Cobitis: A Computational Biology Tool Interoperability Scheme for easy Web services implementation, integration and access.

Uberto Pozzoli[1,*] , Matteo Cavalleri[2], Giorgia Menozzi[1], Manuela Sironi[1].

[1] Bioinformatics Lab. - Scientific Institute IRCCS "E.Medea" - Bosisio Parini (LC) – Italy
[2] Biomedical Engineering Lab. - Scientific Institute IRCCS "E.Medea" - Bosisio Parini (LC) – Italy
[*] Corresponding author: email: upozzoli@bp.Lnf.it

Introduction

The number of computational algorithms in biology, their availability and their utilization is steadily increasing. Nonetheless to use them from software applications and computational systems still requires programming and data format conversion. Moreover their accessibility from web interfaces, while expanding their diffusion, makes even harder their programmatic use. On the other hand the complexity of the tasks they are involved in, almost always, requires integration between different methods and repetitive runs on different datasets. To face this problems, primarily in our lab, we are developing a set tools based on XML and SOAP. Our goal is to obtain a system able to meet the following requirements.

- Easy programmatic access.
- Easy data interchange and deployment.
- Easy web interface implementation.
- Optimization of networked hardware resources.
- Programming language and platform independence.
- Utilization of free/open source software.

Given the focus on programmatic access, the definition of ontologies on data types is not one of our target.

Cobitis core module

The core of Cobitis is an XML scheme. This scheme defines a hierarchy of simple data types that ideally can describe any data type and structure. The most important feature is the definition of the Container element that allows to virtually represent any data structure. An ANSI C++ hierarchy of classes maps each data type defined in cobitis, allowing data manipulation, XML and SOAP encoding/decoding. Utilization of these classes enables easy Web content production by the way of an XSLT filter, easy Web form creation (once Browsers will be XForms enabled) and SOAP based Web services access. Classes constitute the code base on which the CobiServer and CobiClient modules relies but they can also be used in algorithm implementation. It is straightforward to implement layers between Cobitis Classes and data structures defined elsewhere (i.e. NCBI toolbox, Matlab, R, Perl. Php etc); in this way cobitis based applications can easily access algorithms already implemented. On the other hand it is also possible to access methods based on Cobitis from other systems (i.e. Matlab, BioPerl, R). The core module is based upon STL and gSOAP, a freely available, open source package. Particular care have been given to minimize and isolate code dependency.

Cobitis server module

The server module is based on the cobitis class hierarchy. It behaves as a multi-threaded SOAP server and as a basic multi-threaded HTTP server (allowing WSDL publishing and data retrieving). Its methods can be either local (implemented as cobitis based dynamic

library modules) or remote (it contains a copy of the client module). It publishes a WSDL based on available modules and on accessible remote methods. The server can be contacted from any SOAP enabled client. Three specific methods are always exposed and used by the cobitis client to make calls to all other methods (possibly in an asynchronous way) and to retrieve results. This is of particular interest when long computational tasks are needed. A request mapper checks requested method existence and parameters consistency and decides if the request has to be managed locally or remotely whenever both the options are available. Servers generates URLs pointing to results, allowing both HTTP and SOAP retrieving.

Cobitis client module

The Cobitis client module can query a server to get WSDL and call methods of interest by the way of three specific standard SOAP calls (always exposed by a cobitis server). CobiCall gets method name and a container as parameters. It checks if the method is exposed by one of the server the client is aware of and, if the container elements are consistent with the method requested, it queries the server asynchronously returning results in a container only when the task is completed. The CobiCallURL method works in the same way but it returns as soon as the server responds with the results URL. This URL is valid even if the request is not yet completed on server side and contains information about the task status. It can be fetched using the CobiGetResult method or an HTTP GET URL.

All together now!

From the client point of view, Cobitis servers, thanks to their client capabilities, can interoperate transparently. This allows the sharing of computational resources and methods between different machines over a network. User authentication can be used to selectively share methods. Furthermore the same method can be exposed by more than one server and, when a request is received, a workload criteria can be transparently used to select which server (i.e. machine) will actually perform the task, thus obtaining a better exploitation of hardware resources.

Client applications can be easily developed integrating client modules with the base cobitis classes and, possibly, with layers to other data structures. For instance we developed a Matlab client that makes possible to call CobiServer methods from within a complete computational environment. A Web Server (i.e. an Apache module) is being developed to allow Cobitis Services use from the Web.