BIOINFORMATICS

Vol. 20 no. 17 2004, pages 3045–3054
doi:10.1093/bioinformatics/bth361

Taverna: a tool for the composition and enactment of bioinformatics workflows

Tom Oinn[1], Matthew Addis[2], Justin Ferris[2], Darren Marvin[2], Martin Senger[1], Mark Greenwood[3], Tim Carver[4], Kevin Glover[5], Matthew R. Pocock[6], Anil Wipat[6] and Peter Li[6,*]

[1]EMBL European Bioinformatics Institute, Hinxton, Cambridge, CB10 1SD, UK, [2]IT Innovation Centre, University of Southampton, SO16 7NP, UK, [3]Department of Computing Science, University of Manchester, M13 9PL, UK, [4]MRC Rosalind Franklin Centre for Genomics Research, Wellcome Trust Genome Campus, Hinxton, Cambridge, CB10 1SB, UK, [5]School of Computer Science and Information Technology, University of Nottingham, NG8 1BB, UK and [6]School of Computing Science, University of Newcastle, NE1 7RU, UK

Received on February 5, 2004; revised on May 25, 2004; accepted on May 31, 2004
Advance Access publication June 17, 2004

ABSTRACT

Motivation: *In silico* experiments in bioinformatics involve the co-ordinated use of computational tools and information repositories. A growing number of these resources are being made available with programmatic access in the form of Web services. Bioinformatics scientists will need to orchestrate these Web services in workflows as part of their analyses.

Results: The Taverna project has developed a tool for the composition and enactment of bioinformatics workflows for the life sciences community. The tool includes a workbench application which provides a graphical user interface for the composition of workflows. These workflows are written in a new language called the simple conceptual unified flow language (Scufl), where by each step within a workflow represents one atomic task. Two examples are used to illustrate the ease by which *in silico* experiments can be represented as Scufl workflows using the workbench application.

Availability: The Taverna workflow system is available as open source and can be downloaded with example Scufl workflows from http://taverna.sourceforge.net

Contact: taverna-users@lists.sourceforge.net

INTRODUCTION

An *in silico* experiment is a procedure involving the use of local and remote resources to test a hypothesis, derive a sum mary or search for patterns (Stevens *et al*., 2003). In bioinformatics, resources may be information repositories such as the EMBL and Swiss-Prot databases, or computational analysis tools like BLAST and ClustalW. The analysis performed during an *in silico* experiment frequently involves a combination of these resources which are linked in a specific order, thus forming a workflow process. For example, a workflow to investigate the evolutionary relationships between proteins might begin with acquiring amino acid sequences belonging to a protein family from Swiss-Prot and then applying the ClustalW algorithm to align and identify patterns between sequences.

Recently, organizations have started to provide programmatic access to bioinformatics information repositories and analysis tools based on Web services (Stein, 2002), a new distributed computing architecture which uses existing Internet communication and data exchange standards (Booth *et al*., 2003, W3C http://www.w3.org/TR/ws-arch/). Resources with Web service access provide a web-based, published, application programming interface for interaction with other applications. Web services have wide support in terms of commercial and open source tools for developing Web services and client applications that use them. Examples of bioinformatics Web services include the XEMBL (Wang *et al*., 2002), openBQS (Senger, 2002, http://industry.ebi.ac.uk/openBQS/) and Soaplab analysis services (Senger *et al*., 2003) hosted by the European Bioinformatics Institute (EBI), the services provided by XML Central of DDBJ (Miyazaki and Sugawara, 2000), the KEGG API (Kawashima *et al*., 2003, http://www.genome.ad.jp/kegg/soap/) and a range of analysis services offered by the PathPort project (Eckart and Sobral, 2003).

The ^{my}Grid e-Science project aims to provide high-level, service-based middleware to support data-intensive *in silico* bioinformatics experiments using distributed resources (Goble *et al*., 2003). These bioinformatics analyses depend on a workflow system which can converse with the interfaces of Web services and direct the flow of data between resources. There is considerable development in workflow tools in both the e-Science and e-business community, but it is a broad

[*]To whom correspondence should be addressed.

area with many competing proposals and no accepted standards (van der Aalst, 2003). At the time it was required, no tool satisfied all the ^{my}Grid requirements, which include open source availability and the provision of a graphical user interface for composing workflows. ^{my}Grid also aims to support the recording of results from *in silico* experiments and provenance information about how those results were obtained. In addition, there are existing bioinformatics applications with their own custom invocation mechanisms which life scientists will want to incorporate into workflows. Interaction with these applications may be stateful, with scripts used to define a series of activities to be performed. Examples of such tools are Talisman applications (Oinn, 2003) and Soaplab services (Senger *et al.*, 2003). The latter are Web services which have been created using Soaplab, a tool that can wrap command-line programs with a Web service interface based on a description of the analysis tool to be deployed. Tools which have been wrapped by Soaplab include the programs distributed within the European Molecular Biology Open Software Suite (EMBOSS) (Rice *et al.*, 2000) and algorithms such as BLAST (Altschul *et al.*, 1990). When scientists use Web services for *in silico* experiments, they need to be abstracted from the detail and complexity of Web service programming. Furthermore, scientists will want to integrate new types of services as and when they arise in the bioinformatics community. These requirements have led to the inception of the Taverna project, which has developed an open source workflow tool enabling scientists to orchestrate bioinformatics Web services and existing bioinformatics applications in workflows. The initial users of Taverna are bioinformaticians who can both develop and run workflows. Ongoing developments will address the wider life sciences community from those wanting to find their group's existing workflows, run them and browse the results to those who want guidance in the modification of existing workflows to match their specific needs.

SYSTEMS AND METHODS

The Taverna project provides a graphical workbench tool for both creating and running workflows that represent *in silico* bioinformatics experiments. In Taverna, a workflow is considered to be a graph of processors, each of which transforms a set of data inputs into a set of data outputs. These workflows are represented in the Scufl language.

Scufl language specification

Current languages were deemed unsuitable for composing scientific workflows since the existing standards are in flux, and high quality, free tools were not available to support standards (Oinn *et al.*, 2004). In addition, Web service standards do not have the levels of user abstraction necessary for most bioinformaticians and do not offer support for the specification of data, processes or resources at a semantic level. These requirements led to the specification of the Simple conceptual unified flow language (Scufl). It is a high-level, XML-based, conceptual language in which each processing step of the workflow represents one atomic task. A workflow in the Scufl language consists of three main entities:

(1) *Processors.* A processor is a transformation that accepts a set of input data and produces a set of output data. Processors have a name within the Scufl model and a set of both input and output ports. During the execution of a workflow, each processor has a current execution status which is one of initializing, waiting, running, complete, failed or aborted. The main processor types currently available are:

- *Arbitrary WSDL type*: This type of processor allows a single call on a Web service operation. The port names are derived from the message names in the Web Service Description Language (WSDL) file of the Web service for the identified operation.
- *Soaplab type*: This processor type calls a complete Soaplab invocation as one unit. The endpoint should be the full Uniform Resource Locator (URL) of the service endpoint so including, e.g. 'edit:seqret' at the end. The port names are extracted from the keys of the Soaplab input and output Map objects.
- *Talisman type*: This processor enables the invocation of a Talisman session as a task in the Scufl workflow. The Talisman processor requires a URL of a Talisman 'tscript' XML file describing the Talisman application definition, a set of input and output data and the operational behaviours of this service in terms of Talisman trigger invocations. The port names are taken directly from the tscript file.
- *Nested workflow type*: A processor of this type can invoke another child workflow. Currently, only child workflows in Scufl are supported. The input and output entities of the workflow are the input and output ports, respectively, of this processor.
- *String constant type*: This type of processor has a single output port on which it returns a constant string value. This processor is of particular use where another processor in the same workflow requires a default value which acts as a parameter. Another use of this processor is the replacement of an input entity in test workflows (Figs 1A and B).
- *Local processor type*: This processor can be used to add new local functions that are coded as classes to comply with a simple Java interface. The local functions currently available for use in workflows are shown in Figure 2D.

A workflow can also possess input and output data entities. A workflow input can be considered to be a source

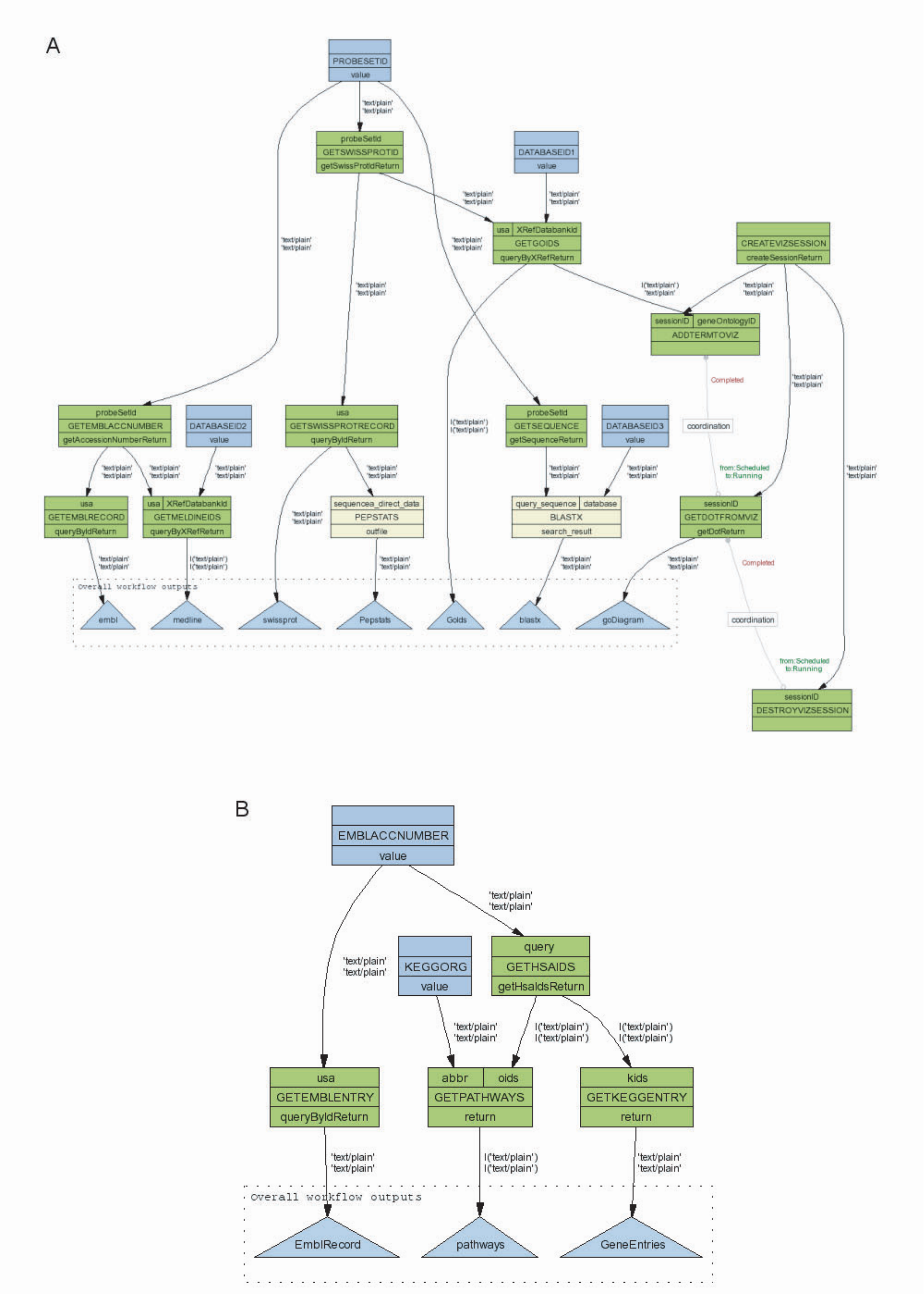
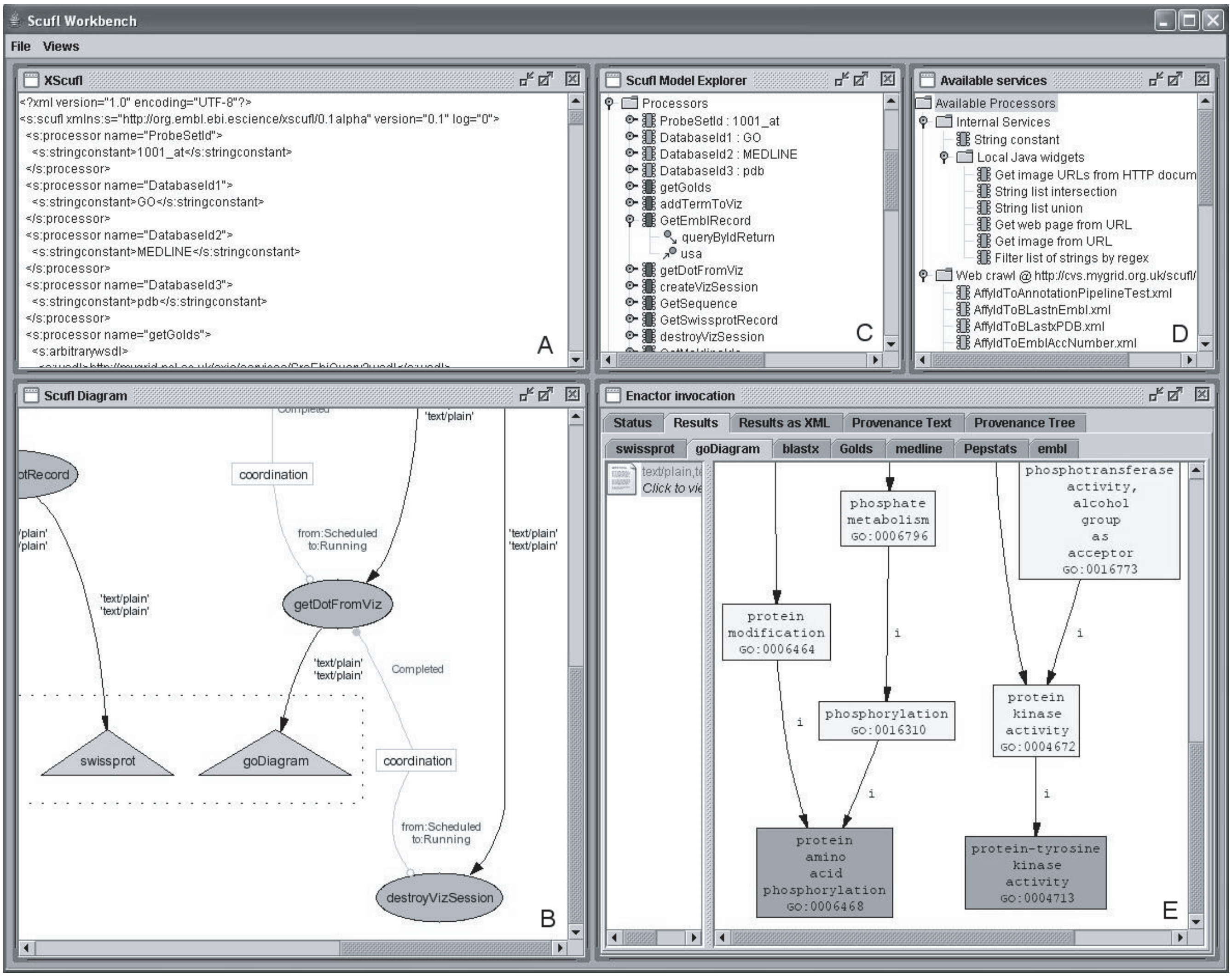


Fig. 1. Two example Scufl workflows. (**A**) The AffyidToGeneAnnotation workflow used for obtaining information for a given gene referenced by an Affymetrix probe set identifier. This workflow uses WSDL Web service operations (green boxes) and Soaplab services (yellow boxes). Default inputs/parameters in the form of string processors are shown as blue boxes and workflow output objects are displayed as blue triangles. (**B**) The EmblAccToKeggPathway workflow retrieving pathway information associated with a given gene identified by its EMBL accession number. This workflow integrates two disparate bioinformatics data resources, SRS and KEGG.



```
<?xml version="1.0" encoding="UTF-8"?>
<s:scufl xmlns:s="http://org.embl.ebi.escience/xscufl/0.1alpha" version="0.1" log="0">
 <s:processor name="ProbeSetId">
  <s:stringconstant>1001_at</s:stringconstant>
 </s:processor>
 <s:processor name="DatabaseId1">
  <s:stringconstant>GO</s:stringconstant>
 </s:processor>
 <s:processor name="DatabaseId2">
  <s:stringconstant>MEDLINE</s:stringconstant>
 </s:processor>
 <s:processor name="DatabaseId3">
  <s:stringconstant>pdb</s:stringconstant>
 </s:processor>
 <s:processor name="getGoIds">
  <s:arbitrarywsdl>
```

Fig. 2. The Scufl workbench. A number of views are provided by this application for the composition and enactment of Scufl workflows. A workflow can be viewed in its XML format (**A**), in graphical format (**B**) and as a tree structure using the Scufl Model Explorer which is also used to manipulate the workflow (**C**). Expanding a processor node reveals its inputs and outputs (**C**). The workbench includes a service palette for browsing local and remote services, and workflows (**D**). An enactor launch panel is used to display provenance information and the results generated by enacted workflows (**E**). The enactor launch panel (E) shows the result of the goDiagram workflow output (B) which is a subgraph of the GO corresponding to terms associated with a Swiss-Prot identifier.

processor which executes instantaneously and makes the input value available on its virtual output port. A workflow output can be considered as a sink processor which receives a value from its virtual input port but never actually executes. Both workflow sources and sinks can be annotated with metadata. Three types of metadata can be associated with workflow inputs and outputs: a MIME type, a semantic type based on the ^{my}Grid bioinformatics ontology (Wroe *et al*., 2003) and a free textual description.

(2) *Data links.* Data links mediate the flow of data between a data source and a data sink. The data source can be a processor output or a workflow input. The data sink can be a processor input port or a workflow output. Each data sink will receive the same value if there are multiple links from a data source.

(3) *Coordination constraints.* A coordination constraint links two processors and controls their execution. This level of control is required when there is a process where the stages must execute in a certain order and yet there is no direct data dependency between them. For example, coordination constraints can be used to allow one processor to go from scheduled to running if another processor has status completed. In most cases,

no concurrency constraints are required since data links will ensure that some processors stay in their waiting state until the data they require is available.

Scufl workbench

The Taverna tool contains an application called the Scufl workbench which enables bioinformaticians to write workflows without having to learn the Scufl language (Figs 2 and 3). This application acts as a container for a number of user interface components which provide read-only views and read-write controllers/views involved in the composition and enactment of Scufl workflows. The Scufl model explorer is a controller view that shows the state of the current model as a tree structure, and is also used for defining the flow of data between processors (Fig. 2C). At the top level are the different types of entities within a Scufl model; overall workflow inputs and outputs, processors, data links and coordination controls. Processor nodes may be expanded to reveal their inputs and outputs (Fig. 2C). The Scufl diagram view provides a graphical display of the current workflow (Figs 1A and B and 2B). This view uses the Dot tool from GraphViz to render the workflow as a PNG image (Gansner and North, 1999). There is a range of display options supported by the graphical view. Users can view processors with all ports displayed, no ports (Fig. 2B) or only those ports which are bound to data links (Fig. 1A and B). The Scufl workflow can also be viewed in its XML representation as XScufl (Fig. 2A). The XScufl and graphical displays of the workflow are read-only since only the Scufl model explorer is used to edit workflows.

The Scufl workbench contains a service browser that provides a palette of processors (Fig. 2D). Context menus in the service panel allow new processors to be added to the current Scufl workflow model. There are two methods of populating the palette with services. Processors in the current Scufl model can be ‘scavenged’ which involves extracting the set of processors contained within the model and adding them to the service palette. The exact relationship between a processor and an available service depends on the processor type. An arbitrary WSDL processor corresponds to a Web service operation and when a WSDL document is scavenged, the palette is populated with a service entry containing sub-entries for each Web service operation. With Soaplab processors, a set of Soaplab services from the same factory will be added to the palette. The palette can also be populated from the Web using scavengers for each processor type. Each scavenger requires a URL which, when pointed to a directory, will perform a naïve search to find files that it can process. Pointing to a Web scavenger at http://taverna.sourceforge.net/webservices/index.html will provide access to the XEMBL, PathPort, DDBJ and KEGG services. The Soaplab scavenger is pointed at http://industry.ebi.ac.uk/soap/soaplab/ by default which will provide a list of EMBOSS and BLAST tools for use in workflows. The service browser can also be populated with workflows when directed with a URL to a directory of Scufl workflows (Fig. 2D).

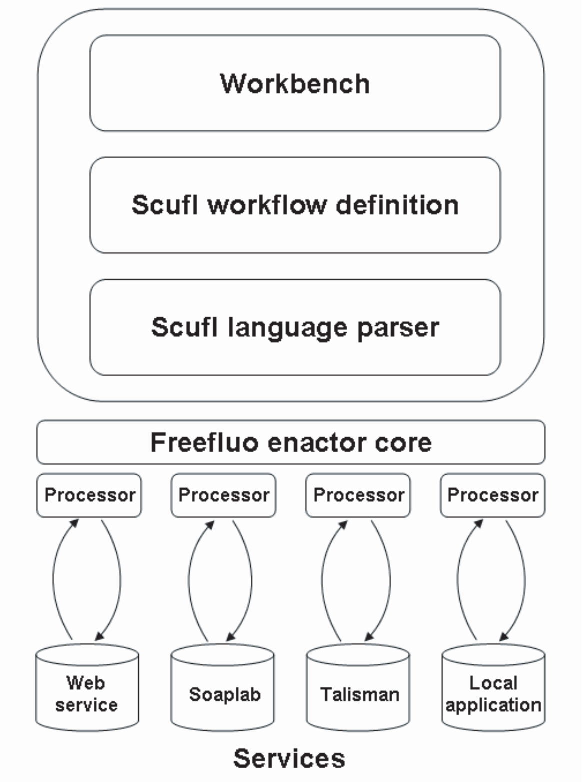


Fig. 3. An architectural view of the Taverna workflow system. The Taverna workbench is used to compose Scufl workflows which are parsed into a form that can be enacted by the Freefluo enactor. Different types of services can be invoked by Freefluo; currently the enactor can invoke WSDL Web services, Soaplab services, Talisman and local applications.

Workflows can be executed in the Scufl workbench using the enactor launch panel (Fig. 2E). This panel allows inputs to be specified for the workflow and launches a local instance of the Freefluo enactment engine (Fig. 3). Freefluo is a Java workflow orchestration tool for Web services that supports a subset of the Web Services Flow Language as well as Scufl (Addis *et al*., 2003). This flexibility of Freefluo is provided at its core by a reusable orchestration framework that is not tied to any workflow language or execution architecture. The enactor core supports an object model of a workflow in the form of a directed graph where each node has a state machine that defines its lifecycle. Scheduling and state transitions are driven by message passing between nodes as the workflow progresses. The core of the enactor is decoupled from both the textual form of a workflow specification and the details of service invocation and data model allowing it to orchestrate workflows in a generic way (Fig. 3). This flexibility is exploited when Taverna is extended to cope with a new processor type. The core of the enactor is unchanged but Freefluo is extended with a parser for the new XScufl processor and a plug-in for the required service invocation.

RESULTS

The ease by which bioinformatics Web services can be integrated with one another using Taverna is demonstrated by two exemplar bioinformatics workflows. The first workflow is a ^{my}Grid annotation pipeline process for obtaining information about a gene. In the second workflow, third party bioinformatics Web services are used to retrieve metabolic or signalling pathway maps and associated nucleotide sequence information for a given gene product. A tutorial and videos demonstrating how workflows can be composed and enacted using the Scufl workbench is available at http://taverna.sourceforge.net/.

Gene annotation pipeline

The querying of information repositories and the application of analysis programs are frequently combined within an annotation pipeline for investigating the biology of genes. An annotation pipeline using disparate bioinformatics resources is easily composed by Taverna using the Scufl workbench. An example of such a workflow called 'AffyidToGeneAnnotation' is shown in Figure 1A. This was one of a number of workflows written for biologists working on a microarray-based approach to the genetic analysis of Graves' disease to evaluate whether Taverna was capable of performing their analyses (Addis *et al*., 2003; Stevens *et al*., 2003).

The workflow obtains information about genes which may be involved in Graves' disease that have been identified by using Affymetrix U95 microarray chips. Three distinct resources were used within this annotation pipeline: an instance of the Sequence Retrieval System (SRS) at the EBI, a mapping database called Affymapper and programs deployed as Soaplab services. The key functionality in this workflow was the ability to map a candidate gene as referenced by its Affymetrix probe set identifier to entries in various biological databases. The mappings from an Affymetrix probe set identifier to its EMBL accession number and Swiss-Prot identifier were obtained from the Affymapper database service. In addition, the SRS service was used to link from an EMBL accession number to Medline identifiers for obtaining bibliographic citations of the gene in scientific literature. The SRS service was also used to obtain the records associated with the EMBL accession number, Swiss-Prot and Medline identifiers to obtain sequence and published information about the gene and its gene product. Identifiers to terms in the Gene Ontology (GO) (Ashburner *et al*., 2000) associated with the Swiss-Prot identifier were also retrieved in this workflow. The GoViz service then extracted the sub-graph of the GO associated with the GO identifiers to provide a more general view to the function of the gene product (Fig. 2E).

Two bioinformatics tools performed analyses on the nucleotide sequence associated with EMBL accession number and the amino acid sequence from the Swiss-Prot record. A tBLAST search with the nucleotide sequence against the Protein Data Bank found proteins with known structures that may be related to the translated nucleotide sequence (Fig. 4). The Pepstat program in EMBOSS generated protein information such as the molecular weight and isoelectric point, based on the amino acid sequence contained within the Swiss-Prot record which was mapped from the Affymetrix probe set identifier.

Pathway map retrieval

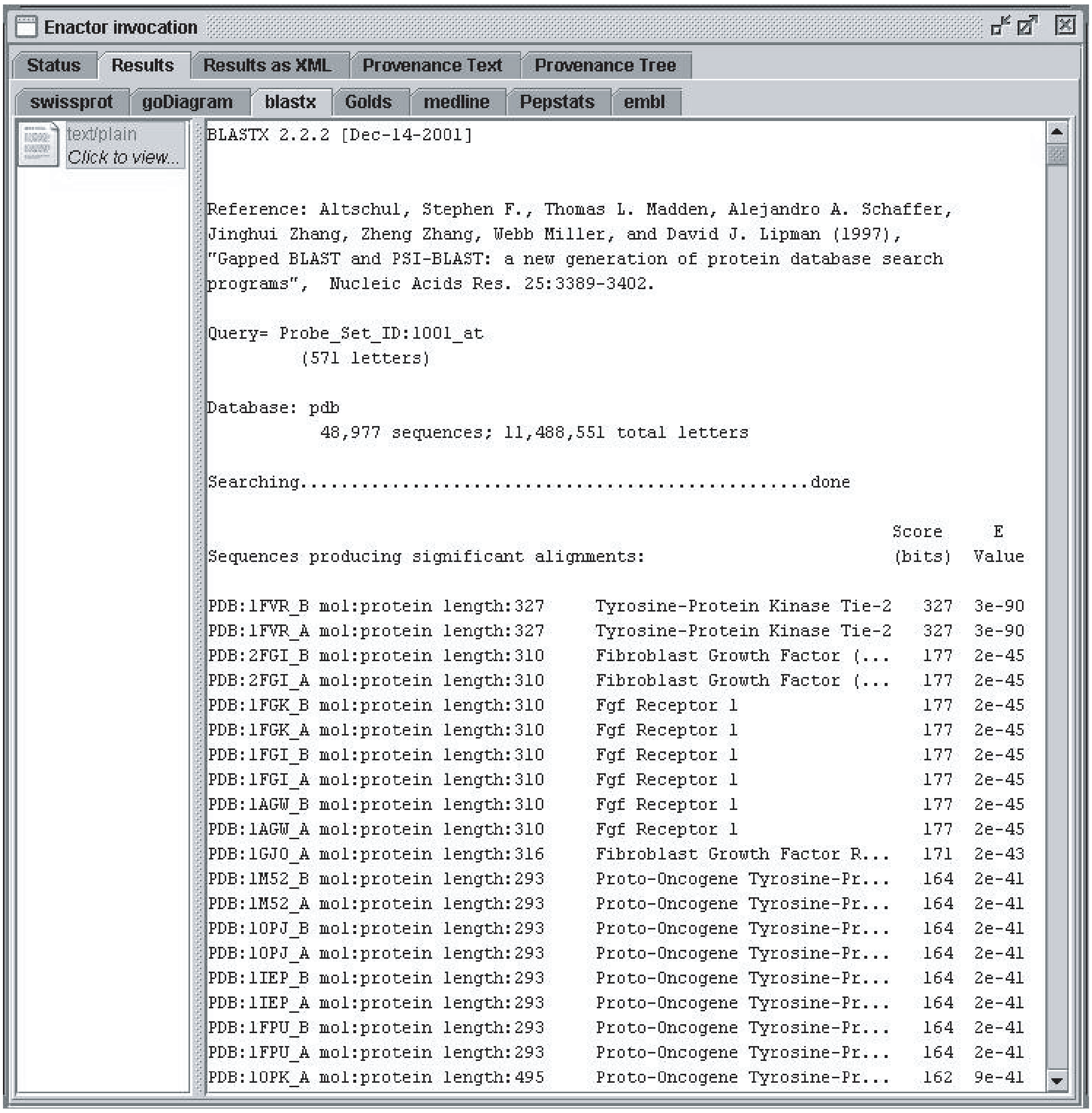
Obtaining information about the metabolic or signalling pathways associated with a gene product is an analysis which is commonly performed in bioinformatics (Yanai *et al*., 2002; Lee and Sonnhammer, 2003; Hannenhalli and Levy, 2003). One approach to this analysis is to integrate data available from nucleotide sequence databases with a database containing pathway information, as shown by the workflow called 'EmblAccToKeggPathway' in Figure 1B.

This workflow starts by mapping the EMBL accession number gene handle to KEGG gene identifiers using a custom Keggmapper service. The workflow retrieves the EMBL record associated with the EMBL accession number using the SRS service while entries associated with the KEGG gene identifiers are obtained using the KEGG Web service. The KEGG gene identifiers are then used to search the pathway diagrams in the KEGG Pathway database which contains the gene. Objects in the pathway diagrams that correspond to the genes on the map are highlighted and the URL for each pathway diagram is returned by the KEGG Web service.

On completion of the workflow, the Scufl workbench displays the pathway diagram specified by each returned URL (Fig. 5). This functionality is based on the ability to associate syntactic and semantic metadata to input and output data objects in Scufl workflows. For example, the x-taverna-url MIME type was associated with each pathway URL returned by the GETPATHWAYS task (Fig. 1B). Any output data object associated with this x-taverna-url MIME type results in the document located at the URL being retrieved and displayed within the workbench (Fig. 5). This ability to associate appropriate viewers with workflow outputs, based on their types, is one of the benefits of the Taverna workbench.

Tracking of data provenance

The tracking of data provenance is also required during the execution of *in silico* experiments. Provenance is information that identifies the source and processing of data by recording the metadata and intermediate results associated with the workflow. This type of information can be a useful audit trail when investigating how results, in particular erroneous or unexpected ones, have been produced by workflow processes.



```
BLASTX 2.2.2 [Dec-14-2001]


Reference: Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schaffer,
Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997),
"Gapped BLAST and PSI-BLAST: a new generation of protein database search
programs",  Nucleic Acids Res. 25:3389-3402.

Query= Probe_Set_ID:1001_at
         (571 letters)

Database: pdb
           48,977 sequences; 11,488,551 total letters

Searching..................................................done

                                                                  Score     E
Sequences producing significant alignments:                       (bits)  Value

PDB:1FVR_B mol:protein length:327     Tyrosine-Protein Kinase Tie-2   327  3e-90
PDB:1FVR_A mol:protein length:327     Tyrosine-Protein Kinase Tie-2   327  3e-90
PDB:2FGI_B mol:protein length:310     Fibroblast Growth Factor (...   177  2e-45
PDB:2FGI_A mol:protein length:310     Fibroblast Growth Factor (...   177  2e-45
PDB:1FGK_B mol:protein length:310     Fgf Receptor 1                  177  2e-45
PDB:1FGK_A mol:protein length:310     Fgf Receptor 1                  177  2e-45
PDB:1FGI_B mol:protein length:310     Fgf Receptor 1                  177  2e-45
PDB:1FGI_A mol:protein length:310     Fgf Receptor 1                  177  2e-45
PDB:1AGW_B mol:protein length:310     Fgf Receptor 1                  177  2e-45
PDB:1AGW_A mol:protein length:310     Fgf Receptor 1                  177  2e-45
PDB:1GJO_A mol:protein length:316     Fibroblast Growth Factor R...   171  2e-43
PDB:1M52_B mol:protein length:293     Proto-Oncogene Tyrosine-Pr...   164  2e-41
PDB:1M52_A mol:protein length:293     Proto-Oncogene Tyrosine-Pr...   164  2e-41
PDB:1OPJ_B mol:protein length:293     Proto-Oncogene Tyrosine-Pr...   164  2e-41
PDB:1OPJ_A mol:protein length:293     Proto-Oncogene Tyrosine-Pr...   164  2e-41
PDB:1IEP_B mol:protein length:293     Proto-Oncogene Tyrosine-Pr...   164  2e-41
PDB:1IEP_A mol:protein length:293     Proto-Oncogene Tyrosine-Pr...   164  2e-41
PDB:1FPU_B mol:protein length:293     Proto-Oncogene Tyrosine-Pr...   164  2e-41
PDB:1FPU_A mol:protein length:293     Proto-Oncogene Tyrosine-Pr...   164  2e-41
PDB:1OPK_A mol:protein length:495     Proto-Oncogene Tyrosine-Pr...   162  9e-41
```

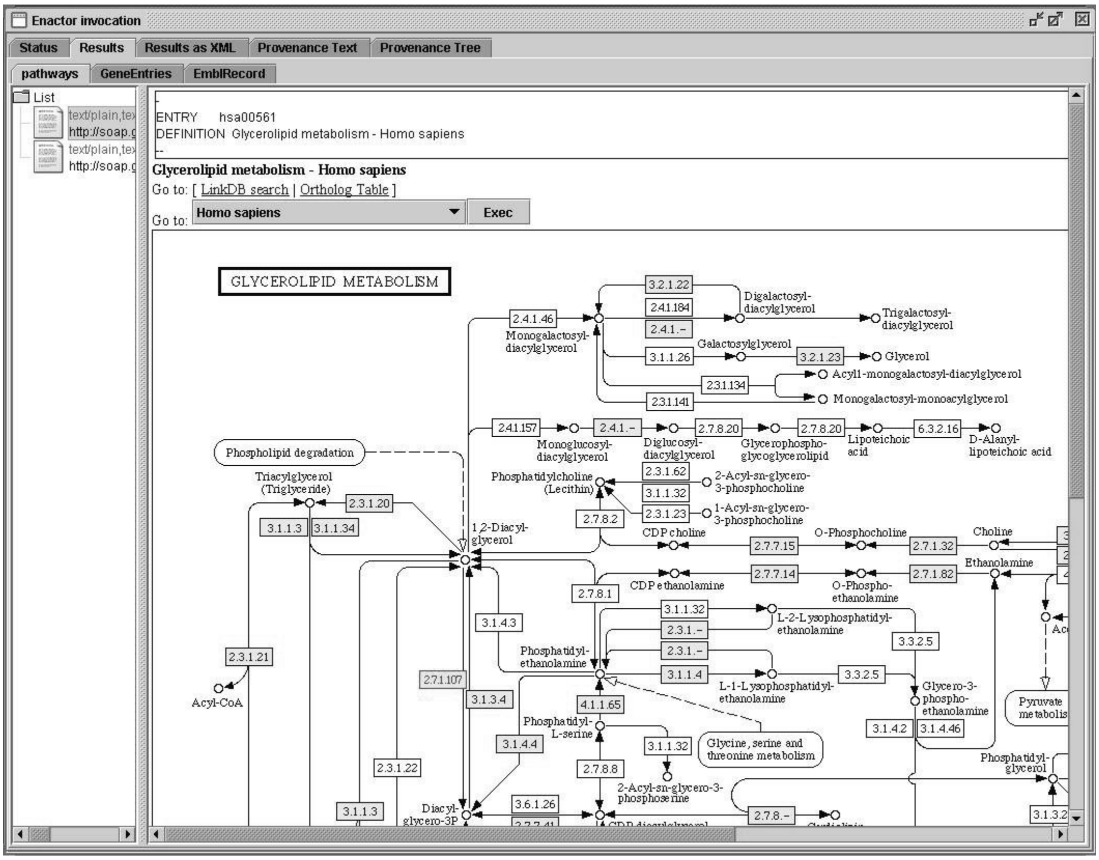
Fig. 4. The tBLAST report generated by the gene annotation pipeline workflow shown in Figure 1A.

During the enactment of workflows in Taverna, the provenance information recorded comprises of technical metadata showing how each task has been performed (Fig. 6). The type of processor, status, start and end time and a description of the service operation used are recorded and may be browsed (Fig. 6).

In the future, we aim to unify the provenance production and collection from a variety of disparate systems, including workflow, database query and manual annotation tools. By defining a standard set of relations and classes to be used in Resource Description Framework statements generated as a side effect of the primary function of these systems, statements can be merged from the different sources that data may have passed through en route to its final form. This will allow questions about the origin of any given item of data in a data store to be answered even though such data may be derived from a combination of otherwise distinct operations.

DISCUSSION

With the increasing number of bioinformatics databases and computational programs being made available as Web services, a workflow system is an essential tool for e-Scientists if they are to take full advantage of such resources. A tool for the creation and enactment of scientific workflows has been developed by the Taverna project. This project has developed the Scufl language for the scripting of workflows and a parser for the Freefluo enactor to enable it to execute Scufl workflow definitions. The Scufl workbench provides a



ENTRY hsa00561
DEFINITION Glycerolipid metabolism - Homo sapiens

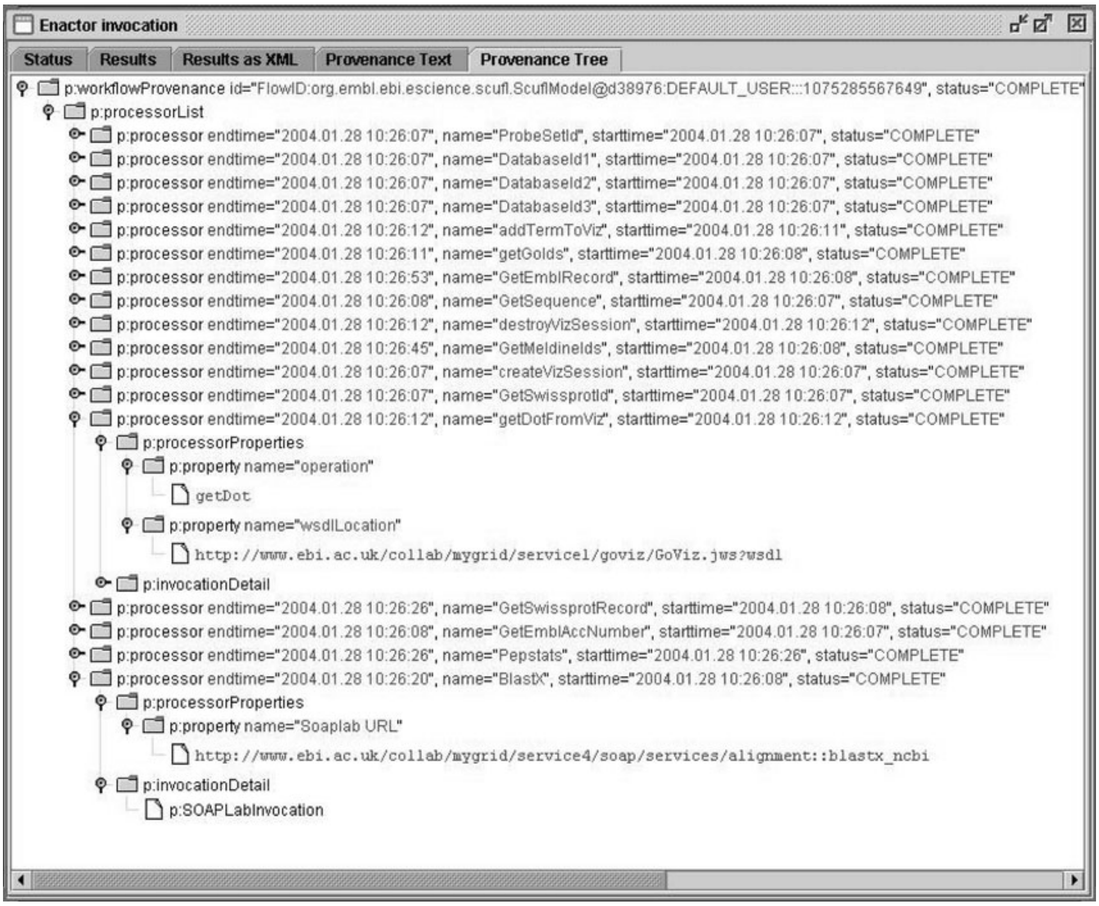
Glycerolipid metabolism - Homo sapiens
Go to: [LinkDB search | Ortholog Table]
Go to: Homo sapiens ▼ Exec

Fig. 5. One of the pathway diagrams retrieved from the KEGG Pathway database as a result of the workflow shown in Figure 1B.

graphical user interface for the composition of Scufl workflow definitions. This allows Taverna users to concentrate on capturing their *in silico* research experiments as workflows, rather than having to spend time and effort learning the syntax of the Scufl workflow language before they can compose workflows.

The two workflows presented in Figure 1A and B shows how Taverna orchestrates data resources and computational services to undertake bioinformatics analyses. The gene annotation pipeline workflow demonstrates the ease by which data resources can be integrated with analysis programs (Fig. 1A). The interoperability of Taverna is highlighted in the pathway retrieval workflow which consumed the KEGG Web service, a data resource located at Kyoto University, Japan that was developed independently of the Taverna or ^{my}Grid projects (Fig. 1B). Taverna is also able to consume other third party services such as XEMBL (Wang *et al*., 2002), and those provided by DDBJ and the PathPort project (Kawashima *et al*., 2003; Eckart and Sobral, 2003). Scufl workflows demonstrating how these services can be used are available at http://taverna.sourceforge.net/webservices/workflows/. The pathway retrieval workflow also demonstrates the benefit of labelling data objects with metadata that describe their syntactic and/or semantic type. The additional information provided by the syntactic metadata provides applications with a handle to select an appropriate method for viewing data. This was shown in the pathway retrieval workflow in which the URL data product was associated with the x-taverna-url MIME type which subsequently led the Scufl workbench to display the pathway maps located at the URL (Fig. 5).

The Taverna workflow system provides a tool, which can integrate resources that are shared as Web services among the bioinformatics community. In a similar fashion, the Scufl workflows created using Taverna are resources in their own right that can be shared among scientists. This is in contrast to Perl scripts which can be used to compose workflows but which are not often shared. This is due, at least in part, to the difficulties in knowing the exact environment requirements, e.g. resources invoked via system calls, which inhibit the portability and maintenance of Perl scripts. This issue does not occur with Scufl workflows since a common invocation mechanism in the form of Web services is used and the service interface is usually readily available on the Web in the form of a WSDL file. However, it is not necessary for resources to be exposed with a Web service interface for them



```
p:workflowProvenance id="FlowID:org.embl.ebi.escience.scufl.ScuflModel@d38976:DEFAULT_USER:::1075285567649", status="COMPLETE"
  p:processorList
    p:processor endtime="2004.01.28 10:26:07", name="ProbeSetId", starttime="2004.01.28 10:26:07", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:07", name="DatabaseId1", starttime="2004.01.28 10:26:07", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:07", name="DatabaseId2", starttime="2004.01.28 10:26:07", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:07", name="DatabaseId3", starttime="2004.01.28 10:26:07", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:12", name="addTermToViz", starttime="2004.01.28 10:26:11", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:11", name="getGoIds", starttime="2004.01.28 10:26:08", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:53", name="GetEmblRecord", starttime="2004.01.28 10:26:08", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:08", name="GetSequence", starttime="2004.01.28 10:26:07", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:12", name="destroyVizSession", starttime="2004.01.28 10:26:12", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:45", name="GetMeldineIds", starttime="2004.01.28 10:26:08", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:07", name="createVizSession", starttime="2004.01.28 10:26:07", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:07", name="GetSwissprotId", starttime="2004.01.28 10:26:07", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:12", name="getDotFromViz", starttime="2004.01.28 10:26:12", status="COMPLETE"
      p:processorProperties
        p:property name="operation"
          getDot
        p:property name="wsdlLocation"
          http://www.ebi.ac.uk/collab/mygrid/service1/goviz/GoViz.jws?wsdl
      p:invocationDetail
    p:processor endtime="2004.01.28 10:26:26", name="GetSwissprotRecord", starttime="2004.01.28 10:26:08", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:08", name="GetEmblAccNumber", starttime="2004.01.28 10:26:07", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:26", name="Pepstats", starttime="2004.01.28 10:26:26", status="COMPLETE"
    p:processor endtime="2004.01.28 10:26:20", name="BlastX", starttime="2004.01.28 10:26:08", status="COMPLETE"
      p:processorProperties
        p:property name="Soaplab URL"
          http://www.ebi.ac.uk/collab/mygrid/service4/soap/services/alignment::blastx_ncbi
      p:invocationDetail
        p:SOAPLabInvocation
```

Fig. 6. The provenance log recorded after the enactment of the gene annotation pipeline workflow shown in Figure 1A. Information is displayed about the task performed by each processor in the Scufl workflow.

to be used within Scufl workflows in Taverna. Resources with other invocation mechanisms may be used in Taverna by, first, creating a plug-in for the Freefluo enactor to access the resource and, second, implementing a corresponding Scufl processor type. Legacy bioinformatics tools with other invocation mechanisms have been incorporated into Taverna as Talisman (Oinn, 2003) and Soaplab (Senger *et al*., 2003) Scufl processor types (Fig. 1A).

New features will continuously be added to the Taverna workflow tool in response to user feedback. The Scufl workbench is an area targeted for improvement to a level such that workflows can be composed by laboratory biologists. Services are currently located within the Scufl workbench by instigating a Web crawl to locate XScufl, WSDL and Talisman tscript files at a given URL. Work is in progress to link the Scufl workbench with service registries such as those offered by the ^{my}Grid (Lord *et al*., 2003) and BioMoby (Wilkinson and Links, 2002) projects. In addition, both Scufl and Freefluo aim to support the ability to define an expression that is evaluated before a task is executed. If the expression is not true, the enactor can either skip the step entirely or wait until the condition is true before executing the step. The decision whether a condition is true or false may also require human intervention. The act of user intervention in workflows is currently being investigated in the Taverna project.

In the interests of creating a ‘bioinformatics nation’ whereby biological data can be seamlessly accessed by scientists, Stein (2002) suggested that data and tool providers equip their resources with a Web service interface. The Web services framework is a W3C standard which is being adopted by the bioinformatics community and this is shown by the increasing number of bioinformatics Web services being made available such as those from the cancer Bioinformatics Infrastructure Objects project (http://cabio.nci.nih.gov/) (Stein, 2002). To this end, there are open source tools available for the generation of Web service wrappers such as Apache Axis (http://ws.apache.org/axis/) which was used to Web service-enable the Affymapper and Keggmapper databases for the gene annotation and pathway retrieval workflows (Fig. 1A and B). It is hoped that our Taverna workflow tool will encourage bioinformaticians to use the resources currently available as Web services, and also to provide Web service access to their databases and programs for sharing within the bioinformatics community. This will enable the sharing of workflows which further illustrates the value of the co-ordinated use of a growing repertoire of bioinformatics Web services.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the ^{my}Grid team: Nedim Alpdemir, Rich Cawley, Tracy Craddock, Neil Davis, Alvaro Fernandes, Robert Gaizaukaus, Carole Goble, Chris Greenhalgh, Yikun Guo, Keith Hayward, Anath Krishna, Phil Lord, Simon Miles, Luc Moreau, Arijit Mukherjee, Juri Papay, Savas Parastatidis, Norman Paton, Milena Radenkovic, Peter Rice, Nick Sharman, Robert Stevens, Victor Tan, Paul Watson and Chris Wroe; and our industrial partners: IBM, Sun Microsystems, GlaxoSmithKline, AstraZeneca, Merck KgaA, genetic Xchange, Epistemics Ltd, and Network Inference. This work is supported by the UK e-Science programme EPSRC GR/R67743.

REFERENCES

Addis,M., Ferris,J., Greenwood,M., Li,P., Marvin,D., Oinn,T. and Wipat,A. (2003) Experiences with e-Science workflow specification and enactment in bioinformatics. In: *Proceedings of UK e-Science All Hands Meeting 2003*, pp. 459–466.

Altschul,S.F., Gish,W., Miller,W., Myers,E.W. and Lipman,D.J. (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Ashburner,M., Ball,C.A., Blake,J.A., Botstein,D., Butler,H., Cherry,J.M., Davis,A.P., Dolinski,K., Dwight,S.S., Eppig,J.T. *et al*. (2000) Gene Ontology: tool for the unification of biology. *Nat. Genet.*, **25**, 25–29.

Booth,D., Haas,H., McCabe,F., Newcomer,E., Champion,M., Ferris,C. and Orchard,D. (2003) Web Services Architecture.

Eckart,J.D. and Sobral,B.W. (2003) A life scientist's gateway to distributed data management and computing: the PathPort/ToolBus framework. *OMICS*, **7**, 79–88.

Gansner,E.R and North,S.C. (1999) An open graph visualization system and its applications to software engineering. *Softw. Pract. Exper.*, **S1**, 1–5.

Goble,C.A., Pettifer,S., Stevens,R. and Greenhalgh,C. (2003) Knowledge integration: *in silico* experiments in bioinformatics. In Foster,I. and Kesselman,C. (eds), *The Grid 2: Blueprint for a New Computing Infrastructure*, 2nd edn, Morgan Kaufmann.

Hannenhalli,S. and Levy,S. (2003) Transcriptional regulation of protein complexes and biological pathways. *Mamm. Genome.* **14**, 611–619.

Kawashima,S., Katayama,T., Sato,Y. and Kanehisa,M. (2003) KEGG API.

Lee,J.M. and Sonnhammer,E.L. (2003) Genomic gene clustering analysis of pathways in eukaryotes. *Genome Res.*, **13**, 875–882.

Lord,P., Wroe,C., Stevens,R., Goble,C., Miles,S., Moreau,L., Decker,K., Payne,T. and Papay,J. (2003) Semantic and personalised service discovery. In Cheung,W.K. and Ye,Y. (eds), *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*, Halifax, Canada, pp. 100–107.

Miyazaki,S. and Sugawara,H. (2000) Development of DDBJ-XML and its application to a database of cDNA. *Genome Informatics*. Universal Academy Press, Inc., Tokyo, pp. 380–381.

Oinn,T. (2003) Talisman—rapid application development for the grid. *Bioinformatics*, **19**(Suppl. 1), i212–i214.

Oinn,T., Addis,M., Ferris,J., Marvin,D., Greenwood,M., Wipat,A., Li,P. and Carver,T. (2004) Delivering Web service coordination capability to users. *Accepted as Short Note and Poster for WWW2004*.

Rice,P., Longden,I. and Bleasby,A. (2000) EMBOSS: the European Molecular Biology Open Software Suite. *Trends Genet.*, **16**, 276–277.

Senger,M. (2002) Bibliographic query service.

Senger,M., Rice,P. and Oinn,T. (2003) SoapLab—a unified Sesame door to analysis tools. *Proceedings of the UK e-Science All Hands Meeting 2003*.

Stein,L. (2002) Creating a bioinformatics nation. *Nature*, **417**, 119–120.

Stevens,R., Glover,K., Greenhalgh,C., Jennings,C., Pearce,S., Li,P., Radenkovic,M. and Wipat,A. (2003) Performing *in silico* experiments on the Grid: a users perspective. *Proceedings of UK e-Science All Hands Meeting 2003*, pp. 43–50.

Wang,L., Riethoven,J.J. and Robinson,A. (2002) XEMBL: distributing EMBL data in XML format. *Bioinformatics*, **18**, 1147–1148.

Wroe,C., Stevens,R., Goble,C., Roberts,A. and Greenwood,M. (2003) A suite of DAML+OIL ontologies to describe bioinformatics web services and data. In *Int. J. Coop. Inform. Syst. (special issue on Bioinformatics)*, **12**, 197–224.

Wilkinson,M.D. and Links,M. (2002) BioMOBY: an open source biological web services proposal. *Brief. Bioinform.*, **3**, 331–341.

van der Aalst,W. (2003) Don't go with the flow: web services composition standards exposed. *IEEE Intell. Syst.*, 72–76.

Yanai,I., Mellor,J.C. and DeLisi,C. (2002) Identifying functional links between genes using conserved chromosomal proximity. *Trends Genet.*, **18**, 76–79.