Web Service Applicability in Telecommunication Service Platforms

Alberto Baravaglio[1], Carlo Alberto Licciardi[2] and Claudio Venezia[3]

[1]Telecom Italia Lab,
Via Reiss Romoli 274, 10148 Torino, Italia
alberto.baravaglio@telecomitalia.it

[2] Telecom Italia Lab,
Via Reiss Romoli 274, 10148 Torino, Italia
carlo.licciardi@telecomitalia.it

[3] Telecom Italia Lab,
Via Reiss Romoli 274, 10148 Torino, Italia
claudio.venezia@telecomitalia.it

Abstract: **The paper describes how a Telecom Operator can leverage on Web Services to provide integrated services which span over several technologies. In addition it addresses how SIP protocol (Session Initiation Protocol) can be used along with Web Service technologies to enable a new set of integrated IT and Telco services. The paper highlights the benefits and drawbacks of application of Web Service paradigm in a Telco environment and the results of research and prototyping activities carried out within Telecom Italia Lab (the R&D of Telecom Italia group).**

Keywords: Web Service, SIP, SOAP, UDDI, NGN TLC Network, Service Oriented Architecture

I. Introduction

Today's telecommunication and information services are delivered through separate, vertically integrated, service specific networks. For instance:

- Entertainment being delivered over broadcast, satellite and CATV systems.
- Mobile telecommunications being delivered via a number of base-stations.
- Information services provided over the Internet (WEB), via WAP (mobile terminal) or via SMS/MMS
- Fixed telecommunications (and most data communications) being delivered over the leased line and "telephony" access equipment.

The conclusion is that there is currently very little convergence but at the transport layer. At the service layer there are almost no shared services, specifically because each network has its own vendor proprietary services platform. This vertical silos like approach has an extremely debilitating effect on a service provider's ability to develop revenue streams from rich services that span the networks. The current vertically integrated networks are expected to migrate to horizontally layered structures supporting markets and businesses of the future [7]. However in the new networks rollout (e.g. 3G, SIP) there is an obsession with finding the "killer" service or application that can economically justify new infrastructure. Indeed this is a questionable approach; it complicates service integration and does not help at all in speeding up the service delivery process.

"Operators must recognize that this wireless digital infrastructure will become embedded into everything that individuals and groups within our society undertake – not just a single service, like video or gaming. Birth of the Killer Environment. By implication, therefore, wireless operators should be creating a killer environment." [Nicolas Foggin, Director of Strategy & Futurology, Orange SA - Alcatel Telecommunications Review, 2002]

At the control layer Session Initiation Protocol is considered the converging protocol for call and message signaling. Either fixed or mobile networks will leverage on SIP for providing integrated capabilities. SIP will improve the ability to build new services and will play the role that Web Services (WSDL + SOAP) are playing in the IT world [6] (the universal glue).

Telecom Italia Lab (TILAB) is facing up the challenges of integrating IT resources into SIP services and making SIP resources available to the IT community.

SIP networks are characterized by services that can be accessed through very heterogeneous terminals provided with a SIP stack on board but not always SOAP enabled. Moreover, instead of what usually happens in the discovery phase of SOA, in Telco environments pushing service information to terminals is a primary facet to make users aware of new available services. To supplement a SIP Network with a Web Services Architecture it is necessary to extend it to meet these critical requirements.

Finally, the main features of a telecommunication service inferred by a typical communication service scenario are:

- asynchronous interaction with the service
- very interactive and incremental service sessions
- typically person-to-system-to-person interactions

II. Overall Architecture

Tilab solution for SIP networks is made up of a set of systems which interact through the SIP protocol and a service bus (see Figure 1). Giving a brief description of the most important architecture entities and interactions is useful to outline the role that Web Services play in such environment.

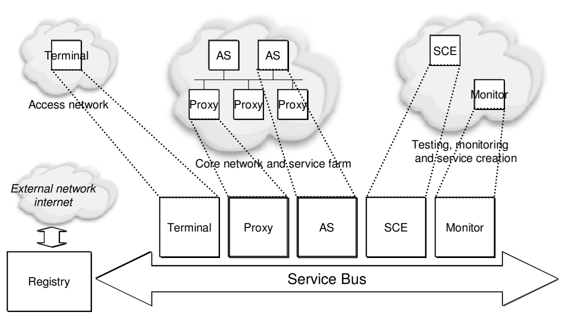


Figure 1. SIP components interaction

Within a SIP network a user agent accesses SIP services provided by some AS. A SIP application server is the core element of the architecture which encompasses all the typical elements of SOAs such as publish, discovery and binding functionalities combined with non functional elements such as monitoring or deployment facilities.

StarSIP is a SIP-based software platform for the creation, fast-prototyping, execution and control of multimedia services. It supports the creation, the deployment and control of a wide range of next generation services also integrating voice-over-IP (VoIP) technology. StarSIP provides a set of Java API to access network resources for creating services.

The StarSIP Service Bus is a SIP based middleware which addresses deployment, monitoring and provisioning of services and application to SIP network elements and terminals (see Figure 2).

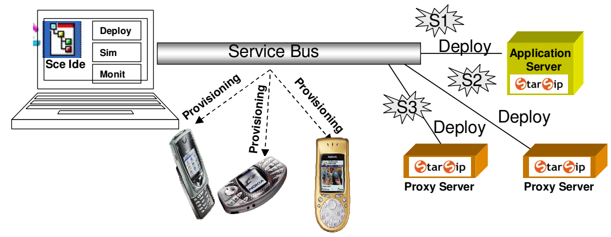


Figure 2. StarSIP service bus

SIP service bus allows information push via a "publish subscribe" model. This push mechanism may be used to address the transport of different types of information such as monitoring data or service descriptions.

The StarSIP Registry collects relevant information from a SIP network, stores and distributes it. Circulating on the Service Bus, this information manages both service and network elements descriptions and states. The StarSIP Registry is available both for network resources (where services are running), service managers (watching services behavior) and service users (interested in invoking services).

III. TLC service concepts

Telecommunication services may be grouped in two categories:

- Telco specific services
- hybrid services

The main characteristics of the services belonging to the first category are strong real-time requirements and asynchronous interactions. These are typically deployed in network environments strongly controlled by Telco operators. Voice mail, call forwarding and ring back tone are typical examples of this service category.

Hybrid services integrate different IT resources with Telco functionalities. While the Web Service model is unsuitable for the first category it can fit the second. The following is an example of a simple information retrieval hybrid service:

1. The user invokes the service by sending an SMS whose body contains the information needed to retrieve the closest merchant of a particular category (e.g. restaurant, bar or cinema).
2. The service localizes the user, retrieves the information requested and replies with a SMS containing the information retrieved.
3. Furthermore the user can send another SMS to be connected with the found merchant via an audio call.

Given this service categorization, two scenarios are feasible:

- the Telco operator offers Web Service interfaces to provide access to its resources to a third party
- the Telco operator integrates IT resources to enrich its service portfolio offer

Depending on market opportunities both the scenarios must be supported by a target service delivery platform.

IV. Web Service enabled architecture

Given the scenario depicted above let us describe the evolution of a SIP network to enable Web Service adoption. To reach this goal we have to take in account several critical issues:

- how to invoke Telco services from SOAP clients which don't provide a SIP stack
- how to correlate discovery mechanisms based on UDDI with the ones adopted in our SIP network
- how to invoke SOAP services from clients which don't provide a SOAP stack

Figure 3 describes how the integration can be carried out. We need two gateway functions: the Registry Gateway to grant the synchronization of UDDI [1] and SIP registry and the SIP/SOAP adapter to enable service invocation over different protocols. The SIP adapter is responsible for adding a SOAP interface to SIP services and vice versa.

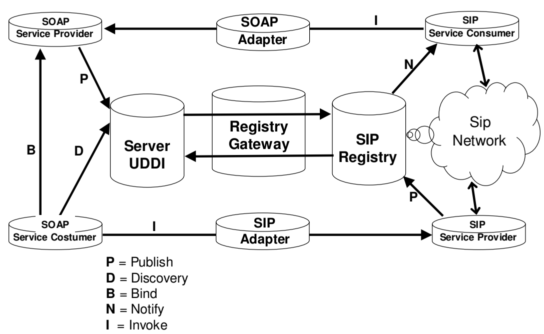


Figure 3. Web Service integration architecture

A target service delivery platform is therefore responsible either for business logic execution or for protocol adaptation. This will allow users to access any service over any protocol (i.e. SIP, SOAP) depending on their needs or device capabilities.

A. Service registry

In SOAs, publish and discovery processes are supported by a service registry. While UDDI [4] is the standard technology for publishing and discovery of Web Services, our solution for SIP networks relies on a specialized registry based on the SIP protocol. Our approach to publication and discovery is slightly different from the one proposed by UDDI: the primary goal of our registry is the asynchronous push of service descriptions to the clients, according to some service categorization criteria. The solution is strongly based on the publish/subscribe mechanisms natively supported by the SIP protocol, and allows real time distribution of service descriptions to a wide range of clients.

The requirement of integrating Web Services technologies in our SIP environment has to go through the integration of the two service registry types, enabling the publication of Web Service descriptions on a SIP network and vice-versa.

The goal has been reached by adding a Registry Gateway in the architecture, that allows the distribution of normalized service descriptions to different registries, transporting and, whenever necessary, transforming the service description information that are common to the two registries (see Figure 4).

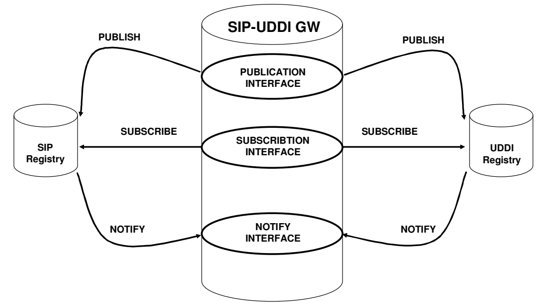


Figure 4. Registry Gateway overview

The Registry Gateway is extensible to other registry types thanks to an adaptation layer introduced between the gateway engine and the registries (see Figure 5).

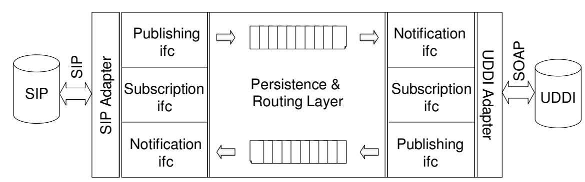


Figure 5. UDDI gateway adapters

The Registry Adapters (SIP or UDDI) connect to the target registry and their role is:

- publish service descriptions
- receive notifications of new services (both synchronously or asynchronously)
- transform service descriptions from the registry representation to the internal one and vice versa

The Persistence & Routing Layer queues up the service descriptions and distributes them to all the Adapters subscribed.

B. Secure and personalized access to UDDI registry

The UDDI specification is very much focused on the information model that enables a convenient categorization of the published services, but it does not address the following critical requirements:

- authorization: there is no mechanism in UDDI that allows defining and enforcing complex authorization policies for service requestors when inquiring the registry and retrieving the details of the services
- reference validity: UDDI does not guarantee that the service reference returned to the application (in response

to a Get Service operation) is correct, i.e. there is effectively a Web Service pointed by it

- late binding: since service references are published as static data, Web Services are forced to be up and running continuously on a given URL. No dynamic instantiation of services and references is therefore possible. For this reason UDDI does not provide mechanisms by which a service reference returned to an application may depend on load balancing policies on the service instances
- personalization: UDDI does not support any form of personalization, i.e. the result of a specific query is the same for any requestor

In order to meet these requirements a "UDDI proxy" has been added to the Web Services architecture (see Figure 6). The proxy routes inquiries from a client application to the UDDI registry and provides additional and personalized capabilities, mediating the access to the actual UDDI registry.

The proxy implements (among the others) the following capabilities:

- control the access to the information contained in the UDDI Registry allowing/denying the access based on a Service Requestor's Authorization Profile
- dynamically create the Web Services instances, assuring the existence of the Web Service
- personalize the Web Services instances based on the Service Requestor identity
- balance the load on the Web Services instances allowing a higher level of availability and fault tolerance of the Web Service

The main characteristics and advantages of UDDI proxy solution are the following:

The proxy exposes standard UDDI interfaces to the applications, so that the interactions with it are exactly the same as the ones with ordinary UDDI registry (i.e. UDDI clients use the same UDDI API). This ensures compatibility with all existing applications that are based on UDDI standards, since UDDI data model is not modified; therefore the client application is completely unaware of proxy existence.

The solution is modular and has minimal impact on the pre-existing architecture since it does not require modifying the existing elements. In fact it only implies to:

- add a separate node (the proxy),
- reconfigure the applications by providing the reference to the new node
- configure the UDDI registry to accept inquiries from a new client (the proxy itself).

Different types of capabilities can be implemented in the proxy (e.g. data validation, access control, caching, load balancing, logging, etc) allowing maximum flexibility.

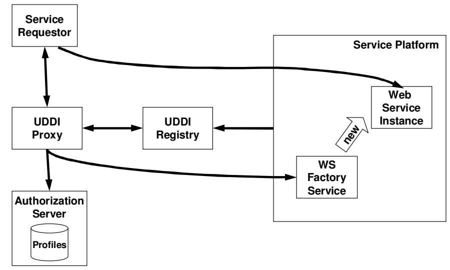


Figure 6. UDDI proxy architecture

This approach can also be applied to make semantic extensions to a UDDI registry. This will allow enriching search criteria in UDDI to enable the discovery and composition of user centric services.

C. Telco services exposure via Web Service standard interfaces

Standardization initiatives like those leaded by Parlay [3] or OMA, have been specifying several standard Web Service interfaces to the most common Telco functionalities, opening them to the IT community by means of widely known interfaces like Web Services.

Hence, if on one hand we lose the granularity of the proprietary interface, on the other we gain in terms of interoperability.

This is the list of Web Services we provide in compliance with the relevant standards (whenever available):

- *Third party call* provides the capability to initiate a call between two actors generated and managed by a third party
- *Multi media conference* provides the capability to initiate an audio/video conference with two or more actors within a session
- *Messaging* is a set of Web Services which provides the capability to send Instant Messages, SMS and MMS
- *Presence* provides the capability to retrieve user availability information in a network domain
- *Users' provisioning* provides the capability to interact with a Data Provisioning DB System by means of retrieving and storing user profiles information supporting various communication protocols and devices.

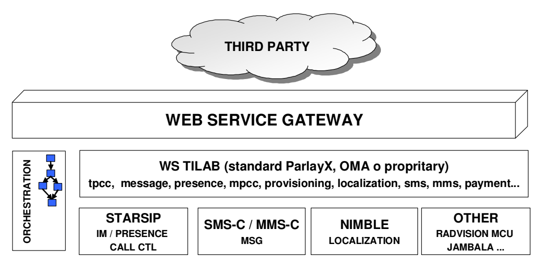


Figure 7. Telco Web Service exposure

The effort of exposing Web Service interfaces over different network systems has pointed out a set of shared non functional needs such as security, billing, monitoring and tracing. These considerations made us aware of the need for a mediation layer responsible of all the access control activities which are mandatory in a production environment (see Figure 7).

V. Service composition

In order to provide Value Added Services (VAS) the service platform must be endowed of a composition engine which easily allows building new VAS starting from a set of components. Using Web Service orchestration is a feasible approach for this objective.

Orchestration is a way to describe the interactions and connections among Web services defining a higher level business process. Standard body (OASIS) has defined a language to describe orchestration, namely Business Process Execution Language (BPEL) for WS. This language is suitable to describe a workflow that is executed on a central BPEL engine, which controls execution and message flow.

Our research on BPEL4WS [5] language and related service execution environments highlighted some drawbacks in the implementation of some TLC services using a workflow paradigm based on Web Service:

- Many TLC services have strong performance requirements such as low latency time and high throughput which BPEL engines don't meet
- Component interactions are strongly asynchronous and Web Services still lack in supporting this interaction model
- From the language semantic perspective there is a set of TLC relevant workflow patterns which BPEL4WS doesn't support natively

On the other hand BPEL4WS suits long running business processes with loose requirements in terms of performance.

The above considerations led us to develop an event-based Service Logic Execution Environment - named StarSLEE, and the corresponding SCE (StarSCE).

A. StarSIP Service Logic Execution Environment (StarSLEE)

The StarSLEE engine (see Figure 8) supports the execution of event-based service logics that span across different application platforms and networks.

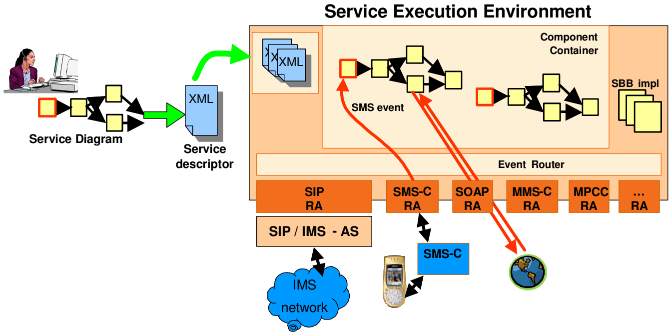


Figure 8. StarSLEE

The service logics are described through the composition of elementary components (Service Building Blocks), with an event-oriented XML based language.

The service descriptors are generated by a Service Creation Engine in a graphical way. The StarSLEE engine is modeled according to the emerging JAIN Service Logic Execution Environment (JAIN SLEE) specification [8] and can be easily connected to different application platforms or network elements via a "Resource Adaptor" mechanism. A SOAP resource adaptor is responsible for the SOAP message exchange between the platform and external Web Services; on the other hand the services provided by the platform can be exposed as Web Services. Another lesson learned exposing Telco services as Web Services is the importance of separating two service facets: service core logic and service interface. This separation can be obtained by the introduction of *Façade* elements which "encapsulate" the service core logic. These *Façade* elements are responsible for the interactions between a service and any external resource without concerns of the protocol (i.e. SIP, SOAP). The same user may want to use the same service over different protocols depending on his actual device capability (i.e. Web Service, SMS, Instant Message).

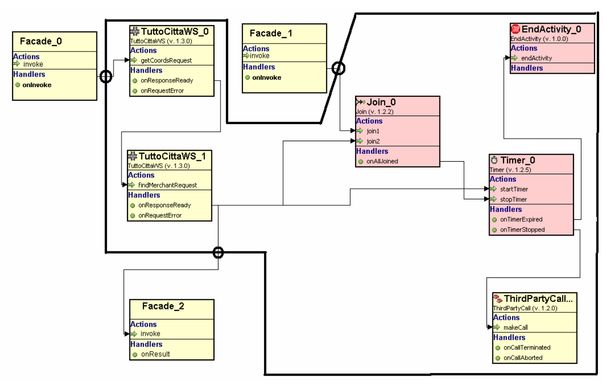


Figure 9. Service description

The figure above depicts a service obtained with the graphical service creation environment. The service design facility highlights this distinction by bordering the service core logic. Only the *Façade* elements placed outside the border are in charge of outbound/inbound interactions and, according to the various bindings they are instantiated via a specific resource adaptor.

VI. Conclusions

The process of integrating Web Services in Telco platforms and services has shown that the Web Service orchestration approach fits very well to Business Process Oriented services which neither have strong real-time requirements nor encompass interactions which are mainly asynchronous, typical of Telco world. Probably new standards like WS-Notification or OASIS Asynchronous Service Access Protocol (ASAP) will improve the applicability of Web Services in the Telco environment. On the other side, event based asynchronous engines (such as JAIN SLEE) are designed for Telco environment and are capable of integrating also Web Services.

On the service discovery side, UDDI has shown some limitations concerning personalization and security that have to be addressed with ad hoc solutions. Furthermore, on a SIP network, service discovery can leverage on the SIP protocol to enable a push paradigm, which is generally more suitable for Telco networks because it is less bandwidth consuming and uses more effectively the native signaling protocol.

References

[1] Organization for the Advancement of Structured Information Standards, "Introduction to UDDI: Important Features and Functional Concepts", 2004

[2] J. Chung, K. Lin, R. Mathieu, *"Web Services Computing: Advancing Software Interoperability"*, IEEE Computer, pp 35-37, 2003.

[3] Parlay X Working Group, "Parlay X Web Services White Paper", the Parlay Group, 2002.

[4] UDDI Spec Technical Committee, "UDDI Version 3.0.2", Organization for the Advancement of Structured Information Standards, 2004.

[5] Organization for the Advancement of Structured Information Standards, "Business Process Execution Language for Web Services (Version 1.1)", 2003

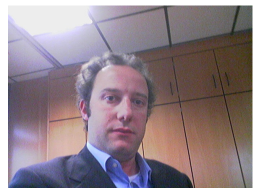
[6] P. Lago, C.A. Licciardi ET alii, "An architecture for IN-Internet hybrid services", *Computer Networks Journal Special Issue on Intelligent Networks and Internet Convergence*, Elsevier, Vol. 35 pp. 537-549, 2001.

[7] P. Falcarin, C.A. Licciardi, "Analysis of NGN service creation technologies", *IEC Annual Review of communications*, volume 56, 2003.

[8] Sun Microsystems, "JAIN SLEE Specification - JSR 22", 2002

Author Biographies

Alberto Baravaglio received a degree in Computer Science from University of Turin (Italy) in 1993. In 1995 he joined Telecom Italia Lab (formerly CSELT - Turin, Italy, leader company in research and development for Telecommunication) where he worked in several domains IT, Telco and mixed. In the first years he has been working on ERP systems and on the topic of system integration, now he is currently working in projects in the area of Service Lifecycle Management dealing with new paradigms of service creation, execution and invocation



Carlo Alberto Licciardi received a Dr. Ing. degree in electronic engineering from Politecnico di Torino (Italy) in October 1990. In 1992 he joined Tilab (formerly CSELT - Turin, Italy, leader company in research and development for Telecommunication)) where he has worked in long term aspects of Intelligent Network and in the design of software architecture for the provisioning of Advanced Telecommunication Services. He has contributed to standardization activities (ITU-T, 3GPP, Parlay, JAINSLEE and OMA) and to worldwide research projects (RACE, ACTS, TINA-C and EURESCOM). He is currently involved in Mobilife WWI project, where is responsible for the TILAB trial site. He is currently leading internal research projects in the area of Service creation, execution and invocation and in innovative context aware application and services.

Claudio Venezia was born in Turin April 29, 1974. He received a degree in Economics from University of Turin (Italy) in July 1998; afterwards he gained technological formation and certification on operating systems and OO programming. He worked as technology Consultant in several domains (Banking, Automotive, and E-commerce). In 2002 joined Telecom Italia Lab (formerly CSELT - Turin, Italy, leader company in research and development for Telecommunication). He has been contributing to standardization activities (JAINSLEE) and working on European Projects (SeCSE). His research interests include the definition of new paradigm for service creation and execution by using open APIs, web services technologies and scripting languages XML based.