9th Annual Bioinformatics Open Source Conference

BOSC 2008

1. Welcome

Welcome to BOSC 2008! This is the 9th annual Bioinformatics Open Source Conference held as a Special Interest Group (SIG) meeting in conjunction with the Intelligent Systems for Molecular Biology Conference.

BOSC is sponsored by the Open Bioinformatics Foundation (O|B|F), a non-profit group dedicated to promoting the practice and philosophy of Open Source software development within the biological research community. Many Open Source bioinformatics packages are widely used by the research community across many application areas and form a cornerstone in enabling research in the genomic and post-genomic era. Open source bioinformatics software has facilitated rapid innovation and dissemination of new computational methods as well as informatics infrastructure.

Since the work of the Open Source Bioinformatics Community represents some of the most cutting edge of Bioinformatics in general, the overall theme for the conference this year is “Tackling Hard Problems with Emerging Technologies.” Session topics under this umbrella include cyberinfrastructure, grid computing and workflow management and discovery, and visualization. In keeping with this theme, we welcome our keynote speaker, Julian Lombardi. Dr. Lombardi is one of the original architects of the open source, peer-to-peer OpenCroquet platform for creating and deploying deeply collaborative multi-user online applications and virtual worlds. We will also hear updates about the main Open Source Bioinformatics Software suites.

One of the hallmarks of BOSC is the coming together of the open source developer community in one location. A face-to-face meeting of this community creates synergy where participants can work together to create use cases, prototype working code, or run bootcamps for developers from other projects as short, informal, and hands-on tutorials in new software packages and emerging technologies. We have two formats to facilitate these collaborations: lightning talks (short, 5 minute talks for focused ideas and demos) and Birds of a Feather sessions where small groups gather around a common interest. In short, BOSC is not just a conference for presentations of completed work, but is a dynamic meeting where collaborative work gets done.

Next year will mark the 10th Anniversary of BOSC. In preparation for this milestone, we invite you to join the O|B|F (application forms are available) and the BOSC 2009 planning committee.

Finally, BOSC 2008, our keynote speaker, and student travel fellowships were made possible by Helicos Biosciences Corporation and an anonymous donor. Thank you.

Organizing Committee
Kam Dahlquist (Chair)
Darin London
Hilmar Lapp
Jason Stajich
Chris Dagdigian

Sponsorship

Helicos
BioSciences Corporation

2. Schedule (Day 1)

Friday, July18		
9:00-9:15	Kam Dahlquist (Chair BOSC 2008) Jason Stajich (President O|B|F)	[welcome] Open Bioinformatics Foundation Update
9:15-10:15	Julian Lombardi	[keynote] Croquet: An Open Collaboration Architecture for Scientific Visualization and Simulation
10:15-10:45	**Coffee Break**	
10:45-11:10	Hervé Ménager	[cyberinfrastructure] Mobyle: a New Full Web Bioinformatics Framework
11:10-11:35	Richard Smith	[cyberinfrastructure] Intermine – Open Source Data Warehouse and Query Interface
11:35-12:00	Paul Gordon	[cyberinfrastructure] Bridging Existing Bioinformatics Tools and Data into Semantic Web Services
12:00-12:25	Peter Rice	[cyberinfrastructure] Soaplab2: More Reliable Sesame Door to Bioinformatics Programs
12:30-1:30	**Lunch**	
1:30-1:50	Barry Sanders & David Arcoleo	[cyberinfrastructure] BeeSpace: An Interactive System for Functional Analysis
1:50-2:10	Andrew Jenkinson	[cyberinfrastructure] BioDas Project Update
2:10-2:30	Kazuharu Arakawa	[visualization] E-Cell 3D: 3-Dimensional Visualization of Dynamic Cell Simulation
2:30-2:50	Peter Rice	[bio*update] EMBOSS: European Molecular Biology Open Software Suite
2:50-3:10	Michael Heuer	[bio*update] BioJava Project Update
3:10-3:15	Justin Wilson	[lightning talk] Use the Make Utility for the Maintenance of Complex Bioinformatics Pipelines
3:15-3:30	Additional Lightning Talks/Demos/Birds of a Feather See <http://www.open-bio.org/wiki/BOSC_2008_schedule> for updated listings	
3:30-4:00	**Coffee Break**	
4:00-5:30	Lightning Talks/Demos/BOFs	

2. Schedule (Day 2)

Saturday, July 19		
9:00-9:05	Kam Dahlquist	Announcements
9:05-9:30	Mark Wilkinson	[workflows] Interoperability with BioMoby – Past, Present, and Future
9:30-9:55	Joshua Orvis	[workflows] Ergatis: A Web-based Bioinformatics Pipeline Management and Collaborative Development System
9:55-10:15	Aleksi Kallio	[workflows] Chipster – User Friendly DNA Microarray Analysis Software
10:15-10:45	**Coffee Break**	
10:45-11:10	Jiten Bhagat	[workflows] myExperiment: Social Software for Sharing Workflows
11:10-11:30	Todd Smith	[workflows] BioHDF: Open Binary File Formats for Large Scale Data Management
11:30-11:50	Aaron Kitzmiller	[open source software] The Open HeliSphere^{TM} Project: True Open Source from the Inventors of True Single Molecule Sequencing
11:50-12:10	Ed Lee	[open source software] Apollo: a Sequence Annotation Editor (with demonstration to follow)
12:30-1:30	**Lunch**	
1:30-1:50	Hilmar Lapp	[bio*update] BioSQL Reloaded: 1.0 Release, PhyloDB Module, and Future Features
1:50-2:10	Syed Haider	[bio*update] EMBRACE – BioMart Developments and Future
2:10-2:30	Tiago Antão	[bio*update] Biopython Project Update
2:30-3:30	Lightning Talks/Demos/BOFs See <http://www.open-bio.org/wiki/BOSC_2008_schedule> for updated listings	
3:30-4:00	**Coffee Break**	
4:00-5:30	Lightning Talks/Demos/BOFs	

3. Abstracts

Abstracts appear in the following pages in the order that they are presented at the conference.

Mobyle: a new full web bioinformatics framework

Bertrand Néron[1], Hervé Ménager[1 *], Corinne Maufrais[1], Nicolas Joly[1], Pierre Tuffery[2], Catherine Letondal[1]
[1]: Groupe Logiciels et Banques de Données, Institut Pasteur, 28, rue du Dr Roux, 75724 PARIS Cedex, France
`{bneron,hmenager,maufrais,njoly,letondal}@pasteur.fr`; [*]: presenting author

[2]: UMR-S 726, RPBS, Université Paris Diderot-Paris 7 75205 PARIS CEDEX 13, France
`pierre.tuffery@univ-paris-diderot.fr`
website: `http://bioweb2.pasteur.fr/projects/mobyle/`,
downloads: `ftp://ftp.pasteur.fr/pub/gensoft/projects/mobyle/`

Mobyle is the successor of Pise, a system that provides a web environment to define and execute bioinformatics analysis software. The design of the new system is based on the need to extend functionalities towards service interoperability and data integration, while optimizing the usability of the interface for biologists. Based on extensive user studies, we developed the end-user interface as a Web Portal that provides a **global and integrated view** of all the elements needed to perform analyses. The system itself is based on a set of program descriptions that automate their use, from the construction of the web interface up to the software execution. These descriptions also include a syntactic and semantic description of the processed biological data, which is the technical basis for the advanced features such as:

- **Data reusability**: the preservation of user data and parameters facilitates the reuse of input values or results between different programs.
- **Automatic data validation and format conversion**: the description of the expected data and their format allows the system to verify and convert input values if necessary.
- **Service discovery and workflow authoring assistance**: services are provided through a searchable menu; furthermore, data type compatibility mechanisms between results and potential program inputs let users either interactively pipe tasks or build complete workflows to run them.

These various features have been implemented within an architecture that consists of:

- **The Mobyle server**, based on a set of python modules, handles the various aspects of job, data and session managements. The most critical element, job submission, includes: various parameter validations, command line building, and the interaction with an execution system.
- **The Web portal** provides a unified access to the system. We use Ajax to coordinate the different types of information that need to be available and to enable the user to explore many functionalities in a single page.
- **The Storage System**, this includes job submissions, as well as the most important data of a user's workspace. The details of the job execution ensure its traceability, while the semantic tagging of the data improve their reusability.

Future work is planned to provide additional functionalities:

- A unified access to multiple mobyle servers, from one portal: **the Mobyle Grid**. This will facilitate the interoperation (chaining, data reuse) of distributed resources,
- The possibility to build and execute **Workflows** composed of Mobyle jobs from the Web Portal,
- The automated publication of mobyle programs as **Web Services** (BioMoby, SOAP). Some work has already been done in that direction with the PlayMOBY project[1].

Written in python, Mobyle should be compatible with any Unix-like system, requires an Apache web server, and has been tested with various job queueing systems (local batch, Torque/PBS, SGE). It is distributed under the terms of the GNU GPLv2. Two servers already provide access to the system: `http://mobyle.pasteur.fr` and `http://bioserv.rpbs.jussieu.fr/cgi-bin/MobylePortal/portal.py`.

[1]PlayMOBY is an independent project from the INRA LIPM bioinformatics team. You can find more information about it on `http://lipm-bioinfo.toulouse.inra.fr/biomoby/playmoby/`

InterMine - open source data warehouse and query interface

Richard Smith[12], *Sergio Contrino*[1], *Hilde Janssens*[1], *Jakub Kulaviak*[1], *Rachel Lyne*[1], *Kim Rutherford*[1], *Julie Sullivan*[1], *Dan Tomlinson*[1], *Matthew Wakeling*[1], *Xavier Watkins*[1] *and Gos Micklem*[1].

[1]*Systems Biology Centre, University of Cambridge, UK. info@flymine.org*
[2]*corresponding author: richard@flymine.org*

InterMine (www.intermine.org) is an open-source system for building query-optimised data warehouses. It supports data integration from standard biological formats and makes it easy to add your own data. A sophisticated web application provides flexible query access for any data model. Queries can be run via RESTful web services with results returned as HTML to embed in other web pages – promoting data re-use and mash-ups.

InterMine was developed to enable FlyMine (www.flymine.org) and is now used in other projects, including the $57m modENCODE project. InterMine is written in Java, all code is freely available under the LGPL license (see www.intermine.org/wiki/SVNCheckout).

Data Integration

InterMine makes it easy to integrate multiple data sources into a central data warehouse. It has a core data model based on the sequence ontology and supports many biological data formats. The object-based data model is defined as XML from which a database schema and Java classes are automatically generated. It is simple to extend the data model and integrate your own data. Supported formats include GFF3, FASTA, Chado, GO gene association, UniProt XML, PSI XML (protein interactions), PDB XML and Ensembl.

Web Application

A web application provides flexible query access to the data warehouse. The interface allows users to create custom queries, use template queries - web forms to run 'canned' queries, upload and operate on lists of data and analyse lists with interactive graphical or statistical 'widgets'. Queries and lists can be saved in a MyMine account.

The web application works 'out of the box' with any data model and can be heavily customised. Much of the presentation can be controlled by non-programmers: an admin user can publish new template queries and lists at any time and can change report pages by tagging.

Web Services

Custom queries, template queries and lists are available via RESTful web services. Results can be returned as XML, tab or comma separated formats. It is also possible to fetch HTML to include features of InterMine in other web sites.

All template queries have an 'embed this query' link which provides a URL to include results in report pages of another web interface. This feature is designed to promote the 'mash-up' model – other sites can fetch information related to the data they present without the need for local data integration. Results will change as the InterMine database is updated, avoiding maintenance overhead or the inclusion of stale data.

InterMine and FlyMine are actively developed by a team of seven people at the University of Cambridge, UK. The project is funded by the Wellcome Trust.

Bridging Existing Bioinformatics Resources Into the Semantic Web

Paul M.K. Gordon (gordonp@ucalgary.ca) & Christoph W. Sensen, University of Calgary

We have identified the need for, and built, client-side software ("Seahawk") in Bioinformatics that facilitates the *ad hoc* composition of Semantic Web Services (SWS) by biologists. Through user studies, we have also identified the unavailability of existing services from trusted sources as a barrier to adoption of SWS by biologists. We explore how to make the range of tools biologists already use conform to an established SWS protocol in the Life Sciences (BioMoby). We attack the problem of wrapping existing software with a semantic layer from three perspectives: 1) legacy data format conversion (both to and from), 2) semantic annotation of software interfaces and 3) end-user programming/wrapping for widespread adoption.

For service provision, we present a suite of four SWS wrapping tools that build on one another to cover a wide spectrum of semantic wrapping scenarios, namely 1) Java-language servlet annotation, 2) command-line bioinformatics tool annotation, 3) WSDL Web Services annotation, and 4) HTML/CGI form annotation (in progress at time of writing). Guiding principles behind the design of these tools are to distribute the wrapping work appropriately between those who are most motivated to use them, and those who can specify the annotations best. Code re-use is also strongly emphasized.

For service usage and basic workflow construction, we present the applet Seahawk, which provides a hypertext interface to BioMoby services, and workflow-by-example capabilities (reusable in the Taverna workflow engine) that provide a gentle learning curve for large-scale analysis by biologists without the assistance of a bioinformatician.

The software ("Daggoo") and its source code can be found on the Java developer's homepage of the BioMoby Web site (www.biomoby.org). Like all BioMoby repository code, tools mentioned are available under the terms of the Perl Artistic License.

Soaplab2: more reliable Sesame door to bioinformatics programs

Senger M.[1, 2] (*), Rice P. [1](pmr@ebi.ac.uk), Bleasby A. [1], Oinn T. [1], Uludag M. [1]
[1] European Bioinformatics Institute, [2] Consultative Group on International Agricultural Research

Soaplab2 is a refactoring and enhancement of the Soaplab Web Services framework for command-line bioinformatics programs. Enhancements include the removal of legacy layers and addition of memory management components that make Soaplab2 servers more reliable than before. While the new Spinet web client opens Soaplab2 services to a larger audience, support for the document/literal wrapped protocol makes Soaplab2 services more interoperable.

Soaplab allows service providers to make their command-line programs Web Services accessible based on metadata descriptions of the programs, without needing any programming effort in most cases. It uses a generic interface that makes it possible to use the same interface when accessing any services disregarding their implementation details. The interface includes methods to find an available service, discover what inputs it requires and what outputs it produces, to start it and to obtain results [1].

Although Soaplab opened the *Sesame door* to bioinformatics programs, the necessity of running legacy Applap CORBA servers in parallel was a maintenance headache for service providers. During refactoring this legacy layer was removed and the logic for handling management of jobs moved into Soaplab2 core libraries. In order to make Soaplab2 servers more reliable, new memory management components were implemented that periodically check for completed jobs and silent services, and returns their memory usage back to JVM. Similarly, hanging jobs are terminated and deleted based on a configurable timeout value.

Other enhancements for service providers include refactored build/install/deploy tasks that now use scripting power of *ant* together with dependency management power of *maven*. In order to make Soaplab2 platform-independent, some of the modules previously written in Perl were rewritten using Java. The new batch-client module allows defining test suites through Java configuration files and can be used to start concurrent test requests. Soaplab2 also includes built in support for EMBOSS programs and a predefined test suite for testing EMBOSS services.

One important new feature in Soaplab2 is its AJAX based new web interface, Spinet, which allows users to select a service, to specify its inputs in a usual HTML form, start the service, and to display its results. Spinet can be used from any modern web browser and comes with no extra cost to service providers.

Soaplab2 has a richer client library and a richer set of ready to use client scripts based on this library. Its architecture now has an extensible protocol layer. Support for the document/literal-wrapped protocol is already implemented for better interoperability with standard Web Services client libraries. The RPC/encoded protocol is still supported for backward compatibility. Taverna Soaplab plugin was updated by replacing Axis library calls with the Soaplab2 client library calls and is now able to communicate using both protocols.

References:
1. Senger M., Rice P., Oinn T., “Soaplab - a unified Sesame door to analysis tools“, Proceedings, UK e-Science- All Hands Meeting 2003, p. 509-513, 2003

Website: http://soaplab.sourceforge.net/soaplab2/

(*) Development of Soaplab2 is driven by Martin Senger, original Soaplab author

BeeSpace: An Interactive System for Functional Analysis

Barry Sanders, David Arcoleo, Bruce Schatz
Institute for Genomic Biology, University of Illinois at Urbana-Champaign

Presenter Email: sandersb@uiuc.edu
Project URL: http://www.beespace.uiuc.edu/
Download URL: http://arcoleo-asus.igb.uiuc.edu/~arcoleo/BeeSpace-Download/

BeeSpace is a National Science Foundation-funded project whose five-year mission is to develop an open bioinformatics resource supporting functional analysis of social behavior, leveraging the honey bee research community. We have built a software system that can support automatic curation of genome databases, using text mining of biomedical literature. We will briefly discuss the features of BeeSpace Navigator v4.0, and outline our proposed technique for integrating Navigator functionality within the GMOD framework.

BeeSpace Navigator was developed using open source software and open frameworks. It is deployed as a web application. The fourth major version (v4.0) has a new AJAX user interface, built on the open source Ext JS toolkit. It is currently under development, with anticipated late-2008 release. The new metadata rich drag-and-drop AJAX web interface has been redesigned from the ground up to enable the biologist to search and sort vast quantities of relevant documents quickly and efficiently using a familiar format with powerful controls.

In addition to the new user interface, we will be publishing key BeeSpace functionality as web services, which can be easily integrated into GMOD or other software tools, similar to the way in which YouTube videos are integrated into web pages – via URL. Through these services, researchers will be able to execute searches on biology document collections, cluster search results by concept and quickly navigate through result sets of semantically related documents (called “spaces”) without leaving familiar software tools.

The BeeSpace system enables users to create their own collections, which are semantically indexed into concept spaces. Community annotation is supported by enabling users to share spaces and concepts with others. This collaborative feature produces high efficiency for teams of researchers working toward similar goals by allowing them to easily and powerfully share their clustered search results.

BeeSpace source code is distributed under the GNU Public License v2.

The BioDAS project - *http://www.biodas.org/*

Andrew Jenkinson[1,A], *Ewan Birney*[1], *Hagen Blankenburg*[3], *Robert Finn*[2], *Henning Hermjakob*[1], *Tim Hubbard*[2], *Felix Kokocinski*[2], *Eugene Kulesha*[1], *Gabrielle Reeves*[1] *and Andreas Prlic*[2]

[1] European Bioinformatics Institute, Cambridge, CB10 1SD, UK
[2] Wellcome Trust Sanger Institute, Cambridge, CB10 1SA, UK
[3] Max Planck Institute for Informatics, Saarbrücken, 66123, Germany
[A] andy.jenkinson@ebi.ac.uk

The Distributed Annotation System (DAS) is an open protocol outlining a mechanism for the dynamic integration of biological data from geographically diverse sources. It was designed as a framework defining the communication protocol, data formats and visualisation standards for distributed genomic sequence annotations. The BioDAS community maintains client and server libraries implementing the DAS protocol in both Java and Perl programming languages. All are distributed with open source licenses (GPL/L-GPL). Various full client implementations, including Ensembl, Dasty and SPICE, are also available on an open source basis.

The core premise of DAS is that rather than aggregate data from multiple sites into a centralised database, data remains distributed. Within this basic strategy, data providers retain control over access and there is no requirement for synchronisation. However, proprietary interfaces restrict the usability of the data. DAS attempts to alleviate this concern by providing a common interface and format for obtaining and displaying annotations. The principal strength of DAS in this endeavour is its “dumb server, clever client” architecture. Since data providers are more numerous than visualisation clients, DAS aims to be simple for data providers to implement and has been widely adopted for this reason. DAS sees use both by public databases wishing to expose their data, and by groups of researchers wishing to share data between themselves.

In recent years, DAS has been extended to facilitate the communication of additional data types beyond genomic sequences and their annotations. There is now extensive use of DAS in communicating protein sequences and annotations, as well as 3D structures. Similarly, pairwise and multiple alignments are supported, with the inclusion of a paging option that is crucial for very large alignments such as Pfam protein family alignments. Latest developments include the addition of flexible support for molecular interactions, DASMI, as well as the introduction of the Protein Feature Ontology for standardisation of protein annotation types. Both have been adopted by the BioSapiens network, a collaboration of 25 institutions based in 14 countries throughout Europe.

DAS also remains actively supported in the genomic community, with the number of sources steadily increasing. Ensembl is committed to the use of DAS for federating data in its genome browser, and a number of high profile projects such as the collaborative ENCODE project have adopted DAS. In addition, the nature of contemporary functional genomics data has also driven further development of DAS. The high volume and density of data generated by sequencing or microarray experiments creates a technical challenge, in that researchers often wish to visualise huge volumes of data from a remote source simultaneously. DAS has introduced measures to improve efficiency in response to this requirement, and also additional visualisation options such as histograms and colour gradients. A new extended version of the DAS specification, 1.53E, encompasses all extensions made to the protocol to date.

The continuous growth in use of DAS has necessitated the development of a registry of public servers. The DAS Registry (http://www.dasregistry.org/), a discovery broker between servers and clients, is now an integral part of client applications - allowing client software to filter applicable servers and end users to distinguish and discover the various data available. As a result of this inclusion, DAS can be considered in principle a Service Oriented Architecture (SOA), having the three required components of service providers (servers), service requestors (clients) and service broker (registry). As evidence of the continued growth of DAS, registered DAS sources (i.e. only those publicly advertised) number around 400. The total number in use worldwide is far in excess of this.

DAS/2 emerged several years ago following a bifurcation of the BioDAS project with the aim of adding and improving features. At present and following considerable development to the specification, DAS/2 has some server and client implementations and offers features not available in DAS (e.g. writeback, for read-write operations). However, other features of DAS are not supported in DAS/2 (e.g. interactions), and no provision is made for backwards compatibility. Since future funding for DAS/2 has not been secured, we intend to explore possibilities for reconciling the two projects.

E-Cell 3D: 3-Dimensional Visualization of Dynamic Cell Simulation

Kazuharu Arakawa (gaou@sfc.keio.ac.jp)

Institute for Advanced Biosciences, Keio University 252-8520 Japan

The E-Cell Project is an international research project aiming to model and reconstruct biological phenomena *in silico*, and to develop necessary theoretical supports, technologies, and software platforms to allow precise whole cell simulation. Cell simulation realizes theoretical experiments *in silico*, in order to enhance our understandings of dynamic cellular activities through systems biology approaches. E-Cell Simulation Environment version 3 (E-Cell SE 3) is a potential platform for this purpose, which allows multi-algorithm and multi-timescale simulation that is scriptable with Python programming language.

E-Cell 3D is a novel 3D visualization interface for the simulation results of E-Cell SE3. Given a model, this system automatically lays out all molecular entities (corresponding to “species” in SBML) and displays all connections between them (corresponding to “reactions” in SBML) in 3-D space. Then the time-course changes in concentration and reaction rates are animated according to the simulation results, where molecular concentrations are represented by the size and brightness (changing from blue to yellow) of the nodes, and the flux is represented by the speed of particles traveling along the edges that connect the molecular entities. Since every component of a model is present in this visualization, system biologists can capture the dynamic workings of the entire system at a glance. Moreover, when there is large amount of flux in or out of a node, the coordinates of the node are altered, allowing rapid identification of controlling factors within the cellular network. In highly dynamical models such as the circadian clock, we can clearly observe the synchronization of oscillations within the network, and identify the controlling molecules of feedback oscillators. Although still experimental, this type of visualization approach may complement the use of graphs for the interpretation of modeling and simulation. E-Cell 3D is developed using Quartz Composer (graphical programming environment for 3D animations available on MacOS X) and OpenGL API.

License: GNU General Public License v.2

Software and source code: http://ecell3d.iab.keio.ac.jp

The European Molecular Biology Open Software Suite (EMBOSS)

Peter Rice (pmr@ebi.ac.uk), Alan Bleasby, Jon Ison, Mahmut Uludag
European Bioinformatics Institute, Hinxton, Cambridge, CB10 1SD, United Kingdom.

The European Molecular Biology Open Software Suite (EMBOSS) is a mature package of software tools developed for the molecular biology community. It includes a comprehensive set of applications for molecular sequence analysis and other tasks and integrates popular third-party software packages under a consistent interface. EMBOSS includes extensive C programming libraries and is a platform to develop and release software in the true open source spirit.

A major new stable version is released each year and the current source code tree can be downloaded via CVS. All code is open source and licensed for use by everyone under the GNU Software licenses (GPL with LGPL library code).

EMBOSS is free (as in "free beer"): you can download it free of charge.
EMBOSS is free (as in "free speech"): you can do whatever you want with the code, provided you preserve that same freedom for others. Submitted code from outside contributors is welcome.

There have been many thousands of downloads including site-wide installations all over the world since the project inception. EMBOSS is used extensively in production environments reflecting its mature status and has been incorporated into many web-based, standalone graphical and workflow interfaces including wEMBOSS, EMBOSS Explorer, JEMBOSS, SoapLab, Pise, SRS, Taverna and several commercial workflow packages.

The demonstration will highlight the well established key features and also new features in EMBOSS 6.0 in the following areas :

- Stable open source licensing
- Documentation and support for developers and system administrators
- Commandline validation
- UNIX commandline, Web, GUI and web service interfaces
- Data in any common format
- Unlimited sequence length
- Simple site configuration and customization
- Support for local databases
- Remote data servers
- Availability and distribution
- Quality assurance
- C programming library development
- Functionality and Integration

Project home page: http://emboss.sourceforge.net/
Release download site: ftp://emboss.open-bio.org/pub/EMBOSS/
Anonymous CVS server: http://www.open-bio.org/wiki/SourceCode

BioJava Project Update

Holland, R.C.G.[1], Down, T.[2], Pocock, M.[3], Prlic, A.[4], Huen D.[5], James K.[4], Foisy, S.[6], Dräger, A.[7], Yates, A.[8], Heuer M.[9], and Schreiber, M.J.[10]

[1] Eagle Genomics Ltd., Cambridge, UK [2] Wellcome Trust/Cancer Research UK Gurdon Institute, Cambridge CB2 1QN, UK [3] University Newcaste Upon Tyne, Newcastle Upon Tyne, NE1 7RU, UK [4] Wellcome Trust Sanger Institute, Genome Campus, Hinxton, Cambridgeshire CB10 1SA, UK [5] Department of Genetics, University of Cambridge, Cambridge CB2 3EH, UK [6] Laboratory in Genetics and Genomic Medicine of Inflammation, Montreal Heart Institute, Montreal, Québec, Canada. [7] Eberhard Karls University Tübingen, Center for Bioinformatics (ZBIT), Germany [8] European Bioinformatics Institute (EMBL-EBI), Genome Campus, Hinxton, Cambridgeshire CB10 1SD, UK [9] Harbinger Partners, Inc., St. Paul, Minnesota, USA [10] Novartis Institute for Tropical Diseases, 10 Biopolis Road, Chromos #05-01, Singapore 138670

BioJava was conceived in 1999 by Thomas Down and Matthew Pocock as an API to simplify bioinformatics software development using Java (Pocock et al., 2000). It has since then evolved to become a fully-featured framework with modules for performing many common bioinformatics tasks.

As a free and open-source project, BioJava is developed by volunteers coordinated by the Open Bioinformatics Foundation (O|B|F, http://open-bio.org/) and is one of several Bio* toolkits (Mangalam, 2002). Over the past eight years, the BioJava has brought together nearly fifty different code contributors, hundreds of mailing list subscribers, and several wiki contributors. All code and related documentation is distributed under version 2.1 of the GNU Lesser General Public License (LGPL) license (Free Software Foundation, Inc., 1999). All wiki documentation is made available online under version 1.2 of the GNU Free Documentation License (Free Software Foundation, Inc., 2000).

BioJava has been used in a number of real-world applications, including Bioclipse (Spjuth et al., 2007), BioWeka (Gewehr et al., 2007), Cytoscape (Shannon et al., 2003), and Taverna (Oinn et al., 2004), and has been referenced in over fifty published studies. A list of these can be found on the BioJava website.

The latest BioJava release (version 1.6, released on 13 Apr 2008) offers more functionality and stability over the previous official releases. The phylogenomics package was improved and expanded by our 2007 Google Summer of Code (GSOC'07) student Boh-Yun Lee. It now contains fully-functional Nexus and Phylip parsers, and tools for calculating UPGMA and Neighbour Joining, Jukes-Kantor and Kimura Two Parameter, and MP. The PDB file parser was improved by Jules Jacobsen for better dealing with PDB header records. Andreas Dräger provided several patches for improving the genetic algorithm packages. The version 1.6 release also contains numerous bug fixes and documentation improvements.

The BioJava website is http://biojava.org/. The version 1.6 release can be downloaded from http://biojava.org/wiki/BioJava:Download.

References

Free Software Foundation, Inc. (1999) GNU Lesser General Public License, version 2.1, http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html, accessed 10 May 2008.

Free Software Foundation, Inc. (2000) GNU Free Documentation License, version 1.2, http://www.gnu.org/licenses/fdl-1.2.html, accessed 10 May 2008.

Gewehr JE, Szugat M, Zimmer R. (2007) BioWeka—extending the Weka framework for bioinformatics Bioinformatics 2007 23(5):651-653.

Mangalam H. (2002) The Bio* toolkits – a brief overview. Brief Bioinform., 3, 396-302.

Oinn T, Addis M, Ferris J, Marvin D, Greenwood M, Carver T, Pocock MR, Wipat A, Li P. (2004) Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics, 20, 3045–3054.

Pocock M, Down T, Hubbard T. (2000) BioJava: Open Source Components for Bioinformatics. ACM SIGBIO Newsletter 20(2), 10-12.

Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T. (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome Research 2003 Nov; 13(11):2498-504.

Spjuth O, Helmus T, Willighagen EL, Kuhn S, Eklund M, Wagener J, Murray-Rust P, Steinbeck C, Wikberg JE. (2007) Bioclipse: an open source workbench for chemo- and bioinformatics. BMC Bioinformatics. 2007 Feb 22;8:59.

Use the Make Utility for the Maintenance of Complex Bioinformatics Pipelines

Justin Wilson (wilsonjr@umich.edu), Manhong Dai, Stanley Watson and Fan Meng
Psychiatry Department and Molecular and Behavioral Neuroscience Institute, U of Michigan, US

Complex bioinformtics pipelines usually integrate data from multiple sources using different data processing methods. For example, the SNP Function Portal developed in our group integrates SNP function annotation data from several dozen public data sources as well as data derived from multiple in-house data annotation procedures. Updating the SNP Function Portal is a complex and time-consuming process. Existing tools like cron, wget, shell scripts and custom parsers are adequate for projects requiring a small number of datasets with simple dependencies. In order to efficiently manage complex data download, processing and integration procedures involved in our SNP Function Portal data integration project, we decided to use the classic Make utility to augment existing maintenance techniques with dependency tracking. We also devised methods that allow the checking of table dependency and the management of jobs on computer cluster with the Make utility.

As described in the make utility documentation, `make' is not limited to programs. One can use it to describe any task where some files must be updated automatically from others whenever the others change. A Makefile is a dependency tree in which a target often depends on several prerequisite files. Only when a target file is older than any of its prerequisite files, the action that generates the target file would be triggered. This is very useful for maintaining a complicated bioinformatics pipeline involving many data sources and processing methods. Make can figure out which targets need to be updated based on which source files have changed, and remake them all and only, in a proper order automatically.
Make enable easier collaboration. Each collaborator only needs to implement his/her own dependency tree without knowing any details of others. The interface between each other is just a few targets. For example, in our SNP function annotation integration project, one member of our team is responsible for writing parsers to extract information from dbSNP and export results in predefined format, such as Oracle tables and a Fasta file containing SNP sequences from different organisms. Other members of the team or outside collaborators only need to put the results needed in his/her prerequisite list and there is no need to worry about the details of dbSNP extraction process.

Make has many helpful options. A good example is '-j', which specifies the number of jobs to run simultaneously. User can always adjust this number based on computer's work load, the target making order is always proper.

Since bioinformatics data integration often involves multiple tables with complex dependencies, it is essential to have the capability to check table dependency for a pipeline management solution. Although the Make utility cannot check table's dependency in database, we get around this shortcoming by creating a tag file for each database table to enable Make-based dependency tracking.

Another challenge is there is often a need to submit large-scale data processing jobs to a computer cluster. We also figured out a way to let Make communicate with cluster automatically. We also use command like "while qstat `cat $<`; do sleep 60; done" to make local processing that need utilize results from a prerequisite job running on a cluster will not start till the cluster job is done.

In summary, we believe the classic Make utility provides significant advantages for the management of complex bioinformatics work flows. Combined with methods for checking table dependency and computer cluster job management, Makefiles greatly enhances existing solutions in terms of flexibility, power, manageability, and automation. Sample codes for dbSNP data extraction, Oracle table dependency checking and cluster job management using the Make utility can be downloaded from the SNP Function Portal Data Integration Project website (http://brainarray.mbni.med.umich.edu/Brainarray/Database/SearchSNP/integrate.aspx). These codes use the GNU General Public License.

Interoperability with BioMoby - Past, Present, and Future

Ben Vander Valk, Luke McCarthy, Mark Wilkinson
Heart + Lung Research Institute, St. Paul's Hospital,
Center for High Throughput Biology, University of British Columbia,
Vancouver, British Columbia, Canada
markw@illuminae.com

In 2001 the term “Deep Web” was coined to describe the information on the Web that was not accessible to search engines – for example, data from databases or analytical tools that were hidden behind Web Forms; it was estimated at the time that the Deep Web contained upwards of 100-fold more data than the Web itself. In that same year, the BioMoby project was founded with the goal of establishing a framework for syntactic and semantic interoperability between Deep Web-style bioinformatics resources. By mid-2002, a prototype interoperability architecture and supporting codebase had been developed that had four main components: an ontology of bioinformatics data-types (“Objects”), an ontology of bioinformatics identifier-types (“Namespaces”), an ontology describing various computational operations (“Services”), and a novel Web Service Registry API (“Moby Central”) that was aware of these three ontologies and was capable of semantic brokering between Web Services providers and the clients who required these Services.

BioMoby has been moderately successful in providing an interoperability platform within discreet communities of users in the absence of competing standards; however it failed to be widely adopted throughout the life sciences community. In the interim, novel technologies have emerged from the World Wide Web consortium's Semantic Web activity that achieve many of the same goals as the BioMoby framework does, but have the broader endorsement of the W3C itself. Nevertheless, the “niche” that BioMoby fills – the intersection of semantically and syntactically-standardized data-types and the ability to discover and invoke appropriate Web Services based on the type and content of these data elements – remains unfilled by current Semantic Web activities within the Health Care and Life Science domains. As such, it is now prudent for the BioMoby project to examine its past successes and failures, learn from these, and re-invent itself. The next generation of Moby will use these emergent Semantic Web standards to represent the Deep Web, and thereby parallel their utilization by the wider life sciences community to represent traditional Web data.

To this end, I introduce here the architecture of the Moby 2.0 interoperability proposal, and its first prototype implementation in the CardioSHARE (Cardiovascular Semantic Health and Research Environment) project. Moby 2.0 is a proposed extension to the SPARQL Protocol and RDF Query Language in which predicates in a SPARQL query are mapped onto Web Service categories (effectively the BioMoby Service Ontology), and Web Service registry queries and executions are achieved in-line with SPARQL query resolution. Thus data is generated dynamically on an as-needed basis in response to specific queries, prior to the resolution of those queries. This is further extended by the CardioSHARE architecture such that concepts (subjects/objects) in the SPARQL query can refer to concepts defined in OWL ontologies, where the necessary and sufficient conditions defining those concepts are evaluated through a similar process of Web Service discovery and execution. In this way, complex data-types referred to in the query, but not present in the initial data-store, can be assembled by a combination of DL-reasoning and Web Service invocation prior to query resolution. If successful, this will simplify the process of discovery and query-answering, and will significantly change the way we interact with data both in our local databases, as well as on the Web.

Websites: http://biomoby.org; http://cardioshare.icapture.ubc.ca
License: The Artistic License; http://www.perl.com/pub/a/language/misc/Artistic.html

Ergatis: A web-based bioinformatics pipeline management and collaborative development system

Orvis J[1*], Crabtree J[2], Galens K[1], Inman JM[2], Lee E[4], Riley D[1], Sundaram J[2], Whitty B[3] , White O[1], Wortman JR[1], Angiuoli SV[1]

[1] Institute for Genome Sciences, University of Maryland School of Medicine. [2] J. Craig Venter Institute, Rockville MD. [3] Michigan State University, East Lansing MI. [4] Lawrence Berkeley National Laboratory
* corresponding author (jorvis@som.umaryland.edu)

Ergatis is a web-based utility that is used to create, run, and monitor reusable computational analysis pipelines. It contains pre-built components for common bioinformatics analysis tasks. These components can be arranged graphically to form highly-configurable pipelines. Each analysis component supports multiple output formats, including the Bioinformatic Sequence Markup Language (BSML). The current implementation includes support for data loading into project databases following the CHADO schema, a highly normalized, community-supported schema for storage of biological annotation data.

Ergatis uses the Workflow Engine to process its work on a compute grid. Workflow provides an XML language and processing engine for specifying the steps of a computational pipeline. It provides detailed execution status and logging for process auditing, facilitates error recovery from point of failure, and is highly scalable with support for distributed computing environments. The XML format employed enables commands to be run serially, in parallel, and in any combination or nesting level.

This open-source framework has been employed in the annotation of several large, eukaryotic organisms, including Aedes aegypti and Trichomonas vaginalis as well as published comparative genome analyses and primary processing of the CAMERA metagenomic datasets.

Recent development has focused on better code portability and ease of collaborative pipeline development. An Ubuntu-based VMware image is available with Ergatis and all dependencies pre-installed and configured. A JeOS distribution has also been ported for use on Amazon's EC2 cloud computing service and work is ongoing to make it an available resource on TeraGrid.

Ergatis is distributed under the Artistic License at available at http://ergatis.sf.net.

Chipster – user friendly DNA microarray analysis software

Aleksi Kallio, Jarno Tuimala, Taavi Hupponen, Petri Klemelä, Eija Korpelainen
(Aleksi.Kallio@csc.fi, Eija.Korpelainen@csc.fi)
The Finnish IT Center for Science CSC

Chipster (http://chipster.csc.fi) is a user friendly graphical software for DNA microarray data analysis. It offers a comprehensive collection of up-to-date analysis methods, including those from the R/Bioconductor, and it supports all the major DNA microarray platforms. Preprocessing, statistical tests, clustering, and annotation are complemented with e.g. linear (mixed) models, bootstrapping hierarchical clustering results, and promoter analysis tools. In addition to showing images produced by the R/Bioconductor, Chipster can also produce interactive visualizations for hierarchical and SOM clustering, 2D and 3D scatter plots, histograms and time series plots.

Chipster is a Java based system where the graphical interface runs on user's computer and accesses server components to perform analysis runs and connect to external data sources. Chipster's user interface and the supporting infrastructure has been developed iteratively together with users from the start. Analyzing user feedback lead us to design a workflow system which functions under the data oriented user interface: as the user uses the analysis tools all the analysis steps are recorded as workflows. The workflows can be later rerun with different datasets, and they can be extended as they are simple Java/BeanShell scripts. The individual analysis tools are functionally quite large, because this results in simpler workflows which was considered as a major usability gain by the users.

Chipster is published under GPL (see http://chipster.sourceforge.net).

myExperiment: social software for sharing workflows

[1]David De Roure, [2]Carole Goble, [2]Jiten Bhagat, [1]Don Cruickshank, [3]Marco Roos
University of Southampton, UK; University of Manchester, UK; [3]University of Amsterdam, The Netherlands

The ^{my}Grid project (http://www.mygrid.org.uk) develops the popular Taverna workflow management system, now used throughout the world by 350+ organisations for a whole range of Life Science problems. An open source initiative from the outset, Taverna's core development activity is supported by the Open Middleware Infrastructure Institute UK (http://www.omii.ac.uk), so that e-scientists can rely upon it as part of their regular collection of tools.

Workflow design is challenging and labour-intensive. Reusing a body of prior designs through catalogues is highly desirable. Reuse is a particular challenge when scientists are outside a predefined Virtual Organisation or enterprise. These are individuals or small groups, decoupled from each other and acting independently, who are seeking workflows that cover processes outside their expertise from a common pool of components, especially when workflows are shared across discipline boundaries and when inexperienced scientists need to leverage the expertise of others. As Taverna's popularity increased we observed a workflow exchange activity emerging amongst our users and a strong desire for a place were workflows could be collected and opened to peer review.

myExperiment (http://myexperiment.org) is an initiative from the ^{my}Grid project to create a Virtual Research Environment which makes it easier for workflow workers to gossip about and exchange workflows, regardless of the workflow system – Taverna, Kepler, Triana, BPEL etc. Scientists rarely care about the workflow engine they use: they typically care about the workflow itself, its function and the services it uses. Rather than just making workflows available for sharing, myExperiment actually facilitates and encourages that – it takes a step beyond existing workflow repositories by crossing project, community and product boundaries, emphasising social networking around the workflows, providing gateways to other environments and forming the foundation of a personal or laboratory workbench.

myExperiment enables scientists to *discover, reuse and repurpose workflows*, and to *enact them* from a web page using a remote enactment service. In contrast to social websites, it has detailed support for ownership, attribution, licensing, sharing and permissions – meeting the particular needs of the scientist. More generally, the myExperiment concept is about sharing digital objects which include data, results, provenance information, tags, associated documentation, etc. These individual items can be collected together to form *research objects*, for example to record an experiment. myExperiment *aids reuse* because workflows can be discovered not just by what they do but based on how they are used by the community, with tags and reviews adding to the 'collective intelligence'.

myExperiment is an *open source codebase* (released under BSD licence), and individuals and laboratories are free to install their own myExperiment instances; they can then link them up using a federation model if and as they wish. The system is implemented using the Ruby on Rails open source Web application framework using an agile development method. The public site (www.myexperiment.org) was made available in November 2007 and over the period January-April 2008 the site gained over 750 registered users, with many others accessing public content without needing an account; visits came from 78 countries. The site has been well received by the community, carrying over a third of all publicly available Taverna workflows.

While the public site provides a service for those who do not already have sharing and collaboration mechanisms in place, we also expose myExperiment functionality through simple RESTful APIs so that it can be accessed through existing interfaces, including wikis and web pages. This also enables the creation of other interfaces such as Google Gadgets, myExperiment add-ons for sites such as Facebook and functionality mashups over myExperiment. Much of our planned effort is in linking up with other web tools and platforms.

Although myExperiment focused initially on workflows, its underlying architecture is neutral about the objects it manages. Web services and workflows are inter-twined; in Taverna, workflows commonly call remote web services. In the same way that workflows can be catalogued, tagged and curated by the community, so should the growing band of web services in bioinformatics. With this in mind, the ^{my}Grid project, along with the EBI, has just launched (May 2008) the BioCatalogue project. This project builds on myExperiment to create a fully curated catalogue of Web services covering their functional, operational, usage and provenance metadata. By analysing the designs and patterns of services in workflows we can also improve the curation of services.

BioHDF : Open binary file formats for large scale data management

Todd Smith(1), Christian Chilan (2), Rishi Sinha(3), Elena Pourmal(2), Mike Folk(2).

1. Geospiza, Inc. 100 West Harrison St. North Tower #330, Seattle WA 98119. 2. 1901 S. First St., Suite C-2 Champaign, IL 61820. 3. Microsoft Corporation, Redmond WA.

The first wave of Next Generation ("Next Gen") sequencing technologies are providing large numbers of laboratories with "Genome Center" kinds of throughput to make discoveries and develop new assays never before imagined. However, widespread adoption of Next Gen will be hindered because current bioinformatics programs do not scale; they are inefficient in data storage, processing, and memory utilization. The most popular programs typically copy and recopy data to new files many times during processing, require that all data be maintained in random access memory (RAM) when running, and cannot incrementally process data. To overcome these issues, fundamental changes in data management and processing are needed.

Geospiza and The HDF Group are collaborating to develop portable, scalable, bioinformatics technologies based on HDF5 (Hierarchical Data Format - http://www.hdfgroup.org). We call these extensible domain-specific data technologies "BioHDF." BioHDF will implement a data model that supports primary DNA sequence information (reads, quality values, meta data) and results from sequence assembly and variation detection algorithms. BioHDF will extend HDF5 data structures and library routines with new features (indexes, compression, graph layouts) to support the high performance data storage and computation requirements of Next Gen Sequencing. BioHDF will include APIs, software tools, and a viewer based on HDFView to enable its use in the bioinformatics and research communities. Using BioHDF, researchers will be able perform *de novo* sequencing, do resequencing-based SNP discovery, analyze genotyping data, and export datasets in formats ready for submission to key databases. As a programming environment, BioHDF can be easily extended to accept data from new data collection platforms and format data for interchange with many databases. BioHDF will be delivered to the research community as an open source technology.

In preliminary studies, HDF5's feasibility for managing large volumes and complex biological data was tested. The first test case looked at DNA sequencing-based SNP discovery. Through this study, HDF's strengths and data organization features (groups, sets, multidimensional arrays, transformations, linking objects, and general data storage for other binary data types and images) were evaluated to determine how well these features would handle SNP data. Other test cases were added to test the ability of HDF to handle extremely large datasets associated with HapMap data and chromosomal scale LD (Linkage Disequilibrium) calculations. Data from preliminary studies and new work with Next Gen sequence data will be presented.

Title: The Open HeliSphere™ project: True open source from the inventors of True Single Molecule Sequencing.
Author: Aaron Kitzmiller (akitzmiller@helicosbio.com)
License: GPL v2
Site: http://open.helicosbio.com

A recent publication describes the single-molecule sequencing-by-synthesis capabilities of Helicos' technology (Harris, et al. Single-Molecule DNA Sequencing of a Viral Genome Science 4 April 2008: 106-109), demonstrating for the first time the direct interrogation of unamplified strands of DNA to produce sequence information. To avoid the limitations on innovation and barriers to scientific discovery of closed instrument software, Helicos is opening its bioinformatics tool suite in a true open source project under a GPL v2 license. When fully launched, the project will feature:

- Tarball downloads of source code
- Mailing lists
- SVN checkout of trunk and release branches
- patch submission to the core code base
- bug tracking

Similar to the EMBASSY suite that complements the EMBOSS package, a second repository will be made available for tools and libraries wholly owned by users and potentially released under other licenses.

The project contains libraries and tools that support bioinformatic analysis beginning with the instrument-generated raw read container (SRF file) and extending through the creation of consensus alignment and summary reports. Components include:

- Core C/C++ and Perl libraries
- File manipulation tools for the Helicos SMS file format, including SRF and text conversion
- Alignment tools
- Tools for calculation of error rates and length distributions
- Complete Perl analysis pipelines
- Sample datasets

APOLLO: A SEQUENCE ANNOTATION EDITOR

Ed Lee[1], Suzanna Lewis[1], Nomi Harris[1], Mark Gibson[1], Steven Searle[2], Raymond Chetty[3]

[1] Berkeley Bioinformatics and Ontologies Project, Lawrence Berkeley National Laboratory, Berkeley, CA, USA
[2] Wellcome Trust Sanger Institute, Hinxton, Cambridge, UK
[3] The Arabidopsis Information Resource, Carnegie Institution of Washington, Stanford, CA, USA

The well-established inaccuracy of purely computational methods for annotating genome sequences necessitates an interactive tool to allow biological experts to refine these approximations by viewing and independently evaluating the data supporting each annotation. Apollo was developed to meet this need, enabling curators to inspect genome annotations closely and edit them. FlyBase biologists successfully used Apollo to annotate the Drosophila melanogaster genome and it is increasingly being used as a starting point for the development of customized annotation editing tools for other genome projects.

Apollo is an Open Source Java application that is easy to install and run on any platform. Apollo is highly configurable and has been adapted by external groups to make it better suit their individual needs. These adaptations range from adding new data types to the display configuration file, to writing new data loading modules to enable Apollo to read annotation data from proprietary databases.

Apollo is the annotation editor component of the Generic Model Organism Database (GMOD) project, which aims to provide a ready-to-use toolkit to genome centers.

After a long hiatus, development on Apollo has resumed and many new features and bug fixes have been implemented or are planned for the near future. Some of these include Generic Feature Format Version 3 (GFF3) support, improved Chado connectivity, undo, Sequence Ontology (SO) integration, and multiple sequence alignment integration to the exon detail editor.

The official Apollo website is: http://apollo.berkeleybop.org. Apollo is distributed under the Artistic License.

Contact: Ed Lee (elee@berkeleybop.org)

BioSQL Reloaded: 1.0 Release, PhyloDB Module, and Future Features

Authors: Hilmar Lapp(1), Richard Holland(2), Aaron Mackey(3), William Piel(4), Mark Schreiber(5)
1 National Evolutionary Synthesis Center (NESCent), Durham, NC 27705, U.S.A. (hlapp@nescent.org)
2 EMBL—European Bioinformatics Institute, Hinxton, Cambridge, CB10 1SD, United Kingdom
3 GlaxoSmithKline, Collegeville, PA 19426, U.S.A.
4 Peabody Museum of Natural History, Yale University, New Haven CT 06511, U.S.A.
5 Novartis Institute for Tropical Diseases, 138670 Singapore, Singapore

BioSQL (http://biosql.org) is a generic relational model for persistent storage of sequences, features, sequence and feature annotation, a reference taxonomy, and ontologies (or controlled vocabularies) in a way that is interoperable between the Bio* projects. While in its original incarnation (in 2001) conceived by Ewan Birney as a local relational store for GenBank, the project has since become a collaboration between the BioPerl, BioPython, BioJava, and BioRuby projects. The core schema of BioSQL has essentially been stable since November 2004, after it underwent substantial revisions at the 2002 and 2003 BioHackathons in Tucson, Cape Town, and Singapore. Here we report on a number of significant BioSQL developments that have taken place in the past 18 months.

Perhaps most notably, the 1.0 version of the core schema and supporting software was released in March 2008. While - intentionally - the 1.0 release does not introduce any structural changes to the model, for the first time of the project it defines a stable reference point that allows a roadmap for moving forward to be drawn. The 1.0 release was set in motion at the BioHackathon 2008 in Tokyo, which brought together a critical mass of leaders from several Bio* projects. Aside from starting the 1.0 release work, the concerted efforts at this event and those following it yielded for the first time a logo, a project wiki with information for both developers and users, a bug and feature request queue, and a language binding for BioRuby. The pre-release cleanup work, among many other things, also finally effected the change of the BioSQL license terms from the Perl-specific Artistic License to the widely used LGPL (Lesser GNU Public License). The future changes being charted at present range from better supporting generic object-relational mapping toolkits to versioning of sequence features, creating an audit trail, and supporting chimeric sequences.

Another significant development started more than a year ago at the Phyloinformatics Hackathon, which took place in December 2006 in Durham, North Carolina. At this event the BioSQL core schema was supplemented with a module, called PhyloDB, that extends the data types covered by BioSQL to phylogenetic trees (or networks). The module, which was considerably revised at the 2008 BioHackathon, follows the same design principles as the core schema, allowing arbitrary metadata attributes, typed by controlled vocabularies or ontologies, to be attached to nodes, branches, and trees. Tree nodes can be linked to sequences or taxa, and trees are scoped by a namespace similarly as databank entries. The module is accompanied by a number of common topological queries formulated in SQL. Furthermore, the PhyloDB module was the subject of a 2007 Google Summer of Code^{TM} project under NESCent as the mentoring organization. The project resulted in a collection of stand-alone Perl scripts to import, export, manipulate, and query phylogenetic trees from or to the file formats supported by the Bio::TreeIO modules, which are part of BioPerl.

Aside from the significance of the accomplishments themselves, they continue the consistent history of BioSQL development sprints connected to events fostering intensive face-to-face collaboration. It is evidence for how an inherently collaborative cross-project effort such as BioSQL thrives, and possibly even relies on environments that allow deep and instant collaboration between key stakeholders and developers.

EMBRACE – BioMart Developments and Future

Syed Haider[1], Benoit Ballester[1], Richard Holland[1], Damian Smedley[1], Peter Rice[1], Arek Kasprzyk[2]
[1]EMBL European Bioinformatics Institute, The Wellcome Trust Genome Campus, Hinxton, Cambridge, CB10 1SA, UK.
[2]Ontario Institute for Cancer Research, Toronto, Ontario, Canada, M5G 0A3.

The EMBRACE network of excellence aims to provide a standardized application programming interface (API) to a majority of the core biomolecular databases including Ensembl, UniProt, MSD, ArrayExpress, Wormbase and several other information resources hosted at but not limited to EMBRACE partner institutes. The project has significantly enhanced the usability of these resources by improving data presentation, web services, APIs and e-learning hence facilitating their use by 'wet-lab' biologists.

BioMart features a built-in query optimization engine and provides support for data federation between remote repositories without physical relocation and post processing. Each data source can be accessed through any of the following:

- Web service
- BioMart-DAS server
- Perl/Java API
- URL Access
- MartView (web based GUI)
- MartExplorer (Java desktop application)
- MQL (shell based Mart Query Language).

Substantial improvements to the BioMart web service and web interface have been carried out to simplify the interaction with the system. As a result the BioMart web service interface has now been adopted by a number of third party software packages such as Taverna, BioConductor-biomaRt, Galaxy and Bioclipse, adding new ways to manipulate the data. This includes, an *out of the box* BioMart-DAS server which enables lateral data-federation offering data/annotation sharing between disparate clients. For advanced users, complex scientific workflows are also possible over any of the mart repositories through Taverna's BioMart plugin.

The web service interface provided by the BioMart project has significantly reduced difficulties with data-integration. Optimization features included in the package have been particularly useful when working with large data repositories storing high throughput experiment data such as genomic sequence or microarray experiments. Consequently, EMBRACE-Grid has adopted BioMart as its primary data management system, aiming to offer in the near future a new, versatile platform for biological data analysis.

The process of the conversion of a data source into a BioMart compliant schema followed by its configuration and deployment is completely automated now using *martj* tools. Further developments to ease the deployment process, scalability, secure access and empowering the system with visualizations and analysis plugins are underway.

BioMart is completely Open Source, licensed under the LGPL, and freely available to anyone without restrictions.
www.biomart.org
www.embracegrid.info
Project Source code: *www.biomart.org/install-overview.html*

Biopython project update

Tiago Antao[1], Peter Cock[2]

In this talk we present the current status of the Biopython project. We start by giving a short overview of Biopython (presenting existing functionality) and useful software libraries for computational biology in the Python development 'ecology' (from plotting libraries capable of producing publication quality figures to numerical libraries, among others). We then focus on features developed since BOSC 2007, future plans for the project and present example usages of the new population genetics module.

The latest Biopython release is 1.45 made available on 22 March 2008.

Some of the new features are:

1. A new population genetics module including support for coalescent simulation, selection detection and the GenePop file format. The new module relies on existing open source external software (e.g., the open source Simcoal2 for coalescent simulation which can take advantage of multiple core CPUs for computationally intensive tasks).
2. Improved documentation.
3. Deprecation of many modules which were either obsolete or had been superseded by other code.
4. Plus many bugs were fixed, including updates for evolving file formats.

Since the Biopython 1.45 release, further work is planned to extend the Population Genetics module (e.g., with a statistics component). A new sequence alignment module is also being implemented with a uniform API for reading and writing various alignment files, based on the approach of the Bio.SeqIO module added last year for working with sequences. Work to improve Biopython's BioSQL support is also ongoing.

Time permitting, the talk will also show usage examples of the new population genetics module. The focus will be put not only on the population genetics side, but also on strategies to easily use all available computational power on new multiple core computers. This is useful for users of most scripting languages as most language interpreter implementations impose stern limits on multi-threaded programming efficiency, a topic that becomes important when using computational biology code that is CPU intensive. We will take this opportunity to discuss strategies to overcome those interpreter limitations.

Web site: http://www.biopython.org

License: Biopython License Agreement (an MIT style license)
http://www.biopython.org/DIST/LICENSE

1 Liverpool School of Tropical Medicine, UK
2 University of Warwick, UK