BioRuby 2010 updates: moving to agile bioinformatics

Raoul J.P. Bonnal[1], Naohisa Goto[2], Mitsuteru Nakao[3], Jan Aerts[4], Pjotr Prins[5] ,Toshiaki Katayama[6]

1 Fondazione INGM, Milan, Italy bonnalraoul@ingm.it; 2 Research Institute for Microbial Diseases, Osaka University, Japan; 3 Database Center for Life Science, Tokyo, Japan; 4 Wellcome Trust Sanger Institute, Cambridge, UK; 5 Wageningen University and Groningen Bioinformatics Center, Netherlands; 6 Institute of Medical Science, University of Tokyo, Japan

URL: http://bioruby.org
Code: http://github.com/bioruby/bioruby
License: The Ruby License

BioRuby provides tools and libraries for the Ruby programming language with the aim of providing an integrated environment for bioinformatics. The Ruby programming language was incepted around 1993 and is a reflective, general purpose object-oriented programming language that combines syntax inspired by Perl, with Smalltalk-like features (http://www.ruby-lang.org). Ruby has gained in popularity by virtue of its clean object oriented programming (OOP) design and its functional programming characteristics. These characteristics allow programming with less source code, thereby improving programmer productivity (Aerts and Law, 2009). The BioRuby project is started in 2000 and after the initial release in 2001, we made a steady development to its functionality. The project is supported by the Open Bioinformatics Foundation (OBF) initiative, which hosts the website and mailing list. The current software development tree is hosted on GitHub (https://github.com/), a public distributed source control system, which allows any developer to start contributing code to the BioRuby project.

Here we report recent developments in BioRuby. Since the last BOSC presentation in 2007 we had two major releases, 1.3.0 and 1.4.0. Firstly, we enhanced the support for web services as many bioinformatics resources are now being provided through SOAP and REST services. Notable efforts are made to utilize REST web services like the ones provided by TogoWS (http://togows.dbcls.jp/) and NCBI (http://eutils.ncbi.nlm.nih.gov/). Sponsored by the DBCLS during the BioHackathons in 2008, 2009, and 2010 we then tried to define next challenges for BioRuby. We made a lot of refactoring to ensure the correctness of file formats with conversion among rich annotated sequence objects. Internally, we introduced the ActiveRecord model provided by the Ruby on Rails (http://rubyonrails.org) framework for the BioSQL (http://biosql.org/) module to abstract the RDBMS layer inside. With the help of Google Summer of Code (http://code.google.com/soc/) and National Evolutionary Synthesis Center (NESCent http://www.nescent.org/) in 2009, BioRuby can handle PhyloXML data efficiently. A lot of bioinformatics centers are using or acquiring data from Next Generation Sequencers (NGS), so the FASTQ file formats is supported in all of its three variants, and supports for DNA chromatogram data in SCF and ABIF formats have been addded to the framework. Most recently, in collaboration with other Open Bio* projects like BioPython, we started to implement new Semantic Web technologies like RDF and SPARQL for the bioinformatics data mining. Finally, we adopted the Ruby Unit Testing framework to improve the quality of our code, and to break the barriers for newbies, all the documentation has been revised adding new tutorials and a better code description.

BioRuby hopes to have more contributors without influencing the stable core of the framework, so we'll introduce a plugin system to embrace the "agile" development reflecting the Rails' approach. This plugin system gives the chance for both BioRuby developers and other bioinformaticians to try the beta status code. This mechanism shows a potential when data needs not just to be analyzed in an efficient and easy way but, also presented in the right way; usually that means a fancy way. Bio::Graphics (http://bio-graphics.rubyforge.org/) is an example, which adds functionality to graphically represent biological objects. It fits exactly into the plugin philosophy and introducing the concept of view (in MVC model) into any BioRuby object that can be exported or published over the web. To handle the plugin system, we will improve the BioRuby shell and promote the integration with Rails. Heterogenous data integration is another aspect that should be more "agile", introducing new technologies like semantic web and supporting more web-services could reduce the needs to replicate information.