 2003], 2nd Nov 2002.

[3] OSDL Carrier Grade Linux Working Group, Carrier grade linux technical scope white paper. http://www.osdl.org/projects/cgl/cgl_technical_scope_whitepaper.pdf [2nd Feb 2003], Jun 2002.

[4] Mehaffey, John, Carrier Grade Linux: what you need to know. http://www.commsdesign.com/story/OEG20020827S0008 [2nd Feb 2003], 27th Aug 2002.

[5] Kukkonen, Mika, Introduction to carrier grade Linux. http://www.applied-computing.net/articles/kukkonen.shtml and http://www.applied-computing.net/articles/kukkonen.2.shtml [2nd Feb 2003], 2002.

[6] The Carrier Grade Linux Requirements subgroup, Carrier Grade Linux requirements definition version 1.1. http://www.osdl.org/docs/cgl_requirements_definition___11_pdf.pdf [3rd Feb 2003], 25th Sep 2002.