Triple Space Computing for Semantic Web Services – A PhD roadmap

M. Omair Shafiq
Digital Enterprise Research Institute (DERI),
University of Innsbruck (UIBK)
6020 Innsbruck, Austria.
omair.shafiq@deri.org

Abstract. This thesis will address how to enable Triple Space Computing as a communication paradigm for Semantic Web Services. Currently, Semantic Web Services are following a message based communication paradigm. Triple Space Computing is envisioned as communication and coordination paradigm for Semantic Web Services which is an extension of tuple space computing to support RDF and then use it for communication based on the principle of persistent publication and read of data. Web Service Modeling Ontology (WSMO) is our conceptual model for Semantic Web Services. Web Service Execution Environment (WSMX) is one of the reference implementations of the WSMO conceptual model. The paper presents an overview of technical insights about integration of WSMX with Triple Space Computing and proposes that how WSMX can use Triple Space computing for its communication and coordination in terms of dynamic components management, external communication management, resource management and coordination of different interconnected WSMXs.

1 A roadmap to enable Triple Space Computing in Semantic Web Services

The communication paradigm in Semantic Web Services (SWS) is synchronous, i.e. users communicate with SWS and SWS communicate with real world Web Services synchronously by sending direct messages. The synchronous communication requires quick response as its makes sender halt until response is received, which is not possible in case of execution process in SWS as it involves a heavy processing of semantic descriptions in terms of discovery, selection, composition, mediation, execution. This problem is to be overcome by introducing asynchronous communication and Triple Space Computing is perfect solution as being semantic based asynchronous communication paradigm. In this thesis we take Web Service Execution Environment (WSMX) [2] as reference implementation of Web Service Modeling Ontology (WSMO) to concretely solve the concerned issues. Using Triple Space Computing for asynchronous communication between different WSMXs enables and brings them a step closer to their architectural goal, i.e. to support greater modularization, flexibility and decoupling in communication of different WSMX nodes. Similarly, it enables WSMX to be highly distributed and easily accessible.

This thesis will address the integration of Triple Space Computing [1] with WSMX by analyzing that how and where exactly the two technologies fit together. The integration has been proposed as three major entry points which are (1) enabling components management in WSMX using Triple Space Computing, (2) External communication grounding in WSMX using Triple Space Computing, (3) Resource Management in WSMX using Triple Space Computing and (4) enabling communication of different inter-connected WSMX. After achieving these integration aspects, the goal will be then to build an application scenario to show its viability. Each of the integration aspect has been described in details below:

WSMX has a management component [6] that manages the over all execution by enabling coordination of different components based on some execution scenario [4] specified by user in

Goal. In this way there is a clear separation between business and management logic in WSMX. The individual components have clearly defined interfaces and have component implementation well separated with communication issues. Each component in WSMX have wrapper to handle the communication. WSMX manager and the individual components wrappers are needed to be interfaced with Triple Space in order to enable the WSMX manager manage the coordination of the components over Triple Space. The communication between manager and wrappers of the components will be carried out by publishing and subscribing the data as a set of RDF triples over Triple Space. The wrappers of components that handle communication will be interfaced with Triple Space middleware.

WSMX acts as a semantic middleware between users and real world web services. Currently, due to existence of message oriented communication paradigm, users communicate with WSMX and WSMX communicate with Web Services synchronously. The external communication manager of WSMX is needed to provide a support to communicate over Triple Space. The interfaces for sending and receiving external messages by WSMX are needed provide a grounding support to alternatively communicate over Triple Space. This needs to be resolved by addressing several issues, i.e. invoker component in WSMX is needed to support Web Services Description Language (WSDL) and Simple Object Access Protocol (SOAP) communication binding over Triple Space. The Entry point interfaces will be interfaced with Triple Space middleware in order to provide the glue between existing Web Services standards and Triple Space Computing.

WSMX contains different repositories to store ontologies, goals, mediators and web services descriptions as WSML based files. The internal repositories of WSMX are needed to be made optional and enable to store the WSML based data as set of RDF named graphs in Triple Space Storage. This is mainly concerned with transforming the existing representation of data in form of WSML into RDF representation. The repository interfaces are needed to be interfaced with Triple Space middleware.

After enabling WSMX with Triple Space Computing, the next step will be to enable the communication and coordination of different WSMXs over Triple Space, i.e. forming a cluster of different interconnected WSMX nodes to support distributed service discovery, selection, composition, mediation, invocation etc. The management component in WSMX is will be enhanced to coordinate with WSMX managers in other WSMXs over Triple Space to form a cluster. After the implementation of integration of Triple Space Computing in WSMX, an application scenario will be analyzed, designed and developed over WSMX to show the usefulness of the new communication paradigm in WSMX. It will be a travel agent based application that will use semantic descriptions of several real life Web Services, like Amazon, Google, currency converter, money transfer etc. and would require extensive and distributed discovery, selection, composition, mediation and invocation of Semantic Web Services to fulfill the user's requirements.

Acknowledgements: Author acknowledges the guidance support from Michal Zaremba being the chief architect of Web Services Execution Environment (WSMX), Dieter Fensel being the initiator of WSMX and Triple Space Computing and Ying Ding for reviews, suggestions and further guidelines.

References

1. D. Fensel, Triple-space computing: Semantic Web Services based on persistent publication of informatio: In Proceedings of the IFIP International Conference on Intelligence in Communication Systems, INTELLCOMM 2004, Bangkok, Thailand, November 23-26, 2004.
2. C. Bussler et al, Web Service Execution Environment (WSMX), W3C Member Submis-sion, June 2005. Available at http://www.w3.org/Submission/WSMX
3. Michal Zaremba, Matthew Moran, Thomas Haselwanter, WSMX Architecture, D13.4v0.2 WSMX Working Draft.
4. M. Zaremba, C. Bussler: Towards Dynamic Execution Semantics in Semantic Web Services, In Proceedings of the Workshop on Web Service Semantics: Towards Dynamic Business Integration, International Conference on the World Wide Web (WWW2005). Chiba, Japan, 2005.
5. R. Krummenacher, M. Hepp, A. Polleres, C. Bussler, and D. Fensel: WWW or What Is Wrong with Web Services. In Proc. of the 2005 IEEE European Conf on Web Services (ECOWS 2005), Växjö, Sweden, November 14-16, 2005.
6. T. Haselwanter, Maciej Zaremba and Michal Zaremba. Enabling Components Management and Dynamic Execution Semantic in WSMX. WSMO Implementation Workshop 2005 (WIW 2005), 6-7 June, Innsbruck, Austria.