Problem Solving Environment Approach to Integrating Diverse Biological Data Sources

Eric G. Stephan, Kyle R. Klicker, Mudita Singhal, and Heidi J. Sofia.
Pacific Northwest National Laboratory
PO Box 999
Richland, Washington 99352,U.S.A.
{eric.stephan, kyle.klicker, mudita.singhal, heidi.sofia}@pnl.gov

Abstract

Scientists face an ever-increasing challenge in investigating biological systems with high throughput experimental methods such as mass spectrometry and gene arrays because of the scale and complexity of the data and the need to integrate results broadly with heterogeneous other types of information. Many analyses require merging the experimental results with datasets returned from public databases, such as those hosted by the National Center for Biotechnology (NCBI), Kyoto Encyclopedia of Genes and Genomes (KEGG), and protein interaction databases such as the Biomolecular Interaction Network Database (BIND). Because data sources such as these are constantly evolving the researcher is faced with hurdles to manually gather, integrate and manage the data into cohesive datasets. To overcome these technical problems, we have been building a three-tier software system that includes a client-side graphical user interface for rich interaction with the data, an application server that hides the messy technical details of data collection, integration, and management tasks from the researcher, and a flexible database schema that efficiently manages mixed data source content. The software is being developed using Java for portability and Open Source technology so that it can one day be freely distributed. This problem-solving environment is called the Computational Cell Environment (CCE) and is designed to provide scalable and agile connectivity to diverse data stores and eventually provide data retrieval, management, and analysis through all aspects of biological study.

1.0 Introduction

Through our experience working with biologists who routinely rely on many different private and community resources we have identified six areas that are required to prepare biological data/information for data mining: (1) discovery, (2) collection, (3) transformation, (4) integration, (5) management, and (6) portrayal of data into a format that can be analyzed by computational biology tools. The challenge for the biological community is to reduce the cost and time involved in preparing data, performing analyses, and getting results. To do this we are developing next generation Problem-Solving Environment [1] tools and services to bridge current and future technological solutions.

2.0 Approach

Our strategy is to develop and deploy an extensible computer architecture designed to automate the process of collecting, integrating, and managing experimental results, annotation, and bioinformatics analyses to aid in the study of higher level perspective of biological systems.

The Computational Cell Environment (CCE) architecture was implemented using Java for the user interface, a JBoss [2] application server for handling data services, and PostgreSQL [3] database for permanently managing user datasets and temporarily caching data collected from remote data sources. The software was chosen to offer multi-platform appeal, while at the same time providing a tool that is freely available to the bioinformatics community. CCE is being used today on Macintosh, PC, and Linux platforms.

The Computational Cell Environment architecture can be broken into 3 tiers: client interface, data services, and data repository. Each tier has been designed to serve a specific purpose.

The *client* interface provides users a graphical user interface to collect and manage their data, and provides visualization and bioinformatics applications access to the user datasets. The user interface provides a tree-view project space for the user to manage their datasets. Users can create new projects, organize their existing projects, or create new datasets. Once a dataset has been selected, users can view the dataset contents in a spreadsheet form.

The *data services* tier accepts requests from clients using a domain specific language (DSL). The DSL provides a high level interface so that the complexity of the underlying and ever-changing data source access protocols are hidden from the client. This way, the clients are able to rely on an interface that changes less rapidly than the underlying data source access protocols.

The *data repository* tier is responsible for both temporarily caching data users request from data sources, and permanently storing the data in datasets for the user. The schema devised for CCE is extensible, that is as new data services are registered, new corresponding database schema is also registered for the content. Because datasets can be derived from multiple data sources, the data repository tier manages the data model separately from how it is viewed. This model-view-controller design allows the view to be loosely coupled to the underlying model, so that if the view is changed there is little or no impact to the underlying model.

3.0 Discussion

CCE's research efforts have been largely driven by the types of data sources researchers require and by the applications they want to use to analyze the data. Much of our initial research focused on providing PNNL researchers access to their proteomics data and linking their query results to pathways and sequences. Our recent focus has been to provide data service access to pathways (KEGG [5]), sequences (NCBI [6]), protein-protein interaction (BIND [7]), and delimited imported data types (e.g. microarray). CCE is also making use of available cross reference services to merge different data types together. CCE data services relies on underlying Seqhound [4] application interface, hypertext transfer protocol (HTTP) page scraping, and structured query language requests which change occasionally. Because of our design only the underlying code within the data service has been required to change, and the client code has been largely unaffected.

To analyze the data users can register applications that are uniform resource locator (URL) based or installed locally on the users machine. An example application is Cytoscape [8] used to visualize our spreadsheet data as network.

4.0 Conclusion

CCE is an extensible problem solving environment enabling scientists to overcome the technical barriers they face collecting, integrating, and managing their analysis data. With CCE biologists and bioinformaticists have a greater ability and increased flexibility in manipulating, collecting and analyzing their data, and solving their complex problems.

5.0 Acknowledgements

The research described in this paper was conducted under the LDRD Program at the Pacific Northwest National Laboratory, a multiprogram national laboratory operated by Battelle for the U.S. Department of Energy under Contract DE-AC06-76RL01830.

5.0 References

[1] Rice JR and RF Boisvert. 1996. "From Scientific Software Libraries to Problem-Solving Environments." IEEE Computational Science & Engineering, Fall: 44-53.
[2] JBoss: http://www.jboss.org/
[3] PostgreSQL: http://www.postgresql.org/
[4] http://www.blueprint.org/seqhound/
[5] KEGG: http://www.genome.ad.jp/kegg/
[6] NCBI: http://www.ncbi.nlm.nih.gov/
[7] BIND: http://bind.ca
[8] Cytoscape: http://www.cytoscape.org