Service architecture for 3GPP IP Multimedia Subsystem – the IBM and Swisscom proof-of-concept experience

By

Xavier Weibel, technology strategist, Application and Service Enablement, Swisscom Mobile;
Christophe Gourraud, IMS expert, Swisscom Mobile;
Olivier Polette, VP Consulting, Mov'Age and
Lee Longmore, lead IMS architect EMEA, IBM

Contents

2 *Executive summary*
3 *Project highlights*
4 *Architecture overview*
6 *Service Delivery domain*
8 *Device domain*
9 *Service Creation domain*
10 *Observations*
11 *Conclusion*
11 *For more information*

Executive summary

Commencing August 2005, Swisscom Mobile and IBM collaborated to explore the effectiveness of the 3GPP IP Multimedia Subsystem (IMS) for service innovation and cost reduction. Swisscom's rigorous partner-selection process had revealed that both organizations shared a common vision of IMS, and assured Swisscom of IBM's ability to execute against challenging goals and time frames. In addition, Swisscom trusted IBM's commitment to host an integrated service architecture for IMS in its European Telecommunications Solutions Lab (TSL) in Montpellier. This architecture comprises technology components from IBM and a number of other innovative players, notably IMS technologies from Nortel and an event-driven JAIN SLEE platform from jNetX.

Within just five months of starting the exercise, the joint team had succeeded in designing, building and testing a preliminary service architecture for IMS over which initial innovative rich-voice[1] services were demonstrated on 3G devices.

Highlights

It was agreed to collaborate on the creation of two rich-voice services for a demonstration to the Swisscom Mobile executive management.

Project highlights

The first priority of the partnership was to complete an IMS service architecture proof-of-concept, with the objectives of:

- *Analyzing the technical, organizational and time-to-market benefits of a Horizontal Services Architecture*[2] *for IMS for efficiently leveraging core network capabilities*
- *Combining IT process-oriented execution environments with event-oriented communication execution environments to support the transition from application-to-peer (A2P) to peer-to-peer (P2P) services*
- *Leveraging IMS capabilities to realize the efficient delivery of innovative services in conjunction with third parties*

To this end, it was agreed to collaborate on the creation of two rich-voice services for a demonstration to the Swisscom Mobile executive management.

In order to derive maximum value from the exercise, the team set out to address a number of key questions during the proof-of-concept, for example:

- *Can a model-driven service-creation toolset be used to create and deploy IMS-based services?*
- *How can service behavior be directed at run time based on business rules and without any impact on the service developer?*
- *What is a service capability interaction manager (SCIM)*[3] *and does it have a valuable role to play in a service architecture for IMS?*

Highlights

From the outset, all parties shared a common architecture vision based on the premise of mixing reusable services, service components and enablers into innovative services with the ability to customize the behavior of these services at run time.

Architecture overview

From the outset, all parties shared a common architecture vision based on the premise of mixing reusable services, service components and enablers into innovative services with the ability to customize the behavior of these services at run time.

The IBM Service Architecture for IMS was used as a basis for the proof-of-concept and, where necessary, extensions were identified in order to meet the agreed-upon goals. This architecture is based on the IBM Service Delivery Platform (Service Provider Delivery Environment [SPDE] Framework) with extensions to accommodate the specific requirements of the 3GPP IMS architecture. The framework has evolved to keep pace with new design paradigms such as service oriented architecture (SOA).

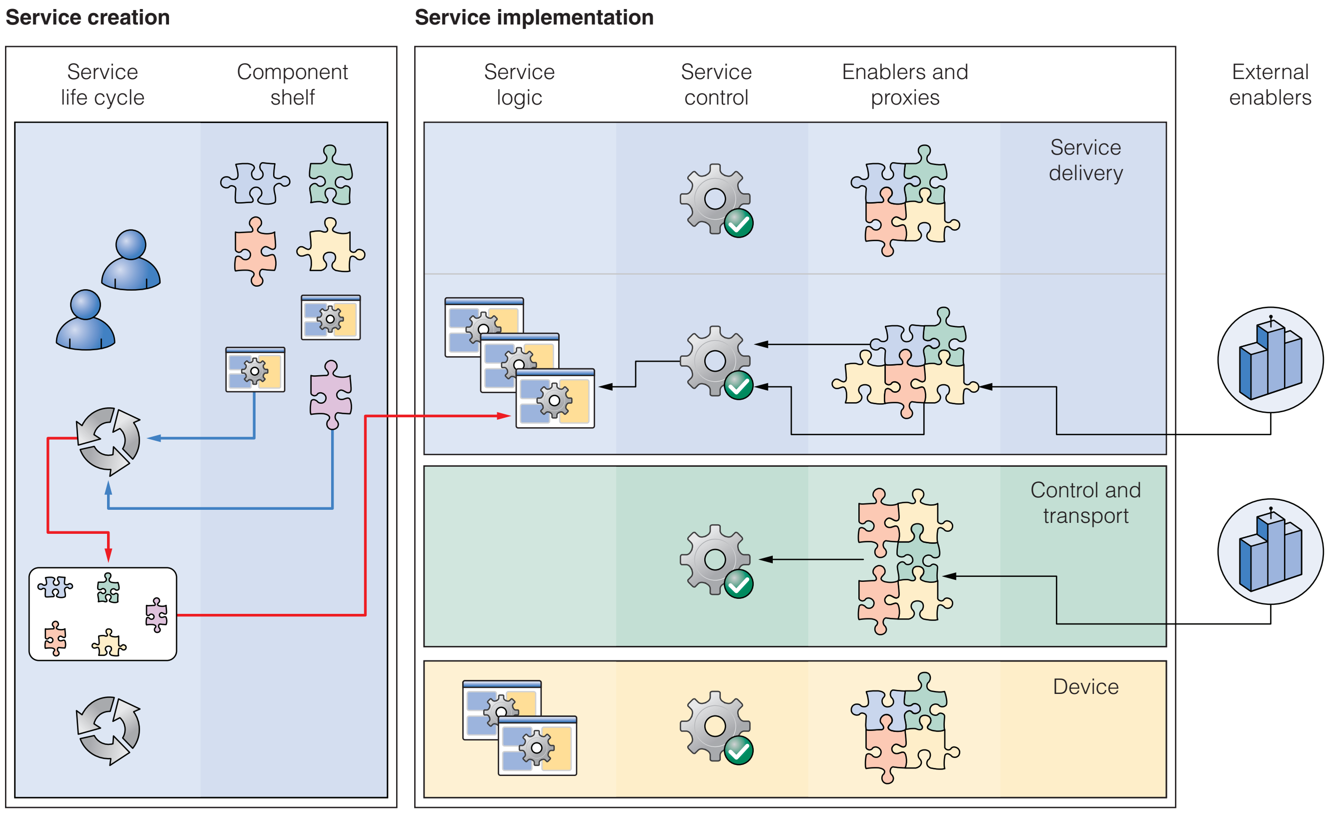

Figure 1. Proof-of-concept architecture overview

Highlights

As illustrated in Figure 1, the architecture implemented for the proof-of-concept comprised two primary environments:

- *Service Execution Environment, which comprises a number of domains:*
  - *Service Delivery—provides an integration, exposure and execution environment for service components and service building blocks. This domain was realized using IBM WebSphere® technology and selected technologies from IBM Business Partners, such as jNetX.*
  - *Control & Transport—provides session control and management capabilities through call session control function (CSCF) and home subscriber server (HSS) nodes and associated enablers, and core IMS protocols, such as Session Initiation Protocol (SIP) and Diameter. This domain was realized using technology from the Nortel converged multimedia services (CMS) architecture linked to Swisscom Mobile 2.5/3G networks.*
  - *Device—includes an execution environment within an end-user device for initiating and interacting with services, and for controlling the behavior of the device in accordance with the needs of these services. This domain was realized using device-client technology developed by IBM in collaboration with Qualphone, an IBM Business Partner, and operating in Motorola 3G mobile devices.*
- *Service Creation Environment, which provides a model-driven and integrated set of tools for creating, deploying and reusing services and software components. The environment includes a well-defined and automated process offering guidance across the service creation life cycle. This domain was realized using an integrated set of tooling from IBM Rational®, IBM WebSphere and jNetX integrated within the Eclipse open framework to provide seamless flow of artifacts between tools and roles through stages of the life cycle.*

The Service Delivery, Device and Service Creation domains are described in more detail in the following sections. Other important domains defined in the SPDE Framework, such as Services Management, Identity Management, User and Service Management and BSS/OSS Integration, were not included in the scope of the proof-of-concept; however, they are areas in which IBM has significant experience.

Highlights

In addition to the domains outlined, a number of enablers were included in the proof-of-concept, including:

- *Presence*
- *Group list management server*
- *Media server*
- *Media gateway*
- *Location server*
- *Address book*
- *Map and route server (third party)*

Service Delivery domain

Figure 2 illustrates the Service Delivery domain designed for the proof-of-concept.

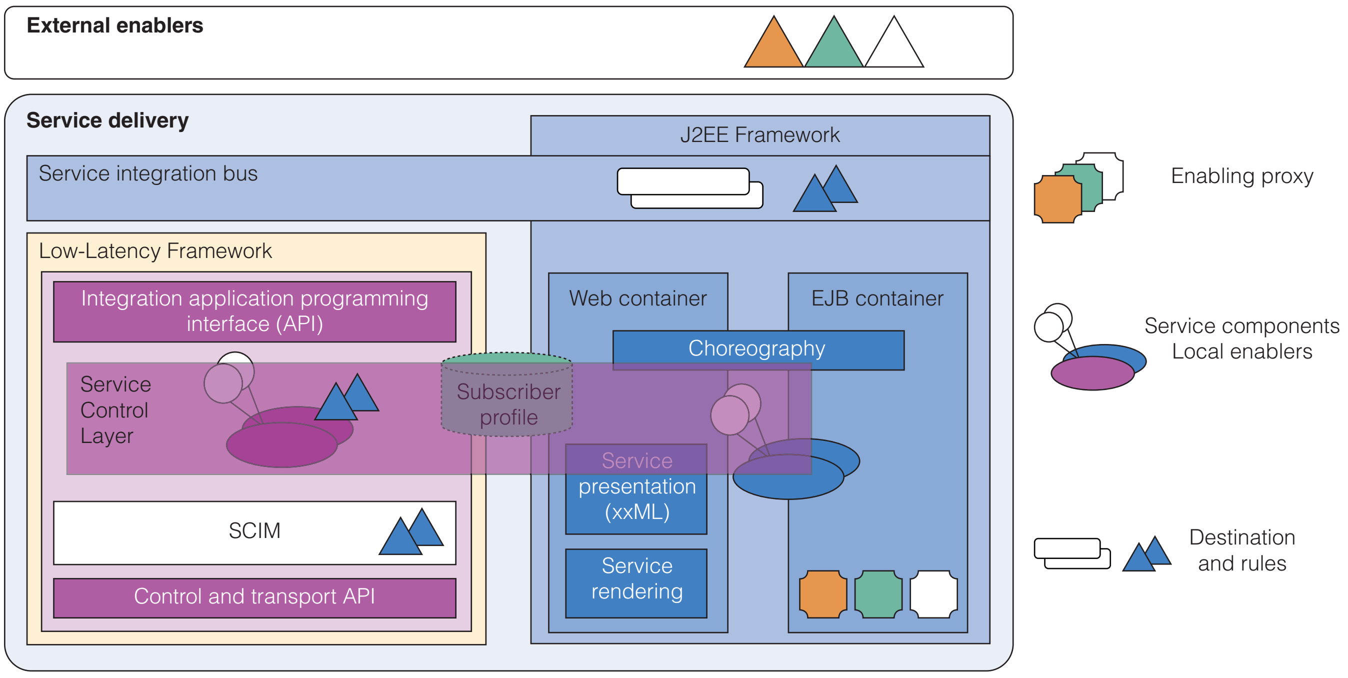

Figure 2. Service Delivery domain

Highlights

The Service Delivery domain comprises five key components:

- *Java™ 2 Platform, Enterprise Edition (J2EE) Framework – this framework provides a powerful execution environment for:*
  - *Running Business Process Execution Language for Web services (BPEL4WS) in order to choreograph all, or a subset, of the components used to deliver a service*
  - *Exposing functionality of enablers through wrapper components called enabling proxies, typically using Web services*
  - *Implementing specific service components (for example, a Web front end) and service building blocks (for example, a Java application)*

  The J2EE Framework is realized using IBM WebSphere technology.
- *Low-Latency Framework – supporting both Java Specification Request 116 (JSR116) and JAIN SLEE application servers, this framework is used to implement service components and service building blocks that are characterized by their use of IMS protocols and in some cases, a demand for low-latency performance. Through its support for core IMS protocols such as SIP, XCAP and Diameter, the Low-Latency Framework provides the gateway between the Service Delivery and the Control and Transport domains.*
- *Service Control Layer – this innovative component provides an intuitive, rules-driven mechanism for dynamically controlling the behavior of a service, for example, policy and charging. Services can be configured to behave in different ways for different users, without placing any demands on the developer of the service. The Service Control Layer is also key to controlling usage of P2P multimedia services or communications that would otherwise be beyond the control of the operator. The Service Control Layer is realized using the capabilities of Service Delivery domain.*

Highlights

- *Service Capability Interaction Manager (SCIM) – using predefined rules and dynamic user- and service-profile information, the SCIM provides a mechanism for chaining SIP-based services and service components into a SIP dialog at run time. It can also be used as a mechanism to enable the collocation of subscriptions and services in order to address performance, scalability and quality-of-service needs. The SCIM is realized using the Low-Latency Framework.*
- *Service Integration Bus – offering powerful asynchronous messaging facilities and support for key specifications and protocols, such as Web services, HTTP and Java Message Service (JMS), the bus provides a common exposure and integration capability for service components and service building blocks residing in the J2EE and Low-Latency Frameworks, as well as for external enablers and third-party services. The Service Integration Bus is realized using the J2EE Framework.*

Services are likely to comprise multiple components, both internal to and external to the Service Delivery domain.

Services are likely to comprise multiple components, both internal to and external to the Service Delivery domain. For example, a service might be initiated by a third-party service provider using BPEL4WS hosted in the J2EE Framework. This in turn can call a low-latency component (exposed as a Web service to the BPEL4WS component through the Service Integration Bus) to initiate an outbound SIP dialog with the subscriber's device using the Control and Transport domain. The device can itself host components that provide the processing required for the incoming SIP dialog.

Device domain

In order to achieve a consistent "look and feel" for services and device-side reuse, a device client was developed for the proof-of-concept. In addition to supporting a range of protocols, such as SIP, XCAP and HTTP, the device client also provides access to the host device to control its behavior (for example, call manipulation, GUI controls).

Several reusable service building blocks were developed for the device client, including, for example, a presence-aware buddy client integrated with the group-list-management server and presence enablers.

Highlights

Both the services and supporting end-to-end platform architecture were modeled in IBM Rational Software Architect software — a key prerequisite to making good architecture and design decisions and to enabling reuse and automation.

Service Creation domain

A major focus of the proof-of-concept was to investigate the extent to which integrated tooling can facilitate the service-creation process and promote reuse. To this end, business analysts, designers and architects adopted a model-driven approach using Unified Modelling Language 2 (UML2)-capable tooling. Both the services and supporting end-to-end platform architecture were modeled using the IBM Rational Software Architect product – a key prerequisite to making good architecture and design decisions and to enabling reuse and automation. Significant emphasis was placed on customizing the tools and service-creation process for IMS, as well as identifying areas for software automation. For example, the IBM Rational Unified Process for SOA and IBM Rational Software Architect products were customized for IMS, and a Rational transformation was developed to automatically generate service components for the low-latency environment from a UML finite state machine model.

Proof-of-concept services overview

As a proof-of-concept, Swisscom and IBM jointly demonstrated the creation and deployment of several innovative multimedia services, including:

- CallTones – *an entertaining service that allows users to announce themselves with multimedia content (ring tone, picture, video, text and so on) on the receiving party's handset when they place a call. CallTones is a combinational service that seamlessly chains the delivery of multimedia content with the initiation of a conference, using presence information. It is an example of straightforward multimedia content sharing that uses IMS enablers.*
- Infopush – *a service that a subscriber can call to query any multimedia content (routing and maps, weather reports, movie programs and so on). Infopush is an enriched-voice service that mixes voice communication with P2P real-time delivery of multimedia content from independent third-party providers. Infopush simplifies the access to multimedia content:*
  1. *The subscriber calls a call center and interacts with an agent to order and retrieve the content.*
  2. *The agent possesses all required information to deliver the requested content (location of the subscriber, mobile station international ISDN [MSISDN] number, characteristics of the device and so on).*
  3. *The voice communication continues while multimedia data is pushed to the device so that the subscriber can acknowledge the delivery and if necessary, request more-detailed information.*

Highlights

Although comprehensive, the 3GPP IMS architecture does not address key architecture domains such as those described in this paper. Having a well-defined architecture for these and other domains is vital if the promises of IMS are to be realized.

Observations

A number of observations were made during the exercise concerning architecture and service delivery within an IMS environment. Some examples are as follows:

- *Although comprehensive, the 3GPP IMS architecture does not address all of the necessary architecture domains, including those described in this paper. Defining the architecture for these and all other domains is vital if the promises of IMS are to be realized.*
- *To realize reuse and achieve time-to-market reductions requires investment in service-creation tooling, such as capturing IMS domain expertise, as well as an architecture that supports runtime service composition, and design time and runtime reuse. This enables operators to think "horizontally" when creating a service through the extensive reuse of assets.*
- *Subscribers will configure and interact with services through a number of device types, both wireline and wireless. From a subscriber's perspective, it is important to provide an intuitive and consistent user experience across devices. From a technical perspective, it is important to strive for a modular device client architecture that can be reused and easily ported to new device types as they are introduced.*
- *SIP can be a "verbose" protocol and as such, can have implications on performance. On one hand, good service design should seek to minimize the exchange of SIP messages and, where more appropriate, combine the use of SIP with other protocols such as HTTP, for uses such as content download. On the other hand, the usage of SIP extends potentially beyond the strict usage employed by operators, because it can be used to achieve a significant part of the control also required in an IT environment.*
- *The capacity to reuse process-oriented or event-driven features of the execution environment enables operators to implement services in ways that were previously not possible. For example, it becomes possible to realize the benefits of combinational services—that is, combining services in flexible ways to form new services, for example, mobile gaming with conferencing.*

Highlights

The proof-of-concept proved that it is possible to implement a service architecture for IMS that facilitates the rapid assembly of innovative combinational services.

Conclusion

The proof-of-concept proved that it is possible to implement a service architecture for IMS that facilitates the rapid assembly of innovative combinational services. From a service creation perspective, the exercise confirmed that business analysts, software designers and architects must be equipped with common tooling and processes that enable differing levels of abstraction, cooperation and design for reuse. A flexible service creation environment alone, however, is not sufficient; rules- and profile-driven "horizontal" components in the service execution environment—such as the choreography engine, Service Control Layer and SCIM—all play an important role in adding additional layers of flexibility to user services.

For more information

To find out more about IBM's IP Multimedia Subsystem solutions, go to **ibm.com**/industries/telecom/doc/content/landingdtw/1234109102.html

[1] Extension of voice services with other media types and data services to further increase average revenue per user (ARPU).

[2] An architectural approach to facilitate the reuse of common capabilities so as to reduce time to market and flexibility, increase return on investment (ROI) and improve the end-user experience.

[3] As defined in 3GPP TS 23.218.

G299-0903-00