Goal-based Visualization and Browsing for Semantic Web Services

Michael Stollberg and Mick Kerrigan

Digital Enterprise Research Institute (DERI),
University of Innsbruck, Austria
{firstname.lastname}@deri.at

Abstract. We present a goal-based approach for visualizing and browsing the search space of available Web services. A goal describes an objective that a client wants to solve by using Web services, abstracting from the technical details. Our visualization technique is based on a graph structure that organizes goal templates – i.e. generic and reusable objective descriptions – with respect to their semantic similarity, and keeps the relevant knowledge on the available Web services for solving them. This graph is generated automatically from the results of semantically enabled Web service discovery. In contrast to existing tools that categorize the available Web services on the basis of certain description elements, our tool allows clients to browse available Web services on the level of problems that can be solved by them and therewith to better understand the structure as well as the available resources in a domain. This paper explains the theoretic foundations of the approach and presents the prototypical implementation within the Web Service Modeling Toolkit WSMT, an Integrated Development Environment for Semantic Web services.

1 Introduction

The provision of suitable search facilities for Web services is one of the major challenges for realizing sophisticated SOA technologies. One desirable feature are browsing facilities that support Web service application developers in the search and inspection of potential candidate services for a specific problem. Existing tools for this mostly follow the registry approach already defined in UDDI [3]: Web services are categorized with respect to certain description elements, and graphical tools support the search and browsing of these registries.

We take a different approach for visualizing the search space of available Web services. Our technique is based on goal templates as generic and reusable descriptions of objectives that clients want to achieve by using Web services. We organize them in a graph structure with respect to their semantic similarity, and we keep knowledge on the suitability of the available Web services for solving the goals that is obtained from Web service discovery runs. Our graphical user interface visualizes this graph and provides browsing facilities for this. This is a novel approach that allows clients to browse and understand the available Web services on the level of the problems that can be solved by them.

The basis for our search space visualization is the *Semantic Discovery Caching* technique (short: SDC), a caching mechanism for enhancing the computational performance of automated Web service discovery engines [14]. Its heart is the so-called SDC graph that organizes goal templates in a subsumption hierarchy and captures knowledge on the suitability of the available Web services from Web service discovery results. The SDC graph is generated automatically by semantic matchmaking of sufficiently rich formal descriptions of goals and Web services. It provides an index structure for the efficient search of suitable Web services, and thus serves as the data structure of our search space visualization. The graphical user interface is implemented in the Web Service Modelling Toolkit WSMT [9], an Integrated Development Environment for the Semantic Web service technology developed around the WSMO framework [5].

This paper explains the theoretic foundations of our visualization technique and presents the prototype implementation. Section 2 recalls the idea of the goal-based approach for Semantic Web services. Section 3 explains the structure, definition, and properties of the SDC graph, and Section 4 presents the visualization and browsing support developed in WSMT. Section 5 discusses the approach and positions it within related work, and Section 6 concludes the paper. For illustration and demonstration, we use the shipment scenario from the SWS Challenge, a widely recognized initiative for the demonstration of SWS techniques (see `www.sws-challenge.org`).

2 The Goal-based Approach for Semantic Web Services

In order to introduce into the overall context, the following explains the aim of the goal based approach for Semantic Web services as promoted by the WSMO framework and recalls the foundations of our approach from previous works.

While the initial Web service technology stack as well as most approaches in the field of Semantic Web services (SWS) only pay little attention to the client side of SOA technology, the WSMO framework promotes a goal driven approach for this [5]. Therein, a goal is the formal description of an objective that a client wants to achieve by using Web services. Goals focus on the problem to be solved, abstracting from technical details on how to invoke a Web service. The overall aim is to facilitate problem-oriented Web service usage: the client merely specifies the objective to be achieved as a goal, and the system automatically discovers, composes, and executes the necessary Web services for solving this. Therewith, goals shall allow to lift the client-system interaction to the knowledge level in the tradition of previous AI technologies for automated problem solving.

Figure 1 provides an overview of the goal-based SWS framework that has been developed throughout several works around WSMO [7,16,15,18,14]. This distinguishes *goal templates* as generic and reusable objective descriptions which are stored in the system, and *goal instances* that denote concrete client requests and are defined by instantiating a goal template with concrete inputs. On this basis, we can separate *design time* and *runtime* operations. At design time, suitable Web services for goal templates are detected. For simplification, we here use

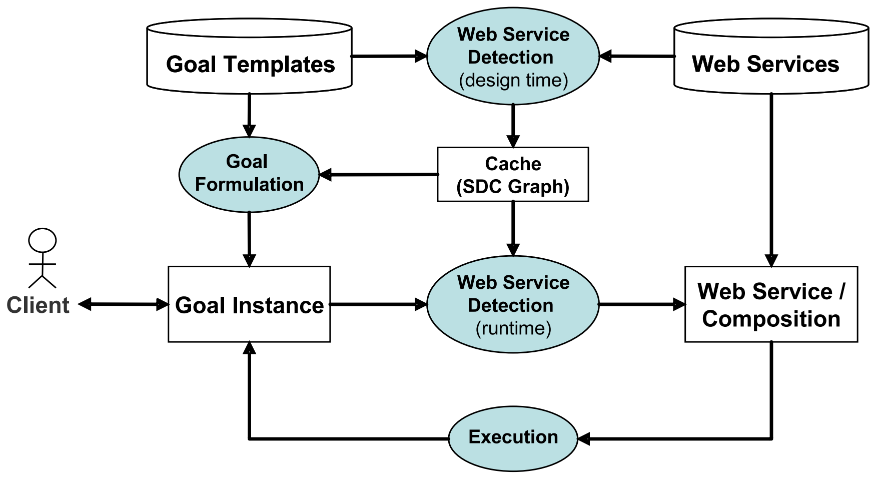

Goal Templates
Web Service Detection (design time)
Web Services
Goal Formulation
Cache (SDC Graph)
Client
Goal Instance
Web Service Detection (runtime)
Web Service / Composition
Execution

Fig. 1. Overview of Goal-based SWS Framework

'Web service detection' as a general term for SWS discovery, selection, ranking, and composition techniques. The result is captured in the SDC graph which provides an index for the efficient search of goal templates and Web services (see Section 3). At runtime, a client defines a concrete objective in terms of a goal instance, and the captured knowledge is used to detect the suitable Web services in an efficient manner. Finally, the detected Web service or composition is executed and the result is reported to the client.

For illustration, let us consider the following setting from the shipment scenario as described in the SWS Challenge. A client wants to ship a package of 40 lb weight from New York City to Bristol in the UK, and there are several available Web services that offer package shipment. In our framework, a goal template describes the objective of package shipment on the schema level, e.g. that the sender is located in a US city and the receiver in a European city, and the maximal weight class (e.g. from 0 - 1.5 lb, 1.5 - 10 lb, etc.). Let the client create a goal instance for this goal template by defining the concrete input data. To determine the suitable Web services for solving the goal instance, the system can make use of the knowledge on design time Web service detection results. This allows to develop efficient and workable SWS technologies. For example, in this specific use case there are only 4 Web services out of all available ones that are suitable to solve the goal template. One of these must be chosen to be executed for solving the goal instance: this requires a detailed investigation at runtime of only 4 Web services instead of all the available ones.

Summarizing, goal instances are the primary element for clients to interact with the system, i.e. end-users who want to use Web services to solve a certain task. The aim of the visualization and browsing technique presented in this paper is to aid clients in the goal instance formulation process as well as to provide graphical support for better understanding the available resources.

3 The Semantic Discovery Caching Technique

The following resumes the *Semantic Discovery Caching* technique (short: SDC), a caching mechanism for enhancing the computational efficiency of Web service discovery [14]. Its heart is the SDC graph that provides an automatically generated index structure for efficient search of goals and Web services, which serves as the basis for the search space visualization presented in Section 4.

The main purpose of the SDC technique is to increase computational performance of the Web service discovery task, which is considered as the first processing step for detecting the Web services that are suitable for solving a goal [12]. It captures design time discovery results for goal templates in the SDC graph, and effectively uses this knowledge in order to minimize the computational costs of Web service discovery for goal instances at runtime. The SDC graph organizes goal templates in a subsumption hierarchy, and the leaf nodes represent the available Web services whose suitability for solving goal templates is explicated by directed arcs. Therewith, the SDC graph provides a directed graph structure that describes the usability of Web services in a problem domain with respect to the goals that can be solved by them. The following explains the structure of the SDC graph and the underlying semantic matchmaking techniques.

3.1 Formal Functional Descriptions and Semantic Matchmaking

The SDC technique is based on the Web service discovery approach presented in [15] that performs semantic matchmaking on the goal template and the goal instance level on the basis of rich functional descriptions.

A functional description $\mathcal{D} = (\Sigma, \Omega, IN, \phi^{pre}, \phi^{eff})$ is defined over a signature Σ with respect to a domain ontology Ω; IN is the set of input variables, ϕ^{pre} is the precondition, and ϕ^{eff} is the effect wherein the predicate $out()$ denotes the outputs. Such a functional description precisely describes the possible executions $\{\tau\}_W$ of Web services, respectively the possible solutions $\{\tau\}_G$ of goal templates with respect to the start- and the end-states. We denote the functional usability of a Web service W for a goal template G by the matchmaking degrees shown in Table 1; therein, $\phi^{\mathcal{D}}$ is a FOL-formula of the form $\phi^{pre} \Rightarrow \phi^{eff}$ that defines the formal semantics of $\mathcal{D}$ as an implication between the possible start- and the possible end-states. The first four degrees distinguish different situations wherein W is usable to solve G, and the *disjoint* degree states that this is not given.

A goal instance is defines as a pair $GI(G) = (G, \beta)$ with G as the corresponding goal template and β as the input binding for the input variables IN defined in the functional description $\mathcal{D}_G$. A goal instance must be a valid instantiation of the corresponding goal template in order to be a consistent objective description. This is given if $\mathcal{D}_G$ is satisfiable under the input binding β. Then it holds that $\{\tau\}_{GI(G)} \subset \{\tau\}_G$, meaning that the solutions of the goal instance are a subset of those for its corresponding goal template. In consequence, only those Web services that are suitable for solving the goal template G are potential candidates for every of its goal instances $GI(G)$ and no others can be. This allows to effectively distinguish design- and runtime operations as explained above.

Table 1. Definition of Matching Degrees for $\mathcal{D}_G$, $\mathcal{D}_W$

Denotation	Definition	Meaning
$\textbf{exact}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \forall \beta.\ \phi^{\mathcal{D}_G} \Leftrightarrow \phi^{\mathcal{D}_W}$	$\{\tau\}_G = \{\tau\}_W$
$\textbf{plugin}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \forall \beta.\ \phi^{\mathcal{D}_G} \Rightarrow \phi^{\mathcal{D}_W}$	$\{\tau\}_G \supset \{\tau\}_W$
$\textbf{subsume}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \forall \beta.\ \phi^{\mathcal{D}_G} \Leftarrow \phi^{\mathcal{D}_W}$	$\{\tau\}_G \subset \{\tau\}_W$
$\textbf{intersect}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \exists \beta.\ \phi^{\mathcal{D}_G} \wedge \phi^{\mathcal{D}_W}$	$\{\tau\}_G \cap \{\tau\}_W \neq \emptyset$
$\textbf{disjoint}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \neg\exists \beta.\ \phi^{\mathcal{D}_G} \wedge \phi^{\mathcal{D}_W}$	$\{\tau\}_G \cap \{\tau\}_W = \emptyset$

3.2 The SDC Graph – Structure and Definition

The SDC graph is automatically generated from the results of design time Web service discovery on goal templates. It organizes goal templates in a subsumption hierarchy with respect to their semantic similarity, which constitutes the indexing structure of the available Web services. The leaf nodes represent the Web services that are functionally usable for solving the goal templates.

We consider two goal templates G_i and G_j to be similar if they have at least one common solution. Then, mostly the same Web services are usable for them. We express this in terms of *similarity degrees* $d(G_i, G_j)$; defined analog to Table 1, they denote the matching degree between the functional descriptions $\mathcal{D}_{G_i}$ and $\mathcal{D}_{G_j}$. The SDC graph is defined such that the only occurring similarity degree is $subsume(G_i, G_j)$. This allows efficient search because then (1) the solutions for the child G_j are a subset of those for the parent G_i, and thus (2) the Web services that are usable for G_j are a subset of those usable for G_i.

In consequence, the SDC graph consists of two layers. The upper one is the *goal graph* that defines the subsumption hierarchy of goal templates by directed arcs. This constitutes the index structure of the available Web services with respect to the goals that can be solved by them, whereby the subsumption hierarchy represents the specialization in a problem domain. The lower layer is the *usability cache* that explicates the usability of each available Web service W for every goal template G by directed arcs that are annotated with the usability degree $d(G, W)$. This is generated from the results of Web service discovery on the goal template level that is performed at design time. The discovery operations use this knowledge structure by inference rules of the form $d(G_i, G_j) \wedge d(G_i, W) \Rightarrow d(G_j, W)$ that result from the formal definitions.

Figure 2 illustrates the SDC graph for our running example along with the most relevant inference rules. There are three goal templates: G_1 for package shipment in Europe, G_2 for Switzerland, and G_3 for Germany. Their similarity degrees are $subsume(G_1, G_2)$ and $subsume(G_1, G_3)$, which is explicated in the goal graph. Consider some Web services, e.g. W_1 for package shipment in Europe, W_2 in the whole world, W_3 in the European Union, and W_4 in the Commonwealth. Their usability degree for each goal template is explicated in the usability cache, whereby redundant arcs are omitted: the usability degree of W_1 and W_2 for G_2 and G_3 can be directly inferred, thus the arcs are omitted.

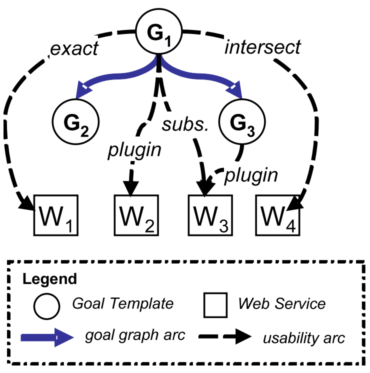
Structure of SDC Graph	Inference Rules for $subsume(G_i, G_j)$
 G1 exact intersect G2 subs. G3 plugin plugin W1 W2 W3 W4 Legend Goal Template Web Service goal graph arc usability arc	$$\begin{array}{ll} (1)\ exact(G_i, W) & \Rightarrow plugin(G_j, W). \\ (2)\ plugin(G_i, W) & \Rightarrow plugin(G_j, W). \\ (3)\ subsume(G_i, W) & \Rightarrow exact(G_j, W) \vee \\ & \quad plugin(G_j, W) \vee \\ & \quad subsume(G_j, W) \vee \\ & \quad intersect(G_j, W) \vee \\ & \quad disjoint(G_j, W). \\ (4)\ intersect(G_i, W) & \Rightarrow plugin(G_j, W) \vee \\ & \quad intersect(G_j, W) \vee \\ & \quad disjoint(G_j, W). \\ (5)\ disjoint(G_i, W) & \Rightarrow disjoint(G_j, W). \end{array}$$

Fig. 2. Example of a SDC Graph and Inference Rules

The SDC graph provides an index structure of the search space of the available Web services with respect to the goals that can be solved by them. It can be automatically generated, and – in contrast to most existing clustering techniques for Web services – it is based on semantic matchmaking that ensures a high accuracy of the obtained graph structure. The generation and maintenance is supported by algorithms that ensure that the SDC graph exposes its properties at all times. We refer to [14] for details on this, in particular for the resolution of intersection matches in the goal graph which is necessary to ensure that the only similarity degree is *subsume*, *cf.* clause (i) in the following definition.[1]

Definition 1. *Let $d(G_i, G_j)$ denote the similarity degree of goal templates G_i and G_j, and let $d(G, W)$ denote the usability degree of a Web service W for a goal template G. Given a set $\mathcal{G}$ of goal templates and a set $\mathcal{W}$ of Web services, the SDC graph is a directed acyclic graph $(V_{\mathcal{G}} \cup V_{\mathcal{W}}, E_{sim} \cup E_{use})$ such that:*

(i) $V_{\mathcal{G}} := \mathcal{G} \cup \mathcal{G}^I$ is the set of inner vertices where:
- *$\mathcal{G} = \{G_1, \ldots, G_n\}$ are the goal templates; and*
- *$\mathcal{G}^I := \{G^I \mid G_i, G_j \in \mathcal{G}, d(G_i, G_j) = \mathsf{intersect}, G^I = G_i \cap G_j\}$ is the set of intersected goal templates from $\mathcal{G}$*

(ii) $V_{\mathcal{W}} := \{W_1, \ldots, W_m\}$ is the set of leaf vertices representing Web services

(iii) $E_{sim} := \{(G_i, G_j) \mid G_i, G_j \in V_{\mathcal{G}}\}$ is the set of directed arcs where:
- *$d(G_i, G_j) = \mathsf{subsume}$; and*
- *not exists $G \in V_{\mathcal{G}}$ s.t. $d(G_i, G) = \mathsf{subsume}$, $d(G, G_j) = \mathsf{subsume}$.*

(iv) $E_{use} := \{(G, W) \mid G \in V_{\mathcal{G}}, W \in V_{\mathcal{W}}\}$ is set of directed arcs where:
- *$d(G, W) \in \{\mathsf{exact}, \mathsf{plugin}, \mathsf{subsume}, \mathsf{intersect}\}$; and*
- *not exists $G_i \in V_{\mathcal{G}}$ s.t. $d(G_i, G) = \mathsf{subsume}$, $d(G_i, W) \in \{\mathsf{exact}, \mathsf{plugin}\}$.*

[1] The SDC prototype is open source software available from the SDC homepage at `members.deri.at/~michaels/software/sdc/`. It is realized as a discovery component in the WSMX system (the WSMO reference implementation, `www.wsmx.org`). We use VAMPIRE for matchmaking, a FOL automated theorem prover.

4 Visualization and Browsing

We now turn towards the visualization and browsing support for the search space of Web services. The aim is to provide a graphical representation that allows clients to better understand the available Web services as well as the problems that can be solved by them. This occurs to be desirable in order to determine which tasks in a client application can be solved by the Web services. Moreover, the graphical user interface supports clients in the selection of an appropriate goal template for expressing a specific objective or request, therewith providing graphical support for the goal instance formulation process (see Section 2).

The search space visualization uses a SDC graph as the data structure. As explained above, this provides an index structure of Web services on the level of goal templates that is obtained by semantic matchmaking and thus exposes a high accuracy of the relevant relationships of goals and Web services. The SDC graph visualization is implemented as a new plug-in for the Web Service Modeling Toolkit WSMT [9], an Integrated Development Environment (IDE) implemented in the Eclipse framework that provides the graphical user- and developer tools for the Semantic Web service technologies developed around the WSMO framework. In particular, we extend the WSMT Visualizer [8] that provides a graph-based editor and browser for ontologies.[2]

For illustration of the visualization and browsing facilities, we have created the SDC graph for the original data set of the shipment scenario as defined in the SWS Challenge. This defines 5 Web services that offer package shipment from the USA to different destinations in the world, and a collection of exemplary client requests. These correspond to goal instances in our framework, while goal templates define the generic and reusable objective descriptions.[3]

The following presents the technical realization of the SDC graph visualization and browsing in WSMT, and explains the obtained surplus value for clients as well as the integration with SWS environments for automated goal solving by the discovery, composition, and execution of Web services.

4.1 SDC Graph Visualization in WSMT

As explained above, a SDC graph consists of two layers: the upper one is the *goal graph* wherein the existing goal templates are organized in a subsumption hierarchy, and the lower layer is the *usability cache* that explicates the suitability of the available Web services for each goal template (*cf.* Definition 1). The visualization of an SDC graph that is initially presented to the user displays the subsumption hierarchy of the goal graph along with a cluster of the usable Web services for each goal template. This allows to get a complete overview of the

[2] The Web Service Modeling Toolkit is open source software available for download from the sourceforge web site at: `http://wsmt.sourceforge.net`.

[3] The complete resources for this use case are available at `http://members.deri.at/~michaels/software/sdc/resourcesSWSC.zip`; the scenario description is given at `http://sws-challenge.org/wiki/index.php/Scenario:_Shipment_Discovery`.

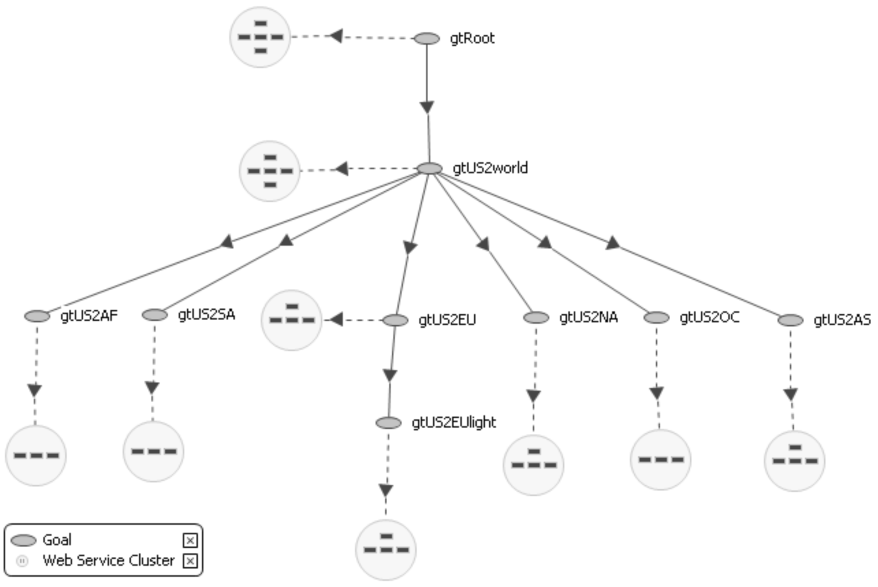


Fig. 3. SDC Graph Visualization in WSMT

available resources. The user can then browse and navigate through the search space via more detailed perspectives as we shall explain below.

Figure 3 shows the visualization of the SDC graph for the shipment scenario in WSMT. The root node of the SDC graph `gtRoot` is a goal template that describes the objective of shipping a package of any weight from anywhere in the world to anywhere in the world. This denotes the most general objective description in the problem domain of package shipment. The goal template `gtUS2world` is concerned with package shipment from the USA to anywhere in the world; this is a child node of `gtRoot` in the goal graph because every solution for `gtUS2world` is also a solution for `gtRoot` but not vice versa (see Section 3.2). Analogously, the child nodes of `gtUS2world` describe the objective of package shipment from the USA to specific continents, the goal templates on the subsequent levels of the goal graph specify local regions and cities, and finally the lowest levels differentiate the weight classes for the packages.

Two interesting properties result from the structure and definition of the SDC graph. At first, the Web service cluster of a child node contains a subset of the Web service cluster of its parent. For example, from the 5 Web services for `gtUS2world` only 3 are usable for the goal template `gtUS2AS` that defines package shipment from the USA to Asia. This reflects the nature of most problem domains: the deeper a goal template is allocated in the goal graph, the more specialized is the described objective, and the fewer Web services are usable for it. Secondly, a SDC graph may contain disconnected subgraphs when there are two goal templates that do not have any common solution. One can understand each connected subgraph to cover a particular problem domain: apart from the one for the shipment scenario, there could be a set of goal templates for the problem domain of ticket booking, and another one that is concerned with purchase order management. Furthermore, it is possible to generate the goal templates for a problem domain from a given goal description and the underlying domain ontologies. For example, in the shipment scenario we can generate `gtUS2world` from `gtRoot` by restricting the molecule for the sender location in the goal description from *world* to *USA*. Eventually, we can generate all goal templates that can be expressed on the basis of the domain ontologies and therewith maximize the granularity of the SDC graph [14].

The SDC graph visualization plug-in reuses the JPowerGraph[4] graphing library developed for the visualization of ontologies and other WSMO elements in WSMT [8]. This provides a powerful framework with a number of layout algorithms for user-facing, interactive graphs with any type of interconnected data. The prototype employs a simple vertical-tree layout; however, a spring-layout algorithm where the nodes in the graph repel each other while the edges between nodes draw them back together can be employed to display larger and more complex SDC graphs. The display of Web service clusters extends the technique for the clustering of ontology instances in larger knowledge bases.

4.2 Browsing Facilities

The SDC graph visualization as explained above is complemented with browsing facilities that allow to inspect goal and Web service descriptions on more fine-grained levels. By double-clicking on a goal template in the SDC graph, the user can step down to the next level wherein the visualization is focused on the selected goal template. This view shows the relevant branch of the goal graph, and the suitable Web services for the selected goal template along with the concrete usability degrees as the disaggregation of respective Web service cluster. Figure 4 below illustrates this for the goal template `gtUS2EU` that corresponds to the introductory example discussed in Section 2. As the next level of detail, the user can browse the specification of individual goal and Web service descriptions by double-clicking on the element in the SDC graph; these facilities already exist in WSMT, and we omit further screen shots here with respect to space limitations. The segmentation of the browsing support into multiple levels allows to manage the complexity of the SDC graph while presenting all relevant information to the user in a browsable fashion.

In addition, the SDC graph visualization is equipped with a number of features for adjusting the visual presentation. Zoom and rotate functionalities allow to position and size the graph. The graph display can also be adjusted by dragging and dropping nodes into other positions. Individual node types can be filtered to increase the understandability, e.g. filtering out the Web service clusters allows to see the goal graph more clearly. In the case of multiple sub-graphs

[4] The JPowerGraph is a Java library available under an LGPL open source license and can be downloaded from `http://jpowergraph.sourceforge.net`.

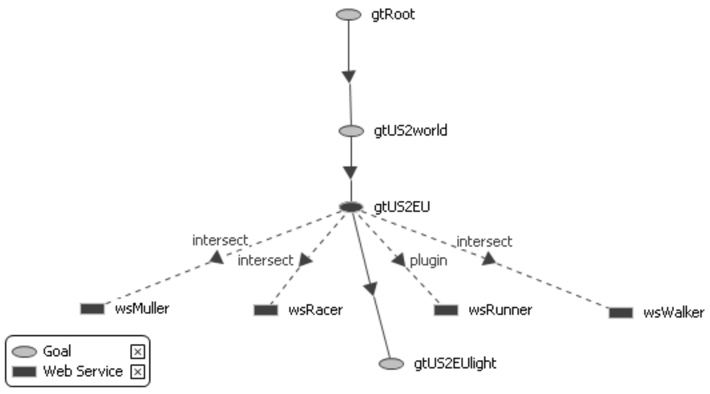


Fig. 4. SDC Graph Browsing for a Single Goal Template

– i.e. when several problem domains are captured in the SDC graph as explained above – the root nodes can be used to filter out entire subgraphs so that the user can focus on the relevant problem domain.

4.3 Integration with SWS Environments

The Web Service Modeling Toolkit WSMT is an IDE for the WSMO framework that aims at supporting developers and end-users through all activities in the software life-cycle of Semantic Web services. It currently encompasses text- and graph-based editing facilities for WSMO elements, a graphical tool for the creation of ontology mappings, and the graphical editing and browsing tool that we have extended for visualizing and browsing of SDC graphs.

Alongside the textual and graph-based editing support, the WSMT provides full syntactic and semantic validation of the entities created by the developer in order to ensure that modeling mistakes are caught early in the life cycle and do not propagate to later activities. Beyond the validation, the WSMT embeds reasoning support and discovery engines that allows the automatic and manual testing and verification of ontologies, goals, and Web service descriptions. Furthermore, the WSMT deployment functionalities allow to register WSMO elements within a Semantic Execution Environment (SEE) like WSMX [6] or the IRS system [2], which provide the facilities for the automated discovery, composition, and execution of Web services in order to solve a given goal.

Thus, the development of the SDC graph visualization as a new plug-in for the WSMT ensures a tight integration and reusability of already existing editing facilities as well as a seamless interaction with SWS environments for the automated solving of goals by the detection and execution of Web services. In fact, the SDC graph along with the presented visualization can be seen as a registry technique for goal-based SWS applications.

5 Related Work

We are not aware of any other existing approach or available tool that provides visualized browsing of Web services in a similar way. Goal-based approaches for Semantic Web services are rare in literature; however, some works that are not based on the WSMO framework also adopt the idea of managing Web services on the level of semantically described client requests, e.g. [11].

Most existing graphical user interfaces for the search and inspection of available Web services follow the registry approach already supported by UDDI, i.e. to organize Web services with respect to certain description elements that often are based on domain ontologies (e.g. [13,17,1]). Other tools provide Web service search engines in a google-like style, e.g. [10]. Our approach is based on a graph structure that is automatically created by semantic matchmaking of sufficiently rich semantic descriptions, which ensures a high precision and accuracy of the relevant relationships and does not require any additional descriptions apart from those required by the SWS techniques for automated goal solving.

Existing graphical user interfaces for managing WSMO elements provide form-based registry views that simply enlist the available goals and Web services (e.g. in the IRS system [2] or in WSMO Studio [4]). Our visualization tool overcomes the deficiencies in the comprehensibility and scalability by providing multi-leveled browsing facilities that allow clients to navigate from the complete overview of the search space down to detailed views on individual resources.

6 Conclusions and Future Work

This paper has presented a novel approach for the visualization and browsing of Web services that allows clients to comprehend and inspect available Web services on the level of problems that can be solved by them, abstracting from the technical details which are the focus of most existing tools. We use goals as the constituting element, and SDC graphs that define subsumption hierarchies of goal templates as the indexing structure and capture knowledge on the suitability of the available Web services from discovery results. The multi-leveled browsing facilities allow to navigate from the overall view of the search space down to more detailed views, and the realization as a plugin-in for the WSMT ensures a tight integration with SWS environments for automated goal solving.

For the future, we plan to extend the SDC graph visualization with graphical user interface and API for the creation and management of goal instances, and to further integrate the SDC technique within SWS environments.

Acknowledgements. This material is based upon works supported by EU funding under the projects SUPER (FP6-026850) and SEEMP (IST-4-027347-STP).

References

1. W. Abramowicz, K. Haniewicz, M. Kaczmarek, and D. Zyskowski. Architecture for Web services Filtering and Clustering. In *Proc. of the 2nd International Conference on Internet and Web Applications and Services (ICIW 2007), Mauritius*, 2006.

2. L. Cabral, J. Domingue, S. Galizia, A. Gugliotta, B. Norton, V. Tanasescu, and C. Pedrinaci. IRS-III – A Broker for Semantic Web Services based Applications. In *Proc. of the 5th International Semantic Web Conference (ISWC 2006), Athens(GA), USA*, 2006.
3. L. Clement, A. Hately, C. von Riegen, and T. Rogers (eds). UDDI Version 3.0.2. Approved Standard, OASIS, 2004. online: http://uddi.org/pubs/uddi_v3.htm.
4. M. Dimitrov, A. Simov, V. Momtchev, and M. Konstantinov. WSMO Studio - A Semantic Web Services Modelling Environment for WSMO (System Description). In *Proc. of the 4th European Semantic Web Conference (ESWC)*, Innsbruck, Austria, June 2007.
5. D. Fensel, H. Lausen, A. Polleres, J. de Bruijn, M. Stollberg, D. Roman, and J. Domigue. *Enabling Semantic Web Services. The Web Service Modeling Ontology.* Springer, Berlin, Heidelberg, 2006.
6. A. Haller, E. Cimpian, A. Mocan, E. Oren, and C. Bussler. WSMX - A Semantic Service-Oriented Architecture. In *Proceedings of the International Conference on Web Service (ICWS 2005), Orlando, Florida*, 2005.
7. U. Keller, R. Lara, H. Lausen, A. Polleres, and D. Fensel. Automatic Location of Services. In *Proceedings of the 2nd European Semantic Web Conference (ESWC 2005), Crete, Greece*, 2005.
8. M. Kerrigan. WSMOViz: An Ontology Visualization Approach for WSMO. In *Proc. of the 10th International Conference on Information Visualization (IV06)*, London, England, July 2006.
9. M. Kerrigan, A. Mocan, M. Tanler, and D. Fensel. The Web Service Modeling Toolkit - An Integrated Development Environment for Semantic Web Services (System Description). In *Proc. of the 4th European Semantic Web Conference (ESWC)*, Innsbruck, Austria, June 2007.
10. H. Lausen and T. Haselwanter. Finding Web Services. In *Proc. of the 1st European Semantic Technology Conference (ESTC), Vienna, Austria*, 2007.
11. E. M. Maximilien. Human-Based Semantic Web Services. In *Proc. of the 1st SWS Challenge Workshop, Stanford, California, USA*, March 2006.
12. C. Preist. A Conceptual Architecture for Semantic Web Services. In *Proc. of the 2nd International Semantic Web Conference (ISWC 2004)*, 2004.
13. N. Srinivasan, M. Paolucci, and K. Sycara. Adding OWL-S to UDDI – Implementation and Throughput. In *Proc. of the 1st International Workshop on Semantic Web Services and Web Process Composition at the ICWS, San Diego (USA)*, 2004.
14. M. Stollberg, M. Hepp, and J. Hoffmann. A Caching Mechanism for Semantic Web Service Discovery. In *Proc. of the 6th International Semantic Web Conference (ISWC 2007), Busan, South Korea*, 2007.
15. M. Stollberg, U. Keller, H. Lausen, and S. Heymans. Two-phase Web Service Discovery based on Rich Functional Descriptions. In *Proc. 4th European Semantic Web Conference (ESWC 2007), Innsbruck, Austria*, 2007.
16. M. Stollberg and B. Norton. A Refined Goal Model for Semantic Web Services. In *Proc. of the 2nd International Conference on Internet and Web Applications and Services (ICIW 2007), Mauritius*, 2007.
17. K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar, and J. Miller. METEOR-S WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services. *Journal of Information Technology and Management*, 6(1):17–39, 2005. Special Issue on Universal Global Integration.
18. T. Vitvar, M. Zaremba, and M. Moran. Dynamic Service Discovery through Meta-Interactions with Service Providers. In *Proc. of the 4th European Semantic Web Conference (ESWC 2007), Innsbruck, Austria*, 2007.