A Semantic Enhanced Service Exposure Model for a Converged Service Environment

Cuiting Huang, Gyu Myoung Lee, and Noël Crespi, Institut Telecom — Telecom SudParis

ABSTRACT

Service exposure, as a prerequisite for the reuse of existing services, plays a significant role in next-generation service delivery. However, some issues such as cross-domain interoperability and user-centricity are still unresolved. Therefore, this article proposes a semantic enhanced converged service exposure model that enables services derived from different domains (i.e., telecom/web/device/user) to be integrated into a composite service regardless of their underlying heterogeneities. Meanwhile, the user-centric feature is enhanced by applying the semantic annotation to both the service description and the user request, as well as empowering a user to publish and share his/her created service in a user-friendly way. Moreover, a variety of interfaces, including APIs, widgets, and natural-language-based tools, are provided to satisfy the multilevel user requirements, enabling users to profit from a highly personalized, meaningful communication and interaction experience.

INTRODUCTION

The telecom/Internet market is more and more customer-driven. How to deploy revenue-generating and competitive services in a fast-paced environment, reduce both their cost and their time to market, and optimize their life cycle are the very relevant, challenging questions for network operators and service providers. Service composition, which enables the reuse of existing services to create novel services, has been widely recognized as an important trend for next-generation service delivery. Service exposure plays a significant role in the evolution of service composition, since enabling a service to be reusable means that this service needs to be known and accessible by users and/or developers. Various service exposure solutions have been proposed. However, current solutions mainly target a specific group of developers and are not friendly enough to users. In this article, we propose a converged service exposure framework for user-generated application creation in a more open and sustainable marketplace. This framework is intended to be one in which even non-professional users can easily reuse existing services, capabilities, and resources to create new services, and thereby profit from a highly personalized, meaningful communication and interaction experience.

SERVICE EXPOSURE: A BACKGROUND

In the Internet domain, most service composition solutions are based on service-oriented architecture (SOA). The initial objective of SOA is to facilitate service integration across independent entities or organizations by using a set of service publication and discovery facilities. Web services are mainly exposed through application programming interfaces (APIs), which are typically defined as a set of Hypertext Transfer Protocol (HTTP) request messages along with a definition of response messages, following Simple Object Access Protocol (SOAP), Representational State Transfer (REST), or Extensible Markup Language — Remote Procedure Call (XML-RPC) principles. To facilitate service discovery, a set of mechanisms such as Web Service Description Language (WSDL), Universal Description Discovery and Integration (UDDI), as well as semantic annotation and ontology technologies (e.g., Web Ontology Language [OWL], Web Service Semantics [WSDL-S], and Semantic Annotations for WSDL and XML Schema [SAWSDL]) have been specified for helping the publication component to better describe the services and the discovery component to more easily identify the right content/service matching user's requests, context and preferences. Relying on these basic service exposure facilities, a variety of advanced solutions have been proposed by academic and industry specialists to improve the service discovery efficiency and user experience by associating with some auxiliary information such as context information [1], quality of service (QoS) [2], user intent [3], and service reputation [4].

0163-6804/12/$25.00 © 2012 IEEE

Company	Project	Exposed services
BT	Ribbit	• SMS, voice call, conference call, presence, location
Orange	Orange Partner	• Advanced APIs: contact everyone, multimedia conference, device capability enabler, location, SMS internet, click-to-call, M2M, API manager • Instant APIs: location, SMS, MMS, click-to-call, click-to-conference, voicemail, voicemashup, email, mobeasy • Personal APIs: authentication, personal calendar, personal contacts, personal content, personal favorites, personal messages, personal photos, personal profile, personal rich profile, payline
Deutsche Telekom	Developer Garden	• Send SMS, conference call, voice call, IP location, voice record lab
Telefonica	BlueVia	• O2 litmus APIs: network connection status, O2 litmus location, manage device, view account status, manage post pay bolt on's, view device compatibility, O2 litmus customer charging • Open moviforum APIs: receiving SMS, receiving video call, SMS send, MMS send, SMS 2.0, copiagenda • Developers movistar APIs: SMS, MMS, WAP-push
Vodafone	Betavine	• SMS, WAP-push, application trigger messages

Table 1. *Telecom service exposure platforms and their exposed services.*

In the telecom domain, an application-independent capabilities layer is separated from the control layer so as to allow the reuse of the existing telecom services, as specified in IP multimedia subsystem (IMS). Besides the original serving-call session control function (S-CSCF)-based service chaining mechanism and the solutions that rely on an additional entity for service brokering (e.g., Service Capability Interaction Manager [SCIM], OMA Service Environment [OSE]) which are purely Session Initiation Protocol (SIP)-based, another recent trend is to use web technologies to access and compose telecom capabilities. This trend is closely tied to particular service exposure frameworks such as Parlay/OSA Gateway, ParlayREST, Next Generation Service Interface (NGSI), and OneAPI specified by the standards bodies, which aim to make service capabilities inside an operator's network accessible for external applications by translating telecom protocols and capabilities into a set of APIs. Along with standards bodies, operators, service providers, and the academic and telecom communities are trying to propose their solutions for catering to this trend. For instance, Telefonica developed a service platform [5] to expose telecom capabilities through RESTful APIs and the Portable Service Element; ORACLE deployed a Communications Service Gatekeeper [6] to provide operators a single, centralized service exposure platform with diverse choices of application interfaces, including native telecom interfaces, SOAP web services, SOA web services, and RESTful web services; FOKUS defined a JavaScript-based Web 2.0/Telco Service Exposure Gateway [7] relying on Parlay/OSA Gateway and OSE specifications to expose telecom enablers to Web 2.0 mashup; the Wholesale Applications Community (WAC) [8] was established in 2010 by 24 operators to create a unified open platform that provides a single interface to access device functionalities and network resources by using mobile web technologies. Moreover, there are other user-friendly exposure platforms provided by operators as summarized in Table 1 (but not exhaustively). All of these listed platforms expose the telecom services through APIs and/or widgets.

Nevertheless, current service exposure solutions are mainly designed for a specific group of developers and/or a specific network. Thus, when a developer or a user wants to create a converged service that involves telecom, web, or device-based services, he/she has to search for services from among several different service platforms, understand the different service description patterns, as well as their underlying heterogeneities. The need to manipulate different platforms increases the complexity of integrating the heterogeneous services into a unified composite service. Moreover, most of the current solutions aim at professional developers, even some so-called user-centric service creation platforms (e.g., YahooPipe, Microsoft Popfly, Mashmaker, MARGMASH, and MARMITE). Relying on these service exposure solutions, these platforms are still too complicated for ordinary users. To encourage user involvement instead of exclusivity and really include non-professional users, user-centric features need to be significantly enhanced.

An Overall Architecture for a Converged Service Exposure Platform

The main objective of this article is to propose a semantic enhanced converged service exposure framework for a variety of services, including telecom\web\device\user-generated services, thereby enabling a new ecosystem for the creation of innovative services through user-centric modes. Figure 1 illustrates our proposed reference model.

This service exposure framework acts as an intermediary for the adaptation and abstraction

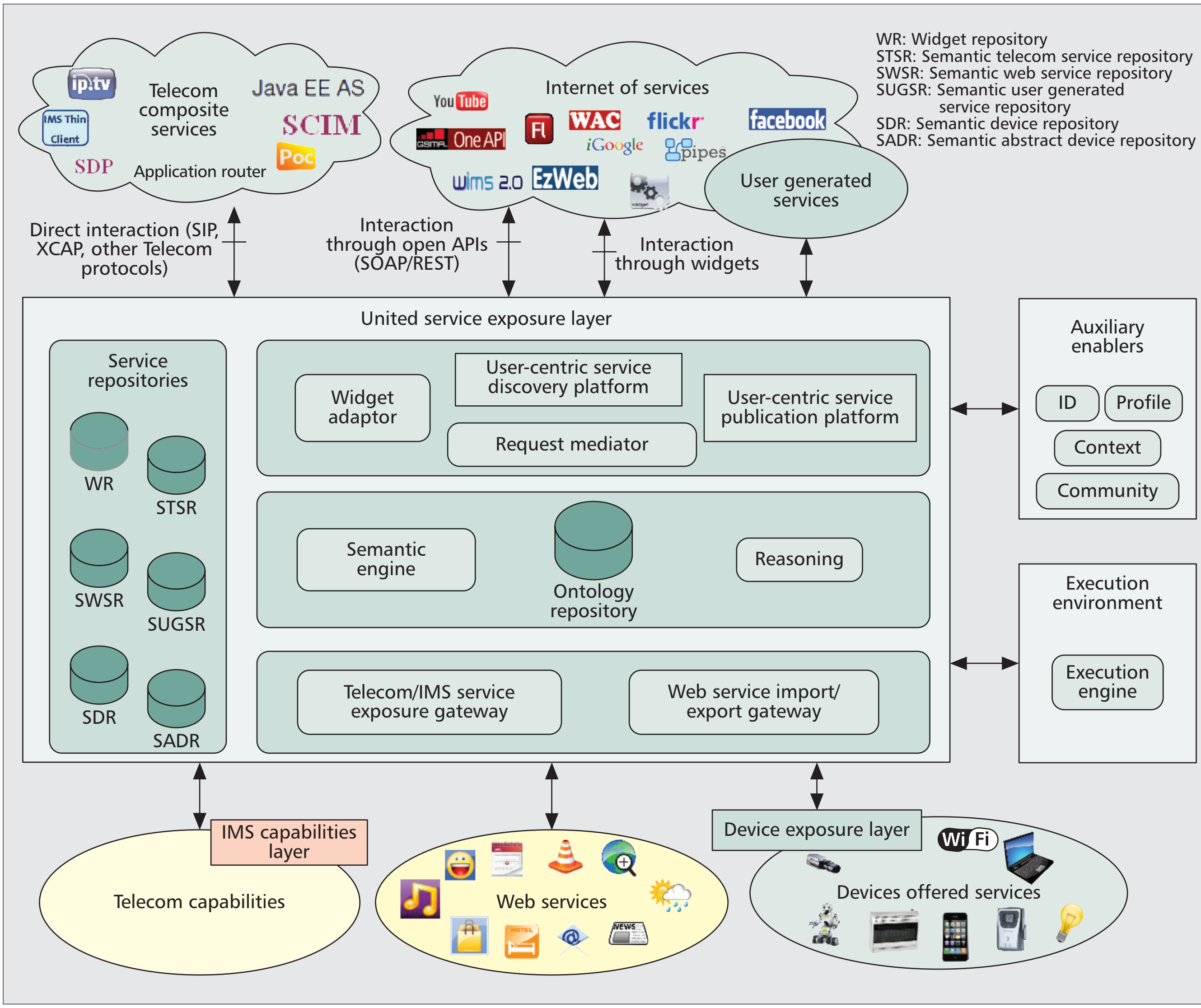

Figure 1. *Overall architecture for a converged service exposure*

of the services provided by telecom, Internet, and smart space devices, and enables the convergence of the different kinds of services.

There are four groups of entities that can be identified:

- User-centric adaptation group: user-centric service discovery platform, user-centric service publication platform, request mediator, and widget adaptor
- Semantic enrichment group: semantic engine (SE), ontology repository, and reasoning
- Service exposure and access control group: various exposure gateways
- Service repositories: a set of repositories for storing the semantic-rich service description files

During the service life cycle, all of these entities collaborate tightly to enable seamless interaction among the heterogeneous services, and facilitate their reuse by both professional and non-professional users. In the following sections, the cooperation among these entities is illustrated by considering the different kinds of services' exposure and discovery.

Telecom Service Exposure

Generally, three kinds of users/developers reuse telecom services and capabilities: telecom-savvy developers, web-service-familiar developers, and non-professional users. Accordingly, telecom service exposure should satisfy the different levels of user requirements by providing different interfaces: native telecom-based interface, web-based open API, and user-centric interface.

Telecom service access: As shown in Fig. 2, a telecom service exposure gateway (TSEG) is used to connect telecom capabilities with outer-world applications; thus, it provides a network abstraction function and exposes telecom capabilities through APIs. Briefly, the TSEG intercepts the incoming web-service-based requests (e.g., HTTP), and the protocol adaptor in the TSEG translates the requests into telecom-specific protocol-based messages (e.g., SIP). These translated messages are then forwarded to the corresponding components for service invocation.

Since an operator's API is used not only by internal developers, but also by third-party developers and non-professional users, access

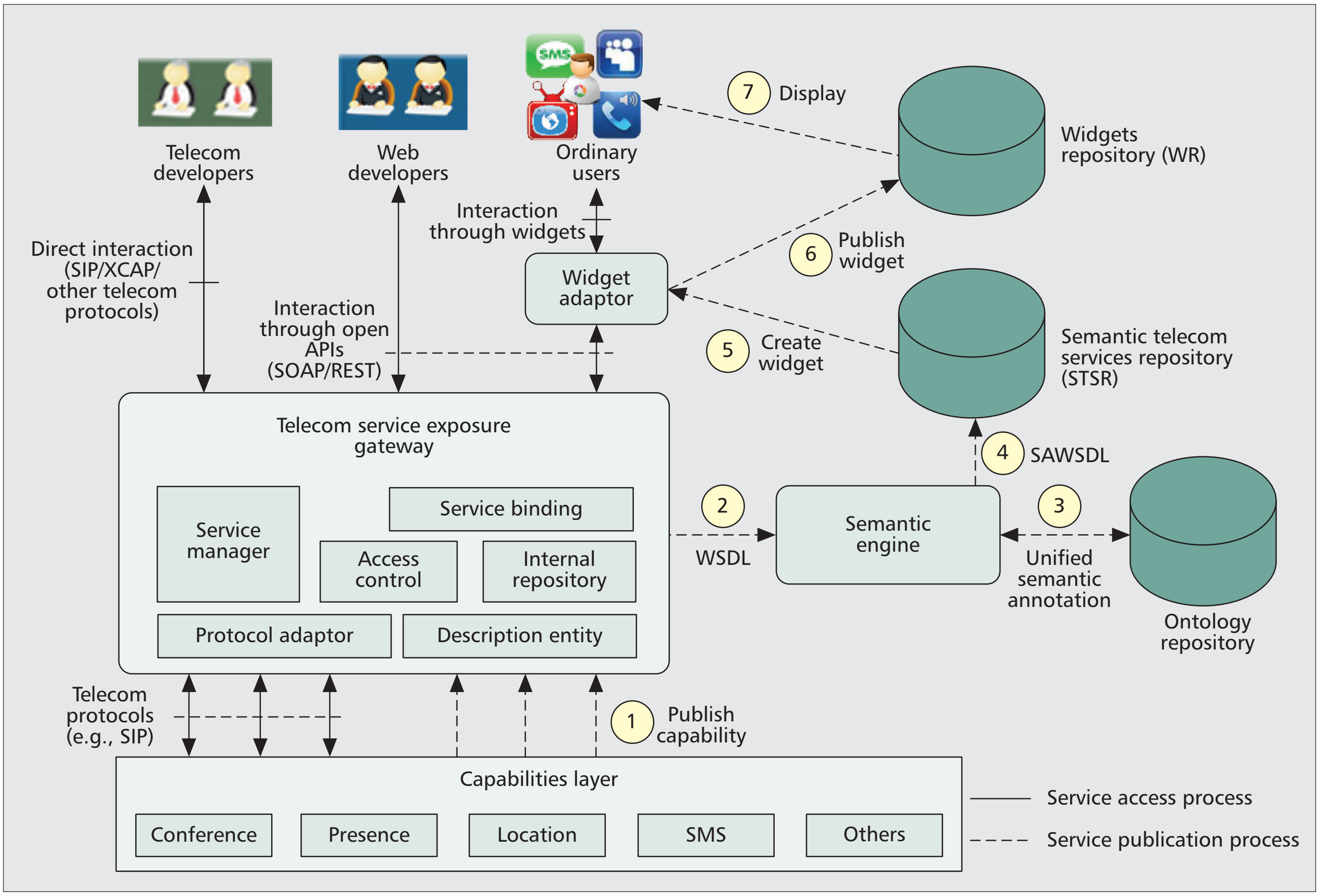

Figure 2. *Telecom service exposure*

control is necessary. This means when the TSEG receives a service invocation request, the service manager retrieves the corresponding service information from its internal repository, and simultaneously contacts access control in order to verify the user's identity or credentials. After this verification, the TSEG performs the necessary authorization and authentication. Users are able to access the different network capabilities according to the different authentication and authorization results.

The internal repository in the TSEG is used to store the actual location and relevant confidential information for access control. It thus hides the network topology and details from the outer world. In fact, there are two kinds of service description repositories: the internal repository and the global repository (Semantic Telecom Service Repository) for storing the general functional and non-functional descriptions with semantic annotations that are retrievable by external users and developers. The two repositories provide double protection for the underlying telecom network, since malicious attackers cannot locate the internal servers, and malicious messages can be intercepted and handled by the TSEG.

For the telecom developers, since their applications mainly use the same technologies as in telecom functions, the relevant messages invoke only access control instead of invoking all the functionalities provided by the TSEG.

Telecom service publication and discovery: A service exposure platform should also enable rapid and quality-assured service discovery. This requirement calls for the possibility of publishing a telecom capability in a commonly understandable and comprehensive way, thereby making it discoverable and usable by other services. Such a service publication process can rely on WSDL, for example.

When operators publish their capabilities, they need to publish the functionality and access requirements of their capabilities to the description entity inside TSEG. This service description file is then split into two description files: internal and external. The external file is transferred to the SE. The semantic annotation is added into the external service description file by utilizing the predefined domain ontology stored in the ontology repository. Finally, the semantic annotated service description file is stored in a central service registry (e.g., a UDDI registry). The mapping information for maneuvering between the internal and external files is retained in the internal repository. This means that users can use similar methods as web services for discovering these telecom services.

Web Service Exposure

The semantic annotation mechanisms have been widely applied in service description for providing a new way to represent the knowledge that

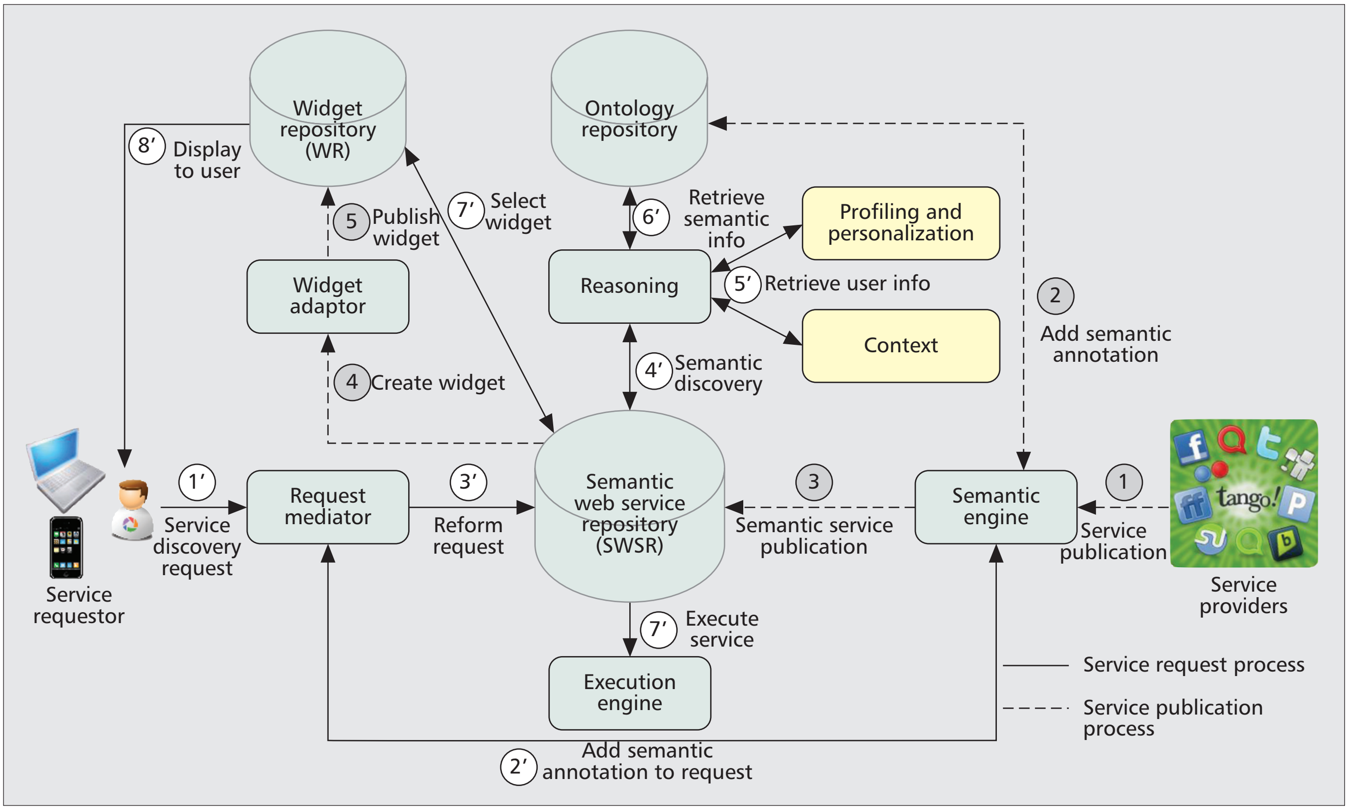

Figure 3. *Web service exposure.*

empowers advanced reasoning mechanisms [9]. However, there is no universal domain ontology that can act as a shared web service vocabulary dictionary for the annotated terms in service semantic description. Obviously, if service instances correspond to the same service provider, the composition and discovery of the relevant services are feasible. However, if a developer or a system tries to compose several services derived from different service providers with different semantic annotations, it would indeed be a burdensome task.

In our proposed solution (Fig. 3), when service providers publish services, they can use their preferred service publication mechanisms. This service description file has to pass through the SE. The SE then contacts the ontology repository, which identifies the semantics for describing service's functionalities, workflow, and invocation to translate the received service description file into a unified and semantically annotated service description file (e.g., SAWSDL). During this process, a user-friendly interface may be displayed which asks the service provider to add some additional information including the non-functional features of the service.

If this published service is represented as a widget, the semantic service description file is sent to the widget adaptor, and then a corresponding widget is generated and stored in the widget repository.

The semantic service annotation is used not only for the service description, but also for the service request. Since users requesting services are likely to possess a wide range of technological competence, the incoming service request could be in the format of simple natural language text (e.g., "send SMS") or compliant with the service annotation ontology. Therefore, a request mediator has been added to make incoming requests easy to understand. When the request mediator receives a service request, it transforms the request into a structure that corresponds to the semantic service annotation mechanism adopted for service publication.

The reasoning engine collaborates with the semantic web service and ontology repositories for the discovery and selection of appropriate services in accordance with the relevant information retrieved from the profiling and personalization and context enablers. Finally, a list of corresponding services is sent to the user for further selection, or a selected service is sent to the execution engine to be executed directly according to the user's configuration and the tools used (e.g., Natural Language Composition Environment). If a user wants to select the relevant widgets, the list of selected services is sent to the widget repository, and then a list of the corresponding miniature widget icons is displayed to the user.

Device Service Exposure

Along with the popularization of the Internet of Things (IoT), devices are more and more involved in service composition, providing real-time data about the physical world via gateways or service-enabled interfaces (e.g., SOAP or RESTful APIs) [10]. To make use of these device-offered services, the first step is to make them machine-meaningful. Generally, devices

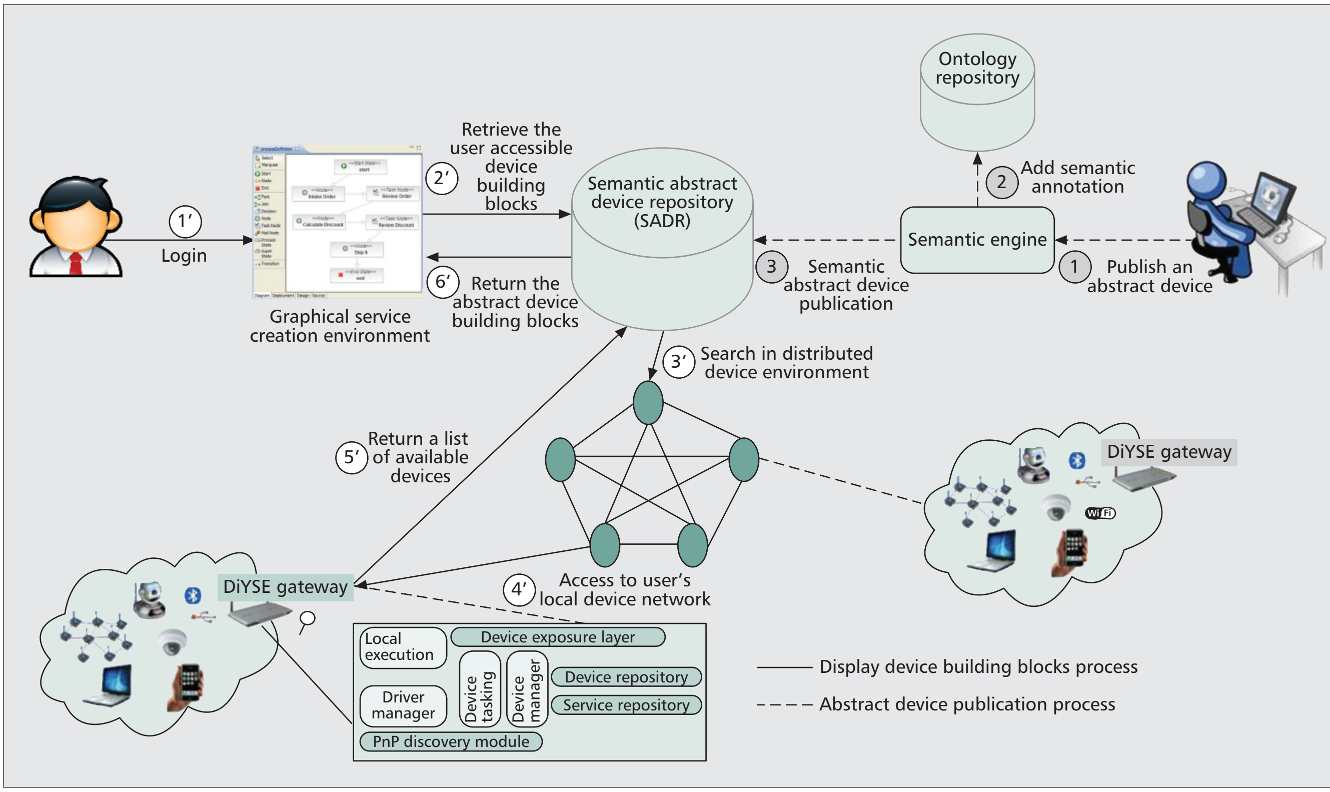

Figure 4. *Device service exposure.*

(intelligent equipment, sensors, actuators, etc.) interconnect with each other in smart space. Their resource-constrained aspects (e.g., limited computation, communication, and storage capabilities) make them more constrained than telecom or web services. Moreover, compared to telecom and web services, devices' availability may change more frequently. This highly dynamic information is impossible to capture in a service description file stored in a global repository, because this leads to an unbearable level of network traffic. Therefore, some form of gateway is adopted to enable communication between these devices and external applications, and to handle the runtime discovery of devices and their services. In consideration of these scenarios, we divide the device service exposure into two levels: local service exposure, relying on local gateways, and global service exposure (GSE), which reports the basic information about the devices and their services to the global repositories via local gateways.

To facilitate the introduction of device service exposure, we adopted the DiYSE Gateway [11] as an example for local service exposure, as illustrated in the amplified portion of Fig. 4.

From the viewpoint of the global network, there are many local gateways dispersed in the network, managing their own local networks using disparate technologies. Furthermore, when a user or developer is creating a new service, he/she may not be in the same network as the devices and is not necessarily familiar with the particular characteristics. This distributed device component information needs to be published in a globally comprehensible manner to enable device capabilities to be discovered even when they are located behind local gateways, executed and composed in a unified manner.

Correspondingly, in terms of the service publication process, there are two kinds of device capability publication processes: abstract and concrete. An abstract device means a virtual device that only contains the generalization information of one kind of device, and can map to several concrete devices. For example, "camera" is an abstract device, and it can map to several concrete devices provided by different manufacturers such as Sony, Logitech, and Canon, and owned by different persons such as Alice and Jon. The impetus for proposing such abstract device capability publication is that the number of devices in the network is much larger than telecom and web services, while most have the same or similar functionalities. Storing all of these concrete devices in a global repository will make it difficult for them to be discovered and managed.

Abstract device publication: The abstract device publication is handled by professional developers or service creation platform providers. They send a device description file that only contains the generalization information of one type of device to the SE. The SE contacts the ontology repository to add the necessary semantic annotations. The transformed service description file is then sent to the global semantic abstract device repository.

Concrete device publication: Device-offered services generally can be viewed or controlled by users via a web browser, for example, as specified in Universal Plug and Play (UPnP) for representation. This makes it easy for users to get the service description file via the same device-

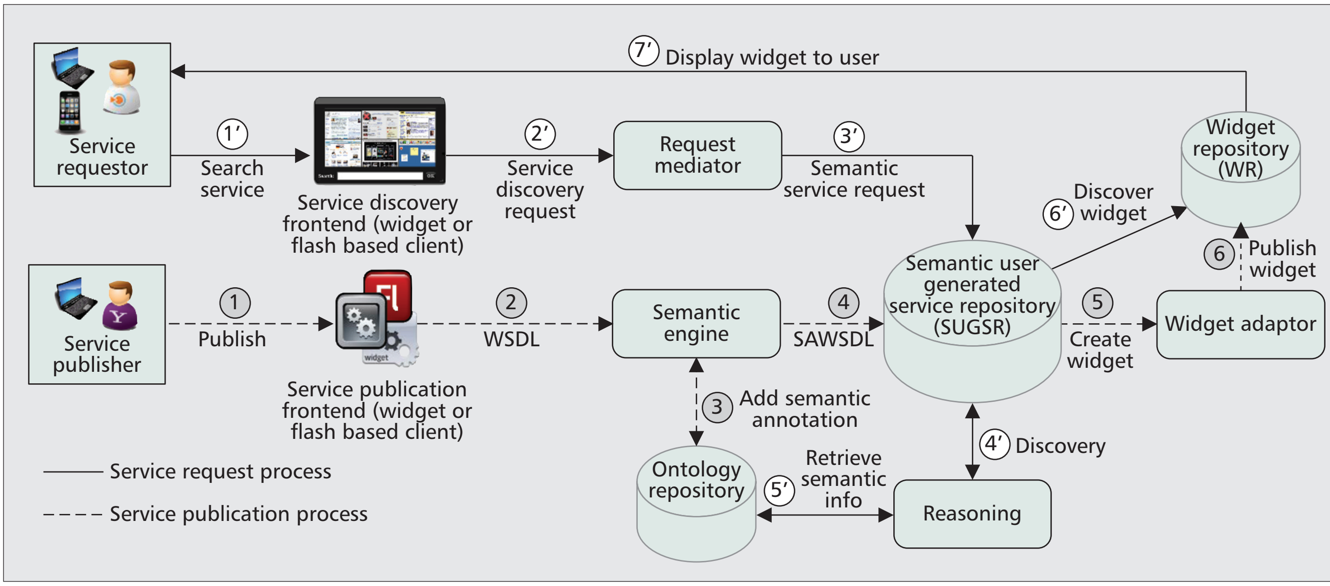

Figure 5. *User generated service publication and discovery.*

monitoring page. Once the device description file is retrieved, SE adds the semantic annotations to the file and sends it to the semantic concrete device repository. Users can also publish their devices as widgets. This situation is actually the publication of a concrete rather than an abstract device.

If a user wants to reuse device-offered services to create a service, when he/she logs into a graphical service creation environment (SCE), the SCE contacts the GSE framework with the user's ID. The GSE framework can then search the distributed gateways using flooding to determine which specific local gateway this user can access and which specific device behind that gateway can be invoked. The corresponding abstract device building blocks are then displayed in the user's SCE. After the user has created a composite service by chaining the different building blocks, the service discovery and selection processes are handled by the corresponding components. Telecom and web-based services are discovered and selected by the GSE framework's components as indicated above, and for device-offered services, the service discovery request is transferred by the GSE framework to the corresponding local gateways, and then the device repository and device discovery module inside a local gateway select the relevant device description files and send these selected files back to the GSE framework. Finally, the reasoning component cooperates with the GSE framework, as well as the profiling and personalization and context enablers to select the most appropriate concrete device(s) for the user.

User Generated Service Exposure

One goal for this service exposure model is to enhance the user-centricity feature. When a user generates a new service, it is normal that he/she would like to share it with others, and/or this newly created service may be reused by other services.

The main difference between the service exposure of user-generated services and the telecom/web/device service exposure is that the targeted actors are amateurs or non-professional users, which requires the exposure and discovery processes to be as simple as possible and the underlying mechanisms to be transparent to users.

Figure 5 presents our solution for this kind of service publication and discovery process.

Service publication process: Before the publication process, the user has to be identified by the system, and his/her information (e.g., publisher's name) is automatically added to the semantic service description file.

Next, the user initiates the service publication process by interacting with a user-centric service publication front-end, which can be invoked as a widget or a flash-based client. He/she can provide the uniform resource locator (URL) of the functional description of his/her service to the service publication client if he/she has already created this service description file. Considering that the creation of a WSDL file may be difficult for non-professional users, this service publication front-end should also provide an easier way to generate it. For example, a template could ask the user to fill in the necessary information about his/her created service (e.g., service's name, input/output). This information would then be extracted and completed by the default collocation information to create a formal functional description file. Some non-functional parameters and business models (e.g., the cost for use of this service) can be added to the service description file.

The service description file is then transferred to the SE. Finally, a semantic service description file is generated by linking the functionalities with semantic concepts, and is then sent to the semantic user generated service repository. When a service cannot be parsed automatically, a page on which to add the semantic annotations manually will pop up with a list of semantic concepts for the different options. If no relevant semantic concept matches the service, the ontology update process, which is out of the scope of this article, is invoked for adding a new concept.

Service discovery process: This functionality can be invoked through an embedded widget or a flash-based client in a user's personal service environment (PSE) or SCE. It supports both natural language and complex searches based on semantics. Therefore, similar to web service discovery described above, the incoming service request is translated into a unified semantic-based request by the request mediator, and then sent to the semantic service repository. The displayed services vary depending on the users' intent. For users who just want to invoke a service, only the widget-based services are displayed in his/her PSE, since some back-end services have no front-end with which to use them. For users who want to create new services, both the widgets and the back-end services, as well as the relevant UI components, are presented as building blocks in the SCE. Moreover, some additional information (e.g., the publishers' names and the costs of these services) extracted from the semantic service description files is also displayed to users.

Case Study and Discussion

As a first proof of concept, a simplified version is being implemented by reusing some entities implemented in the DiYSE project such as the DiYSE-specific domain ontology based on the DOGMA-MESS methodology, the THERE device gateway, and the profiling and personalization and context enablers. An MSN robot was implemented as a natural-language-based SCE. A backend service composer is implemented as a contact; thus, when a user wants to create a service, he/she just needs to tell this contact what he/she wants to do. For example, when Bob enters "If the outside temperature is less than 10°C, turn on the heater in my house and send my house current temperature to me," an abstract service defining the involved functionalities is generated by the service composer, which cooperates with the SE and request mediator. In this case, web service (weather), device-based services (heater and temperature sensor), and telecom service (SMS) are selected. The service composer then contacts the service exposure platform to discover and select the most appropriate services for the abstract service by collaborating with the SE, ontology repository, service description repositories, and relevant device gateways. After all the concrete services have been selected, the MSN robot validates the concrete service logic and executes this composite service by invoking the corresponding atomic services (e.g., Yahoo! weather, Bob's heater and temperature sensor in his house, and Orange's SMS). In this use case, several different kinds of services are invoked; however, the user only needs to use a unified interface to discover services, instead of having to surf different websites to discover the different services and spend time learning how to use them.

The proposed service exposure model relies on the enhancement of user-centric and convergence features, as well as on providing a unified access to telecom/web/device/user-generated services, meanwhile respecting the diversity of the service market. It thereby enables third-party service providers, developers and users to reuse the existing services, using the tools or technologies they prefer while following SOA, resource-oriented architecture, or widget-oriented architecture principles. By separating service discovery and selection functionalities from the SCE, this service exposure model not only releases the SCE from the cumbersome service discovery task, but also enables the reuse of larger-scale services, regardless of their underlying heterogeneities, thus further improving the sharing of different kinds of services among different platforms. It also provides the possibility of applying enhanced controls for service exposure in accordance with different operators' or service providers' requirements via gateways. The user-centric aspect is enhanced by using widgets for service publication, discovery, and composition, thereby greatly facilitating the service generation process, which encourages more users to contribute to the service ecosystem. The adoption of semantic annotation for both service description and service requesting improves service discovery efficiency and accuracy.

To fully realize this converged service exposure model, some challenges still require further attention. These challenges include defining a widely acceptable common data model for all types of service descriptions, and persuading operators and service providers to share their service information. An efficient business model is also indispensable, considering that the investments from operators and service providers for technologies' enhancement are revenue-driven. If users could also profit from their contribution to new services, it would greatly encourage everyone to become involved in this new service ecosystem. The successful Apple App Store reveals this great potential. However, the single access to a service exposure platform may become a bottleneck when the number of services increases dramatically, and be susceptible to malicious attacks. The party that will control this kind of service exposure platform is still unresolved.

To fully realize this converged service exposure model, some challenges still require further attention. These challenges include defining a widely acceptable common data model for all types of service descriptions, and persuading operators and service providers to share their service information.

Conclusion

Based on the user-centric and convergence trends in next-generation service delivery, this article has explored a semantic enhanced service exposure model for enabling different kinds of services to be shared, independent of their original domain (i.e., telecom/web/device/user) and supporting technologies. It facilitates service discovery and access for both professional and non-professional users, thereby improving both the user experience and user involvement. It also simplifies service implementation and optimizes the service life cycle.

Further work will be focused on defining a widely usable service description data model, and more efficient service discovery and selection algorithms. A distributed storage mechanism for service information to avoid the potential bottleneck is also left for further study.

Further work will be focused on defining a widely-usable service description data model, and the more efficient service discovery and selection algorithms. A distributed storage mechanism for service information to avoid the potential bottleneck is also left for further study.

ACKNOWLEDGMENT

This work was supported by the EU ITEA-2 Project 2008005, "Do-it-Yourself Smart Experiences" (DiYSE), funded by DGCIS, France.

The authors are grateful to Mrs. Rebecca Copeland and Dr. Angel Cuevas Rumin from Institut Telecom, Telecom SudParis (France) and Prof. Xiaodi Huang from Charles Sturt University (Australia) for reviewing and providing many helpful comments that have improved the quality of this article.

REFERENCES

[1] G. Spanoudakis, G. Mahbub, and K. Zisman, "A Platform for Context Aware Runtime Web Service Discovery," *IEEE Int'l. Conf. Web Services*, 2007, 9–13 July 2007, pp. 233–40.

[2] T. Yu, Y. Zhang, and K. Lin, "Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints," *ACM Trans. Web*, May 2007.

[3] C. Rolland, R. S. Kaabi, and N. Kraiem, "ISOA: Intentional Services Oriented Architecture," *Advanced Info. Sys. Eng.*, vol. 4495, June 2007, pp. 158–72.

[4] P. Wang *et al.*, "A Reputation-Based Service Selection Scheme," *IEEE Int'l. Conf. E-Business Eng.*, Washington, DC, Oct. 2009, pp. 501–06.

[5] D. Moro, D. Lozano, and M. Macias, "WIMS 2.0: Enabling Telecom Networks Assets in Future Internet of Services," *LNCS*, vol. 5377, 2008, pp. 74–85.

[6] K. Lee, "Monetizing Networks with Service Exposure," Oracle White Paper, 2009.

[7] N. Blum, T. Magedanz, and F. Schreiner, "Service, Enablers and Architectures: Definition of a Connected Web 2.0 Telco Service Broker to Enable New Flexible Service Exposure Models," *Int'l. Conf. Intelligence in Next Generation Networks*, Bordeaux, France, Oct. 2008.

[8] Wholesale Applications Community, WAC Informational Whitepaper, Mar. 2010, http://www.wholesaleappcommunity.com/About-Wac/BACKGROUND percent20TO percent20WAC/whitepaper.pdf.

[9] S. Amit, "Semantic Scales Up: Beyond Search in Web 3.0," *IEEE Internet Computing*, vol.15, issue 6, Nov. 2011, pp. 3–6.

[10] D. Guinard *et al.*, "Interacting with the SOA-based Internet of Things: Discovery, Query, Selection, and On-Demand Provisioning of Web Services," *IEEE Trans. Services Computing*, vol. 3, no. 3, July 2010, pp. 223–35.

[11] DiYSE project, http://dyse.org:8080/pages/viewrecentblogposts.action?key=hometest.

BIOGRAPHIES

CUITING HUANG (cuiting.huang@it-sudparis.eu) is a Ph.D. student in Institute TELECOM SudParis and received a Master degree (Diplôme d'ingénieur) from SUPELEC in 2006. She worked from 2006 to 2008 in France Telecom R&D (Beijing). In 2008 she began her Ph.D. research in Institut TELECOM SudParis. Her current research interests are in intradomain and interdomain service composition, SOA, semantic service description, service exposure, and service life cycle optimization.

GYU MYOUNG LEE (gm.lee@it-sudparis.eu) holds a Ph.D. degree from the KAIST (former, ICU), Korea. He is currently with Institut Telecom as an adjunct associate professor specializing in network/service architecture and standards development. Since 2007 he has been working in ETRI, Korea. He worked at KAIST (former ICU) and NIST, United States, in 2007. His research interests include the Internet of Things and multimedia services. He has actively contributed to standardization in ITU-T as a Rapporteur and in IETF.

NOËL CRESPI (noel.crespi@it-sudparis.eu) holds a Master's from the Universities of Orsay and Kent, a diplome d'ingénieur from Telecom ParisTech, and a Ph.D. and an Habilitation from Paris VI University. He worked from 1993 in CLIP, Bouygues Telecom, France Telecom R&D in 1995, and Nortel Networks in 1999. He joined Institut Telecom in 2002 and is currently professor and program director, leading the Service Architecture Laboratory. He is appointed as coordinator for the standardization activities in ETSI and 3GPP.