WSMO Studio – A Semantic Web Services Modelling Environment for WSMO (System Description)

Marin Dimitrov, Alex Simov, Vassil Momtchev, and Mihail Konstantinov

Ontotext Lab. / Sirma Group
135 Tsarigradsko Shose Blvd., Sofia 1784, Bulgaria
`firstname.lastname@ontotext.com`

Abstract. The Web Service Modelling Ontology (WSMO) provides a unique, highly innovative perspective onto the Semantic Web Services domain. Robust and easy-to-use tools play crucial role for the adoption of any technological innovation and indeed the overall value of the innovation can be severely undermined by the lack of proper tools supporting it. In this paper we present a prototype of an integrated modelling environment that supports and elaborates the innovative WSMO perspective.

1 Introduction

Robust and easy-to-use tools play a crucial role for the adoption of any new technology. Indeed, the overall value of a technological innovation can be severely undermined by the lack of proper tools to support it. The Web Service Modelling Ontology (WSMO, [1][2]) provides a unique, highly innovative perspective onto the Semantic Web Services domain.

Unfortunately, current tool support in the area is still lagging behind the theoretical advancements but real progress can be achieved only when the Semantic Web Services technology is easily usable. Another major problem with the tool landscape at present is that almost all of the available tools focus on only one of the relevant Semantic Web Services aspects (for example only ontology management or only service composition), while end users rarely focus on a single task and an integrated modelling environment will be more appropriate and productive.

In this paper we present a prototype that supports and elaborates the innovative WSMO perspective, making the technology accessible and easy to use for early adopters. In particular, we present *WSMO Studio*¹ – an open source integrated modelling environment for the Semantic Web Services domain, which is being developed within several EU-funded research projects².

¹ <http://www.wsmostudio.org>

² The work described in this document is partly funded by the European Commission under the IST projects DIP (FP6-507483), InfraWebs (FP6-511723), SemanticGov (FP6-027517) and SUPER (FP6-026850).

This paper is organised as follows: section 2 presents the goals of *WSMO Studio*. section 3 provides details on present *WSMO Studio* functionality. section 4 attempts to evaluate the extent to which *WSMO Studio* has achieved the defined goals and contains a brief overview of several prominent Semantic Web Services tools and their relation to *WSMO Studio*. Finally, section 5 provides details on work in progress for domain specific extensions of *WSMO Studio*.

2 WSMO Studio Goals

The three main goals of *WSMO Studio* are:

1. Providing a prototype that supports and elaborates the WSMO approach to Semantic Web Services, making the technology accessible and easy to use for early adopters. As we have already pointed out, adequate tool support is crucial for the adoption of a technological innovation.
2. Providing an integrated environment that maximises the productivity of the user. Current SWS tools usually focus on only one aspect (ontology editing, service composition, etc.) but users most often need functionality that covers various tasks in an integrated manner.
3. Providing an extensible environment where 3rd party tool providers can easily add new functionality or modify and customise existing functionality.

With respect to the first goal, we have already presented in [3] a summary of the tasks that a Semantic Web Services modelling environment should cover, such as:

- ontology modelling,
- semantic annotation of existing web services,
- working with semantic repositories for publishing, browsing and querying of WSMO ontologies, services, goals and mediators,
- goal-based service discovery,
- specification of service compositions (i.e. orchestration and choreography interfaces),
- interaction with Semantic Web Service runtime environments (such as WSMX³ and IRS-III⁴).

The second goal we have set is providing functionality that covers various related Semantic Web Services tasks in an integrated modelling environment, so that users will not need to use several different tools to accomplish their goals. As noted in [4] users rarely focus on only one aspect or task when they work – indeed, users usually play different roles and perform different related tasks, so the environment should provide functionality covering various perspectives and tasks.

³ <http://www.wsmx.org>

⁴ <http://kmi.open.ac.uk/projects/irs/>

Finally, extensibility is a requirement that is mostly ignored by the current Semantic Web Services tools. But as noted in [4]:

... open-ended extensibility is essential in the commercial IDE arena because no IDE vendor could possibly provide a sufficient set of useful tools to satisfy all customer needs. Which third party tool will be bundled as an add-in for a particular IDE is determined by market forces.

This statement, based on commercial IDE experience, can equally well be applied for SWS modelling environments. Indeed, tool extensibility is even more important for an emerging domain such as the Semantic Web Services one, so that when the domain evolves, tools will be able to follow this evolution and provide the relevant functionality.

3 WSMO Studio Functionality

This section presents a summary of the main features of *WSMO Studio*, with respect to the requirements identified in [3]. A detailed overview of *WSMO Studio* is available in [5].

WSMO Studio is built on top of the Eclipse platform⁵. The functionality of *WSMO Studio* is available both as a standalone application and as a set of individual components (called *plug-ins*), that can be incorporated into 3rd party applications.

An important feature of Eclipse is that its component model presents a declarative specification of ways to extend an application, called *extension points* [6]. New plug-ins may extend existing plug-ins and may be easily incorporated into an application.

3.1 Core Components

The core runtime layer provides functionality common across all components of the *WSMO Studio*.

The most important component of this layer is the *wsmo4j* plug-in, based on the *wsmo4j* framework⁶. The functionality includes: creating WSMO models, validating models, export and import from various WSML formats (as defined in [7]) and languages such as RDF and a subset of OWL-DL.

The *wsmo4j* plug-in publishes several extension points, so that 3rd party extensions (for example a new parser) can be easily integrated.

In addition, the runtime layer contains several utility components for workspace management and a shared entity cache.

3.2 WSMO Editor

This plug-in provides the main User Interface for modelling of WSMO ontologies, goals and services. Import and export from WSML formats, RDF and a subset of

⁵ <http://www.eclipse.org>

⁶ <http://wsmo4j.sourceforge.net>

OWL-DL is provided via the core layer. Most of the User Interface elements of the WSMO editor are also available for extension and customisation via published extension points.

3.3 Choreography Editor

The Choreography editor (Figure 1) provides the User Interface for describing WSMO choreography interfaces, as defined in [8].

A WSMO choreography description is comprised of:

- a *state signature* that specifies the concepts and relations of an ontology that will be used to represent the choreography states, together with their respective roles.
- a set of *transition rules* that express the state changes.

Fig. 1. WSMO Studio – choreography editor

The choreography descriptions, created by *WSMO Studio*, are used by execution environments (such as WSMX) during the execution of Semantic Web Services.

3.4 SAWSDL Editor

The SAWSDL editor (Figure 2) provides functionality for attaching semantic annotations to existing WSDL descriptions according to the SAWSDL recommendation [9]. SAWSDL is used at present as the grounding mechanism for

WSMO, i.e. mapping the semantic interface description of a service (in terms of imported ontologies, capability and choreography interface) to its WSDL interface.

The User Interface allows that semantic annotations are attached to the following WSDL elements: simple / complex XML types, messages, operations and interfaces.

Fig. 2. WSMO Studio – SAWSDL editor

At present the SAWSDL editor does not provide support for the lifting/lowering schema, i.e. the XSLT expressions that define the exact mappings between complex XML types defined in WSDL documents and concepts in an ontology, but such functionality will be provided in future versions.

3.5 Repository Front-End

The Repository front-end provides an abstraction of an Semantic Web Service repository for storing and querying WSMO descriptions.

WSMO Studio will be able to interact with a particular repository as long as it provides a special adaptor based on the WSMO API repository interfaces⁷.

⁷ <http://wsmo4j.sourceforge.net>

At present *WSMO Studio* provides an integrated ORDI repository⁸ as well as adaptors for remote IRS-III and WSMX repositories. Adaptors for other repositories will be developed in the future.

3.6 Service Discovery Front-End

Discovery of WSMO services at present is based on the idea of goal-based discovery [10], i.e. the user provides a request represented as a WSMO goal (possibly with some additional information such as quality-of-service or cost restrictions), and the discovery component returns a list of matching services, ranked according to some ranking criteria.

The Discovery front-end in *WSMO Studio* provides a simple User Interface for goal-based discovery engines. At present only the EPFL QoS-enabled Service Discovery Component [11] is supported but future versions will provide integration options with other discovery engines and matchmakers.

3.7 Integrated Validator and Reasoners

WSMO Studio provides integrated WSML-Flight and WSML-DL reasoners via the WSML2Reasoner framework⁹. The integrated reasoners can be used for validation and satisfiability tests when building WSMO ontologies. The currently supported reasoners are MINS, KAON2 and Pellet.

WSMO Studio contains an integrated WSML validator from the *wsmo4j* framework. All errors, warnings and notifications produced from the validator are listed in the standard *Problems* view of Eclipse. The information associated with each problem is: severity, explanation message and problematic location. To minimise the processing overhead, validation is performed only when a WSML file is opened or saved.

4 Evaluation and Related Tools

This section provides an evaluation of the extent to which *WSMO Studio* has achieved the defined goals and contains a brief overview of several prominent Semantic Web Services tools and their relation to *WSMO Studio*

4.1 Evaluation of WSMO Studio

With respect to the outlined goals in section 2, the following assessments can be made:

- *Functionality* – the *WSMO Studio* functionality, outlined in section 3, covers most of the basic tasks for Semantic Web Services modelling, namely support for modelling WSMO ontologies, services, goals; support for choreography

⁸ <http://www.ontotext.com/ordi/>

⁹ <http://dev1.deri.at/wsml2reasoner/>

descriptions; means for attaching semantic annotations to WSDL documents; front end to Semantic Web Service repositories and matchmakers; integrated reasoners and validators.

Other functionality, such as interaction with Semantic Web Services run-time environments (e.g. WSMX and IRS-III) or support for ontology mediation is still not available in *WSMO Studio* but is already provided by other tools (e.g. WSMT) and may be easily integrated into *WSMO Studio*.

- *Integration* – *WSMO Studio* provides functionality covering various Semantic Web Services modelling aspects in a single integrated modelling environment. Since *WSMO Studio* is based on the Eclipse platform, the functionality is provided by means of plug-ins, which can be also integrated into other products and applications.
- *Extensibility* – the Eclipse extensibility mechanism allows that functionality is added, replaced or customised by means of published extension points. *WSMO Studio* defines several such extension points, which allows customisation and replacement of specific User Interface elements, as well as adding new functionality (e.g. adaptors for new Semantic Web Services repositories or matchmakers, reasoners, etc.).

WSMO Studio extensions have already been developed by 3rd parties within the DIP and InfraWebs research projects.

An additional aspect that influences all three goals is the *flexible licensing* – *WSMO Studio* is licensed under LGPL [12], which allows that individual components (plug-ins) are incorporated within and distributed with 3rd party applications without any restrictions on the licensing terms of the latter. Our expectation is that the Open Source licence increases the visibility of a tool and the chances for 3rd party contributions and adoption, though it is not possible to provide a reliable assessment on that.

4.2 Related Tools

Current tool support in the area of Semantic Web Services is still lagging behind the theoretical advancements. Integrated toolsets, that provide functionality covering many aspects of Semantic Web Services are still not readily available and tools generally lack means of extensibility, that could improve their reusability and adoption.

This section contains brief overviews of several prominent Semantic Web Services tools related to *WSMO Studio*.

The Web Services Modelling Toolkit¹⁰ (WSMT) is an Eclipse based application which provides an advanced ontology viewer, an ontology mediation component and a WSMX management component as well as support for modelling WSMO ontologies and an integrated reasoner. Since WSMT is Eclipse based, its plug-ins can be used from within *WSMO Studio* and vice versa.

¹⁰ <http://sourceforge.net/projects/wsmt>

METEOR-S¹¹ is a Semantic Web Services toolset that provides functionality for adding semantics to web services standards like WSDL, UDDI and WS-BPEL. *Radiant* is an Eclipse based tool for adding semantic annotations to WSDL documents according to the SAWSDL and WSDL-S recommendations. *Lumina* provides an Eclipse based user interface for discovery of web services. Additionally, the METEOR-S toolset also includes a Service Discovery Engine and a semantically enhanced service registry based on UDDI.

OWL-S Editor¹² is a Protégé¹³ plug-in for modelling OWL-S services. The editor provides means for describing atomic and composite processes (i.e. the OWL-S Process definition). The OWL-S Editor also provides some limited grounding support, by generating 'skeleton' OWL-S services from WSDL files. The options for integrating new functionality and extensions of existing functionality are quite limited.

Semantic Tools for Web Services¹⁴ is a set of Eclipse plug-ins which offer two main functionalities: semantic annotation of existing WSDL documents into WSDL-S [13], and semantic matching and composition of web services (based on inferencing via the ABLE rule engine), where the queries are represented by WSDL files and the results are references to the matched WSDL service descriptions or a composition of services. The plug-ins are not extensible. STWS is not directly comparable to *WSMO Studio* since its focus is on WSDL-S based semantic annotations for WSDL.

5 Future Work

WSMO Studio has already been used as the Semantic Web Services modelling environment within two EU-funded research projects, DIP¹⁵ and InfraWebs¹⁶.

Within the scope of two active research projects, SUPER¹⁷ and Semantic-Gov¹⁸, *WSMO Studio* will be enhanced with domain specific extensions for the Business Process Modelling and eGovernment domains.

Within SUPER, *WSMO Studio* will provide functionality for modelling and querying of Semantic Business Processes using ontologies and Semantic Web Services, according to the Business Process Modelling Ontology (BPMO) [14][15]. In summary, BPMO provides an abstraction over business process modelling languages such as EPC [16] and BPMN [17], support for workflow patterns [18], and added semantics in terms of using ontology concepts for explicit modelling of the dataflow and attaching abstract WSMO goals to process tasks. The BPMO descriptions will be semi-automatically translated to sBPEL, which is the semantic version of WS-BPEL.

¹¹ <http://lsdis.cs.uga.edu/projects/meteor-s/>

¹² <http://owlseditor.semwebcentral.org/>

¹³ <http://protege.stanford.edu/>

¹⁴ <http://www.alphaworks.ibm.com/tech/wssem>

¹⁵ <http://dip.semanticweb.org>

¹⁶ <http://www.infrawebs.org>

¹⁷ <http://www.ip-super.org/>

¹⁸ <http://www.semantic-gov.org/>

Within SemanticGov, *WSMO Studio* will provide means for modelling of Public Administration services based on the Governance Enterprise Architecture (GEA) and the WSMO-PA [19][20] extension of WSMO for the Public Administration domain.

6 Conclusion

In this paper we presented a prototype of an integrated modelling environment for the Semantic Web Services domain, called *WSMO Studio*. The prototype has already been used for modelling of Semantic Web Services in several research projects.

Our future work will be focused on providing domain specific extensions for the application of WSMO in the eGovernment and Business Process Modelling domains.

References

1. Feier, C., Roman, D., Polleres, A., Domingue, J., Stollberg, M., Fensel, D.: Towards intelligent web services: Web Service Modeling Ontology (WSMO). In: In Proc. of the International Conference on Intelligent Computing (ICIC 2005), Hefei, China (2005)
2. Roman, D., Keller, U., Lausen, H., de Bruijn, J., Lara, R., Stollberg, M., Polleres, A., Feier, C., Bussler, C., Fensel, D.: Web Service Modeling Ontology. *Applied Ontology* **1** (2005) 77–106
3. Dimitrov, M., Simov, A., Momtchev, V., Ognyanov, D.: WSMO Studio - an integrated service environment for WSMO. In Bussler, C., Fensel, D., Keller, U., Sapkota, B., eds.: In Proceedings of the 2nd Workshop on WSMO Implementations (WIW 2005). Volume 134., Innsbruck, Austria (2005)
4. Rivieres, J.D., Wiegand, J.: Eclipse: A platform for integrating development tools. *IBM Systems Journal* **43** (2004)
5. Dimitrov, M., Simov, A., Momtchev, V., Konstantinov, M.: WSMO Studio Users Guide. (2006) Available online at <http://www.wsmostudio.org>.
6. Gruber, O., Hargrave, B.J., McAffer, J., Rapicault, P., Watson, T.: The Eclipse 3.0 platform: Adopting OSGi technology. *IBM Systems Journal* **44** (2005) 289–299
7. de Bruijn, J., Lausen, H., Krummenacher, R., Polleres, A., Predoiu, L., Kifer, M., Fensel, D.: D16.1: The Web Service Modeling Language WSMML. WSMML working draft, DERI (2005) <http://www.wsmo.org/TR/d16/d16.1/v0.3/>.
8. Roman, D., Scicluna, J., Fensel, D., Polleres, A., de Bruijn, J.: D14: Ontology-based choreography of WSMO services. WSMO working draft, DERI (2006) Available at online at <http://www.wsmo.org/TR/d14/v0.4/>.
9. Farrell, J., Lausen, H.: Semantic annotations for WSDL and XML Schema. W3C working draft, W3C (2006) Available online at <http://www.w3.org/TR/sawSDL/>.
10. Keller, U., Lara, R., Polleres, A., Toma, I., Kifer, M., Fensel, D.: D5.1: WSMO web service discovery. WSMO working draft, DERI (2004) Available online at <http://www.wsmo.org/TR/d5/d5.1/v0.1/>.
11. Vu, L., Hauswirth, M., Porto, F., Aberer, K.: A search engine for QoS-enabled discovery of Semantic Web Services. *International Journal of Business Process Integration and Management (IJBPM)* (2006)

12. Free Software Foundation: GNU Lesser General Public License, version 2.1 (1999) Available at <http://www.opensource.org/licenses/lgpl-license.php>.
13. Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.T., Sheth, A., Verma, K.: Web service semantics – WSDL-S. W3C member submission, W3C (2005) Available at <http://www.w3.org/Submission/WSDL-S/>.
14. 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, Beijing, China (2005) 535–540
15. Hepp, M., Belecheanu, R., Domingue, J., Filipowska, A., Kaczmarek, M., Kaczmarek, T., Nitzsche, J., Norton, B., Pedrinaci, C., Roman, D., Stein, S.: Business process modelling ontology and mapping to WSMO. SUPER technical report, Project IST-026850 SUPER (2006)
16. Mendling, J., Neumann, G., Nüttgens, M.: Towards workflow pattern support of Event-Driven Process Chains (EPC). In: 2nd GI Workshop XML4BPM - XML for Business Process Management, BTW 2005, Karlsruhe, Germany (2005) 23–38
17. Object Management Group: Business process modeling notation specification (BPMN). Technical report, Object Management Group (2006) Available at <http://www.bpmn.org>.
18. Aalst, W.M.P.V.D., Hofstede, A.H.M.T., Kiepuszewski, B., Barros, A.P.: Workflow patterns. *Distributed and Parallel Databases* **14** (2003) 5–51
19. Wang, X., Vitvar, T., Peristeras, V., Mocan, A., Goudos, S., Tarabanis, K.: WSMO-PA: Formal specification of public administration service model on semantic web service ontology. In: Proceedings of the Hawaii International Conference on System Sciences (HICSS), Waikoloa, Big Island, Hawaii (2007)
20. Peristeras, V., Mocan, A., Vitvar, T., Nazir, S., Goudos, S., Tarabanis, K.: Towards semantic web services for public administration based on the web service modeling ontology (WSMO) and the governance enterprise architecture (GEA),. In: Proceedings of the DEXA, Krakow, Poland (2006)