DEFINITION OF A SERVICE DELIVERY PLATFORM FOR SERVICE EXPOSURE AND SERVICE ORCHESTRATION IN NEXT GENERATION NETWORKS

Niklas Blum, Thomas Magedanz, Florian Schreiner
Fraunhofer Institute FOKUS, Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany
{niklas.blum, thomas.magedanz, florian.schreiner}@fokus.fraunhofer.de

ABSTRACT
Modern telecommunication networks and classical roles of operators are subject to fundamental change. Many network operators are currently seeking for new sources to generate revenue by exposing network capabilities to 3rd party service providers.
At the same time we can observe that applications on the World Wide Web (WWW) are becoming more mature in terms of the definition of APIs that are offered towards other services. The combinations of those services are commonly referred to as Web 2.0 mash-ups.
This report describes our approach to prototype a policy-based service broker funtion for Next Generation Networks (NGN)-based telecommunications service delivery platforms to provide flexible service exposure anchor points for service integration into so called mash-ups and mechanisms for the orchestration of service enablers.

Keywords: SOA, NGN, service enabler, orchestration, IMS, SDP, SaaS, mash-up.

1 INTRODUCTION

The convergence of fixed and mobile telecommunications networks and applications, cable networks, as well as the Internet leads into a global all-IP based Next Generation Network (NGN). Flexible and powerful service platforms, so called Service Delivery Platforms (SDPs) are in charge to support the efficient design, creation, deployment, provisioning and management of seamless services across different access networks supporting various business models. The reuse of an extensible set of existing service components to create rapidly new market driven applications is a key aspect of telecommunications platforms since many years. Today, Service Oriented Architectures (SOA) are considered as the state-of-the-art for Service Delivery Platforms.

In this context, flexible Service Delivery Platform for telecommunications may serve for multiple purposes. For those operators that do not have a legacy infrastructure (e.g. ISPs or cable operators), the SDP allows them to re-use their service enablers, to expose network capabilities and services to 3rd parties and seamlessly integrate OSS and BSS functions as provisioning, monitoring and customer relationship management.

But currently, most telecommunications operators (especially incumbents) find themselves in a position of transition between legacy and NGN-based infrastructures. This is the place where a NGN SDP is capable of adding further value-add, the SDP may serve for legacy as well as for NGN-based core networks and therefore acts as a bridging element during the time of transition. Applications that make use of such SDPs are fully abstracted from the network nodes, specific protocols and APIs. Such a SDP may serve as an access or exposure gateway for 3rd party applications and Web based mash-up services as it provides a single anchor point for each application.

This article depicts the functionality of a network agnostic service delivery platform based on service oriented architecture principles and names its main components and required functionality as it is currently prototyped at Fraunhofer FOKUS for IP Multimedia Subsystem (IMS)-based core networks. The following section deals with SOA principles in general and presents our blueprint of a SOA-based SDP. Section 3 provides a brief overview of the 3rd Generation Partnership Project (3GPP) IP Multimedia Subsystem as an underlying NGN infrastructure for message signaling and as a docking station for SDPs. Sections 4 to 10 depict the functional entities of our SDP architecture with special focus on the service broker, service exposure mechanisms for 3rd party service providers and APIs dedicated for the usage in Web 2.0 mash-up based application scenarios.

2 BLUE PRINT OF A SOA-BASED SDP ON TOP OF IMS

Service Oriented Architecture is an architectural style that guides all aspects of creating and using business processes, packaged as services, throughout their life cycle, as well as defining and provisioning the IT infrastructure that allows different applications to exchange data and participate in business processes loosely coupled from the operating systems and programming languages underlying those applications [1]. SOA represents a model in which functionality is decomposed into distinct units (services), which can be distributed over a network and can be combined together and reused to create business applications. These services communicate with each other by passing data from one service to another, or by coordinating an activity between two or more services.

Web Services can be used to implement a Service Oriented Architecture. A major focus of Web Services is to make functional building blocks accessible over standard Internet protocols that are independent from platforms and programming languages. These services can be new applications or just wrapped around existing legacy systems to make them network-enabled. Each SOA building block may play one or more of three roles:

- **Service provider** – The service provider creates a web service and possibly publishes its interface and access information to the service registry.
- **Service broker** – The service broker is responsible for making the Web Services interface and implementation access information available to any potential service requestor. The broker might be utilized as a central control instance that orchestrates the components of the overall architecture.
- **Service requestor** – The service requestor or Web Services client locates entries in the broker registry and binds to the service provider in order to invoke one of its Web Services.

A SOA may also be regarded as a style of information systems architecture that enables the creation of applications that are built by combining loosely coupled and inter-operable services [2].

These services inter-operate based on a formal definition (or contract, e.g., WSDL [3] or usage policy) that is independent of the underlying platform and programming language. SOA-based systems can therefore be independent of development technologies and platforms. High-level languages such as Business Process Execution Language (BPEL) extend the service concept by providing a method of defining and supporting orchestration of fine grained services into more coarse-grained business services, which in turn can be incorporated into work-flows and business processes implemented in composite applications or portals [4].

The concept of SOA as described above has a long history in telecommunications. Its origin can be identified at the development of the Intelligent Network (IN) in the 1980's. The major goal was the development of a programmable network environment for the delivery of new value added services extending the Plain Old Telephony System (POTS) and thus generating new revenues. The idea was to define an overlay service architecture on top of a physical network and to extract the service intelligence from the legacy network switches into dedicated central service control points (SCPs). Service independence of the IN architecture should have been provided by the definition of reusable service components, which could be chained adequately for the realization of new services. [5]

Based on the above mentioned roles of building blocks in a SOA, we have developed a blue print for a network agnostic service delivery platform.

Application or service enablers that are mapped to specific network protocols abstract from network centric services like call control, conferencing, presence, etc. using Web Services bindings and serve as **service providers** towards the SDP. From the internal perspective of the SDP or for applications that want to make use of the service enablers, the underlying network protocols and the accompanying specific service functionality is transparent; only the Web Services API provided by the service enablers is visible.

The **service requestor** may be an application residing in a 3rd party domain that accesses the SDP through a secured 3rd party interface. A dedicated network exposure mechanism has to be provided by the SDP that serves for the definition of Service Level Agreements (SLA) between the operator and the service provider. Such an exposure syntax needs to provide flexible constructs to define the individual usage of each service enabler or even the exposure of more complex, composed services.

The **service broker** serves as the organizational glue between service enablers, applications and SDP internal functions as service repositories and service registry. Furthermore, it may initiate processes during runtime to assure a certain service level for dedicated fulfillment of the service execution. The broker may also compose services based on constraints expressed by the service request. Such complex services consisting of the execution of several services may then be stored at the service repository for future usage and become new service provider.

The following figure 1 depicts our blue print of a SOA-based SDP suited for NGNs as well as for legacy networks.

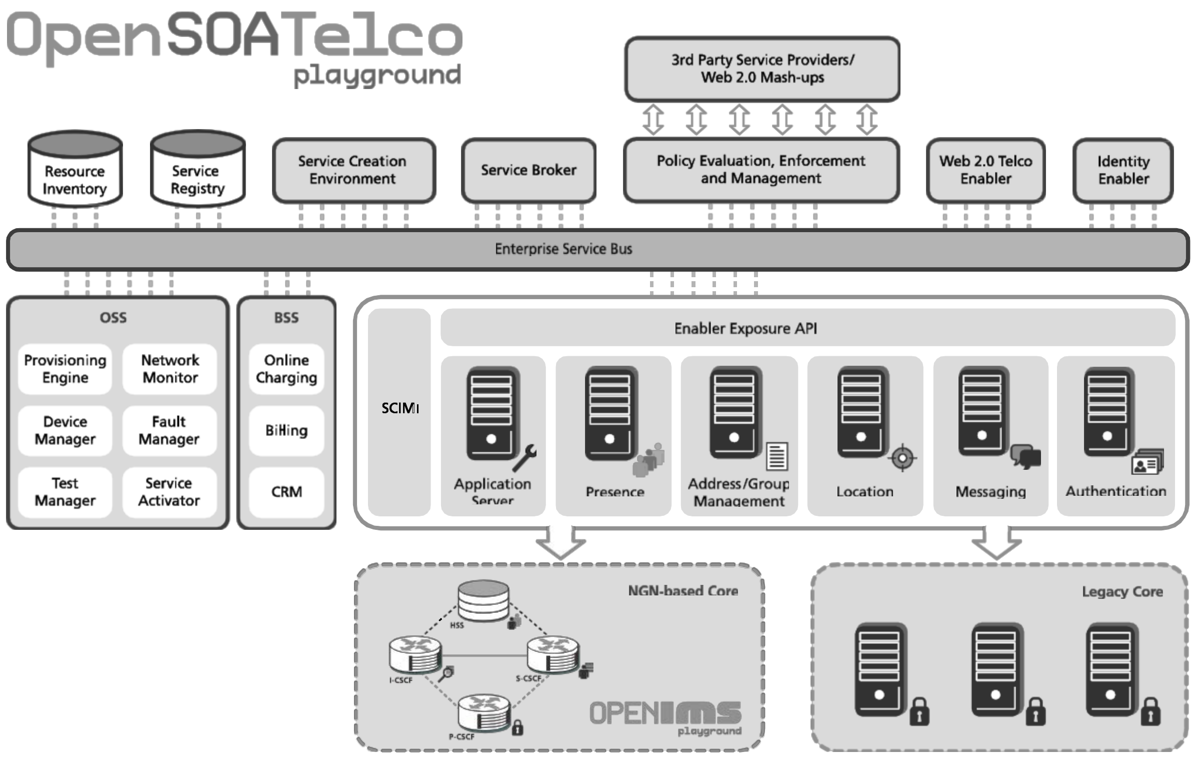

Figure 1: FOKUS Open SOA Telco Playground Architecture

The following sections depict each of the building blocks in more detail with special emphasis on exposure mechanisms of the service enablers.

3 THE 3GPP IP MULTIMEDIA SUBSYSTEM

Even if the IMS is not part of the SDP it provides the interfaces for interaction and underlying communication control infrastructure. The IP Multimedia Subsystem [6], [7] and [8] is defined from 3GPP Release 5 specifications on as an overlay architecture on top of the 3GPP Packet Switched (PS) Core Network for the provision of real time multi-media services.

Due to the fact that the IMS overlay architecture is widely abstracted from the air interfaces, the IMS can be used for any mobile access network technology as well as for fixed line access technology as currently promoted by the European Telecommunications Standards Institute's (ETSI) Telecoms & Internet converged Services & Protocols for Advanced Networks (TISPAN) within the Next Generation Network reference architecture definition.

The central session control protocol is the Session Initiation Protocol (SIP) [9]. The SIP Application Server (AS) is the service relevant part in the IMS. How multimedia applications are programmed is out of scope of the standardization committees. But the SIP AS needs to support well defined signaling and administration interfaces (3GPP ISC and Sh-interfaces) to connect to the standardized network architecture.

This enables developers to use several programming paradigms within a SIP AS, such as legacy IN servers, Open Service Access (OSA) / Parlay servers/gateways, or any proven Voice over Internet Protocol (VoIP) SIP programming paradigm, like SIP Servlets, call programming language (CPL) and Common Gateway Interface (CGI) scripts, etc. Figure 2 depicts the simplified IMS architecture.

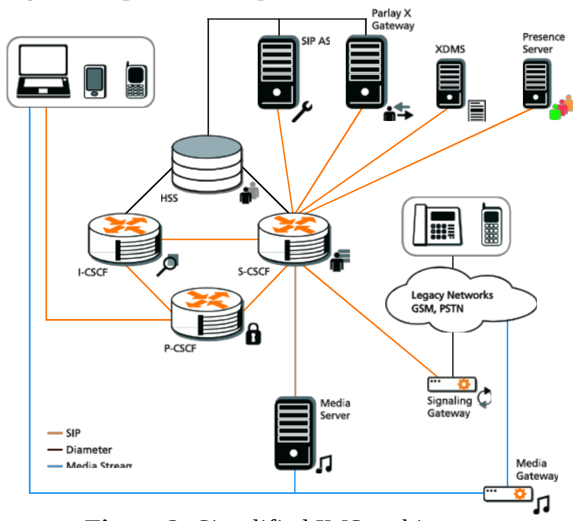

Figure 2: Simplified IMS architecture

There are four additional key functionalities that mark the IP Multimedia Subsystem as the future technology in a comprehensive service and application oriented network.

1. The IMS provides easy and efficient ways to integrate different services, even from third parties. Interactions between different value added services are anticipated.
2. The IMS enables seamless integration of legacy services and is designed for consistent interactions with circuit-switched domains.
3. The IMS supports mechanisms to negotiate Quality of Service (QoS). Within a session a user may request QoS for certain Packet Data Protocol (PDP) Contexts on the critical 3G air interface.
4. The IMS provides appropriate charging mechanisms and it is therefore possible to realize different business models and charge for specific events using an appropriate scheme, such as time or volume based tariffs, QoS, etc.

The particular techniques and methodologies that are required to gain these key functionalities are not new, but the IMS provides the first major integration and the interaction of all key functionalities.

From the perspective of the IMS, a SOA-based SDP acts a simple Application Server. By defining logical entities that are connected to each other through standardized protocols, a plug-and-play architecture has been created that offers the possibility to physically place each function at different locations and to assemble an IMS with functions from different vendors.

4 SERVICE CAPABILITY INTERACTION MANAGER

3GPP has introduced Service Capability Interaction Manager (SCIM) [10] as a function within the SIP application server domain of IMS for managing the interactions between Application Servers. However, the service interaction management functionalities of SCIM are not specified and research in this field is in progress.

Basically, there are different ways of achieving such functionality:

- a request dispatcher within the execution environment
- an interaction manager on the ISC interface between the S-CSCF and Application Servers

Whereas the first solution is part of the upcoming SIP Servlet Specification 1.1 (JSR 289) [11] named "Application Router" and is only specified for JSR 289 compliant implementations, the latter one is regarded as an open subject. SCIM may be regarded as a broker to compose services based on user or network initiated requests that are distributed over different SDPs.

5 IMS APPLICATION ENABLER

Similar to service independent building blocks which form part of the conceptual model for Intelligent Networks, the Open Mobile Alliance (OMA) has defined service enablers for the IP Multimedia Subsystem. The idea was initially born during the specification of a Push-to-Talk over Cellular (PoC) [12] service, a walkie-talkie like communication service between several mobile peers. PoC uses Presence, Group Management and Instant Messaging as enablers to provide information to the users as well as to the PoC service. This lead alongside the standardization of PoC to the definition of Presence SIMPLE [13] for Presence and Instant Messaging and XML Documents Management (XDM) [14] for group and list management. PoC as a public available service never received real acceptance besides the U.S. market, but the concept of abstract application enablers is by now widely used. Further application enablers that are not standardized by OMA but should be part of every network abstraction layer are call- and conferencing control. Other enablers should be charging and depending on the underlying network capabilities legacy messaging like SMS and MMS or location.

6 OMA SERVICE ENVIRONMENT AND POLICY EVALUATION, ENFORCEMENT AND MANAGEMENT

The definitions of several application service enablers by the OMA and the need for a general access function for 3rd party service access led to the specification of the OMA Service Environment (OSE) [15] as a common abstraction environment for service enablers.

Figure 3 illustrates the proposed architecture by the OMA.

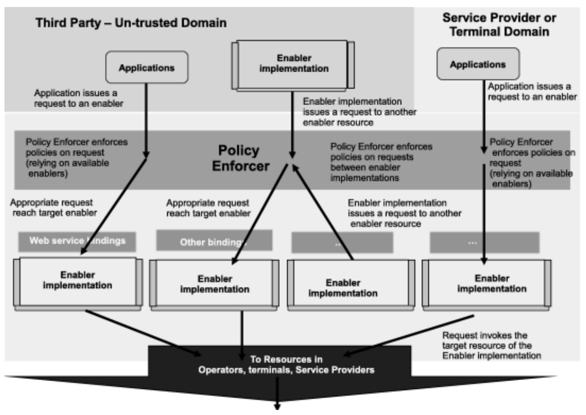
Third Party – Un-trusted Domain
Service Provider or Terminal Domain
Applications
Enabler implementation
Applications
Application issues a request to an enabler
Enabler implementation issues a request to another enabler resource
Application issues a request to an enabler
Policy Enforcer enforces policies on request (relying on available enablers)
Policy Enforcer
Policy Enforcer enforces policies on requests between enabler implementations
Policy Enforcer enforces policies on request (relying on available enablers)
Appropriate request reach target enabler
Appropriate request reach target enabler
Enabler implementation issues a request to another enabler resource
Web service bindings
Other bindings
...
...
Enabler implementation
Enabler implementation
Enabler implementation
Enabler implementation
Request invokes the target resource of the Enabler implementation
To Resources in Operators, terminals, Service Providers

Figure 3: OMA Service Environment Architecture

It defines an enabler layer which incorporates specific enabler components that offer northbound interfaces to services that implement certain application logic. These applications either reside at the operator domain or are hosted at a 3rd party domain. An enabler component can either be part of the OSE or the OSE can act as an application overlay that offers interfaces to other service enabler functions.

Basically, an OSE incorporates Web Services interfaces and translates Web Services requests either directly into enabler logic or to an enabler specific protocol. OMA does at the point of writing not standardize any mapping to a specific middle-ware messaging technology but leaves this open to the implementation of specific service environments. An enabler could as well be a non standardized implementation towards a specific telephony platform or an IN platform. Furthermore an enabler can be implemented towards several protocols to provide a network converging functionality. NGN technologies with legacy networks, e.g. a messaging enabler can be mapped to SIP, short message peer-to-peer protocol (SMPP) [16] to communicate with a SMS-C for sending out SMS and MM-7 [17] to communicate towards a MMS-C.

The Policy Enforcer or Policy Evaluation, Enforcement and Management (PEEM) component as the function has been named officially by the OMA can be used to intercept service requests from a foreign domain as well as from any other service requestor and apply certain rules (policies) that a stored at a policy repository. Basically, policies are used for the authorizations of requests meaning that service invocation requests that are intercepted by the PEEM are checked for valid authorization and authentication. PEEM may furthermore be used to define enabler capabilities for exposure based on request policies. Depending on the business model different charging rules may also be applied for service requests through specific policies. In this regard the definition of policies may be considered the expression of Service Level Agreements between a network operator and a service provider.

A PEEM function forms the main integral component of an OMA Service Environment and provides additional functionality based on the definition policy for the OMA enabler concept. PEEM may serve as an access gateway authentication function but its capabilities are much greater in regard of the orchestration and manipulation of enabler capabilities. The OMA names two different Policy Expression languages, Common Policy by the Internet Engineering Task Force (IETF) [18] for authorization policies and Business Process Execution Language (for Web Services) WSBPEL 2.0 defined by Advancing Open Standards for the Information Society (OASIS) [19] for the orchestration of enablers.

7 SERVICE BROKER

The functional components of service brokerage are diverse; they likely have different operations and management platforms, and yet have to inter-operate with one another, e.g., to provide an integrated modeling and brokering service. Traditionally, one approach to this issue is to employ one of existing distributed computing technologies, e.g., CORBA and DCOM [20]. However, such an approach means tight coupling between all the parties involved and may require them to use the same vendor platform. These are serious limitations, especially for service brokerage applications whose main functionality is to facilitate partnerships and inter-operation among a potentially large number of services.

In contrast, Web Services provide a flexible and loosely coupled means of integration. XML-encoded WSDL interfaces and SOAP messages allow for platform independence and promote concurrent development and testing. Use of HTTP as the transport mechanism means that SOAP messaged can traverse network boundaries without making policy and configuration changes in most cases. As such, Web Services are an ideal choice of integration technology for realizing service brokerage applications.

A powerful service broker is the major function providing the orchestration of all components in a SOA. Dynamic service activation, service fulfillment and the composition of services from multiple service enablers requires the involvement of many functions of an operator's network. The service broker interacts between all the components connected to the service bus and functions as a binding component between service repositories offering description of available services, Policy Evaluation, Enforcement and Management for service and user specific policies, Operations and Business Support Systems (OSS/BSS) for e.g. provisioning, specific service monitoring or service activation and the application or service enablers.
As the main service orchestration engine of a SOA-based SDP, the service broker should be capable of:

- **Service Orchestration** – Enablers can be strung together in predefined patterns and executed via "orchestration scripts" which are either a complex policy stored at PEEM or make use of process description languages to apply several policies on different enablers.
- **Resource delegation** – Different resources may be mapped to service requests for dynamic service fulfillment

The term enabler in this regard extends the notion of application enablers provided by exposure APIs as Parlay X [21] and comprises all services attached to the Enterprise Service Bus (ESB). This provides the possibility of mapping a complete service life-cycle as an orchestration script. The chosen orchestration language for our architecture is WSBPEL 2.0. But there are other promising orchestration specifications as the Web Service Choreography Interface (WSCI) [22] standardized by the W3C that is an XML-based interface description language describing the flow of messages exchanged by a Web Services interacting with other Web Services. Furthermore Business Process Modeling Language (BPML) has been proposed by Business Process Management Initiative Organization (BPMI), but BPMI [23] has dropped support for this in favor of BPEL and joined the Object Management Group (OMG) in 2005.

However, it shall be noticed that interface descriptions are low level, technical statements (e.g. WSDL statements) that are understandable by software professionals but far to be comprehensible by business people. At the same time, the notion of a service is familiar to the management world [24] and with the growing acceptance and popularity of SOA, computing systems now aim to extend far beyond the firewall to automate enterprise-wide business processes, covering sales, supply chain, manufacturing, delivery, payment, human resources, and more. To attain this, it is necessary to adapt SOA to a mainstream practitioners' level and bridge the gap between high level business services and low level software services [25].

The position supported in this paper is to follow the suggestion by [26] to move from the *function-driven* SOA to *intention-driven* SOA as a way to expose services and to communicate towards the broker entity. Whereas the former lies on a functional view of services, the latter proposes to spell out the purpose, the intention behind a service. As a consequence, interfaces of these services will bring out the business goal that the service allows to fulfill instead of defining the signatures of basic operations that can be invoked on class objects. This will avoid the current mismatch of languages between low level services expressions such as WSDL statements and business perceived services.

The usage of intent-based service request syntax allows the composition of multiple NGN application enablers and the exposure of composed services through a single interface and API. The service broker has the task to compose the requested application enablers through the usage of BPEL. Furthermore intent-based service requests allow a very loosely coupled infrastructure for service fulfillment and the consideration of constraints during runtime. For example in the case of an incumbent operator that owns a fixed and a mobile network, the intent-based service request to create a 3rd party call may be mapped to different service enablers for call control depending on the location of the participants of the call. As a result the provided underlying infrastructure becomes fully transparent to the service requestor and may be altered or replaced without the need to change service exposure mechanisms or 3rd party applications.

The proposed intent-based service request syntax facilitates also the usage of the infrastructure for service providers as there is only one interface to deal with that allows the usage of the complete telecommunications infrastructure.

8 INTEGRATION OF OPERATIONS AND BUSINESS SUPPORT SYSTEMS

The landscape of standardization consortia responsible for the provisioning of NGN management standards and system is broad. The International Telecommunication Union Telecommunication Standardization Sector (ITU-T) as the overarching standardization consortium for telecommunication networks strives for harmonization of the different management standards and approaches. The TeleManagement Forum's (TMF) eTOM [5] standard was accepted as an ITU-T standard and serves the ETSI TISPAN as a foundation for their NGN OSS standard in the same way as its predecessor the Telecom Operations Map (TOM) serves the 3rd Generation Partnership Project (3GPP) as a fundamental reference.

Therefore it can be stated that TMF's New Generation Operations Systems and Software (NGOSS) [27] framework (of which eTOM is one of the most important cornerstones) more and more serves the whole telecommunications OSS standardization as major reference. Nevertheless, some terminology from the Telecom Management Network (TMN), like the differentiation between element, network, service and business management layers, is still commonly in use.

NGOSS is a work program governed by the TMF to deliver a framework that is supposed to facilitate the integration of Operations and Business Support Systems into SOA-based services. The goal of NGOSS is to facilitate the rapid development of flexible, low cost of ownership, OSS/BSS solutions to meet the business needs of the Internet enabled economy. NGOSS is based on the following five principles:

1. **Separation of Business Process from Component Implementation** – Current TMF approaches like the Service Delivery Framework Program propose that future NGOSS-based OSS processes are managed as part of the centralized service infrastructure, using a work-flow engine that is responsible for controlling business processes between the applications. Therefore, the work-flow engine would initiate a process for an application, which would then return control to work-flow engine, which would then

call another application, and so on. In this way it is always possible to find out about the state of individual processes as part of the flow.

2. **Loosely Coupled Distributed System** – "Loosely coupled" means that each application is relatively independent of the other applications in the overall system. Therefore, in a loosely coupled environment, one application can be altered without the alteration necessarily affecting others. This can be viewed as producing the ability to "plug and play" applications, where they are so independent that they can be changed without affecting the overall system behaviour. This distributed system is emphasizing that NGOSS is not implemented using a single monolithic application to manage all its activities, but is instead using a set of integrated and co-operating applications that offer well-defined APIs and/or protocol interfaces.

3. **Shared Information Model** – Integrating OSSs but also for SOA in general means that data must be shared between the applications. For this to be effective, either each application must understand how every other application understands/interprets that part of the data that is shared, or there must be a common model of the shared data. A single information model for data that is shared between applications provides a solution to this problem. The TMF solution to this is called the Shared Information/Data Model (SID) [28].

4. **Common Communications Infrastructure** – NGOSS describes the use of a Common Communications Infrastructure (CCI). In this model, OSSs interface with the as it provides a common communication channel. In this way, each application only requires one interface (to the CCI) rather than many (to other applications). The CCI may also provide other services, including security, data translation, etc.

5. **Contract defined interfaces** – Given the description above of how applications interface to the CCI, it is obvious that a way of documenting those interfaces, both in terms of the technology employed (e.g. is it Java/JMS or Web Services/SOAP) but also the functionality of the application, the data used, the pre- and post-conditions, etc. is needed. The NGOSS contract defined interfaces provide a means to document these interfaces. NGOSS contracts can be seen as extensions of API specifications.

NGOSS targets the use of commercial off-the-shelf information technologies, instead of technologies unique to the telecommunications industry, as many legacy management systems have done in the past. This approach significantly reduces costs and improves software reuse and operational flexibility, enabling NGOSS-based systems to support a range of new services and new technology environments more easily.

9 SERVICE DISCOVERY / SERVICE REPOSITORIES

Composite services require several service enablers to exchange information. A very simple, purely IMS service enabler based composed service could be: "Send an Instant Message to all my online Buddies". Although this appears to be a straightforward command, several service enabling components have to interwork to fulfill this request.

At first, the complete buddy list has to be downloaded from the OMA XML Document Management Server (XDMS) via Parlay X Address List Management Web Services API. Second, the presence state of all buddies has to be checked via the Presence server by initiating a Web Services request towards the Parlay X Presence API. Finally, an Instant Message is sent via the Parlay X Send Multimedia Message API.

In order to compose such a service, the workflow engine has initially to query the OSS through Java (OSS/J) [29] Inventory API in order to check for the availability of the required service enablers (products in TMF eTOM taxonomy). The Inventory API triggers a service discovery request, via OSS/J Discovery API which looks up available service enablers by querying the Universal Description, Discovery and Integration (UDDI). The workflow engine receives a list of semantically enriched WSDL-S [30] from the UDDI and after input, output, parameters and arguments (the service specification) have been matched, composes the entire service workflow. Finally and most importantly, the workflow engine composes a new WSDL-S for the created services, which via the same mechanism as described above is being stored at the UDDI for later usage. Following this mechanism, iteratively more and more complex services can be created out of already existing composed services. Figure 4 depicts such a workflow mapped to our architecture.

The same applies to the management of newly generated service compositions. At the current stage, with the help of a Provisioning Server and a Fault Management System, the IMS core network as well as the service enabling layer can automatically be provisioned by triggering the OSS/J Order Management API.

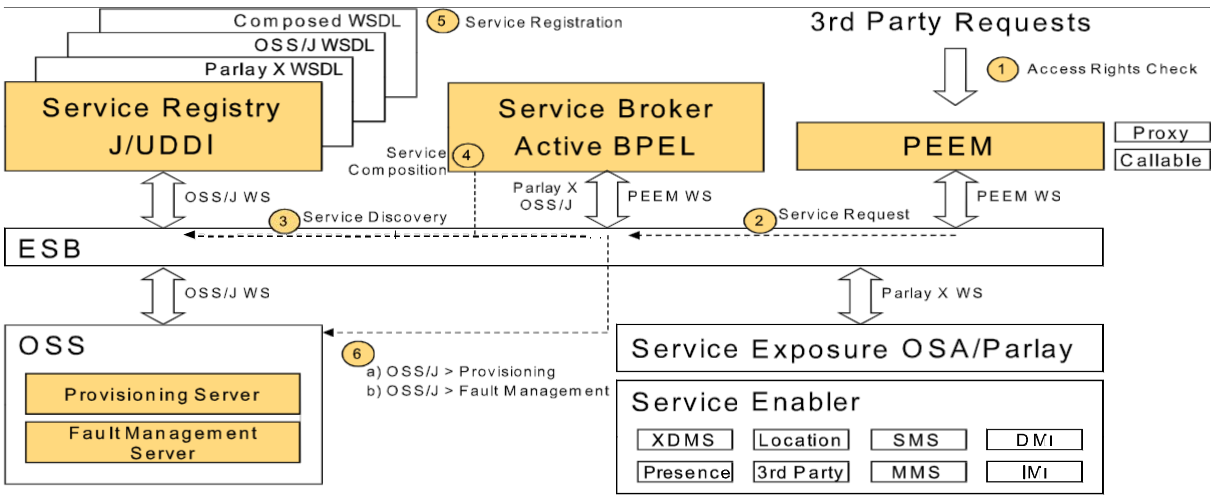

Figure 4: Automated Workflow based Service Discovery, Composition and Management

Furthermore, for service assurance, via the OSS/J Fault Management API, the process of active and passive service monitoring mechanism is being triggered. Therefore, the workflow engine after the discovery of the relevant management services (Order Management for Fulfillment and Fault Management for Assurance) initiates the operation support processes as part of one and the same service composition choreography.

10 TELCO / WEB 2.0 ENABLER

Operators that have implemented an enabler layer as part of their SDP strategy are able to combine such network abstraction with an OMA PEEM based environment to offer these enablers to any service developer on the Internet. Nevertheless, most high level Web developers are used to different programming paradigms and data structures than those offered by e.g. the Parlay X APIs. Therefore, it might be necessary to provide specific interfaces or enablers to address the needs of programmers of web applications. We call these interfaces Telco Web 2.0 Enabler that consist of JavaScript [31] APIs that can be incorporated easily into Ajax (Asynchronous JavaScript and XML) [32] based web applications.

Calling a Web Service from within an Ajax application is restricted to local Web Services, due to the same origin policy enforced by modern web browsers. However, Web Services are used to be consumed beyond server limits at external endpoints. In order to call external Web Services from JavaScript, the service request has to be routed via a service expore gateway that can be offered by an operator. This gateway provides interfaces for the client-side JavaScript, mapping it to the particular interface of the external Web Service. Thus, all parameters of an incoming Ajax request are passed on to the Web Service Endpoint – e.g. Parlay X enablers. The JavaScript client does access a single server as usual acting as the web 2.0 / Telco service gateway. This gateway may then call Web Services that are provided by service enablers or offer more complex composed services at an operator's SDP remotely. Web Services are typically accessed via SOAP messages that are difficult to handle with JavaScript. The utilization of a gateway allows replacing SOAP by any protocol, as the protocol can be translated within the gateway. Therefore, more convenient data description formats, such as the JavaScript Object Notation (JSON) [33] and simple XML, can be used to ease the Web Services access.

The following figure provides an overview of the described functionality provided by a JSON bridge:

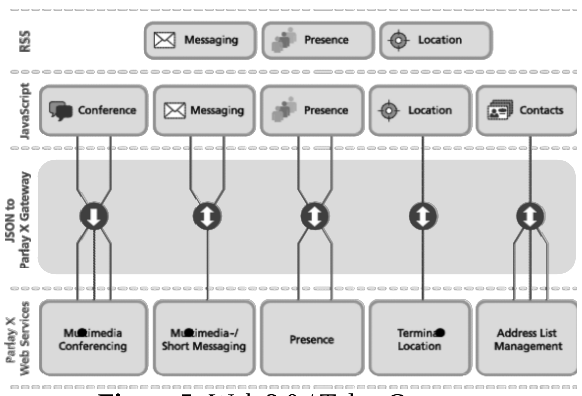

Figure 5: Web 2.0 / Telco Gateway

Furthermore, the Web Services access can be simplified by abstracting the actual Web Services interface. Thereby, underlying Web Services business logic can be hidden from the JavaScript developer as well as offered functionality can be expanded at the server side. For example, a JavaScript method call from an Ajax web application might initiate an orchestrated application at the operator's SDP.

11 IDENTITY ENABLER

As the Internet world is quickly adopting possibilities for representing digital identities (like OpenID) [34] which stems from the blogosphere where it helped to simplify authentication for blog entries and is constantly growing in a grass-root integration process with other identity solutions or Microsofts Information Cards which can be used by today with many computers by using the CardSpace [35] identity selector (which explicitly supports authentication at an identity provider over smartcards), first telecommunications operators are starting to look into the possibilities of becoming identity providers for themselves. The target is to establish a digital representation of the cards that can be found in the wallets of users today and beyond to provide verified data such as age, solvency, location, etc. Telcos are in a very good position to become identity providers for user-centric solutions for themselves as they have been dealing with large scale identity management. They have already the trust of users not to tamper with their privacy and the 3GPP Generic Bootstrapping Architecture [36] provides already a secure option to mutually authenticate users and the network. While GBA may also be used to assert identities for services hosted with the operator itself, there will also be many cases when a user can re-use that asserted identity information to present it also to Internet service providers, e.g. to create blog entries or orders online in an easy manner from a mobile device

12 CONCLUSIONS

SOA principles have been used inside telecommunications domains for many years, although different terms have been used over the last decades to describe the idea of realizing a programmable network to provide an open market of services. Today, Web Services based APIs including emerging Web 2.0 interfaces represent the state of the art in SOA-based telecommunications, which are going to be integrated with the emerging IMS. However, there is still a lot of research and development needed in this domain, as the challenge is to provide a secure and deterministic service environment and not just a best effort service environment we know today from the Internet.

We have depicted in this report our blueprint for service delivery platform based on SOA principles that takes the latest standards and concepts in telecommunications into account. The major work is based on policy-based service exposure environments and a service broker in charge of the orchestration of several service enablers for composed services. The service request paradigm that has been applied makes use of an intent-based API that allows the flexible orchestration of enablers in a loosely coupled manner. Furthermore it eases the exposure of services and their usage as requestors will only have one flexible interface to use.

The system has been prototyped as part of the Open SOA Telco Playground [37] at Fraunhofer FOKUS. The Open SOA Telco Playground is the north bound extension of the FOKUS Open IMS Playground [38] founded in 2004. As the IMS is considered today as the unifying architectural framework for the provision of seamless IP based services on top of converging networks and the south bound foundation for many Service Delivery Platforms, the Open SOA Telco Playground provides the possibility to experience a SOA on top of converging networks. However, this vendor independent playground is not only limited NGNs, but also supports the provision of services on top of legacy fixed and mobile telecommunication networks as well as the next generation Internet. The major focal point of the Open SOA Telco Playground is on the provisioning of telecommunications oriented service capabilities based on state of the art SOA principles to an open set of business domains. This vendor and provider independent technology playground represents an open testbed for research and experiencing and validating the development, orchestration, provisioning, execution, and management of converging NGN and future Internet applications based on SOA principles.

The Open SOA Telco Playground is provided by the Fraunhofer Institute FOKUS Next Generation Network Infrastructures' department and is an integral component of the FOKUS SOA Laboratory.

ACKNOWLEDGEMENT

The research described in this paper has been performed within the Fraunhofer FOKUS next generation networks infrastructures (NGNI) competence center in close cooperation with the chair for next generation networks (AV) within the electrical engineering and computer sciences faculty of the Technische Universität Berlin.

13 REFERENCES

[1] Newcomer, Eric; Lomow, Greg, Understanding SOA with Web Services. Addison Wesley. ISBN 0-321-18086-0, 2005

[2] Channabasavaiah, Holley and Tuggle, Migrating to a service-oriented architecture, IBM DeveloperWorks, http://www-128.ibm.com/developerworks/library/ws-migratesoa/, 16 Dec 2003

[3] W3C, "Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language", http://www.w3c.org/TR/wsdl20

[4] Service-oriented architecture, http://en.wikipedia.org/wiki/Service-oriented architecture, 2008

[5] T. Magedanz, N. Blum, S. Dutkowski, "Evolution of SOA Concepts in Telecommunications - A Déjà vu?", Special Issue on Service Oriented Architectures, IEEE Computer, November 2007, ISSN 0018-9162

[6] 3GPP, TS 23.228 V7.6.0, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; IP Multimedia Subsystem (IMS); Stage 2 (Release 7)", December 2006

[7] ETSI TISPAN WG, http://portal.etsi.org/tispan/TISPAN ToR.asp

[8] Enabler Release Definition for IMS in OMA, Open Mobile Alliance, OMA-ERELD-IMSinOMA-V1 0-20050809- A, August 2005

[9] H. Schulzrinne et al., "RFC 3261 SIP: Session Initiation Protocol", June 2002

[10]3GPP, TS 23.002 V7.4.0, "Network architecture", December 2007

[11]Java Community Process, "JSR 289: SIP Servlet v1.1", January 2008

[12]Open Mobile Alliance (OMA). Enabler Release Definition for Push-to-talk over Cellular. Candidate Version 2.0, 11 Dec 2007

[13]Open Mobile Alliance (OMA). Presence SIMPLE Architecture Document. Approved Version 1.0.1, 28 Nov 2006

[14]Open Mobile Alliance (OMA). XML Document Management Architecture. Candidate Version 2.0, 24 Jul 2007

[15]Open Mobile Alliance (OMA). OMA Service Environment. Approved Version 1.0.4, Feb 2007

[16]Short Message Peer-to-Peer Protocol Specification v5.0, 19 Feb 2003

[17]3GPP, TS 23.140 V6.14.0, "Multimedia Messaging Service (MMS); Functional description; Stage 2", June 2006

[18]H. Schulzrinne et al., "RFC 4745 Common Policy: A Document Format for Expressing Privacy Preferences", February 2007

[19]OASIS Web Services Business Process Execution Language Version 2.0, http://www.oasis-open.org/committees/ wsbpel/, April 2007

[20]COM: Component ObjectModel Technologies, http://www.microsoft.com/com/default.mspx

[21]The Parlay Group, Parlay X Web Services, Specification,http://www.parlay.org/en/specifications/pxws.asp

[22]W3C, "Web Services Choreography Interface (WSCI) 1.0", http://www.s3c.org/TR/wsci

[23]BPMI, http://www.bpmi.org/

[24]Piccinelli, G., Emmerich, W., Williams, S-L., Stearns, M.: A Model-Driven Architecture for Electronic Service Management Systems. In: Orlowska, M.E., Weerawarana, S., Papazoglou, M.M.P., Yang, J. (eds.) ICSOC 2003. LNCS, vol. 2910, pp. 241–255, Springer, Heidelberg (2003)

[25]Arsanjani, A., "Service-oriented modelling and architecture", November 2004, http://www-128.ibm.com/developerworks/webservices/library/ws-soa-design1/

[26]C. Rolland, R. Samia and N. Kraiem, "On ISOA: Intentional Services Oriented Architecture", LNCS, vol. 4495/2007, pp. 158-172, Springer, Berlin/Heidelberg (2007), ISBN 978-3-540-72987-7

[27]TeleManagement Forum, "The NGOSS Lifecycle and Methodology", April 2004

[28]TeleManagement Forum, NGOSS SID, http://www.tmforum.org/TechnicalPrograms/NGOSSSID/1684/Home.html, 2008

[29]The OSS through Java Initiative (OSS/J), http://www.ossj.org

[30]W3C, Web Service Semantics - WSDL-S, 2005

[31]ECMAScript Language Specification, 3'rd Edition, 1999, http://www.ecma-international.org/publications/standards/Ecma-262.htm

[32]J. J. Garrett. *Ajax: A new Approach to Web Applications.* 2005.

[33]JSON-RPC Specification 1.1, Working Draft, August, 2006, http://json-rpc.org/wd/JSON-RPC-1-1-WD-20060807.html

[34]OpenID, http://openid.net/

[35]David Chappell, "Introducing Windows CardSpace", http://msdn2.microsoft.com/en-us/library/aa480189.aspx

[36]3GPP, TS 33.220 V7.10.0, "Generic Authentication Architecture (GAA); Generic boot-strapping architecture", December 2007

[37]FOKUS Open SOA Telco Playground, http://www.opensoaplayground.org

[38]FOKUS IMS Playground, http://www.open-ims.org