jet INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2010, VOL. 56, NO. 2, PP. 117–124
Manuscript received January 15, 2009; revised June, 2010.
PAN POLSKA AKADEMIA NAUK www.journals.pan.pl
10.2478/v10177-010-0015-2

Using JAIN SLEE as an Interaction and Policy Manager for Enabler-based Services in Next Generation Networks

Mosiuoa Tsietsi, Alfredo Terzoli, and George Wells

***Abstract*—The IP Multimedia Subsystem is a telecommunications framework with a standard architecture for the provision of services. While the services themselves have not been standardised, standards do exist for basic technologies that can be re-used and aggregated in order to construct more complex services. These elements are called service capabilities by the 3GPP and service enablers by the OMA, both of which are reputable standards bodies in this area. In order to provide re-usability, there is a need to manage access to the service capabilities. Also, in order to build complex services, there is a further need to be able to manage and coordinate the interactions that occur between service capabilities. The 3GPP and the OMA have separately defined network entities that are responsible for handling aspects of these requirements, and are known as a service capability interaction manager (SCIM) and a policy enforcer respectively. However, the internal structure of the SCIM and the policy enforcer have not been standardised by the relevant bodies. In addition, as the SCIM and the policy enforcer have been defined through complementary yet separate processes, there is an opportunity to unify efforts from both bodies. This paper builds on work and standards defined by the bodies, and proposes the design of an interaction manager with features borrowed from both the SCIM and the policy enforcer. To help validate the design, we have identified a platform known as JAIN SLEE which we believe conforms to the model proposed, and we discuss how JAIN SLEE can be used to implement our ideas.**

***Keywords*—Service broker, SCIM, policy enforcer, NGN, IMS, JAIN SLEE, 3GPP, OMA.**

I. Introduction

FIXED and mobile telecommunication operators have experienced falling revenues in recent years due to the advent and rapid growth of relatively simple Internet-based applications, some of which are available to consumers for free. Service-oriented architectures such as the Internet Protocol Multimedia Subsystem (IMS) hold promise for reclaiming these losses by enabling the deployment of rich, interactive, multimedia services that consumers can access on a range of different communication devices such as mobile phones and PCs, and across different types of access networks such as 3G and Wi-fi.

In their technical specifications, standards bodies such as the 3GPP (3rd Generation Partnership Project), ETSI (European Telecommunications Standards Institute) and the OMA (Open Mobile Alliance) have standardised architectures for the provision of services in an IMS network that includes the definition of an Application Server (AS) that offers some form of value added service. These bodies have been prudent enough not to standardise the services themselves, a move that would have stifled creativity among operators and vendors alike. Rather, they have defined units of self-contained functionality, or service building blocks, which can be re-used in service applications.

In 3GPP and ETSI parlance, these functionalities are known as service capabilities (SCs)[2], [10] whereas in the OMA, they are called enablers. (We use the acronym SC and the term enabler interchangeably in this paper, and more specifically use SC to refer to 3GPP interpretations and enabler to refer to OMA interpretations). The 3GPP has standardised some SCs including presence, conferencing and messaging [4], [6], [5]. The OMA, whose specifications for service enablers are the most advanced and complete according to [12], has standards for presence, location and XML Document Management (XDM) [17], [18], [16] among others. These SCs can be shared between different ASs so that the same basic functionalities are not duplicated needlessly. As self-contained functional elements, these SCs can be grouped together in order to compose larger, more complex services. For example, in [11], the conceptual designs of a multimedia messaging service (MMS) and a multi-party gaming service are described that involve the use and combination of presence, conferencing and messaging SCs.

The benefits of this form of service development as opposed to the monolithic approach of inserting all the logic for a service onto one node are clear. Since there is re-use of code, application development can be quicker since the developer has to worry less about the underlying mechanisms. Services are also more efficient and code redundancy is eliminated. These properties can have significant economic benefits for operators who employ this approach to service deployment. However, with the new opportunities that accrue, come new challenges. Application developers must be able to discover the capabilities of the network and what SCs are available to them. Access to these SCs must be regulated and policies must be put in place to allow the operator to grant and revoke access to the SCs. Also, we need to solve the problem of service invocation where SCs must be able to interact in order to collectively provide a complex service to the users.

This work was undertaken in the Distributed Multimedia Center of Excellence at Rhodes University, with financial support from Telkom SA, Comverse, Stortech, Tellabs, Amatole Telecom Services, Mars Technologies, Bright Ideas 39 and THRIP.

M. Tsietsi, A. Terzoli, and G. Wells are with the Department of Computer Science, Rhodes University, South Africa (e-mail: m.tsietsi@rucus.net; {a.terzoli, g.wells}@ru.ac.za).

In section II we provide some background on typical service invocation in the IMS. Sections III and IV introduce some of the initial work towards defining the functional architecture of an interaction manager and policy enforcer. In section VI we present a novel architecture that borrows from 3GPP and OMA standards. We proceed to interrogate this model in section VII and in sections IX and X we identify a platform through which we discuss how the model could be implemented using an existing platform known as JAIN SLEE.

II. Service Invocation in the IMS

The execution of a service in the IMS on behalf of a user involves interplay between the Home Subscriber Server (HSS), the Serving Call Session Control Function (S-CSCF) and the Application Server (AS) [8]. When a user registers, the S-CSCF downloads the user profile which contains the user's service profiles from the HSS. Each service profile contains sets of initial filter criteria (iFC) which determine service invocation. The iFCs specify service point triggers (SPTs) which are criteria that must be met by a SIP request in order for that request to be passed onto a particular AS. Thus, when a service request arrives from a registered user to the S-CSCF, the S-CSCF is responsible for evaluating the request against the iFC in order to determine which AS the request should be forwarded to.

This basic form of service handling can be referred to as *static service interaction management* because it is based on predefined and predictable rules and resolution information [1]. While this type of service management is effective, it is not suitable for cases where dynamic rules and conditions must be applied for service handling. Examples of dynamic conditions can be calendar events or some result which can only be obtained at run-time. In addition, it is only effective for trivial conditions where a single AS is to be invoked, but is not sufficient for more complex services where multiple SCs hosted on different ASs may be required in order to fully realise an integrated service [21].

III. Interaction Management in the 3GPP

It was determined by the 3GPP that in order to orchestrate complex services, an element would need to be defined that would be responsible for co-ordinating the interactions between multiple SCs. The name that was given to this element is a Service Capability Interaction Manager, or SCIM. In the 3GPP technical specification on network architecture, the SCIM is defined as an optional entity which performs the role of interaction management between SCs, but its exact behaviour was deemed to be outside of the standards [3].

In a subsequent technical report, the SCIM (referred to in the report as a service broker) was investigated in order to determine the impact of a SCIM on the overall IMS services architecture [1]. The report found that there were shortcomings in the current specification that would need to be revised for proper SCIM integration, and also proposed some potential layout models for where a SCIM would fit in the architecture with regards to the services and control layers. The lack of clarity from the perspective of the standards makes it difficult

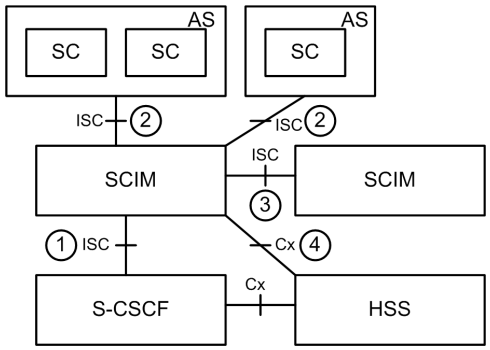

Fig. 1. Architecture for interaction management by SCIMs.

to understand precisely the way in which the SCIM performs its duty of interaction management, a role which may have far reaching implications for telecommunication operators.

The crucial aspect to the insertion of a SCIM into the IMS is procedural, and relates to how the SCIM interacts with other entities, and the nature of the interfaces between those entities and itself. In [11] and [20], four types of interactions have been identified, which are reflected in Figure 1. Interaction 1 between the SCIM and the S-CSCF is implemented using the IP Multimedia Service Control (ISC) interface, through which the SCIM is able to download the user profile. Interaction 2 occurs between the SCIM and the SCs on the designated ASs and also utilises the ISC interface, through which the SCIM is able to perform its function of integrated service invocation. Interaction 3 is between one SCIM and another to allow SCIMs in one IMS domain to trigger SCs in another IMS domain, though no protocol has been suggested that could be used to achieve this purpose. Interaction 4 is between the SCIM and the HSS through which the SCIM downloads and updates interaction logic which is described next.

The study related in [1] proposes an interaction logic between the SCIM and the SCs to perform dynamic service interaction management. It proposes that if several SCs are to be executed on different ASs, and the ordering of those executions is dependent on the results of processing performed by a previous SC in the chain, it should be possible to provision these dynamic interaction rules in the SCIM. An example would be the storage of conditional statements such as the following:

```
1. If(ServiceA, Success) SKIP ServiceB
2. If(ServiceB, Failure) SKIP ServiceC
3. If(ServiceB, Failure) SKIP ServiceD
```

In the scenario above, the priority order (ServiceA, ServiceB, ServiceC, ServiceD) is given, thus the integrated service begins at ServiceA. If this service exits with a success code, ServiceB can be skipped and the next service, ServiceC is invoked. Otherwise, if a failure code results, ServiceB is invoked. If a failure is in turn reported after ServiceB has executed, SIP routing ends and control is returned to the SCIM. Otherwise ServiceC and ServiceD are executed in sequence.

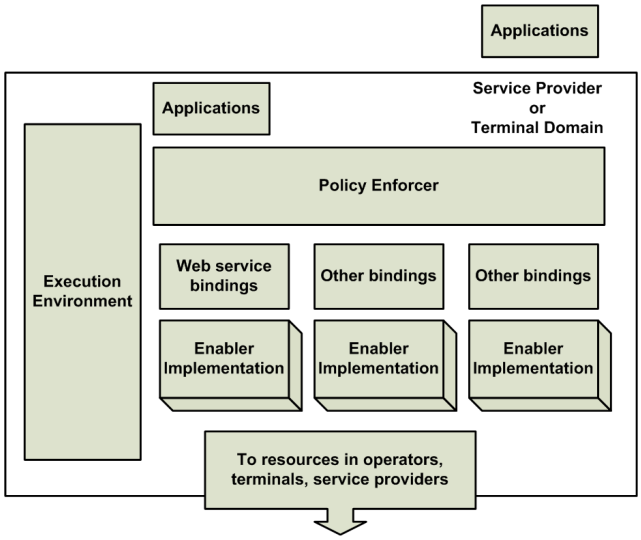

Fig. 2. OSE Architecture elements. Adapted from [19].

IV. Enablers in the OMA/OSE

In the introduction we mentioned that the OMA has been tauted as the standards body with the most complete and advanced set of enabler specifications. While the OMA recognises the need to develop basic building blocks for the development of services, it also seeks to define these enablers within the context of an overall architecture. Without it, the deployment and use of enablers would be complicated and expensive, and there would be high implementation cost for services seeking to use several enablers. For this purpose, the OMA has specified the OSE (OMA Service Environment) [19] which is an abstract architecture that consists of elements (including enablers) and the definition of the relationships between those elements. Figure 2 shows the architecture of the OSE.

In the OSE specification, an enabler is formally defined as "*a technology intended for use in the development, deployment or operation of a Service; defined in a specification, or group of specifications, published as a package by OMA*". The OSE requires enablers to define interfaces through which they can be accessed, allowing bindings such as web services, Java, or C to be developed in a production environment. An execution environment is also defined that allows the OSE to control enablers by providing functions such as process monitoring and software life-cycle management.

A notable element in the OSE is the policy enforcer. It is the responsibility of the policy enforcer to apply policies that govern appropriate access to enablers as well as to manage requests made to enablers for purposes such as billing, logging or the enforcement of user preferences and privacy. As the element with direct access to the enablers, the policy enforcer is well placed for composing enablers into higher level functions or complex services. It can also interface with a discovery enabler in order for applications in the home environment (HE) or outside the HE to discover enablers. Though the OMA has gone to great lengths to standardise enablers, there is no standard for a policy enforcer enabler, though the OSE suggests OMA's policy enforcement and enforcement management (PEEM) enabler as a possible stepping stone towards developing a policy enforcer enabler [19].

V. Consolidating Efforts From 3GPP and OMA

The material covered in the two preceding sections is exemplary of the potential for overlap between the standardisation work of the 3GPP and the OMA. While the 3GPP is the main driving force behind the evolution of the IMS, service capabilities such as presence and XDM are also part of the IMS. The OMA consists of players in the mobile services domain, and are spearheading the specification of enablers to be used in such contexts. However, the working groups in the two bodies do work together where there is scope for collaboration. It is generally agreed that the role of the OMA will be to generate requirements for the IMS and for the 3GPP to extend the IMS to meet these requirements [9].

However, there is an opportunity that we have identified that has not fully been taken advantage of by the two bodies in their efforts towards providing an architecture that makes extensive use of SCs for application services. The 3GPP has solved part of the problem by introducing a SCIM between the S-CSCF and the ASs, and by proposing a draft interaction management scheme based on building a history of SC invocations. The OMA has also solved part of the problem by introducing a policy enforcer into their OSE that handles authorisation and discovery of enablers by HE and third party services. What is needed is a model that consolidates both sets of work to provide a powerful, standards-based service interaction manager for an enabler-driven IMS network. The proposal borrows from the contributions from both the 3GPP and the OMA and is summarised in Figure 3. For simplicity, only those interfaces that have undergone modifications since Figure 1 are labeled.

VI. Proposed Architecture

In Figure 1 what stands out the most is that the SCs are no longer embedded in the ASs. This bold move is motivated by two main reasons. Firstly, if the SCs are co-located with the ASs, it is not possible to provide the attractive feature of enforcing policies on AS access to SCs. Secondly,

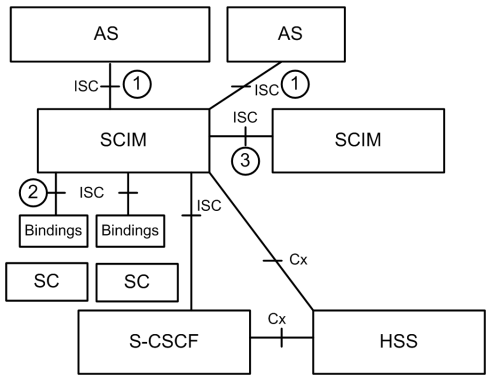

Fig. 3. Proposed architecture.

it provides the freedom to use services without having to bind development to specific enablers. This will also allow us to provide the feature of dynamic discovery of SCs in the network.

A. Interface between the SCIM and the SCs (Interface 2)

SCs are provisioned by the network operators. When a new SC is inserted, it must advertise its existence to the SCIM. While some enablers will be SIP based such as presence, some enablers will not be, such as XDM. As such, the ISC interface between the SC and the SCIM will not necessarily be SIP, a design option which is sanctioned by 3GPP specifications [7]. Various standard Internet protocols can be used here, depending on the nature of the SC. There are bindings that the SCIM uses to interact with the SCs. The SCs must advertise these bindings to the SCIM. If the SC wishes to request special policies for its access, it uses the same interface to communicate these. Otherwise, when the SC has registered with the SCIM, the SCIM can assign a default policy for those that have no special requirements.

B. Interface between the AS and the SCIM (Interface 1)

Between the AS and the SCIM, the ISC is still in effect, however its behaviour is also subject to modifications in our design. Since we have removed SCs, ASs would be interested in discovering underlying enablers in the network. This will help inform application development, allowing the developer to know what is available and what enablers the application will have access to during execution. The AS makes a discovery request over the ISC and the SCIM must evaluate this request against the policy framework. Note that unlike in the OSE, which has opted for a discovery enabler to assist in this process, this role in our design if fulfilled by the SCIM itself. The rationale behind this is that is simplifies the overall process since enablers are registering with the SCIM, thus it has information regarding all embedded enablers and enforces policies governing not only discovery of those enablers but their access and use. Whether the SCIM authorises the request or not, the ISC is used to communicate the response. The ISC interface would also be used to remove and request additional enablers to be accessed by the AS. These requests would also be subject to policies in the SCIM.

C. Interface between the HE SCIM and a remote SCIM (Interface 3)

When a SCIM from another network tries to perform service orchestration with enablers in another domain, the SCIM in the HE must be responsible for authorising that SCIM to discover and interact with the enablers. The interaction between the remote SCIM is similar to that of the HE SCIM and an AS in the home environment. It is possible however that policies that will be enforced in this case will be more stringent, and must certainly be affected by additional policies related to SLAs that may exist between the two domains.

VII. Service Orchestration

In addition to defining interfaces between the elements in the architecture, our design also defines a service orchestration model for the SCIM. Section III related a mechanism through which service chains are defined and dynamic conditions are evaluated in order to determine the next SC to be invoked. The mechanism that is proposed here is largely based on this mechanism, but takes into consideration some of the extensions that have been proposed in the previous section.

When a request is received by the SCIM from the S-CSCF, the SCIM examines the request against iFC from the user profile over the Cx interface and forwards the request to the relevant AS. The AS contains the logic for executing a task and uses the ISC application level interface to communicate with the SCIM on how to execute the service. When the SCIM receives a request from the AS, it examines the request and translates it into a service chain of enablers which must then be invoked in order to complete the task. Much like the 3GPP model, after each invocation, the SCIM can use dynamic values and non-SIP information to inform the next SC to be invoked, while communicating to the AS whenever needed. The AS in turn will also communicate back to the client in order to deliver the required functionality.

VIII. Identifying a Target Platform

In creating the model we have outlined, we are contributing new ideas to an area in telecommunications and service development that is in need of further investigation. Moreover, we are spurred on by the open research questions regarding interaction and policy management. As an initial step towards identifying a target platform in which to test our ideas, our attention was drawn to a Java platform called JAIN SLEE. It seemed to provide some of the functionalities that are required to implement a SCIM and realise the interfaces that a SCIM would need to have. In the next section we explore the architecture of JAIN SLEE and will highlight the reasons why we have selected it as a possible starting point towards developing a SCIM for IMS. It is important to note that we are not identifying JAIN SLEE as a generic environment for service orchestration as it is more suited to managing applications in a JAIN SLEE environment, but it does conform to our model and lessons can be drawn from it.

IX. JAIN SLEE

The JAIN Service Logic and Execution Environment (JAIN SLEE) is a Java architecture that has been standardised through the Java Community Process [13]. It defines a component model for structuring applications through re-usable object oriented components. JAIN itself refers to Java APIs for Integrated Networks, and represents an initiative to provide Java APIs for commonly used IP protocols, the aim of which is to integrate these protocols to enable converged services. JAIN SLEE utilises this concept to its benefit by allowing applications to use interfaces defined by JAIN APIs, as well as non-JAIN Java APIs [14].

In order to interact with external resources such as protocol stacks, network devices and databases, JAIN SLEE defines

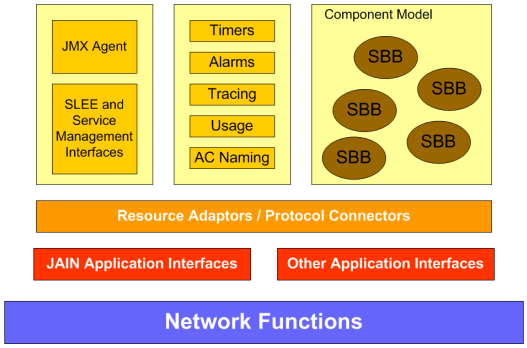

Fig. 4. JAIN SLEE Component Model.

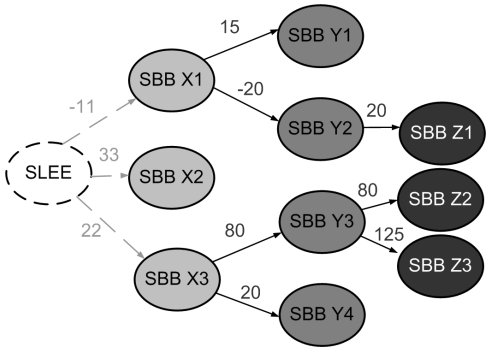

Fig. 5. SBB Graph example.

resource adapters which adapt the protocol messages into an appropriate format that can be handled by the SLEE. The reusable object oriented components are known as Service Building Blocks (SBBs) and form the basis for service provisioning. SBBs possess event handlers for each type of event they are interested in receiving. This is how converged services are enabled since an SBB can register interest in numerous types of events that are associated with different protocols.

As an event-driven architecture, JAIN SLEE has a single, logical event handler known as an event router. The event router receives events from event producers such as resource adapters and delivers them to event consumers such as SBBs. Trace, Alarm, Timer and other facilities are defined which also generate events that the SBBs can register for. As an example, an alarm clock SBB can receive events from the Timer facility in order to notify it when a Timer has expired so it can wake someone up. Figure 4 shows the architecture of JAIN SLEE.

SBBs in a SLEE can be likened to SCs in an IMS services environment. SBBs are also attractive in that they can provide multimedia services using JAIN and non-JAIN interfaces. In addition, the event generation, subscription and delivery mechanisms performed by the logical event router in the SLEE bear resemblance to the service integration function of a SCIM, which in essence delivers events to SCs according to some order and receives events from them. The next section elaborates more on the inner workings of the SLEE and maps these functions to the service integration requirements of a SCIM for IMS.

X. JAIN SLEE As an Interaction Manager

In this section, we will revisit the interfaces introduced in section VI to provide answers as to how JAIN SLEE can implement the needed interfaces of a SCIM.

A. Interface between the SCIM and the S-CSCF

For the ISC interface between the SCIM and the S-CSCF, JAIN defines an API for SIP known as JAIN SIP and as such, a JAIN SIP resource adapter could be used for this purpose. The JAIN SLEE specification, anticipating the popularity of this protocol, defines a recommended adaptation of a JAIN SIP v1.1 resource to the SLEE. It is also this SIP functionality which would allow for the parsing of SIP events in order to match against iFC for a particular service.

B. Interface between the SCIM and the SCs

For the SCIM to invoke a service, it has to be aware of the SCs involved in the composition of the service. Referring to work proposed by [20], it is possible to arrange SBBs in an hierarchical tree formation, with a root SBB behaving as a core SC (CSC). A root SBB can be associated with one or more child SBBs which would each fulfill the role of an auxiliary SC (ASC). JAIN SLEE defines an initial event as one that can cause the root SBB of a service to become instantiated. Thus a SIP INVITE message for example, can be used as an initial event for the instantiation of the CSC of a service.

JAIN SLEE uses an event delivery priority model in order to pass events from one SBB to another. When the root SBB is done performing some application logic based on the event, it passes control back to the SLEE which will use this priority model to relay the event to the next appropriate child SBB. The priorities are set for each child SBB by the service developer and must be integers in the range (-128, 127). Child SBBs with a higher positive event priority receive the events first, while those with a lower or negative event priority will receive the event later. SLEE implementations are allowed to determine the behaviour of the SLEE when the same priority is assigned to two or more vertices in the tree. Figure 5 illustrates the root SBB invocation and event delivery priority model of JAIN SLEE.

This event delivery priority model is similar to static service interaction management for SCIMs that has been defined, however it is possible to enable dynamic service interaction in JAIN SLEE as well. Firstly, JAIN SLEE defines an API for manipulating the priorities associated with child SBBs. *Getter* and *setter* methods exist which can be used for this purpose. Thus, if a root SBB is executing and a dynamic condition necessitates it, it can change the priorities associated with its children so that a different child SBB is invoked than the one that would have been had static rules been followed.

Each resource adapter, such as a JAIN SIP resource adapter, is capable of creating an entity known as an Activity Context. This can be used to store attributes which multiple SBBs can share, read and update at run-time. If an SBB can read the value of a shared attribute, and in certain conditions update that value, dynamic service interaction can be enabled. Instead of relying on an hierarchical, priority based model, we can flatten the interaction model and set a common priority to all

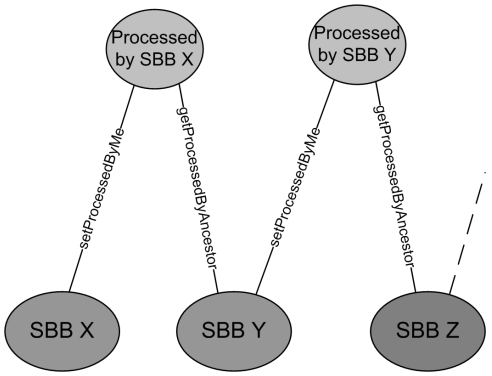

Fig. 6. Service chaining for dynamic interaction management.

SBBs involved in a service. Since SBBs can attach themselves to Activity Contexts, the Activity Context can behave as an event channel on which an SBB can fire an event and it will in turn distribute events to other SBBs that are also attached to it. When the event arrives at a particular SBB, that SBB can evaluate the shared attribute, and then decide if it processes it, or passes the event onwards.

Figure 6 illustrates this type of interaction, where three SBBs are involved in a service, each of which is attached to the same Activity Context. If SBB X receives the event, it sets the shared attribute to reflect that it has modified it. When another SBB, say SBB Y receives the event, it can examine the shared attribute and decide if it will process the event further, or return control to the SLEE. The same applies to SBB Z. Thus both static and dynamic service capability interaction management can successfully be performed using JAIN SLEE due to its support for service capability prioritisation and management of shared attributes.

In terms of policy management between the SCIM and the SCs, we make note of the suggestion by the OMA in their PEEM enabler specification to use protocols such as XDM, FTP or SIP presence mechanisms [15]. All three of these protocols however could equally be supported by providing the appropriate JAIN resource adapters for them. As an XDM agent, JAIN SLEE would allow the definition of XML application usages for policies. The XCAP protocol which is used to address segments of an XML document would be used for the management of these policies. FTP would assist in the uploading, replacing and deletion of policies in file format in the SLEE. A JAIN SIP resource adapter would be used in the SIP case, where presence events would be used as an extra channel for SCs to manage and subscribe to XDM-based policies.

C. Interface between the AS and the SCIM

Between the AS and the SCIM we propose the same approach as between the SCIM and the S-CSCF of implementing the ISC using the JAIN SIP adapter. If the AS were interested in querying the SLEE for policies using for example XCAP requests, JAIN SLEE would be able to provide such information in an XCAP response by evaluating XML files created for enablers.

D. Interface between the HE SCIM and remote SCIM

Between the HE SCIM and a remote SCIM we propose the same approach as between the SCIM and the S-CSCF of implementing the ISC using the JAIN SIP adapter. In a similar manner to above, the SCIM from the remote domain could also use XCAP requests to query policies on enablers.

XI. Conclusion

In this paper, we have presented a services architecture for NGNs and the functional design of a service broker in this context. We have compared the efforts of organisations such as the 3GPP and the OMA and identified the similarities between the two. We have used work from them to present a possible enabler-based services architecture using current standards wherever possible. Implementation concerns such as privacy and user preferences have been catered for in this architecture by defining a policy function in the SCIM. In an effort towards testing our ideas, we have mapped the interfaces and functions of the SCIM to a practical platform, namely JAIN SLEE, and shown how it can be used to implement the model we have presented. Future work will focus on the use of JAIN SLEE to implement the functions defined, which will help inform and validate our design, which may lead to modifications and extensions to the work discussed here.

References

[1] 3GPP, "TR 23.810: Study on architecture impacts of service brokering - Release 8," 3GPP, Tech. Rep. TR23.002v8.0.0, September 2008, http://www.3gpp.org/ftp/Specs/archive/23_series/23.810/.
[2] ——, "TS 22.105: Services and Service Capabilities - Release 9," December 2008, http://www.3gpp.org/ftp/Specs/archive/22_series/22.105/.
[3] ——, "TS 23.002: Network Architecture - Release 8," December 2008, http://www.3gpp.org/ftp/Specs/archive/23_series/23.002/.
[4] ——, "TS 24.141: Presence service using the IP Multimedia (IM) Core Network (CN) subsystem - Release 8," December 2008, http://www.3gpp.org/ftp/Specs/archive/24_series/24.141/.
[5] ——, "TS 24.147: Conferencing service using the IP Multimedia (IM) Core Network (CN) subsystem - Release 8," December 2008, http://www.3gpp.org/ftp/Specs/archive/24_series/24.147/.
[6] ——, "TS 24.247: Messaging service using the IP Multimedia (IM) Core Network (CN) subsystem - Release 8," December 2008, http://www.3gpp.org/ftp/Specs/archive/24_series/24.247/.
[7] ——, "TS 23.228: IP Multimedia Subsystem; Stage 2 - Release 9," September 2009, http://www.3gpp.org/ftp/Specs/archive/23_series/23.228/.
[8] ——, "TS 29.228: IP Multimedia Subsystem Cx and Dx interfaces; Signalling flows and message contents - Release 8," March 2009, http://www.3gpp.org/ftp/Specs/archive/29_series/29.228/.
[9] G. Camarillo and M. Garcia-Martin, *The 3G IP Multimedia Subsystem (IMS)*. CRC Press, 2008.
[10] ETSI TISPAN, "TR 181.004: NGN generic capabilities and their use to develop services," ETSI TISPAN, Tech. Rep. TR181.004v1.1.1, March 2006.
[11] A. Gouya, N. Crespi, and E. Bertin, "SCIM (Service Capability Interaction Manager) implementation issues in IMS service architecture," in *ICC '06: IEEE International Conference on Communications*. IEEE, June 2006, pp. 1748–1753.
[12] M. Ilyas and S. Ahson, *IP Multimedia Subsystem (IMS) handbook*. CRC Press, 2009.
[13] JCP, "JSR 22: JAIN Service Logic Execution Environment API Specification," Available Online, http://jcp.org/en/jsr/detail?id=22.
[14] S. B. Lim and D. Ferry, "JAIN SLEE 1.0 specification, Final Release," Avaliable Online, 2004, http://jcp.org/aboutJava/communityprocess/final/jsr022/ind- ex.html.
[15] OMA, "Policy Evaluation, Enforcement and Management Architecture - Candidate Version 1.0," August 2008, http://www.openmobilealliance.org/Technical/release_program/peem_v1_0.aspx.

[16] ——, "Enabler Release Definition for Location in SIP/IP Core," August 2009, http://www.openmobilealliance.org/technical/release_program/locsip_v1_0.aspx.

[17] ——, "Enabler Release Definition for OMA Presence SIMPLE," September 2009, http://www.openmobilealliance.org/technical/release_program/Presence_simple_V2_0.aspx.

[18] ——, "Enabler Release Definition for XML Document Management," August 2009, http://www.openmobilealliance.org/Technical/release_program/XDM_v2_0.aspx.

[19] ——, "OMA Service Environment- approved version 1.0.5," October 2009, http://www.openmobilealliance.org/technical/release_program/ose_v1_0.aspx.

[20] Y. Wang, K. Yangi, R. Li, and L. Zhang, "An improved SCIM-based service invocation mechanism for integrated services in IMS," in *Mobility '08: Proceedings of the International Conference on Mobile Technology, Applications, and Systems*. New York, NY, USA: ACM, 2008, pp. 1–5.

[21] N. Xia and W. J. Zhai, "Study on the IMS service broker," in *ICCIT '08: Third International Conference on Convergence and Hybrid Information Technology*. IEEE, November 2008, pp. 340–343.