Adding Semantics to Software-as-a-Service and Cloud Computing

FRANCISCO GARCIA-SANCHEZ
ENEKO FERNANDEZ-BREIS
RAFAEL VALENCIA-GARCIA
Universidad de Murcia
Facultad de Informatica
30100 Espinardo (Murcia)
SPAIN
frgarcia@um.es
enekofb@um.es
valencia@um.es

ENRIQUE JIMENEZ
JUAN M. GOMEZ
JAVIER TORRES-NIÑO
Universidad Carlos III de Madrid
Computer Science Department
28911 Leganes (Madrid)
SPAIN
enrique.jimenez@uc3m.es
juanmiguel.gomez@uc3m.es
javier.torres@uc3m.es

DANIEL MARTINEZ-MAQUEDA
Indra Software Labs
Alcanto 11, 28911 Madrid
SPAIN
dmmaqueda@indra.es

Abstract: The Web is evolving from a mere repository of information to a new platform for business transactions and information interchange. Large organizations are increasingly exposing their Business Processes through Web Services Technology for the large-scale development of software, as well as for the sharing of their services within and outside the organization. New paradigms for Software and Services Engineering such as the Software-as-a-Service (SaaS) and the cloud computing model promise to create new levels of efficiency through large-scale sharing of functionality and computing resources. However, there are few academic works in this area and many gaps are still open. In this paper, we present a Business Process based on Semantics platform where services are executed from a SaaS perspective. The SITIO platform allows external developers to create add-on applications that integrate into the main SITIO application and are hosted on Cloud Computing infrastructure.

Key–Words: Cloud Computing, Software-as-a-Service, Business Process Management, Semantic Technology

1 Introduction

Recently, the economy has taken a downturn, which has forced many companies to reduce their costs in IT. According to experts, one interesting approach could be to outsource parts of their business through third parties. This will allow companies to free resources invested in outsourced business processes and to focus their investments on those processes that are unique to the organization. The future Internet will be based on services and this new trend will have significant impact on domains such as e-Science, education and e-Commerce. Recently, SaaS (Software-as-a-Service) has emerged as a model of software deployment suitable for implementing this new service-oriented strategy. SaaS is a software distribution model in which additionally to providing software, the supplier offers additional services such as maintenance and support. The advantage of SaaS is that the software is distributed and hosted on the Internet, eliminating the requirement for the users to install the software on their own computing infrastructure, as well as any related data.

Tightly related to the SaaS model is cloud computing: an IT infrastructure provisioning model in which applications and services from many organizations are hosted in a single large-scale facility. Cloud computing and SaaS open the doors for large economies-of-scale, but are faced with a number of challenges. Foremost among these challenges are: (i) the lack of proven models for determining under which conditions it is cost-effective for IT user organizations to migrate towards these models, taking into account legal, business and technical factors; and (ii) the lack of proven methods (e.g. architectural guidance) for facilitating such migration. Specific challenges include how to represent SaaS and cloud computing capabilities and requirements, and how to enable brokers to match between such capabilities and requirements.

Accordingly, one of the key aims of our work was to develop a framework, including models and methods, to facilitate cost-effective access to hosted business services and cloud computing services, from the perspective of a broker. The SITIO platform can also deal with the problem of the increasing number of Business Processes available as Services on the Web, which gives rise to a new challenging problem: the integration of heterogeneous applications. Semantic Technologies deal with adding machine-understandable and machine-processable metadata to Web resources through its key-enabling technology:

ontologies. Ontologies are a formal, explicit and shared specification of a conceptualization. The breakthrough of adding semantics to Business Processes for SaaS leads to an architecture to integrate various Information Systems from the perspective of emerging scenarios which benefit from the use of such technologies. The aim of SITIO is gathering these emerging concepts (SaaS, Semantic Technologies, Business Process Modeling, Cloud Computing) to foster dramatic evolution of a new platform oriented towards interoperability and cost reduction which can impact significantly in industry. Thus, SITIO can be defined as a platform for reliable, privacy-aware, secure and cost-efficient Semantics-based Software-as-a-Service Creation, Integration and Management.

The rest of the paper is organized as follows. Section 2 contains an overview of the state-of-the-art of the technologies involved in this project. The components that take part in the platform and its overall architecture are described in Section 3. Finally, conclusions and future work are put forward in Section 4.

2 Technological Background

Four main technologies are involved in this project, namely, Software-as-a-Service, ontologies, Business Process Modeling and Cloud Computing. Combining the state of the art of these technologies, as it is the goal of our project, one can construct a platform oriented towards interoperability and cost reduction which can impact significantly in industry. A brief review of these technologies is provided next.

2.1 Software-as-a-Service

Sotware as a Service (SaaS) [5] is a software distribution model in which additionally to providing software, the supplier offers additional services such as maintenance, help and support. The advantage generated by the use of SaaS is that the software is distributed and hosted on the Internet, eliminating the requirement for the user to install the software on his personal computer, as well as any related data. The software and all of the additional data required are stored on an external server, provided by the company that offers these services. In particular, the main benefit of such a model is that by using this model, the provider can reach any segment of the market [27], being a large corporation, a standard user or a small organization.

This framework provides numerous additional advantages, such as the possibility to access, distribute, and manage the software across the Web, which empowers the user with greatly increased access to the components. A further interesting issue concerning this new software delivery paradigm is that the management and use of applications takes place in centralized servers, instead of being hosted in the client organizations' IT infrastructure. Clients will be able to access and configure the services provided by these applications by means of the Web [9].

Furthermore, in relation to data storage, it should be mentioned that user data are stored on an external server, thereby ensuring that clients do not require large storage capacities in the case that they work with large amounts of information, such as metadata. This also guarantees that software users always have secured copies of their data, entrusting the supplier company with responsibility for data storage, without having to be concerned about the data.

The distribution model is based on the premise that the supplier company offers the maintenance and support service. This entails the assumption that all the information, processing and business logic is stored in the same location, a fact from which ICT companies greatly benefit, given that the lack of such a concept (which is, in fact, a rather frequent occurrence) usually leads to the dispersion and lack of integration and communication between distinct business processes.

The paradigm shift brought by the SaaS delivery model also leads to some further implications [16]:

- Decentralized model for the use of software applications.
- Scalability increases to the point of being unlimited.
- The importance of the services stay in the services, leaving in second step technology, applications and other technical aspects.
- The clients become to be virtual entities.
- Forms a public infrastructure where all companies can take out a subscription to the services provided.
- Platforms are built for N clients.
- The client obtains the following benefits:
    1. Risk descent.
    2. Decrease in costs and inicial investment.
    3. High scalability.
    4. Is focused in the business.
    5. Increase of the security due to a professional maintenance of the equipment.
    6. High adaptability and quick response to changes.

7. Allows an efficient use of company resources and time.

8. Unique vision of company information, avoiding misunderstandings by direct communication between human and computer.

9. Higher connectivity with every layer of the company.

2.2 The Semantic Web: Ontologies

The World Wide Web (WWW) was invented in 1989 by Tim Berners-Lee and it changed the way people gather and access information. Nowadays, the Web is a huge data repository in continuous growth in such a way that a major bottleneck has appeared when trying to exploit the represented information, namely, how to find the specific piece of information we are looking for. The Semantic Web [3, 32] was conceived with the purpose of solving this issue. It aims at adding semantics to the data published on the Web (i.e., establish the meaning of the data), so that machines are able to process these data in a similar way a human can do. Therefore, it can be said that the Semantic Web is characterized by the association of machine-accessible formal semantics with the traditional Web content.

The knowledge representation technology used in the Semantic Web is the ontology, which formalizes such meaning and facilitates the search for contents and information [15], as well as improves crawling [36]. A number of different ontology definitions can be found currently in literature. In this work we have adopted the following one: "an ontology is a formal and explicit specification of a shared conceptualization" [33]. In this context, formal refers to the need of machine-understandable ontologies. This definition emphasizes the need for agreement in carrying out a shared conceptualization. Besides, the ontology language selected in this work was OWL (Web Ontology Language) [22], which is the the ontology language recommended by the World Wide Web Consortium (W3C). Ontologies provide a common vocabulary of an area and define -with different levels of formality- the meaning of the terms and the relations between them. Knowledge in ontologies is mainly formalised using five kinds of components: classes, relations, functions, axioms and instances [12]. Classes in the ontology are usually organised into taxonomies. Sometimes, the definition of ontologies has been diluted, in the sense that taxonomies are considered to be full ontologies [33].

2.3 Business Process Management

The idea of a service-based architecture comes from the eighties. However, this practice did not come into widespread use until relatively recently. Nowadays, the most common use of Service-Oriented Architectures (SOA) [21] is at a functional level, where it is required to integrate the internal applications of the company. The main problem was connecting and communicating the different units and departments in an easy and scalable way. By means of a SOA, this communication can be solved more easily because its functions are published as services, and they are consumed independently by other departments which utilize the same or other systems. The second use of SOA was derived from the results of the previous advantage, migrating the idea to a superior level, to the business processes instead of functional units. Therefore, Business Process Management (BPM) [10] was developed along with execution languages such as BPEL (Business Process Execution Language), notation languages (BPMN) and monitoring tools (BPA). As a result, a business process can be defined in a specific language and each activity within this process can be connected to the functional unit it depends on.

2.4 Cloud Computing

Cloud computing [7] represents a paradigm shift in the delivery architecture of information services. There is little consensus on how to define cloud computing, but we have adopted the following definition [11]: "*A large-scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet*". The National Institute of Standards and Technology (NIST) has established a group focused on promoting the effective and secure use of cloud technology within government and industry. They identify five essential characteristics (on-demand self-service, ubiquitous network access, location independent resource pooling, rapid elasticity and pay per use), three service models (Software-as-a-Service -SaaS-, Platform-as-a-Service -PaaS-, and Infrastructure-as-a-Service -IaaS-) and four development models (private, community, public and hybrid clouds) [24]. Virtualization constitutes the cornerstone technology for cloud computing and can be referred to as the abstraction of physical IT resources from the people and applications using them.

Cloud computing is defined as a new technology, but this is not completely right [18]. Cloud computing represents a new perspective that groups already known technologies such as operating systems,

databases, servers, networks, multitenancy, middleware, virtualization (abstraction of computer resources), management tools, etc., which have evolved during the last decades.

Also, it is often mistaken with grid computation, autonomous computation (can be managed by itself) or utility computing [4]. Certainly, cloud computing usually depends on grids for its implementation, possesses some autonomy (for example, reacts to high demand periods and readjust the resources automatically) and it is a pay per use, on-demand self-service. However, the cloud computing paradigm has a wider scope. The essence of cloud computation is not in the tools it is using (software, technologies, platforms, infrastructures, etc.), but in the way it use, group and integrate them, merging the most important basic principles for creating a unique and differentiator dynamics.

3 SITIO Architecture

The proposed architecture of the SITIO platform is depicted in Fig. 1. Three main elements can be distinguished: the user interface, the process and business services, and the metadata services. Next, these components are described in detail.

3.1 User Interface

The aim of the user interface is to enable users with different profiles and necessities to access to the SITIO platform and to use and manage their accounts and applications. The users in the platform can play two roles, namely, tenant or consumer. Tenants are software developers and IT providers that make use of the platform in order to deploy their applications. By means of the utilities provided by our platform, tenants can manage the applications they have placed in the system, setup various customized versions of these applications, upload new applications, and check, with different levels of details, the operating costs of their applications for using the underlying infrastructure. The objective is to assist in managing these applications in the most seamless way possible, and to abstract all maintenance duties from the developers themselves. Consumers, on the other hand, can be also referred to as the final users of the platform. They are allowed to search in the applications repository through a human-like natural language search interface [6], and get subscribed to those they are interested in. The final users have also access to an application management view in which they can review their subscriptions (i.e., add new applications and remove the unwanted ones) and monitor the applications they are subscribed to.

A set of JavaScript (JS) libraries constitutes the technology to build these user interfaces. In the last few years, several JS libraries have emerged in the dynamic HTML framework. Much of these libraries provide free access to a number of functions for creating Ajax controls and generating different visual effects. Furthermore, in most cases, these libraries are standard-compliant. For the purposes of our work, the user interfaces will be developed using dynamic HTLM with JS libraries and AJAX, when possible, to provide an agile and responsive communication capability with the central server. By using these technologies, we guarantee the provision of a usable, efficient and attractive user interface with the additional advantages of being the cross-browser compatible and OS independent.

The final decision as to what JavaScript framework to use was taken and the chosen one is ”Mochaui (http://mochaui.com/). This decision is based on different key factors related to the need for providing a good user experience. The first one is merely the appearance of the interface and a desktop application look and feel. The advance of this interfaces has been a key factor to start talking about a Cloud OS (/url-http://es.eyeos.org/), a user persistent session that can be accessed via web browser. The objective of SITIO is to make use of the advantages of these interfaces to have an interface that makes the user think that they are working in a desktop-like environment. Another key factor is the possibility to customize and modify mochaui and to create our own new and personalized SITIO application. New versions of this framework allow other possibilities like skin selection and will be available in new releases of SITIO interface.

At the time of writing this article the design of the application is almost complete. All the functionalities provided by the platform are depicted in a set of views and functionalities. There is special interest on the development of the tenant interface because it will be the base to manage, configure and provide SITIO main features. One of the main functionalities is to provide an easier management of the cloud and his applications and hiding the most reiterative and tedious operations behind an easy and intuitive interface instead of the more complex interfaces or APIs used by most of the Cloud Computing providers. Decisions like virtualization, OS, Web Server choice are pre-selected and the tenant only needs to worry about the development of his applications.

The interface, as shown in Fig. 2, is divided in several areas. There is a big frame on the centre where the different views are going to be shown. The left frame of the application contains the menu and at the right frame additional or configuration information will be displayed. When an application is selected, the

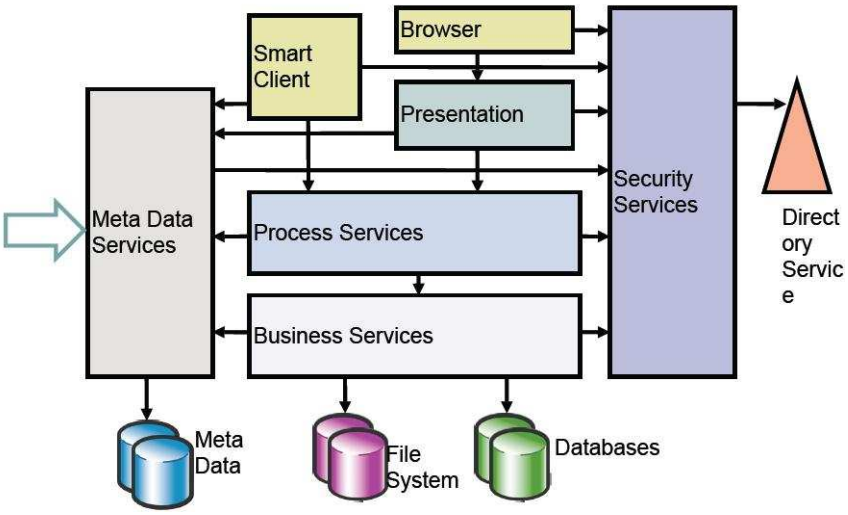

Figure 1: SITIO Architecture

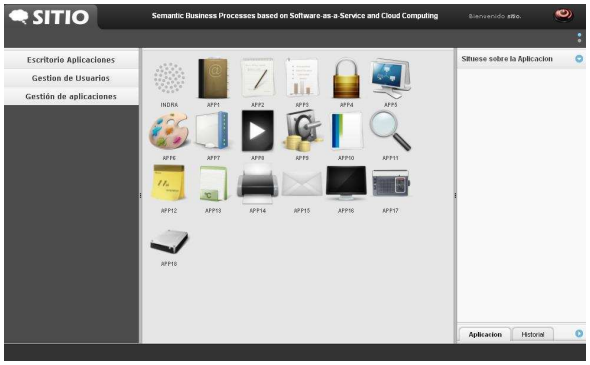

Figure 2: Interface snapshot

right view displays information such as disk storage, the owner and in a future it will be possible to configure the distribution of resources changing the number of master and slaves to solve performance issues. This is the starting point of SITIO interface, the tenant desktop view, and from here the tenant will be able manage his applications in a centralized view. Others view allow the tenant to upload new applications, to manage users, etc

3.2 Process and Business Services

These two layers of the SITIO architecture provide the ability to run arbitrary web services in the cloud computing platform. To achieve this, they add distribution, load balancing and data persistence services to a given application transparently. This is made possible by imposing some restrictions on the services that run on top of the platform, as being distributed as a Java EE 5 standard application.

Distribution is handled by uploading the application to a cluster of application servers. This cluster is managed from a central node (the domain administration server) that is used to ensure that all the application servers are synchronized, including all the configuration files common to the cluster and the applications running on it. That means that an application only need to be uploaded to a single server that will distribute it to the rest of the cluster automatically.

The rest of the servers are to be distributed among difference data centers, so a problem in a datacenter does not implies a service disruption. This is combined with the load-balancing service that distributes requests across all the instances running application servers available at the given time, means that if an instance fails, all the traffic gets redirected to other instances. To make this as smooth as possible to the end-user, all the session data is stored in a distributed database, as well as in the server handling client requests so, in case the instance serving the client fails, it will get reassigned to another instance without data loss (not needing to login again, etc.).

Furthermore, all data saved to the database servers provided by SITIO is automatically replicated in multiple database nodes for reliability and availability. The data from these nodes is also periodically backed up to an external backup device for added data security. The load balancing service also works in conjunction with the database servers to balance the load among all database servers, splitting read and write queries for increased read performance, which is the most common type of query in most applications. Reads are directed to a slave database server, in the same datacenter than the application server, while writes are directed to a master node of the database system.

There is also the possibility of using an external database system, whose configuration can be specified while uploading the application using the declarative language CARL, which also includes all the details needed for an application to work in SITIO. This allows a tenant to use their already existing database, saving the migration hassle or in the event that it doesn't trust the privacy offered by SITIO. The privacy regarding multiple tenants using the same database is implemented in a database access library that must be used to access the database. Given that this library resides in SITIO servers (and is accessed via JMX), it cannot be modified by the user, thus ensuring that he can't access other tenants data.

3.3 Metadata Services: Semantic Annotation of Web Services

In order to facilitate users to locate the desired application quickly, a mere syntactic description of the applications in the cloud registered in the platform is not enough. By making use of machine-processable content, it is possible to enable the automatic processing of users' queries. Thus, ontologies are employed in SITIO to provide the semantic description of the available services. In order to properly annotate a Web service, the annotator must be aware of the real purpose of each method of the service. Therefore, in most cases it will be the Web services developers themselves who are in charge of semantically annotating the services. Regretfully, Web services developers are hardly ever aware of the methods and principles concerning the Semantic Web and ontology-based knowledge representation mechanisms. It is thus necessary to elaborate a methodology to assist users in the annotation process by identifying relevant information in the description and implementation of the services and then suggesting annotations.

3.3.1 Tools for the semantic annotation of Web Services: related work

The joint application of the Semantic Web and Web Services technologies with the purpose of creating intelligent WS is referred as to Semantic Web Services [23]. Semantic Web Services add a semantic description to Web services, so that discovery, composition and invocation processes for these services can be done in an automatic way. The W3C is currently examining various approaches with the purpose of reaching a standard for the Semantic Web Services technology. At this point in time, there are four submissions under consideration, namely, OWL-S [20], WSMO [31], SWSF [1] and WSDL-S [17], and a complementary approach which is a W3C recommendation, namely, SAWSDL [34]. The first three approaches propose an ontology that semantically describes all relevant aspects of Web Services. On the other hand, WSDL-S and SAWSDL identify some WSDL and XML Schema extension attributes that support the semantic description of WSDL components.

Many research groups have worked on the development of tools and methodologies with the goal of automating the semantic annotation process. Some of these tools have been subject to a detailed investigation to establish their pros and cons. The platforms included in our study are "METEOR-S: Web Service annotation framework" (MWSAF) [28], "Automated Semantic Service Annotation with Machine Learning" (ASSAM) [13], "Automatic Annotation of Web Services Based on Workflow Definitions" (AAWSW) [2], "A proposal for semi-automatically annotating web services using SAWSDL" (ASWSDL) [30].

MWSAF was one of the first solutions in adding semantic information to WSDL files. It creates an intermediate representation (called "*SchemaGraph*") from a given WSDL file and an ontology in the repository. Then, it makes use of matching functions that take into account linguistic-based concept similarity measures to return a set of triples of the form ¡WSDL element, Ontology concept, matching score¿ which are used for creating the annotations. ASSAM, on the other hand, is a machine learning-based tool that provides the user with suggestions on how to annotate the elements in a WSDL file. Given a WSDL file, the system is able to iterate and semantically classify its operations, message parts, and types. A further approach is AAWSW, which focuses on Web services that take part in error-free workflows. In this context, the platform can generate new annotations for input/ouput operation parameters from the already annotated ones that belong to the operations to which they are linked. This approach is based on the semantic compatibility of the parameters. The last solution analyzed is ASWSDL. It proposes a methodology to extend existing SAWSDL annotation ontologies with concepts from ontologies. The matching and alignment between ontologies is performed by making use of a technique called "*semantic fields*" [26].

3.3.2 A methodology for semi-automatically annotating Web Services

From the results of our analysis, we elaborated a semi-automatically three-steps annotating methodology. The annotator must follow three sequential phases:

1. Collect information from the web service;

2. Find mappings between domain ontologies and the web service;

3. Web service annotation and expert user validation for the suggested annotations.

In the first step, to make easier the web service candidate elements identification process, two complementary approaches are followed: (i) manually annotating the services by using special tags and commentaries, and (ii) using natural language techniques. The assumption in the first approach is that while implementing a Web service, the developer adds keywords or named entities that precisely define the functionality provided by the methods that form part of the service. The natural language processing tools of the second approach aim at identifying "named entities" from the information available concerning a particular Web service (implementation, WSDL description, etc.). Based on the Web services-related information gathered in the previous step, the annotator looks for the elements in domain ontologies that are relevant for describing these services. The aim is, therefore, to carry out a classification process in which the Web services-related identified information elements are mapped to ontology components (concepts, relations, attributes, instances, etc.). Taking into account this mapping, the semantic description of the services is produced and the user is asked to validate the proposed annotations and to set the mappings for the elements that the tool has not been able to process automatically.

A software prototype in the form of a Web application has been developed that follows the proposed methodology. The prototype has been implemented in Java and makes use of Sesame, an RDF Schema-based repository and querying facility, to store the required ontologies. The main page of the application is shown in Fig. 3.

4 Related Work

Given the potential impact of cloud computing in industry, a large number of research projects are being carried out to improve the underneath technologies. A recurrent topic of these investigations is the addition of semantics to this formula in order to leverage the approach dynamism. Some of the works developed in this matter are discussed next.

In [25], the authors emphasize the importance that cloud computing techniques have for addressing the problems of scale in processing semantic data. They point out that a number of applications, particularly search engines, will have to cope with the ever increasing amount of semantic-based structured data that is being made available on the Web. The properties of the Web makes it suitable for a distributed approach to the management of data, and the authors believe that cloud computing is the most promising solution.

In [35], the authors undertook a task to construct a unified ontology of cloud computing, where the cloud is sectioned into five main layers - applications, software environments, software infrastructure, software kernel, and hardware. This ontology is used to classify different cloud systems and to capture the interrelations between the different cloud components to compose one cloud service from one or more other cloud services.

In [8], the authors propose an infrastructure composition model that aims at increasing the adaptability of the capabilities exposed through it by dynamically managing their non-functional requirements of on-demand services provisioned from the cloud. This is achieved by a mediator exposing a virtualized interface of the resource enhanced with a Secured Profile (SP). The SP itself will be formed using semantic technologies, which will describe the different components along with their constraints and allow for their dynamic selection and composition. The current implementation allows an enterprise to expose different capabilities as web services in a secure, dynamic, and virtualized manner.

In [14], the authors combine logic programming and communication abstractions to first specify independently from implementation Technologies cloud applications and then synthesize them automatically. The specifications are given for data, operation, and connectivity layer. Target cloud is characterized with a set of agent types, a communication policy and a data access policy. In this abstraction, a cloud consists of a finite set of nodes connected by ideal (no information loss), bidirectional, and point-to-point communication channels.

In [29], the authors advocate the rise of service parks, which would serve sets of Web services with their own sets of rules for combining and modifying them. These service parks are very constrained and thus technically feasible. Instead of providing access to Web services with heterogeneous semantics from a large variety of providers they offer Web services with homogeneous semantics by a selection of providers. These service parks offer customization possibilities, however limited, as well as guaranteed service-level agreements. Service parks typically include common runtime systems that greatly facilitate implementation of the services. Thus it would be natural to assume that these parks could operate on clouds from cloud computing providers.

In [19], the authors describe an infrastructure,

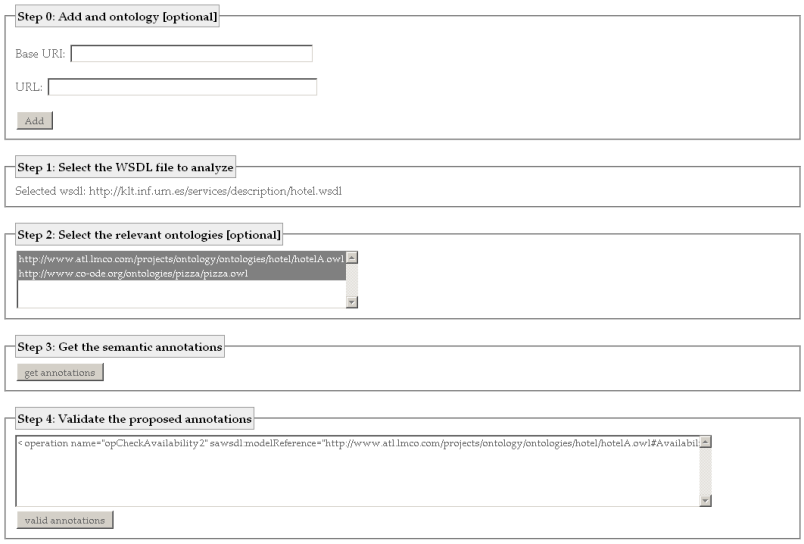

Figure 3: Interface snapshot of the SITIO Semantic Annotation Tool

namely Meandre, that takes advantage of semantics to simplify creation of cloud applications by non tech-savvy users. Meandre provides, (1) tools for creating components and flows, (2) a high-level language to describe flows, and (3) a multicore and distributed execution environment based on a service-oriented paradigm. Semantics in Meandre is used for simplifying creation of flows, while flow components need no adaptation for execution at cloud.

5 Conclusion

The impact of new paradigms of Integration and Management of Information Systems has been dramatic in most scenarios of society's day-to-day life. The revolution undergone and provoked by Information Technologies (IT) over the last years was unforeseeable, particularly the transformation of the Web from a mere repository of information to a vehicle for business transactions. Emerging approaches such as Semantic Technologies, Software-as-a-Service (SaaS), Business Process Management and Service-oriented Architectures are changing the current business landscape, as well as domains such as e-Science, education and e-Health, especially regarding their adaptability and interoperability, and derived from both of these characteristics, the incorporation of cognitive and reasoning capabilities.

However, the potential synergies of these emerging trends in IT have not yet been explored. In this work, we aim to gather these emerging concepts (SaaS, Semantic Technologies, Business Process Modeling and Cloud Computing) to foster dramatic evolution of a new platform oriented towards interoperability and cost reduction which can impact significantly in industry. SITIO can be defined as a Business Process based on Semantics platform where services are executed from a SaaS perspective. It allows external developers to create add-on applications that integrate into the main SITIO application and are hosted on Cloud Computing infrastructure.

Several issues remain open for future work. Semantics is currently only focused on the description of Web Services functionality and capabilities. This approach is satisfactory in most cases to assist users in finding the applications the are looking for and to enable the automatic composition of services to generate added-value applications, since Web services represent the most prominent technology for service provision on the Web. However, in order to make the SITIO platform as broadly applicable as possible, other service-based technologies (e.g. RMI, CORBA, REST, etc.) must be contemplated. On the other hand, security has received little attention in literature, even though it is a major concern in this kind of environments.

Acknowledgments: This work has been partially supported by European Union through the EUREKA Initiative ($\Sigma!4989$), and the Spanish Ministry for Industry, Tourism and Commerce through projects SITIO (TSI-0204000-2009-148), GO2 (TSI-020400-2009-127) and SONAR II (TSI-020100-2009-263).

References:

[1] S. Battle, A. Bernstein, H. Boley, B. Grosof, M. Gruninger, R. Hull, M. Kifer, D. Mar-

tin, S. McIlraith, D. McGuinness, J. Su, and S. Tabet. Semantic web services framework (swsf) overview. W3c member submission 9 september 2005, World Wide Web Consortium, 2005. http://www.w3.org/Submission/SWSF/.

[2] K. Belhajjame, S. M. Embury, N. W. Paton, R. Stevens, and C. A. Goble. Automatic annotation of web services based on workflow definitions. *ACM Transactions on the Web*, 2(2), 2008.

[3] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May:34–43, 2001.

[4] R. Buyya. Market-oriented cloud computing: Vision, hype, and reality of delivering computing as the 5th utility. In F. Cappello, C.-L. Wang, and R. Buyya, editors, *CCGRID*, page 1. IEEE Computer Society, 2009.

[5] M. Campbell-Kelly. Historical reflections - the rise, fall, and resurrection of software as a service. *Communications of the ACM*, 52(5):28–30, 2009.

[6] J. L. Chirlaque, R. Valencia-García, F. García-Sánchez, D. Castellanos-Nieves, and J. T. Fernández-Breis. Owl-path: an ontology-guided natural language query editor. In A. Quesada-Arencibia, J. C. Rodriguez, R. M.-D. jr., and R. Moreno-Diaz, editors, *12th International Conference on Computer Aided Systems Theory (EUROCAST)*, pages 326–327, Las Palmas de Gran Canaria, Spain, February 2009.

[7] M. Creeger. Cloud computing: An overview. *ACM Queue*, 7(5):2, 2009.

[8] P. de Leuss, P. Periorellis, T. Dimitrakos, and P. Watson. An architecture for non functional properties management in distributed computing. In M. Helfert, editor, *DCSOFT*, pages 26–37. INSTICC Press, 2008.

[9] K. Deshpande, P. Jalote, and S. K. Rajamani, editors. *Configurability in SaaS (software as a service) applications*. ACM, 2009.

[10] M. Dumas and M. Reichert. Guest editorial: Business process management. *Data and Knowledge Engineering*, 68(9):775–776, 2009.

[11] I. T. Foster, Y. Zhao, I. Raicu, and S. Lu. Cloud computing and grid computing 360-degree compared. *Computing Research Repository (CoRR)*, abs/0901.0131, 2009.

[12] T. R. Gruber. Knowledge acquisition. *A translation approach to portable ontology specifications*, 5:199–220, 1993.

[13] A. Heß, E. Johnston, and N. Kushmerick. Assam: A tool for semi-automatically annotating semantic web services. In S. A. McIlraith, D. Plexousakis, and F. van Harmelen, editors, *International Semantic Web Conference*, volume 3298 of *Lecture Notes in Computer Science*, pages 320–334. Springer, 2004.

[14] E. K. Jackson, W. Schulte, and D. Lucrdio. Synthesis of cloud applications using logic programming: Bam! http://research.microsoft.com/apps/pubs/default.aspx?id=77370.

[15] X. Jiang and Ah-HweeTana. Learning and inferencing in user ontology for personalized semantic web search. *Information Sciences: an International Journal*, 179(16):2794–2808, July 2009.

[16] P. A. Laplante, J. Zhang, and J. M. Voas. What's in a name? distinguishing between saas and soa. *IT Professional*, 10(3):46–50, 2008.

[17] K. Li, K. Verma, R. Mulye, R. Rabbani, J. A. Miller, and A. P. Sheth. Designing semantic web processes: The wsdl-s approach. In J. Cardoso and A. P. Sheth, editors, *Semantic Web Services, Processes and Applications*, volume 3 of *Semantic Web And Beyond Computing for Human Experience*, pages 161–193. Springer, 2006.

[18] H. Liu and D. Orban. Gridbatch: Cloud computing for large-scale data-intensive batch applications. In *CCGRID*, pages 295–305. IEEE Computer Society, 2008.

[19] X. Llorà, B. Ács, L. S. Auvil, B. Capitanu, M. E. Welge, and D. E. Goldberg. Meandre: Semantic-driven data-intensive flows in the clouds. In *ESCIENCE '08: Proceedings of the 2008 Fourth IEEE International Conference on eScience*, pages 238–245, Washington, DC, USA, 2008. IEEE Computer Society.

[20] D. L. Martin, M. H. Burstein, D. V. McDermott, S. A. McIlraith, M. Paolucci, K. P. Sycara, D. L. McGuinness, E. Sirin, and N. Srinivasan. Bringing semantics to web services with owl-s. In *World Wide Web*, pages 243–277, 2007.

[21] C. M. P. Matos and R. Heckel. Migrating legacy systems to service-oriented architectures. In *International Conference on Graph Transformation*, volume 16, 2008.

[22] D. L. McGuinness and van Harmelen F. (eds.). Owl web ontology language overview. W3c recommendation 10 feb 2004, World Wide Web Consortium, February 2004. http://www.w3.org/TR/owl-features/.

[23] S. McIlraith, T. C. Son, and H. Zeng. Semantic web services. *IEEE Intelligent Systems*, 16(2):46–53, 2001.

[24] P. Mell and T. Grance. The nist definition of cloud computing. Technical report, National Institute of Standards and Technology, Information Technology Laboratory, 2009. http://csrc.nist.gov/groups/SNS/cloud-computing/index.html.

[25] P. Mika and G. Tummarello. Web semantics in the clouds. *IEEE Intelligent Systems*, 23(5):82–87, 2008.

[26] I. Navas-Delgado, M. del Mar Roldán-García, and J. F. Aldana-Montes. Semantic fields: Finding ontology relationships. In S. S. Bhowmick, J. Küng, and R. Wagner, editors, *DEXA*, volume 5690 of *Lecture Notes in Computer Science*, pages 427–434. Springer, 2009.

[27] E. Pascalau and A. Giurca. Towards enabling saas for business rules. In W. Abramowicz, L. A. Maciaszek, R. Kowalczyk, and A. Speck, editors, *BPSC*, volume 147 of *LNI*, pages 207–222. GI, 2009.

[28] A. A. Patil, S. A. Oundhakar, A. P. Sheth, and K. Verma. Meteor-s web service annotation framework. In S. I. Feldman, M. Uretsky, M. Najork, and C. E. Wills, editors, *WWW*, pages 553–562. ACM, 2004.

[29] C. J. Petrie and C. Bussler. The myth of open web services: The rise of the service parks. *IEEE Internet Computing*, 12(3):96–95, 2008.

[30] A. J. Roa-Valverde, J. Martinez-Gil, and J. F. Aldana-Montes. Una propuesta para la anotación semiautomática de servicios web usando sawsdl. In *V Jornadas Científico-Técnicas en Servicios Web y SOA*, 2009.

[31] D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, M. Stollberg, A. Polleres, C. Feier, C. Bussler, and D. Fensel. Web service modeling ontology. *Applied Ontology*, 1(1):77–106, 2005.

[32] N. Shadbolt, T. Berners-Lee, and W. Hall. The semantic web revisited. *IEEE Intelligent Systems*, 21(3):96–101, 2006.

[33] R. Studer, R. Benjamins, and D. Fensel. Knowledge engineering: Principles and methods. *Data and Knowledge Engineering*, 25(1-2):161–197, 1998.

[34] K. Verma and A. P. Sheth. Semantically annotating a web service. *IEEE Internet Computing*, 11(2):83–85, 2007.

[35] L. Youseff, M. Butrico, and D. Da Silva. Toward a unified ontology of cloud computing. In *Grid Computing Environments Workshop, 2008. GCE '08*, pages 1–10, Nov 2008.

[36] H.-T. Zheng, B.-Y. Kang, and H.-G. Kim. An ontology-based approach to learnable focused crawling. *Information Sciences: an International Journal*, 178(23):4512–4522, 2008.