A WorkFlow Management System for Bioinformatics Grid

Giovanni Aloisio, Massimo Cafaro, Sandro Fiore, Maria Mirto

Center for Advanced Computational Technologies/ISUFI
and
National Nanotechnology Lab/INFM&CNR, Italy
{giovanni.aloisio, massimo.cafaro, sandro.fiore, maria.mirto}@unile.it

Abstract. In this paper we describe a Workflow Management System, named ProGenGrid (Proteomics and Genomics Grid, developed at the University of Lecce) which aims at providing a tool where e-scientists can simulate biological experiments through the composition of existing analysis and visualization tools, wrapped as Web Services. Since bioinformatics applications are compute- and data-intensive, needing clusters or many workstations to reduce the execution time, we exploit a Grid infrastructure for interconnecting wide-spread tools and hardware resources. As an example, we are considering some algorithms and tools needed for alignment of sequences, providing them as services, through easy to use Web interfaces and Web services built using the open source gSOAP Toolkit. As Grid middleware, we are using the Globus Toolkit 4.1, exploiting some protocols such as GSI and GridFTP.

Keywords: Bioinformatics, Web Services, Computational Grid, Grid Portal, Globus Toolkit.

1. Introduction

To date, many bioinformatics tools are available to query and analyse biological data. These software are heterogeneous, support several kinds of data and the analysis of the produced results is often very difficult. Sometimes, it is needed to edit by hand the output, and adjust some values in order to submit it to another software. It is desirable, from a user point of view, the integration of various tools, allowing, for instance, to (i) extract from a data bank a given sequence, (ii) predict its three-dimensional structure, (iii) visualize it and in case adjust it (adding or deleting aminoacids, lateral chains or loops), (iv) verify if it belongs to a given family, and (v) access a set of papers describing its features.
Moreover, some of these operations are computationally expensive and need an adequate infrastructure to produce the results in a reasonable time.
The development of a system for the integration of biological tools and data is equivalent at building a "virtual laboratory" where biologists can simulate "in silico" experiments, using computer science tools, without knowing the machines where they are actually running.
At last, for an efficient distributed and ubiquitous system, it seems appropriate to develop the system using the framework offered by Computational Grids [1], through an interface that allows hiding the complexity of a Grid, e.g., the real location of applications and data resources and the submission of jobs.
Our goal has been the design and implementation of a system that allows, through our editor, to use a well known graphical formalism such as Unified Modelling Language (UML) [2] for simulating a biological experiment. Using an easy-to-use GUI, the user can select some tools, compose them, synchronising some operations (using join and fork symbols) and monitoring the execution of these applications, visualizing also the intermediate results. These tools are modelled with an ontology, have a Web Service (WS) [3] interface and are discovered at run time through an opportune lookup WS. The execution of the experiment is by mean of a workflow scheduler (also a WS component) that schedules the job flow on a set of resources dynamically chosen in a Computational Grid.

The editor allows saving the diagram that represents the experiment, translating each UML symbol in XML notation. So it is possible to re-run the experiment for checking the results or to update some steps. When the application must be run, the workflow scheduler takes activities in the XML file and run them taking into account the state and availability of Grid resources and involved bioinformatics tools. Moreover, through the system it is possible to monitor the job flow.
The paper is organized as follows. In Section 2 we briefly summarize related works on bioinformatics workflow, underlying features and differences with respect to ProGenGrid, then in Section 3 we describe the structure of the ProGenGrid Workflow and its main components in detail. In Section 4 some implementation details are reported examining a sample application, the alignment of sequences. Finally we conclude the paper in Section 5.

2. Related Works

Workflows are increasingly important in the development of new computing applications for life sciences. Moreover, it is crucial to provide easy-to-use tools that do not burden bionformaticians, but benefit from state-of-the-art technologies. Grid-enabled workflow management tools are crucial for successful building and deployment of bioinformatics workflows. Several workflows that we analyze in this section are: Taverna, Pegasus, Proteus and Pegasys.
Taverna [4] is a workflow orchestration tool for web services and it is part of the myGrid project. It can handle WSDL based web service invocation, interrogate a standard UDDI registry, taking into account user's preferences to obtain actual service instances to invoke. It supports an XML workflow definition language that is SCUFL (Simple Conceptual Unified Flow Language) Workflow Language for invoking remote and local services and pass results between them. In practice, one conceptual operation in the design of a workflow translates to one processor in a SCUFL definition. The disadvantage of Taverna is that the user can not define complex operations such as loops (for a given number of iterations) or conditions.
Pegasus [5] is a workflow management system designed to map abstract workflows onto Grid resources, through Globus RLS, Globus MDS and Globus MCS to determine the available resources and data. Using Pegasus requires producing an abstract workflow in DAX format. The disadvantage of Pegasus is that it lacks the interface with some de-facto bioinformatics tools and it does not support a checking tool for workflow.
Another grid-based workflow management project is Proteus [6]. Proteus uses ontologies to classify software and data tools, and it also employs workflows to provide an easy problem-solving environment for bioinformaticians. It uses the PEDRO system for modelling experiments. The disadvantage of Proteus lies in its specific applicability to proteomics domain only. Also, it does not utilize web services.
Finally, another interesting work is Pegasys [7], a workflow management system for bioinformatics. The system includes numerous tools for pair–wise and multiple sequence alignment, ab initio gene prediction, RNA gene detection, and masking repetitive sequences in genomic DNA as well as filters for database formatting or processing of raw output from various analyses. Pegasys allows building directed acyclic graphs without the possibility to insert loops or other conditions such as fork and join.
What differentiates our work from Taverna is the different language for designing and mapping abstract into concrete workflows, the different language of implementation of Web Service (Java – Axis vs C - gSOAP), the ontology of the software that in the case of ProGenGrid is integrated into the editor for validating the experiment during the design phase and finally the use of GridFTP and DIME for transferring large amount of data. The difference between Pegasus and ProgenGrid is that ProGenGrid workflow has an editor for composing and validating the experiment in the modelling phase; it uses the Globus-based GRB (Grid Resource Broker) libraries [8] for job execution on the grid and the iGrid information service [9], as web service system for resource and web services discovery. The main difference w.r.t. Proteus lies in our web services extensions to existing bioinformatics tool so that these can, without user intervention, seamlessly connect to grids. This

approach enables legacy applications to use grid-processing power without any significant change. Finally the main difference between Pegasys and ProGenGrid is both the graphical interface that allows operations such as binding more tools in an acyclic graph and the meta scheduler that in the former system allows submitting jobs to a PBS node whereas in the latter uses its own scheduler for invocating the web services interface and GRB libraries for submitting jobs to the Globus Gatekeeper. Pegasys does not exploit a Grid framework and ontologies. Moreover whereas in the Pegasys system is not possible to carry out job migration and Pegasys must be installed on a cluster machine, one of the goals of ProGenGrid is to use a flexible scheduler that supports the dynamic assignment of resources belonging to a Grid environment.

3. ProGenGrid Workflow

The issue of execution in a given order of complex computational tasks in a Grid environment has been discussed by the "Grid Computing Environment Working Group" (GCE-WG), in the Global Grid Forum (GGF) [10], which has proposed an XML-based interface, for the definition of a workflow to be submitted in a grid environment. According to the definition given by "Workflow Management Coalition", a workflow is: "*The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules*" [11].
A "business process" is the set of procedures required for obtaining a given result, therefore workflow is the automation of a set of operations that allows obtaining a given result, with the exchange of information among involved entities and with respect to defined procedural rules.
The operations involved by a workflow are called *activities*; an activity is a part of the entire work and it represents a logical step in the process.
To run a workflow into a Computational Grid it is necessary to define a common language to be interpreted and manipulated automatically from a management system, which should allow defining a set of activities, their relation, involved entities (i.e. applications, data resources, etc.) and some criteria for determining the start and end of the processes. According to GCE-WG, the language suitable to match those requirements is XML for several reasons and features: well defined syntax, availability of various open source tools (in Java, C, Perl, etc.) for the processing of XML documents, independence from platforms and vendors, support for a large set of international characters for data exchange and finally, a "human readable" format.
To define an experiment using a well known notation it is necessary to define a graphical formalism for modelling the experiment and describing it in detail. We chose UML because it is an OMG [12] standard which provides a visual modelling notation valuable for designing and understanding complex systems. UML has several general advantages: it is the most widely known Object-Oriented (OO) modelling notation, it has a graphical notation which is readily understood, and a rich set of semantics for capturing key features of OO systems.
Finally the automation of complex operations in a Grid environment is useful because the users that want to use available resources must not necessarily manually discover and schedule the operations. The goal is to automate, as much as possible, the generation process related to the set of operations. According to this goal we have introduced the concepts of *abstract workflow*, for specifying, as logical links, the applications to run and the input data and *concrete workflow*, for describing the physical links of the applications, data files and specific resources.
The abstract workflow specifies the order in which the operations composing a workflow should be run while the concrete workflow allows selecting specific files and resources and eventually to introduce some additional operations such as conversion of data format or file transfer.
The first step for building a system that allows scheduling automatically the operations on a Grid is the development of an interface that allows the user to select and eventually configure intuitively the applications. The generation of the concrete workflow will be addressed by the system, that will be able to map logical activities, specified by the user, to operations to be run in a Computational Grid.

Thus, we designed an editor based on a workflow description language (for generating the abstract and concrete workflow) and a workflow scheduler for the execution and control of the flow of tasks.

3.1 Architecture

The architecture of the ProGenGrid workflow is depicted in Figure 1. A user describes a job flow using the ProGenGrid editor. The Metadata Ontology Repository (MOR) describes a set of software used in the bioinformatics domain. The Workflow editor supports some features such as component discovery and workflow editing.
Regarding to component discovery, the ProGenGrid editor discovers available bioinformatics tools, data banks and graphics tools modelled through the ontology. In particular we classified ProGenGrid components as: data banks, bioinformatics algorithms, graphics tools, drug design tools and input data types. This initial ontology, written in DAML+OIL [13], has been stored in a relational database. To date, we have implemented the Blast [14] and Fasta [15] algorithms as Web Services using the gSOAP Toolkit [16] and GSI-Plugin [17]. Details about this implementation are out of the scope of the present paper. Regarding to data banks, we are working to provide a semantic Data Access and Integration (DAI) service for accessing heterogeneous and distributed bioinformatics database, typically in a flat file format, such as Swiss-Prot, PDB and CATH. In order to provide graphics tools for visualizing biological data we have considered the Rasmol [18] software that is a very powerful tool and it is publicly available. Through our system, that exploits the GRB libraries, we do not need to install any Rasmol software on the local machine but run it by a remote resource and redirect the output on our screen. Finally regarding the drug design tools, to date we are exploiting the AutoDock [19] software, updating its architecture and making it a parameter sweep application to support a large screening of component drug and the docking phase with a target molecule. Since we are considering such components as web services, we use the novel GridLab iGrid information service to manage registration and retrieval of such services. Using iGrid (it is Web Services compliant and has been developed with the same technologies of our workflow – gSOAP and GSI plug-in), we can register, unregister, update and lookup our services knowing details about the WSDL location url and access url to contact the service. These information are used by the scheduler for retrieving chosen services.
Discovered components are made available to a semantic editor that allows the design (i.e., the activities are modelled using UML) of an experiment (abstract workflow). During workflow creation the abstract workflow is validated through rules derived by metadata and ontology. As an example of rule we have considered all kinds of formats supported by the analyzed algorithms so we have defined the following rule:

<activity_prev:activity_succ:format>

where *activity_prev* and *activity_succ* represent the order of the applications and *format* is the kind of supported format.
The scheduler searches for resources and assign them to matched tasks. In this phase the abstract workflow is translated into an execution plan (concrete workflow) containing the activities order and the logical name of the resources (needed for their discovery in a Grid environment). The execution plan (EP) is coded through a set of XML instructions extending the GGF workflow specification. Finally, the tasks defined in the execution plan are executed, the status is monitored and the results are reported.

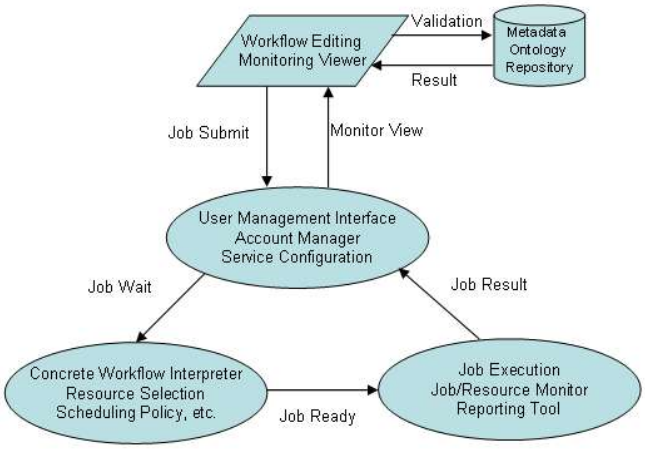

Fig. 1. ProGenGrid Workflow Architecture.

4. Implementation

In this section we describe the design and simulation of a biological experiment that consists in the alignment of protein sequences and visualization of the related set of similar proteins obtained as result. This operation represents a basic component of more complex biological experiments and it involves a few activities, requiring a pre-defined set of hardware and software resources. In particular for the alignment of sequences we chose the Blast algorithm which carries out a local alignment between a sequence provided by the user and a set of sequences, stored in a data bank. The example input sequence is a protein sequence whose PDB code is 1LYN, that is a fertilization protein in FASTA format, while the data bank of reference is Swiss-Prot.
The result of the comparison between the input (target) sequence and those of reference (one-to-more alignment) is a list of proteins similar to the target one, ordered on the basis of a decreasing score. Each resulting sequence is identified by a unique number that is used for carrying out an association with other data banks including those containing secondary structures of protein sequence. The data bank considered is the Protein Data bank (PDB) that contains more than 25000 protein sequences with known 3D structure. Therefore such codes have been searched in the PDB database and the related structure has been extracted. The structure of the protein is in a text file, compressed, containing a set of generic information such as the bibliography or description, chemical-biological information related to electrostatic bonds between each atom of the structure, geometric spatial coordinates, etc. The Rasmol visualization software allows analysing the protein choosing the type of visualization, the angle between atoms and other information useful for the experiment. Usually a user must install different software tools for the computation, has to query via web the data banks cataloguing the information, and must have a good knowledge of the used tools. Through our workflow editor, the user may compose various activities, whose correctness is guaranteed by the semantic editor that checks the validity of the information flow. Hardware and software resources available through a computational grid allow the user to carry out all of the operations without installing any software, the operations are carried out in parallel on several nodes of the grid and hence the result is obtained much faster when compared to the traditional approach (Fig. 2).

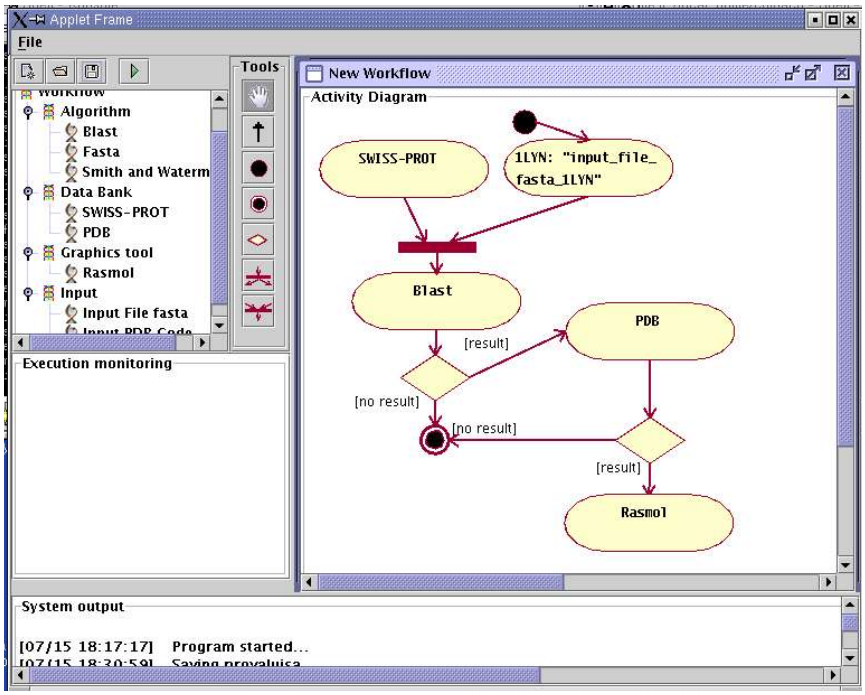

Fig. 2: ProGenGrid editor after the execution of the experiment.

5. Conclusion

In this paper we described the ProGenGrid Workflow Management System that comprises a semantic editor for discovering, selecting and composing bioinformatics tools available in a Grid environment, and a workflow scheduler for running resulting applications. Moreover we presented a case study where we used our workflow system to design and execute a simple sequence alignment application using the bioinformatics tools and data banks installed on our production Grid and wrapped as web services. The main advantage of this system is the possibility for bioinformaticians to compose, model and execute their own biological experiments having a set of tools already configured and running in a distributed environment. The bioinformaticians may not know technical details about the requirements of an application or where applications are actually running. Moreover, the system assists the user in the composition of the experiment, indicating if any error has occurred.
The Globus Alliance and IBM recently proposed the Web Service Resource Framework (WSRF) [20], a set of specification designed to merge Grid and Web technologies by re-phasing the Open Grid Services Infrastructures (OGSI) concepts in terms of currents Web Service standards. So we plan to adapt our architecture to this new specification. Finally we plan to conduct a performance evaluation of our system and to optimize its workflow scheduler.

Reference

[1] Foster, C., Kesselman. The Grid: Blueprint for a New Computing Infrastructure. Published by Morgan Kaufmann (1998).
[2] R. Eshuis and R. Wieringa. Verification support for workflow design with UML activity graphs. In CSE02. Springer Verlag, 2002.
[3] Kreger, H. Web Services Conceptual Architecture. WSCA 1.0. IBM, 2001.
[4] Taverna workflow authoring environment. Site address: http://sourceforge.net/projects/taverna.
[5] Ewa Deelman, James Blythe et al. Mapping Abstract Complex Workflows onto Grid Environments. Journal of Grid Computing 1(1) (2003), 25-39. Site address: http://pegasus.isi.edu.

[6] M. Cannataro, C. Comito, F. Lo Schiavo, and P. Veltri. Proteus, a Grid based Problem Solving Environment for Bioinformatics: Architecture and Experiments. IEEE Computational Intelligence Bulletin 3(1) (2004), 7-18.
[7] Sohrab P Shah, David YM He, Jessica N Sawkins, Jeffrey C Druce, Gerald Quon, Drew Lett, Grace XY Zheng, Tao Xu, BF Francis Ouellette. Pegasys: software for executing and integrating analyses of biological sequences. BMC Bioinformatics 2004, 5:40. Site address: http://bioinformatics.ubc.ca/pegasys/.
[8] Aloisio, G., Blasi, E., Cafaro, M., Epicoco, I. The GRB library: Grid Computing with Globus in C. Proceedings HPCN Europe 2001, Amsterdam, Netherlands, Lecture Notes in Computer Science, Springer-Verlag, 2110 (2001), 133-140.
[9] G. Aloisio, M. Cafaro, I. Epicoco, S. Fiore, D. Lezzi, M. Mirto and S. Mocavero. iGrid, a Novel Grid Information Service. Proceedings of the First European Grid Conference (EGC) 2005, LNCS 3470, Lecture Notes in Computer Science, Springer-Verlag, pp. 506—515, P.M.A. Sloot et al. (Eds.), 2005.
[10] H. P. Bivens. Grid Workflow. Grid Computing Environments Working Group Document, 2001. Site address: http://dps.uibk.ac.at/uploads/101/draft-bivens-grid-workflow.pdf.
[11] WfMC. Workflow management coalition reference model. Site address: http://www.wfmc.org/.
[12] OMG. Uml- unified modeling language: Extensions for workflow process definition. Site address: http://www.omg.org/uml/.
[13] Daml.org. Daml+oil language. Site address: http://www.daml.org/2001/03/reference.html.
[14] Altschul, Stephen F., Gish Warren, Webb Miller, Eugene W. Myers, and David J. Lipman (1990). Basic local alignment search tool. J. Mol. Biol. 215:403-410.
[15] W. R. Pearson and D. Lipman. Improved tools for biological sequence comparison. PNAS, 85:2444-2448, 1998. Site address:http://fasta.bioch.virginia.edu/.
[16] R.A. Van Engelen, K.A. Gallivan. The gSOAP Toolkit for Web Services and Peer-To-Peer Computing Networks. Proceedings of IEEE CCGrid Conference, May 2002, Berlin, pp. 128-135.
[17] M. Cafaro, D. Lezzi, R.A. Van Engelen. The GSI plugin for gSOAP. Site address: http://sara.unile.it/~cafaro/gsiplugin.html.
[18] Roger A. Sayle and E. J. MilnerWhite. RasMol: Biomolecular graphics for all. Trends in Biochemical Science (TIBS), September 1995, Vol. 20, No. 9, p.374. Site address: http://www.umass.edu/microbio/rasmol/.
[19] AUTODOCK. Site address: http://www.scripps.edu/pub/olsonweb/doc/autodock/.
[20] The WS-Resource Framework. Site address: http://www.globus.org/wsrf/.