

Open Server Architecture

April 2008

Technology Paper

The Open Server Architecture is flexible, open and easier to build applications on. This is achieved primarily through open and standardized interfaces and a layered structure.

Contents

1	Executive summary	3
2	Introduction	4
3	Key Challenges and Drivers	4
3.1	New development paradigm	4
3.2	Hardware independence	5
3.3	Industry forum co-operation and open source	5
3.4	Integration based on late binding	6
4	The Open Communication Server Architecture	7
4.1	The hardware layer	8
4.2	Open Operating Systems and Virtualization	8
4.3	Middleware for Telco grade performance	9
4.4	Application servers and Java EE	10
4.5	3PP Eco system	11
5	Conclusion	11
6	Glossary	12
7	References	13

1 Executive summary

The traditional way of supplying a proprietary vertical solution including everything from the hardware through operating system, middleware and even applications is being challenged. JAVA EE has emerged as a de-facto standard for building Application Servers.

A paradigm shift is occurring where communication servers are undergoing standardization and being shaped in horizontal layers with in-between standardized interfaces.

Such a new standardized architecture will provide for a rich eco system of components, which can be taken advantage of in the quest of ever increasing efficiency gains and shortening of the Time To Market (TTM) with best of breed choices.

The TTM gains from the ability to reuse functionality developed in many solutions is not the sole benefit of standardized horizontal layers. Depending on the completeness of the interface layers, a higher level of portability without taking special measures in the development of the application parts is achieved.

A looser coupling of components through standardized interfaces and availability of pre-developed applications enables late integration and is as well a fundament for all architectures making use of SOA characteristics. The later requires new methods to introduce guaranteed characteristics on interfaces and also between or in combinations of several components.

The loose coupling and primarily the availability of components enables late integration of solutions aka late binding. This requires new methods were characteristics must be guaranteed on component level.

The Open Communication Server Architecture addresses the problems listed above through a layered approach where each layer is isolated by standardized interfaces.

2 Introduction

The traditional way of delivering systems, by providing a vertical solution including the complete stack from hardware to application (often proprietary and vendor unique) is being challenged. The future way of viewing a system will be horizontally.

It has long been common practice to use commercial hardware, commercial operating systems, and other components such as databases for IT servers.

A paradigm shift where communication servers are undergoing standardization even for Telco Grade characteristics is underway.

Standardization in the IT-world, SA-Forum [1], JCP [2] and through alliances like SCOPE [3], Linux Foundation [4], etc and not least by the increased use of Open Source software has made this not only possible but inevitable.

With standardized API in middleware's, Java coming of age for communication systems, and a flourishing eco system, the world of communication servers will change.

3 Key Challenges and Drivers

3.1 New development paradigm

In the quest for ever increasing efficiency, and as individual areas become more or less non-differentiating, the possibility to source components and integrate them into products with limited effort becomes vital. Several aspects come in to play here:

Less time will be spent on developing existing, non-differentiating functionality (or as open source, free implementations). Instead, an approach where it is possible to efficiently incorporate already existing implementations into the relevant context is needed to improve Time To Market.

A support to this change in development is of course the large number of 3PP products, both as a commercial eco-system and in the shape of Open Source.

A company can prepare for this new way of working by developing internal applications in an Open Source fashion.

3.2 Hardware independence

Hardware requirements on a system, in terms of its characteristics will change over time and is often not easily defined due to e.g.:

- Different customers have a different views
- The application SW may be deployed in different environments
- The same functional components are used in different contexts
- The context for the same software may change over time

Software solutions will require an environment that is independent of the hardware; both from a design and a deployment perspective.

The key enabler for such an environment is a standardized approach to strict horizontal interfaces between layers, allowing for a decoupling of hardware and software. This enables the use of hardware with different characteristics (such as e.g. NEBS), without affecting the software.

For certain market it may be required for regulatory reasons to support deployment on certified hardware.

3.3 Industry forum co-operation and open source

A significant effort is spent on defining common layered solutions where Telecom companies, as well as IT companies are co-operating.

This has created a number of forums such as Service Availability Forum (SA-F) which is defining a high availability middleware with standardized interfaces towards applications as well as standardized hardware interfaces.

Initiatives like SA-F generally gather most companies within each eco-system and produce specifications or implemented functionality that are very widely spread. Furthermore they also align the market around a consolidated number of specifications. An example of such an industry initiative gathering many Network Equipment Providers is the SCOPE alliance.

Parts of the work referred to above will be done in open source, defining de-facto standards rather than standardization through specification. At the same time, the barriers for adoption are lowered thanks to the early availability of code. With software leaving a lot of choices, the reference implementations of open source is used to determine many of the details thus helping in the standardization of the server architectures.

Further benefits of Open Source are large communities sharing the work, the possibility to contribute and many persons scrutinizing the code ensuring that the code is of very high quality.

3.4 Integration based on late binding

With standardized interfaces comes the opportunity of late choices of components including application software, the middleware and the hardware and operating system.

A multiple choice of components in each layer, opens up for a number of permutation/combinations when integrating a vertical solution posing a risk of increasing the verification efforts and also of creating a number of set and rigid vertical integrations with the required characteristics, in which case we would be back to the more familiar un-interchangeable vertical stack.

Examples of issues that must be handled in this new setup are:

- Characteristics requirements must be addressed at component level, i.e. at the eco system part development.
- Certification of such eco system parts must be addressed before integration to a complete system.

This must be done without defeating the overall purpose, which in the end is to save cost, time and effort by allowing a selection of components which are optimal for a particular case.

4 The Open Communication Server Architecture

The purpose of the Open Communication Server Architecture is to serve as model for how to build server nodes with open interfaces and standard components without sacrificing the telecom heritage in terms of: high availability and low cost of operation.

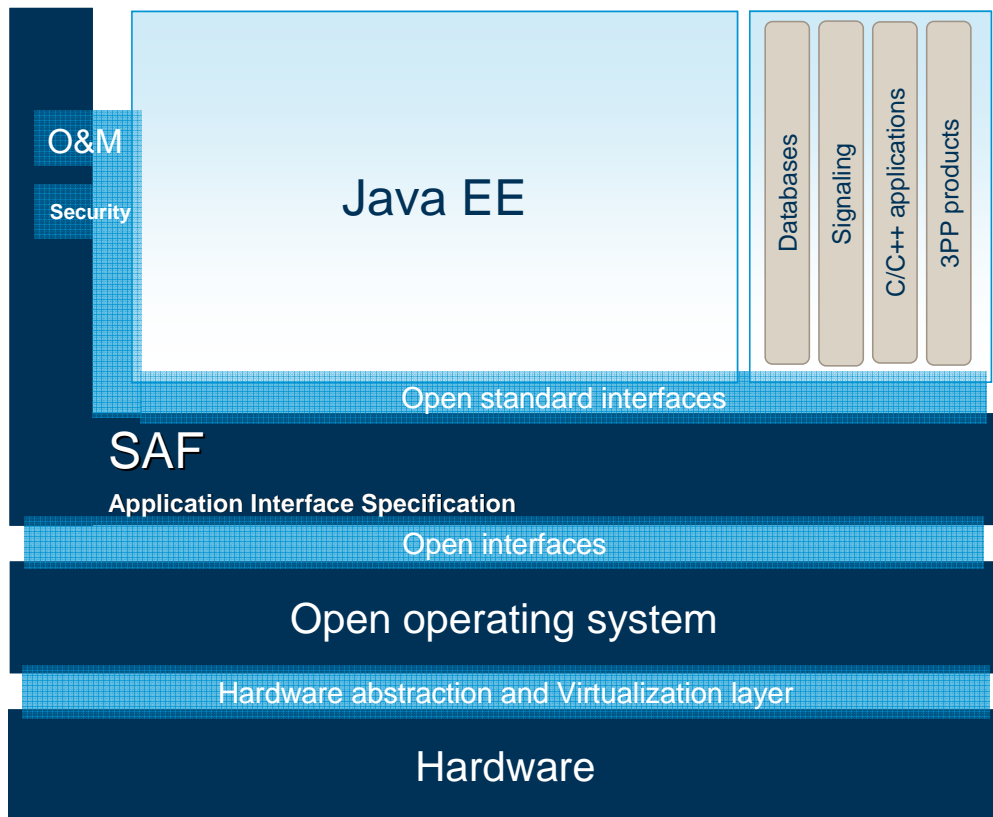


Figure 1 Open Communication Server Architecture reference model

The horizontal layering in the open communication server architecture is coupled with the basic elements of any application; the hardware, operating systems, middleware and application code. The ability to combine third party software with internally developed software is crucial from a Time To Market perspective.

4.1 The hardware layer

Building server hardware is easy enough; almost any telecom vendor could easily do it themselves. Building more generic hardware that is commonly accepted by all operating systems and component suppliers is more complicated. The intimate co-operation needed with processor, chipset suppliers, communication components and operating system suppliers requires a great deal of coordination.

The term Commercial Off The Shelf (COTS) is probably one of the most misused and misunderstood abbreviations in the industry. Important aspect of COTS hardware is that it is supported by most open operating systems and software component suppliers, supporting open interface standards (e.g. HPI) and developed according to generic market requirements.

Server applications are often well isolated from specific ties to the hardware. The same requirements apply for most servers in terms of processing power, memory, network interfaces and I/O solutions.

Thus creating hardware for a new server handling heterogeneous applications with varying needs requires more from the hardware and will ask more of the COTS hardware.

The opportunity for HW independence is made possible by e.g. the Hardware Platform Interface (HPI), which is an abstracted interface for managing computer hardware. The interface is developed by the SA-Forum, but conveniently enough it also exists in open source through openHPI [5].

4.2 Open Operating Systems and Virtualization

The open communication server architecture relies on two very important components in order to secure loose coupling between hardware and software.

The most important is of course using open operating systems such as Linux. Combining Linux with virtualization gives a high degree of portability and a very good approach to securing good resource utilization.

The operating system and its accompanying tool chains are extremely important means in the productivity chain of developing systems. The innovative and agile Linux community constantly provides tools and characteristics above or beyond what can be achieved by closed environments.

The earlier discussions about carrier grade Linux, has now settled down to a more sensible approach. There are some aspects of carrier grade Linux that are still very important. Examples is the need for long support cycles which within the Telecom industry can extend to 10 years or beyond, support for telecom defined communication protocols and clarifications of what parts of existing standards (Crypto, IPv6, Routing protocols, Real Time POSIX parts, ...) that are needed in telecom systems.

HW without single points of failure, the application architecture together with the middleware forms the fundamental explanation of the end characteristics for the behaviors of the whole system.

Virtualization is the other increasingly important component in the application server domain; the ability to efficiently deploy new services requires the ability to deliver services that can grow and shrink with customer demand.

Virtualization offers the benefit of being able to use the same hardware for many different roles, thereby making the hardware investment more future proof. In the open communication server Architecture, the strength and ruggedness of clustered systems, are combined with the ease of deployment that virtualization enables.

Virtualization has so far been concentrated to the problem domain of generic IT usage where the driving use-case is to increase the utilization of CPU's. Since this is not the main case for telecom (normal utilization 10 times higher than in IT HW), virtualization will bring some benefits also to telecom as mentioned but not such significant business improvements as in the IT industry. Other benefits given by virtualization for telecom is instead the ability to have CPU's dedicated to signal termination and processing CPU's in the same chip, running legacy code on one CPU and interface and new functionality on another, and so forth.

4.3 Middleware for Telco grade performance

Telecom manufacturers have earlier provided all applications a customer would ever want. Now there's general consent that third party vendors will be as important if not more important in creating the end-user applications.

The operator's role in this context depends on their ability to provide mechanisms and business models for efficient service and content delivery to the end users. The ability to provide new services will require open platforms with standardized interfaces for interoperability and flexibility.

Middlewares that used to be the core asset of any telecom equipment manufacturers now needs to be opened up for third parties in order to make the eco-systems grow. Combining the strengths of service creation with IT application servers, maintaining the Telco grade robustness is needed.

The telecom middleware is no longer a major differentiator but must also be balanced with an ability and openness to integrate 3pp; the ability to use one common middleware will drive the eco-systems considerably faster. The Service Availability Forum (SA-F) is the first organization that has succeeded in bringing forward a standard that is good enough to base a Telco grade middleware on. The specifications have already been implemented to a varying degree by multiple vendors/projects and specifically in the project OpenSAF that is a complete HA MW implementation in open source and the evolution continues.

The industry momentum around SAF is very strong, however the SAF specifications used to be very C/C++ centric. In order to enable the Java applications to take full advantage of the SAF high availability framework; a standardization activity in the Java community is ongoing and some work is also done within SAF.

4.4 Application servers and Java EE

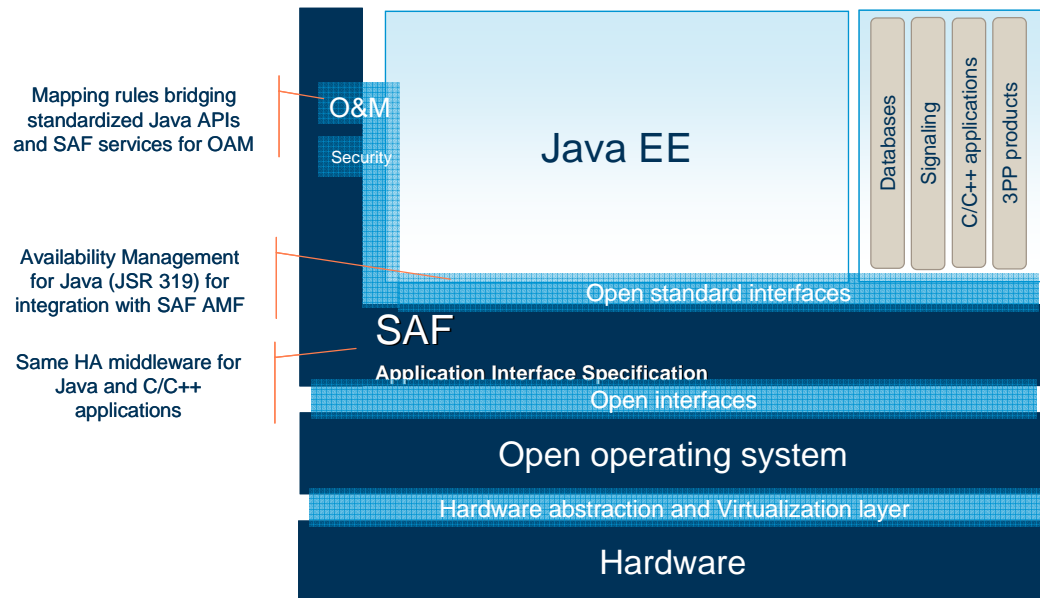


Figure 2 Examples of Java standardization

There is a growing understanding in the industry that JavaEE is a vehicle for service creation. With its support for service oriented architectures and web services it is well suited to serve as the base for e.g. IMS application servers.

The VM architecture for Java code enables suitable mechanisms for hardware abstraction and an excellent ability to work in virtualized server solutions obtaining high portability of software.

A number of Java Specification Requests, like JSR281¹ and JSR289² for IMS services within Java clients and servers and JSR 319³, treating Telco grade in middlewares, provides efficient infrastructure for creating IMS applications using SIP protocols and web services solutions.

¹ JSR281 IMS Services API

² JSR289 SIP Servlet v 1.1

³ JSR 319 Availability Management for JAVA

4.5 3PP Eco system

The standardized interfaces in the open telecom server architecture create an eco system of third party products. Using the eco system is not a competitive advantage in itself, since by definition, so do many others. The competitive advantage comes with the possibility to use a larger portion of your total resources on developing things in the areas where you have your core competence and differentiation, since you can achieve a lot of the non-differentiating things by means of utilizing the eco system.

It is important to be part of the eco system and drive the right requirements to ensure that it becomes and remains the right eco system. There is a strong industry momentum around SAF and the industry is requiring SAF support for third party products like databases, signaling stacks etc.

JavaEE has a large and fast growing community with an increased use within telecommunications.

5 Conclusion

The Open Server Architecture is flexible, open and easier to build applications on. This is achieved primarily through open and standardized interfaces and a layered structure.

Java is the established vehicle for Service creation, but is now making its way into control layers obtaining really good Telco characteristics through standardization and additions in Java EE.

Operating systems is not a differentiating factor and high availability characteristics will not depend on the OS alone.

To be able to handle new processor architectures like multicore, and to enable high portability, virtualization will be a crucial component. Virtualization should be seen as an extension to the OS, making up for a deficit of handling multicore. There is a need for standardization of virtualization.

Standardization will allow for the creation of an eco-system,. Such standardization will be a mix of classic specification standardization, but also based on Open Source de-facto standardization.

Reaching its full potential with standardized interfaces and an evolved eco system of readily available components, TTM will be shortened and a lower OPEX can be expected thanks to a common way of handling such a system.

6 Glossary

AIS: Application Interface Specification

HPI: Hardware Platform Interface

JAVA™ is a trademark of Sun Microsystems

JCP: Java Community Process Program. Community developing the Java technology

SCOPE Alliance: Industry alliance committed to accelerating the deployment of carrier grade base platforms for service provider applications.

NEBS: Network Equipment Building Standard. Safety, spatial and environmental design guidelines applied to telecommunications equipment primarily in the United States.

SAF: Service Availability Forum. Industry consortium fostering an ecosystem that enables the use of commercial off-the-shelf building blocks in the creation of high availability network infrastructure products, systems and services (AIS; HPI...)

7 References

- [1] The SCOPE alliance www.scope-alliance.org
- [2] OPENSATF www.opensaf.org
- [3] SA-Forum www.saforum.org
- [4] Java Community Process Program www.jcp.org
- [5] Linux Foundation www.linux-foundation.org
- [6] Open-HPI www.openhpi.org