SEMTOUR-Studio: A Semantic Web Services Creation Tool for the Tourism Sector.

F.J. Lacueva (fjlacueva@ita.es), M.A. Barcelona (mabarcelona@ita.es), L.Garcia (lgarcia@ita.es), E.Mendoza (emendoza@ita.es), J. Lalana (jlalana@ita.es), J. Vea-Murguia (jvea@ita.es)

Instituto Tecnológico de Aragón, María de Luna 7-8, 50018, Zaragoza, Spain

Abstract. SEMTOUR-Studio is a semantic Web Services composition editor developed to allow the population of the SEMTOUR Tourism Web Services Platform (STWSP). On the one side SEMTOUR Studio allows to deploy basic Web Services within the STWSP: already existing services in external systems can be "normalized" (semantically annotated, grounded and BPEL wrapped), in order to make them deployable and consumable within the STWSP. On the other hand it allows creating new virtual services by the composition of already deployed services.

Keywords: SWS, SAWSDL, BPEL, WSMO-Lite, OWL, XSLT, Grounding, SEMTOUR, Service Composition, e-Tourism, Virtual Enterprises.

1 Introduction.

The SEMTOUR project [1] took the results of the Composetour [2] project to create a Tourism Value-Added Web Services Platform. The platform has three main components:

1. The Community of Service Users: travelers who want to make a trip, who are registered with in the community and who can give their opinion about the consumed product and the Web Services used to contract the product.
2. The Semantic Web Service Platform, where services are deployed, executed, monitored and evaluated, both by the user and by the Platform (liability, response time, …)
3. The SEMTOUR-Studio, which is introduced on this paper, a semantic Web Services composition editor developed to allow the population of the SEMTOUR Tourism Web Services Platform (STWSP).

Fig 1 introduces the system architecture of the SEMTOUR Platform. The main goals of the SEMTOUR project are:

- Populating the infrastructure with touristic semantic Web Services: already existing services which are normalized, that is semantically annotated, grounded and BPEL wrapped, in order to be consumed within the SEMTOUR Platform.

- Creating and establishing the community of users for the SEMTOUR Platform which evaluates and grants reputation about the consumed products and services.
- Creating an algorithm for service discovery based on the reputation of the services (or of the products they provide) and on its QoS.
- Making the creation easier of “virtual” travel agencies by the composition of “SEMTOUR Services”.

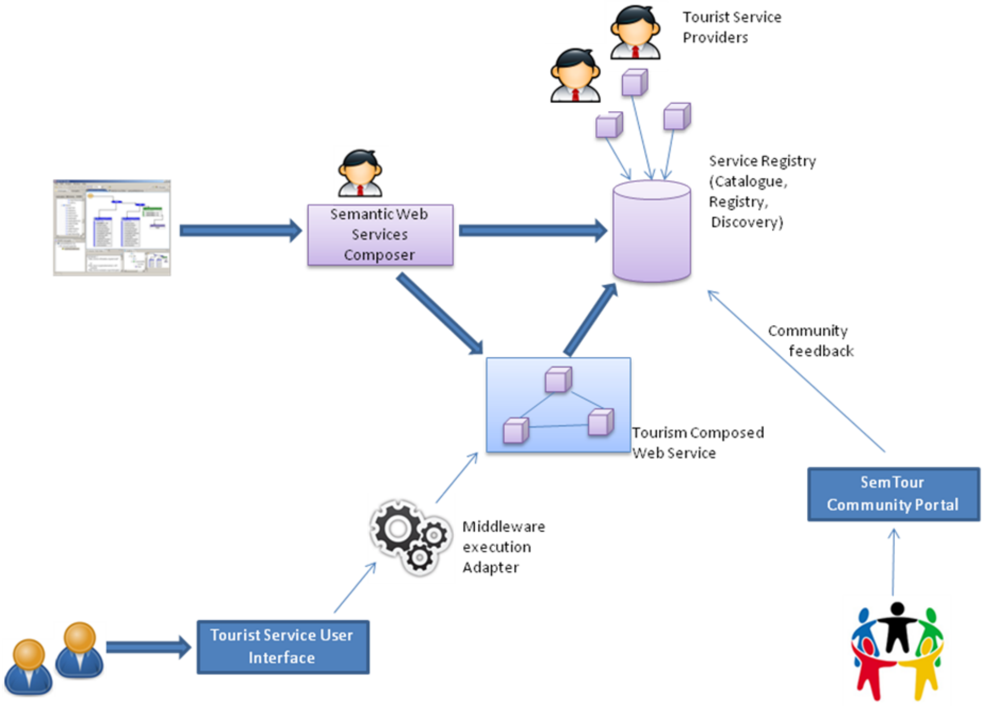
Tourist Service Providers
Service Registry (Catalogue, Registry, Discovery)
Semantic Web Services Composer
Community feedback
Tourism Composed Web Service
SemTour Community Portal
Middleware execution Adapter
Tourist Service User Interface

Fig. 1. Conceptual System Architecture of SEMTOUR.

As we previously mentioned, this paper introduces SEMTOUR Studio, a semantic Web Services composition editor developed to allow the population of the STWSP. On the one side SEMTOUR Studio allows to deploy basic Web Services within the STWSP: already existing services in external systems can be semantically annotated, grounded and BPEL wrapped, in order to make them deployable and consumable within the STWSP in a standard way, such as what we call SEMTOUR Services.
On the other hand, the SEMTOUR Studio allows the creation of new virtual services by the composition of already deployed services. It allows the selection of the existing services both by their semantic description and by their reputation and the reputation of the products they provide.

The rest of the paper is dedicated to summarize our work within the project. It starts by the definition of the requirements of the ECSWS, it follows with the definition of its architecture and the interfaces with the SEMTOUR Platform and it concludes with the implementation and test of the solution.

2 SEMTOUR Studio Requirements.

We briefly introduce here the Requirements of the ECSWS, which are gathered in the deliverable E3.2 [3] of the SEMTOUR project. This deliverable was approved by the project consortium after their revision and comments inclusion.

As we already mentioned in the introduction paragraphs, main objectives of the ECSWS are to allow the population of the SEMTOUR Tourist Web Services Platform (SWSP) and the creation of new services by composing SEMTOUR Services. Both kinds of services, either Basic Services or Composed Services, have to be described by (annotated against) a WSMO-Lite [4] ontology and their grounding processes have to reference a community shared ontology and - virtualized by a BPEL [5] process.

Additional high level requirements are: integration with the platform security in order to avoid that anyone could publish, consume or create SEMTOUR Services; and the possibility of filter discovered services by both functional and not functional (reputation, QoS) criteria.

From these high level requirements we refined the functional and non functional as are summarized in next paragraphs.

2.1 Functional Requirements.

Next table summarizes the functional requirements of our tool which are more important for the aim of this paper.

FR id.	Requirement
FR0003	High Level Functionalities of the ECSWS would be: a WS annotation tool for annotating WS, a tool to access to the SWS repository; a tool to access to the Ontology Repository; an ontology visualization; a service composition.
FR0005	“Files” to define the different elements to use should be based on XML derived languages such as OWL, WSMO-Lite, BPEL or XLST
FR0012	The user should be able to obtain the ontology definition, the security and reputation policies and the services deployed in the community.
FR0018	The local WSDL files that syntactically describe the WS should be annotated against the SEMTOUR community ontology.
FR0022	Each of the WS will produce a Basic Business Process
FR0023	Business Processes also can be created as the composition of services already deployed in the community.
FR0025	The data to use within the composed BPs should be concordant with the Domain ontology.
FR0027	The code to be generated should include all the information necessary to fulfill the security policies, execution log, …
FR0030	The composed BPs should also be annotated against the community ontology. As far as possible it should be performed automatically.

FR0031	In order to allow the consumption of the SEMTOUR community services from outside the community, their WSDL files should be made public available.

Table 1. Relevant Functional Requirements.

2.2 Non Functional Requirements.

Next table summarizes the Non Functional Requirements of the ECSWS.

NFR id.	Requirement
NR0001	WSMO-Lite will be used to annotate SWS
NR0002	The SEMTOUR Studio should be able to be used in different hardware and software platform.

Table 2. Relevant non Functional Requirements.

3 Another Editors Initiatives.

Semantic Web Services, SOA, (Dynamic) Process Compositions and BPM are going to be the research items which will be the foundations of the future enterprises. As a consequence; there have been a lot of initiatives which worked in trying to put all together to work in the achievement of the Future Internet realization.

Before start working in the SEMTOUR Studio we perform a state of the art of similar tools which already exists or which are under development. WSMO Studio [6], INFRAWEBS [11, 12], SOA4ALL [9] and SUPER [10] are examples of the studied tools.

Some of their libraries and tools are being used and adapted for implementing our ECWSW. The details about the modifications or additions included by us will be explained in the section 5 of this paper.

4 SEMTOUR-Studio Architecture.

Once we defined the requirements of the ECSWS we proceed to design it by defining its architecture. On the one hand it has to fit the defined requirements; on the other hand it has to correctly assemble with the other components of the SEMTOUR Platform architecture [14]. To get it we proceed to define a top-down module definition as it figures on deliverable PT3.E2 of the SEMTOUR project [13]. We briefly introduce the architecture and the use cases to be implemented on the next paragraphs.

4.1 Use Case to Be Implemented.

In [15] Papazoglou classified Web Services in Basic Services and Composite Services, in consequence we decided to implement two use cases: the Basic Service annotation, grounding and "BPELLING" processes and the Definition of Composed Services (by SEMTOUR deployed services aggregation).

We introduce both use cases in next paragraphs and they are described on Figure 2 included under the "Create BP" use case.

Deployment of Basic Services.

In order to deploy an existing Web Service within the SEMTOUR community it has to be annotated, grounded and coded ("BPELLED"). These activities are included in the "Create Basic BP" of the previous diagram.

In [16] they define the way in which WSDL files [17] describing Web Services should be annotated using the SAWSDL [18] extension. From [16] it can follow that the semantic description of an existing Web Service consists of two steps (which have not necessarily to be performed in this order):

- The WSDL file is annotated against the ontology (the domain + WSMO-Lite ontology) in order to classify the operations, kinds of messages and data types. The resulting file is an SAWSDL file.
- The XML schemas of the original WSDL file are used to create the Grounding transformation files [22] with respect to the community shared ontology: Lowering when data are used to invoke a service operation and Lifting for the returned data.

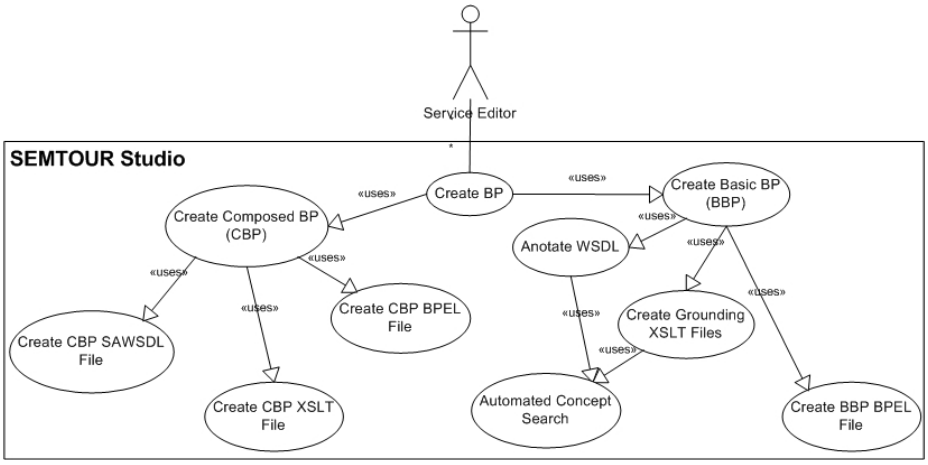

Fig. 2. SEMTOUR Studio use cases.

As the resulting XSLT files of this processes has to be referenced by the resulting SAWSDL file of the previous step, it has no sense to determine in which order it has to be performed. But, if some automatic annotation would be possible, as we explain after, perhaps we can consider that the presented order may be "more" suitable.

The final step is the creation of the code to be used within the community in order to consume an external existing WS. In our case, we use BPEL as the Business Process language [23]. The BPEL capability of subscribing to different kind of messages is used in order to give access to the different operations a WSDL file may content. The generation process of the BPEL code automatically includes the invocation to the original Web Service operation and the invocation to the transformation processes based on the XSLT files generated on the Grounding step.

Proceeding on this way, every Basic Service deployed in the community, that is a SEMTOUR Basic Service, will be always accessed in a standardized way. To get it some automatic changes have to be performed in the SAWSDL file in order to operate in a correct way:

- The XML schemas of the type definition have to be changed to the ones representing the ontology. The grounding processes transform from the ontology data representation to the original service data representation, the XML schemas defined in the original service WSDL file (Lowering), and vice versa (Lifting). In consequence, the input and output data of the exposed operations to the SEMTOUR community fulfill the XML schema representing the ontology.
- The endpoints of the original services have to be changed to the ones through the SEMTOUR Platform is going to offer the SEMTOUR service wrapping the original service. When we are creating a Basic Service, one of the processes is the creation of a BPEL file. This file has to be executed within in the community by a BPEL engine which will make the operations accessible in a proprietary endpoint. This is the endpoint which should appear within the SAWSDL file describing the service to the community.

Although the first change can be performed either by the editor or by the community platform, the endpoint change can only be done once the BPEL is deployed in the execution engine that is when the SEMTOUR service endpoint is known. In consequence we consider that it is better to be implemented by the SEMTOUR Platform.

Composed Services Development/Deploying.

Composed SEMTOUR services are created by the aggregation/orchestration of services which have already been deployed in the SEMTOUR community. SEMTOUR services are, at least from the input and output data point of view, standardized processes: every service deployed within the SEMTOUR Platform should have inputs and outputs which are instances of the SEMTOUR ontology. That will help in some aspects of the creation of the SEMTOUR composed services.

In the first place, as all the data of the input and output messages of the operations of a SEMTOUR service must be instances of the domain ontology, when we want to invoke an operation of a SEMTOUR service, it will not need to make any transformation, in consequence grounding transformations would not be needed. Even that, and in order to make everything as standardized as possible, when we create Composed SEMTOUR Services we decided to invoke the identity Lowering and Lifting functions.

In the second place, the automation can be applied to the BPEL generation too. If the SAWSDL file of a given SEMTOUR service would contain the choreography of the original service, for example using the WS-DCL [21] as suggested [22], it will not be very difficult to create the orchestration of their operations. Even that, our first approach to the implementation of the composed services is to detect the matching between the output and the inputs of the operations of a given SEMTOUR service. This matching process also allows us to detect the order of the services under composition.

With this approach our intention is to start working in the integration of the SEMTOUR Studio with the SEMTOUR Platform. We are aware of the limitations of this approach but it will allow us to have a look into the problems we need to solve in order to advance on the road to the dynamic service composition. Moreover, even if the choreography of the services is already established, the editor should consider the user interaction in order to give him the opportunity to create its own service, for example, to include the invocation to log services.

In order to make the composed services consumable they have to be published by creating its SAWSDL file and deploying it together with the BPEL file and the grounding files, in the same way as a Basic Service is published. Again, the creation of the SAWSDL file can be done automatically. The composed service is going to have just one operation, resulting from the orchestration of the operations of the used/included SEMTOUR services. For this operation, the input and output messages can be obtained from the input of the first invoked operation and the output of the last one.

The user interaction will be only needed to determine the name of the service and to annotate the elements of the WSDL describing the composed service. If we try to name or to annotate the new service/operation automatically we can consider two ways to do it. The first one is to annotate the elements of the composed service WSDL with the annotations of the correspondent WSDL elements of all the source services. The second one is by trying to make it automatically by searching the matching between the name of the SAWSDL elements and the ontology concepts just as we do with during the Basic Service annotation and grounding definition process. Even that, from our point of view, as it is not possible to build a complete ontology dictionary, the interaction with the user is necessary, although it can be reduced to the minimum.

Finally, as it happens with the Basic Service, the endpoint should be changed before the new service could be consumed within the community, that is, during the deployment process.

4.2 ECSWS High Level Architecture.

SEMTOUR Studio is the interface of the SEMTOUR Platform to the actual services owners. As we already mentioned in previous paragraphs its objective is to allow non technical staff to share their business services (Basic Services) with the SEMTOUR community and to create new business process from the composition of already

deployed SEMTOUR services composing new business processes using already deployed services.
Figure 3 shows the architectural definition of the SEMTOUR Studio Editor and the relation of each of its component with the rest of the components of the SEMTOUR Platform. We briefly introduce these components:

- The concerns of the ST_ECSWS_SemanticManager are the manual and semi-automatic semantic annotation of the Web Service description (WSDL) against the Tourism domain ontology and the WSMO-Lite ontology. As a consequence it has to interact with the community to obtain the OWL ontology definition and the set of ontology concepts in order to get the service semi-automatically annotate.
- Once the services has been semantically annotated, the next step is to standardize the data they exchange with the rest of the community, that is to define their Lowering and Lifting (XSLT) mapping processes. The ST_ECSWS_GroundingEditor is created to do it. Although some automatic process can be performed in the case of the Grounding it has to be mainly performed manually.
- From the development team point of view every "executable" element should have to be deployed in the SEMTOUR Platform in a standard way. All the services, either basic or composed, have three elements: one SAWSDL file containing their semantic description in terms of the community ontology; one grounding file containing the Lifting and Lowering XSLT schemas to transform their data; and one BPEL file containing the code to access to the actual services. Obviously the ST_ECSWS_SemanticBPEditor is in charge to create automatically the BPEL wrapper of the Basic Services, and to allow users to create composed services by glueing already deployed services.
- Finally the ST_ECSWS_LocalRepositoryManager is in charge to hide the exchange of information between the computer in which the ECSWS is being executed and the community repositories which are represented by the ST_ECSWS_RemoteRepositoryManager in which services are deployed.

Although most of the interactions of the editor's components are designed to be done through the repository manager's components, some of them can be done directly against some of the SEMTOUR platform modules. For example, the ST_ECSWS_SemanticBPEditor allows the validation of the code of a composed service under development against the ST_ECWSWS_ExecutionEngine in order to avoid as many errors as possible, before the service deployment.

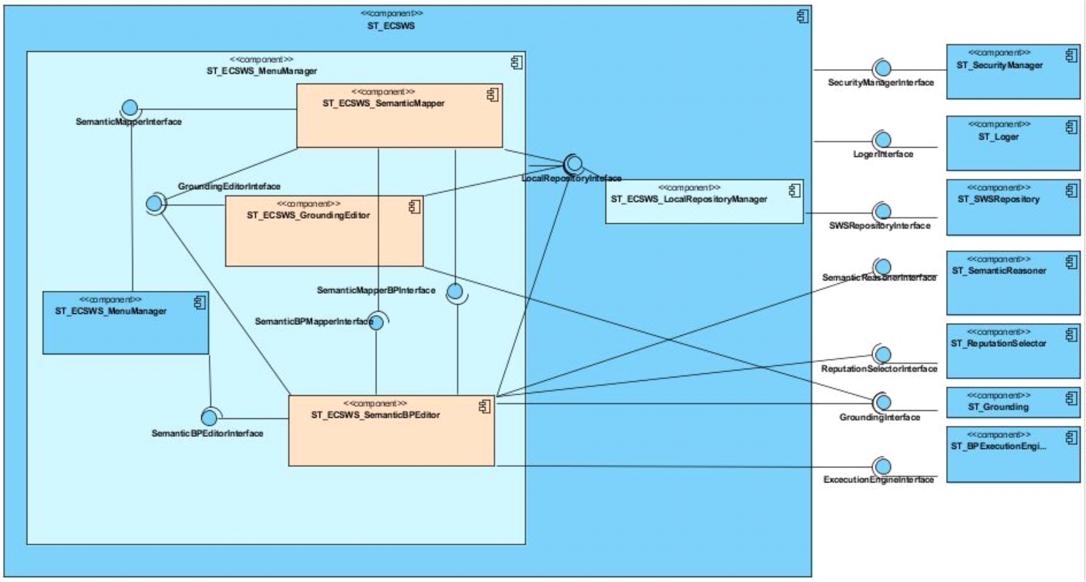

Fig. 3. SEMTOUR Studio Architecture.

5 ECSWS Development State.

By the time this paper is being written we are still developing the ECSWS. It is going to be released by July of 2011 in order to be tested and integrated with the rest of the SEMTOUR Platform. From the previous components diagram, we are currently working in the Grounding Editor and in the Semantic Business Process Editor. Next paragraphs summarize the adaptation of the Open Source Modules we are integrating and some limitations related to the implementation we assumed because of the scope of the project.

At the moment, apart from the integration test to be performed with the two other components presented in these paragraphs, ST_ECSWS_SemanticManager component is already finished and integrated within the ST_ECSWS_MenuManager. The other two components are still under development. The ST_ECSWS_GroundingEditor is close to be finished while the ST_ECSWS_SemanticBPEditor has already passed the Ecuador of its implementation: the Basic SEMTOUR Services is already released while the Composed Services are still under development.

5.1 ST_ECSWS_SemanticManager Development State.

The ST_ECSWS_SemanticManager has been already released. It has adopted some components from WSMO Studio modeling environment, modifying them in order to correctly generate SAWSDL files annotated against WSMO-Lite ontology instead of against WSMO ontology. We have added an automatic annotation process based on the capability to associate synonyms to ontology concepts which was introduced by Bonino et col. in [23, 24, 25, 26].

We adapt some of the DOSE [25] libraries in order to allow the automatic annotation of a WSDL file. We developed an OSGI bundle, just another kind of software services prepared to be deployed on an OSGI runtime framework. Between other, the bundle has an operation which receives two input parameters: the first parameter is the URI of a file containing the concepts of the ontology together with their synonyms (which can be expressed in different languages); the second parameter is the word for which the meaning within the SEMTOUR community is search. If the meaning of the word is found (it exists a concept within the ontology having the searched word in its synonym list) then the URI of the concept representing the meaning of the word is returned in order to be used to annotate the WSDL file.

We use this bundle for trying to reduce the manual annotation of the WSDL to the minimum. Our method proceed on this way: we take the names of the operations, services and datatypes elements of the original WSDL file and we iterative invoke the bundle operation parameterized with the element name together with the file of synonyms of the ontology. For each of the names of the WSDL elements, if an URI is returned, we include it within the element annotation of the SAWSDL file under creation.

Although this process can help to reduce the annotations to be written by hand, the quality of the results is close related to the synonyms file content and, obviously the human interaction will be needed to revise the annotations and to add some references which cannot be automatically done. For example, consider the different ways to write the concept "room booking". It is easy to imagine this concept write like: "roombooking", "room_booking", "reserva de habitación" or "reservahabitacion"… within WSDL files describing services or operations for booking a room. What is not obvious but is possible is that "RB", "xy" or "reserva" could have the meaning of "room booking" within a WSDL file.

We want to mention that in the development of the Semantic components we have used the easyWSDL and easySAWSDL libraries in order to create and validate the different WSDL files [27]. Both, together with other useful libraries, have been developed under the SOA4ALL project [9].

5.2 ST_ECSWS_GroundingEditor

As we have already mentioned, we try to automatically annotate the data types. It is just a part of the work it has to be performed in order to make WSDL data and schemas semantically available for the SEMTOUR community. The second part is the creation of the Grounding files and their references in the SAWSDL file. As we introduced in paragraph 4.2, it is going to be implemented by the ST_ECSWS_GroundingEditor. Once the SAWSDL has been annotated we proceed to create the Lowering and Lifting files. The matching between the XML schemas has to be performed by hand. The results of the matching processes are the XSLT files which are going to be used to transform the XML inputs and outputs of the semantic SEMTOUR Services or their operations to the inputs and outputs data formats of the original services.

In order to make the implementation of the matching process easier, we automatically create a XML Schema Definition (XSD) file containing a data structure definition equivalent to the instance structure of the OWL ontology. Proceeding on this way, the creation of the grounding files is reduced to the problem of creating XSLT files which transforms XML fulfilling an XSD, to XML data fulfilling another XSD. Obviously, we create different files to transform the input (Lowering) and output (Lifting) data. We use the OWL2XSD classes of the OWL2WSDL[29] library in order to automatically create the XSD file which is equivalent to the ontology instances.

We know that the transformation of the OWL entities structure to an XML Schema could suppose some semantic lack. For example, OWL allows defining the Accommodation class as the union of the Hotel, BB, Refuge, GuestHouse … classes, which is not possible using an XML Schema. From our point of view, this lack of semantic is not important for the problem we are solving. On the one side, the SEMTOUR Services are Web services and they can be invoked both from the SEMTOUR community and outside by external customers. So the input and the output parameters of a SEMTOUR service should be standard, they are XML data, and as SEMTOUR service, fulfills the XSD of the ontology. On the other side, the invoked Web services receive and return XML data fulfilling the XML Schemas declared within its WSDL file. Briefly, they are using XML data.

If within a SEMTOUR Service there is the necessity of transform XML to ontology entities, for example to choose a hotel according to with the requirements of customer, the data should have to be transformed to ontology instances, this instances should have to be added to the ontology and the ontology with the instances should be sent to a reasoner together with the corresponding description logic query. In other words, an additional service should have to be implemented by the SEMTOUR platform and additional code should have to be automatically generated during the Basic Process generation, to invoke this new infrastructure service. This additional code can be used, for example, for ordering the returned values following additional criterias such as the quality and reputation of the returned products or the provider service.

5.3 ST_ECSWS_SemanticBPEditor Developent State.

The ST_ECSWS_SemanticBPEditor component aims to allow a user to create the BPEL code which implements either a Basic SEMTOUR service or a new Composed Service.

The Editor of Basic SEMTOUR Services.

After annotating the WSDL, generating the Grounding files and include reference to their URIs in the WSDL file, the BPEL for a Basic SEMTOUR service can be created.

Once the the SAWSDL file has been created, it is possible to create a “BPELED” wrapper which will allow to access the original Web Service through it: it will contain a message subscription for each of the new messages, their data will be compliant with the XSD of the ontology; the message subscription will invoke the Lowering transformation of the input data; the returned data will be used to invoke the original Web Service; finally the returned data will be transformed to the ontology format by invoking the corresponding Lifting transformation.

Finally, as we describe in paragraph 4.1, before the new Basic Service can be deployed two additional changes should have to be performed in its SAWSDL. The first step can be either realized within the ECSWS or by the remote repository manager. This step consists in the substitution of the original WSDL data schema definition by the corresponding to the ontology and extending this change to all the messages definition. The second change can only be performed within the community as it concerns with the replacement of the original endpoint by the one that corresponds to the server in which the BPEL is going to be deployed. Obviously this change can only be performed once the service is deployed.

The Editor of Composed SEMTOUR Services.

Composed services are created by the aggregation of the operations of already deployed SEMTOUR Services (either Basic or Composed). To create them we proceed in opposite direction to the one we follow with Basic Services. We first generated the BPEL, and then we generated the SAWSDL file which will be used to publish and to discover the new service once it would be deployed. As the services to be composed are SEMTOUR Services, they all will receive standard messages and in consequence the Grounding files should not have to be created (if needed they can be always created as the identity function).

The creation of the new service starts with the selection, by the user, of the services and their operations which are going to be aggregated and orchestrated by the new BPEL coded. Once they are selected, an automatic process is run to match inputs and outputs of the services.

6 Conclusions and Future Work.

The authors of this paper are members of the eLogistica work group of the Instituto Tecnológico de Aragón. Our interest is to research in the way ICT (both hardware and software) can be used to model, simulate and improve enterprise processes. Supply Chain Logistic Processes is our main sector of application as its processes are a well known field of exemplary synergies between the real world processes and the virtualized one. Although Supply Chain logistic is an historic field of use of the ICT, the popularization of use of the ICT facilities had created the actual trends to apply similar ideas within the eTourism, the eHealth or the eLearning sectors between many others.

The extension and generalization of the use of the ICT to the proposed or to others, to improve their business processes, can easily be reached. It depends on the existence of software systems supporting the different stages of an application creation (definition,, implementation, management and maintenance). This systems should be abstract enough to separate the sector (the domain data and processes definition and description) from the basic infrastructural services needed to create the business ecosystems (service publishing, discovering, execution, monitoring, …). And they can take advantage of the use standards interfaces to describe the services (WSDL), to allow the interoperability between services, between the services and the infrastructure (XML) and to aggregate them in new services (BPEL)

The SEMTOUR platform is an example. If we replace the domain ontology definition, the tourism one, by and eHealth ontology and we populate it with Basic eHealth Services, we will have the SEMHEALTH platform and the SEMHEALTH Studio. Obviously some improvements have to be made in SEMTOUR Studio and in SEMTOUR platform before their use can be generalized.

Some improvements to SEMTOUR Studio have been mentioned on this paper. An example is the inclusion of code for transforming XML data into OWL entities in the BPEL, useful if some reasoning is necessary with the returned data of an invoked service or service operation. Another example is the inclusion of facilities to define the security policies which the service follows and, in the case of Composed Services, the service demands to their providers.

The former is one of the possible improvements to make in order to implement Dynamic Service Composition both in the BPEL generation and in its execution. Another possible improvement is the heterogeneity of business processes to perform and action. For example, consider the creation of a composed service for making the reservation to travel from Madrid to Geneva spending three nights in Paris and resting a week around Geneva before coming back to Madrid. Suppose there are two candidate hotel in Paris A and B. The reservation process of A proceeds as follow: the process of answering the user data, the date of arrival and the date of departure, check the availability, answer for the confirmation and offers the payment process. The reservation process of the Hotel B receives the user data, the check in date and checkout date, request the confirmation and start the payment process. With this scenario, how can a virtualized process to create the full trip be created?

The answer seems to be clear. As with the ontology it is a two steps process: first, the knowledge about the domain processes should have to be expressed in pattern processes representing the operations work flows; next, the WSDL service description should be improved to include the matching of their operations to the standard processes and/or processes operations. In some cases, this will suppose the choreography definition to orchestrate two or more service operations in order to perform one service pattern operation.

For us, SEMTOUR Studio, is the second step of a bottom up approach to build a P2P semantic service integration platform. We started by defining the reference architecture and making an implementation of openBIP [32]. Next steps are to improve the discovering capabilities of openBIP with semantic search and to include BPEL engine. It would allow us using SEMTOUR Studio as the user tool to deploy services within openBIP. They are steps on our road to make software services facilities of the virtual world.

7 References.

[1] Semtour Home Page. http://www.ines.org.es/Semtour/

[2] Composetour Home Page. http://www.ines.org.es/composetour/.

[3] F.J. Lacueva, P.Peña, J.Vea-Murguia, E. Mendoza, S.Bilbao. Semtour. E3.2 ECSWS Requeriments Document.

[4] WSMO-Lite. http://www.wsmo.org/ns/wsmo-lite/

[5] Web Services Business Process Execution Language Version 2.0. http://docs.oasis-open.org/wsbpel/2.0/CS01/wsbpel-v2.0-CS01.pdf

[6] Marin Dimitrov , Alex Simov , Vassil Momtchev , Damyan Ognyanov. WSMO Studio - an Integrated Service Environment for WSMO (2005). In Proceedings of the 2nd Workshop on WSMO Implementations (WIW 2005). Volume 134.

[7] WSMO WG. Home page. http://www.wsmo.org/index.html

[8] Conceptual Model for Services. Home page. http://cms-wg.sti2.org/

[9] SOA4ALL. Home Page. http://www.soa4all.eu/

[10] SUPER. Home Page. http://www.ip-super.org/

[11] INFRAWEBS. http://www.infrawebs.eu/

[12] Agre G. INFRAWEBS designer - A graphical tool for designing semantic Web Services [Internet]; 2006 [cited 2011 Apr 18]. Available from: www.scopus.com

[13] F.J. Lacueva, P.Peña, J.Vea-Murguia, E. Mendoza. PT3.E2. ECSWS Design Document.

[14] S. Bilbao, PT3.E2. SEMTOUR Platform Architecture Design Document.

[15] M.P. Papazoglou, D. Georgakopoulos, Service-Oriented Computing, special issue, guest editors introduction, Commun. ACM 46 (10) (2003) 24–28.

[16] T. Vitvar, J. Kopeck y, J. Viskova, D. Fensel. WSMO-Lite Annotations for Web Services. In the Semantic Web: Research and Applications, ESWC 2008.

[17] Semantic Annotations for WSDL Working Group. http://www.w3.org/2002/ws/sawsdl/

[18] Web Service Definition Language (WSDL). http://www.w3.org/TR/wsdl

[19] M. Dimitrov, A. Simov, M. Konstantinov and V. Momtchev: WSMO Studio - a Semantic Web Services Modelling Environment for WSMO (System Description). In ESWC 2007, Proceedings of the 4th European Semantic Web Conference, June 2007, Innsbruck, Austria

[20] D. Martin, M. Paolucci, and M. Wagner: Towards Semantic Annotations of Web Services: OWL-S from the SAWSDL Perspective. In OWL-S Experiences and Future Developments Workshop at ESWC 2007, June 2007, Innsbruck, Austria

[21]Kunal Verma and Amit Sheth: Semantically Annotating a Web Service, in IEEE Internet Computing 11 (no. 2), March–April 2007, pp. 83–85.

[22] WSMO-Grounding. http://www.wsmo.org/TR/d24/d24.2/v0.1/20070427/

[23] Web Services Business Process Execution Language Version 2.0. http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[21] N. Kavantzas, “Web Services Choreography Description Language 1.0,” editor’s draft, 3 Apr. 2004, W3C; http://lists.w3.org/Archives/Public/www-archive/2004Apr/att-0004/cdl_v1-editors-apr03-2004-pdf.pdf

[22] Papazoglou MP, Traverso P, Dustdar S, Leymann F. Service-oriented computing: state of the art and research challenges. Computer 2007];40(11):38–45.

[23] D. Bonino, F. Corno, L. Farinetti, A. Bosca. Ontology Driven Semantic Search. WSEAS Transaction on Information Science and Application, Issue 6, Volume 1, December 2004, pp. 1597-1605

[24] D. Bonino, F. Corno, L. Farinetti, A. Ferrato. Multilingual Semantic Elaboration in the DOSE platform. SAC 2004, ACM Symposium on Applied Computing, March 14-17, 2004, Nicosia, Cyprus

[25] D. Bonino, F. Corno, L. Farinetti. DOSE: a Distributed Open Semantic Elaboration Platform. ICTAI 2003, The 15th IEEE International Conference on Tools with Artificial Intelligence, November 3-5, 2003, Sacramento, California

[26] D. Bonino, F. Corno, L. Farinetti. Semantic annotation and search at the document substructure level. Poster at ISWC2003 - 2nd International Semantic Web Conference, Florida (USA), October 2003.

[27] EASY WSDL. Home Page. http://easywsdl.ow2.org/index.html

[28] XML2OWL. Home Page. http://xml2owl.sourceforge.net/index.php?input=about

[29] OWL2XSD. Home Page.

[30] Bouras, A.; Gouvas, P.; Kourtesis, D.; Mentzas, G.: Semantic Integration Of Business Applications Across Collaborative Value Networks. Springer: Boston 2007

[31] ITA. ICT for Logistic Work Group, http://sigma.ita.es/elogistica/index.php/en

[32] ITA. OpenBIP. http://sigma.ita.es/openBIP/