Web Service Composition using Service Suggestions

Rui Wang, Chaitanya Guttula, Maryam Panahiazar, Haseeb Yousaf, John A. Miller, Eileen T. Kraemer and Jessica C. Kissinger

University Of Georgia

Presented by John A Miller

At

IEEE International Workshop on Formal Methods in Services and Cloud Computing
July 2011
Outline

• Composing Web Services: Motivating Scenario
• Creating Parts of Workflows Is Straight Forward
• Unfortunately, One Is Often Left with Gaps that Are Difficult to Fill
• Service Suggestions Provide a Means for Reducing this Problem
• Shouldn’t Suggestions Be Type Safe
• How to Select Amongst Several Candidate Service Operations
  • Ranking Annotated Service Operations
• Integrating Service Suggestion Engine with Galaxy
• Output to Input Matching Algorithms
• Evaluation of Matching Algorithms
• Evaluation of Levels of Annotation
Motivating Scenario:
Analyzing Enzymatic Protein Sequences

Need to find similar enzymes in other organisms

G6PD protein
[Homo sapiens]

MAEQVALSRTQVCILREELFQGDADFHQSDTHIFI
MGASGDIAKKKIYPTIWWLFRDGLLPENTFIVGY

……
DLTYGNRYKNVKLPDAYERLILDVFCGSQMHFVRS
DELREAWRTPLLHIELEKPKIPYIYGSRGPT
EADELMKRQYFQYEGTYKWVNPHKL

Glucose-6-phosphate dehydrogenase

BLAST Database

BLAST search

Get Results

XML output returned by BLAST that contains multiple hits

Basic Local Align Search Tool
Analyzing Protein Sequences (cont.)

Wish I could make a workflow to make BLAST & ClustalW work together

XML input including multiple sequences in FASTA format

Multiple sequence Align Tool
Filling the Gap

• For a human designer to fill the gap would require extensive knowledge of bioinformatics tools and Web services as well as programming skills.

• An alternative is to have a system that helps the human designer to find and link appropriate Web services in order to fill the gap.
  • Data Mediation
  • Service Suggestions
Service Suggestions (IOFPE)

- **Type Safety**
  Ensuring that the output values of an operation be supplied as Input to another operation such that type errors are minimized.

- **Semantic Similarity**
  Measure the similarity between semantic concepts based on ontology. Used in ranking candidate Web services operations based on Input - Output (IO) matching as well as Functionality (F) matching.

- **Contract Compliance**
  Ensuring Preconditions (P) are satisfied when operations are suggested. Effects (E) used to update state.

- **Data Mediation**
  Resolving the heterogeneities between the input & output structures and transforming a subset of the output structure to the input structure.
## Strategies for Providing Type Safety

<table>
<thead>
<tr>
<th></th>
<th>XSD</th>
<th>OWL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal Typing</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Structural Typing</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Description Logic Based Typing</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
<operation name="getXMLResult">
  <input name="XMLResultReq" message="XMLResultReqMsg"></input>
  <output name="XMLResultResp" message="XMLResultRespMsg"></output>
</operation>

<message name="XMLResultRespMsg">
  <part name="parameters" element="XMLResultResp"></part>
</message>

<element name="XMLResultResp"> <complexType>
  <complexType name="tHits"> <seq> <element name="hit" type="tHit"/>
  </seq>
  </complexType>
  <complexType name="tAligns"> <seq> <element name="Aligns" type="tAligns"/>
  </seq>
  </complexType>
  <complexType name="tAlign"> <all>
    <element name="id" type="string"/>
    <element name="score" type="int"/>
    <element name="expectation" type="double"/>
    <element name="querySeq" type="string"/>
    <element name="pattern" type="string"/>
    <element name="match" type="float"/>
    <element name="gaps" type="float"/>
  </all>
</complexType>
</complexType> </element>

Output type of getXMLResult: XMLResultRespMsg

<operation name="filterSequence">
  <input name="SequenceReq" message="SequenceReqMsg"></input>
  <output name="SequenceResp" message="SequenceRespMsg"></output>
</operation>

<message name="SequenceReqMsg">
  <part name="parameters" element="SequenceReq"></part>
</message>

<element name="SequenceReq"> <complexType>
  <complexType name="tsequences"> <seq> <element name="sequence" type="tSequence"/>
  </seq>
  </complexType>
  <complexType name="tSequence"> <all>
    <element name="expectation" type="double"/>
    <element name="sequenceId" type="string"/>
  </all>
</complexType> </element> </complexType> </element>

Input type of filterSequence: SequenceReqMsg
Output: XMLResultResp to Input: SequenceReq

XSD Type 1

XSD Type 2
A problem to address: determine whether tAlign is a subtype of tSequence.

In our case, they are clearly not subtypes of each other.

In order to make them subtypes, several Complex types would have to be modified, e.g.,

```xml
<complexType name="tAlign" base="tSequence">
  <all>
    <element name="score" type="int"/>
    <element name="querySeq" type="string"/>
    <element name="pattern" type="string"/>
    <element name="match" type="float"/>
    <element name="gaps" type="float"/>
  </all>
</complexType>
```

In practice, it is unlikely that independently developed Web services will use type extension features of XSD as shown above.
XSD Structural Subtyping

- **Nominal typing** is **too inflexible** to be effective in the Web services domain.
- **Structural typing** is more flexible, because it analyses the structure of two Complex types without needing **explicit nominative relationships** (e.g., base=) between them.
- In our example, `tAlign` is a structural **subtype** of `tSequence` because:
  - they share the elements expectation and sequenceld, and
  - those elements are type-compatible at the primitive level.
Still Have Problems: Semantic Heterogeneities

- While structural typing provides advantages we can still have semantic mismatches.
- Suppose expectation value is called e-value in the tSequence complex type of the filterSequence operation.
- Unfortunately, structural typing would indicate type incompatibility.
- If additional semantic information were available that indicated expectation and e-value “mean the same thing”, then we could avoid this incompatibility.
- For Web services, a simple mechanism for providing such information is semantic annotation.
- Conversely, there will be cases where the names match, but the meanings do not.
A Low Cost Solution: Semantic Annotations


- **Model References**
  - Used for specifying semantic types for inputs and outputs
  - Used for specifying the functionalities / objectives of operations

- **Lifting / Lowering Schema Mappings**
  - Used to map WSDL / XSD types to semantic types (OWL concepts)
  - Lifting: transforms XML data to ontology instances
  - Lowering: transforms ontology instances to XML data

- **Preconditions & Effects**
  - Used for contract compliance
Using Semantic Annotations

Class: **Alignment**

SubClassOf:
- has_ID SOME String,
- has_score SOME int,
- has_expectation SOME double,
- has_querySeq SOME string,
- has_pattern SOME string,
- has_match SOME float,
- has_gaps SOME float,
- has_queryMatch SOME float

Alignment is an OWL class in Manchester Notation

Semantic Level

Lifting Schema Mappings

XSD Level

Output

Input
Type Safety: Same Model References

• In the case that both input and output types have model references to the same OWL class (e.g. Alignment), type safety is ensured unless:

• There is a missing property, i.e., a property of the input type that is
  • non-optional,
  • does not have a default value in the Ontology and
  • does not appear in the output type.
Type Safety: Different Model References

- In the case that both input and output types have model references to different OWL classes (e.g. $C_{out}$, $C_{in}$), type safety is ensured if

  - $C_{out}$ is subsumed by $C_{in}$ and
  - There are no missing properties.

Reasoners such as Pellet or HermiT can be used for Subsumption checking.
Semantic Similarity

- To provide suggestions it is useful to consider semantic similarity in addition to type safety.
- Several operations could be type safe, but need not necessarily be that desired.
- Therefore a ranking scheme based on semantic similarity is used.
- Aspects considered in Semantic Similarity
  - Output to input matching including data mediation
  - Functionality / Objective specification matching
  - Contract Compliance based upon preconditions and effects.
Ranking for Bidirectional Suggestions

Data Mediation Score

\[ S_{dm}(OP_x) = \frac{(S_{dm}([l_0, O_1, O_2], I_x), S_{dm}([O_x], I_3))}{2} \]

Functionality Score

\[ S_{fn}(OP_x) = w_{sem} \cdot \text{conSim}(C_s, C_d) + w_{syn} \cdot \text{synSim}(N_s, N_d) \]

Formal Service Specification Score

\[ S_{pre}(OP_x) = \begin{cases} 1 & \text{if } \text{st}_{pre} = \text{pre}(OP_x) \text{ and } \text{st}_x = \text{st}_{suf} \\ 0 & \text{otherwise} \end{cases} \]
Weighted Score for Semantic Similarity

\[ S = w_{dm} \cdot S_{dm} + w_{fn} \cdot S_{fn} + w_{pe} \cdot S_{pe} \]

where \( w_{dm} = w_{fn} = w_{pe} = 1/3 \)

- \( S_{dm} \): Score based on data mediation
  Scores calculated during data mediation

- \( S_{fn} \): Score based on functionality
  Compare the user specified functionality \( F_x' \) with candidate service’s functionality \( F_x \)

- \( S_{pe} \): Score based on formal service specification:
  Preconditions \( P_x \), effects \( E_x \) (requires WSDL-S)
  whether current state \( st \) will entail precondition of the candidate service
Providing Suggestion Capabilities for Galaxy Workflows

- **Galaxy** is a Web based platform that provides an integrated environment enabling bioinformaticians to analyze data with the help of numerous tools integrated into Galaxy.
- It also provides a way to **construct workflows** using the existing tools in a very simple fashion using a graphical designer based on **Yahoo pipes**.
- **WS Annotations Group** at University of Georgia has developed a tool which allows adding **Web Services** to Galaxy.
  - [http://mango.ctegd.uga.edu/jkissingLab/SWS/index.html](http://mango.ctegd.uga.edu/jkissingLab/SWS/index.html)
- The group also provides software for annotating and discovering Web services via **Radiant Web**.
- This work focuses on adding a **service suggestion capability** to Galaxy.
Connect BLAST with ClustalW Operations: Ask System for Suggestions

[Diagram of workflow canvas showing steps and options for BLAST and ClustalW operations]
List of Candidate Web Service Operations

Choose One of the Forward Suggestions
Adding the Suggested Web Service Operation
Ask for Suggestions to Finish Bridging the Gap

Choose One of the Bidirectional Suggestions
Completed Workflow
Multiple Algorithms for Output to Input Matching

- **Leaf Based Algorithm**
  Find an optimal matching for each of the leaf nodes in the input structure to a leaf node in the output structure. (Exclusive match requires Hungarian Algorithm.)

- **Path Based Algorithm**
  Find an optimal matching for each of the full paths in the input structure to a corresponding bottom path in the output structure. (A bottom path is one that starts at a leaf and has the same length as the corresponding input path.)

- **Structure Based Algorithms**
  - **Homomorphism Algorithm**
    Find an optimal sub-tree of the output structure that is isomorphic to the input structure.
  - **Homeomorphism Algorithm**
    Find an optimal sub-tree of the output structure that is topologically equivalent to the input structure.
Evaluation: Matching Algorithms

Average degree of overlap for 3 of the 4 matching algorithms
Working with Different Levels of Annotations

<table>
<thead>
<tr>
<th>Information</th>
<th>Syntax used for $S_{dm}$</th>
<th>Semantic annotations for $S_{dm}$</th>
<th>Semantic annotations for $S_{fn}$</th>
<th>Semantic annotations for $S_{pe}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 0</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Case 1</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Case 2</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Case 3</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Case 4</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Case 5</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Case 6</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Case 7</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Case 8</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Case 9</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Case 10</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Case 11</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Evaluating Different Levels of Annotation
Conclusions

• Using service suggestions to design workflows / compose Web services is comparatively a new area of inquiry.

• Our approach includes:
  • Type Safety
  • Semantic Similarity
  • Contract Compliance
  • Data Mediation

• Low overhead approach based on using SAWSDL for semantic annotations.

• Completing a first prototype of Service Suggestion Engine and its integration with Galaxy.

• Work also includes enriching bioinformatics ontologies (e.g., Ontology for Biomedical Investigations) as targets for semantic annotations.
Thank you
Related Work

• Personalized Reliable Web service Compositions
  Daniela Barreiro Claro1, Oriana Licchelli2, Patrick Albers2, and Raimundo Jose de Araujo Macedo1

• Semantic Web Service Composition Through a Matchmaking of Domain
  Freddy Lécué1, Alain Léger

• Kozlenkov et al, Architecture-Driven Service Discovery for Service Centric

• Spanoudakis G., Zisman A.: Discovering Services During Service-Based System

• Zisman, et al. "A Service Discovery Framework based on Linear Composition,“
  IEEE International Conference on Services Computing (SCC 2007), 2007

• Sycara et al., Automated discovery, interaction and composition of Semantic
  Web services, Web Semantics: Science, Services and Agents on the World
  Wide Web Volume 1, Issue 1, December 2003, Pages 27-46

• Mokhtar S.B et al, 2006. Efficient semantic service discovery in pervasive
  International Conference on Middleware (Middleware '06)
Related Work contd.


• A. Gao, D. Yang, and S. Tang, "Web Service Composition Based on Message Schema Analysis," in *Advances in Databases: Concepts, Systems and Applications*. vol. 4443,