A Framework for the Evaluation of Semantics-based Service Composition Approaches
(Work presented in the European Conference on Web Services)

Eduardo Silva, Luís Ferreira Pires, Marten van Sinderen
Centre for Telematics and Information Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
University of Twente
November 19th 2009

Outline

1. (Semantic) Service Composition
2. Problem Definition
3. Design Issues and Requirements
4. Our Evaluation Framework
5. Example
6. Conclusions and Future Work

Outline

1. (Semantic) Service Composition
2. Problem Definition
3. Design Issues and Requirements
4. Our Evaluation Framework
5. Example
6. Conclusions and Future Work

Service Composition

- **Combine services** towards the creation of a new service;
- **Basic constructor:** composition of service \( S_i(\text{Output}) \) with service \( S_j(\text{Input}) \):
  \[ S_i(\text{Output}) \rightarrow S_j(\text{Input}) \]
- "Traditionally" done at the **syntactic level**, e.g.: BPEL.

However, such manual and syntactic-level service composition becomes complex when large amounts of services are handled, or when it is made by non-professional users...
Service Composition

- Combine services towards the creation of a new service;
- Basic constructor: composition of service i output with service j input:
  \[ S_i(\text{Output}) \rightarrow S_j(\text{Input}) \]
- "Traditionally" done at the syntactic level, e.g.: BPEL.

However, such manual and syntactic-level service composition becomes complex when large amounts of services are handled, or when it is made by non-professional users...

Semantics-based Service Composition

- Use semantic information (from domain conceptualisations) to enrich the service description;
- Allows automated reasoning;
- Allows automatic service composition.

There are many approaches being proposed for semantics-based service composition.

Outline

1. (Semantic) Service Composition
2. Problem Definition
3. Design Issues and Requirements
4. Our Evaluation Framework
5. Example
6. Conclusions and Future Work
Problem

- How to evaluate semantics-based service composition approaches
- Nowadays we are doing evaluation in an "ad-hoc" manner!
- How can we do this systematically, so that we evaluate and are able to compare different approaches?

In this work we propose an initial methodology to evaluate semantics-based service composition approaches.

Outline

1. (Semantic) Service Composition
2. Problem Definition
3. Design Issues and Requirements
4. Our Evaluation Framework
5. Example
6. Conclusions and Future Work

Basic Requirements

- Service collections and ontologies common to all Evaluators;
- Generation of common evaluation scenarios:
  - Service requests;
  - Matching service compositions;
- Evaluation metrics;
- Common way to report the results.

We assume that the composition approaches are automated, i.e., they find matching service compositions given a service request.
Basic Requirements

- Service collections and ontologies common to all Evaluators;
- Generation of common evaluation scenarios:
  - Service requests;
  - Matching service compositions;
- Evaluation metrics;
- Common way to report the results.

We assume that the composition approaches are automated, i.e., they find matching service compositions given a service request.
System Architecture

But things are not so trivial... there are many issues here!

Services Collections

Requirements:
- Large collections of “realistic” semantically annotated services;
- Collection of ontologies used to annotate the semantic services.

Two direction to create Service Collections:
- **Existing semantic services**: S3-Contest (OWL-S, SAWSDL)[multiple ontologies], SWS-TC (OWL-S)[one ontology], OPOSSum (gathers existing collections and allows submit new services);
- **Automatically generated Services**: We found only the WS-Challenge generating automatically semantic services. We tried, but it is very complex to create “realistic” semantic services!

The first direction is being preferred and is receiving more attention.
## Generation of Evaluation Scenarios

**Requirements:**
- Services collections and service requests (SR) must be common to all the Evaluators;
- Services collections have to yield compositions! For each SR a set of matching reference service compositions (RSC) (at least one!) must exist.

**Approaches for generation of scenarios:**
- **Top-down:** Introduce services that yield to compositions, in an existing services collection;
- **Bottom-up:** Inspect an existing collection for possible service composition(s) and define service requests that yield such compositions;

**Issues:**
- **Top-down:**
  - Manual creation of reference service compositions (RSC) and service requests (SR);
  - New services have to be related with the existing ones, so the designer always has to analyse the services in the existing collections.
- **Bottom-up:**
  - Manual looking for possible service compositions in the existing service collections;
  - If the collection is large it is difficult to find the RSC (Reference Service Compositions).

For the bottom-up we can use a reference service composition approach (assuming that it is not complete). In the paper we give an example with bottom-up approach.
Evaluation Metrics

Metrics:
- **Confusion-matrix based**: measure the quality of the composition found by an approach;
- **Time-based**: measure how fast a composition approach is, and how it scales when the number of services in the registry change.

There may be other metrics, and other aspects, to consider when evaluating an approach, we now focus only on these two.

Outline

1. (Semantic) Service Composition
2. Problem Definition
3. Design Issues and Requirements
4. Our Evaluation Framework
5. Example
6. Conclusions and Future Work

Generation of Evaluation Scenarios

Generating all the artifacts to perform the evaluation process:

The Designer defines the reference services, ontologies, service requests and reference service requests, and Evaluators have to make sure to translate them to their approach description formalisms (languages).
Generation of Evaluation Scenarios

Generating all the artifacts to perform the evaluation process:

- **The Designer** defines the reference services, ontologies, service requests and reference service requests, and **Evaluators** have to make sure to translate them to their approach description formalisms (languages).

Evaluation Metrics

Confusion Matrix-based

<table>
<thead>
<tr>
<th>Actual Values</th>
<th>P</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classified Values</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP</td>
<td>TP = A ∩ FC \ RSC \ P</td>
<td></td>
</tr>
<tr>
<td>FP = FC \ P \ A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FN = RSC \ FC \ P</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TN = Unknown</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **P**: set of valid existing compositions;
- **N**: other (non-valid) compositions (unknown);
- **RSC**: reference service compositions (RSC ⊆ P);
- **A**: found compositions present in RSC;
- **A’**: valid by Judgement of Evaluators;
- **B**: not valid compositions;
- **C**: valid from RSC, but not found by the approach;

- **TP** = |TP| = |TP| + |FP|
- **TP** = |TP| + |FN|
- **FP** = |FP| + |TP|
- **FP** = |FP| + |TP|
- **FN** = |FN| + |FP|
- **FN** = |FN| + |FP|
- **TN** = Unknown

Graphical comparison of two approaches in terms of confusion-based metrics:
Evaluation Metrics

Time-based

It is difficult to do time-based evaluation and comparison of different approaches... different people have different hardware and communication means!

To overcome this we propose two metrics:
- **Composition Processing Time** ($\text{compProcTime}$): time taken to perform the whole process. To report on this metric, the Evaluators have to describe their supporting environment (CPU, Memory, etc).
- **Scalability**: used to compare different approaches. It provides an abstraction from the processing time, by measuring its variation when the number of services in the registry vary.

$$\text{Scalability} = \left( \frac{\partial \text{compProcTime}}{\partial \# \text{servs}} \right)^{-1} \approx \frac{1}{N-1} \left( \frac{\sum_{i=2}^{N} \text{compProcTime}(i) - \text{compProcTime}(i-1)}{\# \text{servs}(i) - \# \text{servs}(i-1)} \right)^{-1}$$

The lower the $\text{compProcTime}$ variation, the higher the scalability!
Semantics Services Collection

We consider the SWS-TC (Semantic-Web-Services Test Collection) set of services as our collection.

The reason for this is that it has only an ontology to describe all the collection services (241 OWL-S Services), and the services yield to compositions, on the contrary for example to the S3-Contest collection, that is larger but does not allow to find compositions.

Evaluation Scenarios

\[ \text{ServiceRequest}(SR) = \text{find a service (composition) that: given a title of a book retrieves the price of the book} \]

Reference Service Compositions (RSC):

This corresponds to 10 different RSC.

Evaluation of Composition Approaches

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Approach 1</th>
<th>Approach 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision of PPR</td>
<td>0.71</td>
<td>0.71</td>
</tr>
<tr>
<td>Recall of TPR</td>
<td>0.83</td>
<td>0.83</td>
</tr>
<tr>
<td>FDR</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>ACC</td>
<td>0.72</td>
<td>0.72</td>
</tr>
<tr>
<td>SNR/NC</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

Table: Metrics comparison
Conclusions: Approach 1 has a higher precision than Approach 2, and a lower FDR, i.e., Approach 1 retrieves a higher percentage of correct compositions than Approach 2. Although Approach 2 has a higher Recall, which means that it retrieves more correct compositions, however it also retrieves a many incorrect compositions. This can also be confirmed by the higher accuracy ($\text{Acc}^{TP}$) ($10/16$) and Signal-to-Noise Ratio ($\text{SNR}_{TP}^{FP}$) ($10/4$) of Approach 1.

Future Work

- **Semantics-based discovery** has received already some attention, but not so much the composition process;
- We propose an initial framework to evaluate semantics-based service composition approaches;
- We assume the existence of semantic services collections and ontologies (which is still another big issue!), from which we define composition scenarios (service requests, reference service compositions, etc);
- In our paper we present an examples where we use the SWS-TC collection. This is a small collection of services, so we may have to consider larger service collections (e.g.: OPOSSum). However, the majority of existing collections do not yield compositions, so they may require some adaptation;
- Include qualitative metrics, e.g.: design-time or runtime composition approach, type of user to be supported, etc. This will allow the framework to be more fair;
- Experiment with the framework, by applying it into some state-of-the-art semantics-based approaches.
Questions?

(Also Artists agree that composition is important! :) – [Banksy, Liverpool])

web: www.cs.utwente.nl/goncalvesem
email: e.m.g.silva[at]cs.utwente.nl