White Paper

OSS/BSS reference architecture and its implementation scenario for fulfilment
# Table of Contents

**Abstract** 3

**Operator Challenges** 3
Operator’s business environment 3
Operator’s systems environment 3

**Standardisation work and technology advances** 4

**Architecture** 5
A layered architecture 5
Service oriented architecture 6
Integration enablers 6

**Case: Mobile Services Fulfilment** 7
The Problem 7
End-to-end Automated Fulfilment Solution 8

**Conclusions** 9
Terms and definitions 9
References and trademarks 10

About Nokia 11
About TietoEnator 11
Abstract

Mobile operators face challenges arising both from the changing business environment and from today's rapid diversification of systems. In order to maintain a competitive advantage an operator must rise to these challenges by making the systems environment more flexible and extensible so that it does not present any technical or economic barriers to change. The advances made in understanding operator processes and systems interfaces should be embraced in order to build a sustainable basis for business.

This white paper outlines the OSS and BSS integration reference architecture and its implementation using a service-oriented OSS middleware architecture based on NGOSS, J2EE™ and OSS/J. It further illustrates the benefits of an approach to BSS integration that is built on standard component-based OSS middleware.

Operator Challenges

Operator’s business environment

Operators face an increasingly competitive business environment as the dominant source of ARPU shifts from traditional circuit switched voice, data and SMS messaging into a diverse set of packet data services. In the mobile domain there are a large number of service providers, service life cycles are becoming shorter and operators are under pressure to invest in new service capabilities that promise only modest returns.

In the fixed line domain, data services are replacing PSTN as the main source of revenue.

To retain customers an operator needs an appealing service portfolio along with transparent subscription management, service quality, availability and pricing. At the same time an operator should be able to cut the OPEX going into customer service and systems management. CAPEX and cycles of systems integration should be minimized. Operators are also under pressure to lower their margins when sharing revenue with 3rd party service providers. The business structure of successful operators should be more and more customer focused.

Operator’s systems environment

The shift from legacy systems into component-based OSS and BSS solutions enables faster introduction of new technical capabilities. However, a diverse set of systems brings sizable integration challenges. The number of systems in the OSS and BSS domains is increasing as operators try to avoid getting locked in by one supplier. Furthermore, operators should be able to use the same OSS/BSS infrastructure to manage customers in both the fixed and mobile domains. The number of managed network elements in an operator’s service core is climbing steadily. Operators are under pressure to shift investments from commodity communications enablers into higher value-added solutions, such as ICD (Intelligent Content Delivery), PoC (Push to talk over Cellular) and Presence. The increasing number of external mobile service providers forces operators to open interfaces for service management and activation in order to reduce the need for manual integration work. The crucial point is that both an operator's own systems portfolio and any systems belonging to external parties need to be as pluggable as possible.
Standardisation work and technology advances

In the OSS industry, there are two major initiatives, which complement each other: the New Generation OSS (NGOSS) by Telemanagement Forum (TMF), and the OSS through Java™ Initiative (OSS/J). These two initiatives can be utilized to alleviate cumbersome systems integration.

TMF provides a development framework that can be used to steer the future direction of the OSS systems. OSS/J provides a practical implementation guideline for new OSS applications and interfaces.

**Figure 1. NGOSS and OSS/J.**

NGOSS and OSS/J lay the foundation for OSS middleware components market by defining industry architecture and interfaces. TeleManagement Forum’s eTOM® model provides the processes outline for OSS/BSS systems. The model is widely deployed in the industry.

J2EE, XML and Web Services are the main technologies suggested for use with OSS/J.

**Figure 2. Technology choices.**

The following chapters cover the OSS/BSS reference architecture, which is based on the aforementioned standards and technologies.
Architecture

In order to use the same OSS/BSS infrastructure to manage customers in mobile and fixed domain, evolution from vertical silos to well defined modular, component-based architecture is needed.

The new architecture should not re-invent the wheel. Instead, the systems should utilise mature mainstream IT middleware, such as J2EE, and technologies in order to simplify the infrastructure and provide hardware independency.

Standard OSS middleware enhances the IT middleware/platform with OSS specific functionality, such as network inventory, adaptation to new type of network elements, alarm collection, etc. OSS middleware makes it possible to harmonise the management platform of different OSS systems and this way reduce the OSS/BSS integration effort. OSS middleware hides the network complexity and enables faster introduction of new technical capabilities. Open interfaces for example in the service management and activation area reduce the manual integration work.

Component-based OSS and BSS solutions improve the management of service planning, deployment and operations in a multi-service, multi-vendor, multi-technology environment.

A layered architecture

The traditional approach of operator systems architecture consists of three layers:
1. Business Support System (BSS) layer
2. Operations Support System (OSS) layer
3. Networks layer

In addition to these traditional layers, integration enablers are needed to provide the ability to successfully interconnect OSS and BSS.

The **BSS layer** has a focus towards customers and business partners. The BSS also manages most of the financial transaction processing as well as act as a front-end towards the end-users of the systems.

The **OSS layer** contains the infrastructure for technical network management. The OSS systems are built around product, service and resource inventories.
The **networks layer** contains the network operator’s network infrastructure. Multiple networks can be managed simultaneously by one system.

The OSS and BSS systems should communicate using **integration enablers** in order to maximize application pluggability and overall solution flexibility.

### Service oriented architecture

Service orientation is the principal concept behind the architecture. This means that the functions on each layer are modeled as services as far as possible. Each layer may bundle or split some of the higher or lower layers’ services. Service orientation is especially important in the integration enablers components – even if the service implementation in the network changes, the abstraction layers (provided by OSS) make the change transparent to BSS.

In order to have an ecosystem that efficiently supports service orientation there are some pre-requisites which need to be met:
1. Data managed in the services should be modelled in a consistent way – even if exactly the same data model is not applied through the infrastructure, there should be unambiguous mappings. In the presented architecture the data is modelled following the TMF SID.
2. Interfaces should be consistent – they should be mappable through parameter translations or transformations and compositions of service calls. In the presented architecture the interfaces are implemented following the OSS/J specifications and complementing them as needed for the so far unspecified parts.

### Integration enablers

The integration enablers consist of the integration technology platform (OSS and IT middleware), with both generic framework and OSS/BSS-specific abstraction services, which are accessed through standard APIs when possible (please refer to the picture below).

![OSS/BSS integration enablers](image-url)
Standard APIs form a façade that hides the actual service implementations from the interfacing systems. This enables the systems to be genuinely pluggable. The integration technology platform provides the runtime environment with scalability and usability management facilities.

OSS/BSS integration enablers in brief:

- **OSS/BSS and OSS Access APIs** provide standard interfaces towards OSS applications and middleware software components to enable simple OSS/BSS integration. APIs are mainly based on the OSS/J, when a standard interface exists.
- **Custom API toolkits** (e.g. EAI tools) are used for integrating systems without standard interfaces
- **Standard OSS middleware** provides the generic framework and OSS specific abstraction services such as Service inventory and Service activation.
- **Standard IT middleware**, such as J2EE container and database.
- **Standard service management interfaces** (e.g. Web Services)

Standard OSS middleware provides components, such as

- **Topology** – Network and service topology component.
- **Inventory** – Network and service inventories.
- **Service activation** – Component for interfacing towards different service directories and retrieving activation context.
- **Network activation** – Abstraction service utilizing needed specific service components.
- **Fault management** – Component for alarm monitoring and processing for network elements.
- **Performance management** – Component for performance data collection, KPI calculation and reporting
- **Business Process Engine (BPE)** – Managing OSS sub-processes (such as Service deployment), which are integrated to higher-level business processes.

The actual integration technologies are J2EE / Enterprise Java Beans, Web Services (WS) and an EAI-type message-based integration framework. The OSS/J APIs are implemented as EJBs.

**Case: Mobile Services Fulfilment**

**The Problem**

Mobile services are becoming more and more diversified. They range from basic circuit switched voice service to packet switched data services with multitudes of applications, both in the network and the handset. The services may have complex pricing rules and they require correct settings in multiple independent network elements. Many services are provided by third parties, which are outside the control of network operators.

Future revenues will come from a wider variety of smaller volume services, which are often relatively short-lived. In order to maximise the revenues, the services need to be brought into use quickly. In order to minimise the OPEX the services need to be implemented with minimal manual involvement from the operator. This implies that the service provider must be able to deploy and set up the services automatically and the subscribers must be able to use the services automatically. This paper concentrates on the subscribers’ part of the process.

Ideally the same fulfilment infrastructure will serve both fixed and mobile networks – this case study has been devised with this goal in mind.
End-to-end Automated Fulfilment Solution

Subscribers must be able to activate the services they want easily. Service activation may contain several components: activating settings in the mobile terminal, provisioning usage rights in several network elements, and setting up billing and pricing rules. It may even involve sending an application to the mobile terminal. Everything required to run the service should be part of the fulfilment solution, so that subscribers only need to select the service and accept the activation.

The solution for the end-to-end automated fulfilment consists of the following components:

**BSS layer**
- A self service (web) portal, where the subscriber is able to manage his subscription and activate new services and deactivate unwanted ones
- Customer Relationship Management (CRM) system
- Order management
- Subscriber management
- Pricing and rating information
- Charging & billing system

**OSS layer**
- Resource inventory management
- Service management

**Network layer**
- Network elements
- 3rd party external systems – such as third party content provider systems
Integration enablers

- **Generic framework and specific abstraction services**
  - **Service information discovery** – provides the means to retrieve information regarding a service from all the systems where the information is stored.
  - **Customer/Subscriber information discovery** – provides the means to retrieve information regarding a service from all the systems where the information is stored.
  - **Network activation** – implements service that hides the network activation solution(s) from the specific services needing to use it. The actual interface(s) towards the underlying network adaptation are implemented within this this module. Implementation of relaying the activation information to other OSS and BSS systems, which need the information, is also implemented within the network activation framework.
  - **Service activation** – models and implements the service activation process and its business logic. The process implementation consists of interaction with the service information discovery and customer/subscriber information discovery services, process orchestration functionality and the network activation. The component is configured and instantiated for the all the services to be activated.

- **OSS Access API and EAI integration adapters provide the abstraction services for the benefit of client OSS/BSS systems.**
  - OSS/J interfaces, such as Service activation API, are provided in the OSS Access APIs. They implement the adaptation of that interface towards the specific middleware components that provide the services.
  - EAI adapters are used to integrate legacy systems to middleware components.
- **Standard OSS middleware implemented on top of standard J2EE application server.**

Conclusions

The structured, modular and standards based architectural approach proposed in this paper enables the operator to respond to the challenges arising from the changing business and technology environments. When moving into the proposed direction, the operator will gain a number of advantages:

Firstly, it will be easier to add or remove any OSS/BSS applications, which in turn creates greater flexibility and has an impact on many areas, such as when
- changing to new technologies or vendors
- developing and deploying new services
- improving business process performance – lead-times, quality, costs etc.

Standard middleware component-based approach makes it easier to deploy new services from a product life cycle perspective. It will reduce complexity in the application layers – some basic services are provided by the OSS middleware. Being based on standards, it will lower the cost of deploying and managing OSS/BSS applications over their life cycle.

Terms and definitions

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ARPU</td>
<td>Average Revenue Per User</td>
</tr>
<tr>
<td>BPE</td>
<td>Business Process Engine</td>
</tr>
<tr>
<td>BSS</td>
<td>Business Support System</td>
</tr>
<tr>
<td>EAI</td>
<td>The Enterprise Application Integration. The Enterprise Application Integration style implies that there is a message bus, into through which the systems communicate.</td>
</tr>
<tr>
<td>eTOM®</td>
<td>Enhanced Telecom Operations Map</td>
</tr>
<tr>
<td>NGOSS</td>
<td>New Generation Operations Software and Systems by TMF.</td>
</tr>
<tr>
<td>OSS</td>
<td>Operations Support System</td>
</tr>
<tr>
<td>OSS/J</td>
<td>The OSS through Java Initiative. Industry initiative, which specifies J2EE APIs for OSS.</td>
</tr>
<tr>
<td>SID</td>
<td>Shared Information/Data. Data model in OSS/J APIs. Standardized by TMF.</td>
</tr>
<tr>
<td>TMF</td>
<td>TeleManagement Forum</td>
</tr>
</tbody>
</table>
References and trademarks

Java
Java is a trademark of Sun Microsystems, Inc.

J2EE
J2EE is a trademark of Sun Microsystems, Inc.

Linux
Linux is a trademark of Linus Torvalds

eTOM® and NGOSS
eTOM® and NGOSS are trademarks of the TeleManagement Forum
About Nokia

Nokia is an active member of Telemanagement Forum Board and the OSS through Java Initiative, which brings a new level of openness and interoperability to network and service management environments. This work reflects Nokia’s commitment to encouraging standardization and industry collaboration in order to gain new efficiencies through plug and play OSS solutions.

Nokia is the world leader in mobile communications, driving the growth and sustainability of the broader mobility industry. Nokia is dedicated to enhancing people’s lives and productivity by providing easy-to-use and secure products like mobile phones, and solutions for imaging, games, media, mobile network operators and businesses. Nokia is a broadly held company with listings on five major exchanges.

About TietoEnator

TietoEnator is one of the leading architects in building a more efficient information society and the largest IT services company in the Nordic countries. TietoEnator specialises in consulting, developing and hosting its customers’ business operations in the digital economy. The Group’s services are based on a combination of deep industry-specific expertise and latest information technology. TietoEnator has close to 14,000 experts in more than 20 countries.

The information contained in this document represents the current view of TietoEnator on the issues discussed as of the date of publication. Because TietoEnator must respond to changing market conditions, it should not be interpreted to be a commitment on the part of TietoEnator, and TietoEnator cannot guarantee the accuracy of any information presented after the date of publication.