



# D23v1. WSMX System Functionality Scope

WSMO Working Draft 26 November 2004

**This version:**

<http://www.wsmo.org/2004/d23/v1/20041126>

**Latest version:**

<http://www.wsmo.org/2004/d23/v1/>

**Previous version:**

<http://www.wsmo.org/2004/d23/v1/20041012/>

**Editor:**

Michal Zaremba

This document is also available in non-normative [PDF](#) version.

---

## 1 Purpose of the Document

Web Services Execution Environment (WSMX) is an execution environment enabling dynamic discovery, selection, mediation and invocation of Semantic Web Services (SWS). WSMX is built on Web Services Modeling Ontology (WSMO), which describes various aspects related to SWS. This document aims to provide an overview of expected functionality the system is going to provide with each subsequent release of the software. The reader should refer for a more details about conceptual model [Cimpian et al., 2004], execution semantic [Oren, 2004] and architecture [Zaremba et al., 2004] of the system to appropriate documents. By making this document public, developers of WSMX commit themselves to continuing progress of WSMX platform, informing general public on progress of work and future directions for the software. WSMX is an open source project hosted on the sourceforge and all the sources of the system can be accessed directly from the sourceforge website at <http://sourceforge.net/projects/wsmx>

The WSMX working group is open for any new members to develop conceptual basis of the system. We also welcome software developers, who would like to participate in this project and shape with us the future releases of WSMX. Anybody interested in joining our initiative should provide rationale to [Michal Zaremba](#).

Anybody from general public interested in particular functionality to be included in WSMX is advised to log a request with [Michal Zaremba](#) so his/her request can be taken into consideration when defining the next version of "WSMX Functionality Scope" document.

## 2 WSMX Functionality Scope

Developers of WSMX system agreed to deliver three to four official releases of the system every year. Each of the following subsection focus on one of the releases. We provide list of components we are going to work on and expected functionality to be delivered with each of these components. We also provide a list of names of people contributing to particular functionality. Each of the developer can be contacted directly to obtain more details on the progress of work. We also define a timeline for the release and include a list of dependend documents.

### 2.1 Second WSMX release

Deadline for the second release of WSMX has been initially set for the end of November of 2004. Carrying our work we decided to have fainally two deadlines: one at the end of November to release available components and the second release on the beginning of January as a complete set of cooperating components. At the time of writing of this document following functionality was exepcted to be available for the second release of the system.

#### 2.1 Components

Component Name	Main Developer	Contributors	Expected Functionality	Delivered	Expected till November 30th

Core WSMX Engine/ Server Engine/ Events Scanner/ Message Scanner/ Listeners Infrastructure	<b>Michal</b>	<b>Thomas, Maciej</b>	Microkernel architecture based on JMX technology with components which can be plugged at runtime	Multithreaded, event based system	Building a framework enabling components connectivity. Decoupling components one from each other ( <b>shifted till beginning of January 2005</b> )
OOMediator and Process Mediator	<b>Adrian</b>		Complete infrastructure enabling data and process mediation	Basic system capable to carry data mediation; mapping rules design tool	First basic implementation for the process mediation component. Enhancements of the data mediation component - interface of ooMapper (design time component); provide more formal mappings representation, either by using an existing mapping language or by formalizing our own.

Invoker/ Receiver/ Grounding	Matt	Paco	Carrying all the communication with external WSMX entities	RPC style invocation	<p>First simple implementation of grounding for WSMO. Additionally to invoker (sender) we will provide receiver functionality to handle any data returned by a Web Service. Invoker will support RPC-Style and document literal style (at least semi-automatically). Introduction of WSIF to the invoker component.</p>
TSpace			Triple Space Computing mechanism enabling applications to communicate by writing and reading RDF triples in a shared space		<p><b>Dropped:</b> Tuple space storage, which will be used to carry conversation with temporary unavailable Web Services. This component will be capable to store and retrieve messages from a tuple space, which should be normally handled between WSMX and Web Services .</p>

Parser/ Compiler	<b>Eyal</b>		WSML definitions validation. Generating object representation of WSML documents	Parsing available for an old version of the language. Does not point syntax errors	Parser will be an independent tool, which can be run on its own, capable to parse a WSML definitions, point to syntax errors and return back a class model in case of successful compilation.
Choreography	<b>Armin</b>	<b>Emilia</b>	Execution of any choreography described with choreography language		First cut implementation for Choreography component - some hard coded choreographies will be available in the system. Choreography component must be able to cooperate with invoker component. Emilia will address processes inconsistencies.

Resource Manager	Manuel	Michal	Abstraction layer between resources and the system and components	Interfaces defined; JDBC calls to the database implemented	Redesign and enhance the data management layer in. The new layer should minimize (further) the impact of datamodel changes on other components. Part of this is investigating the use of an EJB or XML database instead of relational database. This should remain transparent to the other components, i. e., interfaces are defined using java classes not xml documents.
					Design general architecture: Backend application + RAF + Connectivity to WSMX. Create a BA client version as a Web Service (one that could be addressed on the network) capable to provides

<p>Adapter/Back-End Application Integration</p>	<p>Laurentiu</p>	<p>Edward, David, Brahmananda</p>	<p>Complete framework for JCA</p>	<p>One adapter hardcoded</p>	<p>history log of exchanged messages. Build the BA connectivity with RAF capable to send and receive messages to and from RAF. Provide framework for connecting/disconnecting new adapters. Build RAF interface towards WSMX (methods to invoke WSMX, to send and receive messages towards WSMX, ability to instantiate choreography component). Implementation of asynchronous communication for sending and receiving messages to and from WSMX; coordinate with the choreography implementation.</p>
---	------------------	-----------------------------------	-----------------------------------	------------------------------	---

Discovery	<b>Ioan</b>		Complete framework enabling discovery of SWS and Ontologies	SWS stored in database. Discovery limited to query database	Distributed and keyword enabled discovery
Matchmaker			Matching Web Services with Goals (might be redundant in the future as this functionality might be delivered by discovery)		
Security	<b>Michal</b>		Handling authentication, authorisation and all other security aspects related to service requesters, service providers and components		
Error Handler	<b>Michal</b>		Handling any component and invocation related exceptions and errors	Capable to record some of the exceptional situations in permanent storage. Basic recovery mechanism in place.	
WSMO Editor					
WSMX Monitoring					

Additionally to functionality presented in the table above we are going to work on

several Use Cases to achieve functionality requested by various partners.

## 2.2 Timeline

Component	22 Oct	29 Oct	5 Nov	12 Nov	19 Nov	26 Nov	4 Dec (November release)	15 Dec	7 Jan (January release)
Core WSMX Engine/ Server Engine/Events Scanner/Message Scanner/Listeners Infrastructure									
OOMediator and Process Mediator									
Invoker/Receiver/ Grounding									
TSpace									
Parser/Compiler									
Choreography									
Resource Manager									
Adapter/Back-End Application Integration									
Discovery									
Matchmaker									
Security									
Error Handler									
WSMO Editor									
WSMX Monitoring									

**Legend:**

Start of work



Interfaces complete



Stand-alone implementation



Integrated implementation



Final working version of component

## 2.3 Dependencies

This section lists documents on which the second release of WSMX is dependent on:

- Conceptual model, class model and data model of the system are based on Web Service Modeling Ontology (WSMO). Interfaces of WSMX components are going to be dependent on WSMO API, which will be implemented for version 1.0 of WSMO - <http://www.wsmo.org/2004/d2/v1.0/20040920/>
- The recent version of architecture document for WSMX will constitute the basis for implementation of WSMX system - <http://www.wsmo.org/2004/d13/d13.4/v0.1/20040622/>. Additionally to this document the updated architecture diagram is currently available at <http://www.wsmo.org/2004/d13/d13.4/v0.2/20041012/>.
- Although WSMX v0.2 is going to be the system of several executions semantics, the base execution semantics document remains - <http://www.wsmo.org/2004/d13/d13.2/v0.1/20040531/>
- Details of data mediation as they will be delivered with WSMX v0.2 is covered in <http://www.wsmo.org/2004/d13/d13.3/v0.1/20040906>

## 3 References

**[Cimpian et al., 2004]** Cimpian E., Mocan A., Moran M., Oren E., Zaremba M. (2004). WSMX Conceptual Model. WSMO Working Draft v0.1, Digital Enterprise Research Institute (DERI), available from <http://www.wsmo.org/2004/d13/d13.1/v0.1>

**[Oren, 2004]** E. Oren. WSMX Execution Semantics, WSMO Working Draft v0.1, 31 May 2004, Digital Enterprise Research Institute, available from <http://www.wsmo.org/d13/d13.2/v0.1>

**[Zaremba et al., 2004]** M. Zaremba, M. Moran, E. Oren, E. Cimpian, A. Mocan, WSMX Architecture, WSMO Working Draft v0.1, 30 May 2004, Digital Enterprise Research Institute, available from <http://www.wsmo.org/2004/d13/d13.4/v0.1>

## Acknowledgement

The work is funded by the European Commission under the projects [DIP](#), [Knowledge Web](#), [SEKT](#), [SWWS](#), and [Esperanto](#); by [Science Foundation Ireland](#) under the [DERI-Lion](#) project; and by the Vienna city government under the [CoOperate](#) program.

The editors would like to thank to all the members of the [WSMO](#), [WSML](#), and [WSMX](#) working groups for their advice and input into this document.