An Ontology-Based Conceptual Mapping Framework for Translating FBPML to the Web Services Ontology

Gayathri Nadarajan and Yun-Heh Chen-Burger
Centre for Intelligent Systems and their Applications (CISA)
School of Informatics, University of Edinburgh, U.K.
G.Nadarajan@sms.ed.ac.uk, jessicac@inf.ed.ac.uk

Abstract

This paper presents an ontology-based conceptual mapping framework that translates a formal and visually rich business process modeling (BPM) language, Fundamental Business Process Modelling Language (FBPML) to a Semantic Web-based language, the Web Services Ontology (OWL-S). The translation aims to narrow the gap between Enterprise Modelling methods and Semantic Web services, thus bringing the two communities closer. Another significant contribution of the translation is that it allows more mature technologies such as BPM methods to be utilised within emerging fields that are constantly evolving, such as the Semantic Web. The framework is divided into a data model translation and a process model translation. An implementation and an evaluation of the process model translation are demonstrated and discussed.

1. Introduction

The need for more sophisticated Web-based support tools has become apparent with the fast advancement of the Web and the Semantic Web vision [3]. Business-to-Business (B2B) Electronic Commerce is fast becoming the most important application area of Semantic Web technology in terms of market volume [2]. Enterprise Modelling (EM) methods, on the other hand, are mature, established procedures that are commonly used as an analysis tool for describing and redesigning businesses by entrepreneurs. They have been well recognised for their value in providing a more organised way to describe complex, informal domain [6].

For organisations with business goals, the automation of business processes as Web services is increasingly important, especially with many business transactions taking place within the Web today. The existence of established EM methods, such as Business Process Modelling (BPM) methods, suggests that they could be exploited by emerging technologies such as Semantic Web services to provide a more mature framework incorporating both business- and Web application-specific technologies. In a wider context this aims to bring business-oriented and technical-oriented communities closer in order to achieve common organisational goals.

2. Background

2.1. FBPML

The Fundamental Business Process Modelling Language, FBPML [8] was designed to support today's ever changing workflow environment that meets diversified requirements. It is an inherited, specialised and combined version of several standard modelling languages. In particular, FBPML adapts and merges two established process languages; Process Specification Language (PSL) [16] and Integrated Definition Method (IDEF3) [11] by incorporating the visual capabilities of IDEF3 and the formal semantics for process modelling concepts provided by PSL.

The main aim of FBPML is to provide support for virtual organisations which are becoming more and more pervasive with the advancement of Web technology and services. It ultimately seeks to provide distributed knowledge- and semantic-based manipulation and collaboration. Most importantly, people with different responsibilities and capabilities could work together to accomplish tasks and goals without technological or communication barriers caused by the differences in their roles.

FBPML can express business processes in conventional first order predicate logic. It has two sections to provide theories and formal representations for describing data and processes; the Data Language and the Process Language. The FBPML Data Language (FBPML DL) [5] is first-ordered. The syntactic convention that has been used in Prolog has been adopted for its representation. It provides

definitions for concepts, functions, logical quantifications, predicates and meta-predicates. The Process Language is both visual and formal, thus it provides an intuitive representation and the same convention for syntax as the Data Language.

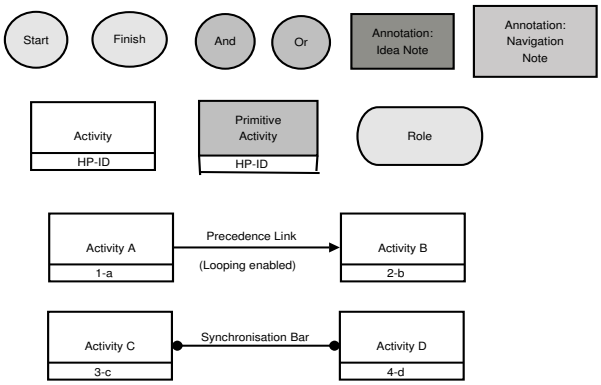

Figure 1. FBPML visual notation.

2.2. OWL-S

OWL-S [10] is a Web service ontology written in the Web Ontology Language (OWL) [12] extended with the Semantic Web Rule Language (SWRL) [14]. Its main aim is to describe Web services in machine-processable forms. OWL-S markup of Web services facilitates the automation of Web service tasks, including automated Web service discovery, execution, composition and interoperation.

Two other Semantic Web-based languages that act as competitors to OWL-S are Business Process Execution Language for Web services (BPEL4WS) [4] and Web Services Modelling Ontology (WSMO) [15]. BPEL4WS, developed earlier than OWL-S, was not selected for the mapping framework because it does not possess the structural similarity that OWL-S has with FBPML. The distinction between the data and process schemas in OWL-S has been one of the major contributing factors for choosing it over the other two. WSMO, on the other hand, is a newer technology than OWL-S. Although its framework is quite extensive, many of its aspects are still under development and it is therefore unclear how a mapping between FBPML and this language could be carried out. Beside the reasons mentioned above, OWL-S is also fast becoming the de facto standard for the composition of Semantic Web services, therefore it is the most appropriate Semantic Web based language to work with.

3. An Ontology-Based Conceptual Mapping Framework

A conceptual mapping framework was devised to translate FBPML to OWL-S, motivated by the fact that both languages have a clear separation between their data and process schemas. FBPML's data model is described in the FBPML Data Language while OWL-S is described in OWL and SWRL. FBPML's process model is described by the FBPML Process Language (FBPML PL), while OWL-S contains its own classes to describe its process model. Thus the mapping framework has been divided into a data model part and a process model part. The two clearly defined mapping parts constitute the main essence of this work.

The data and process models of each language are represented using ontologies. The ontology diagrams for FBPML data and process models were based on the Advanced Knowledge Technologies (AKT) Support Ontology from the AKT Project [1]. By using a semantic representation of the data and process models, the translation is conducted with accordance to mapping principles that are outlined in the following section.

4. Mapping Principles

We have opted to take the following high level mapping principles [7] to translate concepts, relations and processes described in FBPML to OWL-S:
1) One-layered parent-tracking translation. This mechanism proposes that the mapping and tracking of relations between FBPML or OWL-S classes is only recorded at one-layer; the immediate parent level. The grandparent and sibling classes may be derived via those links. This mechanism is directional. This resembles the way an ontology is represented, i.e. the links are made explicit between a child and its immediate parent, which will allow relations between non-parent-child nodes to be derived from it.
2) The use of two types of links (*subclass-of* and *is-related-to*). In a FBPML or OWL-S ontology, the only two links that are used between the classes are subclass relationships that link a child to its parent and the *is-related-to* link which is more general; represented as a dashed link between two classes with the relationship name labeled on the link.

5. Data Model Translation

5.1. Procedure

The mapping of data models between FBPML and OWL-S involves the translation of representations of concepts (or classes), instances (of the concepts) and the relationships between the concepts and instances from FBPML

DL to OWL. It also entails the translation of representations of properties and restrictions (or constraints) from FBPML DL to OWL and SWRL. The procedure for data model mapping could be summarised in the following steps [17]:
1) Pre-processing (if any). E.g. organise file into Prolog readable syntax.
2) Mapping of ontologies.
- Mapping of concepts.
- Mapping of instances.
- Mapping of relations (between concepts).
- Mapping of properties and restrictions (of concepts).
3) Mapping for rules/axioms.

The devised data model translation encompassed the mapping of concepts, instances, relations and properties. Restrictions and rules are expressed in OWL using SWRL, which is currently being extended towards first-order logic (FOL) expressivity. Thus it is still unclear how a translation could be performed between FBPML DL and SWRL FOL.

5.2. Ontologies of the Data Models

The data model of FBPML and OWL-S are represented semantically using ontology notation, as described by Figure 2 and Figure 3.

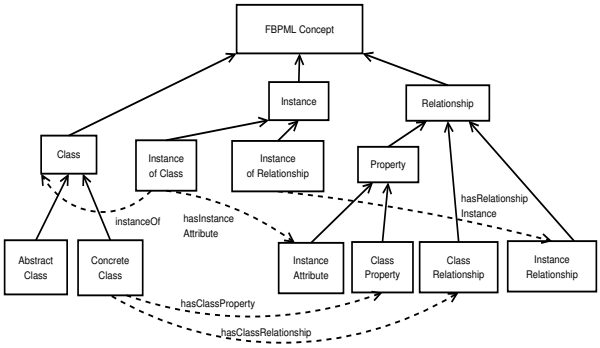

Figure 2. Ontology notation for FBPML's data model.

Applying the mapping principles to the derived ontologies, a class in FBPML is mapped to a class in OWL-S, an instance of a class in FBPML is mapped to an instance of a class in OWL-S and a relationship in FBPML is mapped to a relationship in OWL-S. The following section provides the syntax for the translation from FBPML DL to OWL.

5.3. Mapping of Concepts

Following the Semantic Web layering approach, OWL is an RDF-based language (which is based on XML) that utilises tags and tree-like structures for its representation,

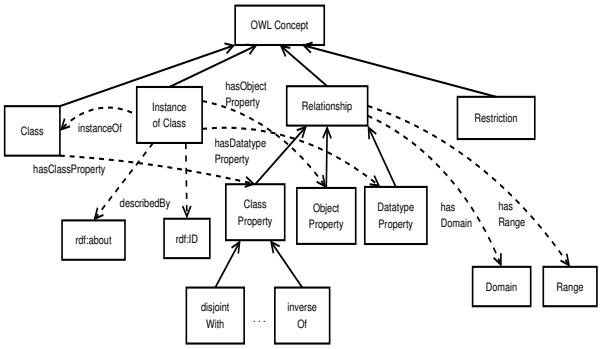

Figure 3. Ontology notation for OWL-S's data model.

Table 1. Mapping of classes and instances between FBPML DL and OWL

Concept	FBPML	OWL
Concrete Class	concrete_class(Name,Description, Example,Rules, CrossReferences, ObjectAttributes)	<owl:Class rdf:ID="ClassName"> <rdfs:comment>A comment</rdfs:comment> </owl:Class>
Abstract Class	abstract_class(Name,Description, Example,Rules, CrossReferences, ObjectAttributes)	<owl:Class rdf:ID="AbstractClassName"> <rdfs:comment>A comment</rdfs:comment> </owl:Class>
Instance	instance_of(InstanceName, ClassName)	<ClassName rdf:ID="Instance Name"/>

whereas FBPML DL is first-ordered. In the following section, syntax for concepts, instances and relations in FBPML DL and their corresponding translations in OWL are provided. The mapping between the classes (concepts) and instances (objects) could be given directly as illustrated by Table 1.

5.4. Mapping of Relationships

Binary relationships can be divided into three types in the context of the data model translation; those between two classes, those between two instances of a class and those between an instance of a class and a class. As some languages prefer to use certain conventions for specification, it should be clarified that properties are relations, just as objects are instances. For example, OWL specifies re-

lations using the elements `owl:DatatypeProperty` and `owl:ObjectProperty`. Thus an instance-to-instance relationship in FBPML (instance_rel/3) is translated to an object property in OWL while an instance-to-class relationship in FBPML (instance_att/3) is mapped to an OWL datatype property. A class property in FBPML is mapped to a class property in OWL. Table 2 provides the syntax for the translation of relationships from FBPML DL to OWL. The mapping procedure for relationships is slightly

Table 2. Mapping of relationships between FBPML DL and OWL

Relationship	FBPML	OWL
Class-to-Class Relationship	class_rel(Relation, Class1, Class2)	`<owl:Class rdf:about="#Class1"> <owl:Relation rdf:resource="#Class2"/> </owl:Class>`
Instance-to-Instance Relationship	instance_rel(Relation, Instance1, Instance2)	`<owl:ObjectProperty rdf:ID="Rel"> <rdfs:domain rdf:resource="#Obj1"/> <rdfs:range rdf:resource="#Obj2"/> </owl:ObjectProperty>`
Instance-to-Class Relationship	instance_att(Instance, AttributeName, AttributeValue)	`<owl:DatatypeProperty rdf:ID"Rel"/> <rdfs:domain rdf:resource="#Obj1"/> <rdfs:range rdf:resource="&xsd;string"/> </owl:DatatypeProperty>`

more complicated, as FBPML DL does not have a separate predicate to explicitly describe a particular relationship, whereas OWL provides elements to differentiate between object and datatype properties. Thus, to perform the translation, one has to extract the relationship name from the class or instance relationship. An obvious difference between FBPML DL and OWL is that in OWL, all distinct objects (classes, instances, data types, etc) have unique Uniform Resource Identifiers (URI), which avoids ambiguity when referring to objects with the same name over the Web but may not be equivalent. FBPML is a BPM language which is, at present, not directly compatible with Web services, thus it does not support URI based naming conventions. However, a default URI (such as a projects local URL) could be used as the base URI when the translation takes place.

6. Process Model Translation

The process model translation entailed the mapping between FBPML PL and OWL-S. FBPML PL is both visual and formal. Some of the process elements of FBPML PL are contained in Figure 1. A process model described in FBPML is made up of *Main Nodes, Junctions, Links* and *Annotations*. The main nodes are *Activity* (process), *Primitive Activity, Role* and *Time Point*. *Links* place temporal constraints on process execution and include *Precedence Links* and *Synchronisation Bars*. *Junctions* connect multiple activities, control the initiation and finishing of parallel processes and define their temporal constraints. The four types of junctions in FBPML are *Start, Finish, And* and *Or*. *And* and *Or* junctions are most commonly used in *Split* and *Joint* contexts and combinations of split and joint constructs are also used to represent more complicated models [8].

OWL-S is made up of four main classes; for the purpose of the process model translation, a subclass of the `Service Model`, the class `Process` was further examined. The ontology diagram for this class is given by Figure 5. The straight-lined arrows denote *subclass-of* links while the dotted-lined arrows denote *is-related-to* links.

The primitives in FBPML PL were mapped to the primitives in OWL-S. Several observations were made from the mapping of primitives between their process elements. In FBPML, a role is a responsibility in context; usually in the form of a *Provider* or a *Requester*. In OWL-S the `Participant` instances include `TheClient` and `TheServer`. Thus the *Requester* role corresponds to `TheClient` in OWL-S and the *Provider* role corresponds to `TheServer` OWL-S. FBPML's notion of role is richer and could refer to an individual, a group of people or software components or a combination of the above [8].

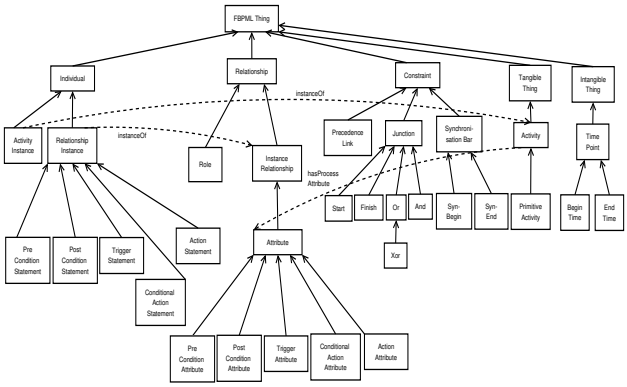

Figure 4. Ontology notation for FBPML process model.

The time ontology is used currently in limited

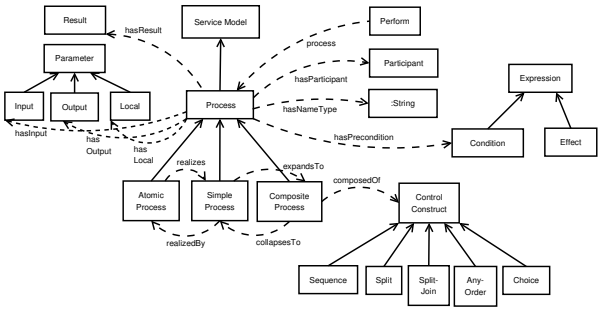

Figure 5. OWL-S process model ontology [10].

ways in the process specification in OWL-S (described in http://www.isi.edu/pan/damltime/time-entry.owl). Although there is no OWL-S construct that maps to FBPML's *Precedence Link*, the order of execution of processes within a sequence enclosed by `<objList:first>...</objList:first>` and `<objList:rest>...</objList:rest>`.

The *And-Split* is slightly modified (with barrier synchronisation) to correspond to the OWL-S `Split` construct where the processes that branch into the junctions are synchronised. The OWL-S `Choice` construct selects one process out of many for execution, which is equivalent to the *Xor* in FBPML, however, OWL-S does not provide a direct equivalent for the *Or* construct. The *Xor* junction is subsumed by the *Or* junction and has been utilised by more recent applications to make it more explicit for automation when bridging to (Semantic Web or Web service) methods that do not have *Or* junctions.

Since OWL-S does not support the notion of trigger, the combination of *Precondition* and *Trigger* in FBPML approximately map to `Precondition` in OWL-S. Postcondition in FBPML describes the effects and conditions which must hold true after the execution of a process.

In FBPML, an *Action* is the actual execution behaviours in a process model which can be stored in a repository. The advantage of the action repository is that actions can be reused and shared. Therefore an *Action* approximately maps to OWL-S `Atomic Process`. A precise specification of what it means to perform a process in OWL-S has not been given yet. Table 3 summarises the mapping of the primitives between FBPML PL and OWL-S process ontology.

6.1. Methodology

After determining the matching primitives, the mapping was followed through using simple process models consisting of sequences, and junctions [13]. The procedure for the mapping of typical (simple) models involved breaking down each process into a composite process, and considering a single construct to be translated, such as `Sequence`, `Split`, `Split-Join` or `Choice`. The whole composite process consists of a sequence of at most two processes. This component based model translation could be incrementally extended to cater for more complex models via a layered model translation. When the procedure is refined to cater for all process models, a general methodology for the process model mapping was achieved (refined from [9]).

1. Decompose FBPML process model in top-down order.
2. Translate model into a sequence process in OWL-S.
3. All activities between start and finish nodes are composite components of the sequence process.
4. Exhaustively decompose each composite component into a sequence of its basic process component, until as far as a simple process construct.

Table 3. Summary of mapping between FBPML and OWL-S process primitives

Primitive	FBPML	OWL-S
Main Nodes	Activity	Composite Process
	Primitive Activity	Atomic Process
	Role	Participant
	Time Point	See *Note*
Links	Precedence Link	(part of) Sequence
	Synchronisation Bar	See *Note*
Junctions	Start	See *Note*
	Finish	See *Note*
	And-Joint	Split-Join
	Or-Joint	See *Note*
	And-Split	Split
	Or-Split	Repeat-While, Repeat-Until
	Xor-Junction	Choice
Annotations	Idea Note	See *Note*
	Navigation Note	See *Note*
Process Components	Precondition	Precondition
	Trigger	See *Note*
	Postcondition	Effect
	Precondition,Trigger and Postcondition	Input/Output
	Action	Atomic Process
	Conditional Action	If-Then-Else

Note: Limited (or no) equivalent convention provided by OWL-S.

6.2. A Working Example

To illustrate the application of the methodology above, a more complex FBPML process model which incorporated a combination of sequences, junctions and composite processes was translated to OWL-S.

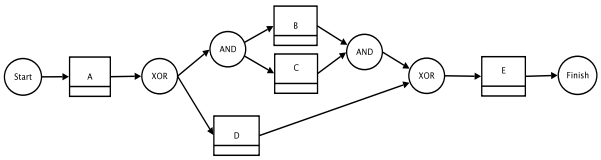

Figure 6. A FBPML Process Model made up of sequences, junctions and composite processes.

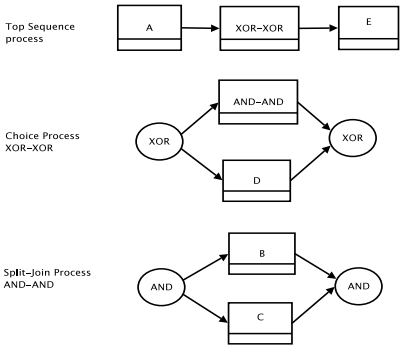

Figure 7. The Decomposition of the FBPML Process Model given by Figure 6.

FBPML formal notation:

start(A). (1)
activity(8,Xor-Xor,Trig1,Precond1,Postcond1,Act1,Desc2). (2)
junction(Xor,A,[And-And,D]). (3)
junction(Xor,[And-And,D],E). (4)
primitive_activity(1,E,Trig2,Precond2,Postcond2,Act2,Desc2). (5)
finish(E). (6)
activity(9,And-And,Trig3,Precond3,Postcond3,Act3,Desc3). (7)
junction(And,J1,[B, C]). (8)
junction(And,[B, C],J2). (9)
primitive_activity(2,A,Trig4,Precond4,Postcond4,Act4,Desc4). (10)
primitive_activity(3,B,Trig5,Precond5,Postcond5,Act5,Desc5). (11)
primitive_activity(4,C,Trig6,Precond6,Postcond6,Act6,Desc6). (12)
primitive_activity(5,D,Trig7,Precond7,Postcond7,Act7,Desc7). (13)

Assumptions and Interpretations:

- The ordering of junctions is from top-most (outer) to downmost (inner). For example the *Xor-Xor* definition (formulae 3-4) appears before the *And-And* definition (formulae 8-9).
- The branches coming out and going into the *Xor* junctions are called J1 and J2 respectively. Following the methodology provided previously, the process model could be decomposed in the following manner:
- The whole process is a composite process made up of a sequence consisting of atomic process A, a composite process containing both the outer most *Xor* junctions, called XOR-XOR, and atomic process, D.
- The Composite process XOR-XOR is a `Choice` process of either atomic process D or a `Split-Join` of B and C (the AND-AND junction).

The order of execution, incorporating OWL-S constructs, in prefix notation, is given as the follows:

`Sequence(A,Choice(Split-Join(B,C),D),E).`

The full translated syntax in OWL-S is provided in [13]. Due to space limitation, only relevant parts of the syntax that constitute the modelling of the process are shown and discussed.

```
<process:CompositeProcess rdf:ID="Complex Model">
...
 <process:Sequence>
  <process:components>
   <process:ControlConstructList>
    <objList:first>
     <process:Perform rdf:ID="A">
      <process:process rdf:resource="#A">
<-- Data flow and Parameter bindings -->
...
    <objList:first>
     <process:Perform rdf:ID="Xor-Xor">
      <process:process rdf:resource="#Xor-Xor">
       ...
    <process:ControlConstructList>
     <objList:first>
      <process:Perform rdf:ID="E">
       <process:process rdf:resource="#E">
       ...
    <objList:rest rdf:resource="&objList;#nil"/>
    </process:ControlConstructList>
   </process:components>
  </process:Sequence>
 </process:composedOf>
</process:CompositeProcess>
<-- End of Composite Process:  Complex Model -->

<process:CompositeProcess rdf:ID="Xor-Xor">
 <process:composedOf>
  <process:Choice>
   <process:components>
    <process:ControlConstructBag>
```

```
    <objList:first>
     <process:Perform rdf:ID="And-And">
     ...
      <process:Perform rdf:ID="D">
      ...
    <objList:rest rdf:resource="&objList;#nil"/>
    ...
<-- End of Xor-Xor -->

<process:CompositeProcess rdf:ID="And-And">
 <process:composedOf>
  <process:Split-Join>
   <process:components>
    <process:ControlConstructBag>
     <objList:first>
      <process:Perform rdf:ID="B">
      ...
       <process:Perform rdf:ID="C">
       ...
<-- End of And-And -->
```

From the application of the conceptual mapping methodology on the process model components, it can be seen that unlike the data model translation, the process model translation is less straightforward and poses some difficulties and challenges. This is interesting as it brings forth the capabilities and limitations of the two process model languages. It also highlights the differences between FBPML and OWL-S that were not revealed by the data model translation. FBPML PL is richer as it has both visual and formal representations for describing process model execution. The FBPML process diagram is intuitive and promotes human reasoning, OWL-S's RDF-based tags, on the contrary, aim to provide improved machine-processability. This accentuates the difference between languages in the Business Process Modelling domain and the Semantic Web services domain.

7. Implementation

A process model translator was developed using SICStus Prolog 3.10.1 on (Red Hat) Linux 9, the design was based on parsing first-order logic (Horn) clauses into hierarchical OWL and RDF tree-like tags. The general algorithm was based on process decomposition as outlined in the previous section. The initial implementation was based on specific simple procedures which were built up incrementally as the complexity of the process model increased. The main aim of the process model translator was to cater for any process model described in FBPML PL to be converted into OWL syntax. As pointed out in the previous section, the process model translation does not encompass all the possible primitives and process constructs, and is thus limited. Hence

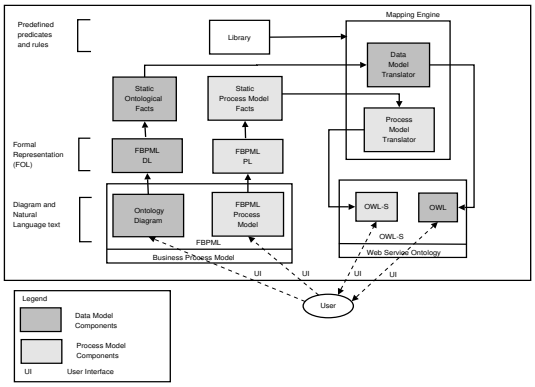

Figure 8. An Architectural Overview of the FBPML to OWL-S Mapping Engine.

the system was implemented to perform the translation as closely, accurately and directly as possible, taking into account some viable assumptions and interpretations.

The system implemented encompassed six types of translations, namely simple *Sequence, And-Split, And-Joint, Xor-Split, And-And* combination and complex layered model with any of these combinations. The overall architecture diagram is given by Figure 8.

8. Evaluation

The conceptual mapping framework and process model translator implementation demonstrated that most of the data model components could be translated directly, while the process model components, in particular the junctions, could only be partially (or not) translated. A constituent of a FBPML process model that could not be translated is recorded in the OWL-S comment construct.

The analysis suggests that the translation between FBPML and OWL-S is both partial and incomplete. Partial because there are some elements that exist in FBPML but not in OWLS (e.g. *Trigger, Or*) and incomplete because some of the translation cannot be conducted due to lack of knowledge about an element that is still in progress (e.g. rules described in SWRL FOL). The reason for this problem lies in the fact that OWL-S uses programming-like control constructs as its basic building blocks, which is inadequate for all process execution modelling [9].

The implementation of the process model translator, although limited, decomposes the sequences and combination junctions in a methodical manner. The problem will arise if loops, which may cause partly overlapped processes, are added to the process model. When this happens, the process model may not be decomposed, thus causing mapping problems. Thus, we can conclude that the formal mapping

between FBPML and OWL-S is very challenging and will require more insight and exploration before a reasonable mapping framework could be formulated. The essence of the analysis is that a much thorough understanding for both languages has been gained and this can contribute as the groundwork towards future directions.

9. Conclusion

We have demonstrated a conceptual mapping framework between two formal languages, FBPML and OWL-S. The former is traditionally used in the context of business process modelling and the latter in the domain of Semantic Web services. We have also attempted to automate the translation of the process modelling aspect between the two languages. The conceptual mapping exercise and implementation have brought to light some vital differences between the constructions of the two languages which suggest that the translation between them is partial. Furthermore, the specifications of some aspects of OWL-S are still in progress and hence, the translation is not complete.

A complete formalism for rules and conditions within OWL would allow for some of the gaps between FBPML and OWL-S to be filled. As the future for OWL-S remains unclear, current effort towards converging OWL-S with WSMO could be a positive step towards the development of a stronger and more stable global standard for Semantic Web services.

As a closure, narrowing the gap between business process technologies and Semantic Web services has opened a window of opportunity for the more established BPM methods to be utilised by the evolving Semantic Web technologies. It is hoped that the growth of Semantic Web standards such as OWL-S could be strengthened and enriched by manipulating more mature technologies such as BPM methods.

References

[1] AKT. *Advanced Knowledge Technologies IRC Project Technology Showcase*. Aberdeen, Edinburgh, Open, Sheffield, Southampton Universities, 2002. http://www.aktors.org.

[2] G. Antoniou and F. vanHarmelen. *A Semantic Web Primer*. MIT Press, Cambridge, MA, USA, 2004.

[3] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5), May 2001.

[4] BPEL4WS. *Business Process Execution Language for Web Services Version 1.1*. IBM, BEA Systems, Microsoft, SAP AG, Siebel Systems, 2003. http://www-128.ibm.com/developerworks/library/specification/ws-bpel/.

[5] Y.-H. Chen-Burger. Informal semantics for the fbpml data language. Technical report, Informatics Research Reports, School of Informatics, University of Edinburgh, 2002.

[6] Y.-H. Chen-Burger and D. Robertson. *Automating Business Modelling: A Guide to Using Logic to Represent Informal Methods and Support Reasoning*. Book Series of Advanced Information and Knowledge Processing, Springer Ver-Lag, 2005.

[7] Y.-H. Chen-Burger and A. Tate. Mapping principles between ix and compendium. Technical report, Informatics Research Report, EDI-INF-RR-0167, School of Informatics, University of Edinburgh, U.K., 2003.

[8] Y.-H. Chen-Burger, A. Tate, and D. Robertson. Enterprise modelling: A declarative approach for fbpml. In *Proceedings European Conference of Artificial Intelligence, Knowledge Management and Organisational Memories Workshop*, 2002.

[9] L. Guo, Y.-H. Chen-Burger, and D. Robertson. Mapping a business process model to a semantic web service model. In *Third IEEE International Conference on Web Services (ICWS04)*, 2004.

[10] D. Martin(editor). Owl-s semantic markup for web services, release 1.1, 2004. http://www.daml.org/services/owl-s/1.1/.

[11] R. Mayer, C. Menzel, M. Painter, P. Witte, T. Blinn, and B. Perakath. *Information Integration for Concurrent Engineering (IICE) IDEF3 Process Description Capture Method Report*. Knowledge Based Systems Inc. (KBSI), sept 1995.

[12] D. McGuinness and F. van Harmelen. *OWL Web Ontology Language*. World Wide Web Consortium (W3C), 2004. http://www.w3.org/TR/owl-features/.

[13] G. Nadarajan. Mapping fundamental business process modelling language to a semantic web based language. Master's thesis, School of Informatics, University of Edinburgh, U.K., 2005.

[14] P. Patel-Schneider. *Semantic Web Rule Language First-Order Logic (SWRL FOL)*. National Research Council of Canada, Network Inference, and Stanford University, 2005. http://www.w3.org/Submission/2005/01/.

[15] D. Roman, H. Lausen, and U. Keller, editors. *Web Service Modeling Ontology (WSMO)*. WSMO Final Draft, 2005. http://www.wsmo.org/TR/d2/v1.2/.

[16] C. Schlenoff, A. Knutila, and S. Ray, editors. *Proceedings of the First Process Specification Language (PSL) Roundtable*. National Institute of Standards and Technology, Gaithersburg, MD, 1997. http://www.nist.gov/psl/.

[17] J. Scicluna, R. Lara, A. Polleres, and H. Lausen. Formal mapping and tool to owl-s. *WSMO Working Draft*, Dec. 2004.