A JSLEE based Service Creation and Service Delivery Framework for value-added services in Next Generation Networks

Thomas Eichelmann, Ulrich Trick
Research Group for Telecommunication Networks
University of Applied Sciences Frankfurt/M
Frankfurt/M, Germany
{eichelmann, trick}@e-technik.org

Woldemar Fuhrmann
Department of Computer Science
University of Applied Sciences Darmstadt
Darmstadt, Germany
woldemar.fuhrmann@h-da.de

Bogdan Ghita
Centre for Security, Communications and Network Research
Plymouth University
Plymouth, United Kingdom
bogdan.ghita@plymouth.ac.uk

***Abstract*—This paper describes a framework for an easy and timesaving generation of value-added services in Next Generation Networks and introduces new possibilities for the service creation and service execution through an extension of the common telecommunication real-time execution environment JAIN SLEE. Value-added services are described with graphical development tools. This offers an easy and fast graphical service design. Through the introduction of extensions to the execution environment the service is automatically composed from predefined and pre-deployed components. These extensions also introduce new possibilities for the composition, execution, management, and reconfiguration of value-added services.**

JSLEE; value-added services; service creation; BPEL; NGN

I. INTRODUCTION

The telecommunication market always demands new value-added services. New service ideas have to be developed and offered to the market before their competitors can react. This requires that the development cycles of the value-added services are as short as possible. Complex communication protocols and complex service development leads to the fact that experts are required for the development of value-added services. But to be able to respond to the needs of specific customer groups, the development of customer specific services should be possible also for non expert developer. Furthermore, a Service Delivery Platform (SDP) must satisfy particular requirements, which exclude the usage of web-services or similar approaches. An SDP which offer multi-protocol support together with a high-throughput and low-latency environment is required.

The solution presented in this paper introduces a service creation and service delivery environment that solves the addressed problems. The Service Creation Environment (SCE) offers graphical user interfaces to describe the value-added service and hide the underlying heterogeneous communication networks. Therefore, the developer does not need any detailed knowledge of communication protocols and is able to focus on the application logic instead. For the service description a language that has been optimized for business processes is suggested: the Business Process Execution Language (BPEL) [1]. BPEL however has not been developed for real time communication services in heterogeneous networks. Therefore, from the business process description a value-added service is generated. The SCE allows a reconfiguration of the service logic even during the execution of the service. The SDP introduces extensions to the Java APIs for Integrated Networks (JAIN) Service Logic Execution Environment SLEE [2]. JAIN SLEE also called JSLEE is a common telecommunication real-time execution environment. The extensions offer an automated service generation, extended service management capabilities, a service execution layer, and a layer which handles the resources and the protocols. This leads to new opportunities for rapid and efficient service creation using a new SCE with higher level of abstraction and automated service generation.

The paper is structured as follows. Chapter II starts with a discussion on some related work in this area of research. Chapter III offers a short overview of JSLEE and discusses the advantages and disadvantages of JSLEE. Chapter IV introduces the new Service Creation and Service Delivery Framework with the new layered structure within the service creation and service delivery framework and gives an example how services are created and executed. The paper ends with the conclusion in chapter V.

II. RELATED WORK

In the work: “Orchestration in Web Services and Real-Time Communications”, L. Lin and P. Lin [3] describes an approach to enable a service creation environment for complex (orchestrated) real-time communication services through a service broker on top of Next Generation Networks (NGN).

The goal of the research is to combine the emerging Web/telecommunications service space. They take a hybrid approach for converged voice-data application by adding Web service interfaces to real-time communication flows. This Web services can be orchestrated along with other Web services in BPEL. This work shows an approach how communication services can be composed with BPEL. In our approach BPEL is only used for the service description of the service. In contrast, no real BPEL engine is required. We also offer a framework for an automated generation, execution, configuration and management of the service.

In the paper: "Creating Value Added Services in Internet Telephony: An Overview and a Case Study on a High-Level Service Creation Environment" [4] a case study of a high level graphical SCE is described. It consists of a graphical user interface (GUI). A service can be developed from the eight functions: Start, Timer, Call, Loop, Join, Sync, Play and End. The service logic can be developed with these functions within the GUI. The scope of the experiment is limited to services that originate calls, like Wake-up Call, Call Center or Third Party Call. The services can be executed in a self-developed SLEE that is based on a Parley/Session Initiation Protocol (SIP) Gateway implementation. Therefore the experiment only supports the SIP protocol and only allows basic telephony services. Furthermore the services have to be compiled before execution and cannot be composed from existing components.

III. Discussion on the JSLEE Framework

To understand the proposed extension of the JSLEE framework, the foundations of JSLEE should be described and the strengths and weaknesses should be clarified. JSLEE is a high throughput, low latency event processing application environment for communication services. It allows the implementation of scalable high available communication services. The access to network resources is offered by Resource Adaptors (RAs). The service components are called Service Building Blocks (SBBs) "Fig. 1". A service consists of one or more SBBs. These SBBs are deployed on the JSLEE Application Server. Invocations are transmitted asynchronously via events. The Service Transport Layer abstracts different protocols in order to enable upper service layers to be independent of a specific protocol. Therefore it supports several communication networks.

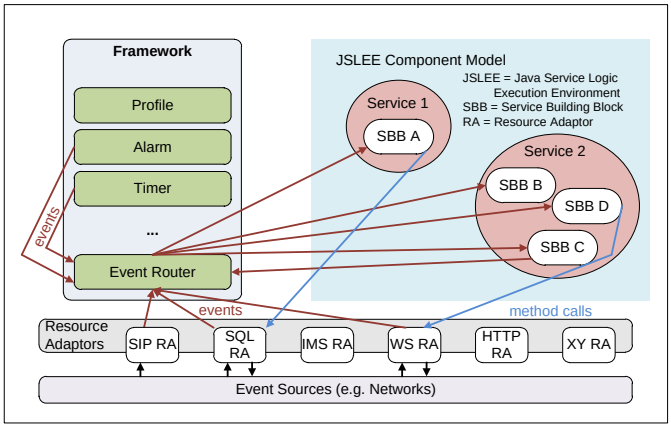

Figure 1. JSLEE Component Container

RAs abstract the underlying infrastructure. When a communication protocol message is received, the corresponding RA translates this message into a Java event class. Afterwards the event is passed to a JSLEE element that is called event router. The event router looks up the services that are subscribed for the specific event and delegates the event to these services. Accordingly the service itself is able to react on the event and to create an answer by using defined Java interfaces. The answer is translated to a communication protocol response by a RA.

As described this framework is ideal for value-added services. But there are some problems which also need to be discussed. The development of JSLEE-based services is very complex. The developer of a value-added service requires expert knowledge of java, JSLEE and of the underlying protocols. The required experts are rare and expensive. Developers which are new to JSLEE suffer from the steep learning curve. This leads to an expensive and long development of new services.

IV. Service Creation and Service Delivery Framework

The architecture is designed to provide advanced possibilities for service creation and execution. It is divided into the layers: Service Creation Environment (SCE), Service Execution Environment (SEE) and the Service Transport Layer (STL) "Fig. 2". Parts of this architecture are based on the architecture developed in the TeamCom project [5].

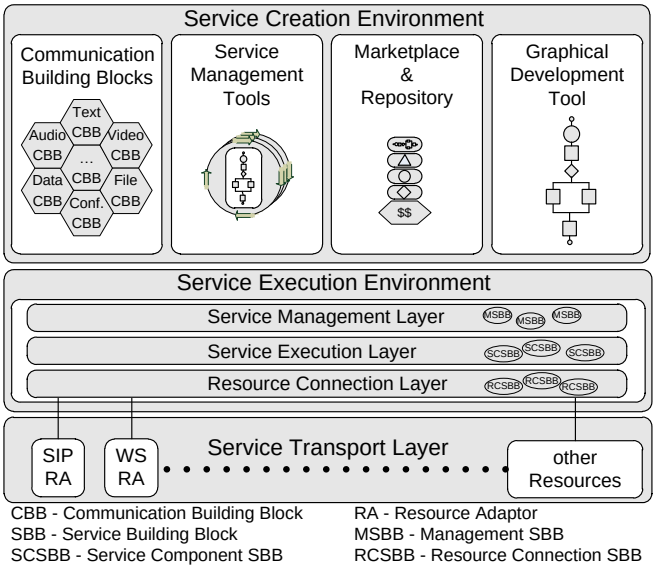

Figure 2. The Framework architecture

A. *The Service Creation Environment*

The Service Creation Environment offers tools to design the services and to acquire new services and other resources. It consists of a repository of Communication Building Blocks (CBBs), the service management tools, and a repository with a marketplace interface to acquire new services and service components. The SCE also offers graphical development tools

to describe the services. BPEL is used as service description language.

BPEL is an established business process specification language normally related to web services. In this case it is used to describe value-added services. It allows the service developer to use existing graphical BPEL design tools to design the service logic. The framework parses the created BPEL file and composes the value-added service from precompiled and already deployed service components. The CBBs are the logical description of the available functionalities. In BPEL these CBBs are available as partner links. They offer a representation for the available resources and functionalities. If a service requires a special functionality, e.g. calling a participant of a conference, the corresponding method from the partner link that handles conferencing issues has to be invoked in the BPEL process.

B. The Extended Service Execution Environment

To solve the described problems with JSLEE (chapter III) this paper proposes an extension to the JSLEE framework which solve these problems and offer a fast and cost efficient development of value-added services. Three new layers are defined within the service component container of the JAIN SLEE Application Server (AS) which enables the support for an automated composition, configuration, and management of value-added services "Fig. 3". The three new layers are the Service Management Layer, the Service Execution Layer and the Resource Connection Layer.

The Service Management Layer consists of management tools. The service management tools control the lifecycle of the value-added services. They offer the functionality to start and stop the services. They can also generate new services from a BPEL process by composing existing components. These tools are implemented as SBBs. To distinguish them from the other SBBs they are called Management Service Building Blocks (MSBBs). The service management supports the control of the available services. The most important MSBBs are the Service Description Parser and the Service Control MSBBs (SCMSBBs) "Fig. 3". An SCMSBB is assigned for the composition, configuration, and lifecycle control of the required service components of a service.

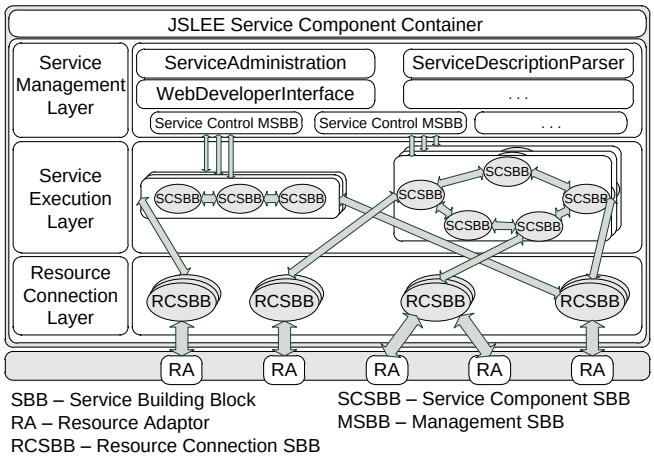

Figure 3. Service Execution Environment

In the Service Execution Layer the implementation of the service logic is located "Fig. 3". This logic is represented by the Service Component SBBs (SCSBBs). The SCSBBs realize the service logic which was described in the workflow of the BPEL process [6]. They are configured and controlled by a SCMSBB. The SCMSBB configure the SCSBBs by setting the required parameters, and decides on which events a SCSBB has to listen and what events it has to fire.

The Resource Connection Layer offers the interfaces to the resources. Special SBBs called Resource Connection SBBs (RCSBBs) form the connection between the service logic and the RAs. RCSBBs implement the methods which are described in the CBBs. They map the logical method description to the functionality which is offered by the RAs. In the service that is generated from the BPEL process, the RCSBBs are called by the SCSBBs to invoke the real functionalities.

C. Service creation and execution

As input for the service creation a service description is required. This description is analyzed by a Service Description Parser. BPEL is supported per default for the service description by the framework. To support non BPEL service descriptions, new Service Description Parsers have to be added to the Service Management Layer "Fig. 4". The SCMSBB composes the required SCSBBs and sends events to configure them. The configuration consists of the information from which component the SCSBB can receive events and what is to do with the received events. The structure of the generated service is derived from the structure of the service description. For each element from the service description a corresponding SCSBB is required "Fig. 5". To signalize that a SCSBB is created and configured, the SCSBB sends an event to the SCMSBB. If all components of the service are ready, the service is ready for execution.

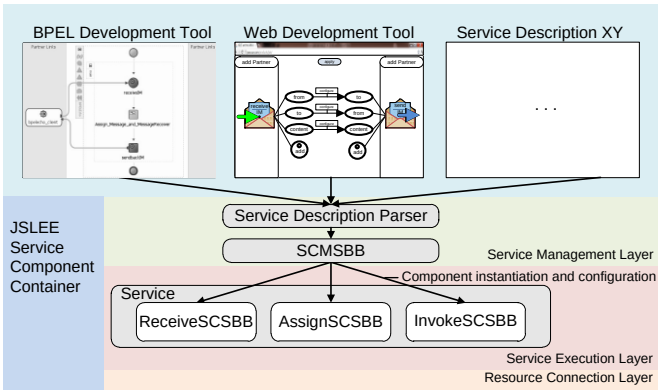

Figure 4. Service Composition

The service that is described in "Fig. 5" waits for incoming mails. If a mail is received, the content of the mail is assigned to a new mail, to an instant message, and to a SMS. Then, all messages are sent out. For each element in the service description (upper part of "Fig. 5") a corresponding SCSBB in the composed service (lower part of "Fig. 5") exists.

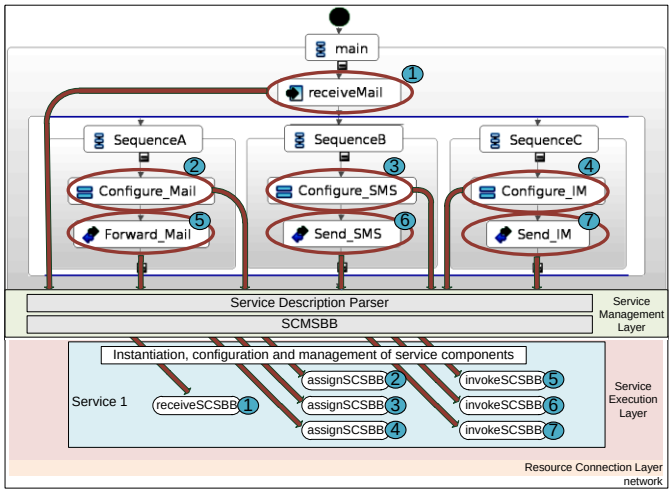

Figure 5. Service Composition Phase

In the execution phase of the service "Fig. 6" all involved components are already configured for execution. When a mail is received by the RA, it fires an event to the appropriate RCSBB and the service is triggered. If the service management decides that a service should be stopped, it fires a service stop event to the SCMSBB of the service. The SCMSBB also fires service stop events to all SCSBBs involved in the service. These SCSBBs answer with confirmation events before they destroy their instance. If all SCSBBs of the service have sent the confirmation, the SCMSBB also sends a confirmation event to the management and removes the service from the AS.

The framework also supports the reconfiguration of running services. If a service should be reconfigured, the management generates a reconfiguration event and fires this event to the SCMSBB of this service. The SCMSBB analyses the changes in the service workflow and initiates the reconfiguration of the involved service components.

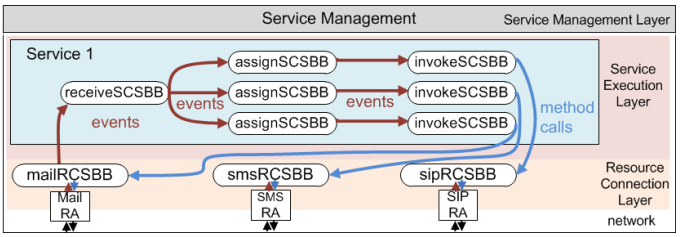

Figure 6. Service Execution Phase

V. Conclusion

This paper proposes a new service creation and service delivery framework. It is based on the JSLEE framework and offers all advantages of the JSLEE framework like a high throughput, low latency, and multi protocol architecture. Additionally the framework offers a simple and fast service development. The services are described with a graphical BPEL development tool. The JSLEE service is automatically composed from the BPEL description. The BPEL elements represent the service logic i.e., the workflow of the service. These BPEL activities have their counterparts in JSLEE as SCSBBs. The required resources/functionalities are described in BPEL as partner links. The service developer does not need special knowledge of the underlying protocols. The required functionalities have only to be invoked in BPEL. Self developed functionalities and resource adaptors can be integrated into the framework by mapping the resources in the RCSBBs to the functionalities described in the CBBs. All required components like services, RAs, CBBs, RCSBBs can also be acquired from the marketplace of the framework. This allows 3rd party developers to offer own resources and services [7]. New protocols can be supported easily by providing the corresponding RAs, RCSBBs and CBBs.

The developed framework prototype already demonstrates the capabilities of the approach. The most important framework elements are implemented. A BPEL service description parser, HTTP and (basic) SIP support are also available. Value-added services which use the implemented components can already be described with BPEL and generated and executed in the framework.

Acknowledgment

The research project ComGeneration providing the basis for this publication was partially funded by the Federal Ministry of Education and Research (BMBF) of the Federal Republic of Germany under grant number 1724B09. The authors of this publication are in charge of its content.

References

[1] OASIS Standard, "Web Services Business Process Execution Language," Version 2.0, OASIS, 2007.

[2] Java Specification Requests, "JSR 240: JAIN SLEE (JSLEE) 1.1 Specification, Final Release," Sun Microsystems, OpenCloud, Sun, 2008.

[3] Lin, L., Lin, P. (2007), "Orchestration in Web Services and Real-Time Communications," IEEE Communication Magazine, 07 2007.

[4] R.H. Glitho, F. Khendek, A. De Marco, "Creating Value Added Services in Internet Telephony: An Overview and a Case Study on a High-Level Service Creation Environment," in Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions, pp 446 - 457, 2003.

[5] A. Lehmann, T. Eichelmann, U. Trick, R. Lasch and R. Tönjes, "TeamCom: A Service Creation Platform for Next Generation Networks," in The Fourth International Conference on Internet and Web Applications and Services (ICIW 2009), M. Perry, H. Sasaki, M. Ehmann, G. O. Bellot, and O. Dini, Eds., ISBN 978-0-7695-3613-2, pp.12-17, IEEE Computer Society, Washington, DC, USA, 2009.

[6] T. Eichelmann, W. Fuhrmann, U. Trick, and B. V. Ghita, "Enhanced Concept of the TeamCom SCE for Automated Generated Services based on JSLEE," in Proceedings of the Eighth International Network Conference (INC 2010), U. Bleimann, P. S. Dowland, S. Furnell, and O. Schneider, Eds., ISBN: 978-1-84102-259-8, pp75-84, University of Plymouth, Plymouth, 2010.

[7] T. Eichelmann, W. Fuhrmann, U. Trick, and B. V. Ghita, "Discussion on a Framework and its Service Structures for generating JSLEE based Value-Added Services," in Proceedings of the Fourth International Conference on Internet Technologies and Applications (ITA 11), S. Cunningham, N. Houlden, V. Grout, D. Oram, and R. Picking, Eds., pp 169-177, ISBN: 978-0-946881-68-0, Glyndwr University, Wrexham, 2011.