

# Biogem: an effective tool based approach for scaling up open source software development in bioinformatics

Raoul J.P. Bonnal<sup>1\*</sup>, Jan Aerts<sup>2</sup>, George Githinji<sup>3</sup>, Naohisa Goto<sup>4</sup>, Dan MacLean<sup>5</sup>, Chase A Miller<sup>6</sup>, Hiroyuki Mishima<sup>7</sup>, Massimiliano Pagani<sup>1</sup>, Ricardo Ramirez-Gonzalez<sup>8</sup>, Geert Smant<sup>9</sup>, Francesco Strozzi<sup>10</sup>, Rob Syme<sup>11</sup>, Rutger Vos<sup>12</sup>, Trevor J Wennblom<sup>13</sup>, Ben J Woodcroft<sup>14</sup>, Toshiaki Katayama<sup>15</sup>, and Pjotr Prins<sup>9†</sup>

<sup>1</sup>Integrative Biology Program, Istituto Nazionale Genetica Molecolare, Milan, Italy; <sup>2</sup>ESAT/SCD, Faculty of Engineering and IBBT Future Health Department, University of Leuven, Belgium; <sup>3</sup>KEMRI-Wellcome Trust Research Program, Kilifi, Kenya; <sup>4</sup>Research Institute for Microbial Diseases, Osaka University, Japan; <sup>5</sup>The Sainsbury Laboratory, Norwich, UK; <sup>6</sup>Biology Department, Boston College, USA; <sup>7</sup>Department of Human Genetics, Nagasaki University Graduate School of Biomedical Sciences, Japan; <sup>8</sup>The Genome Analysis Centre, Norwich, UK; <sup>9</sup>Laboratory of Nematology, Wageningen University, the Netherlands; <sup>10</sup>Parco Tecnologico Padano, Lodi, Italy; <sup>11</sup>Dept Environment & Agriculture, Curtin University, Australia; <sup>12</sup>NCB Naturalis, Leiden, the Netherlands; <sup>13</sup>Silicon Life Sciences, Minneapolis, USA; <sup>14</sup>Department of Biochemistry and Molecular Biology, University of Melbourne, Australia; <sup>15</sup>Laboratory of Genome Database, Human Genome Center, Institute of Medical Science, University of Tokyo, Japan

Associate Editor: Prof. Martin Bishop

## ABSTRACT

**Summary:** Biogem provides a software development environment for the Ruby programming language, which encourages community-based software development for bioinformatics while lowering the barrier to entry and encouraging best practices.

Biogem, with its targeted modular and decentralized approach, software generator, tools, and tight web integration, is an improved general model for scaling up collaborative open source software development in bioinformatics.

**Availability:** Biogem and modules are free and open source software. Biogem runs on all systems that support recent versions of Ruby, including Linux, Mac OS X and Windows. Further information at <http://www.biogems.info>. A tutorial is available at <http://www.biogems.info/howto.html>

**Contact:** Raoul J.P. Bonnal ([bonnal@ingm.org](mailto:bonnal@ingm.org))

## 1 INTRODUCTION

In biomedical science, new technologies, data formats, and methods emerge continuously. Scientists want to take advantage of these developments as soon as possible, which requires bioinformatics software to keep up with new requirements. We support the notion of the Open Bioinformatics Foundation (OBF) that development of collaborative open source software (OSS) is essential for bioinformatics. The OBF represents a number of important projects,

such as BioPerl (Stajich *et al.*, 2002), Biopython (Cock *et al.*, 2009), BioRuby (Goto *et al.*, 2010), and BioJava (Holland *et al.*, 2008). These Bio-star (Bio\*) projects effectively function as community centres and share a centralised approach in software development with large source code repositories. Bio\* projects, generally, aim for consolidated tools, a stable application programming interface (API), and backwards compatibility.

Within the BioRuby project we experienced the drive for stability easily overwhelmed and discouraged developers. Not only because of the complexity of the existing code base, but also because coding standards are enforced, and extensive tests and documentation are required. Furthermore, newly contributed code may be subject to community scrutiny, and in many cases further demands for improving the code follow. The full process introduces a significant delay between initial idea and final acceptance of the code in the main project. Months, even years, may pass between stable releases of main Bio\* projects. It may take a long time before a new feature is publicly released.

To scale up collaborative software development in BioRuby, we recognised existing and new developers need to be encouraged to contribute more code. To achieve this, we created Biogem a Ruby application framework for rapid creation of decentralised, internet published software modules written to lower the barrier to entry. Biogem was initially inspired by the R/Bioconductor packaging system (Gentleman *et al.*, 2004), which encourages software developers to publish software modules independently using simple rules; and Ruby on Rails (RoR) plugins (Thomas *et al.*, 2006), which provides a software generator and modular software plugin system.

\*to whom correspondence should be addressed

†R. Bonnal, T. Katayama and P. Prins contributed equally to this manuscript

## 2 FEATURES

For Biogem we created specific tools to support the creation of bioinformatics software functionalities and to support development 'best practises', i.e. infrastructure for software specification, documentation and tests. We also provide tight web integration based on public websites and services. These websites publish and distribute software modules and give web based access to source code, complete with revision history (see Fig.1). Biogem exposes Ruby bioinformatics modules, and makes developer productivity and module popularity visible.

The primary tool of the Biogem framework is a software generator consisting of templates for bioinformatics scripts, source code, software specification, documentation, and tests. With the generator, required directories and files are automatically created from templates for a new software module. Templates are included for commonly encountered tasks, such as command line parameter handling, error handling, make files etc.

Another Biogem tool publishes the versioned module with its dependencies on the internet. The published module is immediately available for download and installation to bioinformatics users in the form of a Ruby gem (i.e. an archive of modular Ruby code with all the supporting files and information needed for installation by 'package manager' software). We refer to a Biogem module as a 'BioRuby plugin' if the module extends the BioRuby project. Published software modules are easily repackaged by software distributions, e.g. Debian Bio Med (Möller *et al.*, 2010) and BioLinux (Field *et al.*, 2006).

The Biogem website (see availability) makes it easy to find and install software modules. The website also allows people to track releases, software dependencies, development activity, outstanding issues, integration test results, documentation and popularity of published modules. A map shows the location of Biogem developers to help foster a sense of international community.

Biogem encourages software development best practices by providing templates for documentation and multiple test driven development strategies; such as unit tests, behaviour driven development, and a natural language parser for software specification (e.g. Chelimsky *et al.*, 2010). A notable difference to the traditional code contribution procedures of the Bio\* projects is that best practices are encouraged, rather than enforced.

Templates are also included for certain types of functionality, e.g. to generate portable SQL database handlers, and to build a dynamic web site. With Biogem it is possible to create a functional web application, or service, in just a few steps. Generating the different features is handled through work flows (see Figure 1).

We added tutorials for Biogem, which explain the software generators, templates and software publishing. These tutorials are part of the software distribution and available online.

We created 'collections' that bundle important modules together as specific releases. For example, 'bio-core' contains stable modules, and 'bio-core-ext' contains stable modules with bindings to C libraries. Special purpose collections exist such as 'bio-biolinux', which is distributed by the Cloud Biolinux project and merged with the Galaxy CloudMan project (Afgan *et al.*, 2010).

In the first eight months of the Biogem functionality becoming available, over twenty new modules have been published through Biogem, showing a wide variety of subjects. These modules, for

example, target big data handling, next generation sequencing, and parsing of bioinformatics data formats (Table 1).

## CONCLUSION

Biogem provides an environment for rapid bioinformatics software development with a low barrier to entry. Biogem frees potential contributors from code maturity expectations that can be deterring, and encourages Ruby developers to contribute experimental source code early to the BioRuby community. Through Biogem software is published in a modular way, and best practises are encouraged through infrastructure for software specification and testing. All this results in better utilisation of existing and new software development manpower, thereby scaling up open source software development in bioinformatics.

We suggest Biogem can serve as a generic model; not by replacing existing Bio\* projects, but by supplementing them with a decentralised and evolutionary model for collaborative software development.

**Table 1.** The introduction of Biogem has led to a broad range of new BioRuby plugins. An up-to-date list can be found at <http://biogems.info>.

Name	Description
bio assembly	read and write assembly data
bio blastxmlparser	fast, low memory, big data BLAST parser
bio bwa	Burrows Wheeler aligner
bio cnls scraper	nuclear localisation signal prediction
bio six frame	sequence translation
bio genomic interval	detect intervals
bio gff3	fast, low memory, big data GFF3 parser
bio isoelectric point	calculate protein isoelectric point
bio kb illumina	Illumina annotations
bio lazyblastxml	another BLAST XML parser
bio logger	sane error handling
bio nexml	NeXML support, for phylogenetic data
bio ngs	NGS workflows and display, incl. support for bio bwa, Bowtie, TopHat, and Cufflinks
bio octopus	transmembrane domain predictor interface
bio restriction enzyme	DNA cutting operations with REBASE
bio samtools	samtools API
bio signalp	signal peptide prediction interface
bio sge	split huge files for cluster computing
bio tm hmm	transmembrane predictor interface
bio ucsc api	UCSC Genome Database binding

## ACKNOWLEDGEMENTS

We thank our four reviewers for constructive and detailed comments; reviewers Brad Chapman and Hilmar Lapp identified themselves. We also thank Steffen Möller for comments. This work was supported by the Research Council KUL SymBioSys and Flemish Government IBBT [PFV/10/016 to JA] ; the Netherlands Organisation for Scientific Research/TTI Green Genetics [1CC029RP to PP] ; the Japan Society for the Promotion of

Science, Grant-in-Aid for Young Scientists (B) [23791230 to HM] ; and the EC FP7/2007-2013 Marie Curie Fellowship [237046 to RV]

## REFERENCES

Afgan, E., Baker, D., Coraor, N., Chapman, B., Nekrutenko, A., and Taylor, J. (2010). Galaxy CloudMan: delivering cloud compute clusters. *BMC Bioinformatics*, **11 Suppl 12**, S4.

Chelimsky, D., Astels, D., Helmkamp, B., Dennis, Z., North, D., and Hellesoy, A. (2010). *The RSpec Book: Behaviour Driven Development with RSpec, Cucumber, and Friends*. Pragmatic Bookshelf Series. Pragmatic Programmers, LLC, The.

Cock, P. J., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., and de Hoon, M. J. (2009). Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, **25**(11), 1422–1423.

Field, D., Tiwari, B., Booth, T., Houten, S., Swan, D., Bertrand, N., and Thurston, M. (2006). Open software for biologists: from famine to feast. *Nat Biotechnol*, **24**(7), 801–803.

Gentleman, R. C., Carey, V. J., Bates, D. M., and others (2004). Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology*, **5**, R80.

Goto, N., Prins, P., Nakao, M., Bonnal, R., Aerts, J., and Katayama, T. (2010). BioRuby: bioinformatics software for the Ruby programming language. *Bioinformatics*, **26**(20), 2617–2619.

Holland, R. C., Down, T. A., Pocock, M., Prlic, A., Huen, D., James, K., Foisy, S., Drager, A., Yates, A., Heuer, M., and Schreiber, M. J. (2008). BioJava: an open-source framework for bioinformatics. *Bioinformatics*, **24**(18), 2096–2097.

Möller, S., Krabbenhöft, H., Tille, A., Paleino, D., Williams, A., Wolstencroft, K., Goble, G., Holland, R., Belhachemi, D., and Plessey, C. (2010). Community-driven computational biology with Debian Linux. *BMC Bioinformatics*, **11 Suppl 12**, S5.

Stajich, J. E., Block, D., Boulez, K., Brenner, S. E., Chervitz, S. A., Dagdigian, C., Fuellen, G., Gilbert, J. G., Korf, I., Lapp, H., Lehvaslaiho, H., Matsalla, C., Mungall, C. J., Osborne, B. I., Pocock, M. R., Schattner, P., Senger, M., Stein, L. D., Stupka, E., Wilkinson, M. D., and Birney, E. (2002). The Bioperl toolkit: Perl modules for the life sciences. *Genome Res*, **12**(10), 1611–1618.

Thomas, D., Hansson, D., Breed, L., and (Firm), P. P. (2006). *Agile web development with rails*. The facets of Ruby series. Pragmatic Bookshelf, second edition.



1

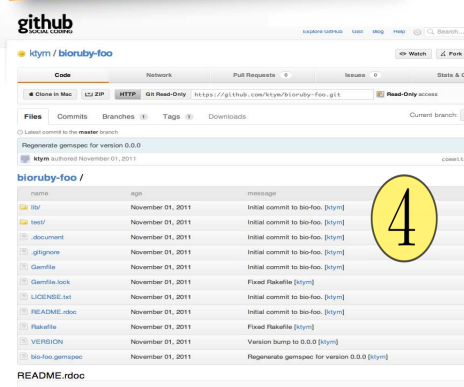
### % biogem foo

```
create scaffold:
bioruby-foo
|-- Gemfile
|-- Gemfile.lock
|-- LICENSE.txt
|-- README.rdoc
|-- Rakefile
|-- VERSION
|-- bio-foo.gemspec
|-- lib
|   |-- bio-foo.rb
|   |-- test
|   |-- helper.rb
|   |-- test_bio-foo.rb
```

### % cd bioruby-foo

```
% edit lib/bio-foo.rb
% edit test/test_bio-foo.rb
% rake test
% git commit -c 'changes'
% rake release
```

2



4

Biogems.info

Biogems		People		Ruby for bioinformatics		RSS							
#	biogem	description	by	date	status	version	released	issues	source	commit	(#)	90%	
1	bio	Bioinformatics library (...)	BioRuby project	cto	1.4.2	2 months	4	0	✓	✓	✓	12872	2205
2	bio gtf3	GTF3 parser for big data (...)	Pjotr Prins	←	0.7.7	3 months	0	0	✓	✓	✓	2348	474
3	bio gem	Biogem helps bioinformaticians start developing plugins/modules for BioRuby creating a scaffold and a gem package (...)	Razal J.P. Bonnal	↗	1.1.1	1 week	3	0	✓	✓	✓	2091	649
30	intermine bio	Biological extensions for the intermine webservice client library (...)	Alex Kalderimis	←	0.98.1	3 months	...	...	✓	✓	✓	104	74
31	bio signalp	A wrapper for the signal peptide prediction algorithm signalp (...)	Ben J Woodroff	←	0.1.0	4 months	0	0	✓	✓	✓	99	67
32	bio phyta	Pipeline to remove contaminations from est libraries (...)	Philipp Comans	↗	0.9.1	4 weeks	0	0	✓	✓	✓	89	89
33	bio data restriction enzyme	Restriction enzyme dataset (...)	Sir Richard J. Roberts, Trevor Wenzel	↗	1.00.0	10 weeks	0	0	✓	✓	✓	79	79
34	bio restriction enzyme	Digests dna based on restriction enzyme cut patterns (...)	Trevor Wenzel	↗	1.0.0	10 weeks	0	0	✓	✓	✓	79	79
35	bio core ext	BioRuby core ext: plugins which require external library or tools so are not pure Ruby plugins (...)	Razal J.P. Bonnal, Pjotr Prins	↗	0.0.1	11 weeks	0	0	✓	✓	✓	77	77
36	bio bioLinux	Meta package for bioLinux distribution (...)	Razal J.P. Bonnal	↗	0.1.0	8 weeks	0	0	✓	✓	✓	65	65
37	bio foo	One-line summary of your gem (...)	Toshiaki Katayama	new	0.0.0	2 weeks	0	0	✓	✓	✓	40	40
38	bio dbsnp	Decoding the dbsnp bitfield (...)	Hiroyuki Mishima	new	0.1.1	2 weeks	0	0	✓	✓	✓	39	39
39	bio ngs	(...)		pre	pre	...	...	...	✓	✓	✓	8	8
40	bio qtlHD	(...)		new	pre	...	...	...	✓	✓	✓	6	6

**Fig. 1.** Biogem eases publication of new bioinformatics Ruby software modules on the Internet, in a few steps. (1) The software generator creates the directory layout and files for a new software module named 'foo'. (2) The developer writes or modifies source code, and (3) quickly and easily publishes the source code and module online, for others to read, install and use. Collaboration (4) is facilitated by publishing source code and changes to navigable websites. Then the workflow continues again at (2). The <http://biogems.info> website tracks published modules. Popularity of each published module is tracked, as well as source code changes, updates, bugs, and issues. Unlike with the practise of publishing scientific papers, collaboration on software often comes *post factum*, i.e. after original publishing of a software module. Therefore it pays to publish software modules early and often. This is reflected in the Biogem workflow.