Enriching Service Semantics through Conceptual Spaces
- ONTOSE Workshop, CAISE’09, Amsterdam, 08 June 2009 -

Stefan Dietze, John Domingue
Knowledge Media Institute, The Open University, UK
Outline

- Semantic Web Services mediation
- Two-fold representation approach
- Prototypical application
- Conclusions
Semantic Web Services:

- Aim at automatic discovery, orchestration and invocation of Web services

- Formal specification of Web services in terms of their capabilities (Cap), interfaces (If) and non-functional properties (Nfp)
  \[ \text{Cap} \cup \text{If} \cup \text{Nfp} \subset \text{SWS} \]

- Capabilities describe assumptions (Ass) and effects (Eff)
  \[ \text{As} \cup \text{Ef} = \text{Cap} \]

- Defined through ontologies O, i.e. as tuple of concepts C, instances I, properties P, relations R and axioms A:
  \[ O = \{C, I, P, R, A\} \subset SWS \]

- Reference models such as OWL-S, WSMO or SAWSDL
Introduction
SWS Mediation
- SWS discovery: matchmaking of capabilities of SWS e.g.:
  \[ As_2 \subseteq As_1 \cup Ef_2 \subseteq Ef_1 \]
- SWS brokers match logical expressions
  (e.g. \[ As_1 \equiv \neg I_1 \cap I_2 \& As_2 \equiv I_3 \cap \neg I_4 \])
Introduction
Semantic Mediation for SWS Discovery

- SWS discovery: matchmaking of capabilities of SWS e.g.:
  \[ A_s \subseteq A_{s_1} \cap E_f \subseteq E_{f_1} \]
- SWS brokers match logical expressions
  (e.g. \( A_{s_1} \equiv \neg I_1 \cap I_2 \) & \( A_{s_2} \equiv I_3 \cap \neg I_4 \))
- Heterogeneous SWS annotations
- Semantic mediation:
  alignment/mapping of SWS concepts/instances across distinct SWS

Semantic-Level Mediation
Mediation between different semantic representations
Semantic Mediation
Issues

SWS Request (Consumer)

SWS₁
(get-flowers)

has-assumption

\[ A_{S₁} \equiv I₁ \cap I₂ \]

<Color rdf:ID="Lilac"/>

<Location rdf:ID="Milton_Keynes"/>
Semantic Mediation
Issues

SWS Request (Consumer)

\( SWS_1 \)
(get-flowers)

\[ A_s_1 \equiv I_1 \cap \neg I_2 \]

has-assumption

<Color rdf:ID="Lilac"/>

<Location rdf:ID="Milton_Keynes"/>

SWS (Provider)

\( SWS_2 \)
(provide-flowers)

\[ A_s_2 \equiv I_3 \cap \neg I_4 \]

has-assumption

<Colour rdf:ID="Purple"/>

<geospatialLocation rdf:ID="M-K"/>
Semantic Mediation Issues

SWS Request (Consumer)

SWS₁ (get-flowers)

\[ A_{S_1} \equiv \neg I_1 \cap I_2 \]

\(<\text{Color \: rdf:ID="Lilac"}/>\)

\(<\text{Location \: rdf:ID="Milton\_Keynes"}/>\)

SWS (Provider)

SWS₂ (provide-flowers)

\[ A_{S_2} \equiv I_3 \cap \neg I_4 \]

\(<\text{Colour \: rdf:ID="Purple"}/>\)

\(<\text{geospatialLocation \: rdf:ID="M-K"}/>\)
Issues:

- Heterogeneous SWS annotations
- Lack of implicit similarity information
  (symbol grounding problem)
- Reliance on either:
  - (i) manual mapping rules
  - (ii) ontology mapping mechanisms
    (exploiting linguistic or structural similarities)
- Costly and error-prone

==> representations needed which are expressive enough to implicitly represent similarities across heterogeneous SWS
Refining SWS ontologies through multiple Conceptual Spaces (CS), i.e. multidimensional, geometrical vector spaces

- Concept C in one SWS ontology $O$ => Conceptual Space CS
- Instance $I$ of $C$ => member $M$ (vector) in CS
- Similarity-computation between SWS instances by means of spatial distances in CS
- Common agreement at schema (i.e. CS) level
- Facilitated through wide-spread use of upper-level ontologies (DOLCE, SUMO…)

Two-fold Approach
Refining SWS through Conceptual Spaces

Agent 1
SWS Ontology O₁

Concept \( c_{1x} \)

instance-of

Instance \( i_{1i} \)

refined-as-member

Agent 2
SWS Ontology O₂

Concept \( c_{2x} \)

instance-of

Instance \( i_{2i} \)

refined-as-member

Conceptual Space \( CS_x \)
CS ontology enabling to represent SWS concepts / instances through CS:

- CS (refining concept C) represented through set of dimensions $d_i$ each associated with a prominence value $p_i$: \[ CS^n = \left\{ p_1 d_1, p_2 d_2, \ldots, p_n d_n \right\} d_i \in CS, p_i \in P \],

- Assignment of measurement scales

- Dimension $d_j$ might be refined by further dimensions:
  \[ d_j = D^n = \left\{ p_1' d_1', p_2' d_2', \ldots, p_n' d_n' \right\} d_k' \in D \] (CS possibly composed of subspaces),

Two-fold Approach
Formal CS Ontology
CS ontology enabling to represent SWS concepts / instances through CS:

- CS (refining concept C) represented through set of dimensions $d_i$ each associated with a prominence value $p_i$: 
  \[ CS^n = \{ (p_1d_1, p_2d_2, ..., p_n d_n) | d_i \in CS, p_i \in P \} , \]

- Assignment of measurement scales

- Dimension $d_j$ might be refined by further dimensions:
  \[ d_j = D^n = \{ (p'_1 d'_1, p'_2 d'_2, ..., p'_n d'_n) | d'_k \in D \} \] (CS possibly composed of subspaces),

- Members $M$ (refining instances I) represented through vectors in CS:
  \[ M^n = \{ (v_1, v_2, ..., v_n) | v_i \in M \} \]

- Similarity between two SWS instances (members) $V$ and $U$ calculated by means of their Euclidean distance:
  \[
  dist(u, v) = \sqrt{\sum_{i=1}^{n} p_i \left( \frac{u_i - \bar{u}}{s_u} - \frac{v_i - \bar{v}}{s_v} \right)^2}
  \]

  (where $\bar{u}$ is the mean of a dataset $U$ and $s_u$ is the standard deviation from $U$)
A Similarity-based Mediator Implementation based on WSMO

- Similarity-based SWS discovery based on Web Service Modelling Ontology (WSMO) and SWS reasoning environment IRS-III
- Similarity-based SWS discovery based on Web Service Modelling Ontology (WSMO) and SWS reasoning environment IRS-III

- WSMO Mediator: computation of similarities between given request (WSMO Goal, $G_1$) and set of $x$ associated SWS ($SWS_1..SWS_x$)

- Implemented through mediation Web service
- MedWS $SWS_{1,1}$ computes $x$ similarity values with $Sim(G_1, SWS_j)$ defined as reciprocal to the mean value of their individual member distances:

$$Sim(G_i, SWS_j) = \left( Dist(G_i, SWS_j) \right)^{-1} = \left( \frac{\sum_{k=1}^{n} (dist_k)}{n} \right)^{-1}$$

- With $dist_k$ being the distance between one particular member $v_i$ of $G_i$ and one member of $SWS_j$ in the same CS.
SWS Mediation based on CS
Prototypical Application

- Uses representational approach based on CS and similarity-based WSMO Mediator
- Aims at retrieval of distributed video resources
- Keyword-based searches across 5 (REST) Web services built on top of the following (logical/physical) repositories
  - YouTube (entertainment feed) [ http://www.youtube.com ]
  - BBC Backstage (world news feed) [ http://backstage.bbc.co.uk/ ]
  - Open Video [ http://www.open-video.org/ ]
  - OU channel on YouTube [ http://www.youtube.com/ou ]
  - YouTube (mobile feed) [ http://www.youtube.com/ou ]
- Similarity-based service discovery for a given request
SWS Mediation based on CS
Prototypical Application

SWS\(_6\): get-video-request
\[ M_{6_1} = \{v_1, v_2, v_3\} \quad M_{6_2} = \{v_1, v_2\} \]

CS\(_1\) Purpose Space

CS\(_2\) Environment Space

SWS\(_1\): OU-youtube

SWS\(_2\): entertain-youtube

SWS\(_3\): open-video

SWS\(_4\): bbc-backstage

SWS\(_5\): mobile-youtube

WS\(_1\): OU-youtube

WS\(_2\): entertain-youtube

WS\(_3\): open-video

WS\(_4\): bbc-backstage

WS\(_5\): mobile-youtube
SWS Mediation based on CS
Prototypical Application

SWS_6:
get-video-request
M_{6_1} = \{v_1, v_2, v_3\}
M_{6_2} = \{v_1, v_2\}

CS_1 Purpose Space

CS_2 Environment Space

O_1::Purp  O_1::Env
SWS_1:
OU-youtube

O_2::Purp  O_2::Env
SWS_2:
entertain-youtube

O_3::Purp  O_3::Env
SWS_3:
open-video

O_4::Purp  O_4::Env
SWS_4:
bbc-backstage

O_5::Purp  O_5::Env
SWS_5:
mobile-youtube

WS_1:
OU-youtube

WS_2:
entertain-youtube

WS_3:
open-video

WS_4:
bbc-backstage

WS_5:
mobile-youtube
SWS Mediation based on CS Prototypical Application

CS1 Purpose Space
- SWS1: OU-youtube
- O1:Purp
- O1:Env

CS2 Environment Space
- SWS2: entertain-youtube
- O2:Purp
- O2:Env
- SWS3: open-video
- O3:Purp
- O3:Env
- SWS4: bbc-backstage
- O4:Purp
- O4:Env
- SWS5: mobile-youtube
- O5:Purp
- O5:Env

\[ (p_1^{\text{information}}, p_2^{\text{education}}, p_3^{\text{leisure}}) = CS_1 \]
\[ (p_4^{\text{resolution}}, p_5^{\text{bandwidth}}) = CS_2 \]
SWS Mediation based on CS
Prototypical Application

- SWS annotated with *Assumption* $Ass_{SWSi}$ defined through conjunction of instances
- Instances refined through vectors (members) as follows

<table>
<thead>
<tr>
<th>SWS</th>
<th>Members $P_i$ in CS$_1$ (purpose)</th>
<th>Members $E_j$ in CS$_2$ (environment)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWS$_1$</td>
<td>$P_{1(SWS1)}$ = {(0, 100, 0)}</td>
<td>$E_{1(SWS1)}$ = {(100, 100)}</td>
</tr>
<tr>
<td>SWS$_2$</td>
<td>$P_{1(SWS2)}$ = {(0, 0, 100)}</td>
<td>$E_{1(SWS2)}$ = {(100, 100)}</td>
</tr>
<tr>
<td>SWS$_3$</td>
<td>$P_{1(SWS3)}$ = {(50, 50, 0)}</td>
<td>$E_{1(SWS3)}$ = {(100, 100)}</td>
</tr>
<tr>
<td>SWS$_4$</td>
<td>$P_{1(SWS4)}$ = {(100, 0, 0)}</td>
<td>$E_{1(SWS4)}$ = {(100, 100)}</td>
</tr>
<tr>
<td>SWS$_5$</td>
<td>$P_{1(SWS5)}$ = {(100, 0, 0)}</td>
<td>$E_{1(SWS5)}$ = {(10, 10)}</td>
</tr>
</tbody>
</table>
- SWS annotated with *Assumption* $Ass_{SWS_i}$ defined through conjunction of instances.
- Instances refined through vectors (members) as follows:

<table>
<thead>
<tr>
<th>SWS</th>
<th>Members $P_i$ in CS$_1$ (purpose)</th>
<th>Members $E_j$ in CS$_2$ (environment)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWS$_1$</td>
<td>$P_{1(SWS1)} = {(0, 100, 0)}$</td>
<td>$E_{1(SWS1)} = {(100, 100)}$</td>
</tr>
<tr>
<td>SWS$_2$</td>
<td>$P_{1(SWS2)} = {(0, 0, 100)}$</td>
<td>$E_{1(SWS2)} = {(100, 100)}$</td>
</tr>
<tr>
<td>SWS$_3$</td>
<td>$P_{1(SWS3)} = {(50, 50, 0)}$</td>
<td>$E_{1(SWS3)} = {(100, 100)}$</td>
</tr>
<tr>
<td>SWS$_4$</td>
<td>$P_{1(SWS4)} = {(100, 0, 0)}$</td>
<td>$E_{1(SWS4)} = {(100, 100)}$</td>
</tr>
<tr>
<td>SWS$_5$</td>
<td>$P_{1(SWS5)} = {(100, 0, 0)}$</td>
<td>$E_{1(SWS5)} = {(10, 10)}$</td>
</tr>
</tbody>
</table>
SWS Mediation based on CS
Prototypical Application

- Requests defined through AJAX-based UI
- Requests consists of:
  - Input parameters: set of keywords (free text)
  - Assumption: defined through vectors
    (measurements describing purpose and environment)
- Similarity-based discovery of most suitable service based on WSMO mediator
DEMO

SWS_6: get-video-request
M_{S_1} = \{v_1, v_2, v_3\}  
M_{S_2} = \{v_1, v_2\}

CS_1 Purpose Space  CS_2 Environment Space

O_1: Purp  O_1: Env
SWS_1: OU-youtube

O_2: Purp  O_2: Env
SWS_2: entertain-youtube

O_3: Purp  O_3: Env
SWS_3: open-video

O_4: Purp  O_4: Env
SWS_4: bbc-backstage

O_5: Purp  O_5: Env
SWS_5: mobile-youtube

WS_1: OU-youtube
WS_2: entertain-youtube
WS_3: open-video
WS_4: bbc-backstage
WS_5: mobile-youtube
Some issues:

- Mediation/matchmaking algorithm entirely based on instance similarities
  => combination of similarity-based and logical-based approach required

- Additional representational effort

- => CS might just shift symbol grounding issue
  (i.e. dimensions lack grounding and are ambiguous)

- Similarity-calculation requires overlapping CS and measurable quality dimensions
Some issues:

- Mediation/matchmaking algorithm entirely based on instance similarities
  => combination of similarity-based and logical-based approach required

- Additional representational effort

- => CS might just shift symbol grounding issue
  (i.e. dimensions lack grounding and are ambiguous)

- Similarity-calculation requires overlapping CS and measurable quality dimensions

..., however:

- Allows to compute similarities between distinct SWS

- Reduces the required level of common ontological agreement
  (only required at schema level).
Thank you!

E-mail: s.dietze@open.ac.uk
Web: http://people.kmi.open.ac.uk/dietze