Semantics Service Composition Using Conceptual Graph for Addressing Imprecise Service Requirements

William Song

Computer Science

Durham University, UK

w.w.song@durham.ac.uk
Introduction

- Web Services has become an important research topic in the fields of the Service-Oriented Architecture (SOA).
- Automatic or semi-automatic service discovery, invocation and composition techniques are on demand.
- The Semantic Web Services seems to be the most promising way towards achieving automatic or semi-automatic service discovery, invocation, and composition.
Problems

• **Insufficient usage context information**: The current work are focusing on ontology based data type semantics and do not sufficiently address how a service is fitted into its usage context.

• **Precise requirements required to locate services**: In order to locate the required services, the current work requires precise service requirements which are difficult to be specified at the preliminary stage of the service discovery.

• **Insufficient information about inter-relationship among service**: The current work has not addressed the inter-relationships among services sufficiently, which makes the service discovery in an isolated manner.
We try to describe services by

- using the inter-service relationships.
- Using the contextual information.
- so easing service composition process.
Our proposal

- A Context-based Semantic Description Framework (CbSDF).
  - To describe services by the usage context aspect using Conceptual Graphs and Spider Model.
  - To use non-monotonic rules to describe the pre-conditions and effects of services and the conditions for service composition.
  - To search for services based on imprecise service requirements.
Example: Learning Resources

- A learning flow with learning resource specifications, but not the physical resources.
- Learning resources located dynamically at learning time based on the specifications.
- The diagram illustrates a learning course with learning resource specifications. The dotted lines represent the links dynamically established at learning time.
Context-based Semantic Description Framework (CbSDF)

- The proposed CbSDF consists of four components:
  - Definitions of atomic and composite services
    - By having clear definitions of atomic service and composite service, we can identify what kind of information is relevant to describing a service.
  - Service Conceptual Graphs
    - Give an overall and abstract description of the relationships between services and their related concepts.
  - Semantic Service Description Model (Spider Model)
    - Semantically describes service itself and the relations with other services.
  - Non-monotonic Rules
    - Describe the pre-conditions and effects of services and the conditions for service composition.
Conceptual Graphs

- A conceptual graph (CG) is a finite, connected, bipartite graph with nodes of one type called concepts and nodes of the other type called conceptual relations.
  - The label of a concept node consists of two fields separated by a colon, \([\text{type}: \text{referent}]\) i.e. \([\text{class}: \text{instance}]\).
  - Conceptual Relations represent the relationships between concept nodes.

- Projection of CG
  - \(\pi : v \rightarrow u\), where \(\pi_u v\) is a sub-graph of \(u\) called a projection of \(v\) in \(u\). \(\pi\) is called the projection operator. \(v\) describes a more generalised concept than \(u\), \(u \preceq v\).
  - Projection concept is important in CG matching and reasoning.
Dependency Graph of Learning Concept

- A learning concept dependency graph $G_d$ is a CG where the concept type is restricted to concepts within the Learning Object ontology

  $G_d = < C, R, E >, \text{type}(C) \in O$

  - $C$: a set of learning concept nodes; $\text{type}(C)$ returns a set of leaf node concepts in the Learning Object ontology.
  - $R$: a set of relation nodes that represent the relations among learning concept nodes, including pre-requisite relation type and conceptual relation type etc.
  - $E$: a set of arcs that associate relation nodes with concept nodes.
  - $O$: the Learning Object ontology.

- An example is illustrated in the diagram

  ![Diagram](attachment:image.png)
Four Types of Semantics in Web Services

- **Data Semantics**
  - Formal definition of data in Input and output message.

- **Functional Semantics**
  - Formal definition of the web service capability.

- **Non-functional Semantics**
  - Formal definition of quantitative or non-quantitative constraints.

- **Execution Semantics**
  - Formal definition of execution flow of services of a process or of operations within a service.
Graphical Illustration of SSDM

- The notations used in the diagram are:
  - $S_i = \text{Service}_i$
  - Service’ can be either the parent or ancestor of Service$_i$
  - $S_1, S_2, S_3,$ and $S_4$ are any services.
  - $I, P$ is the Inputs and Pre-condition, and $O, E$ is the Outputs and Effects.
Learning Objects Ontology

- The ontology represents the concepts of Learning Objects.
- Based on the ACM/IEEE Computing Curriculum.
- Three levels: Area, Unit, and Topic.
- The leaf node of the ontology is a course or part of a course that can be directly taken by learners.
Non-monotonic Rules

• Reason for using non-monotonic rules
  – Handling unpredictable situation in a open service repository.
  – Exception handling.
• The non-monotonic rules are described using Defeasible Logic. A defeasible theory $DT$ is a triple:
  
  $$ DT = (F, R, >) $$

  – $F$: a set of facts;
  – $R$: a finite set of rules;
  – $>\!$: a superiority relation on $R$.
• The rules are divided into two categories:
  – General rules
  – Domain specific rules
• The rules are used in two ways:
  – Describe services pre-conditions and effects.
  – Validate service composition results: Trigger-able validation and Compose-able validation.
General Rules

- The general rules are normally used to construct and validate composite services and they are applicable to all the services, for example:
  - \( r1 \): if a service’s pre-condition is satisfied, then normally it can be executed.
    \[
    \text{satisfy}(S.\text{preCon}) \Rightarrow \text{executable}(S)
    \]
  - \( r2 \): if a service is not available, then definitely it cannot be executed.
    \[
    \neg \text{available}(S) \rightarrow \neg \text{executable}(S)
    \]
  - \( r3 \): if two services are composed through input and output data flow, then normally the data types of the input and output are compatible, i.e. one is a same or sub-type of the other.
    \[
    \text{composable}(S_1, S_2) \Rightarrow \text{type}(S_1.\text{Ipt}) \leq \text{type}(S_2.\text{Opt})
    \]
Domain Specific Rules

- The domain specific rules are normally used to describe the pre-conditions and effects of services and can only be applied to a specific domain, for example:
  - $r1$: if the service is supplied with a valid postcode, then normally the correct result will be returned.
    \[
    \text{valid (postcode)} \Rightarrow \text{result}(S)
    \]
  - $r2$: if the requested address is in UK, then this service is definitely applicable.
    \[
    \text{location (UK)} \Rightarrow \text{applicable}(S)
    \]
Two-Step Service Discovery Mechanism

• The first step is preliminary service discovery step using the CG matching technique.
  – Requirement $\rightarrow$ CG
  – Match with Service Conceptual Graphs

• The second step, validation and ranking step, is to refine the results from the first step based on the service requirements, the SSDM, and the non-monotonic rules.
CG Similarity Calculation

- A CG similarity $Sim$ is calculated through concept nodes similarity $S_c$ and relation node similarity $S_r$.

$$S_c = 2 \left( \sum_{c \in O} \text{weight}(c) \times \beta(\pi_{G_1} c, \pi_{G_2} c) \right) \div \left( \sum_{c \in G_1} \text{weight}(c) + \sum_{c \in G_2} \text{weight}(c) \right)$$

$$S_r = \frac{2m(G_c)}{m_{G_1}(G_1) + m_{G_2}(G_2)}$$

- $O \cup$ is the union of all of the common generalisation graphs of $G_1$ and $G_2$.
- $\beta(\pi_{G_1} c, \pi_{G_2} c)$ is a function to calculate the semantic similarity between two concepts.
- $m(G_c)$ is the number of the relation nodes in the common overlaps of $G_1$ and $G_2$.
- $\pi_{G_i} c$ is the number of the relation nodes of the common overlaps in $G_i$ and the overlaps' adjacent relation nodes.
- $a$ is a value between 0 and 1 representing the impact factor of $S_r$, which make sure that the overall similarity $Sim$ will not be 0 unless both $S_c$ and $S_r$ are 0.
Semantic Similarity Ranking

- In the second step of the service discovery, according to the service requirement and the SSDM, the similarities between the services and the requirement are calculated.

\[
sim(R, S) = \frac{\sum \omega \times \text{dist}(\alpha(R), \alpha(S))}{\max(\lambda(R), \lambda(S))}
\]

- \(\lambda\): a set of all the semantic characteristics functions.
- \(\lambda()\): a function that returns the number of semantic characteristics.
- \(\alpha()\): an element of \(\lambda\) that returns a semantic characteristic which can be, e.g. an element of the metadata in the SSDM or the inputs and outputs of a service.
- \(\text{dist}()\): a function that calculate the semantic distance between two semantic characteristic and its returned value is between 0 and 1.
- \(\omega\): a weight factor that specifies how important a semantic characteristic to a learner is and its value is between 0 and 1.
- \(\max()\): a function returns the greater of its two arguments values.
- \(R\) and \(S\): the service requirement and a candidate service.
Semantic Distance Calculation

Methods

• Tree Based Similarity
  
  - the semantic similarity between two topics in a ontology is defined as a function of the meaning shared by the topics and the meaning of each of the individual topics.

  \[
  \begin{align*}
  \text{sim}(p,q) &= \begin{cases} 
  1 & \text{if type}(p) = \text{type}(q) \text{ and } \text{instance}(p) = \text{instance}(q) \\
  \text{depth}/(\text{depth} + 1) & \text{if type}(p) = \text{type}(q) \text{ and } \text{instance}(p) \neq \text{instance}(q) \\
  2d_e/(d_p + d_q) & \text{if type}(p) \neq \text{type}(q)
  \end{cases}
  \end{align*}
  \]

• Semantic Cosine Similarity
  
  - Two items \(i_p\) and \(i_q\) are considered as two column vectors in the user requirement matrix. The similarity between items is measured by computing the cosine of these two vectors.

  \[
  \text{sim}(i_p, i_q) = \cos(i_p, i_q) = \frac{i_p \bullet i_q}{\sqrt{||i_p|| \ast ||i_q||}}
  \]
Conclusion

• A Context-based Semantic Description Framework (CbSDF) is proposed for service description and a two-step service discovery mechanism for service search.

• Aiming to provide a service description framework and a search mechanism that can tolerant imprecise specified service requirements.

• The key technologies used to capture the semantics from imprecise requirements and validate the service discovery results are the CG and the non-monotonic logic, i.e. Defeasible Logic.

• Continue future research on CG and non-monotonic rules in order to improve service description, discovery, and composition techniques.

• Design a suitable evaluation model to evaluate our work.