Semantic Interoperability of Web Services – Challenges and Experiences

Meenakshi Nagarajan, Kunal Verma, Amit P. Sheth, John Miller, Jon Lathem

meena610@gmail.com
http://lsdis.cs.uga.edu/~meena

LSDIS Lab, Department Of Computer Science, University of Georgia
Semantic Web services and processes at LSDIS

- **Meteor-S**
  - Operation Preconditions and Effects

- **Semantic WS – WSDL-S**
  - Policy SLA
  - Semantic Authentication

- **Interoperability – data mediation**
- **QoS specifications**
- **Semantic Discovery**
- **Semantics in the entire life cycle**
- **Autonomic Computing**
- **Semantic Composition**
- **Semantic Composition Transactions for Web processes**
Web services – then and now

• Surpassed communication, location, system level heterogeneities

• Heterogeneity in structure and semantics continued to exist

• Shifting focus on ‘semantic’ descriptions of service message elements and functionalities
  – Enable automation of discovery, composition, execution etc.

• Semantic match however does not ensure interoperation
A semantic match alone does not suffice

DATA MEDIATION REQUIRED

OUTPUT FROM WEB SERVICE 1

INPUT TO WEB SERVICE 1

OUTPUT FROM WEB SERVICE 2

INPUT TO WEB SERVICE 1
Outline of this talk

• Message level heterogeneities hindering interoperation
  – what are they, why do they exist
  – characterizing the different types

• Resolving such heterogeneities
  – State of the art
  – Our approach

• Semantic Web services : WSDL-S

• Using existing WS technology to achieve (automatic) mediation
  – WSDL + Axis 2
Message level Heterogeneities
More on Heterogeneities

• Databases *
  - **Syntactic heterogeneity** : differences in the language used for representing the elements
  - **Structural heterogeneity** : differences in the types, structures of the elements
  - **Model/Representational heterogeneity** : differences in the underlying models (database, ontologies) or their representations (relational, object-oriented)
  - **Semantic heterogeneity** : where the same real world entity is represented using different terms (or structures) or vice versa

• Web services
  - XML based environment eliminates syntactic and model heterogeneity
  - Structural and Semantic heterogeneities continue to exist

Classifying heterogeneities - 1

Domain Incompatibilities – attribute level differences that arise because of using different descriptions for semantically similar attributes

Naming conflicts
Two attributes that are semantically alike might have different names (synonyms)
Two attributes that are semantically unrelated might have the same names (homonyms)

<table>
<thead>
<tr>
<th>Web service 1</th>
<th>Web service 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student(Id#, Name)</td>
<td>Student(SSN, Name)</td>
</tr>
</tbody>
</table>

Data representation conflicts
Two attributes that are semantically similar might have different data types or representations

<table>
<thead>
<tr>
<th>Web service 1</th>
<th>Web service 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student(Id#, Name)</td>
<td>Student(Id#, Name)</td>
</tr>
<tr>
<td>Id# defined as a 4 digit number</td>
<td>Id# defined as a 9 digit number</td>
</tr>
</tbody>
</table>

Data scaling conflicts
Two attributes that are semantically similar might be represented using different precisions

<table>
<thead>
<tr>
<th>Web service 1</th>
<th>Web service 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marks 1-100</td>
<td>Grades A-F</td>
</tr>
</tbody>
</table>
Classifying heterogeneities - 2

Entity Definition – *entity level differences that arise because of using different descriptions for semantically similar entities*

**Naming conflicts**

- Semantically alike entities might have different names (synonyms)
- Semantically unrelated entities might have the same names (homonyms)

**Web service 1**

EMPLOYEE (Id#, Name)

**Web service 2**

WORKER (Id#, Name)

**Schema Isomorphism conflicts**

- Semantically similar entities may have different number of attributes

**Web service 1**

PERSON (Name, Address, HomePhoneNumber, WorkPhoneNumber)

**Web service 2**

PERSON (Name, Address, PhoneNumber)
## Classifying heterogeneities - 3

<table>
<thead>
<tr>
<th>Type of Conflicts</th>
<th>Description</th>
<th>Web Service 1</th>
<th>Web Service 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abstraction Level Incompatibility</strong></td>
<td>Entity and attribute level differences that arise because two semantically similar entities or attributes are represented at different levels of abstraction.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Generalization conflicts</strong></td>
<td>Semantically similar entities are represented at different levels of generalization in two Web services.</td>
<td>Web service 1: <code>GRAD-STUDENT</code> (ID, Name, Major)</td>
<td>Web service 2: <code>STUDENT</code> (ID, Name, Major, Type)</td>
</tr>
<tr>
<td><strong>Aggregation conflicts</strong></td>
<td>Semantically similar entities are represented at different levels of generalization in two Web services.</td>
<td>Web service 1: <code>PROFESSOR</code> (ID, Name, Dept)</td>
<td>Web service 2: <code>FACULTY</code> (ID, ProfID, Dept)</td>
</tr>
<tr>
<td><strong>Attribute Entity conflicts</strong></td>
<td>Semantically similar entity modeled as an attribute in one service and as an entity in the other.</td>
<td>Web service 1: <code>COURSE</code> (ID, Name, Semester)</td>
<td>Web service 2: <code>DEPT</code> (Course, Sem, .., ..)</td>
</tr>
</tbody>
</table>
Resolving message level heterogeneities

• State of the art solution: Service to service mappings
  – Proposed by most enterprise integration solutions

• Alternate solution: Mapping to a domain semantic model and re-use those to interoperate between services
  – Our approach

• Contributions
  – Comprehensive, practical approach to resolve message / data level heterogeneities
  – Solution borrows from the field of schema/data integration in federated databases
  – A data mediation architecture using extensible elements of WSDL and Axis 2
Our approach

• Leverages the semantic annotation framework provided by WSDL-S to create data mappings

• WSDL-S
  – Semantics in the entire life cycle of Web services
  – Evolutionary and compatible upgrade of existing Web services standards WSDL
  – Externalize the semantic domain models - agnostic to ontology representation languages.
  – W3C member submission
  – Semantic Annotations for Web Services Description Language Working Group - SAWSDL
    http://www.w3.org/2002/ws/sawsdl/
WSDL-S

- Annotating message types (XSD complex types and elements)
  - extension attribute: modelReference (semantic association)
  - extension attribute: schemaMapping (schema/data mapping)

- Annotating operations
  - extension elements: precondition and effect (child elements of the operation element)
  - extension attribute: category (on the interface element)
  - extension element: action (under consideration) (on operation element)
...........

<xs:element name= "processPurchaseOrderResponse" type="xs:string
wssem:modelReference="POOntology#OrderConfirmation"/>
</xs:schema>
</interface>

<operation name="processPurchaseOrder" pattern=wsdl:in-out>
<input messageLabel = "processPurchaseOrderRequest"
    element="tns:processPurchaseOrderRequest"/>
<output messageLabel ="processPurchaseOrderResponse"
    element="processPurchaseOrderResponse"/>

<!—Precondition and effect are added as extensible elements on an operation>
<wssem:precondition name="ExistingAcctPrecond"
    wssem:modelReference="POOntology#AccountExists">
<wssem:effect name="ItemReservedEffect"
    wssem:modelReference="POOntology#ItemReserved"/>
</operation>
</interface>
Annotating Message elements

1. **modelReference** to establish a semantic association
2. **schemaMapping** to resolve structural heterogeneities beyond a semantic match
Example Annotation

WSDL complex type element

<complexType name="POAddress"
wssem:modelReference="POOntology#Address"
<all>
<element name="streetAddr1" type="string" />
<element name="streetAddr2" type="string" />
<element name="poBox" type="string" />
<element name="city" type="string" />
<element name="zipCode" type="string" />
<element name="state" type="string" />
<element name="country" type="string" />
<element name="recipientInstName" type="string" />
</all>
</complexType>

OWL ontology

<POOntology:has_StreetAddress rdf:datatype="xs:string">
{ fn:concat($a/streetAddr1 , " ", $a/streetAddr2 ) }
</POOntology:has_StreetAddress>
Want to know more about WSDL-S?

- W3C submission Web page
  http://www.w3.org/Submission/WSDL-S/

- Project and related tools (annotation tools)
  http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-s/

- Presentation at W3C Workshop on Frameworks for Semantics in Web Services

- OR Talk to me!
Resolving message level heterogeneities using WSDL-S
WSDL-S support for data mediation

- User specified mappings from Web service message element to semantic model concept (say OWL Ontology)
  - Upcast: from WS message element to OWL concept
  - Downcast: from OWL concept to WS message element

```
<complexType name="Address">
  <sequence>
    <element name="StreetAd1" type="xsd:string"/>
    <element name="StreetAd2" type="xsd:string"/>
    ...........
  </sequence>
</complexType>
```
Realizing data mediation

- Web services interoperate by re-using these mappings.
  - Ontologies now a vehicle for Web services to resolve message level heterogeneities

Diagram:
- Web Service 1
  - Input message element
  - Output message element
  - Transform WS1 output to C1
  - Mappings provided

- Domain model 1

- Web Service 2
  - Input message element
  - Output message element
  - Transform C2 to WS2 input
  - Mappings provided

- Domain model 2

C1

C2
Data Mediation System Architecture

- Focus: Easy incorporation of tooling support for SWS in existing tools

- Uses extensibility support offered by WSDL and Axis 2 (handlers)

- Pre-requisites
  - Web services should be described using WSDL-S
  - The upcast and downcast mappings from the Web service message elements to the semantic concepts should be created
  - The Web services must be deployed and the WSDL-S files must be accessible. Axis 2 allows deployment of WSDL-S files.
DM Architecture components

- Part of the METEOR-S Middleware

  - EPR handler – End Point Resolution handler
    - For clients to use the middleware
    - Reroute SOAP messages to middleware

  - DM handler – Data Mediation handler
    - Main component for facilitating data mediation
    - Works with the EPR handler + a mapping processing engine (SAXON for XQuery / XSLT)
DM Handler – a closer look

• Each time a Web service is invoked
  – obtains the 'schemaMapping' functions from WSDL-S locations (using the WSDL-S4J API)
  
  – performs the up cast and downcast mappings on the incoming SOAP message using a mapping processor/engine (SAXON for XQuery and XSLT)
  
  – updates the SOAP message. Appropriate Axis handlers then invoke the Web service with the transformed message.
Walk through example – WS1 invocation

1. **CLIENT WS1 -> WS2**
   - Web service 1
   - Web service 2

2. **AXIS 2.0**
   - User phases
   - Axis phases
   - EPR handler
   - DM handler

3. **METEOR-S Middleware**
   - InFlow
   - OutFlow

4. **Conceptual Model**
   - XML (SOAP B) TO OWL
   - SOAP C
     - SOAP message Modified by DM handler
   - SOAP B
     - SOAP message Modified by DM handler
   - SOAP A
     - Original SOAP message

5. **SAXON Xquery/XSLT Engine**
   - DM Handler
     - * Enlarged view of the DM Handler

6. **owl to xml (soap c)**

7. **SOAP A**
   - Original SOAP message

8. **SOAP B**
   - Modified SOAP message with actual Web service EPR

9. **SOAP C**
   - SOAP message Modified by DM handler

10. **1,2 a**
    - SOAP A

11. **1,2 b**
    - SOAP B

12. **1,2 c**
    - SOAP C

13. **1,2 d**
    - Web service 1

14. **1,2 e**
    - SOAP C

15. **CLIENT WS1 -> WS2**
Walk through example – WS2 invocation

CLIENT WS1 -> WS2

XML (SOAP B) TO OWL

EPR handler

User phases

Axis phases

InFlow

1,2 b SOAP B

1,2 a SOAP A

2c SOAPC

2d

SOAP C

SOAP message

Modified by DM handler

AXIS 2.0

METEOR-S Middleware

SAXON Xquery/ XSLT Engine

DM Handler

* Enlarged view of the DM Handler

SOAP A

Original SOAP message

SOAP B

Modified SOAP message with actual Web service EPR

Web service 1

Web service 2

* DM handler

EPR handler

User phases

Axis phases

OutFlow

OWL TO XML (SOAP C)

Conceptual Model

XML (SOAP B) TO OWL

* DM handler

EPR handler

User phases

Axis phases

InFlow

1,2 b SOAP B

1,2 a SOAP A

1,2 e

CLIENT WS1 -> WS2
Evaluation

<table>
<thead>
<tr>
<th>URI of stock quote Web services</th>
<th>Structural</th>
<th>Schema Isomorphism</th>
<th>Attribute Naming</th>
<th>Entity Naming</th>
<th>Data Repres.</th>
<th>Can achieve interoperability using mappings</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://ws.strikeiron.com/SwanandMokashi/StockQuotes?wsdl">http://ws.strikeiron.com/SwanandMokashi/StockQuotes?wsdl</a></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>* <a href="http://ws.strikeiron.com/HistoricalStockQuotes?wsdl">http://ws.strikeiron.com/HistoricalStockQuotes?wsdl</a></td>
<td>Yes (minor)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://ws.strikeiron.com/BasicRealTimeQuotes?wsdl">http://ws.strikeiron.com/BasicRealTimeQuotes?wsdl</a></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>* <a href="http://www.webservicex.net/stockquote.asmx?wsdl">http://www.webservicex.net/stockquote.asmx?wsdl</a></td>
<td>Yes (minor)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.xmethods.net/sd/StockQuoteService.wsdl">http://www.xmethods.net/sd/StockQuoteService.wsdl</a></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>glkev.webs.innerhost.com/glkev_ws/StockServices.asmx?wsdl</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><a href="http://www.gama-system.com/webservices/stockquotes.asmx?wsdl">www.gama-system.com/webservices/stockquotes.asmx?wsdl</a></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>glkev.webs.innerhost.com/glkev_ws/HistoricalStockQuotes.asmx?wsdl</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ws.cdyne.com/delayedstockquote/delayedstockquote.asmx?wsdl</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>* <a href="http://www.xignite.com/xquotes.asmx?WSDL">www.xignite.com/xquotes.asmx?WSDL</a></td>
<td>Yes (minor)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

All stock quote Web services to interoperate with investment helper service – available at http://lsdis.cs.uga.edu/~meena/ICWS06/Eval.html

* These services could interoperate with the investment helper service using very minor mappings between the message schemas
Discussion

• Matching and mapping are hard problems
  – That is not what we claim to solve

• Need for a light weight semantic framework for Web services – WSDL-S
  – Simply extending this to achieve complete actual interoperation

• Interoperation in a multiple ontology environment
  – Inter ontology matching and mapping
Conclusion

- Comprehensive solution for resolving message level heterogeneities
  - Extending available semantics to pre-define message level mappings
  - Extending the state of the art

- Data mediation is a hard problem
  - WSDL-S and this work is an important first step