Semantic Description, Publication and Discovery of Workflows in myGrid

Simon Miles[1], Juri Papay[1], Chris Wroe[2], Phillip Lord[2], Carole Goble[2], Luc Moreau[1]

[1]School of Electronics and Computer Science, University of Southampton, UK
[2]Department of Computer Science, University of Manchester, UK
Contact: sm@ecs.soton.ac.uk
Technical Report Number: ECSTR-IAM04-001
ISBN: 0 854 32 804

Abstract

The bioinformatics scientific process relies on in silico experiments, which are experiments executed in full in a computational environment. Scientists wish to encode the designs of these experiments as workflows because they provide minimal, declarative descriptions of the designs, overcoming many barriers to the sharing and re-use of these designs between scientists and enable the use of the most appropriate services available at any one time. We anticipate that the number of workflows will increase quickly as more scientists begin to make use of existing workflow construction tools to express their experiment designs. Discovery then becomes an increasingly hard problem, as it becomes more difficult for a scientist to identify the workflows relevant to their particular research goals amongst all those on offer. While many approaches exist for the publishing and discovery of services, there have been few attempts to address where and how authors of experimental designs should advertise the availability of their work or how relevant workflows can be discovered with minimal effort from the user. As the users designing and adapting experiments will not necessarily have a computer science background, we also have to consider how publishing and discovery can be achieved in such a way that they are not required to have detailed technical knowledge of workflow scripting languages. Furthermore, we believe they should be able to make use of others' expert knowledge (the semantics) of the given scientific domain. In this paper, we define the issues related to the semantic description, publishing and discovery of workflows, and demonstrate how the architecture created by the myGrid project aids scientists in this process. We give a walk-through of how users can construct, publish, annotate, discover and enact workflows via the user interfaces of the myGrid architecture; we then describe novel middleware protocols, making use of the Semantic Web technologies RDF and OWL to support workflow publishing and discovery.

Keywords: *workflow, semantic description, e-Science, bioinformatics, service discovery, Grid, personalisation*

1 Introduction

Traditionally, the biological scientific process has involved experiments on living systems, in vivo, or on parts of a living system in a test tube, in vitro. Bioinformatics has focused on supporting the experimental biologist by enabling many more experiments to be carried out in silico, that is computationally. As this better supports automation and also harnesses the collective knowledge of the

discipline, in silico biological experiments have greatly enabled the process of validating hypotheses, and gathering additional information to shape the design of future experiments. If experiments can be easily shared, adapted and reused, hopefully science will become more efficient; distributed architectures on the Internet promise to be the most effective mechanism to achieve this goal for in silico experiments.

Both the Web Services and Grid architectures [32, 25] have adopted a service-oriented approach, in which computational resources, storage resources, programs and databases can be represented by services. In such a context, a service is a network-enabled entity capable of encapsulating diverse implementations behind a common interface. The benefit of such a uniform view is that it facilitates the composition of services into more sophisticated services, hereby promoting sharing and reuse of resources in distributed environments. To this end, a number of workflow languages have emerged which are capable of describing complex compositions of services, e.g. WSFL [33], XLANG [39], BPEL [8], XScufl [29].

However, service-oriented architectures currently provide no mechanism to facilitate the sharing of workflows. At present, workflow authors simply make a list of the available workflows available via a Web page. With an increasing number of workflows, and sites listing them, searching for them in this way will soon become untenable.

DAML-S [5] considers workflows as largely equivalent to services with regards to publishing and discovery because they are functional entities that are identified by their interface (inputs and outputs) and overall function. Therefore, as with services, workflows need to be published in order to be discovered and reused. However, publishing a workflow involves two distinct steps: first, the workflow script must be archived in a repository from which it can be publicly retrieved; next, a description of and a reference (e.g. a URI) to its script need to be advertised in a registry. In this context, a registry is defined as a service holding descriptions of workflows and services. Many protocols for publishing service descriptions, including de-facto standards, such as UDDI [31], Jini [18], and BioMoby [34] do not, in themselves, address publication and discovery of workflows. DAML-S, on the other hand, is an ontology capable of describing complex processes, but is not a registry system for publishing and discovery.

Once a published workflow has been discovered, scientists use their expert knowledge of the scientific field to judge whether a design is applicable to their own work. Unfortunately, such domain-specific knowledge is not readily available from workflow scripts, which are engineered in terms of programmatic notions such as interfaces, ports, operations and messages of the service-oriented architecture in use [37]; furthermore, domain knowledge cannot be inferred from these low-level notions. However, semantic descriptions can be added to workflows, in order to make high-level knowledge explicit; these must be machine interpretable if tools are to be capable of recommending the applicability of workflows based on the domain-specific knowledge of a scientist.

myGrid [23] is a pilot project funded by the UK e-Science programme to develop Grid middleware in the biological sciences context. The goal of the myGrid project [23] is to develop a software infrastructure that provides support for bioinformaticians in the design and execution of workflow-based in silico experiments using the resources of the Grid [12]. In silico experiments can operate over the Grid, in which resources are geographically distributed and managed by multiple institutions, and the necessary tools, services and workflows are discovered and invoked dynamically. It is a data-intensive Grid, where the complexity is in the data itself, the number of repositories and tools that need to be involved in the computations, and also in the heterogeneity of the data, operations and tools. The myGrid architecture includes components for composing workflows, annotating them with semantic descriptions, publishing semantically described workflows, reasoning over semantic descriptions, discovering workflows from semantic queries and executing them. In previous papers,

we have discussed various facets of our approach to service publication and discovery, namely its preliminary design [19], its protocol for annotating service descriptions [21] and its performance [22, 20]. The purpose of this paper is to discuss the final design of our architecture for workflow publication and discovery, and its implementation and integration in an electronic lab-book, for manipulation by the scientist. Specifically, this paper focuses on the following technical contributions of the myGrid architecture for publishing and discovering workflow-based in silico experiments.

- A definition of the protocol used to publish, annotate and discover workflows in a registry. The protocol is independent of the actual language used to encode workflows. To this end, it relies on a notion of workflow executive summary, which identifies, in an extensible manner, the salient features of a workflow that can be described conceptually or syntactically.
- The use of RDF (the W3C Resource Description Framework) [27], which underpins the Semantic Web effort [7], as the underlying representation to express service and workflow descriptions and to facilitate the attachment of metadata to them. Besides being a flexible and powerful representation formalism, RDF provides for uniform graph-based querying using RDQL [28], which is used in our registry to support workflow discovery.
- The use of OWL ontologies to encode domain-specific knowledge and to allow the inferences required by the discovery process. Specifically, ontologies are used to index workflows according to their functionality and the semantic types of their inputs and outputs, expressed as biological concepts. A semantic find component, which uses a description logic reasoner, provides complete reasoning over the rich OWL-based descriptions of workflows, and facilitates discovery with complex queries over these descriptions.
- A complete implementation of the architecture, organised as a set of Web Services and associated user interfaces are all available for download from http://www.myGrid.org.uk/myGrid/web/download/

Section 2 presents an illustrative bioinformatics case study, including a representative workflow, to aid in describing and demonstrating the usefulness of our work. Section 3 shows the users' perspective in sharing workflows from composition through publishing and description to discovery. In Section 4, we examine the use of semantic technologies to represent the knowledge used for discovery, and in Section 5 we define the protocols used in myGrid to process this information. The implementation of the middleware using this protocol is given in Section 6, the scope of our work and related work is discussed in Section 7 and we draw conclusions and suggest further work in Section 8.

2 Workflows in Graves' disease experiments

We now present a case study to illustrate our approach to semantic description and discovery of workflows. The Graves' Disease application, an exemplar application for myGrid, is intended to help the investigation of a thyroid disorder [24]. Specifically, the purpose of the application is to help biologists identify gene mutations that may be involved in causing the condition.

The Graves' Disease scenario uses a well known and common "candidate gene" approach. We assume that previous biological investigations have been used to isolate a region on the genome in which genes affecting Graves' Disease may lie. By looking through this region for variations between Graves' Disease and normal patients, then determining whether these variations lie within a gene, a number of candidate genes can be found. One of the most common variations is called a Single Nucleotide

Polymorphism (SNP), which is a variation involving only a single nucleotide, rather than a large scale change affecting many nucleotides. But often many of these polymorphisms occur in a region, most of which will be not related to Graves' Disease. The in silico process consists of gathering information from several publicly available data resources, many of which have been made available as services at one or more locations, describing the current state of knowledge about the genes in question. Once such information has been obtained for a set of candidate genes, the scientist can design an in vitro experiment that will test their likelihood of being involved in the disease.

To enable re-running of the experiment and best use of Grid resources, the experiment is encoded as a workflow, a composed set of services or other workflows, which we refer to as CandidateGeneAnalysis. This workflow takes a "probe set ID" referring to a gene sequence in the Affymetrix database [2] as input and ultimately returns a record from the European Molecular Biology Laboratory database containing information about the sequence including SNPs. The workflow's structure can be seen on the left hand side of Figure 2. The specific details can be found in its script, encoded in the Scufl workflow language. The Scufl workflow language, developed as part of myGrid, simplifies the process of creating workflows for biology by making the language granularity and concepts as intuitive as possible for potential users [29].

Since this experiment is more widely applicable than just for the study of Graves' Disease, the biologists may wish to share it with others, and would want to do so in such a way that it can usefully be discovered and re-used. User requirements [10] have identified some questions that scientists commonly ask about such kinds of experiment. Specifically, since they aim to discover SNPs from gene sequence data, they will seek experiments that: 1) process a given sort of data (e.g. genes), 2) retrieve information from a public database about a specific gene 3) provide a given type of output (e.g. SNPs).

Since experiments are represented as workflows, and workflows are characterised by the kind of their inputs, outputs and their function, a user will specifically seek published workflows that: 1) have a given semantic type (e.g. sequence data) as one of their inputs, 2) perform a given type of function (e.g. retrieve a database record about a gene), 3) have a specific semantic type (e.g. SNPs) as one of their outputs, 4) use certain services (e.g. named public genetic information databases). This use case indicates that we need a large number of entities in order to perform in silico experiments; Figure 1 summarises the terminology we adopt in this paper. Next, we examine how users go about publishing workflows, so that the questions above can be answered by the architecture to support the discovery process.

3 The Users' Perspective

The purpose of this section is twofold. On the one hand, we illustrate our approach to workflow publication and discovery, using snapshots of the graphical user interface that the scientist is presented with when using the myGrid system; the functionality of this interface was derived from the user requirements we captured at the beginning of the myGrid project [10]. On the other hand, we identify key technical requirements for the knowledge representation that is required to support our approach. With the user-centric perspective adopted by myGrid, we analyse the kinds of discovery that scientists are confronted with: when composing workflows and when deciding which scientific experiments to run. In order to be discovered, workflows need to have been published, and we examine how suitable semantic descriptions of these workflows can be made available to the system.

	Term	Description	Example	
Basic Concepts	Workflow Language	A language for specifying a *workflow*.	BPEL, Scufl	
	Service	An atomic entity that can be invoked.	Blast service at EBI	Concrete
	Workflow	A composed set of *services* or other *workflows* and a specification of the data flow between them	Candidate Gene Analysis	
	Activity	A *workflow* or a service		
	Service Type	Abstract activity definition that represents the *class of a service* or a *workflow template*	Sequence Alignment, BLAST	Abstract
Workflow language independent	Workflow Executive Summary	The salient features of a workflow that are desired to be described conceptually or syntactically: *inputs, outputs, task(s), component resources*	*Input*: probe set ID; *Output*: EMBL_SNP; *Using*: Affymetrix database, EMBL database, BLAST; *Task*: SNP_annotation	
Workflow language dependent	Workflow Template	A *workflow* in which one or more of the activities are not directly invokable, but represented as a specification which can be resolved into invokable *activity*.	The Candidate Gene Analysis data and control flow, choreographing service types (e.g. BLAST) instead of, or as well as, activities (e.g. BLAST at EBI).	
	Workflow Script	A specification, defined in terms of the *workflow language*, that we can directly enact.	http://www.ecs.soton.ac.uk∼sm/myGrid/AffyIdToEmblSnps.scufl	Concrete

FIGURE 1: myGrid terminology.

3.1 Construction-time Discovery

Designing a workflow means linking together functional entities such as Web Services or other workflows, which we refer to as activities (see Figure 1), so that the outputs of some are used as the inputs of others. Workflows are constructed by linking together sub-activities that pass data between each other. Figure 2 shows the myGrid graphical workflow construction tool Taverna [30].

Workflows need not be created from scratch: they can be adapted and personalised from previously written workflows. As part of personalisation, the workflow's author needs to discover existing activities (workflows and services) so as to include them in their design. Hence, since both services and workflows need to be discovered, both are listed in the 'Available services panel' of Figure 2. Crucially, user requirements have identified that: 1) biologists require activities to be discoverable by the function they perform, that is task orientated discovery and 2) final selection ultimately rests with the scientist, who will select those to be included according to the goal of the experiment they are designing. To this end, scientists need to be able to draw on a wide range of information about activities in order to inform their decisions. Specifically, the following workflow descriptions[1] are used: the

[1]In this paper, we focus on workflow descriptions, but we note that service descriptions are similar. Service descriptions

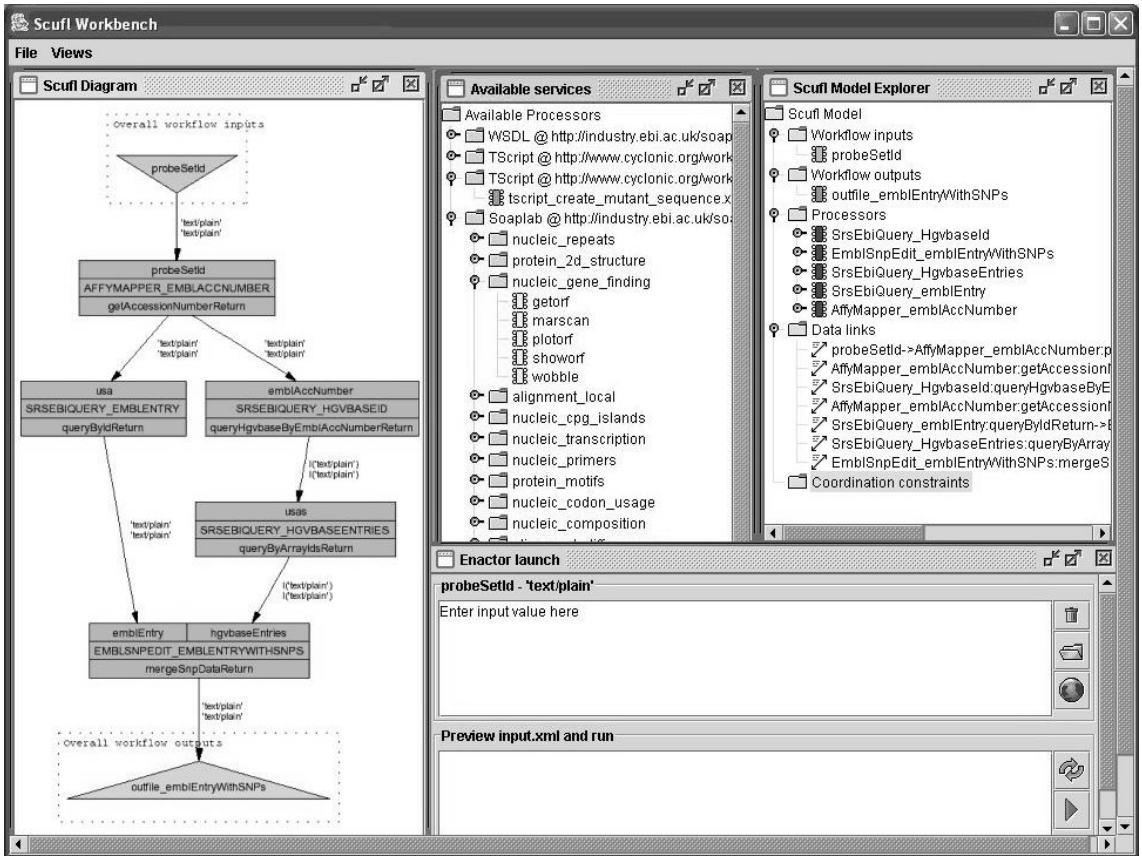

FIGURE 2: Workflow construction using Taverna. The left-hand panel contains a depiction of the workflow itself with each box representing an activity in the workflow; when the workflow is enacted, this activity results in a Web Service operation call or the invocation of another workflow. Data flows from the inputs, represented by inverted triangles, through the linked services to the output triangle at the bottom of the workflow. The 'Scufl Model Explorer' panel shows a hierarchical view of the workflow and 'Enactor launch' relates to test runs of the workflow.

workflow's author and their institution, the function of the workflow, the sub-activities it may invoke (and their function), and the inputs and outputs of the workflow expressed in biological terms. When scientists consider a workflow for insertion in an experiment, they regard it as a gray-box, because they want to know about the activities it is composed of, though the fine details of their dependencies, control and data flows do not matter at this stage.

Selecting activities based on the functions they perform helps guarantee that the overall experiment has the intended behaviour. However, further care is needed to ensure that the composition is operationally consistent at the transport level: data types and formats of outputs must be compatible with the inputs they feed into. In order to verify such constraints, service interfaces [37] and an equivalent concept for workflows need to be made available to the scientists, who will make sure that all data are suitably converted to ensure a coherent composition. In Taverna, the scientist is made aware of the incompatability of data types and formats (encoded as MIME types) by allowing them only to make links between the output of one activity and the input of another with the same type. To that end, Taverna relies on the WSDL interface files of services and workflows, the details of which are hidden from the scientists by the user interface.

differ in that the institution is the one hosting the service, and that the services do not tend to have sub-activities associated with them.

3.2 Experiment-time Workflow Discovery

Scientists undertake their research by iteratively selecting and running workflows and further analysing the data they produce. myGrid aids this process by providing the myGrid workbench, a client side electronic lab-book through which users can perform their in silico experiments, as well as storing and organising their data. A typical work pattern of the scientist consists of selecting a piece of data stored locally and asking which workflows will accept inputs with such biological type. In the screenshot of Figure 3, the user has selected a piece of data, which is an Affymetrix Probe Set ID referring to candidate gene data in the Affymetrix database [2], and asked to find a workflow that is capable of taking this data as input.

User requirements [10] have identified that bioinformaticians also want to be involved in the process of choosing which experiments to run, and therefore, the myGrid system does not offer fully automated workflow selection. Instead, the user is presented with a list of workflow scripts and invited to make the final selection. In Figure 3, two applicable workflows have been discovered and displayed in a list with the workflow graphical depiction shown to the right, on selection.

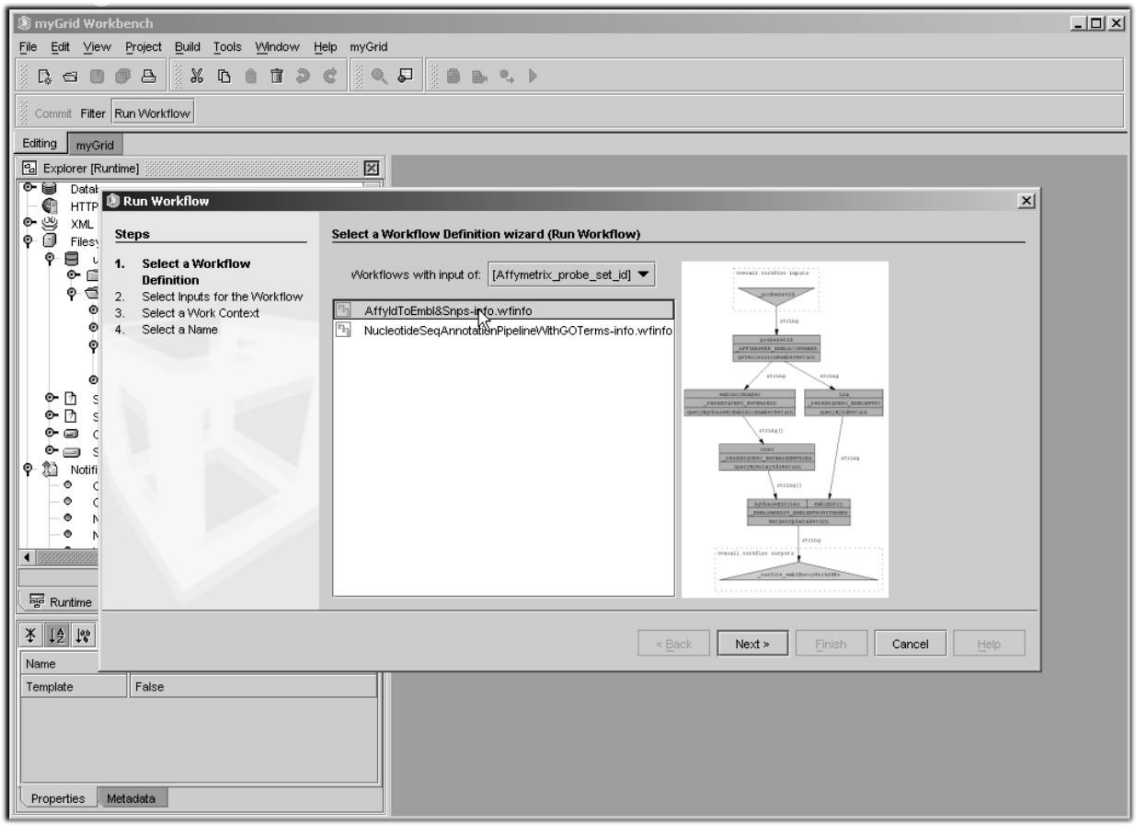

FIGURE 3: Selecting workflow that takes an Affymetrix probe set ID as input. The user has selected a piece of data, which is an Affymetrix Probe Set ID referring to candidate gene data in the Affymetrix database [2], and asked to find a workflow that is capable of taking this data as input.

As well as this data-driven context-sensitive method for discovering experiments, we also wish to enable task-orientated and result-driven approaches, by which workflows can be discovered respectively by the function they perform and by the type of output they produce. To this end, scientists need to be able to browse through published workflows, which have been categorised according their inputs (data-driven), their functionality (task-oriented) and their outputs (result-driven). Figure 4 illustrates

the user browsing available activities categorised according to those three axes[2].

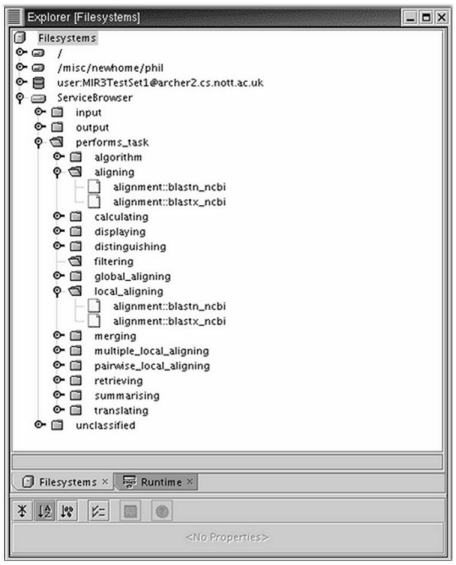

FIGURE 4: Browsing categorised workflows and services. As shown, the user can see two services/-workflows available to do sequence alignment on a gene sequence, using the services BLASTn and BLASTx.

3.3 Workflow Descriptions

Scientists require descriptions so they can judge which workflow is applicable amongst the many available. While the final decision remains with the scientists, we expect the system to help them by sorting workflows according to the various aspects (inputs, outputs, functionality), and possibly to rank them. Therefore, descriptions need to be easily processable by the computer.

Workflow descriptions can be produced by workflow authors, but they need not. Indeed, our experience in myGrid shows that it is useful for a third-party to be able to provide such descriptions. For example, a description that contains useful information about the quality, accuracy or trustability of the results produced by an experiment should typically be provided by end users, rather than the workflow authors. Likewise, a reference ontology of the application domain may be revised after some experiments have been designed; it may then be useful that an ontology expert refines semantic descriptions according to the revised ontology.

Therefore, in myGrid, we allow third-party users to generate workflow descriptions, and provide a separate tool to help users to construct such descriptions. The tool, called Pedro, is displayed in Figure 5, which illustrates its use to create descriptions pertaining to the CandidateGeneAnalysis workflow.

Although we wish descriptions to be easily processable by the computer, some descriptions may be solely aimed at users in judging the applicability of a workflow and so can be written in free

[2]BLAST,"the Basic Local Alignment Search Tool" [4] is an application that encompasses a number of services used to compare a DNA or protein sequence with the large public databases of known sequences. It can therefore accept as input different types of sequence data whether protein or DNA, perform a search over one or more databases and produce its results in a variety of formats. BLAST is highly parameterisable, able to search over many databases with many types of sequence. In fact, BLAST has several instantiations specialised for different sequence types: BLASTn for searching nucleotide sequences over nucleotide sequence databases, BLASTx for nucleotide sequences over protein databases.

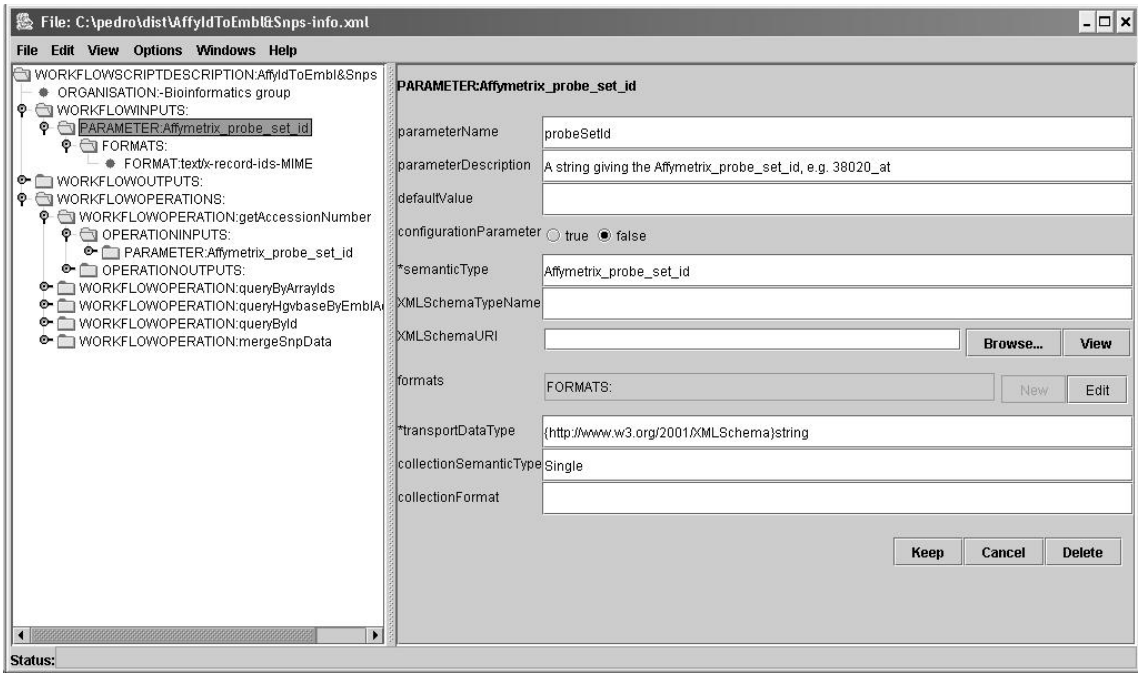

FIGURE 5: Screenshot of a workflow being annotated with semantic description using Pedro. The various components of the workflow that can be annotated with descriptions are displayed in the left hand panel. At a high level, the workflow can be annotated with the organisation that has produced it and with information about the type of biological data it takes as input and produces as output and the overall biological function it performs. The user has focused on a particular workflow sub-activity (named here WORKFLOWOPERATION) and is providing information about one of the inputs (called a PARAMETER) to that sub-activity, specifying a bioinformatics term, 'Affymetrix_probe_set_id', that refers to the type and origin of the data taken by the operation as input.

text. Figure 5 illustrates both forms of annotation. In parameterDescription, a free text description has been added to assist manual search and browsing of workflows. However, fields marked with an asterisk (semanticType and transportDataType) are populated with concepts from a controlled vocabulary. So, for example, Affymetrix_probe_set_id is a term in the myGrid bioinformatics ontology, which provides a controlled vocabulary for bioinformatics terms. Pedro has the ability to choose the controlled vocabulary that is applicable for each field of the annotation by focusing in on a particular region of an ontology. To aid the user in identifying the suitable terms of an ontology to select, the concepts of the bioinformatics ontology can be browsed, as illustrated by Figure 6.

3.4 Run-time Discovery

We have found that users wish to be involved in making the final selection of workflows to be included in their scientific experiments. Therefore, all experiment-related workflows will be chosen at composition time, and we do not anticipate that any of these will be discovered at run-time, i.e. when experiments are being enacted. On the other hand, there exist experimentally neutral workflows, which are composed of activities without any specific biological function ascribed to them (e.g., format conversions, pretty printers). Such workflows could be discoverable at run-time without involving the user. Likewise, multiple providers may host instances of a same service, and these should be automatically discoverable to make better use of resources that are available at runtime. Currently, we consider that discovery can only take place for workflows (and services) that have a functionality and fully-defined interface identified at composition time

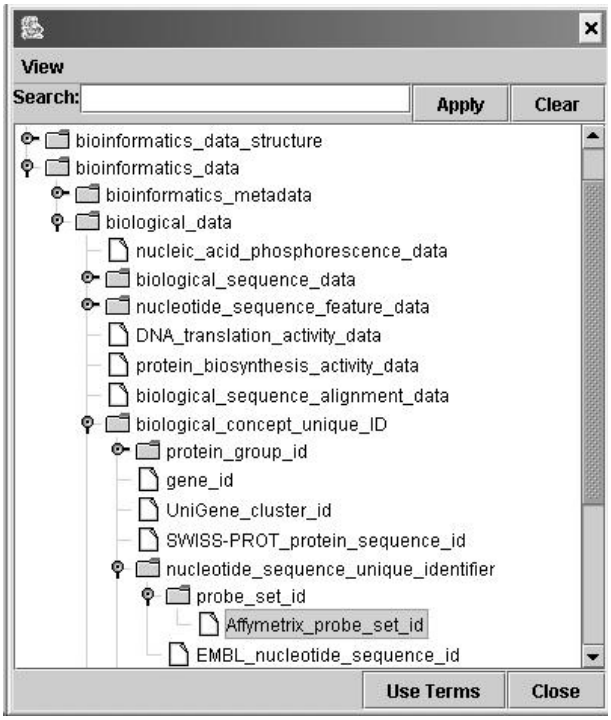

FIGURE 6: Finding the ontology term for describing the workflow's output in the myGrid ontology. The user has followed a classification of the ontology terms, and has found the term 'AffymetrixProbeSetId' which represents an entry in the scientist's database, and will be an input of the CandidateGeneAnalysis workflow.

4 Metadata for workflow discovery

The previous discussion has shown that workflows and services share many common requirements in terms of discovery. During the composition phase, they are nearly undistinguishable, except for the fact that workflows capture a scientific process, and therefore need to expose some of their internal activities to support the scientist's judgement. Fully automatic discovery of potential workflows is undesirable; this would be equivalent to automating scientific investigations and would rob scientists of the essential control of their own experiment. Examples of experimentally neutral workflows are comparatively rare and confined to sub-workflows such as format transformations or data cleaning [35].

To support the discovery process a range of descriptions are associated with a workflow. These descriptions should be:

- Produceable by authors and third-party users;
- Computer processable so that the system can present the user with relevant choices;
- Extensible;
- Based on ontologies so that suitable classifications can be shown to users.

Following this set of requirements, we introduce the notion of a workflow executive summary, which captures the aspects that can facilitate the discovery of a workflow. Specifically, the executive summary includes the following descriptions:

- The overall functional task (or tasks if there is more than one interpretational viewpoint) that a workflow performs expressed in biological terms;
- The type of data that it takes as input and/or produces as output;
- The activities that a workflow is composed of (and their respective descriptions);
- Factual information about the workflow, such as name, organisation producing it, and location.
- Factual information about the provenance of the workflow, such as the authors, and its creation and update history.

For completeness, we note that the workflow executive summary should be differentiated from operational descriptions, which contain information about workflow execution, such as cost, quality of service and access rights. Figure 7 shows the three categories of descriptions commonly used when making a choice: those catering for the executive summary of the workflow, those covering general metadata about the operational context of the workflow as a whole, and those covering the metadata about the provenance of the workflow as a whole (we do not discuss the provenance metadata further in this paper). The executive summary requires descriptions at three levels of abstraction:

- Mandatory interface description and workflow script URI that specify how to enact the workflow, and express the transport data types that the workflow expects and produces;
- Optional syntactic descriptions which might include MIME types of the input and output data, expressing the format in which data is encoded;
- Optional conceptual descriptions that enables users which to discover services based on their knowledge of the specific domain, in this case bioinformatics. We use a controlled vocabulary of terms to describe the biological data types, functions and component resources.

The development of controlled vocabularies and the annotation of workflows with them at publication time are both labour intensive activities. We do not wish to preclude those registered workflows that do not enjoy these descriptions, and so we make them optional, with the commensurate diminished functionality that attends such an omission.

This rich descriptive framework is intended to achieve various discovery capabilities at different times. Interface and syntactic descriptions are used at run time; semantic and syntactic descriptions at the point of composition and experimental selection; operational descriptions at all times [35].

To represent the knowledge embodied in the descriptions we have adopted a hybrid approach, combining two Semantic Web technologies, namely OWL and RDF.

4.1 Representing Semantic metadata: OWL

The representation of conceptual metadata requires encoding a large body of domain knowledge, with a large and highly interconnected set of terms. There has been a significant amount of work on using ontologies to describe Web Services to enable their discovery and composition [5]. DAML-S aims to address the semantic encoding of invocation and execution monitoring of services and of service compositions, but the use of semantics in myGrid has focused primarily on service and workflow discovery.

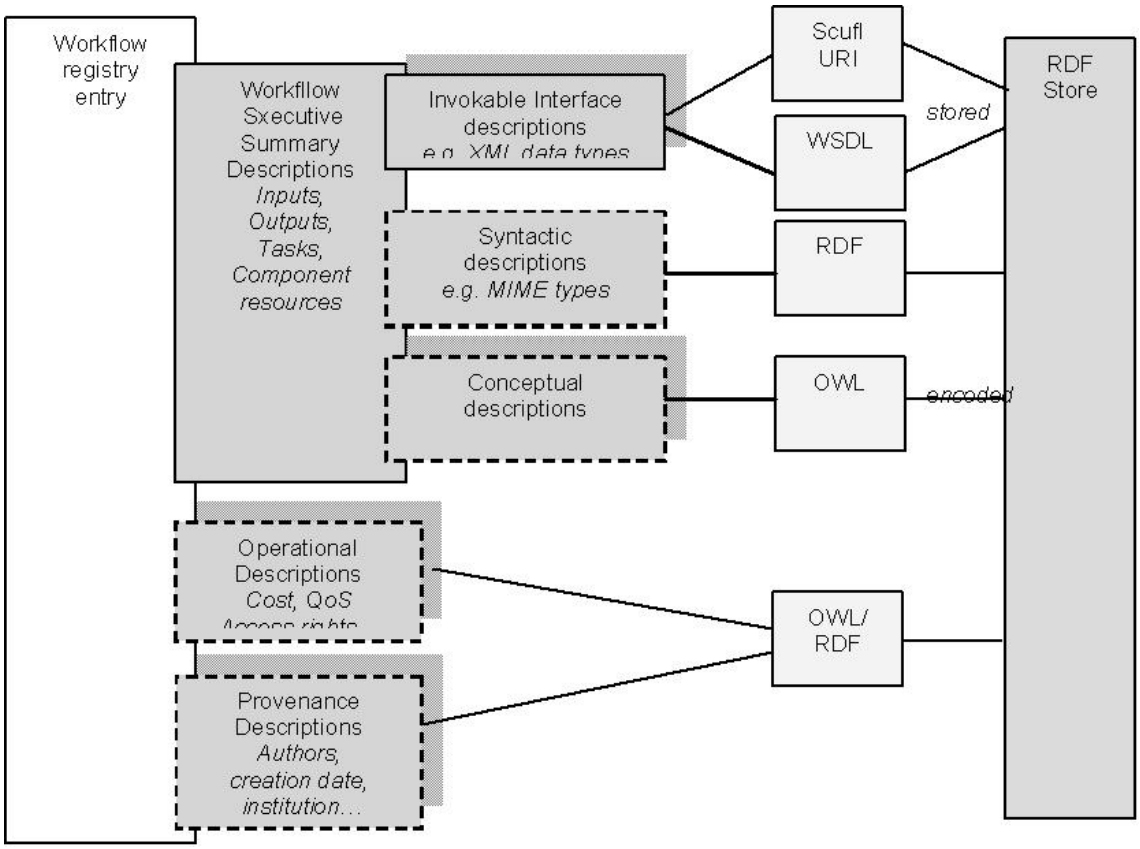

FIGURE 7: The metadata associated with a registered workflow, giving their knowledge representational forms (RDF, OWL, WSDL), all of which bottom out in an RDF store. Broken lines indicate optional metadata; shadows indicate multiple metadata entries are possible.

Within myGrid, we built a suite of ontologies, describing biology, bioinformatics, web services and workflows [35]. We based the workflow ontology on the service profile from DAML-S [5], the domain ontologies on various de facto community standard ontologies such as the Gene Ontology [9] and TAMBIS [6], and models of publication and organisation based on the AKT ontology [3].

The OWL Web Ontology Language has recently emerged as the W3C Proposed Recommendation for representing ontologies [15]. The majority of work in Semantic Web Services has used either OWL or its predecessor, DAML+OIL, and we fall in line with this practice, not only because it is an exchange standard, but because the use of OWL provides us with a number of advantages.

OWL's underlying formal semantics enables reasoners to classify descriptions based on the properties of those descriptions. This provides computational support to enable the building of complex ontologies of the domain. Additionally, when applied to workflow discovery, we automatically classify and discover workflows described in terms of a domain and workflow ontology. Consequently, it is natural for us to form queries for workflows (and services) in terms of their properties. For example, the query below describes a service in terms of the task that it performs. Equally, we can express queries for workflows and services by each of their executive summary components. Queries of this form can be presented in a browsable interface, as shown in Figure 4. This interface takes advantage of the simple expressive capabilities of OWL, in that workflows and services will classify under many different parents, for instance most of the services shown as performing aligning will also classify under "local_aligning"; the latter being a specialisation of the former. The reasoning capabilities of OWL mean that we are not required to pre-enumerate at design time all of the possible workflows classifications, but can generate new ones on the fly, or even change the classifications of services as

we change our ideas about the domain.

$$
\begin{array}{l}
intersectionOf(\\
\quad myGrid\_bioinformatics\_primitive\_service\_operation \\
\quad restriction(performs\_task someValuesFrom(aligning)) \\
)
\end{array}
$$

In many cases, this use of reasoning for forming classifications is sufficient. The multi-axial classifications shown in Figure 4 actually present a large number of different workflow/service classifications, which narrow the choices to a point where the user can select for themselves those that they require. By using OWL, we can also exploit the full expressiveness of this language to build highly complex queries, which we can use to enable more automated selection.

However, the use of OWL also brings some difficulties. The use of reasoning technology can complicate the architecture required to support it. Furthermore, while OWL can be used to present relatively straightforward interfaces for the selection of workflows, it comes with an upfront cost, namely that of producing a large domain ontology, and then describing the workflows in terms of that ontology. At the current time this cost is considerable, although it is hoped that this should lessen as tools, such as Pedro, develop further. For these reasons, we would expect that the primary use of OWL based service or workflow descriptions will be in a curated set of services, workflow, or third party descriptions, for use within a system such myGrid, rather than as a general tool for descriptions of Web Services in general. As a result, within myGrid, we also provide support for workflow discovery based on other description technologies, as detailed in the next section.

4.2 The Use of RDF

Our workflow descriptions have to draw on and seamlessly integrate multiple existing information models, namely WSDL, DAMLS-Profile, and UDDI, and have to support metadata attachment, as we now explain.

- Interfaces have been identified as useful information in the discovery process. As we focus on Web Services, we adopt WSDL as the interface language for services, and we use the same language to define the interface of a workflow, composed of its inputs and outputs.
- Semantic augmentation by authors and third parties requires a mechanism by which additional semantic descriptions can be attached to existing workflow descriptions; hence, our information model requires support for metadata.
- Furthermore, the semantic functionality of a workflow will be structured using OWL, and the myGrid ontologies, as discussed in Section 4.1.
- Additionally, we have identified that runtime discovery could take place for workflows and services, for which an interface and functionality have been identified at composition time. The de-facto standard for Web Service discovery is UDDI; adopting the UDDI information model will help us preserve compatibility with existing systems (such as enactment engines).

We have adopted RDF [27] as the representation formalism to express such complex service descriptions. RDF is a very flexible language in which relations are described between resources, in the form of triples. A triple associates one resource, the subject, to another, the object, by a relation labelled with a specific property. Our reasons for using RDF are based on the technical requirements of publishing and discovery.

- RDF can capture arbitrarily structured metadata, including semantic descriptions that refer to terms in an ontology; it provides a uniform language in which to express multiple information models (UDDI, WSFL, DAML-S).
- RDF is naturally designed to express the attachment of metadata to existing concepts of a workflow description; such a capability is ideally suited to our semantic augmentations.
- Once all the information is expressed uniformly in RDF, it can be searched uniformly (both for data and metadata) using graph-based queries, which can easily be expressed in languages such as RDQL.

In summary, myGrid has adopted a hybrid approach for its knowledge representation. RDF is the underpinning format in which all workflow (and service) descriptions are encoded. This is an extensible format, which provides us with a powerful graph-based query capability using RDQL. The rest of the paper will discuss how this RDF-based information model is used in the registry that holds all workflow descriptions, and which provides us with efficient query capabilities necessary for run-time discovery. Within the registry the semantic executive summary metadata and some of the operational metadata contain semantic descriptions referring to OWL concepts. Semantic reasoning is undertaken by a semantic find component, which deals with the semantic-rich discovery process taking place at construction and experiment selection time.

5 myGrid Protocols for Publishing and Discovery

The myGrid architecture defines protocols for publishing workflows and their executive summaries, and for performing discovery based on those descriptions. The two principal components involved in publication and discovery are the registry, which holds the advertisements for workflows, and the semantic find component, which aids discovery of workflows by matching semantic queries against the semantic descriptions in the registry.

5.1 Encoding a workflow executive summary

The starting point for advertising a workflow is the authoring of semantic descriptions, as described in Section 3.2, and this requires the author of the workflow description to know what components of the workflow they can describe and how. A key requirement of our architecture is that it must support multiple workflow languages, or versions of them, because the myGrid Scufl workflow language is still evolving. Ultimately, this should help to ensure that the architecture is future-proof. So, we have introduced the notion of a workflow executive summary as an abstraction of a workflow, independent of any particular scripting language. At authoring time, usually within the Pedro tool, this executive summary is represented in an XML schema, which is shown in Figure 8.

5.2 Publishing and discovery protocols

The process of publishing a workflow in myGrid is shown in Figure 9 and Figure 10. Overall, the publishing process involves the user, the workflow construction and annotation tools, a storage device to archive workflows, a registry in which advertising and searching are performed and a semantic find component performing any necessary reasoning over any ontology-based semantic descriptions. Our sequence diagrams regard these components as separate, but any specific deployment may seek to

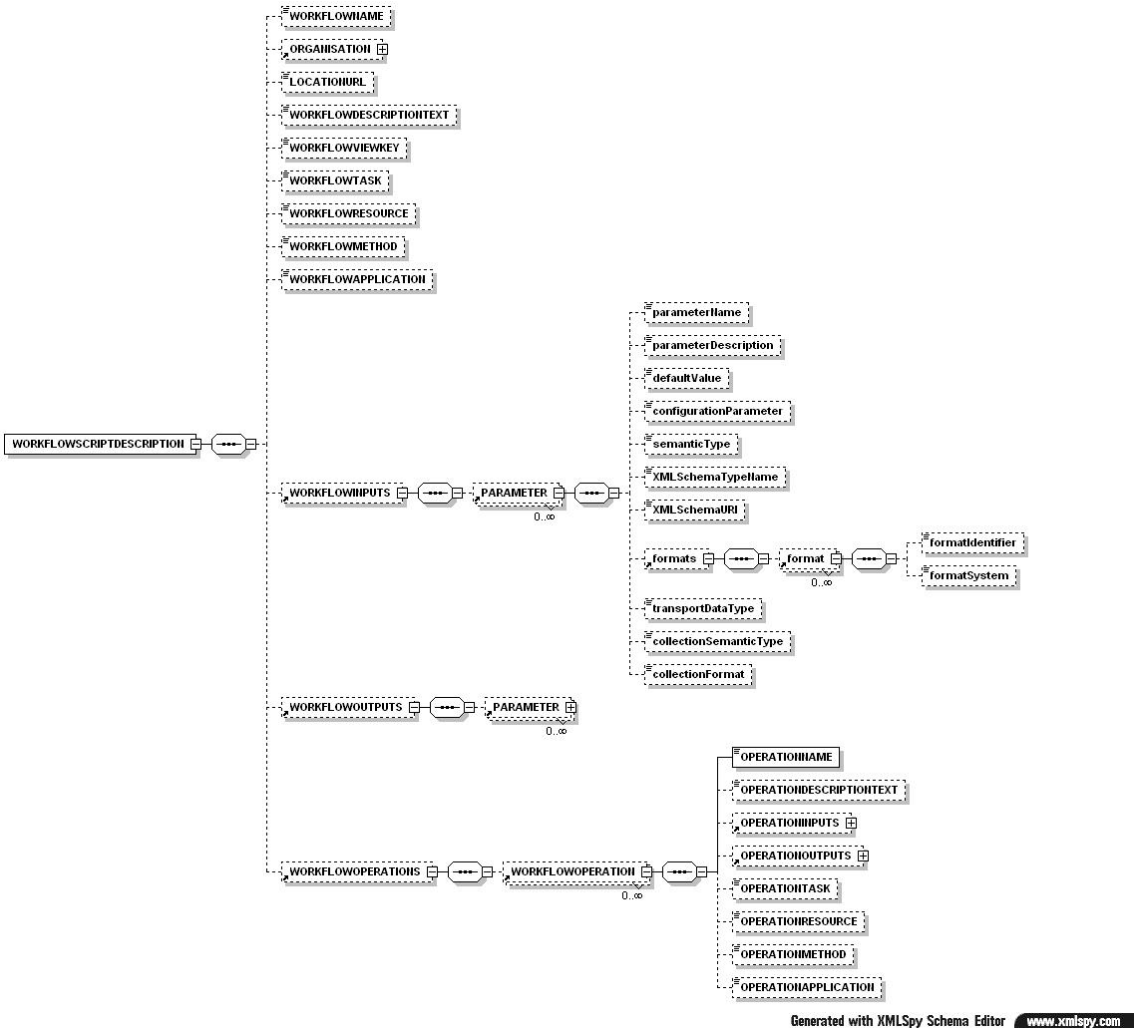

FIGURE 8: Contents of workflow executive summary. The workflow executive summary entities that can be annotated by semantic descriptions are shown in the left hand panel of Pedro in Figure 5, and are the same as those in this figure. Data derived from the invocation metadata, typically an XSD type, used by the SOAP transport layer, is encoded as transportDataType in the executive summary, while the conceptual metadata, is encoded as an arbitrary OWL concept in semanticType. Finally, syntactic metadata normally represented as a MIME type is represented in format. For clarity we have omitted the workflow provenance metadata from this figure.

integrate some of them. The workflow script is archived in a store and made available via a URI, which is advertised in the registry by the user, possibly using a workflow construction tool. Then semantic descriptions and other metadata are attached to the workflow, its inputs and its outputs through successive calls to the directory (see Figure 9). Whenever a new workflow and new metadata are added to the directory, a notification is sent to the semantic find component, with the advertisement referring to the workflow by a unique key; as a result, the semantic find component indexes the workflows by their semantic types in order to support efficient discovery.

The discovery of workflows, or other activities, is shown in Figure 11. Within myGrid, there are two main reasons for discovery: firstly in response to a user request, usually through interaction with the workbench; and secondly during the process of resolving the abstract activity specifications into invokable instances (see Section 3.4).

As users generally wish to discover services in terms of their own domain, this discovery normally

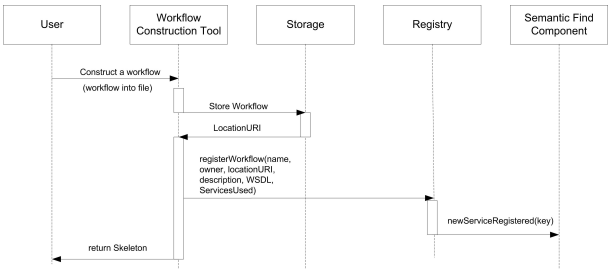

FIGURE 9: Sequence of actions taken in publishing workflow.

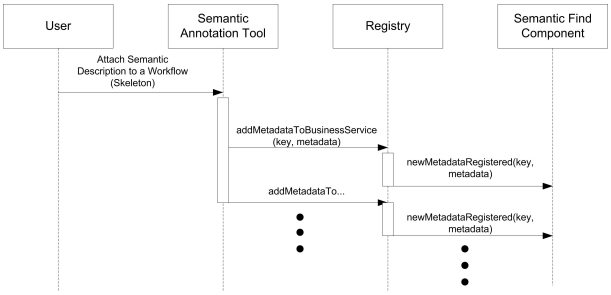

FIGURE 10: Sequence of actions taken in attaching metadata to a workflow.

involves the conceptual metadata, and is shown in Figure 11. Following user activity involving either the context sensitive workflow selection, or browsing interfaces shown in Section 3, the workbench generates a semantic query. This query is sent to the semantic find component, which uses the retrieved semantic descriptions to determine which workflows match the query. The technical details, including the name, interface and endpoint of each applicable workflow script, are extracted from the registry and returned to be displayed to the user. The user can then select the final workflow, if there is more than one, which will then be sent to the enactor.

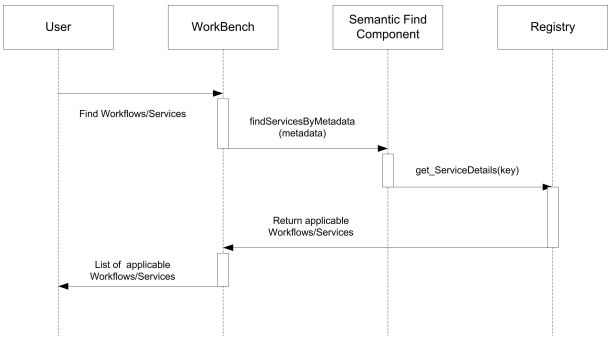

FIGURE 11: Sequence of actions taken in discovering workflow by user through a user interface.

The enactor may also use the registry at run-time. As described earlier a workflow template describes an in silico experiment, where some activity definitions have been defined abstractly by service types rather than end points of specific Web Services. In this case, queries will generally involve the invocation metadata, and will involve only the registry, as shown in Figure 12. Following discovery the enactor can then continue with invoking the returned service.

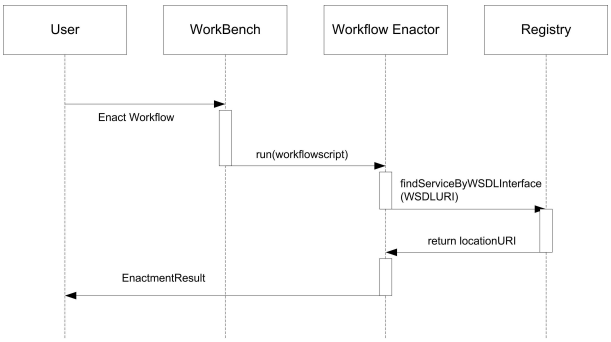

FIGURE 12: Sequence of actions taken in discovering services during workflow enactment.

5.3 Discussion

The design decisions involved in developing the above protocol are driven by the user and technical requirements. The motivation for treating the registry and the semantic find component as two separate modules, and passing messages between them, is that only discovery involving conceptual metadata will require semantic reasoning. So, discovery by the workflow enactment engine will attempt to match a service by its interfaces, ensuring that it can accept the data produced by earlier activities in the workflow, rather than its domain-specific, e.g. biological, type. While conceptually separate, these two modules can be tightly integrated in any specific implementation in order to improve efficiency. The following section will discuss alternative deployments of the semantic find component.

6 Implementation

In this section, we describe the design and implementation of the main components of the myGrid architecture used in publishing and discovering workflows, namely the registry and the semantic find component.

6.1 Registry

Existing protocols for service publishing and discovery, such as UDDI for Web Services [31], do not provide support for workflows. We have taken the approach that workflow scripts and services are almost equivalent for the purpose of discovery. Both are functional entities taking inputs and producing outputs according to some interface and internal algorithm and are available from a given endpoint (where to download the script from in the case of a workflow). Executing them requires different processes, but this is relevant only to enactment and not to the advertising of the workflow/service. By drawing this equivalence between services and workflows, we can reuse the UDDI API to enable their registration and discovery.

The difference in execution, however, does mean that it needs to be obvious which type of activity the advert applies to. This requires us to attach additional metadata to the advert. In previous sections we have also identified the need to attach other additional metadata, in the form of OWL, or RDF to the activity in the registry. Therefore we have built the myGrid registry to be UDDI-compliant, but, in addition, we have specified a protocol for attaching metadata to activities described in the service registry [21]. The metadata can be a simple string value for recording, for example, an estimate of the average time a workflow takes to execute. Alternatively, it can be a URI, to a concept in the ontology.

For a more complex semantic description, for example, in which ontology concepts are qualified by property values, structured RDF [27] metadata can be attached. The message structure for one metadata attachment method is given in Figure 13; similar methods also exist to attach metadata to a service (or workflow), a business, and to query for services or workflows by the metadata attached to them.

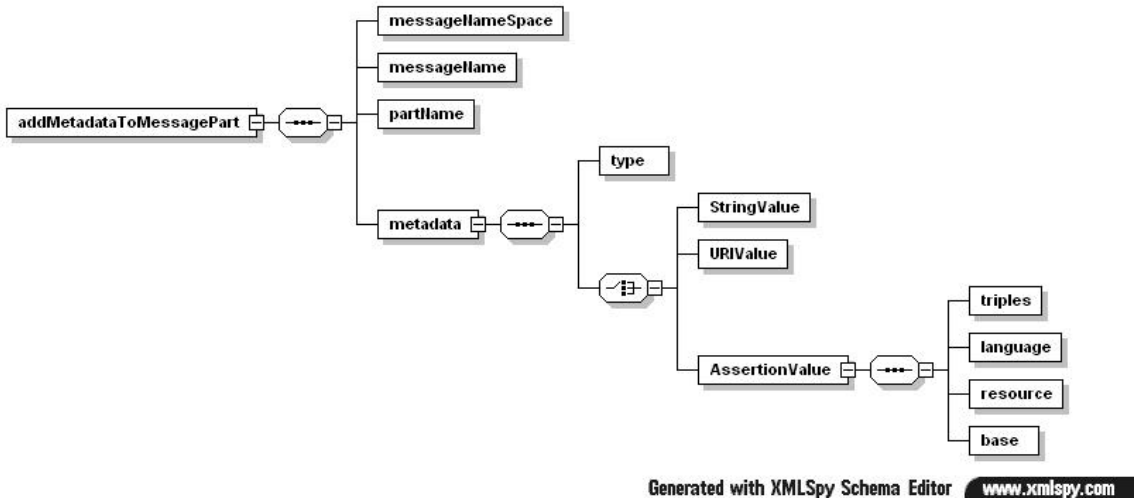

FIGURE 13: API for attaching metadata to WSDL message parts (inputs or outputs of workflows). To attach metadata the client must identify the entity to which metadata is attached and provide all details of the metadata itself. In this case, a message part is uniquely identified, according to the WSDL specification, by the namespace and local name of the message containing that part plus the part name. Metadata in our registry is given a type, by which the client can determine what the metadata is about, and a value. The value may be either a string, a URI (usually an ontology term) or structured metadata expressed as an assertion in one of the triple languages (such as RDF XML or N3).

A key characteristic of the registry is that the underlying information is stored as RDF [27] in a Jena [17] triple store, for reasons discussed in Section 4.2. For completeness, Appendix 1 contains the RDF representation of the CandidateGeneAnalysis workflow advertisment, as contained in the registry. Figure 14 shows the architecture of the registry, which is available as a Web Service in the myGrid distribution. The client interacts with the registry through a set of interfaces, which allow services and workflows to be published and discovered again as UDDI business services, metadata, either conceptual, or operational to be attached and used in discovery. Other features of the registry include sending of notifications when services, workflows and metadata are added or removed, third party annotations of services, federation of the registry and policy-based management of its contents but these are beyond the scope of this paper.

6.2 Semantic Find Component

The myGrid semantic find component is responsible for analysing and making inferences over conceptual metadata, and is used for querying over activities described with this metadata. As this component receives queries expressed in OWL, we can use it to broaden or narrow searches as required. For example, by adding properties to an OWL concept expression we specialise the query and narrow the number of candidate workflows (we travel down the classification lattice); by removing properties we broaden the query and extend the number of candidate workflows that will be classified by the expression (we travel up the classification lattice) [36]. The architecture is depicted in Figure 15. The semantic find component itself is responsible for the following.

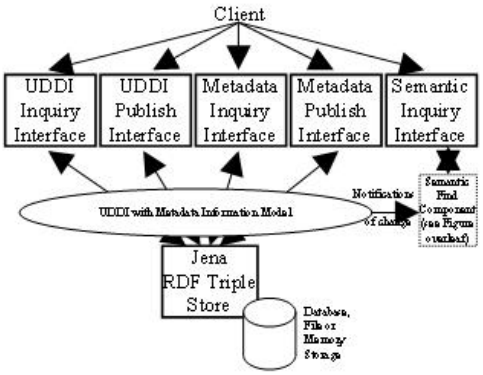

FIGURE 14: Architecture of the Registry.

- Every time a new service is advertised or metadata is updated, the ontology service and associated reasoner indexes items in a descriptions database to ensure efficient retrieval of entries at time of discovery. Storing the descriptions in a commodity database, as opposed to the mature description logic reasoner technology also has obvious advantages for scalability of the system in practice. A fuller description of this technology is available elsewhere [16].
- Discovery queries are processed using the pre-built index or if necessary the ontology service and associated reasoner.

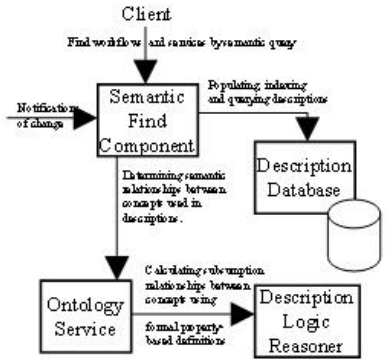

FIGURE 15: Architecture of the Semantic Find Component. The description database holds semantic descriptions gathered from resources published in the registry; the ontology server provides access to the domain ontologies and manages interaction with the description logic reasoner FaCT [14].

Two deployments of the semantic find component are considered. As illustrated in Figure 13, the semantic find component can be embedded in the registry, with queries over the conceptual metadata being processed by the semantic find component, while non conceptual queries would be answered by the registry. Alternatively, the component can be deployed as an autonomous service able to reason over semantic descriptions from a variety of sources including databases and Web pages.

Exact details of the semantic matching algorithms whereby a resource description is matched to semantic query should not impact directly on the architecture described in this paper. In early implementations of this service, we have performed simple subsumption matching between query

and description, although matching algorithms such as those described elsewhere [26] could also be supported.

7 Related Work

The Web Service Architecture details the existence of a directory service for the registration and subsequent discovery of services, and languages for the composition of services into workflows. For directories, the UDDI [31] registry (Universal Description, Discovery, and Integration) has become the de facto standard. Service descriptions in UDDI are composed from a limited set of high-level data constructs (Business Entity, Business Service etc.) which can include other constructs following a rigid schema. Some of these constructs, such as tModels, Category Bags and Identifier Bags, can be seen as metadata associated with the service description. However, while useful in a limited way, they are all restrictive in scope of description and their use in searching the directory. We extend UDDI by allowing arbitrary structured metadata to be attached to not only the services and workflows published, but also to their interfaces.

For workflow languages, numerous candidates have been proposed, including: BPEL4WS (Business Process Execution Language for Web Services) [8], Web Services Flow Language (WSFL) [33], XLANG (Web Services for Business Process Design) [39] and Scufl (Simple Conceptual Unified Flow Language) [29]. These languages differ in their expressiveness and flexibility. It is unlikely that in the foreseeable future a single workflow language will emerge as a universal standard, although there is some encouraging development in this direction represented by BPEL4WS, which integrates the key features of WSFL and XLANG. In myGrid, we have used Scufl to provide a simple representation of the activities of a workflow in such a way that it is easy for a bioinformatician to conceptualise and manipulate the overall experimental design by abstracting away from the details of low level service orchestration [1].

The motivation to discover and compose Web Services in automated and intelligent ways has fuelled many researchers from the Semantic Web community to apply knowledge technologies to service descriptions, often building on past work in Problem Solving Methods [11]. Early work focused on semantic service discovery [5]; more recent work has shifted to automated intelligent service composition, primarily through the use of AI planning techniques [38]. Our semantic descriptions support the composition of services by enabling semantic and syntactic capability checking of input and output types; however, we do not support automated workflow planning as the plan is the biologist's experiment and our experiences suggest they demand complete control over the definition. DAML-S [5] attempts a full description of a service as a process that can be enacted to achieve a goal. A full DAML-S service description incorporates three component perspectives: a planning view of service based on inputs, outputs, preconditions, and effects (the service profile); the workflow view of the more primitive services needed to accomplish a complex goal (the service process); the mapping of the atomic parts of this workflow to their concrete WSDL [37] descriptions (the service grounding). DAML-S provides an alternate mechanism that allows service publishers to attach semantic information to the parameters of a service. Indeed, the argument types referred to by the profile input and output parameters are semantic. Such semantic types are mapped to the syntactic type specified in the WSDL interface by the intermediary of the service grounding. Such a mechanism is welcome but convoluted and limited. The mapping from semantic to syntactic types involves the process model, and it only supports semantic annotations provided by the publisher, and not by third party annotators; a profile only supports one semantic description per parameter and does not allow multiple interpretations. Finally, such semantic annotations are restricted to input and output parameters, but may not be applied in a similar manner to other elements of a WSDL interface specification, e.g. operations or sets of operations collected in port types.

From the distributed Grid computing community, the ICENI project uses OWL for semantic annotation [13] but so far deals only with the ontological description of service interfaces, ignoring other aspects such as the semantic annotation of WSDL documents and workflow discovery. Because the descriptions are added directly to the interfaces in the source code, only the service provider can publish semantic descriptions (not third parties), which imposes restriction on the community using the system. We have opted for the use of a flexible structure which enables annotation with both semantic and other metadata, by both service providers and third parties.

Finally, the biology domain has been investigating its own mechanisms for publishing bio-Web Services. The most well known proposal is BioMOBY [34], a service discovery architecture based on a view of a service as an atomic process or operation that takes a set of inputs and produces a set of outputs. The service, inputs and outputs are given semantic types which also define the message format. However, BioMOBY has a number of limitations: it does not support the UDDI protocol, so specialist clients have to be developed; it does not have a general attachment mechanism for service descriptions; and it does not explicit address the publishing or discovery of workflows. myGrid and BioMOBY are working closely together to develop a common semantic registry framework.

8 Discussion and Future Work

We have demonstrated how the myGrid architecture can be used to construct, publish, semantically describe, annotate and discover workflows as part of scientists' experimental processes. Scientists without detailed computer science knowledge wish to share and use each others' experimental designs, but discovering the designs available becomes difficult when there is a large and increasing number available in a distributed system such as the Web. The myGrid architecture, making use of Web Services, workflows, enhanced service discovery technologies, Semantic Web technologies and semantic descriptions enables scientists to do this more easily. We have shown how the process takes place from the users' perspective and presented the underlying protocol implemented by our middleware.

We recognise that there can be multiple registries owned by different people and organisations, in which many useful workflows may be published. For this reason, future work on the registry will concentrate on federation of registries and the personalisation of registries to contain the information most useful to individuals, which could include semantic descriptions other than those provided by the workflow author.

It is useful to specify activities at construction time without restricting them to a particular interface. These workflow templates contain abstract descriptions in place of one or more services or sub-workflows. However, in practice we find that services with the exact same functionality still often require different ways of being enacted and so cannot be easily substituted one for another [36]. For instance, one of the ways in which an activity can be distinguished from another is in its invocation model, so that one service may perform a function with one operation call that requires multiple calls to another service (the example given in [35] is of different deployments of the BLAST service discussed in Section 3).

The discovery of workflows by the type of input, and classifying them by function for browsing by the user, turn out to be the most helpful applications of the semantic descriptions provided. It has been clear that better tools for the attachment and, later, maintenance (if mistakes or imprecision is found) of semantic descriptions are required, as the annotator should be an expert in the domain of the descriptions rather than the languages and structures in which the description is expressed. Future work in tools will concentrate on two areas: making the publication of semantic annotations

incidental and making discovery invisible in the sense that the user sees the workflow discovery as a part of their natural scientific environment in their terms.

A RDF Representation of a Published Workflow

Below, we find the representation of a published workflow description stored in RDF (in N3 format). The workflow is advertised, following the UDDI specification, as a BusinessService, and marked as a workflow by attaching metadata ('isWorkflowScript') (1). The workflow refers to the location of the workflow script ('AccessPoint') (2) and its WSDL interface. The interface element is further expanded to show the messages that are accepted as input (3) and returned as output, and metadata is added to provide the syntactic (4), semantic (biodata:Affymetrix_probe_set_id) (5) and MIME (6) types.

```
# Base:
 @prefix biodata:  <http://www.ecs.soton.ac.uk/~sm/myGrid/myGrid.daml#>
 @prefix registry:  <http://www.myGrid.ecs.soton.ac.uk/myGrid.rdfs#> .
 @prefix wsdl:  <http://www.myGrid.ecs.soton.ac.uk/wsdl.rdfs#> .
 @prefix uddiv2:  <http://www.myGrid.ecs.soton.ac.uk/uddiv2.rdfs#> .
 @prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

[] a <uddiv2:BusinessService> ;
  <uddiv2:hasName>
    [ a <uddiv2:NameBag> ;
      <rdf:_1> "CandidateGeneAnalysis"
    ] ;
  <uddiv2:hasServiceKey> "d0892afd-198d-404b-bfdf-31c7fa4df8f3" ;
  <uddiv2:hasMetadata>
  [ a <isWorkflowScript> ; (1)
    <rdf:value> "yes" ;
  ] ;
  <uddiv2:hasBindingTemplate>  ...
      [ a <uddiv2:AccessPoint> ; (2)
        <uddiv2:hasText>
          "http://www.ecs.soton.ac.uk/~sm/myGrid/CandidateGeneAnalysis.scufl" ;
        <uddiv2:hasURLType> "http"
      ]  ...
            <uddiv2:hasOverviewDoc>
            [ a <uddiv2:OverviewDoc> ;
              <uddiv2:hasOverviewURL>
                "http://www.ecs.soton.ac.uk/~sm/myGrid/CandidateGeneAnalysis.wsdl"  ...
[] a <wsdl:WSDLOverviewDoc> ;
  <wsdl:hasFilename> "http://www.ecs.soton.ac.uk/~sm/myGrid/CandidateGeneAnalysis.wsdl" ;
  <wsdl:hasMessage>
  [ a <wsdl:MessageBag> ;
    <rdf:_1>
    [ _:b1 <wsdl:Message> ; (3)
      <wsdl:hasQName>
      [ a <wsdl:QName> ;
        <wsdl:hasLocalName> "WholeWorkflowRunRequest" ;
        <wsdl:hasNameSpace> "http://www.ecs.soton.ac.uk/~sm/myGrid/myGrid.daml
      ]
      <wsdl:hasMessagePart>
      [ a <wsdl:PartBag> ;
        <rdf:_1>
        [ a <wsdl:MessagePart> ;
          <wsdl:hasName> "probeSetId" ;
          <wsdl:hasTypeName>
          [ a <wsdl:QName> ;
            <wsdl:hasLocalName> "string" ; (4)
            <wsdl:hasNameSpace> "http://www.w3.org/2001/XMLSchema"
          ] ;
          <uddiv2:hasMetadata>
          [ a <biodata:semanticType> ; (5)
            <rdf:value> "biodata:Affymetrix_probe_set_id" ;
          ];
          <uddiv2:hasMetadata>
          [ a <biodata:formats> ;
            <rdf:value>
            [ a <biodata:formatBag> ;
              <rdf:_1>
              [ a <biodata:format> ;
                <biodata:hasFormatSystem> "MIME" ; (6)
                <biodata:hasFormatIdentifier> "text/x-record-ids" ;
              ]
            ]
          ] ;
...
```

References

[1] Matthew Addis, Justin Ferris, Mark Greenwood, Darren Marvin, Peter Li, Tom Oinn, and Anil Wipat. Experiences with escience workflow specification and enactment in bioinformatics. In *Proceeding of the UK OST e-Science second All Hands Meeting 2003 (AHM03)*, pages 459–467, Nottingham, UK, September 2003.

[2] Affymetrix. http://www.affymetrix.com, 2003.

[3] Akt project. http://www.aktors.org/, 2003.

[4] S.F. Altschul, W. Gish, M. Miller, E. W. Myers, and D.J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, (215):403–410, 1990.

[5] Anupriya Ankolenkar, Mark Burstein, Jerry R. Hobbs, Ora Lassila, David L. Martin, Drew McDermott, Sheila A. McIlraith, Srini Narayanan, Massimo Paolucci, Terry R. Payne, and Katia Sycara. Daml-s: Web service description for the semantic web. In *The First International Semantic Web Conference (ISWC)*, Sardinia, Italy, June 2002.

[6] Patricia G. Baker, Carole Goble, Sean Bechhofer, Norman Paton, and Andy Brass Robert Stevens. An ontology for bioinformatics applications. *Bioinformatics*, 15(6):510–520, 1999.

[7] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5):3443, 2001.

[8] Business process execution language for web services. http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/, 2003.

[9] The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nat Genet*, 25:25–29, 2000.

[10] R. Stevens et al.˙ Performing *in silico* Experiments on the Grid: A Users Perspective. In S. Cox, editor, *Proceedings of the UK OST e-Science Second All Hands Meeting 2003*, pages 43–50, Nottingham, UK, 2003.

[11] D. Fensel and C. Bussler. The web service modeling framework wsmf. Technical report.

[12] Ian Foster and Carl Kesselman. *The Grid, Blueprint for a New Computing Infrastructure. 2nd edition.* Morgan Kaufmann, 2003.

[13] J. Hau, W. Lee, and S. Newhouse. Autonomic service adaptation using ontological annotation. In *4th International Workshop on Grid Computing, Grid 2003*, Phoenix, USA, November 2003.

[14] Ian Horrocks. Fact and ifact. in p. lambrix, a borgida, m. lenzerini, r mller, and p. patel-schneider, editors. In *Proceedings of the International Workshop on Description Logics (DL99)*, pages 133–135, 1999.

[15] Ian Horrocks, PeterF. 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), December 2003.

[16] Instance store. http://instancestore.man.ac.uk/, 2003.

[17] Jena semantic web toolkit. http://www.hlp.hp.com/semweb/jena.htm, 2003.

[18] Jini. http://www.jini.org/, 2003.

[19] Phillip Lord, Chris Wroe, Robert Stevens, Carole Goble, Simon Miles, Luc Moreau, Keith Decker, Terry Payne, and Juri Papay. Semantic and personalised service discovery. in w. k. cheung and y. ye, editors. In *Proceedings of Workshop on Knowledge Grid and Grid Intelligence (KGGI'03), in conjunction with 2003 IEEE/WIC International Conference on Web Intelligence/Intelligent Agent Technology*, pages 100–107, Halifax, Canada, October 2003. Department of Mathematics and Computing Science, Saint Mary's University, Halifax, Nova Scotia, Canada.

[20] Simon Miles, Juri Papay, Vijay Dialani, Michael Luck, Keith Decker, Terry Payne, and Luc Moreau. Personalised grid service discovery. *IEE Proceedings Software: Special Issue on Performance Engineering*, 150(4):252–256, August 2003.

[21] Simon Miles, Juri Papay, Terry Payne, Keith Decker, and Luc Moreau. Towards a protocol for the attachment of semantic descriptions to grid services. In *Proceedings of 2nd European Across Grids Conference (AxGrids 2004)*, 2004.

[22] Luc Moreau, Mike Luck, Simon Miles, Juri Papay, Keith Decker, and Terry Payne. Methodologies and software engineering for agent systems. In *Agents and the Grid: Service Discovery*. Kluwer, 2004.

[23] mygrid uk e-science project. http://www.myGrid.org.uk, 2003.

[24] National graves disease foundation frequently asked questions. http://www.ngdf.org/faq.htm, 2003.

[25] Ogsa. https://forge.gridforum.org/projects/ogsa-wg, 2003.

[26] Massimo Paolucci, Takahiro Kawamura, Terry Payne, and Katia Sycara. Semantic matching of web services capabilities. In *First International Semantic Web Conference (ISWC)*, 2002.

[27] Resource description framework (rdf). http://www.w3.org/RDF/, 2001.

[28] Rdql. http://www.hpl.hp.com/semweb/rdql.htm, 2003.

[29] Scufl simple conceptual unified flow language (scufl). http://taverna.sourceforge.net/schemata/XScufl.html, 2003.

[30] Taverna. http://taverna.sourceforge.net/, 2003.

[31] Universal description, discovery and integration of business of the web. www.uddi.org, 2001.

[32] Web services architecture. Latest version available from http://www.w3.org/2002/ws/arch/, 2003.

[33] Web services flow language. http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf, 2003.

[34] M. D. Wilkinson and M. Links. Biomoby: an open source biological web services proposal. *Briefings In Bioinformatics*, 4(3), 2002.

[35] Chris Wroe, Carole Goble, Mark Greenwood, Phillip Lord, Simon Miles, Luc Moreau, Juri Papay, and Terry Payne. Experiment automation using semantic data on a bioinformatics grid. *IEEE Intelligent Systems*, January 2004.

[36] Chris Wroe, Robert Stevens, Carole Goble, Angus Roberts, and Mark Greenwood. A suite of daml+oil ontologies to describe bioinformatics web services and data. *International Journal of Cooperative InformationSystems*, 12(2):197–224, 2003.

[37] Web services description language (wsdl) 1.1. http://www.w3c.org/TR/wsdl, 2003.

[38] Dan Wu, Bijan Parsia, and Evren Sirin et al. Automating daml-s web services composition using shop2. In *Proceeding of 2nd International Semantic Web Conference ISCW2003*, volume 2870, pages 195–210. Springer-Verlag, Heidelberg, October 2003.

[39] Xlang. $http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm$, 2003.