Homogenizing Access to Highly Time-Consuming Biomedical Applications through a Web-Based Interface

Luigi Grasso, Nuria Medina-Medina,
Rosana Montes-Soldado, and María M. Abad-Grau

Dept. Lenguajes y Sistemas Informáticos - CITIC, Universidad de Granada,
Granada, Spain

Abstract. The exponential increment in the production of biomedical data is forcing a higher level of automatization to analyze it. Therefore, biomedical researchers have to entrust bioinformaticians to develop software able to process a huge amount of data on high performance unix-based servers. However, most of the software is developed with a very basic, text-based, user interface, usually because of a lack of time. In addition the applications are developed as independent tools yielding to a set of specific software tools with very different ways of use. This implies that final users continuously need developer support. Even worse, in many situations only developers themselves are able to run the software every time is required. In this contribution we present a Web-based user interface that homogenizes the way users interact with the applications installed in a server. This way, authorized users can add their applications to the Web site at a very low cost. Therefore, researchers with no special computational skills will perform analysis by themselves, gaining independence to run applications whenever they want at the cost of a very little effort. The application is portable to any unix-like system with a php+mysql server.

Keywords: High performance biomedical applications, Web-based user interface, AJAX.

1 Introduction

Complex diseases are explained by the interaction of many genetic factors together with the environment. To shred light about which factors increment the risk of developing a complex disease and how they interact to each other, genome-wide association studies [1] as well as gene expression profiling [2] or a combination of both [4] are currently being used.

The main feature of these data is their large size, which makes an analysis to be a highly time-consuming task. As an example, genome-wide data sets with thousand gigabytes are becoming a common source of data to be analyzed to detect genetic factors of complex diseases. Analyses are usually performed in high performance computers, running usually under a unix-like operative system and equipped with many processors and a large amount of random access memory. Quite often they are part of large clusters or grids. Perhaps because of the lack of time imposed by the highly competitive that scientific research has become in the biomedical field, software developers focus mostly on functional requirements and computational time. Therefore, most biomedical applications developed at the laboratory have a very simple text-based user interface. In addition, the user manual, in the case it exists, is not complete, hardly understandable and/or does not follow any standard for user manual production. This is the case of rTDT [8], BLink [7], Phase [10], FastPhase [6] or TDT_P [1], all of them used to process genetic data sets.

Generally, biomedical researchers ask to bioinformaticians not only to develop software but also to run it to perform their analyses, as coping with shell commands and scripts in unix-like OS requires a steep learning curve that usually cannot climb. The need of using applications which have text-based user interfaces and scarce user manuals forces them to invest a considerable effort every time they want to run a new software application.

In the last years, many Web-based tools are being offering to biomedical research to easily launch high-performance applications [5], such as those to perform DNA annotations or phylogeny reconstruction. However, as they usually can be freely accessed, the resources on their servers become very busy and they cannot be used to process large files. They usually lack in flexibility too. Therefore, among those that provide a common entrance to launch more than one application, they do not gather applications from different research fields or let alone provide an integrated tool to add a new application to the system. As an example, NPS@ (Network Protein Sequence Analysis) or GPS@ (Grid Protein Sequence Analysis) [3], a more powerful version of NPS@ for grid computing, allows the user to easily interact with many of the most common resources for protein sequence analysis but cannot cope with other tasks such as protein sequence and expression combined analysis. As an example of a Web-based user interface providing access to any software resource in a grid computing environment is the UCLA Grid Portal [9], which therefore is very useful to biomedical researchers with granted access to the grid. However, only portal administrators can add an application to the portal. Moreover, it is hardly portable, as it is only a portal but not a workframe that can be used to create portals on any web server.

In this contribution we present a different approach to provide a Web-based user interface to easily access to any software resource. Our approach consisted on developing BioBench, a framework to create Web-based computational workbenchs, i.e. Web-based user interfaces to provide access to any software installed in a computer system. The main goals of BioBench were (1) efficacy: the user can access through the Web to all disks and processing resources they are granted in the system; (2) portability, so that any laboratory may install BioBench to create their own Web application to homogenize access to their software; (3) flexibility: the Web application evolves depending on the needs of each laboratory, by easily adding every new software that can be useful and removing those that are not used any more; and (4) simplicity: as the tool has been designed to be used by non-expert users. Compared with more complex frameworks, the tool functionality and design is simple. Therefore, for it to run in a grid configuration there must be a software layer between the Web server and the OS with high-level networking protocols and more stringent security capacities.

Section 2 is devoted to explain the main features of *BioBench*, the framework developed for the automatic construction of Web-based computational workbenchs. In Section 3 we introduce BiosH, a workbench (http://bios.ugr.es/BiosH) that has been created to provide and use the software applications through a Web-based user interface. The main conclusions appear in Section 4.

2 BioBench: A Framework to Integrate and to Transform Text-Based User Interfaces into Homogeneous Web-Based Ones

2.1 Description

The main purpose of BioBench was to easily equip text-based bioinformatic applications running under unix-like servers with a more friendly user interface in a way that required very little time for software developers to produce this user interface and for biomedical researchers to launch these applications.

A Web-based application seemed to be a very appropriate way to do this, as it also would reduce installation work for users to zero. The Web interface will ease application sharing among different users and will improve availability. Another important requirement was to use batch processing whenever a task was going to need a long time to be completed. Task completion would be communicated to the user through email. Therefore, BioBench was developed with these features.

Five different types of users are managed by the system, identified by the following roles: unidentified user, visitor, standard user, expert user and administrator. All except an unidentified user are registered users. User roles are related by an inheritance relationship, so that all functionalities of an ancestral role are inherited for all its descendants (see Fig. 1).

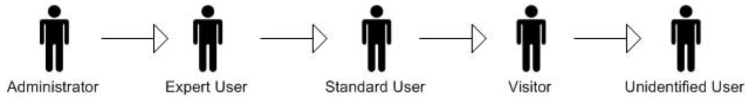


Fig. 1. The inheritance relationship between the user roles in BioBench

An unidentified user is only allowed to login as a registered user, to register as a visitor, to access information about BioBench functionality and to download BioBench. Registered users must have a unix account in the server where a Web-based workbench, i.e. a Web-based application using BioBench, is installed.

Visitors may see their account information and ask to be promoted as an standard or expert user. Standard users can also run any software already registered in the Web-based workbench, see information about other users registered and manage their own user profiles. As standard users may want to know which other users use the applications in the server for a further collaboration, we have added the possibility for them to get that information. In order to launch an application, they have to choose the software to be run, the server path where the input files are placed, the server path where the output files should be place, in case this is necessary, and the software arguments. Expert users have the same rights of standard users plus the ability to register a new application and to delete or modify software created by the same user. The software to be registered must be already installed in the server. The expert user has to provide several data to the system, such as the server path where the software can be found or a description of the parameters required for the software to be run. Additional responsibilities of system administrator are to promote/step-down a user and to install and export BioBench. Its main functions are summarized in Table 1.

Table 1. Main functions of BioBench

Functionality for unidentified users
Register as a visitor
Login as a registered user
Read information about the functionality of BioBench
Read install documentation
Download BioBench
Functionality for visitors
See information about my account
Ask for promotion as standard or expert user
Functionality for standard users
See information about registered users
Run a registered application
Manage system files
Functionality for expert users
Register a new application
Modify/delete a registered application
Functionality for the system administrator
Promote/step-down a user
Install BioBench[1]
Export BioBench

2.2 Design

The logic architecture of BioBench is structured in three main layers, according to the model-view-controller design pattern, in order to separate responsibilities

[1] This is the only function that has to be performed through a text-based user interface (unix terminal).

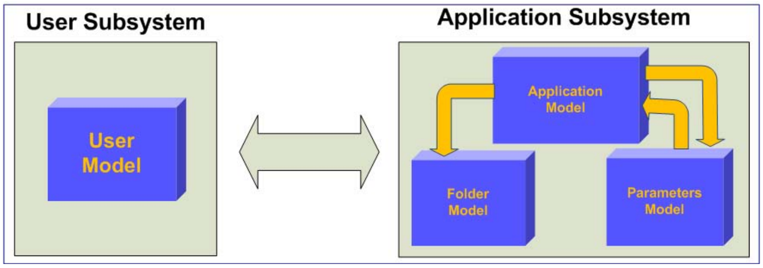


Fig. 2. Logic architecture of BioBench

and ensure that the code implementing the software functionality and accessing the data (the model) is independent of the user interface provided to access to this information (the view). The controller layer updates the view every time a change is performed in the model. The model layer is divided into two subsystems: the user subsystem and the application subsystem. Figure 2 shows the architecture of the model layer. The user subsystem contains the user model, responsible for the representation and management of users. The application subsystem is subsequently divided into three models: (1) the application model, in charge of all the software tools for which BioBench provides a unified Web interface, (2) the parameters model, in charge of the management of the parameters for each software application and (3) the folder model, which represents and controls disk units and folders accessed by users and applications.

The physical architecture of BioBench and its interaction with users, other software and hardware is shown in Fig. 3. BioBench can be used to create a Web-based computational workbench in any server with a unix-like OS, Apache server, php and MySQL. However, for it to work in a cluster or even more, a grid configuration, so that applications and/or data from more than one computer can be accessed by it, other software and additional security procedures are required. Therefore, the use of the Apache capability has to be complemented with an extended version named General Remote Access, which considers the URI of the linked servers (cluster nodes or grids), the list of granted users and their credentials. According to this architecture, we need to install a small application on each linked server to act as a client software that interacts with the main server. This also enables the possibility of monitoring the processes running in the server.

2.3 Software Development

To develop a highly interactive system with a fast human-machine interaction, we chose to use *Asynchronous JavaScript And XML (AJAX)* to develop the Web-based application. To speed up the application development, we chose to use *Xajax*, an open-code PHP library to build Web-based applications using AJAX. The Xajax library includes all the java-script functions in order to build the front-end application or update the view of the Web page obtaining the

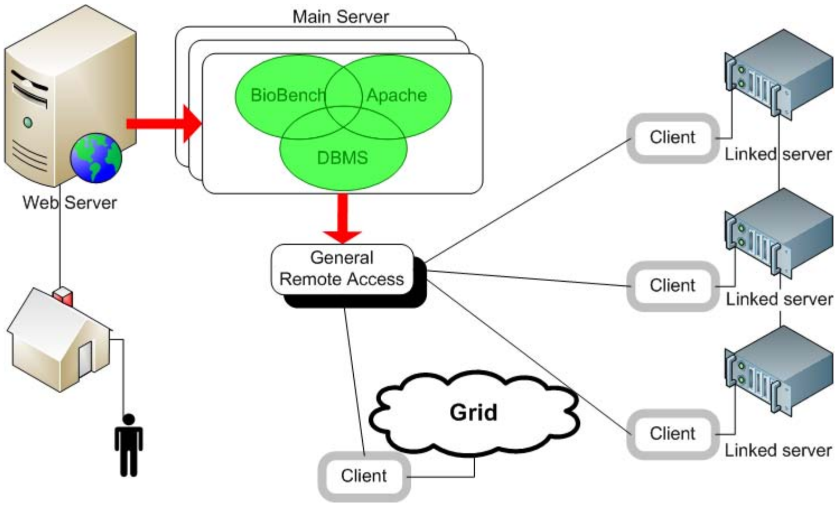


Fig. 3. Physical architecture of BioBench

html code from the php functions. We also used Prototype library (version 1.6) in order to benefit from the high potential of its functionality and simplify the implementation task. The application requires a database to store all the data such as information about all the applications and their parameters, the system users and to set up a permission protocol to model relationships between actions and user roles. BioBench uses a relational database with tables created from a set of 9 entities: Action, Application, Args, Description-App, Description-Arg, Labels-act, Labels-arg, Types and User. As a database management system we chose to use MySQL. Each php object creates a connection with the database using ADOdb, Database Abstraction Library for PHP (and Python) and MySQL. We have adapted a simple library, called eXplorer, which allows a remotely manage of folders and files interacting with the Xajax library. BioBench has been developed under the GNU General Public License (GNU GPL) 3.0. A Web site (http://bios.ugr.es/BioBench) from which the application can be downloaded, has been built at bios.ugr.es, a linux server where several bioinformatic applications have being built for biomedical analyses.

3 BiosH: A Case Study

BioBench has been used to create BiosH, a Website (http://bios.ugr.es/BiosH) to centralize the software that is being used at the laboratory of a group of molecular biologists at the Spanish Council of Scientific Research (CSIC) in Granada. Lately, one member of the team has moved to the University of Sevilla but still does scientific research in collaboration with her former laboratory in Granada. Therefore, the existence of BiosH has become now even more useful, as it is a Web-based tool which does not require to be installed every time a user moves to a different place.

Traditionally, biologists at this laboratory needed bioinformaticians to write for them a new software any time there not was any available software for doing the task. As all the research fields regarding the analysis of genome or transcriptome data sets evolve at an amazingly fast pace, the need of ad-hoc software appeared frequently at the laboratory. As a solution, these bioinformaticians used BioBench to build BiosH and as soon as it was launched, it started evolving with more and more software applications added to it.

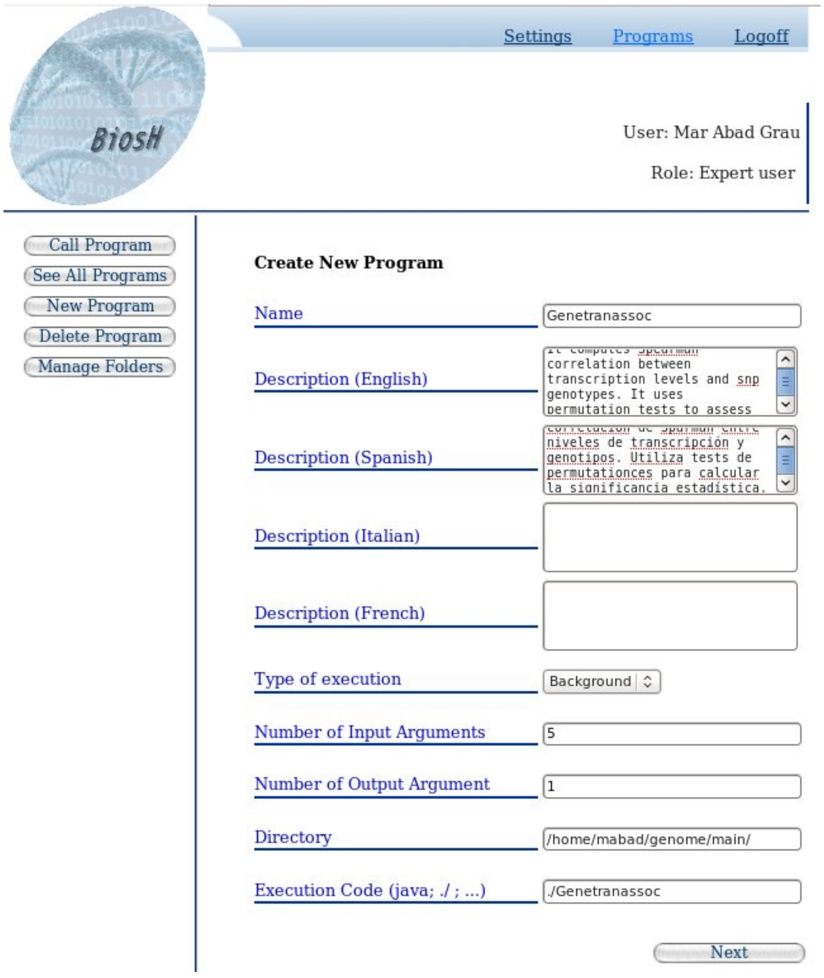
As an example of the potential of BioBench, we describe the steps that were done in order to use BiosH to perform a set of calculations related with genome and transcriptome combined analysis to satisfy the research needs of some biologists at the laboratory. Table 2 shows the input data the biologists at the laboratory had (first row) and the outputs they were looking for (second row). The last row describes the sequence of applications and their arguments (parameters inside squared brackets has to be replaced by the real arguments) that are required to get the outputs from the inputs. Therefore, all these applications were incorporated to BiosH for the biologist to use them. *ImportFormat* and *Transpose* applications are only required to change input and output formats respectively. *SelectCommonSNPs* is an application to perform a preprocessing step of marker filtering. Finally, *Genetranassoc* is the application that performs the computations to obtain association results using the Spearman correlation measure and their p values.

For the whole task to be performed through BiosH, an expert user had to introduce the applications that were not already at BiosH. In this use case, only Genetranassoc was not in BiosH so that it was the only one application that had to be added to the system. However, her developer was not in BiosH either

Table 2. An example of task that was performed through BiosH

Input	I1. Text file with transcriptions for a set of i individuals (columns) and g genes (rows) I2. Text file (makeped format) with genotypes from the same population I3. Text file (makeped format) with genotypes from another population to select major alleles I4. One-column text file with p rows with genetic positions to compute Spearman correlations I5. The amount of permutations to be performed in order to assess statistical evidence
Output	O1. Text file with gene Spearman correlation coefficients and p values O2. Text file with [I5] rows and $g \times p$ columns with the Spearman value for each permutation
Applications	1. ImportFormat PED [I2] [I2].gou 2. ImportFormat PED [I3] [i3].gou 3. SelectCommonSNPs [I2].gou [I2]Selection.gou I4 4. SelectCommonSNPs [I3].gou [I3]Selection.gou I4 5. Genetranassoc [I1] [I2]Selection.gou [I3]Selection.gou [O1].t 6. Transpose [O1].t.csv [O1]

and was added by the Web administrator using the option under the 'Settings' link available to administrators to add users. Figure 4 shows the screenshot with the first form that was filled to add Genetranassoc to BiosH. The main information that had to be provided, besides the application name and path where it is installed, were whether the application has to be run in background, the arguments required and their type and description. On the left of the figure, we can observe all the functionality an expert user has regarding the applications (named programs in the workbench). As Genetranassoc can accept 5 arguments, other 5 forms collecting information for each parameter were filled for the application to be used through BiosH. Once all the applications required for the task in Tab. 2 were in BiosH, the biologist at the laboratory interested in that



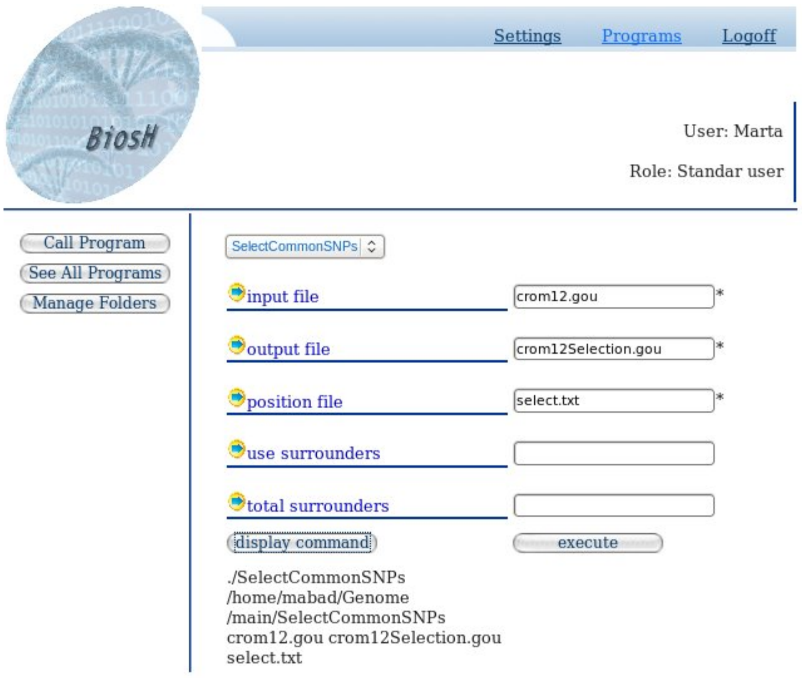
User: Mar Abad Grau

Role: Expert user

Create New Program

Name	Genetranassoc
Description (English)	correlation between transcription levels and snp genotypes. It uses permutation tests to assess
Description (Spanish)	niveles de transcripción y genotipos. Utiliza tests de permutationces para calcular la significancia estadística.
Description (Italian)	
Description (French)	
Type of execution	Background
Number of Input Arguments	5
Number of Output Argument	1
Directory	/home/mabad/genome/main/
Execution Code (java; ./ ; ...)	./Genetranassoc

Fig. 4. Sreenshot showing the first form filled to add the application Genetranassoc to BiosH



User: Marta

Role: Standar user

SelectCommonSNPs

input file	crom12.gou *
output file	crom12Selection.gou *
position file	select.txt *
use surrounders	
total surrounders	

```
./SelectCommonSNPs
/home/mabad/Genome
/main/SelectCommonSNPs
crom12.gou crom12Selection.gou
select.txt
```

Fig. 5. Sreenshot showing the form that has to be filled in order to run the application SelectCommonSNPs through BiosH

task was able to do it without any help and under the role of standard user, provided that she had a user account and enough disk space to store output results in the linux system were BiosH is installed. Figure 5 shows as an example the screenshot with the form that was filled by her to perform the step 3 described in Tab. 2.

4 Conclusions

The quick growth of scientific research in the biomedical field and the huge amount of data from genomes, transcriptomes, etc. that has to be processed is significantly changing the way researchers process them. Hand-made processing is not conceivable any more and software applications are constantly developed to be used in the laboratory. These applications are usually run in high-performance computers with several processors and a large central memory storage under unix-like OS. However, the high increase in work load that bioinformaticians, biostatisticians or any other researchers have as software developers is forcing them to write applications with a simple text-based user interface and no user documentation which are usually only understood and used by their creators. Moreover, many biomedical researchers are not used to text-based interfaces under unix servers and they usually ask somebody to run the applications for them. Therefore, to ease the use of bioinformatic applications is being a very demanded task by biomedical researchers. This way, they instead of the software developers may run these applications. BioBench reaches this goal as a tool to create workbenchs able to provide a friendly and homogeneous Web interface to the applications installed by any user in a server. Opposite to the very few similar tools, BioBench can be installed in any unix-like OS with a mysql+php Web server and every user can add their self-written software so that it can be easily shared.

Acknowledgment. The authors were supported by the Spanish Research Program under project TIN2010-20900-C04, the Andalusian Research Program under project P08-TIC-03717 and the European Regional Development Fund (ERDF).

References

1. Abad-Grau, M.M., Medina-Medina, N., Montes-Soldado, R., Moreno-Ortega, J., Matesanz, F.: Genome-wide association filtering using a highly locus-specific transmission/disequilibrium test. Human Genetics 128(3), 325–344 (2010)
2. Alekseev, O.M., Richardson, R.T., Alekseev, O., O'Rand, M.G.: Analysis of gene expression profiles in hela cells in response to overexpression or sirna-mediated depletion of nasp. Reprod. Biol. Endocrinol. 7, 7–45 (2009)
3. Blanchet, C., Combet, C., Daric, V., Deléage, G.: Web Services Interface to Run Protein Sequence Tools on Grid, Testcase of Protein Sequence Alignment. In: Maglaveras, N., Chouvarda, I., Koutkias, V., Brause, R. (eds.) ISBMDA 2006. LNCS (LNBI), vol. 4345, pp. 240–249. Springer, Heidelberg (2006)
4. Dimas, A.S., Deutsch, S., Stranger, B.E., Montgomery, S.B., Borel, C., Attar-Cohen, H., Ingle, C., Beazley, C., Arcelus, M.G., Sekowska, M., Gagnebin, M., Nisbett, J., Deloukas, P., Dermitzakis, E., Antonarakis, S.E.: Common regulatory variation impacts gene expression in a cell type dependent manner. Science 325(5945), 1246–1250 (2001)
5. Fox, J.A., McMillan, S., Ouellete, B.F.: A compilation of molecular biology web servers: 2006 update on the bioinformatics links directory. Nucleic Acids Research 34, W3–W5 (2001)
6. Scheet, P., Stephens, M.: A fast and flexible statistical model for large-scale population genotype. data: Applications to inferring missing genotypes and haplotypic phase. Am. J. Hum. Genet. 78, 629–644 (2006)
7. Sebastiani, P., Abad-Grau, M.M.: Bayesian estimates of linkage disequilibrium. BMC Genetics 8, 1–13 (2007)
8. Sebastiani, P., Abad-Grau, M.M., Alpargu, G., Ramoni, M.F.: Robust Transmission Disequilibrium Test for incomplete family genotypes. Genetics 168, 2329–2337 (2004)
9. Slottow, J., Korambath, P., Jin, K.: The integration of ajax, interactive x windows applications and application input generation into the ucla grid portal. In: Proceedings of the IEEE International Symposium on Parallel and Distributed Processing (2008)
10. Stephens, M., Smith, N.J., Donelly, P.: A new statistical method for haplotype reconstruction from population data. Am. J. Hum. Genet. 68, 978–989 (2001)