Semantic Web based Content Enrichment and Knowledge Reuse in e-Science

Feng Tao[1], Liming Chen[1], Nigel Shadbolt[1], Fenglian Xu[2], Simon Cox[2], Colin Puleston[3], Carole Goble[3]

[1] Department of Electronics and Computer Science, University of Southampton, Southampton, U.K.
{ft, lc, nrs}@ecs.soton.ac.uk
[2] e-Science Centre, School of Engineering Science, University of Southampton, Southampton, U.K.
{flx, s.j.cox}@soton.ac.uk
[3] Department of Computer Science, University of Manchester, Oxford Road, Manchester, U.K.
{carole, colin.puleston}@cs.man.ac.uk

Abstract *We address the life cycle of semantic web based knowledge management from ontology modelling to instance generation and reuse. We illustrate through a semantic web based knowledge management approach the potential of applying semantic web technologies in GEODISE, an e-Science pilot project in the domain of Engineering Design Search and Optimization (EDSO). In particular, we show how ontologies and semantically enriched instances are acquired through knowledge acquisition and resource annotation. This is illustrated not only in Protégé with an OWL plug-in, but also in a light weight function annotator customized for resource providers to semantically describe their own resources to be published. In terms of reuse, advice mechanisms, in particular a knowledge advisor based on semantic matching, are designed to consume the semantic information and facilitate service discovery, assembly and configuration in a real problem solving environment. An implementation has demonstrated the integration of the advisor in a text mode domain script editor and a GUI mode workflow composition environment. Our research work shows the potential of using semantic web technology to manage and reuse knowledge in e-Science.*

1. Introduction

The GEODISE (Grid Enabled Optimisation and Design Search for Engineering) project [3] aims to provide a Problem Solving Environment (PSE) that couples together

Grid middleware, engineering design packages, a database and a knowledge base to help engineers conduct large-scale distributed simulation of design search and optimisation in a virtual organization.

The Grid [2] has provided an operational infrastructure that enables distributed scientific computing and resource sharing in e-Science, yet it has become increasingly important that resources are consistently and semantically enriched to enable process automation and knowledge reuse within a distributed e-Science community. The Semantic Web technology promises to make Web content machine understandable, enabling software agents to process it and produce value-added knowledge to end users. The Semantic Grid[1] [10,] addresses this issue by applying Semantic Web technologies in Grid computing to enable easy-to-use and seamless automation towards the full richness of e-Science vision of future large-scale science over the Internet where the sharing and coordinated use of diverse resources in dynamic, distributed virtual organization is commonplace.

In order to achieve this vision, we proposed a Semantic Web based knowledge management approach in GEODISE. Knowledge acquisition is carried out through ontology modelling and semantic annotation. An ontology forms the conceptual structure of the knowledge base, and the semantic annotation populates the knowledge base with semantic instances. Knowledge reuse is then achieved through consuming these instances to generate knowledge driven decisions. In e-Science practice, it is common that the activities of generating and reusing the instances are conducted by different parties (e.g. human experts, beginner users, or computers), in different locations, time and environments. For example, in GEODISE, various Grid services and domain software components are used, such as the Java Cog [20] in Globus toolkit and the OPTIONS design exploration package [21] for EDSO. They are wrapped as Matlab functions which form our key resource in Grid enabled engineering problem solving. Semantic instances of these resources can be generated by knowledge engineers using knowledge acquisition tools such as Protégé, or by resource providers themselves using annotation tools such as the Function Annotator [14]. Semantics acquired in either way can be represented in the Web Ontology Language (OWL), which is a W3C standard that aims to help machines to understand data. Third-party programs can be used to process the instances in the knowledge base for different knowledge reuse purposes. This potentially allows for the knowledge to be used outside the awareness of its providers. In GEODISE, the purpose of knowledge support is to help engineers exploit reusable resources. We use the Jena semantic toolkit [13] to process the semantic information of these existing resources and formulate advice on activities during domain script editing and workflow assembly that require appropriate manipulation on these resources.

The rest of the paper is organized as follows. In the next section, we describe the knowledge management approach with respect to the life cycle of semantic web based knowledge management in GEODISE. In section 3, our experience of knowledge modeling is described in the context of GEODISE. This includes knowledge acquisi-

[1] http://www.semanticgrid.org

tion in regard to building ontologies and generating semantic annotations as instances in a knowledge base. We then describe in section 4 knowledge reuse issues, in particularly the workflow advisor that consumes the instances of semantic annotation in the knowledge base and provides value-add outputs to end users in suitable forms. In section 5, implementations are given to demonstrate the knowledge advisor and its integration with a domain script editor and the workflow composer in regards to the knowledge reuse. We finally give related work in section 6 and conclude in section 7.

2. Semantic web based knowledge management approach

A Semantic Web based knowledge management approach is proposed in order to semantically enrich the content of resources and extract actionable knowledge for reuse in an e-Science application. Figure 1, shows our approach, whereby we integrate various knowledge tools and e-Science applications covering the three key phases of the knowledge life cycle – knowledge acquisition, semantic storage and processing, and the (re)use of knowledge in semantic driven applications.

The knowledge acquisition aims to collect necessary information, build an ontology to represent the domain conceptualization and use the ontology to annotate Grid resources. The ontological information is collected by interviewing domain experts and studying domain manuals. Various tools, such as PC-PACK, Protégé [8] and OilEd [5] have been used to facilitate this building process. The ontological information extracted from the resources is used again to annotate these resources.

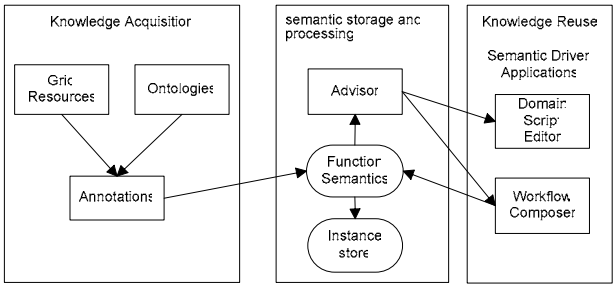


Figure 1 Semantic web based knowledge management approach in GEODISE

The result of the annotations is a set of semantically enriched content represented as instances that conform to the ontology used in the annotation process. These instances are stored in a flat file or database repository so that they can be accessed later. Sophisticated semantic matching and reasoning can be carried out on these instances to deduce knowledgeable decisions. The advisor is designed for this purpose. It retrieves relevant semantic information from the instance repository and processes it in order to provide context-sensitive advice according to the requests from the application side.

The last phase of the life cycle addresses knowledge reuse. In GEODISE, editing domain scripts and building workflows are two frequent tasks. A domain script editor has been developed to help editing domain scripts in a more efficient way. With the advisor integrated, it is capable of yielding contextual advice from processing semantic instances pre-acquired. The advisor has been also integrated into the workflow composition environment (WCE) for the same purpose.

3. Knowledge Modeling

GEODISE makes available a suite of grid-enabled functions [4] that allows design engineers to exploit grid resources when carrying out computational intensive EDSO processes in their favorite PSE (in our case: Matlab). The toolkit can be viewed as a powerful yet flexible script-based environment for grid computing. Components built on it can be used either separately or assembled together, invoked with certain configurations, conforming to best practice, to solve a particular engineering problem. Therefore we choose these grid-enabled Matlab functions and high level components (Figure 2) as the resources to be semantically enriched for knowledge reuse.

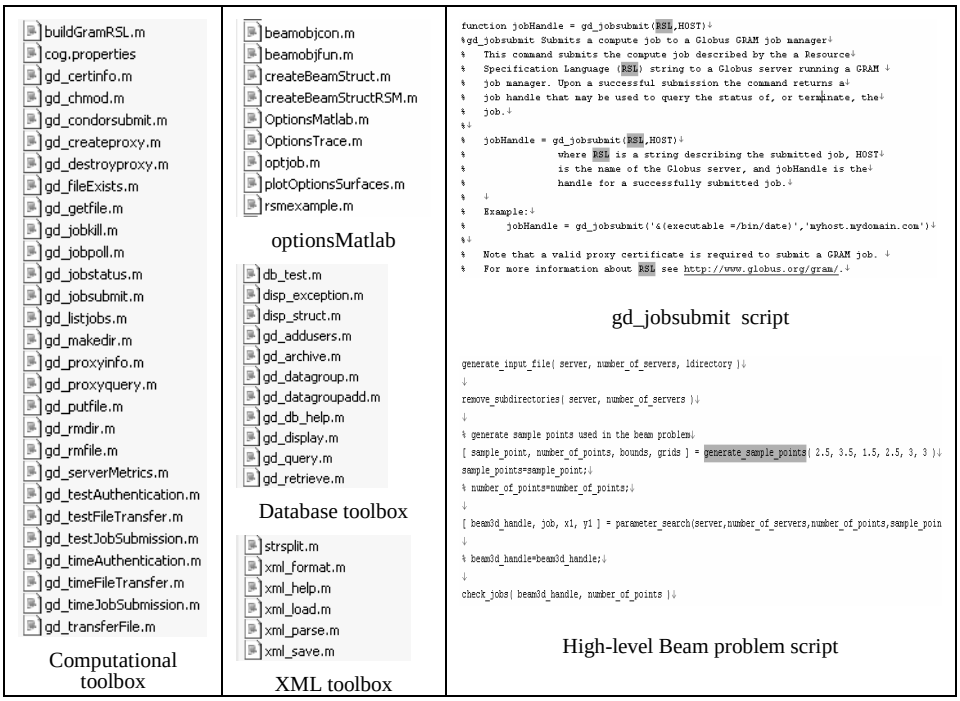


Figure 2 Grid-enabled Matlab functions and scripts

The task of knowledge modeling can be broken down into ontology modeling and instance generation.

3.1 Building ontologies

An ontology is a specification of conceptualization [6]. It explicitly defines the domain concepts and their relationships. It is similar to a dictionary or glossary, but with richer structure, relationship and axioms that describe a domain of interest more precisely. Many languages have been designed to express the ontology and semantic information. Among them, the most recent is the Web Ontology Language (OWL), which is built on top of RDF to provide more expressive power [24]. RDF is a graph model (or sets of triple statements) which is designed for describing and searching resources on the Web. DAML+OIL is a schema language that adds constraints on properties to assist machine reasoning. For example when "daml:TransitiveProperty" is added as a constraint on the property "P1:older_than" of a RDF model, if we have A1:P1:A2 and A2:P1:A3, then A1:P1:A3 can be inferred. This is useful for reasoning and inferring new knowledge that has not been directly stated. DAML+OIL also uses subProperty to describe relationship at different granularities.

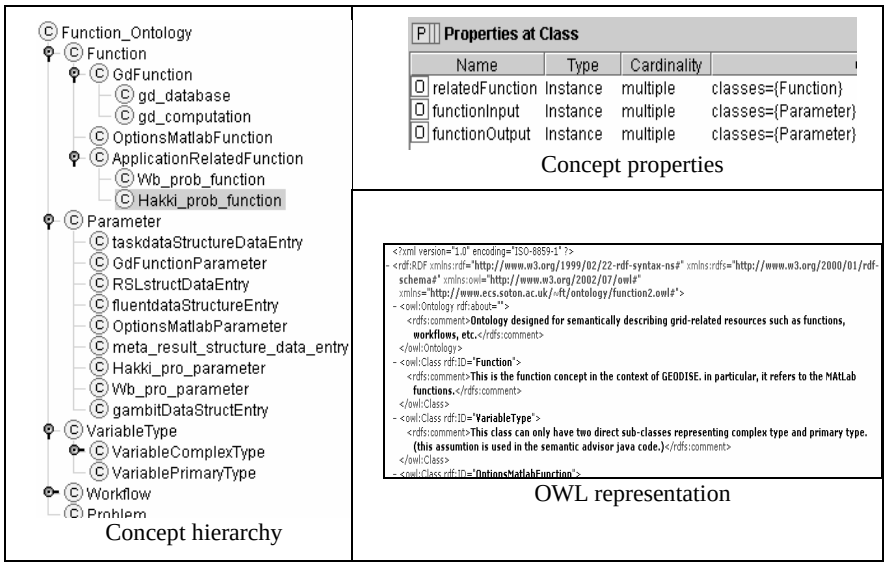


Figure 3 Building Ontologies

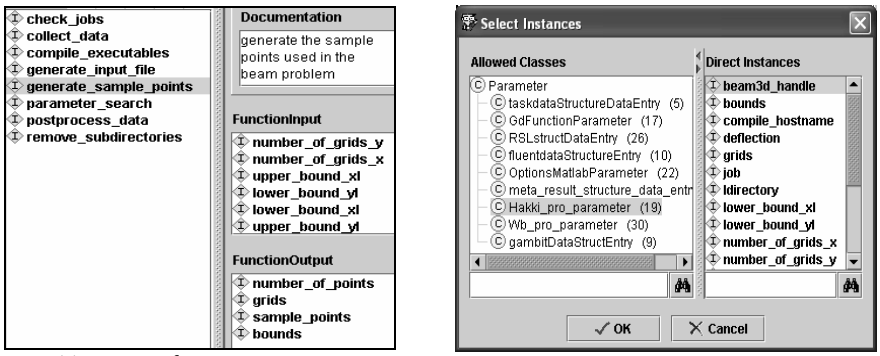
Figure 3, shows our function ontology developed using Protégé with an OWL plug-in. "Function", "Parameter", "VariableType", etc. are key concepts under which further taxonomy are made available to express hierarchical relationships (parent/children) among concepts. Each concept also has its properties defined to express the subject/predicate relationship (who uses who). The ontological information is saved in OWL format for content enrichment through instance generation.

3.2 Instance generation

Whilst an ontology is important in specifying the conceptual structure and a constrained vocabulary set, instances are treated as the concrete content in a semantic knowledge base. Generating the instances involves annotating the raw data source using pre-defined ontologies. In this paper, two methods are used to generate instances. Based on their operational mechanism, they are called "Ontology Instantiation" and "Resource Annotation" respectively.

1) Ontology instantiation

Protégé 2000 [8] is an ontology building and knowledge acquisition tool that has been frequently used for knowledge modelling purposes [15]. It allows knowledge engineers to focus on modelling without worrying about the underlying language and syntax. The modeling work can be saved in various formats including RDF and OWL.



(a) creating function instances (b) selecting parameter instances

Figure 4 Generating semantic instances in Protégé

As illustrated in Figure 4-a, to create function instances relevant information in the function source (Figure 2) is used to instantiate its corresponding ontology classes, such as "Function", "Parameter" and "VariableType", as defined in the function ontology in Figure 3. Each instance in the left column of Figure 4-a represents a function. Its properties ("FunctionInput", "FunctionOutput" as defined in the ontology) are also filled with object instances, the class of which is constrained by class properties defined in the ontology. The object instances can be created on the fly or selected from previously generated instances.

Instances generated in this way can be exported from Protégé (with the OWL plug-in) as is illustrated in Figure 5, where the instances are represented using RDF as well as OWL enhancements for extra semantics. The RDF can be also interpreted as *N*-Triples for efficient machine processing.

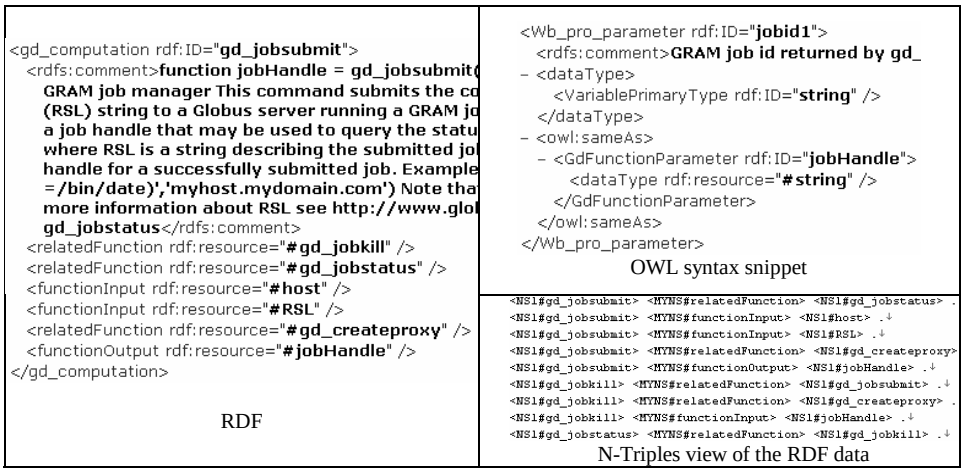


Figure 5 Function semantic instances

2) Resource annotation

While in Protégé, knowledge engineers acquire information about resources to instantiate an ontology, this is often too complicated for resource providers. In order to empower them to capture and publish function semantic instances as well, we have developed the Function Annotator as illustrated in Figure 6, a lightweight knowledge acquisition tool. OWL is used by the Function Annotator to represent the ontologies and for storing the semantic instances in the knowledge repository.

Once function sources are loaded into the source panel (right bottom), they are parsed for potential semantic information listed in the function browser (right top). According to the content to be annotated, users can establish an annotation panel (middle) automatically generated from a particular selected ontology (left). The annotation is carried out by dragging relevant information from the function browser, dropping it into the annotation panel and filling out relevant fields.

Figure 6 Function Annotator

The generated function semantic annotations contain the same information as the function semantic instances. Details can be found in [14].

4 Knowledge Reuse

Once semantic instances are made available, it is possible to access and process these instances for the purpose of knowledge reuse. Since instances are represented in standard OWL language, any OWL compliant API can be used, for example, the Wonder Web OWL API [7] and the Jena ontology API [13]. We use Jena in this work.

4.1 Reusing semantic instances to advise engineers

Functions can only be assembled together if their interfaces semantically match each other to some extent, i.e. a function's input semantically consumes the output of another function. Workflow builders, especially beginners, often are not clear about the semantic interfaces of the functions. However, suggestions can be deduced through semantic interface matching. This is especially useful when the function repository is

dynamically updated or the number of functions is large, which is the case in our engineering e-Science community.

Each function can be viewed as a domain specific service which must be configured correctly and composed with other services to form a problem solving workflow. The granularity of the services varies from low level atomic functions (usually generic) to high level workflow building blocks (often more problem specific) that are made up of low level functions.

There are two types of advice:

1. *Function configuration advice* - this provides automatically generated advice on function configuration. We call this "horizontal advice" as it is triggered during function configuration, i.e., horizontal scripting.

   Semantic decomposing is used when a function parameter is a complex type, e.g., a structure that contains a list of fields which are either primary types or complex types. In this case, the semantic interface can be expanded by decomposing this parameter and its subfields until there are no more complex types. This often yields richer semantic interfaces that contain more concepts and relationships for semantic matching.

2. *Function assembly advice* – functions that can be assembled together according to semantic compatibility of their interfaces. This is named as "vertical advice" which is triggered during vertical assembly of configured function instances.

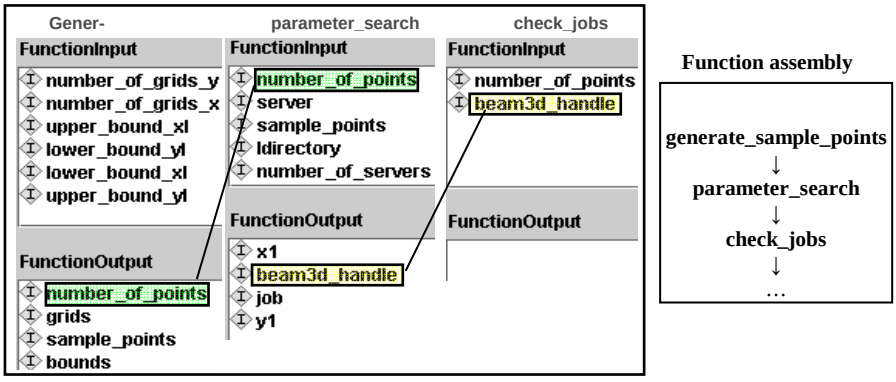


Figure 7 Semantic matching for function assembly

The function assembly advice is base on matching functions, there are two types of elements in the function interface that can be used for matching:

i. Primary data type: two functions can be assembled together only if the second function gets its input interface satisfied. Primary data types such as "string" or "integer" used in function interfaces can be used to consider function compati-

bility when suggesting the next function to use after a currently deployed function.

ii. Semantic data type: this refers to the "ArgumentType" instances (beam3d_handle, number_of_points, etc.) used as function semantic interface. They are used in semantic matching functions for advice on workflow assembly. This is demonstrated in Figure 7 where semantic interfaces of three functions have been listed and the matches (represented as links) implicates a valid function assembly as shown in the right.

Although this is useful in suggesting compatible functions in terms of workflow assembly, there are often occasions where very few or no match exists because the semantic interface of the target function is too restricted. To solve this problem, OWL expressions such as "SameAs" (in Figure 5) are used to map equivalent concepts and therefore relax the semantic matching.

5. Implementations and applications

5.1 Knowledge advisor

The advisor module is based on an API capable of retrieving and post-processing semantic instances expressed in OWL. The process operations include ontology interpretation, semantic matching and reasoning/inference. The advisor is implemented using Jena OWL ontology API [19].

A tutorial Java class demonstrates how the API is used to provide semantic support and advice. Figure 8 shows usage cases related to semantic consumption and advice based on it.

1	List all classes – (all classes defined in the ontology)
2	List subclass of a given class (as defined in the ontology)
3	List all individuals of a class (instances under of particular class, either direct or indirect)
4	List properties of a given individual (declared properties of a particular instance)
5	Expose semantic interface of a given individual function (an example of case 4 on function)
6	**Suggest contextual functions in a workflow**
7	**Expose in/output parameter individual of a given individual function**
8	**Decompose a particular parameter individual**
9	Documentation (provide human readable comment on any semantic resources)
10	Individual exists? (Check instance existence)

Figure 8 Advisor functions on processing semantic instances

We can also use the tutorial class to demonstrate key functionalities of using the semantic advisor API. In Figure 8, numbers 1 to 4, 9 and 10 are generic usage of ontology interpretation and semantic consumption. The rest of the cases are domain specific cases that use the generic API and provide further functionality such as exposing the semantic interface of a particular function individual, advising function candidates for workflow assembly, etc. Some example output of the tutorial class can be seen in Figure 9.

```
    Expose semantic interface
    generate_sample_points

    Semantic Interface is: [http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#number_of_points,
http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#grids,
http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#number_of_grids_y,
http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#lower_bound_yl,

    Decompose a particular parameter individual
    optionsMatlabInputStru

    RDF type is: http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#OptionsMatlabParameter
    Direct decomposed parameter individuals are: [
    org.geodise.knowledge.semanticweb.ParameterIndividual
<http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#OLEVEL> , integer,
    org. geodise.knowledge.semanticweb.ParameterIndividual
<http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#VNAM> , Vector,

    Advice on contextual component (workflow assembling advice based on semantic interface matching)
    parameter_search

    its pre-contextual functions are: [
    org. geodise.knowledge.semanticweb.FunctionIndividual
<http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#generate_sample_points>]
    its post-contextual functions are: [
    org. geodise.knowledge.semanticweb.FunctionIndividual
<http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#postprocess_data>,
    org. geodise.knowledge.semanticweb.FunctionIndividual
<http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#check_jobs>,
    org. geodise.knowledge.semanticweb.FunctionIndividual
<http://www.ecs.soton.ac.uk/~ft/ontology/function2.owl#collect_data>]
```

Figure 9 Example output of the tutorial class

5.2 Using the knowledge advisor

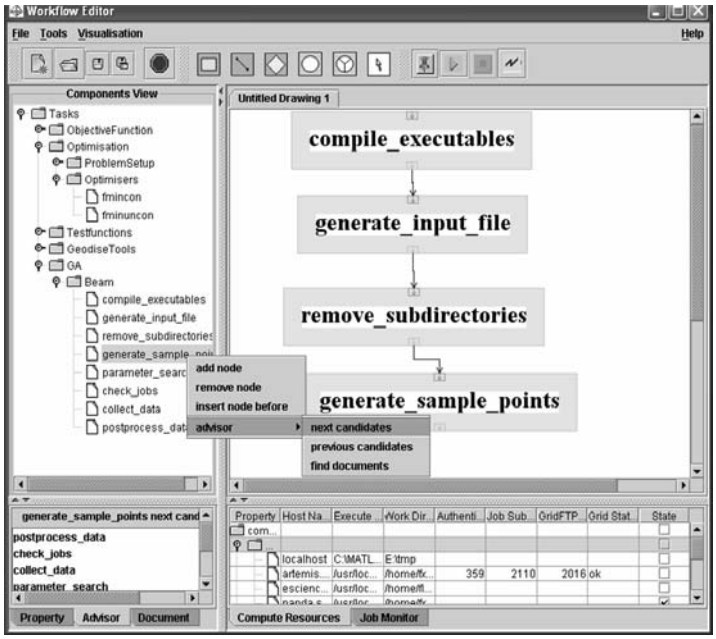
There are two applications in which the advisor can be integrated. In both case, semantic based knowledge can be reused in GEODISE.

a) Workflow Composition Environment (WCE)

The workflow composer in GEODISE is a GUI based application which allows engineers to visually select tasks from a function hierarchy, configure and assemble them into a workflow for e-science problem solving.

The purpose of integrating the semantic based advisor in the GUI based WCE is to make use of the rich semantic content and help the users choose suitable functions and make appropriate configuration during workflow assembly.

As illustrated in Figure 10, each function (in the left hand side panel) that has been previously semantically enriched, the workflow advisor can be called to deduce its contextual functions (as listed in the left bottom panel in Figure 10) that can be deployed before/after. This is achieved by semantically processing the semantic instances as described in section 4.1. In this way, the users can focus on compatible functions can be of use to further assemble the workflow without tediously investigating the semantic interface of all irrelevant functions. It then generates a Matlab script and submits it to a Matlab server for execution. It also takes care of the workflow management, monitoring and execution, but this is outside the scope of the current paper: interested readers can refer to [12] for further information.




```
……

% Compile and transfer the beam3d executable to the client

compile_executables( 'blue02.iridis.soton.ac.uk', server, number_of_servers, ldirectory )

% Generate the input file, and transfer it to the Globus servers

generate_input_file( server, number_of_servers, ldirectory )

% Clean-up. Remove all subdirectories starting with "job"

remove_subdirectories( server, number_of_servers )

% Generate sample points between lower and upper limits

 [ sample_point, number_of_points, bounds, grids ] = generate_sample_points( 2.5, 3.5, 1.5, 2.5, 3, 3 )

……
```

Figure 10 Advisor integrated in the WCE and the generated scripts

b) Domain Script Editor (DSE)

Quite often, engineers need to edit domain related scripts in addition to GUI based design tools, such as the WCE. But manipulating plain texts is painful and tedious. In GEODISE, Matlab is the script language that glues EDSO and grid computing re-

sources together. This motivated the design of a domain script editor with the advisor integrated.

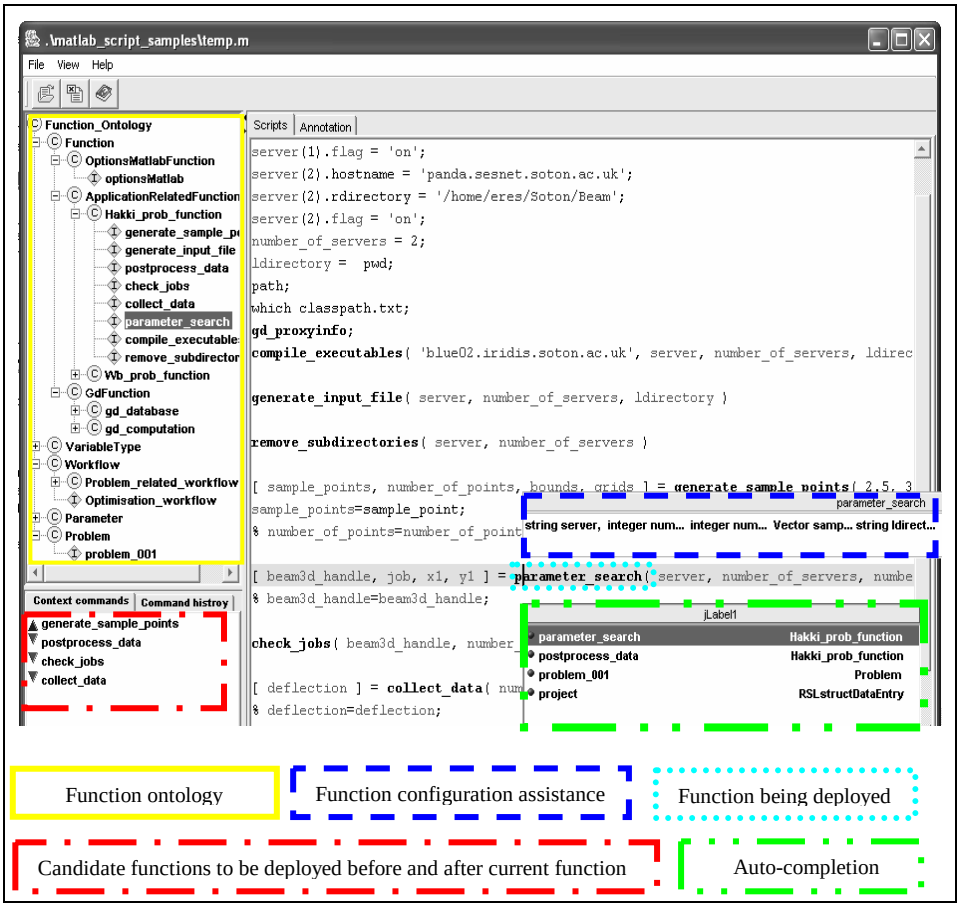


Figure 11 Domain script editor integrated with the advisor

Key features include:

- Component based - It can be delivered as a Java swing GUI component that can be used in any Java application (e.g., in the GUI based workflow composer as an alternative view of the workflow).
- Generic – The DSE is Ontology/Semantic powered meaning that it can be used to advise on different domain scripts when loaded with corresponding semantic annotations. E.g., Gambit scripts, gd_xxx functions including GEODISE computation toolbox and database toolbox, problem specific function scripts in Matlab, etc.

- De-centralized - Semantic instances are collected (in Protégé with OWL plug-in and in the function annotator) separately from their use, i.e., advisor integrated in domain applications.
- Horizontal advice on component configuration – exposing semantic interfaces, tool-tipping semantic annotations, auto-completions, etc, as shown in popping up windows in Figure 11.
- Vertical advice on component assembly – semantic interface matching and reasoning for contextual component recommendation as shown in the left bottom panel in Figure 11, where the blue arrow represents for a pre-contextual candidate and the red one for a consequence candidate.

6. Related work

There are many projects that address the life cycle of knowledge management. Amongst them the Advanced Knowledge Technologies project (AKT) tackles the problems which arise during from knowledge acquisition, through modelling to publication and reuse. In particular, the AKT triple store [17] focuses on knowledge retrieval of RDF triples: the example cited in [17] is populated over an OWL ontology of UK computer science research expertise. Our approach is similar to this in that we construct an EDSO and function ontology based on which semantic annotations of GEODISE functions and related resources are generated and stored in a semantic repository. Instances in the AKT triple store are reused for query and semantic web browsing while the semantic annotated functions in GEODISE are reused for service discovery (function query) and workflow assembly through semantic matching.

The Ontobroker project uses ontologies to annotate and wrap Web documents and provides an ontology-based answering service to enhance the accessibility of their web documents [16]. COHSE Mozilla Annotator [25] and OntoMat-Annotizer [26] are two of the annotators to enrich web page with ontological information.

Pre-defined rules in a JESS rule base were used in [9] to advice on workflow assembly, but this is limited with regard to scalability and has high overhead cost when the rules increase. It is also difficult to elicit rules consistently.

Efforts have been made to locate services by semantically matching the requirements to the service descriptions. In [23], a semantic matching approach is proposed to match between service requests and advertisements described using DAML-S. It aims to extend the representation capabilities of registries such as UDDI and languages such as WDSL so that semantically enriched web services can be discovered through semantic marching. Here we adopt a similar approach but aim to provide advice on service assembly, in particular what can be deployed as a pre/post contextual task. The difference is that as long as there is service already deployed, the user does not need to describe their service request, the semantic matching can be carried out to find compatible services to the deployed one. The users only need to browse the returned services that are semantically compatible and select one of them for service assembly.

7. Summary and conclusion

We describe the life cycle of semantic web based knowledge management from ontology modelling, instance generation to reuse. Resources in the GEODISE project such as grid-enabled functions and workflow building components have been targeted for ontological modelling and semantic instance generation using Protégé with OWL plug-in and our own Function Annotator. We show that semantic instances generated can be consumed to deduce advice. In particular, we use semantic decomposition and semantic matching mechanisms to generate advice on function configuration and assembly. These have been demonstrated through the knowledge advisor suggesting semantically compatible function candidates and their possible configuration. We have also integrated the advisor into the domain text editing and workflow composition developed for the GEODISE project. The examples we have used demonstrate that the approach proposed is feasible and helpful. We intend to support further aspects of the knowledge life-cycle in further work and improve integration of knowledge technologies into users' Problem Solving Environments.

References

1. Tao, F, Cox, S.J, Chen, L, Shadbolt, N.R, Xu, F, Puleston, C, Goble, C, and Song,W. "Towards the Semantic Grid: Enriching Content for Management and Reuse", Proceedings of UK e-Science All Hands Conference 2003, pp. 695-702
2. Ian Foster , Carl Kesselman, "The Grid: Blueprint for a New Computing Infrastructure", 2nd Edition, Morgan Kaufmann, 2004. ISBN: 1-55860-933-4
3. GEODISE project, http://www.GEODISE.org
4. Eres, M.H, Pound, G.E, Jiao, Z, Wason, J, Xu, F, Keane, A.J, and Cox, S.J, "Implementation of a Grid-enabled Problem Solving Environment in Matlab", Proceedings of the International Conference on Computer Science (ICCS 2003), Part IV, Lecture Notes in Computer Science, pp. 420-429
5. Sean Bechhofer, Ian Horrocks, Carole Goble, Robert Stevens. "OilEd: a Reason-able Ontology Editor for the Semantic Web", Proceedings of KI2001, Joint German/Austrian conference on Artificial Intelligence, September 19-21, Vienna. Springer-Verlag LNAI Vol. 2174, pp 396--408. 2001
6. T. R. Gruber. "A translation approach to portable ontologies", Knowledge Acquisition, 5(2):199-220, 1993.
   http://ksl-web.stanford.edu/KSL_Abstracts/KSL-92-71.html
7. WonderWeb API, http://owl.man.ac.uk/api.shtml
8. The Protégé homepage, http://protege.stanford.edu/index.html
9. Tao, F, Chen, L, Shadbolt, N,R. Pound, G, Cox, S.J., "Towards The Semantic Grid: Putting Knowledge To Work In Design Optimisation", Proceedings of I-KNOW '03, 3rd International Conference of Knowledge Management, p.p. 555-566.
10. Carole Goble and David De Roure The Semantic Grid: Building Bridges and Busting-Myths, 16th European conference on Artificial Intelligence ECAI 2004, Valencia, Spain, 23-27 July 2004.
11. De Roure, D. Hendler, J.A., "E-science: the grid and the semantic web", Intelligent Systems, IEEE, Vol. 19, Issue 1, 2004, p.p. 65-71.
    http://ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=1265888

12. Xu, F and Cox, S.J. “Workflow Tool for Engineers in a Grid-Enabled Matlab Environment”, Proceedings of UK e-Science All Hands Meeting 2003, pp. 212-215
13. HP Labs Semantic Web Research, http://www.hpl.hp.com/semweb/
14. Chen L., Shadbolt N.R., Tao F., Puleston C., Goble C., Cox S.J. "Empower Resource Providers to Build the Semantic Grid", submitted to the IEEE/WIC/ACM International Conference on Web Intelligence 2004 (WI'04)
15. Holger Knublauch , An AI tool for the real world: Knowledge modeling with Protégé , http://www.javaworld.com/javaworld/jw-06-2003/jw-0620-protege.html, 2003
16. OntoBroker project. http://ontobroker.aifb.uni-karlsruhe.de/index_ob.html
17. AKT triple store, http://triplestore.aktors.org/
18. Qian, Cheng Yuan, Charles, An Integrated Process of CFD Analysis and Design Optimization with Underhood Thermal Application, SAE Paper 2001-01-0637 SAE 2001 World Congress, Detroit, MI, Mar 5-8, 2001..
19. Jena, a semantic web framework, http://www.hpl.hp.com/semweb/jena.htm
20. Gregor von Laszewski, Ian Foster, Jarek Gawor, and Peter Lane, "A Java Commodity Grid Kit," Concurrency and Computation: Practice and Experience, vol. 13, no. 8-9, pp. 643-662, 2001, http:/www.cogkits.org/
21. Keane, AJ. OPTIONS Design Exploration System http://www.soton.ac.uk/~ajk/options/welcome.html
22. M. H. Eres, G. E. Pound, Z. Jiao, J. L. Wason, F. Xu, A. J. Keane, and S. J. Cox. (2003) Implementation and utilisation of a Grid-enabled Problem Solving Environment in Matlab, Journal of Future Generation Computer Systems, in press.
23. Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, Katia Sycara; "Semantic Matching of Web Services Capabilities." In Proceedings of the 1st International Semantic Web Conference (ISWC2002)
24. Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7-26, 2003.
25. Cohse Mozilla Annotator , http://cohse.semanticweb.org/mozilla/annotation/
26. OntoMat-Annotizer , http://annotation.semanticweb.org/ontomat/index.html