A Development-Oriented Process for Building Web Services Ontology using OWL-S Language: application in medical web services

Zaidi Fayçal
Computer Science Department
Tebessa University-Algeria

Khamer Lazhar
Computer Science Department
Tebessa University-Algeria

Zaidi Mohamed
UFR-SEE - University of
Perpignan Via Domitia – France

ABSTRACT
Semantic Web Services are considered to be the next step in the evolution of Web Services. In addition to Web Services, Semantic Web Services allows software agents to discover, select, call, compose, invoke, and execute automatically a Web services without the intervention of human beings. Specifically, there exists little support in terms of methodologies for designing a semantic web services, and the Web service ontology development is a very complex, creative process, and it is comparable with software engineering in complexity. In this paper, we propose a development-oriented process for building web service ontology using the OWL-S language. This process is explained by relating it to a medical web services.

Keywords
Semantic Web service, ontology engineering, OWL-S, Development Methodology.

1. INTRODUCTION
The Semantic Web Services vision [1][2][3], is to combine the technology of Web service (WSDL, SOAP, UDDI) [4], with Semantic Web [5] technologies (RDF, RDFS, OWL), and through this to enable automatic and dynamic interaction between software systems. Web Service technology allows the description of an interface in a standard way, but there is nothing about what the software system does, or what sequence of messages is used to interact with it. We can overcome this lack using Semantic Web technology.

Specifically, there are many ontology building methodologies suggested for various domains (Uschold and King methodology [6], Grüninger and Fox methodology [7], Methontology [8], ….), but there exists little support in terms of methodologies for designing a semantic web services, and the Web service ontology development is a very complex, creative process, and it is comparable with software engineering in complexity, because the Web service is characterized by the dynamicity aspect (the information transformation (inputs, outputs) and the state transformation (precondition, effect)). So, this task constitutes a major problem in the development of the semantic Web service technology. One of the solutions is to adapt the existing ontology building methodologies to the process of building Web service ontology.

The objective of this paper is to share with the semantic web service community the process followed to develop the web service ontology. This process is oriented development, and it is explained by using the OWL-S language [9], because the most of the work on semantic Web services has been based on OWL-S for semantically describing Web services. And to understand this process, we will step through an example of medical web services [10][11].

To this aim, this work is organized as follows: section 2 provides an overview of the OWL-S language, section 3 presents the phases of our methodology to building the web service ontology, and section 4 presents the related work in this area. Finally, Section 5 is devoted to the conclusion and the future work.

2. THE OWL-S LANGUAGE
OWL-S (Ontology Web Language for Web Service), is an ontology for describing Web Services represented in OWL [12] and as such is comprised of three top-level notions: the Service Profile includes information for 'service advertisement' which is used for Web Service Discovery; the Service Model contains descriptive information on the functionality of a service and its composition out of other services, whereby the service functionality is conceived as a process; the Service Grounding gives details of how to access the service, mapping from an abstract to a concrete specification for service usage.

The following figure illustrates the ontologies OWL-S: a resource provides a service; this service presents a ServiceProfile; is described by Service Model and supports a ServiceGrounding.

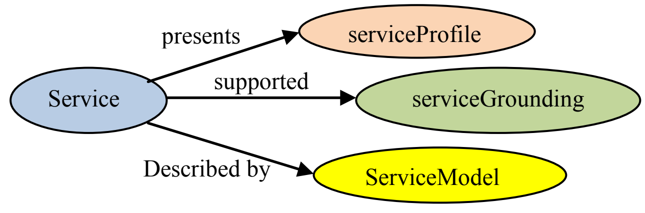

Fig 1: Ontologie OWL-S

3. BUILDING WEB SERVICES ONTOLOGY METHODOLOGY
This section describes the process of building Web services ontology. And we use a medical web services case, as an

example illustrating the use of this development-oriented process.

3.1. Building domain ontology

Before building the web service ontology, we must build the domain ontology, Because the ontology of Web services must uses the concepts of domain ontology to describe and define the concepts specific to Web services such as IOPEs (inputs / outputs / Preconditions / Effects), the Client, the Server, the Service Provider, Service category ... etc.
Several methodologies are used to building domain ontology (Methontology, On-To-Knowledge, Grüninger and Fox methodology,…).
The figure bellow shows an example of medical domain ontology edited by the protégé tool[1] (The Protégé tool is the most popular tool for creating ontologies).

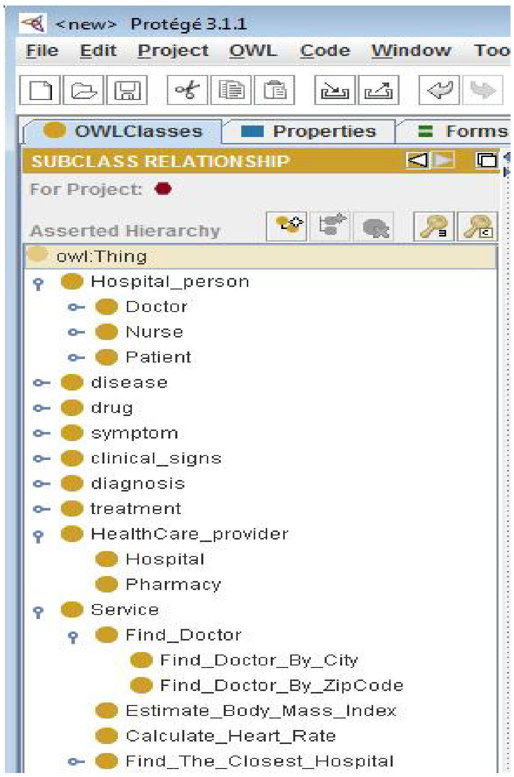

Fig 2: Process scheduling of the composite Web service

3.2. Building Web services ontology

After the domain ontology is built, we proceed to construct the Web services ontology. We following a process of building Web services ontology contains four phases (Specification of requirements, Conceptualization, Implementation and Mapping, Evaluation and maintenance).

3.2.1. Specification of requirements

The specification of requirements was considered the first step in this methodology, this phase is based on the specification step of the METHONTOLOGY methodology. In this phase, we specify in informal language the sources of services collected, the goal of this ontology and the users of this ontology.

In our medical web service case, we have:

- The sources of services collected are: medical encyclopedia, domain expert (doctors…), medical web site …
- The goal of this ontology is: Share collaboratively the medical Web services with users and machines, facilitate the discovery and research of services from multiple medical sources. And allows machines to discover, select, call, compose, invoke, and execute automatically a medical Web services without the intervention of human beings.
- The users of this ontology are: human actors (doctors, patients, lab technicians...), software actors (web services medical, medical applications...).

3.2.2. Conceptualization

Once the Specification of requirements of the Web service ontology has been created, the next step is the conceptualization phase. Many tasks performed in this phase such as:

- Collection of web services.
- Define a text description for each web service.
- Table of provider of service Web.
- Table of synonyms and category of web service.
- Table of Inputs, outputs, preconditions and effects of Web service.
- Identify the type of web service processes (atomic, simple, or composite).
- If the web service is composite, what are the sub web services with their scheduling (Sequence)?

-Collection of web services

In this phase, we will collect the Web services exist in the medical field, and we define a text description of each Web service.
Example : FindDoctorByZipCode, FindDoctorByCity, FindHospitalByZipCode, EstimateBodyMassIndex, …..

-Define the provider of service (service name, provider name, Telephone, email, address and the Web URL) as shown in the table below:

Table 1. Provider of Web services

Service name	Provider name	Tel	email	adress	Web URL
Find doctor by zip code	…..	…..	…..	…..	……

-Define the table of synonyms and the category of Web services as shown in the table below:

Table 2. Synonyms and category of Web services

Web service name	synonyms	Synonyms (French)	category
Calculate Heart rate	-Heart beat number	-rythme cardiaque -répétition	cardiology

[1] http://protege.stanford.edu/

	-Heart pulsation	cardiaque -nombre battements cardiaque -Fréquence cardiaque	
Estimate Body mass index	Estimate Quetelet index	-Indice de masse corporelle -corpulence d'une personne	Statistical device Clinical practice Medical underwriting
……………….	…….	………….	…….

-Define the table of Inputs, outputs, preconditions and effects of each Web service.

The following figure shows the web service estimate-body-mass-index with inputs and outputs.

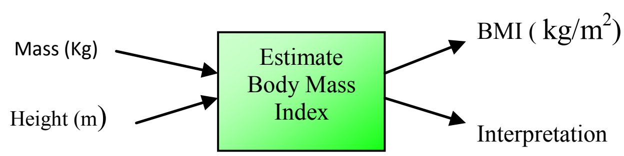

Fig 3: Inputs and Outputs of the Web service "Estimate-Body-Mass-Index"

-The table below shows the IOPEs (inputs, outputs, preconditions and effects) of Web service "estimate-body-mass-index".

Table 3. The IOPEs of the Web service

Service name	Inputs	Outputs	Precon-ditions	Effect
Estimate Body mass index	-Mass -Height	-BMI -Interpretation	Always True	Medical decision
…..	…..	……	…..	…….

-Identify the type of web service processes (atomic, simple, or composite).
In this sub-phase, we identify the type of Web service process as a table.

The process model ontology of OWL-S identifies three types of processes: atomic, simple, and composite:

The atomic processes are directly invocable (by passing them the appropriate messages). The Simple processes are not invocable and are not associated with a grounding, but, like atomic processes, they are conceived of as having single-step executions. The Composite processes are decomposable into other (non-composite or composite) processes; their decomposition can be specified by using control constructs such as SEQUENCE and IF-THEN-ELSE.

The table bellow show an example of the type of Web services in medical field:

Table 4. Type of web service processes

Web service name	Type of processes
Estimate Body Mass Index	atomic
Find the Closest hospital	Composite
…….	……..

-If the Web service is composite, we must indicate the process scheduling (Sequence) as a figure.
The Web service Find the Closest hospital is composed on two atomic Web service (Find hospital by city and Compute Distance) and the scheduling of this service is shown in the figure bellow:

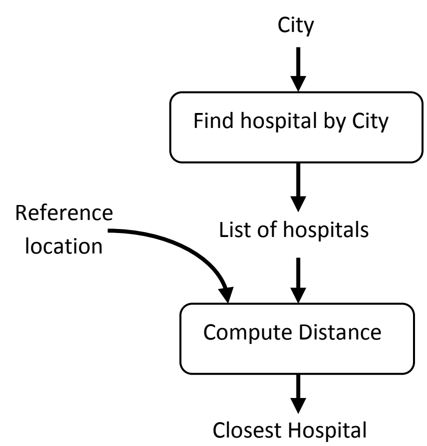

Fig 4: Process scheduling of the composite Web service "Find the Closest hospital"

3.2.3. Implementation and mapping

In this phase, we perform four steps.

First, we implement each web service by an integrated development environment like: Net Beans IDE (very easy), Eclipse, Jbuilder, Microsoft Visual Studio c#… And generate the WSDL file and the URL of the Web service.

Second, we will create the instances OWL-S (ServiceProfile, ServiceProcess, ServiceGrounding) of each Web service by using a tool for create OWL-S descriptions for a Web Service like: OWL-Editor for protégé, ASSAM Web Service Annotator, OWL-S Editor (MALTA University), WSDL2OWL-S,…

Third, we uses the concepts of domain ontology to define and describe the IOPEs (inputs/outputs/Preconditions/Effects), service category, service provider,… of Web service.

Fourth, we make a mapping between OWL-S (ServiceGrounding) and the WSDL file of Web service generated.

In our medical web service case “estimate-body-mass-index”, we have:

3.2.3.1. Implementation of the Web service

In this sub-phase, we implement the Web service by using the integrated development environment like Net Beans IDE. The following figure displays a part of the Web service: “estimate-body-mass-index” encoded in the java syntax using the integrated development environment "NetBeans IDE 6.8"[2].

```
Source  Design
/*...*/

package server;

import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;

/**...*/
@WebService()
public class EstimateBodyMassIndexWebService {

    /**...*/
    @WebMethod(operationName = "EstimateBMIoperation")
    public float EstimateBMIoperation(@WebParam(name = "M")
    float M, @WebParam(name = "H")
    float H) {
        //TODO write your implementation code here:
        float BMI = M/(H*H);
        return BMI;
    }
}
public class InterpretationBMIWebService {

    /**...*/
    @WebMethod(operationName = "InterpretationBMIoperation")
    public String InterpretationBMIoperation
            (@WebParam(name = "BMI")
    float BMI) {
        //TODO write your implementation code here:
     if (BMI < 16.0){       return "Severely underweight " ;  }
     else if (BMI >16 && BMI < 18.5 ){ return "Underweight "; }
     else if (BMI > 18.5 && BMI < 25 ){return "Normal";}
     else if (BMI > 25 && BMI < 30 ){  return "Overweight"; }
     else if (BMI > 30 && BMI < 35 ){  return "Obese Class I"; }
     else if (BMI > 35 && BMI < 40 ){  return "Obese Class II"; }
     else{                             return "Obese Class III";}
     }
}
```

Fig 5: Implementation of The web service in NetBeans IDE

3.2.3.2. Create the instances OWL-S

In this sub-phase, we must create:

- An instance for the service class.
- An instance for the service *Profile* class.
- An instance for the service *Process* class.
- An instance for the service *Grounding* class.

These instances are considerate as instances of the (ServiceProfile, ServiceProcess, ServiceGrounding concepts) of the domain ontology.

The following figure displays the instances OWL-S of the Web service “estimate-body-mass-index”.

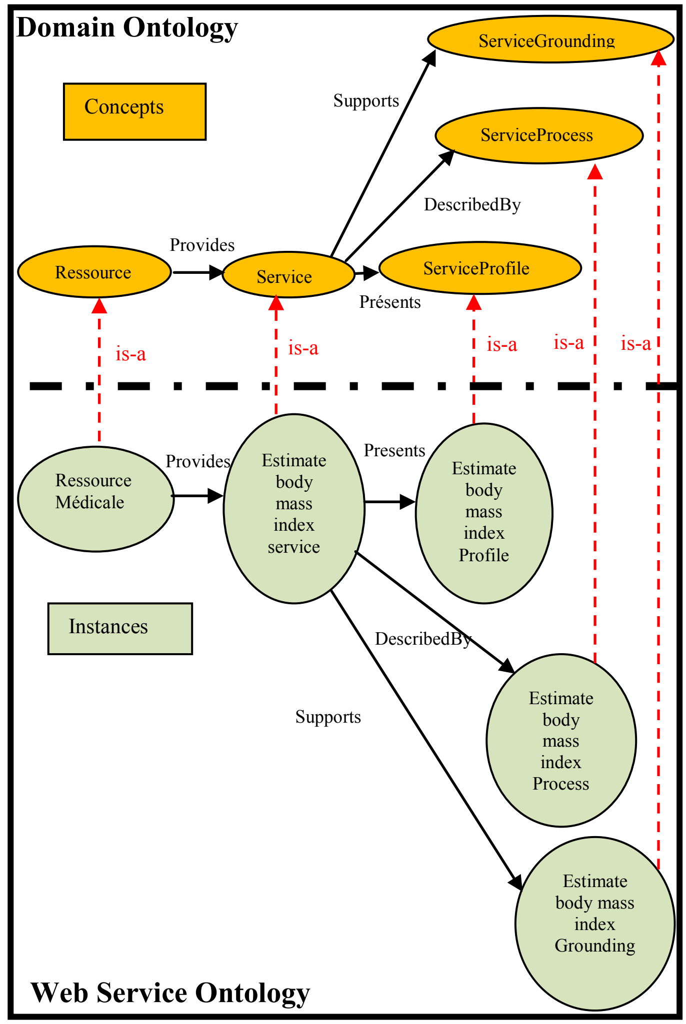

Fig 6: Create the instances OWL-S

The instances OWL-S of the Web service “estimate-body-mass-index” using the tool OWL-S Editor for Protégé[3] is shown in the following figure:

[2] www.netbeans.org/

[3] http://owlseditor.semwebcentral.org/

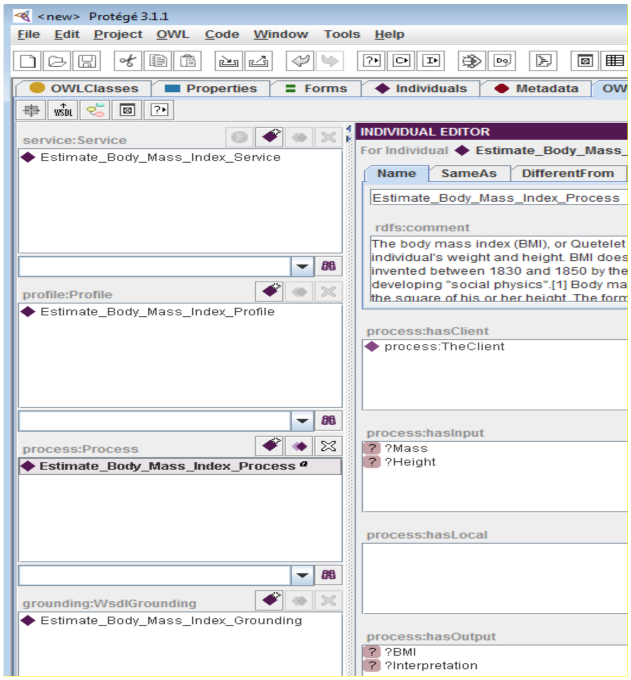

Fig7: The instances OWL-S in Owl-s editor for protégé

3.2.3.3. Define the Web service concepts

(Mapping between Web service ontology and domain ontology) In this sub-phase, we use the concepts of domain ontology to define and describe the Web service concepts: IOPEs (inputs/outputs/Preconditions/Effects), service category, service provider…etc, as shown in the figure below.

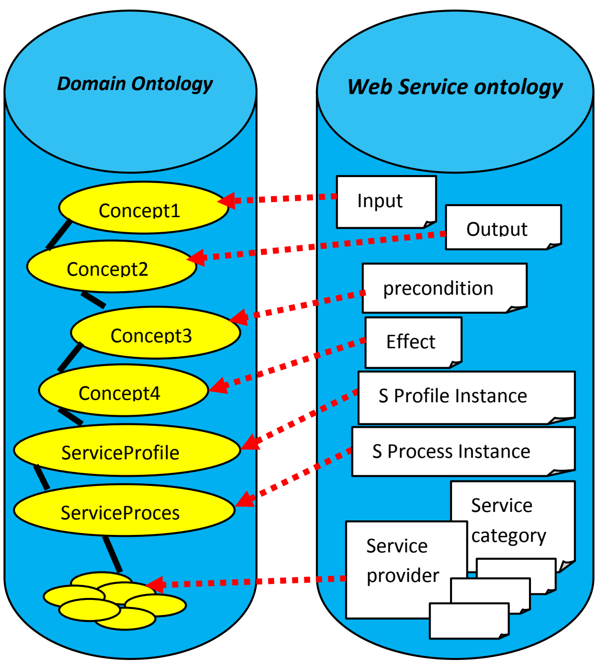

Fig 8: Define the Web service concepts "Mapping between domain ontology and Web service ontology"

3.2.3.4. Mapping between OWL-S and WSDL

In this sub-phase, we make a mapping between the elements of OWL-S (atomic process, owl-s parameter) and the elements of WSDL (wsdl operation, message part) as shown in the figure below:

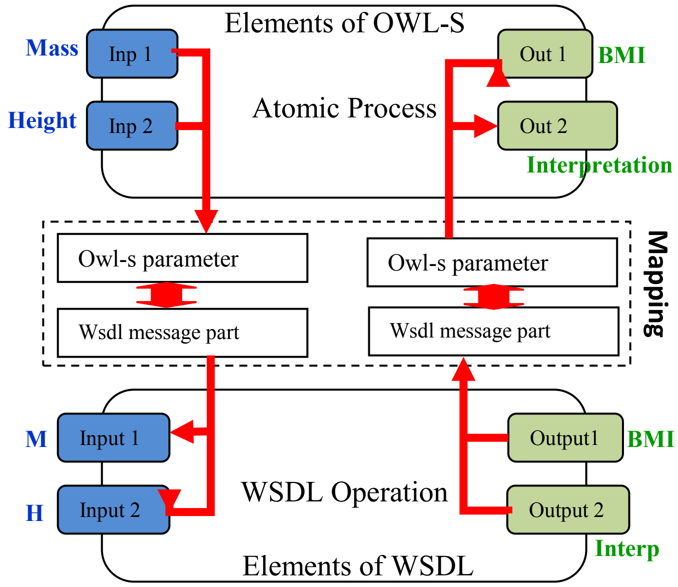

Fig 9: Relationship between the elements of OWL-S and the elements of WSDL

3.2.4. Evaluation and maintenance

The ontology of Web services needs to be evaluated to verify the consistency of the ontology, and to ensure the good functioning of all web services. It should also be evaluated according to criteria such as clarity, the ontology and its terms should be clear and unambiguous, consistency, the ontology needs to be free from contradictions, and reusability, define the possibilities to reuse the ontology and the extent of reuse. It is also important to specify who should update and maintain the ontology and how and when this should be done.

The maintenance sub-phase consists of updates and corrects the ontology if needed, and it's based on the maintenance phase of the METHONTOLOGY methodology: "Anytime, anywhere, someone could ask for including or modifying definitions in the ontology.To maintain the ontology is an important activity to be done carefully. Guidelines for maintaining ontologies are also needed".

The following figure displays the proposed development-oriented process with the four phases:

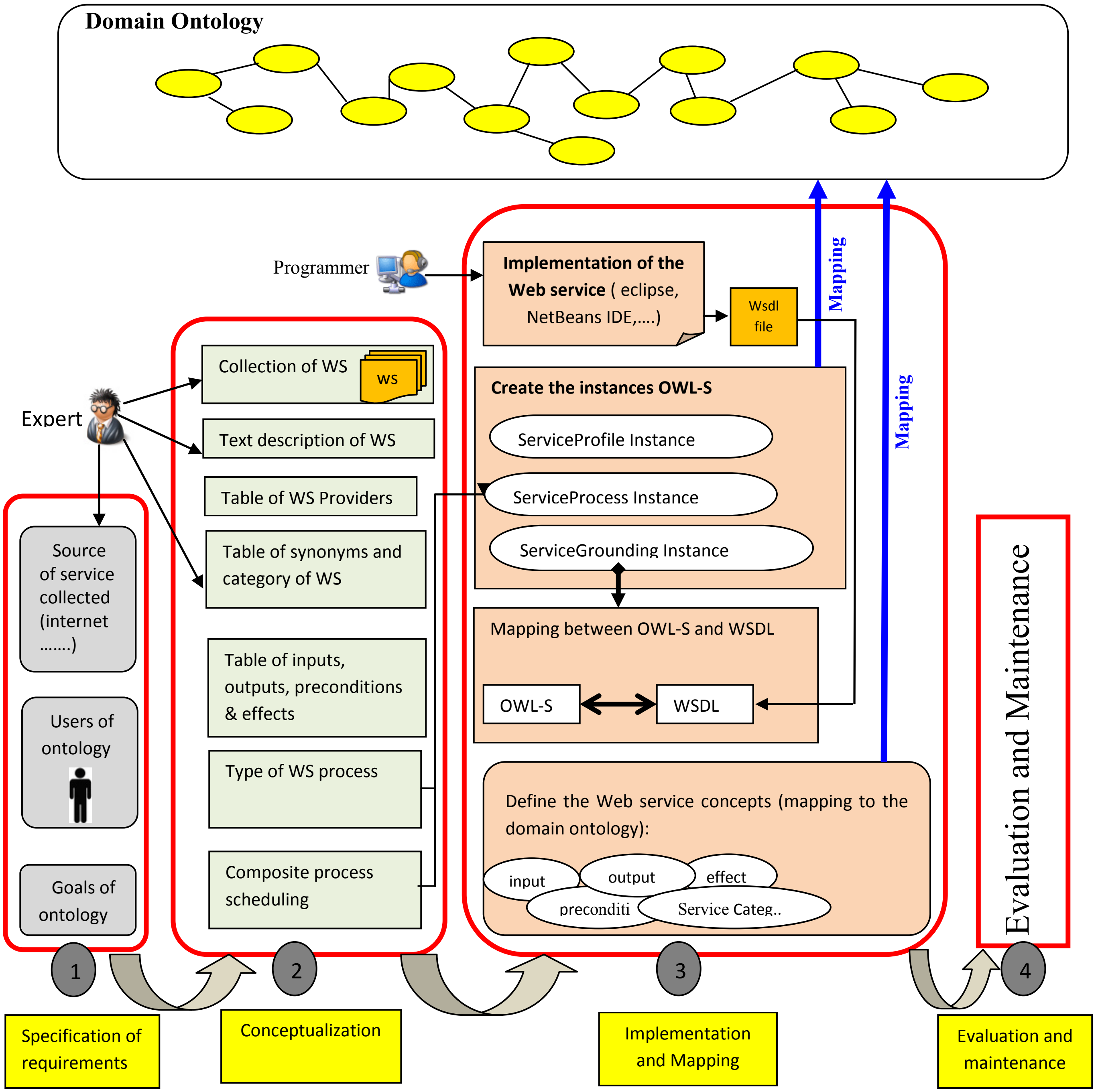

Fig10: The proposed development-oriented process with the four phases

4. RELATED WORK

This section outlines other efforts in the direction of building web services ontology. For this we have selected the work of Marta Sabou [13]. The works of Marta facilitate the process of building Web service ontologies and enhance the quality of these ontologies.

Marta employs two different ontology engineering methods. On one hand, he uses methods relying on generic ontological principles to improve the quality of OWL-S. On the other hand, he adapts ontology learning to support quick acquisition of Web service domain ontologies.

The Marta work's identified a set of methods to analyze generic Web service ontologies and to improve their quality. First, he recommends using the generic ontology to describe real life services in order to increase its modelling expressiveness. Second, he adapts the ontology for use in other, related domains in order to test the generality of the conceptualized knowledge. Third, he uses alignment to a foundational ontology for disambiguating the meaning of the concepts proposed by the ontology and for increasing its axiomatization. He used these methods to analyze OWL-S, identify its limitations and propose solutions to overcome these limitations.

5. CONCLUSION AND FUTURE WORK

In this paper we have presented a development-oriented process for building web services ontology using OWL-S language, this process was divided into four phases: (Specification of requirements, Conceptualization, Implementation and Mapping, Evaluation and maintenance), some phases (specifications of requirements, maintenance) are based on the METHONTOLOGY methodology. With this proposed process, we can effectively build a Web services ontology in different domain. This ontology allows software agents to discover, compose and invoke automatically a Web services without the intervention of human beings.

As future work we suggest to enhance and improve this oriented-development process by the semantic web service community.

6. REFERENCES

[1] R. Studer, S. Grimm, A, and A. Abecker, "Semantic Web Services Concepts, Technologies, and Applications", springer, Germany, 2007.

[2] D. Fensel, F M. Facca, E. Simperl, and I. Toma, "Semantic web services", springer, Austria, 2011.X. Liu, S. Akram, A. Bouguettaya, " Change Management for Semantic Web Services", springer, USA, 2011.

[3] E. Newcomer, "Understanding Web Services XML, WSDL, SOAP, and UDDI", Pearson Education 3rd Edition August 2004, CANADA.

[4] T. Berners-Lee, J. Hendler and O. Lassila, (2001). "The Semantic Web". Scientific American, 284(5):34–43.

[5] M. Uschold, M. King, "a Methodology for Building Ontologies", Workshop on Basic Ontological Issues in Knowledge Sharing, 1995.

[6] M. Grüninger, and M.S. Fox, "Methodology for the Design and Evaluation of Ontologies", Proc Int'l Joint Conf AI, Workshop on Basic Ontological Issues in Knowledge Sharing, 1995.

[7] Mariano FERNANDEZ, Asuncion GOMEZ-PEREZ et Natalia JURISTO. "Methontology: from ontological art towards ontological engineering". In Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering, pages 33–40, Stanford, USA, March 1997.

[8] D. L. Martin, M. Paolucci, S. A. McIlraith, M. H. Burstein, D. V. McDermott, D. L.McGuinness, B. Parsia, T. R. Payne, M. Sabou, M. Solanki, N. Srinivasan, et K. P. Sycara, "Bringing Semantics to Web Services : The OWL-S Approach", Lecture Notes in Computer Science, Springer, 2004, pp. 26–42.

[9] F. Zaidi, and M. Touahria, "Utilisation des technologies de services Web sémantiques dans le domaine medical", In ICIST 2011: Proceedings of the Inforamtion Systems and Technologies, pages 585–594, TEBESSA, ALGERIA, 2011.

[10] F. Zaidi, and M. Touahria, "A semantic Web services for medical analysis using the owl-s language", International Journal of Computer Applications, September 2011.

[11] Horrocks, P. F. Patel-Schneider et F. Van Harmelen, 2003. "From shiq and rdf to owl : The making of a web ontology language". Journal of Web Semantics (1), 726.

[12] Marta Sabou , "Building Web Service Ontologies", Doctorat thesis, Vrije Universiteit, Amsterdam, 2006.