Leveraging the Upcoming Internet of Services through an Open User-Service Front-end Framework

David Lizcano , Miguel Jiménez , Javier Soriano , José M. Cantera , Marcos Reyes , Juan J. Hierro , Francisco Garijo , and Nikolaos Tsouroulas

Universidad Politécnica de Madrid
28660 - Boadilla del Monte, Madrid, SPAIN

Telefónica I+D
28043 - Emilio Vargas 6, Madrid, SPAIN

Abstract. The Internet of the Future is expected to be composed of a mesh of interoperable Web Services accessed from all over the Web. This approach has not yet caught on since a global user-service interaction is still an open issue. This paper states our position with regard to the next generation front-end technology for the Internet of the Future. This approach will enable the massive deployment of services over the Internet in a user-centric fashion. This paper advocates the full development of front-end technologies to bring services closer to users, empowering them anytime and anywhere. It also outlines all the main gaps and technological challenges that have to be addressed. Finally, a model and an architecture are proposed for building these technologies into NESSI's Open Framework Reference Architecture, NEXOF-RA.

1 Introduction

One of Internet's future ambitions is the massive deployment of services and service-oriented systems across the whole Web. Actually, Web Services-based Service Oriented Architectures (SOAs) have gained momentum over the last few years And SOAs are expected to be the key to the user-service interaction take-off. However, these solutions are currently confined to company boundaries, and the desired global provision and consumption of user-centric compositional services from the envisioned Internet of Services is still at an early stage De-spite high expectations, the emergence of a mesh of interoperable Web Services that could foster the massive deployment of user-friendly services is continuously shattered by a series of well-known shortcomings , such as high technical complexity, implementation and maintenance costs, inflexibility and lack of widely accepted standards and open service frameworks. These issues are understandable considering that the underlying technologies were spawned by a machine-to-machine approach, not featuring a "face" for human users

This paper states our position, as Chairs and members of the NESSI User-Service Interaction Working Group (NESSI-USIWG), with regard to the next generation front-end technology for the Internet of the Future. This technology will enable the massive deployment of services over the Internet in a user-centric fashion In this paper we advocate the full development of front-end technologies to bring the services closer to users, empowering them anytime and anywhere. We outline all the main gaps and what technological challenges have to be addressed. We express in the paper what the NESSI USIWG is about to deliver as a research alignment in a forthcoming Position Paper, and we propose both a model and an architecture for building this technology. The outcomes and the vision presented in this deliverable will become part of NESSI's Open Framework Reference Architecture (NEXOF-RA): a coherent and consistent open service framework leveraging research in the area of service-based systems to consolidate and trigger innovation in service-oriented economies. The overall goal of NEXOF-RA, and thus the guiding principle of this paper, is to deliver a coherent set of globally applicable technologies. These technologies are intended to provide Europe with a digital services infrastructure to improve service flexibility, interoperability and quality. In addition, NEXOF-RA will try to establish strategies and policies to speed up the dynamics of the services ecosystem, as well as to foster the safety, security and wellbeing of citizens by means of new societal applications.

The remainder of this paper is organized as follows. First we present the shortcomings of the current SOA technologies with a view to the desired Internet of Services. We then go on to illustrate the guiding principles to achieve our aim. A reference model and its architecture are also proposed. Finally, we explain other related work, the main conclusions that can be drawn from the ideas in this paper, and what we consider to be the work that needs to be undertaken in the future.

2 Current Shortcomings on the Road Towards an Internet of Services

The massive deployment of user-centric services over the Internet demands services that must be accessible for all users (not only enterprise stakeholders). Therefore, services should flexibly and dynamically support common daily processes (both business processes carried out by companies and processes conducted by individuals or groups in their daily life) at any time[4]. Users will see the tools supporting their daily work replaced by composite applications based on Web Services, but traditional Web Services are not well enough tailored to users and their daily processes.Obviously, SOA, as it was originally conceived, represents an architecture focused fundamentally on a B2B context. It is weak for B2C problems, since it does not offer the best prospects for dealing with user-service interaction We can tackle its shortcomings from three different perspectives:

1. SOA's aim: Conventional SOAs merely aim to facilitate seamless machine-machine collaboration. SOA deployments are very abstract and invisible to users. Its customers of choice are medium-sized or larger corporations instead of normal end users along the long tail of Internet. Therefore, with SOA, normal Internet users with little IT expertise have not been able to easily retrieve and use services because services mostly reside within company boundaries and are only accessed for professional use in a corporate context.
2. SOA's technology: Apart from SOA's aims, this architecture relies on a set of complex standards that are not user friendly Because, technically speaking, SOA is extremely complex, there needs to be one or more expert players within the value chain to build and provide solutions for their customers. In contrast to this one-to-many value chain model of numerous SOA use cases (where one expert serves many clients), new value chains should begin to be mostly loosely coupled (many-to-many) networks of self-managed self-sufficient users who can offer and consume resources via the Web.
3. SOA's governance: Finally, SOAs are subject to clearly defined regulatory frameworks since they mostly exist in the corporate context. The design, provision, maintenance, and coupling of services must be compliant with legal frameworks. Therefore, they do not allow for the flexibility that the described new user-services interaction model appears to need.

3 Design Principles Enabling the Internet of Services

The evolution of Web 2.0 sites and applications is a testimony to the progress achieved to improve user relevance and service usability. However, current service front-ends are far from meeting end-user expectations Applications are still based on monolithic, inflexible, and unfriendly UIs. This is a serious obstacle for achieving the benefits of the Internet of Services. In order to build the next generation service front-end for this ecosystem of services we propose three guiding principles. These principles are further detailed in the following.

3.1 Enabling Users to Design and Share their Operating Workspace and Applications

Users should be able to design and implement their own interfaces in a flexible and friendly manner. New generation front-ends should provide new tools aiding end-user UI creation and self-adaptation, while supporting a dynamic computing infrastructure Community-based collaboration tools should also be supported to satisfy the demands for secure social interaction and improve knowledge and resource sharing. More to the point, the following aspects should be covered

1. Empower users to select resources of their interest, annotate, and configure their own personalized operating environment

2. Promote user pro-activeness for creating new resources, to improve versions of resources, and share further expertise about these resources, their use, and their interrelationships
3. Provide social facilities to share services, results, knowledge and resources with other users
4. Foster social interaction by providing visual mechanisms to manage user communities, user identity, privacy and security constraints, and the information to be shared, annotated, or send to specific groups and individuals

The new generation of user-service front-ends should be based on helping users with the flexible composition of applications. Application and service front-ends will no longer be conceived as monolithic blocks, but as a set of interoperable service front-end components –called gadgets–, which are available from a catalogue. A catalogue of components or gadgets will be available for creating application or service front-ends by using and combining these building blocks to construct new components. Users should be able to create their own applications by combining different catalogue components without any help from IT experts or a thorough knowledge of the underlying infrastructure

3.2 Businesses Need to Adapt to the New Reality

Today's competitiveness-driven business markets and the severe time-to-market restrictions on applications, specifically for enterprise IT systems, have increased the business needs to evolve applications to suit this new reality. They can be evolved through the following key guidelines:

1. Businesses need to embrace the Software as a Service (SaaS) model as an effective software-delivery mechanism [9]. This approach helps to reach a marketplace of services that can be composed to create unanticipated business solutions adapted to real needs.
2. Next generation Web Services ecosystems must respond to unforeseen business requirements that emerge or evolve spontaneously. This should be supported by new software development methodologies that ease the integration, adaptation and evolution of service-based applications.
3. Company boundaries must be eroded, evolving towards the Internet of Services vision. This approach can be split into two perspectives:
   (a) Next generation business systems should adopt a user-centric approach to take into account users. Users of these services are no longer just the company's employees; clients should also access the same business resources. This could even affect the entrepreneurial service-based workflows
   (b) Collaboration between companies, irrespective of their size, must be fostered, thus increasing productivity and accelerating innovation The creation of collaborative services by integrating components from disaggregated companies will afford new business opportunities and improve the global service provided to end-users.

3.3 Context-Adapted User-Service Interaction

The proliferation of multiple Internet-enabled devices allows end users to access the Web anytime and anywhere. As a result, there is a wide range of situations in which a user might need to access these services, and they must therefore be provided the right user interface for the right situation. These situations can be defined as the context in which access occurs.

Next generation service front-ends should take into account the following context aspects:

1. The delivery context, that is, a set of attributes that characterizes the environment in which a service is going to be delivered. Delivery context is a crucial aspect with regard to service front-ends, as it provides a clear indication of what the capabilities of the target device, web browser and network are. Such aspects play an important role in the end user experience. The information about these capabilities should be exploited by service front-ends to provide a harmonized experience adapted to the peculiarities of every delivery context.
2. Users and their circumstances: This aspect includes properties such as user identity, profile and roles, social network, tastes and personal preferences.
3. The surrounding environment, including the spatial location, speed, light conditions, temperature, level of noise, nearby objects/things...
4. The situation / time which has to do with variables such as date and time, weather, season, at home or at work, on vacation, on a business trip...

4 High Level Architecture

This section describes our overall proposed architecture for service front-ends (see Fig. 1). The architecture is consistent with the guiding principles and model. The central piece of the architecture is the NG-SOA Broker. The broker stores, indexes and references every available resource empowering users by giving them access to context-aware functionalities, such as resource discovery and recommendation, knowledge sharing through a social network, establishment of marketplace relationships between resource providers and other user roles. Three high-level modelling phases are proposed:

- Delivery Access Layer
- Access Layer Composition
- Access Layer Development.

Each phase takes into account one of the above aspects. It defines models and standards in order to take into account the associated elementary building blocks (workspaces, gadgets, application services, content delivery services...) and all the possible interactions among them. This overview should also define the different user roles we envision in the NG-user/services interaction. These roles should specify how users will exploit the services, how they will adapt services

to their requirements and needs and how (and who) will develop the parts enabling this interaction. To address this functionality several tools must be defined to create, compose and exploit the different models. These tools are the Gadget Development Platform, Mashup Platform and the Access Mechanism.

In this scenario authors or developers can build their own service front-end access point or gadget by combining a universe of accessible resources. Users are not supposed to be technological experts, they will be domain experts able to define and solve concrete problems by combining the provided services and tools. To do so, users should employ the facilities offered by the Gadget Development Platform.

The Application Composer role will build the Service Front-End Layer by composing several gadgets to build a fully interoperable application. The application will be able to enact end user processes in a fast and dynamic manner and deal with situational applications.

It will be the end user that exploits the results of the developed applications. These users will not notice any difference between the conventional applications and the applications built according to the above development method. They will access applications suited to their context in a location-, device-, technology-independent manner.

4.1 Delivery Access Layer

The Delivery Access Layer will be in charge of adapting and enriching the service front-end to meet the restrictions imposed by a given context. Therefore, this layer is responsible for providing a harmonized user experience in every context, hiding back-end services and developers from the underlying complexities. The functionalities implemented by this layer include, but are not limited to:

- Delivery Context Detection using some kind of evidence such as the HTTP headers (User Agent, Accept, UAProf ...)
- Simple Content and Application Adaptation depending on the Delivery Context: sending the correct markup, style sheets and script code, selecting the most appropriate resources taking into account restrictions such as the supported image formats, or the dimensions of the screen, and so on.
- Advanced (Semantic) Content and Application Adaptation. This functionality has to do with exploiting the context and application semantics (knowledge) to perform advanced adaptation processes such as item collection reordering, menu contextualization, content priorization or user interface enrichment.
- Automatic binding of the data coming from the back-end services. This functionality acts as the glue between the services back-end and the front-end
- Execute the application flow according to the author's intentions and taking into account access mechanism restrictions.

The Delivery Access Layer should be guided in order to operate in accordance to the developer's / author's intentions. Therefore, a declarative language capable

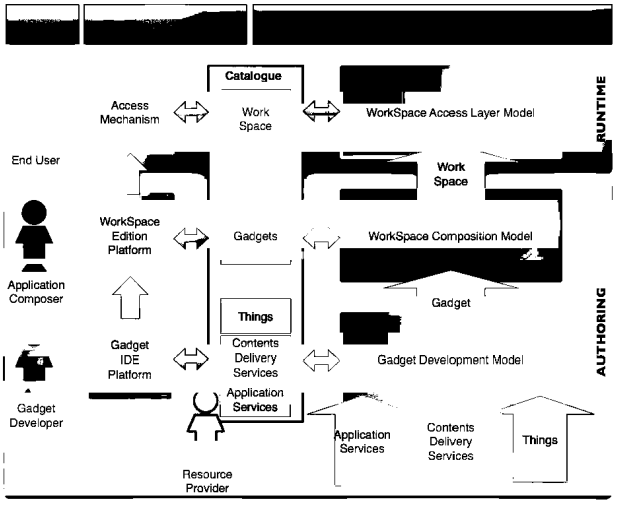

Fig. 1. high level architecture for the next generation of service front-ends

of expressing different "Adaptation Policies" should be available to drive this layer.

4.2 Workspace Layer Composition

Taking advantage of the results produced by Access Layer Development, Access Layer Composition allows end users to build their own instant applications. These applications can be composed by mashing up gadgets extracted from a NG-SOA Broker Gadget Repository. Gadget mashup should be construed generally not only as a spatial composition of gadgets inside a dynamic environment but also as the construction of fully integrated applications. To do so, several platform service modules are defined:

- Wiring Communication Module
- Session Module
- Knowledge Module
- Context Module
- User Preferences Module
- Identity Management Module

These modules are software packages oriented to support specific architectural objectives. They provide support for a set of standard gadget capabilities, so they can use platform functionality in a loosely coupled way. Each gadget should define a set of policies in order to negotiate the usage of available platform capabilities, because not every platform will necessarily have to include all the capabilities. For instance, support for the persistence capability can be optional for one gadget and mandatory for another. The usage of the capabilities should be defined declaratively and will be the main link between a gadget and a specific mashup platform (however it is necessary to define a standard handler mechanism linking capabilities to gadget functionality)(see Fig. 2).

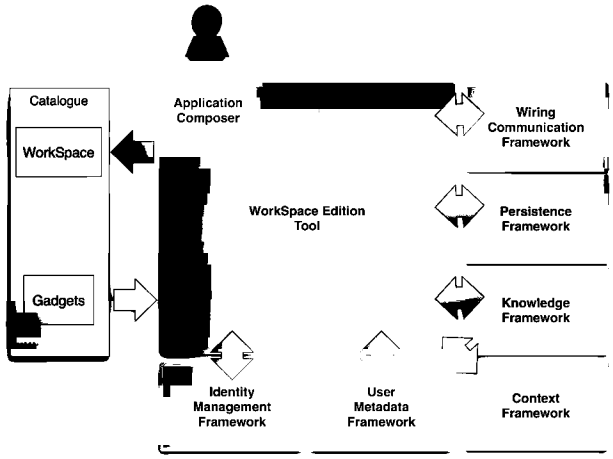

Fig. 2. composition layer of the reference architecture

4.3 Access Layer Development

The next generation SOA access layer consists of composite applications that are based on service front-end resources or gadgets. These gadgets are the building blocks of the interface and behaviour to be designed and developed, and constitute single access elements to the underlying service back-end resources. These gadgets are self-contained components focused on a single goal. This reduces

their complexity. Gadgets make up a substantial part of the interface and logic for invoking one or more use-case services, the user query forms and screens (see Fig. 3).

The gadget development process should be user-centric rather than program-centric, as the user is its final consumer. Thus the developers should bear users in mind when developing the front-end that they will see and interact with. This new approach calls for the use of fully visual development environments. Visual environments are more accessible and flexible for developers, who are domain experts but not necessarily technical users. These developers will visually compose a gadget from its building blocks (i.e. UI artifacts, back-end resources, operators), establish the connection to back-end Web Services, and integrate gadgets with the agnostic graphical user interface, thereby creating a service front-end resource.

The potential users of the next-generation SOA front-end should not be tied to any technology or access channel. This will promote the use of the wide range of services that will be available over the Internet. One of the objectives of the access layer is to serve as a multi-platform access mechanism to the SOA, suitable not only to be accessed by any platform or technology used at the client (desktop PC, mobile device, etc.) but also independently of the deployment infrastructure (i.e. the service front-end workspace platform). This technology independence should also be considered in the design and implementation of the gadgets that will make up the service front-end access layer. As a result, a device- and mashup platform-independent authoring language is needed to define the gadgets (i.e. their behaviour and interface) in a way that can be automatically converted to the target service front-end workspace platform and device technology. The gadgets will be defined in this language agnostically, with no focus on any specific access mechanism, and following a visual approach as mentioned above. The development environment is in charge of maintaining the mappings between the different layers of the authoring language.

Gadgets are not just mere UI parts and behaviour; they can form more sophisticated functionality around a piece of coherent business workflow. Gadget complexity can therefore vary depending on whether the UI comprises a single screen (i.e. simple gadget) or a screenflow inside the gadget itself (i.e. a complex gadget). This idea is especially relevant for highly constrained devices with limited display capabilities, such as mobile devices. Mashup-based compositional applications are seldom well designed for these devices because the user experience is based on disparate simultaneous views that are not affordable on small screens. The use of screenflows to model business logic in a single-piece view is better suited for these devices. The use of complex service front-end resources or complex gadgets in mobile environments is a good example of taking a proactive environment-dependent attitude, which, like technology independence, is a key objective of the SOA service front-end layer. These context-aware gadgets are then programmed to adapt their behaviour, interface and, maybe, part of their functionality to the specific context in which they have been deployed and are running.

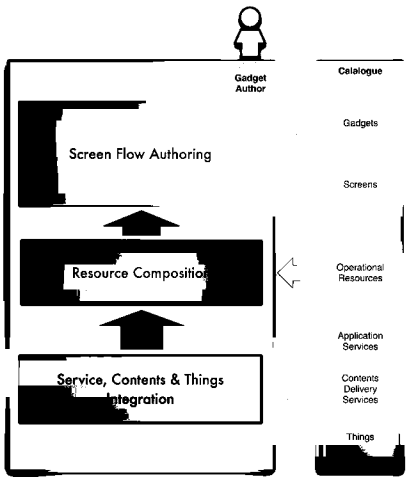

Fig. 3. access layer development

5 Related Work and Future Trends

The approach depicted in this paper is being developed as part of the NEXOF-RA initiative[1], a project partially funded under the European Commission's 7th Framework Programme. NEXOF-RA is contributing to NESSI, the networked European software and services initiative. This initiative asserts that Information and Communication Technology (ICT) will be an essential driving force for innovation and a core enabler of economic growth in the coming years. Enterprises in Europe (both private and public sector) are facing significant structural changes and will rely on software and services to support them in adapting effectively. The main focus of NESSI is that of service and its overall ambition is to deliver NEXOF, a coherent and consistent open service framework leveraging research in the area of service-based systems to consolidate and trigger innovation in service-oriented economies. NEXOF-RA is the specific effort to define a model and a reference for this open service framework.

In addition, there are several projects, led by some of the authors of this paper, related to the approach of this paper and that are very close to NESSI's objectives and ambitions:

- MyMobileWeb[2] is the open source reference implementation of the next generation content and application adaptation platform for the mobile web. MyMobileWeb enables the (time-to-market) creation of high quality mobile applications capable of adapting to multiple delivery contexts.
- EzWeb[3] pursues the development of an enriched enterprise mash-up platform and the development of key technologies to be employed in building the front-end layer of new generation SOA architecture.
- FAST[4] aims at providing an innovative visual programming environment that will facilitate the development of next-generation composite user interfaces. It is a novel approach to application composition and business process definition from a top-down user-centric perspective.

Future work will focus on evolving the proposed reference architecture, as an open source service framework that builds on all the key guiding principles described above and on the proposed vision of the Internet of Services. We expect this architecture to become a major hub for the publishing, brokerage, customization and finally the consumption of Web-based resources on a global, cross-organizational scale, revolutionizing the user-service interaction.

6 Conclusion

The appearance of user-centric approaches to next generation service front-ends, such as the one proposed in this paper, will be a major step forward, providing solutions to currently hard-to-solve problems in the traditional SOA paradigm. The emergence of such service architectures will solve key problems in three different scenarios. Large enterprises may capitalize on faster application development (for what are known as instant applications), a more agile system landscape and the empowerment of their employees to design their own applications that best satisfy their unique requirements, and to share this knowledge with other employees better than in traditional Web service architectures.

On the other hand, the proposed architecture enables SMEs to find, customize, combine, catalogue, share and finally use applications that exactly meet their individual demands by leveraging the SaaS model, viewed as Utopian from a traditional SOA perspective. Supported by the new Internet of Services approach, they can select and combine resources hosted by third parties rather than buying a pre-determined, inflexible and potentially heavyweight solution or deal with complex B2B services.

Finally, individuals benefit from a sharp increase in the potential for personalization and participation. This approach will provide end-users with intuitive, unsophisticated IT ways to discover, remix and use those Web-based services that they consider interesting and useful. It will also allow them to participate,

swap information with other users and service providers and to actively contribute in a way that encourages extensive use of the resources offered. This speeds up the service innovation pace. Focusing on the "long tail" advanced by Chris Anderson rather than a limited number of sophisticated experts, a user-centric SOA will involve the bulk of private users or small businesses and allow for "customer self-service".

References

Alonso, G., Casati, F., Cuno, H., Machiraju, V.: Web Services Concepts, Architectures and Applications. Data-Centric Systems and Applications. Springer (2004)

McAfee, A.P.: Enterprise 2.0: The dawn of emergent collaboration. MIT Sloan Management Review **47**(3) (2006) 21–28

Hierro, J.J., Janner, T., Lizcano, D., Reyes, M., Schroth, C., Soriano, J.: Enhancing user-service interaction through a global user-centric approach to soa. In: Proceedings of The Fourth International Conference on Networking and Services (ICNS 2008), IEEE Computer Society Press (March 2008) 194–203

Schroth, C., Christ, O.: Brave new web: Emerging design principles and technologies as enablers of a global soa. In: Proceedings of the IEEE International Conference on Services Computing, 2007. SCC 2007. (2007) 597–604

Soriano, J., Lizcano, D., Cañas, M.n., Reyes, M., Hierro, J.J.: Fostering innovation in a mashup-oriented enterprise 2.0 collaboration environment. System and Information Science Notes **1**(1) (July 2007) 62–69 SIWN International Conference on Adaptive Business Systems (ICABS2007). Chengdu, China.

Schroth, C., Janner, T.: Web 2.0 and soa: Converging concepts enabling the internet of services. IT Professional **9**(3) (2007) 36–41

Smith, R.: Enterprise mashups: an industry case study. In: Keynote at New York PHP Conference and Expo. (June 2006)

Lizcano, D., Soriano, J., Fernández, R., López, J.A., Reyes, M.: Tackling composite application developments from an enterprise 2.0 mash-up perspective. In: Proceedings of the 14th International Conference on Concurrent Enterprising (ICE 2008). (June 2008)

Gartner: Hype cycle for software as a service. Gartner research, Gartner Inc. (August 2006)

Anderson, C.: The Long Tail: Why the Future of Business Is Selling Less of More. Hyperion (July 2006)

Dey, A.K.: Understanding and using context. Personal and Ubiquitous Computing Journal **5**(1) (February 2001) 4–7

Fielding, R.T.: Architectural styles and the design of network-based software architectures. PhD thesis, University of California, Irvine (2000)