A Semantic Service-based micro-learning framework (Semantics for Microlearning*)

Sinuhe Arroyo
University of Alcala de Henares (Spain)

Abstract: Microlearning deals with relatively short small learning units and short-term focused activities. The applicability and impact of this novel learning technology can be greatly increased by incorporating rich semantics to the description of the subject learning units. In this paper Microlearning is put into the contest of the Semantic Web and Semantic Web Services with the aim of detailing a solid framework and reference SOA that uses the communication facilities of an ESB. The ultimate goal is to evaluate the applicability of the approach to the necessities and requirements posed by Microlearning.

1. Introduction

The advancements in computer science in general, and in knowledge management in particular have provoked a dramatic change on how knowledge is distributed and acquired. On top of this, our current society requires individuals to continue learning in case they want to have access to interesting jobs and follow a brilliant career. However, the time available to acquire new knowledge is not always as much as would be required or at least not at once. Thus, the concepts and ideas around microlearning, microcontent and microknowledge are developed with the aim of bridging this gap.
Microlearning tries to allow people to learn better, more effectively and in an easier, more enjoyable manner. This goal is achieved by means of breaking information into smaller units, administered at smaller steps than in traditional learning, using inter-spaces between different activities (Bruck, 2005).

---

* Contributions to the embedded Expert Workshop "Semantic Learning Objects and Semantic Learning Organizations: Semantics for micro-learning? "at the Pre-Conference Workshop Day of the European Academy for Microlearning. Chairs: Martin Lindner, Research Studios Austria (Austria) Miguel-Angel Sicilia-Urbán, University of Alcalá de Henares (Spain) Ambjörn Naeve, Uppsala Learning Lab (Sweden)

A number of points of view and understandings have posed that try to give definition that set the pillars of what microlearning, microcontent and microknowledge mean. However, so far there seems not to be a general agreement or common understanding about the precise meaning of these concepts. What is common to all of them is that they seek, in a variety of ways, to integrate learning into everyday life (Hug, 2005).

For the time being and as far as this paper concerns the following definitions will be adopted.

> *"Microlearning deals with relatively short small learning units and short-term focused activities"* (Hug, 2005a).

> *"Microcontent refers to small, granular pieces of content, to simplex semantic units or to small-sized semiotic entities"* (Hug, 2005a).

Thus, microlearning can be understood as learning from microcontent – from *„small pieces, loosely joined"* (Weinberger, 2002).

The requirements posed by the microlearning technology require novel approaches. Two main characteristics need to be taken under consideration when designing systems and applications:

- **Rich metadata descriptions.** Microcontent and learning objects in general need to make use of rich metadata descriptions based on Semantic Web technology so they can be easily discovered and composed.
- **Storing and discovery.** Current approaches to storing learning units are based on repositories (Azuma, 2005) where the microcontent and the metadata describing them, if any, are stored. However, the search of microcontent and learning objects in general, requires more dynamic techniques where relevant content can be published directly on the Web and located by means capable search engines (Arroyo, 2005).
- **Ubiquitous and Service based.** The dynamic nature of microcontent poses special requirements on the infrastructure required to deliver contents (Fischer, 2005). It this direction (Web) Services provide the perfect paradigm to encapsulate rich learning unit descriptions and locate them using traditional (Web) Service discovery techniques.

By this means a new approach to design microlearning applications independently of the technology and application domain can be envisioned. In the following the main ideas and principles behind this approach are presented, together with a use case where the whole setting and technology are being applied. The aim is to demonstrate the viability of the whole approach and set the architectural basis for new applications of the microlearning technology.

The paper is organized as follows. Section 2 provides a detailed overview of the architectural context and motivation. It presents and briefly depicts the core ideas behind the Service paradigm, its application to a SOA and the implications of using Web services. Furthermore, the communication requirements of a SOA are put into the context of a ESB and the whole setting complemented by the addition of the Semantic Web technologies. Section 3 introduces the overall architectural model taking care of depicting the main building blocks and their role in the architecture. Section 4, presents LUISA, an EU IST project where the whole approach is applied and tested. Finally, Section 5 resumes the work and provides an overview of future steps to be taken.

2. Architectural context and motivation

The aim of this section is to put into context the LUISA architecture. It provides the background knowledge and perspective that justify the directions and decisions taken. In the following a brief overview of the main architectural concepts to be used, namely, Services, Web Services, Semantic Web Services, Service Oriented Architecture (SOA), Semantic Service Oriented Architecture (SSOA), Enterprise Service Bus (ESB), Semantic Enterprise Service Bus (SESB) and Learning Objects (LO) are presented.

2.1 Services

Roughly speaking a service *is a piece of software that implements some well-defined functionality that can be consumed by clients (e.g. other services), regardless the application or business model. Services communicate with each other by means of message exchanges.*

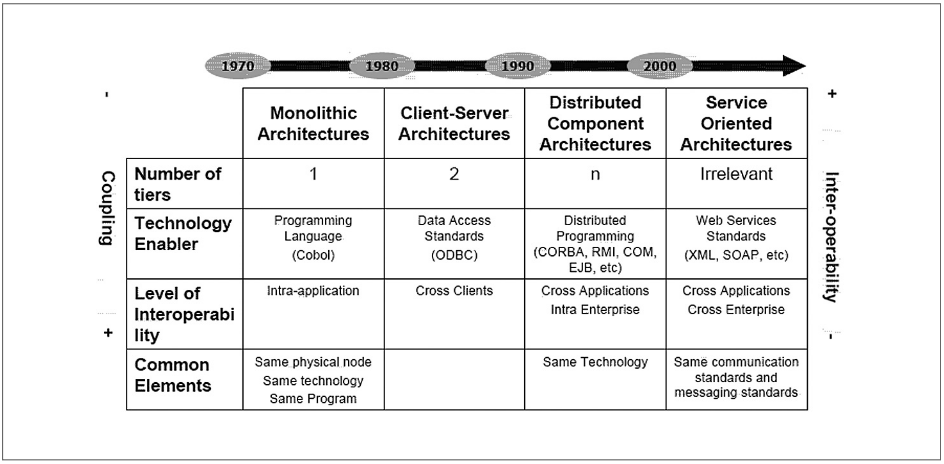
The main advantages provided by the use of services revolve around its interoperability, loose coupling, isolation, compos ability and machine processability.

- **Interoperability.** Interoperability aims at providing seamless connections among software applications. Services allow programs written in different languages, deployed over different platforms and using different protocols to communicate with each other thus favoring interoperability.
- **Loose coupling.** Decoupling or loose coupling refers to the degree of mutual dependency among services. Services expose rich and well-defined message interfaces which allow them to communicate with other services reducing mutual dependencies.
- **Isolation.** It deals with the ability to modify services or their details not impacting other services that might interact with them. Service consumers are abstracted from details of service implementation and location.
- **Compos ability.** Service composition approaches the problem of the creation and provision of complex value-added services out of simpler ones with the aim of achieving new functionality. Services can be easily composed with each other in order to achieve more complex operations and sophisticated added-value services (Mahmoud, 2005).
- **Machine processability.** Deals with the ability of computers to process service descriptions. Thanks to the use of agreed specifications computers can lively process services descriptions favoring their interoperability, loose coupling, isolation and compos ability among others.

2.2 Service Oriented Architecture (SOA)

SOA is an architectural style for building software applications that use services available in a network such as the web as main building blocks (Khushraj et al., 2004). The ultimate goal of SOA is to promote loose coupling while increases interoperability among software components so that they can be reused.

Figure 1 shows the evolution in software architectures. It starts back in 1970 with the monolithic architectures ending nowadays when SOA represent the ultimate architectural paradigm. The evolution path clearly shows an aim for decreasing decoupling while increasing interoperability with stronger emphasis on standardized approaches.



	Monolithic Architectures	Client-Server Architectures	Distributed Component Architectures	Service Oriented Architectures
Number of tiers	1	2	n	Irrelevant
Technology Enabler	Programming Language (Cobol)	Data Access Standards (ODBC)	Distributed Programming (CORBA, RMI, COM, EJB, etc)	Web Services Standards (XML, SOAP, etc)
Level of Interoperabi lity	Intra-application	Cross Clients	Cross Applications Intra Enterprise	Cross Applications Cross Enterprise
Common Elements	Same physical node Same technology Same Program		Same Technology	Same communication standards and messaging standards

Figure 1. Software architectures evolution[1]

The ultimate aim of a SOA is that of providing ubiquitous and autonomous sets of services that dynamically interoperate to achieve common goals. In this direction a SOA provides a vendor-neutral communications framework with the potential to implement highly interoperable service descriptions and message structures.

2.3 Web Services (WS)

According to the W3C a Web Service (Alonso et al., 2003) is *a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web Service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.*

In simple terms, Web Services are software machinery accessible via the Web. A Web Service can provide any type of functionality, ranging from mere information providers (such as stock quotes, weather forecasts, or news aggregation) to more elaborate ones that may have some impact in the real world (such as book sellers, plane ticket sellers, or e-banking), basically any functionality offered by the current Web can be envisioned as a Web Service.

1 Keller, S. and Aegert, C.: „Service Oriented Architectures"

Web Services are the most used way to realize the service and SOA paradigm. In contrast to Services, Web Service due tackle implementation details being linked to concrete specifications and protocols (WSDL, SOAP, XML and HTTP). As far as SOA concerns, it is important to notice that SOA and Web Services are two different things, but Web Services are currently the preferred standards-based way to realize SOA (Alonso et al., 2003). Other alternatives such as REST (Costello) are available for describing and implementing services.

2.4 Semantic Web Services (SWS)

The main drawback of traditional Web Services is their lack proper support for machine processable semantics. This lack makes necessary human intervention to actually discover, combine, and execute Services. The goal is to minimize any human intervention, so the integration of business logics can be done in a task-driven way and with the least support from the user side.

The combination of Semantic Web technology [(Berners-Lee, 1999) (Berners-Lee et al. 2001)], namely ontologies, and Web Services, has been termed Semantic Web Services. Semantic Web Services are defined as "Self-contained, self-describing, semantically marked-up software resources that can be published, discovered, composed and executed across the Web in a task driven semi-automatic way" (Arroyo et al., 2004).

The Semantic Web in general and Ontologies in particular offer the means to describe the capabilities of Services, the protocol used and the data interchanged while communicating, and the business models in a shared vocabulary that can be understood and/or aligned as required. Such a shared functional description is the key element towards the vision of Semantic Web Services and in particular, sets the foundational basis to realize the semantic Web Service usage process. Such process is made of seven different phases, namely: discovery, negotiation, composition, mediation, execution, monitoring, and compensation.

In a nutshell, Semantic Web Services will allow the development and execution of a higher level of Services that will solve increasingly complex tasks by making available new composed Services. The goal is to minimize any human intervention, so the usage process of Semantic Web Services can be done in a semi-automatic way.

2.5 Semantic Service Oriented Architecture (SSOA)

SOAs present insufficient support when the services that compose them use heterogeneous terminologies for representing the business model they serve. By combining the architectural principles of an SOA, with the machine understandability and processability of the Semantic Web, this limitation can be easily overcome, thus giving birth to a new architectural paradigm termed Semantic Service Oriented Architecture (SSOA). An SSOA represent the next natural step in the evolution of SOA where the main building blocks are Semantic Web Services. Every SSOA encloses the same foundational principles of a traditional SOA, plus the incorporation of semantic support. By these means resources using heterogeneous terminologies and understandings can be shared among different systems and platforms enabling the agile discovery, negotiation, composition and interoperation of services in a task-driven way.
In a nutshell, a SSOA is a robust and complete architectural style where Semantic Web Services are the core building block.

2.6 Enterprise Service Bus (ESB)

Enterprise Service Bus (ESB) represents a new type of application integration middleware that provides foundational services for more complex service-oriented architectures via an event-driven and XML-based messaging engine, the bus.
ESB provides support for data transformation, intelligent routing and communication mediation, resource connection via adapters or specific communication protocols, process coordination or orchestration, management of security and quality of services aspects while guaranteeing message delivery. The result is a more flexible approach to application integration that solves in a very simple way the synchronization requirements across two or more applications.
ESB facilitates an abstraction layer which allows exploiting the value of messaging while keeping a simple architectural model. It acts as a lightweight, ubiquitous integration backbone through which software services and application components flow (Farges).

In a nutshell, an ESB represents the backbone that transports and routes messages enabling the standards-based integration in a Service-Oriented Architecture. Figure 2 shows a schematic of an ESB that nicely integrates multiplatform, enterprise and custom applications together with heterogeneous data sources into a common channel that provides intelligent routing and advanced communication facilities among others.

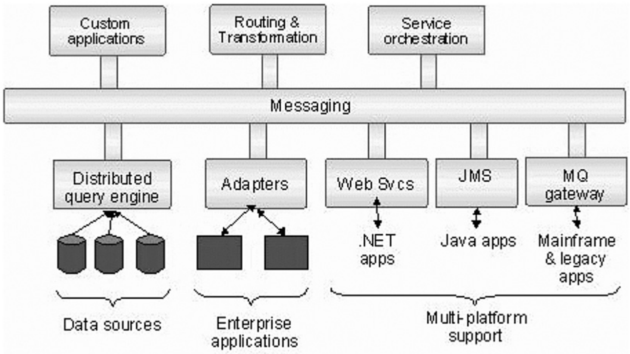


Figure 2. ESB schematic

The main advantage of using ESBs as underlying communication infrastructure is that it abstracts Services from a strong coupling in terms of reference and time (Fensel, 2004) as posed by current communication paradigm based on message exchanges.

2.7 Semantic Enterprise Service Bus (SESB)

Same as in the case of SSOA, Semantic Enterprise Service Bus (SEBS) combine the benefits of the Semantic Web with the event driven, intelligent routing and communication mediation facilities available in plain ESBs. SEBS represents the perfect communication infrastructure for SSOAs as it includes native semantic support. This supports translates in the seamless integration of semantically heterogeneous applications (e.g. services) into a shared communication mechanism that provides all the means for the reliable, asynchronous and flexible delivery of information.

A number of approaches exist (sTuples (Martín-Recuerda et al., 2006), Triple Space Computing, Semantic Web Spaces [(Martín-Recuerda et al., 2006), (Tolksdorf et al., 2004) (Tolksdorf et al. 2005) (Tolksdorf et al., 2005a)] and CSpaces [(Martín-Recuerda et al., 2006) that tackle the problem of realizing complete SEBS. However, none of them provides a solid and complete implementation of the concepts and ideas behind SEBS. Thus, one of the main tasks in this WP, besides defining a complete LUISA reference architecture, will be explore the status and applicability of current initiatives to LUISA. In case none of them suit the project architectural requirements, LUISA will considering realizing a minimal SEBS that serves and fulfils the communication requirements of a fully-fledged SSOA.

3. A layered architectural model

The reference architecture follows a layered model where layers are piled on top of each other building a stack (see Figure 3). Each layer comprises a number of SWS that realize the layer functionality providing support to the ones building of top. Additionally, each one of the layers is also defined as a SWS that communicates with others by means of message exchanges. By these means a fully-fledged SSOA is defined.
In the following the layers that comprise the architectural specification are briefly depicted:

- **Persistence layer:** Encloses a number of services that detail a light weight framework for simple storage management. In this layer the basic LO together with the semantically enhanced metadata that describes them are kept.
- **Semantic Layer:** Defines the main building blocks that implement the semantic support of the framework. It builds on top of the storage layer comprising a reasoning engine that supports mediation among heterogeneous SWS and the LO they interface.
- **SWS layer:** Depicts a number of well defined services that put in place the functionality that enables the publication, discovery, negotiation, choreography, orchestration and execution of SWS and the LOs they interface. This layer builds on top of the basic semantic layer.
- **Communication layer:** Comprises an ESB service that provides a sharable mechanism to manage messages guarantying its delivery and mediation. It facilitates the functionality that allows the architecture to communicate with other systems and internally between the SWS that build it. Every other layer in the architecture makes use of this one for service-to-service communication.

In addition to these main layers two more levels of abstraction have been added to the framework that helps realizing a fully fledged framework and reference architecture.

2 Taken from: Ibarra, F: "The Enterprise Service Bus: Building Enterprise SOA", http://dev2dev.bea.com/pub/a/2004/12/soa_ibarra.html (accessed Oct, 2006)

- **Ontologies:** In order to enriching LO-metadata with machine processable semantics the WSMO ontology has been augmented accommodating the specific requirements of LO. This architectural building block does not comprise a service by itself but rather a necessary artefact for the richer description of LO.
- **Tooling Support:** The enhancement of syntactic metadata describing LO with semantic annotations requires proper tooling support that eases the task. Thus, a LO annotation tool is made available as part of the architecture that fulfils this purpose.

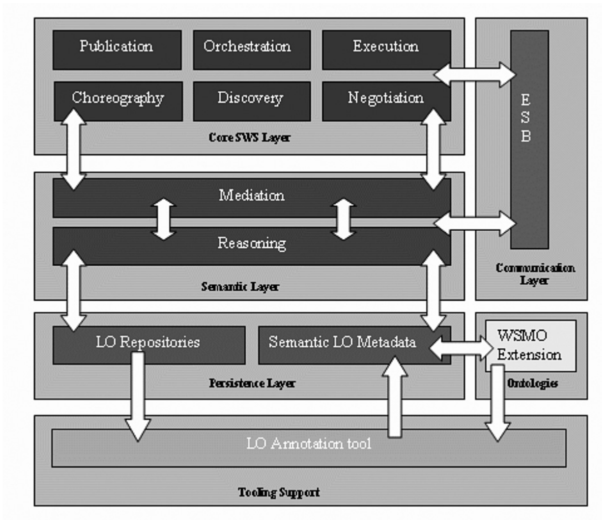


Figure 3. Layered architecture

4. LUISA: Learning content management system Using Innovative Semantic Web Services Architecture

The concept, standards and technology of learning object has resulted in increasing levels of transportability of learning contents across platforms. Nonetheless, the promise of reuse of learning objects depends on the provision of the Web infrastructure that enables semantic interoperability for the discovery, selection, composition and negotiation of learning objects. Such an infrastructure requires rich semantics in learning object metadata and consistent Learning Management Systems (LMS) and Learning Object Repositories (LOR). Semantic Web Services (SWS) are a candidate architecture for such needs that provides the required richness in description and semantic interoperability facilities. Ontologies can be used for the description of learning needs and learning object metadata, and mediation facilities can be used both for the distribution of services and also as a mechanism for flexible semantic interoperability.

The main aim of LUISA is that of developing the reference architecture and providing relevant evaluation case studies for a SWS-based framework that enables the construction of advanced learning technology tools and systems that re-use learning objects and learning designs. SWS components integrated with existing LMS technology will open new possibilities in the design of learning experiences or the selection of existing ones, once they are registered in a semantic LOR.

The architecture presented in the previous section sets the conceptual basis for the work to be conducted in LUISA and which meets the afore mentioned requirements and characteristics.

5. Conclusions and future work

This paper presented the description of reference architecture for the development of microlearning applications and their deployment on different settings regardless of the application domain. The paper tries to adopt the point of view of the instructor as far as it takes under consideration the discovery and composition of microcontents with the aim of building a whole micro course, but also of the learner who will consume the contents in a ubiquitous environment.

As future steps, the ideas presented in this work will be implemented and tested as part of the EU-funded project LUISA, which tries to apply and combine Semantic Web and Semantic Web Services technologies to eLearning in general.

6. Acknowledgements

The work is funded by the European Commission under the project LUISA (IST – FP6-027149), and by the Comunidad Autónoma de Madrid under the project PAWSEL funded by the CAM-PRICYT program.

7. References

Alonso, G., Casati, F., Kuno, H., Machiraju, V. (2003). Web Services. Concepts, Architectures and Applications. Springer-Verlag.

Arroyo, S., Lara R., Gomez, J. M., Berka, D., Ding, Y. and Fensel, D. (2004). Semantic Aspects of Web Services. In Practical Handbook of Internet Computing. Munindar P. Singh, editor. Chapman Hall and CRC Press, Baton Rouge. 2004.

Arroyo, S. (2005). Searching for Semantic Web Services – A Google-based Approach Journal of Digital Information Management. Vol. 3, No. 3.

Azuma, J. (2005). Need for Web-based Metadata Repository on Online Resources Usable for Language Teaching (On Microlearning and Microknowledge in a Microcontent-based Web)". Microlearning: Emerging Concepts, Practices and Technologies after eLearning. Proceedings of Microlearning 2005. Learning & Working in New Media.

Berners-Lee, T. (1999). Weaving the Web. San Francisco: Harper.

Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web. Scientific American, 284(5):34-43, 2001.

Bruck P. A. (2005). Microlearning as strategic research field: An invitation to collobarate" (Introductory Note). Microlearning: Emerging Concepts, Practices and Technologies after eLearning. Proceedings of Microlearning 2005. Learning & Working in New Media.

Costello, R. L. Building Web Services the REST Way. http://www.xfront.com/REST-Web-Services.html.

Farges, N. Enterprise Service Bus (ESB): Lasting concept or latest buzzword? http://searchwebservices.techtarget.com/tip/1,289483,sid26_gci913058,00.html

Fensel, D. (2004), Triple-based Computing. Digital Enterprise Research Institute (DERI) Technical Report DERI-TR-2004-05-31.

Fischer, R. (2005) Microlearning with Mobile Weblogs: On Microlearning and Microknowledge in a Microcontent-based Web. Microlearning: Emerging Concepts, Practices and Technologies after eLearning. Proceedings of Microlearning 2005. Learning & Working in New Media.

Hug, T. (2005). Microlearning and Narration. Fourth Media in Transition Conference, MIT, Cambridge (MA), USA.

Hug, T. (2005a). Microlearning: A New Pedagogical Challenge (Introductory Note). Microlearning: Emerging Concepts, Practices and Technologies after eLearning.

Proceedings of Microlearning 2005. Learning & Working in New Media.

Khushraj, D., Lassila, O., and Finin T., (2004), sTuples: Semantic Tuple Spaces", in Proceedings of the First Annual International Conference on Mobile and Ubiquitous Systems (MobiQuitous'04).

Mahmoud, Q. H. (2005). Service-Oriented Architecture (SOA) and Web Services: The Road to Enterprise Application Integration (EAI), http://java.sun.com/developer/technicalArticles/WebServices/soa/index.html, April 2005.

Martín-Recuerda, F., Nixon, L.J.B., E. and Scicluna, J., (2006), D 2.4.8.1: Technical and ontological infrastructure for Triple Space Computing. Knowledge Web, http://knowledgeweb.semanticweb.org/.

Tolksdorf, R., Nixon, L. J. B. , Liebsch, F., Nguyen, D. M. and Paslaru-Bontas, E., (2004), Semantic Web Spaces. FU Berlin Technical Report B-04-11, 2004.

Tolksdorf, R., Nixon, L. J. B., Paslaru-Bontas, E., Nguyen, D. M. and Liebsch, F. (2005). Enabling real world Semantic Web applications through a coordination middleware. 2nd European Semantic Web Conference ESWC2005, Heraklion, Crete, May 2005

Tolksdorf, R., Paslaru-Bontas, E., and Nixon, L. J. B.(2005a). Towards a tuplespace-based middleware for the Semantic Web. IEEE/WIC/ACM International Conference on Web Intelligence WI2005, Compiegne University of Technology, France, September 2005.

Weinberger, D. (2002), Small Pieces Loosely Joined, Perseus Books.