



# **Services and Support Profile –** **Long Life Cycle Support**

**Version 1.0 November 30<sup>th</sup>, 2007**

Copyright © 2007 SCOPE Alliance. All rights reserved.

The material contained herein is not a license, either expressed or implied, to any IPR owned or controlled by any of the authors or developers of this material or the SCOPE Alliance. The material contained herein is provided on an “AS IS” basis and to the maximum extent permitted by applicable law, this material is provided AS IS AND WITH ALL FAULTS, and the authors and developers of this material and SCOPE Alliance and its members hereby disclaim all warranties and conditions, either expressed, implied or statutory, including, but not limited to, any (if any) implied warranties that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

Also, there is no warranty or condition of title, quiet enjoyment, quiet possession, correspondence to description or non-infringement with regard to this material. In no event will any author or developer of this material or SCOPE Alliance be liable to any other party for the cost of procuring substitute goods or services, lost profits, loss of use, loss of data, or any incidental, consequential, direct, indirect, or special damages whether under contract, tort, warranty, or otherwise, arising in any way out of this or any other agreement relating to this material, whether or not such party had advance notice of the possibility of such damages.

Questions pertaining to this document, or the terms or conditions of its provision, should be addressed to:

SCOPE Alliance,  
c/o IEEE-ISTO  
445 Hoes Lane  
Piscataway, NJ 08854  
Attn: Board Chairman

Or

For questions or feedback, use the web-based forms found under the Contacts tab on [www.scope-alliance.org](http://www.scope-alliance.org)

## 1. PURPOSE

As NEPs endeavor to engineer telecommunication systems by drawing upon the COTS ecosystem, they must have a way to qualify the components that they leverage from this environment. This gives rise to a plethora of terminologies and methodologies to perform this qualification; however, there is little consistency among the NEPs as to how they do this even though, in the end, they are attempting to accomplish the same result. Such a lack of consistency fundamentally complicates the job of ecosystem suppliers by causing them have to respond differently to each NEP. In the end, this fragmentation translates to increased costs, longer cycle times, and general confusion across this aspect of the ecosystem.

The purpose of this paper is to suggest some standard terminology and methods that NEPs may converge around for portions of the carrier grade engineering qualification process.

This paper deals with the ‘Long Life Cycle Support’ aspects. It basically reflects the requirements of Central Office (CO) systems. However, as more and more telecom equipment is operated in Data Centers (DC), many of the CO requirements are now valid also for DCs.

## 2. AUDIENCE

The target audiences of this document are:

- ✓ Network Equipment Providers who deliver applications and solutions to telecommunications and networking carrier companies in the context of Carrier Grade Platforms.
- ✓ Board, module and software vendors who market their products for use in Network Elements and other Network Equipment Provider applications built on Carrier Grade Base Platforms

## 3. REFERENCES

1. High Availability Glossary, [Unpublished] Draft 0.5, (Service Availability Forum – Technical Working Group, 15-Dec-2004).
2. Service and Support Profile - Service Availability, version 1.0, (The SCOPE Alliance, July 2007)
3. Scoping the Scope: Closing the Gaps of an Open Carrier Grade Base Platform, Revision 1.1, (The SCOPE Alliance, March 2007), available at <http://www.scope-alliance.org/scope-technical-position.pdf>.

## **4. TERMS AND DEFINITIONS**

<b>ATCA</b>	Advanced Telecom Computing Architecture
<b>BC</b>	Backward Compatibility
<b>CG</b>	Carrier Grade
<b>CGL</b>	Carrier Grade Linux
<b>CN</b>	Change Notification
<b>CO</b>	Central Office
<b>COTS</b>	Commercial off the Shelf
<b>CRU</b>	Customer Replaceable Unit
<b>DC</b>	Data Center
<b>EOES</b>	End of Extended Support
<b>EOPL</b>	End of Product Lifecycle
<b>EOPS</b>	End of Product Sales
<b>EOSS</b>	End of Standard Support
<b>ETSI</b>	European Telecommunications Standards Institute
<b>FRU</b>	Field Replaceable Unit
<b>NEBS</b>	Network Equipment Building System
<b>NEP</b>	Network Equipment Provider
<b>OPEX</b>	Operational expenditure
<b>QoS</b>	Quality of Service
<b>SA</b>	Service Availability

## 5. LONG LIFE CYCLE SUPPORT

Traditionally, there are significant differences in the support requirements of equipment in Data Centers and Central Offices:

- A typical system life cycle in Data Centers (DC) is about three years, i.e. at the end of the life cycle the complete system (hardware, software, data) is replaced by a new one and the data has to be migrated (re-formatted) before the new system can become operative.
- A typical system life cycle in Central Offices (CO) is ten years and more (from delivery to end of life), i.e. the system is in service and operational for ten and more years, system cabinets stay in an (mostly) unchanged infrastructural environment of a Central Office.

Recently more telecom systems are being installed in DC environments, thus the requirements for long life cycle support become also relevant for Data Center installations.

### 5.1 Definitions

A *life cycle* of a system or system component is the time between the start and stop of its operation. During the life cycle a carrier system or component has to fulfill all contracted requirements regarding functionality, workload volume, QoS, Service Availability, Serviceability, OPEX. In particular

- o no system cabinet will be exchanged/replaced
- o the system can be scaled up/down according to workload requirements
- o The environmental conditions of the system remain (mainly) unchanged

The *Installation* of a system or a system component means to build-up the system or component, to integrate it into the environment and to put it into an operational status.

The *De-installation* of a system or system component means to put it out of operation and remove it physically.

The *Replacement* of a system component means the de-installation of it (e.g. due to an erroneous behavior of this component) followed by the installation of a new one with the same interfaces.

A *Patch* is a fix for a software program where certain binary executable and related files are modified or replaced. It is used to repair a software bug or a security vulnerability, to improve usability or to improve performance.

*Repair* is the process of restoring a failed component to service. For hardware component this mostly means the replacement of an erroneous component; software is normally patched.

An *Update* of a system component means the introduction of minor fixes, changes or improvements (e.g. as patches) into the component (with no loss in functionality or capabilities and without any changes of the component interfaces).

An *Upgrade* of a system component means its replacement by a new one with significantly increased functionality (e.g. features) and/or non-functional properties (e.g. performance).

*Upgradeability* is the characteristic of a system or a component to be replaceable by an improved version of the same without significantly impacting the system's reliability of its, components or services.

A *Field Replaceable Unit (or FRU)* is an entity that can be replaced by a user in the field without negatively influencing service. Not all FRUs are hot swappable. Basic product inventory data for the FRU may be maintained at a proxy management controller elsewhere in the system. For ATCA platforms, examples of this type of FRU might include: backplane (the shelf housing, for all practical purposes), Power Entry Module, fan module, PMC, and RTMs.

A *Hot Swap FRU* is a managed hardware component of a system, which can be replaced without any electrical disturbance or damage to other components and without any service interruption of the system (no system shut down). A Hot Swap FRU has all of the hot-swap protection mechanisms during FRU insertion and extraction.

A *Hot Pluggable FRU* can be removed from the chassis while the system remains operational but requires software intervention before removing the FRU.

A *Customer Replaceable Unit (CRU)* is a component or sub-assembly, normally located on the outside of a system, which may be easily replaced by the user/customer without any specialist service training or specialized service tools. CRUs may be hot swappable or hot pluggable as defined previously.

A *Hot Swap CRU* can be removed from the chassis while the server remains operational and requires no administrative intervention before removing the CRU.

A *Hot Pluggable CRU* can be removed from the chassis while the system remains operational but requires administrative intervention before removing the CRU.

A *Service Interval* is the maximum time allowed for the replacement of a FRU. Service interval does not include time associated with complementary repair operations, such as time required for removing or re-attaching cabling. The intent of the service interval requirement is to ensure that the FRUs can tolerate the airflow changes associated with the shelf airflow patterns during the FRU replacement period without failure or performance degradation.

*Early fault detection* is the identification of an upcoming fault by collection and evaluation of information such as; intermediate errors, thresholds, routine readings, performance monitoring during normal operation and cumulative service operational time.

*Maintenance* of a system means a set of activities performed on a system to retain it or to restore it to a specified state. It is divided into preventive maintenance and corrective maintenance.

*Corrective Maintenance* of a system means a planned or unplanned update or a replacement of at least one of its components to fix a bug, to extend functionality or capacity, to improve the system behavior or to keep/recover it running.

*Preventive maintenance* means planned (scheduled) maintenance without immediate needs, typically scheduled outside of the peak load intervals of the system.

*Backward compatibility (BC)* of a new version of a component means that the new version supports all functions, non-functional properties (performance, resource utilization, emissions,) and interfaces (connectors, API's, protocols, data structures) of the former version to avoid the need for modifications in other parts of the system when the new component is introduced into the system.

*Binary backward compatibility* means the BC of binary software objects, (i.e. BC is guaranteed such that existing binary images will execute without re-compilation of any source code of the software stack including the application).

*Source Code backwards compatibility* means the BC of source code, i.e. to guarantee BC a re-compilation of the source code may be necessary.

A *Base software release* (version) introduces significant new features and/or non-functional behavior to the system.

A *Major software release* is the first or a follow up issue of a software component with relevant functional and/or non-functional improvements (upgrade).

A *Maintenance (minor) release* is an updated major release (specifically contains one or more patches), which can be installed with minor effort, compared to a major release.

An *Emergency Release* has the only goal to remove one or more critical bugs from the system which endanger the service availability. It has typically to be available within a few hours.

## 5.2 Long Life Cycle Details

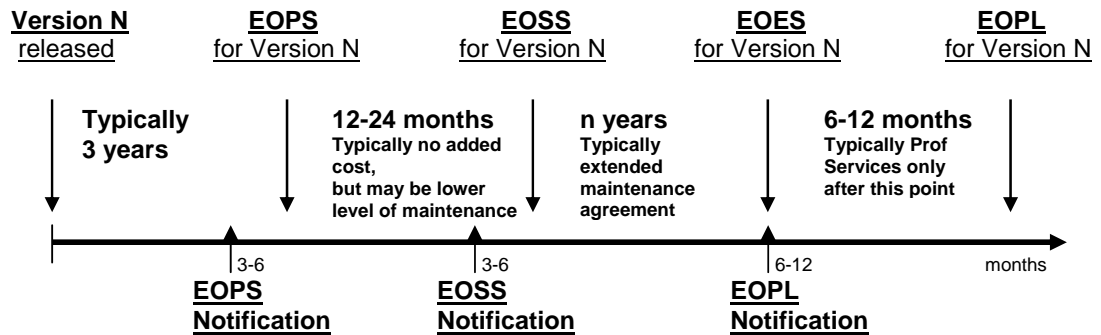
This chapter gives a more detailed description of typical requirements of carrier grade systems. This is related to the system as a whole, system components and processes.

From a system operators point of view the solution has to guarantee over its whole life cycle:

- Continuity in functions and non-functional properties
- Continuous service availability [1,2 ], while

- functional and non-functional properties are improved and extended
- Opex is reduced continuously
- Capacity adoption according work load characteristics
- Backward compatibility of all external interfaces, like network, management, etc.

The figure below shows a typical life cycle of a telecom system from delivery to market (between release and EOPS) to EOPL)



**EOPS** End of Product Sales  
**EOSS** End of Standard Support  
**EOES** End of Extended Support  
**EOPL** End of Product Lifecycle  
 Product is Retired

The requirement for extended support (ES) is a consequence from the extraordinary length of telecom application life cycles, compared to standard life cycles of enterprise applications. Thus, extended support requires special agreements between the application provider and the vendor of COTS modules.

### 5.2.1 Definition of EOxy

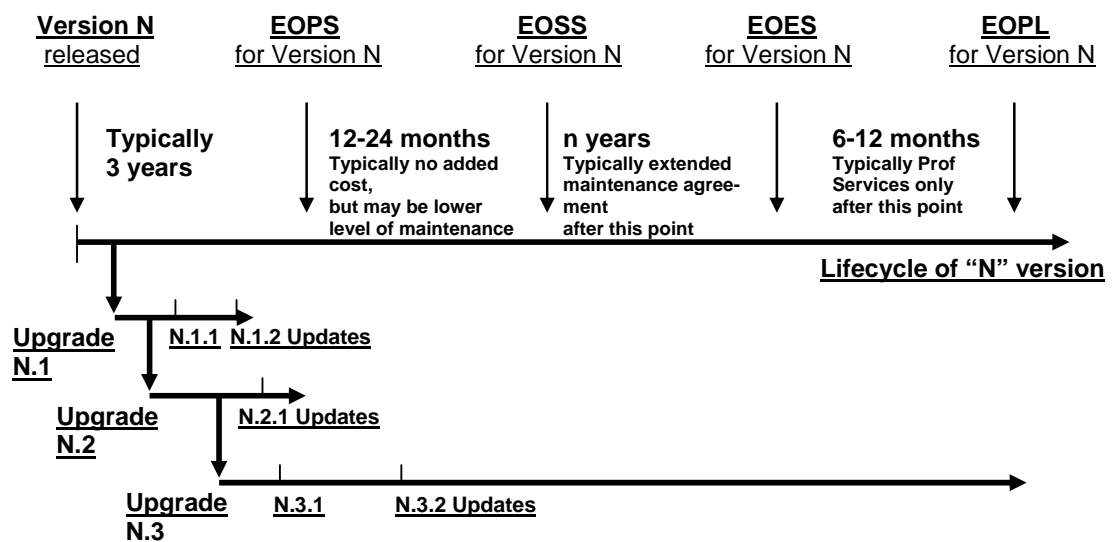
- EOPS, End of Product Sales
  - No new equipment deliveries
  - Fault reports are handled with their assigned priority (Hi, Med, Lo)
  - No new features are provided (major releases)
  - Corrections are at least provided for faults having major technical or commercial impacts on a customer's system
  - For all fault reports forward looking fault mapping is done (i.e. assess if the fault exists in later product releases and take appropriate action).
  - Maintenance releases are provided at scheduled dates.

- Emergency releases are provided as required.
- EOSS, End of Standard Support & EOES, End of Extended Support
  - Fault reports are still handled with their assigned priority (Hi, Med, Lo)
  - No upgrades on installed base
  - Corrections are only provided for faults having major technical or commercial impacts on a customer's system
  - For other fault reports forward looking fault mapping is performed.
  - Maintenance releases provided less frequently
  - Emergency releases are provided as required.
- EOPL, End of Product Life Cycle
  - Product information erased
  - No fault investigation performed
  - All fault reports are answered with “No Action” or similar
  - No fault mapping performed
  - Maintenance releases are not provided

If the system under consideration is a Base Platform [3] it has to be integrated with an application before it can be shipped as a product. This reduces the effective sales interval of a Base Platform as part of a product. Therefore an extended sales interval for Base Platform components, like hardware could be required.

If the product is used by several customers, then in most cases there are different versions of the product in use. The challenge is to handle these different versions in parallel.

The following figure shows the general relation between a Base release (version), Major release upgrades and Maintenance/Minor release updates of this version:



In this example upgrade of a version N stops once a new version N+2 is released (except for customer specific agreements)

## **5.2.2 Hardware**

### **5.2.2.1 System level**

The minimum life cycle expectation for Central Office (CO) systems is typically ten (10) years. Carriers expect stable requirements regarding the environmental conditions, so that no relevant changes of the CO infrastructure are necessary.

If telecom applications are executed on standard IT systems in Data Centers (DC) the system cabinets are normally exchanged more frequently; however, the overall system functions have to be guaranteed for ten years and more without interruption.

Enclosure designs should account for the projected component power dissipation increase over the installed system lifetime.

### **5.2.2.2 Component level**

All components are expected to meet the system's specified hardware service life unless they are explicitly exempted from it (e.g., hard disks and fans), and then they should have a clearly defined alternate hardware service life target and associated non-service affecting maintenance procedures.

In the case where a system components lifetime is shorter than the intended life cycle of the whole system, there is the need for component replacements during the system life cycle. Due to EOL of circuits and other elements there could be the need to replace an old component by an upgraded version. This new version has to fulfill the following (backward compatibility) requirements:

- no changes in interfaces (technical and programmable),
- no changes in software should be necessary (except firmware, drivers),
- new components must be designed to operate within the constraints imposed by the design of the installed target enclosure, in terms of the power dissipation, cooling, acoustic noise emissions, signal integrity, etc.
- old and new version of a component should work seamless together and parallel
- Typically it is required that the new version maintain source and binary software compatibility.

A typical service interval is limited to a few minutes for a single component. Short service intervals require careful preparation and planning. Three mechanisms enable this:

- early fault detection and
- fault isolation, fault recovery, architecture,....

- preventive maintenance

Early fault detection requires the collection of information on any anomalies in the behavior of a system or component; it may be fault counters, irregular power supply, abnormal hot spots, etc. Intelligent algorithms should combine this information to evaluate the urgency of maintenance activities. All information should be remotely accessible.

The primary goal of early fault detection is to avoid unplanned maintenance activities by focused planned maintenance.

### 5.2.3 Software

Over the whole life cycle of the system the software has to provide consistent support to

- The embedded hardware components, including new and modified hardware
- The full functionality as required from the applications, including new and enhanced applications
- The contracted capacity and performance, including up/down-scaling
- System management interfaces

During the life cycle of a CO system, normally more than one major version of the embedded software components has to be included. As they have different life cycles and upgrade intervals, software upgrades are typically challenging.

As today real systems have a layered architecture the life cycle management is more complex as described above. Without the fulfillment of the backward compatibility requirement the life cycle management ends up in

- high costs,
- longer time to market and
- reduced product quality.

The table below shows the minimum BC requirements that the platform must present to the application during a software lifecycle

<b>Subject</b>	<b>must</b>	<b>desired</b>	<b>optional</b>
Updates (patches)	X (binary, source)		
Minor upgrades	X (binary, source) <sup>1)</sup>		

Neighbored ver- sions of Base Sw Releases (n to n+1)	X (binary, source) <sup>2)</sup>		
'Remote' versions  n to n+x (x>=2)		X (source) <sup>2)</sup>	X (binary)

<sup>1)</sup> Minor upgrades may be applied to any lower numbered minor version within the same base software release, (i.e. minor release 2.5 may be applied to minor release 2.1)

<sup>2)</sup> Software features and interfaces deprecated in a major release n+1, may be removed in release n+2.

The table below shows the minimum Upgradeability requirements during a software life-cycle

<b>Subject</b>	<b>must</b>	<b>desired</b>	<b>optional</b>
Updates (patches)	X		
Minor upgrades	X		
Neighbored ver- sions of Base Sw Releases (n to n+1)	X <sup>3)</sup>		
'Remote' versions  n to n+x (x>=2)		X <sup>3)</sup>	

<sup>3)</sup> It is not necessary to support upgrades from version n to n+x (x>=2). However, upgradeability from n to n+1 and n+1 to n+2, etc. is required.

During a system life cycle normally a significant number of *emergency releases* and/or *patches* to fix bugs, close security gaps, etc. have to be installed, in most cases as packages. To make sure these patches don't have any negative side effect or impact to other software components, intensive release testing is required. It is a requirement to the COTS ecosystem supply chain to make sure that these test are executed in realistic environments.

Change Notifications (CN) are required on all SW packages be they upgrades, updates or maintenance releases. The CN must specify all functional or non-functional changes, all solved faults and any changes in internal or external interfaces or documentation. It must also describe any restrictions such as incompatibilities, HW dependencies etc.

SW releases, excluding emergency patches, shall follow a pre-announced release schedule. The certification environment and conditions/restrictions shall be available on request.

SW updates and maintenance releases are unconditionally expected to be backwards compatible. When compatibility between upgrades cannot be achieved, explicit and detailed information on incompatibilities and migration methods is expected as well as tool support.

### **5.3 Standards**

Unfortunately, life cycle management is not a standardized process. As a consequence of this the integration of a product based on components of different vendors is still a challenging task. Standardized interfaces of hardware and software modules can help to reduce the overall integration effort.

Some standardization work regarding Base Platform element interfaces is done or underway by industry initiatives, like PICMG (PCI Industrial Computer Manufacturers Group), TMF(TeleManagement Forum) and SAF (Service Availability Forum). The implementation of these standards in COTS products would reduce the long life cycle management effort significantly and is therefore a strong requirement of NEPs.