Design Abstractions for Innovative Web Applications: The Case of the SOA Augmented with Semantics

Stefano Ceri[1], Marco Brambilla[1], and Emanuele Della Valle[2]

[1] Dipartimento di Elettronica e Informazione, Politecnico di Milano
Piazza L. Da Vinci, 32. I20133 Milano, Italy
{ceri,mbrambil}@elet.polimi.it
[2] CEFRIEL, via Fucini 2, 20133 Milano, Italy
dellavalle@cefriel.it

Abstract. This work presents a retrospective analysis of how we have addressed new challenges in Web technologies and applications. WebML, which was first defined about 10 years ago, has been an incubator for research concerning abstractions, methods, tools, and technologies, acting as a glue within a group of people spread among universities, technology transfer centres, and a spin-off. In this paper, we first illustrate the common approach to innovation, and then show such approach at work in two contexts. One of them, dealing with "Service-Oriented Architectures" (SOA), has reached a mature state; the other one, "Semantic Web Services" (SWS), is at its infancy, but promises to deliver very interesting results in the forthcoming years.

1 Introduction and Motivation

Data-intensive Web applications, i.e. applications whose main purpose is to give access to well-organized content, represented the first industrial application of the Web, and still constitute the most important Web application in terms of volumes and commercial value. All companies have an institutional site showing their business and describing their offers, and many companies address their customers either electively or exclusively via the Web. Therefore, these applications have been most covered by methods and tools, which have been available for a long time.

Among them, Web Modelling Language (WebML) [1] was defined, about 8 years ago, as a conceptual model for data-intensive Web applications. Early deployment technologies were very unstable and immature; as a reaction, WebML was thought as a high level, implementation-independent conceptual model, and the associated design support environment, called WebRatio [9], has always been platform-independent, so as to adapt to frequent technological changes.

WebML can be considered, in MDA terms, as a Domain Specific Language in the area of Web application development. It is based upon orthogonal separation of concerns: content, interface logics, and presentation logics are defined as separate components. The main innovation in WebML comes from the interface logics (patented in 2003) that enables the computation of Web pages made up of logical components (units) interconnected by logical links (i.e., not only the units but also the links have a formal semantics); the computation is associated with powerful defaults so as to associate to simple diagrams all the required semantics for a full deployment, through code generators. WebML anticipated the concepts and methods formally proposed by the MDA framework, introducing the idea of model transformation and code generation.

While the Web has gone through waves of innovation, new application sectors have developed, and revolutionary concepts – such as enabling the interaction of software artefacts rather than only humans – are opening up. While the foundations of the WebML model and method are still the same, the pragmatics of its interpretation and use has dramatically changed through the last years. Several new challenges have been addressed within the WebML context, including:

- Web services and service-oriented architectures [3];
- Integration with business processes [4];
- Personalization and adaptation;
- Context awareness and mobility [5];
- Rich client-side applications;
- Embedded Web applications;
- Semantic Web and Semantic Web Services [6,7].

A retrospective consideration of our work shows that, in all the above situations, we have addressed every new challenge by using a common approach, which indeed has become evident to us during the course of time, and now is well understood and consolidated. For every new research directions, we had to address four different kinds of extensions, respectively addressing the development process, the content model, the hypertext meta-model, and the tool framework.

- *Extensions of the development process* capture the new steps of the design that are needed to address the new functionalities, providing as well the methodological guidelines and best practices for helping designers.
- *Extensions of the content model* capture state information associated with providing the new functionalities, in the format of standard model, e.g. a collection of entities and relationship that is common to all applications; this standard model is intertwined with the application model, so as to enable a unified use of all available content[1].
- *Extension of the hypertext meta-model* capture the new abstractions that are required for addressing the new functionalities within the design of WebML specifications, through new kinds of units and links which constitute a functionality-specific “library”, which adds to the “previous” ones;
- *Extensions of the tool framework* introduce new tools in order to extend those modelling capability falling outside of standard WebRatio components (content, interface logics, presentation logics), or to empower users with new interfaces and wizards to express the semantics of new units and links in terms of existing ones, or to provided direct execution support for new units and links (e.g. sending an email).

[1] Note that any WebML application includes the standard entities User and Group.

This paper demonstrates how this four-step development occurred in the case of Service Oriented Architectures and how we are naturally extending that approach to deal with Semantic Web Services. The treatment of each extension is necessarily concise and visual, for more details we refer readers to published papers and reports.

2 Support of Service-Oriented Architectures

The specification of a Web application according to WebML [2] consists of a set of orthogonal models: the application *data model* (i.e., an extended Entity-Relationship model), one or more *hypertext models* (i.e., different site views for different types of users), expressing the navigation paths and the page composition of the Web application; and the *presentation model*, describing the visual aspects of the pages. A hypertext site view is a graph of pages; pages are composed of units, which are used for publishing atomic pieces of information, and operations, for modifying data or performing arbitrary business actions (e.g., sending e-mails). Units and operations are connected by links, to allow navigation, passing of parameters between the components, and computation of the hypertext. The need for incorporating external logic was felt relatively early, and the initial solution consisted of "custom units" which allow modelling user-defined computations.

The first WebML extension discussed in this paper is towards the Service Oriented Architectures. The requirement addressed in this case is to provide adequate design tools for Web Services and Service-oriented applications. The outcomes of our work included:

- The extension to the development process and the definition of some methodological guidelines for SOA design;
- Two standard models for representing the services and the business processes to be performed;
- New design primitives (namely, WebML units and links) for covering Web service specification and invocation, together with primitives for enforcing business process constraints;
- The support of the specified solutions through a process modeller, a translator of processes into sketches of hypertexts, and an XML2XML mapping tool.

2.1 Process Extensions

The original design process, explained in chapter 6 of [2], included the classic phases of requirement analysis, data design, hypertext design, and presentation design, followed by architecture design and implementation. The 4-step procedure, going from requirements to data to hypertext to presentation, is iterated multiple times through the use of WebRatio, which can be considered as a rapid prototyping environment; and indeed a lot of the advantage of using the approach comes exactly from the ability to generate a prototype whenever required by the need of interaction with stakeholders.

The extension of the original design process to SOA requires adding a phase for modeling the business process and separating application from service design, as shown in Fig. 1. For each addition, new guidelines and best practices were defined.

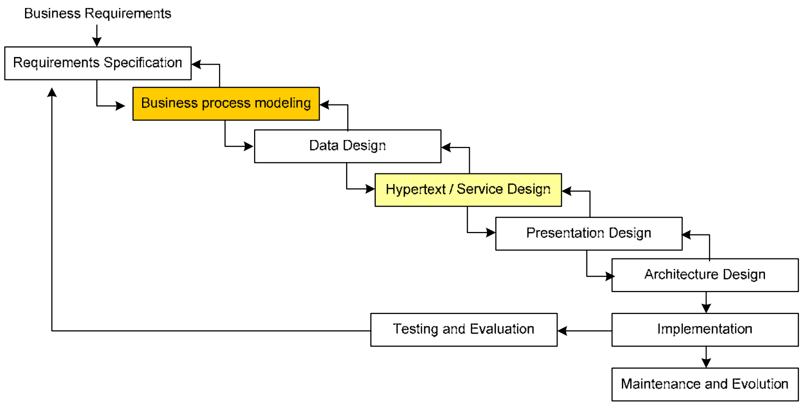

Fig. 1. Development process extensions for SOA

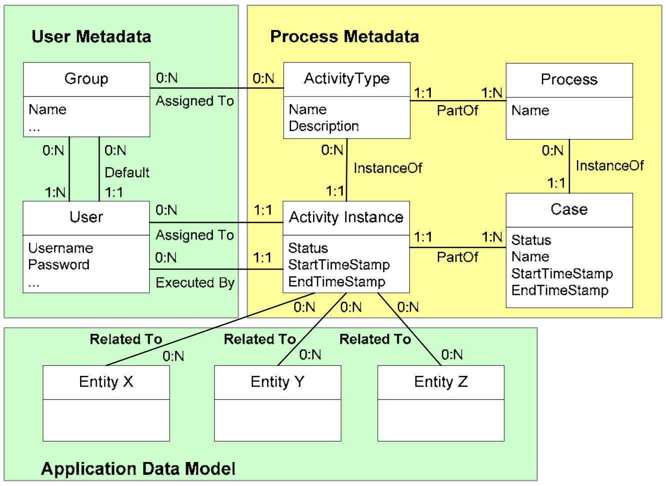

Fig. 2. Standard model for the specification of the business process status

2.2 Content Model Extensions

The standard model for supporting SOA deals with two aspects: a description of Web services and the specification of the workflow state. The first standard model represents Web services according to WSDL, and is omitted here (see [3]); the second standard model represents the information about the implemented business process, shown in Fig. 2 (see [4] for details). In the model, entity Process represents processes and is associated with entity ActivityType, representing the kinds of activities that can be executed in a process. Entity Case denotes an instance of a process and is related to its Process (relationship InstanceOf) and to its activities (via relationship PartOf); entity ActivityInstance denotes the actual occurrences of activities within cases.

2.3 Hypertext Meta-model Extensions

Two groups of new design primitives have been added to WebML, describing Web services and workflow-based applications.

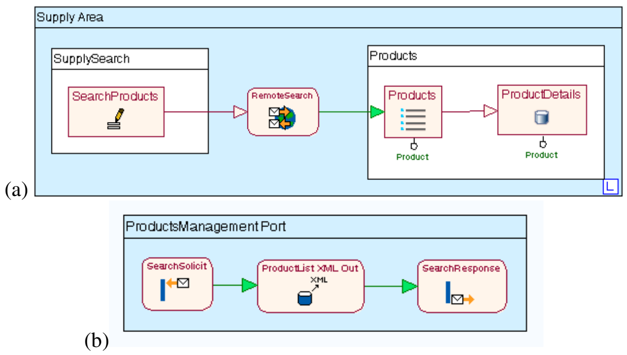

Fig. 3. Example of WebML hypertext model with invocation of a remote service

A new library of *Web service units* [3] has been defined, corresponding to the WSDL classes of Web service operations. These primitives consist in:

- Web service publishing concepts, including *Service view* (a new view supported in WebML specifically dedicated to publishing a service), *Port* (corresponding to the WSDL port concept), *Solicit unit* (representing the end-point of a Web service), and *Response unit* (providing the response at the end of a Web service implementation);
- Web service invocation primitives, namely *Request-response* and *Request* units, to be used within the application for invoking remote services.

For instance, Fig. 3 shows a hypertext that includes the model of a Web service call and of the called Web service. In *Supply Area* of Fig. 3a, the user can browse the *SupplySearch* page, in which the *SearchProducts* entry unit permits the input of search criteria. From this information, a request message is composed and sent to the *RemoteSearch* operation of a Web service. The user then waits for the response message, containing a list of products satisfying the search criteria. From this list, a set of instances of *Product* are created, and displayed to the user by means of the *Products* index unit in the *Products* page; the user may continue browsing, e.g., by choosing one of the displayed products and looking at its details. Fig. 3b represents the model of the *RemoteSearch* service invoked by the previously described hypertext. The interaction starts with the solicit *SearchSolicit* unit, which denotes the reception of the message. Upon the arrival of the message, an XML-out operation extracts from the local data source the list of desired products and formats the resulting XML document. The *SearchResponse* unit produces the response message for the invoker[2].

To cover the development of B2B Web applications implementing business processes, new primitives have been defined for specification of activity boundaries (namely *Activity areas* within the hypertext) and business process-dependent navigation (namely *workflow links*). Fig. 4 shows some of these primitives: site areas marked as “Activity Areas” (A); special incoming links for checking the correctness of the status and starting an activity (i.e., Start and Resume links); special outgoing links for closing an activity (Complete and Suspend links).

Distributed processes and *SOA* can be obtained by combining the workflow primitives with Web services primitives [4].

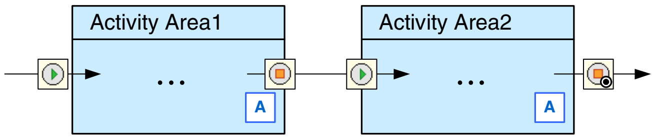

Fig. 4. Two activity areas and corresponding Start and End links

2.4 Tool Framework Extensions

For supporting the design of the new classes of applications, some facilities have been prototyped and are currently being ported to commercial versions of WebRatio:

- A workflow modeling editor that allows to specify business processes according to the BPMN notation.
- A set of model transformations that translate a business process model into a skeleton of WebML hypertext model.
- A visual editor for XML2XML mapping for helping the design of XML transformations to better support messages exchange between Web services.

3 Support of Semantic Web Services

Traditionally, the service requestor and service provider are designed together and then tightly bound together when an application is created. The emerging field of Semantic Web Services (SWS) [10] provides paradigms for semantically enriching the existing syntactic descriptions of Web services; then, the service requestor can search, either at design or at run time, among a variety of Web-enabled service providers, by choosing the service that best fits the requestor’s requirement. Such a flexible binding of requestor and providers allows for dynamic and evolving applications to be created utilizing automatic resource discovery, selection, mediation and invocation.

[2] Service ports are an example of software component that is modelled by using WebML and yet has no interaction with users (hence, no “presentation logics”), and shows that the original motivation of the model has shifted to adapt to new requirements. Even more radical shifts will be needed to deal with semantic web services, as illustrated in the sequel.

Our purpose in approaching the SWS is obviously not to design new methods for performing the above tasks: a community of researchers is working on them. Instead, we aim at extending WebML and WebRatio so as to generate, on top of conventional models (of: processes, data, services, and interfaces), a large portion of the semantic descriptions required by the SWS in a semi-automatic manner; this possibility descends from the fact that WebML is a very rich model, with a lot of embedded semantics - to the point that code can be completely generated from the model with no user intervention. In the same way, some SWS annotations can be automatically generated, conveying a large fraction of the semantics that is typically carried by manual SWS annotations.

In the rest of the section we highlight the following extensions to WebML to cope with SWS[3] requirements:

- Extension of the development process by adding phases for ontology import and for semantic annotation of services;
- Extensions of the standard model, together with a discussion of the relationships between meta models and ontologies;
- Definition of the new primitives in order to manipulate semantic contents;
- Implementation of new tools integrating Semantic Web Service editors and execution environment.

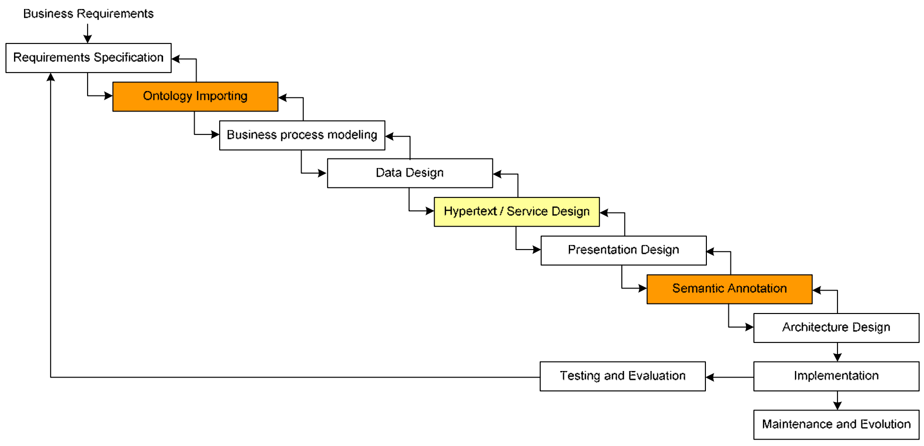

Fig. 5. Development process extensions for SWS

3.1 Process Extensions

To address the new SWS requirements, we extended the process defined for SOA in Fig. 1 with two additional tasks, shown in Fig. 5:

- **Ontology Importing,** for importing existing domain ontologies that may be exploited for describing the domain of the Web application under development.

[3] In our approach we considered WSMO, but being, WSMO the most comprehensive approach to SWS, our experience can be easily extended to OWL-S and WSLD-S approaches.

The imported ontologies should be integrated, at the model level, with the application-specific E-R model, so as to offer an integrated view to the designer.

- **Semantic Annotation,** for specifying (either manually or automatically) how the hypertext pages or services will be annotated using existing ontological knowledge.

3.2 Content Model Extensions

The management of content in Semantic Web applications, thus also in SWS applications, needs to address two main concerns: *(i)* the possibility of importing and integrating existing third-party ontologies and *(ii)* the capability of combining relational data sources with ontological contents.

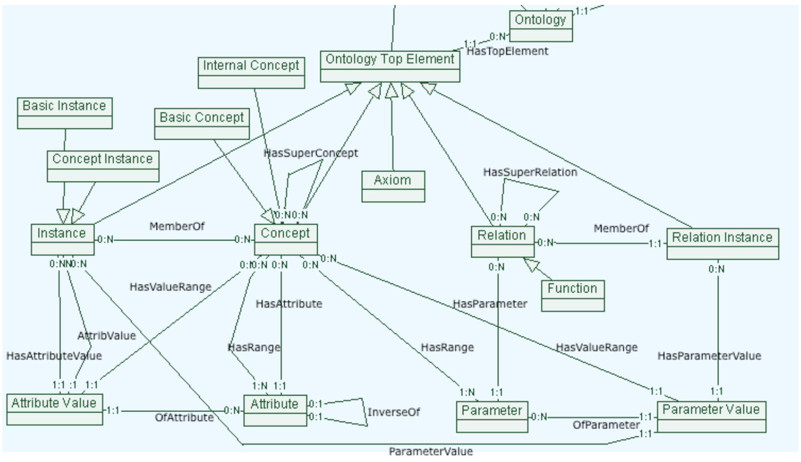

Fig. 6. Standard model of the WSMO ontology structure

We address these two issues by defining a E-R standard model representing ontological concepts, thus allowing to associate in a seamless way semantic content to conventional content defined for the application; Fig. 6 shows a piece of E-R model representing WSMO ontological language. Imported ontological data can be either copied into an application-specific implementation of the E-R model (typically a relational database) or queried directly on a remote ontology. Different implementations of ontology query primitives must be developed in the two cases (see Section 3.3).

3.3 Hypertext Meta-model Extensions

The basic WebML primitives for data retrieval have been used up to now for querying implementations of E-R models, but their generality makes them perfectly fitting in the role of query and navigation of ontologies. The additional expressive power of ontological languages, however, requires some extensions. We have therefore introduced a new set of primitives (inspired by SPARQL and RDF-S syntax) to describe advanced queries over ontological data. These units (see Fig. 7) allow queries on classes, instances, properties, and values; checking the existence of specific concepts; and verifying whether a relationship holds between two resources. Other units import content from an ontology or return the RDF description of a given portion of the standard ontological model for exporting. Operations such as lifting and lowering have renamed specific XML2XML mappings used in the context of SOAs.

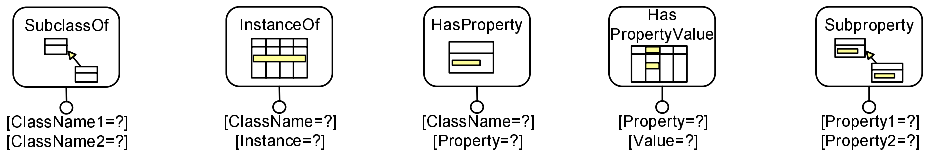

Fig. 7. Ontological query units

These primitives may have different implementations: when invoking a remote semantic repository, the implementation can exploit ontological query languages; when querying ontological data stored internally, hence already integrated within a relational source, the implementation is directly mapped to such source.

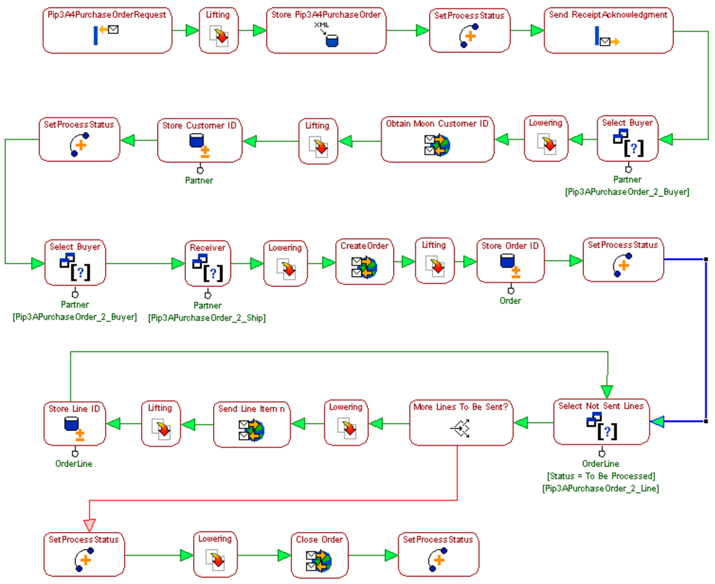

Fig. 8. WebML model of a mediator

These units, together with the standard WebML primitives and the solutions introduced for the SOA, allow specifying completely new kinds of applications with respect to the ones for which WebML was originally conceived. For instance, Fig. shows the WebML model of a WSMO mediator [7] in the context of a B2B purchase interaction for the SWS Challenge 2006 [19]. The logics of the mediator is that of receiving a single purchase order request, containing multiple lines bundled together, and then dispatch each order line to a service which exposes multiple ports, including one for accepting the general information about new orders and one accepting each line separately. We do not expect that the mediator specification can be appreciated in detail, but the reader should notice that the specification is fully graphic, that it embodies a complex workflow, and in particular it incorporates several request-responses for the SWS orchestration. Clearly, no user interaction is involved.

3.4 Tool Framework Extensions

The framework has been extended by providing automatic generators of WSMO-compliant descriptions (goals, choreographies, capabilities, and mediators) from the models already available in WebML, i.e., business processes, content models, and application logics models. Annotations that are automatically generated cannot express the full semantics of SWS services and applications[4], but they give initial descriptions, that can later be integrated manually. In particular, in the contest of the SWS Challenge 2006, we used WSMT [13] as ontology and SWS editor. As a result of the annotation process, applications and services can be deployed on a SWS runtime environment which provides generic services (e.g., service discovery engines, goal matchers, mediators). Again, in the SWS Challenge we have used the Glue discovery engine [18] as reasoner specialized for Web service discovery.

4 Related Work

Our approach has several elements which are common to a number of research centres and companies working towards improving Web Engineering methods and tools; here we list only a few of them. Traditional Web design methodologies (like OOHDM [15]) and new approaches (like Hera [16]) are now focusing on Semantic Web applications. MIDAS is a framework based on MDA for Semantic Web applications [14]. Research efforts are converging on the proposal of combining Semantic Web Services (SWS) and Business Process Management (BPM) to create one consolidated technology, called Semantic Business Process Management (SBPM) [17].

Our approach largely benefits from the WSMO [10] initiative, which provides a comprehensive framework for handling Semantic Web Services; specifically, we benefit from the WSMO conceptual model [10], the WSML language [11], the WSMX execution environment [12], and WSMT design environment [13].

[4] For instance, the process description yields to deriving a specific orchestration of the services, but in a full SWS specification we need to define choreographies, i.e., rules that indicate all the legal sequences of SWS invocations. Such rules must be derived by extending the initial annotations.

5 Conclusions

The “WebML approach” has acted as a framework for continuous innovation and exploration of new research directions. This is made possible by a unique combination of features:

- Availability of well-defined conceptual models;
- Extensibility of the model thanks to a plug-in based structure;
- Formally defined development process for Web applications;
- Availability of a CASE tool for fast prototyping of application and easy integration of new features and components;
- Strong link between the research (mostly performed in university) and the application development (performed within a spin-off);
- Interactions with real world requirements, enabled by interaction with customers.
- Participation to the international research community, through experience and people exchange and several EU-funded projects.

This mix of ingredients has allowed us to follow our peculiar pathway to innovation.

Acknowledgements

We wish to thank all the people who work in the WebML framework: the WebML group at Politecnico di Milano (and particularly Piero Fraternali), the CEFRIEL Semantic Web Activities group, and the WebRatio staff (and particularly Aldo Bongio). Work on the SOA was performed together with Piero Fraternali, Ioana Manolescu, and Sara Comai; work on SWS was performed together with Federico Facca, Christina Tziviskou, Dario Cerizza, Irene Celino and Andrea Turati.

References – WebML

[1] S. Ceri, P. Fraternali, A. Bongio. Web Modeling Language (WebML): a modeling language for designing Web sites. WWW9 / Computer Networks 33, 2000.
[2] S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, M. Matera. Designing Data-Intensive Web Applications. Morgan Kaufmann, 2002.
[3] I. Manolescu, M. Brambilla, S. Ceri, S. Comai, P. Fraternali. Model-Driven Design and Deployment of Service-Enabled Web Applications. ACM TOIT, 5:3, 2005.
[4] M. Brambilla, S. Ceri, P. Fraternali, I. Manolescu. Process Modeling in Web Applications. ACM TOSEM, 15:4, 2006.
[5] S. Ceri, F. Daniel, M. Matera, F. Facca. Model-driven Development of Context-Aware Web Applications, ACM TOIT, 7:1, 2007.
[6] M. Brambilla, I. Celino, S. Ceri, D. Cerizza, E. Della Valle, F. Facca. A Software Engineering Approach to Design and Development of Semantic Web Service Applications. International Semantic Web Conference (ISWC2006), Athens, USA, November 2006, Springer LNCS 4273.

[7] M. Brambilla, S. Ceri, D. Cerizza, E. Della Valle, F. Facca, P. Fraternali, C. Tziviskou. Coping with Requirements Changes: SWS-challenge phase II. SWS Challenge 2006, Phase II, Budva, Montenegro, June 2006.
[8] WebML: http://www.webml.org.
[9] WebRatio: http://www.webratio.com/.

References – Related Work

[10] Fensel, D., Lausen, H., Polleres, A., de Bruijn, J., Stollberg, M., Roman, D., Domingue, J.: Enabling Semantic Web Services – The Web Service Modeling Ontology. Springer (2006)
[11] de Bruijn, J., Lausen, H., Polleres, A., Fensel, D.: The web service modeling language: An overview. In: Proceedings of the 3rd European SemanticWeb Conference (ESWC2006), Budva, Montenegro, Springer-Verlag (2006)
[12] Haller, A. , Cimpian, E., Mocan, A., Oren, E., Bussler, C.: WSMX - A Semantic Service-Oriented Architecture. In Proceedings of the 2005 IEEE International Conference on Web Services (ICWS 2005), Orlando, FL, USA, 321–328, 2005.
[13] Kerrigan, M.: The WSML Editor Plug-in to the Web Services Modeling Toolkit. In Proceedings of 2nd WSMO Implementation Workshop (WIW2005), Innsbruck, Austria, 2005.
[14] Acuña, C. J., Marcos, E.: Modeling semantic web services: a case study. In Proceedings of the 6th International Conference on Web Engineering (ICWE 2006), Palo Alto, California, USA, 32-39.
[15] Schwabe, D., Rossi, G. The Object-Oriented Hypermedia Design Model. In Communications of the ACM, 38 (8), 45-46.
[16] Vdovjak, R., Frasincar, F., Houben, G. J., Barna, P.: Engineering semantic web information systems in Hera. Journal of Web Engineering, Rinton Press, 2(1-2), 3 -26, 2003.
[17] Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel, D.: Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management. In Proceedings of the IEEE ICEBE 2005, October 18-20, Beijing, China, 535-540.
[18] Della Valle, E., Cerizza, D.: The mediators centric approach to automatic webservice discovery of Glue. First Intl. Workshop on Mediation in Semantic Web Services (MEDIATE 2005), Amsterdam, The Netherlands, December 2005.
[19] Semantic Web Service Challenge 2006: http://www.sws-challenge.org/.