

A Research Collaboratory for the Open Source Software Research

Yongqin Gao, Matthew Van Antwerp, Scott Christley and Greg Madey

Department of Computer Science and Engineering
University of Notre Dame
ygao1, mvanantw, schrist1, gmadey@nd.edu

Abstract

Various research approaches have been proposed to study the OSS movement. To facilitate this OSS related research, we designed and implemented an online research collaboratory. It is not only a repository including over two years of monthly database dumps from SourceForge.net, but also an online community supporting OSS related research. We describe the design and implementation of the research collaboratory and usage statistics for the year 2006.

1 Introduction

A collaboratory is, as a commonly accepted definition by Cogburn [3], “more than an elaborate collection of information and communications technologies; it is a new networked organizational form that also includes social processes; collaboration techniques; formal and informal communications; and agreement on norms, principles, values, and rules.”

As described by Cogburn, we designed our collaboratory as not only a repository, where the raw data set and related analytical toolkits are stored, but also a community, where researchers sharing the same interest can work and discuss with each other about the related research.

2 Related Work

Problems of geographic separation are especially present in large research projects. The time and cost for traveling, the difficulties in keeping contact with other scientists, the control of experimental apparatus, the distribution of information, and the large number of participants in a research project are just a few of the issues scientists are faced with. Therefore, collaboratories have been put into operation in response to these concerns and restrictions [5].

Chin and Lansing [4] state that the research and development of scientific collaboratories had, thus far, a tool-centric approach. The main goal to design the collaboratories may now move beyond developing general communication mechanisms to evaluating and supporting the very nature of collaboration in the scientific context [4].

Although our collaboratory may be one of the pioneers in supporting software-developing research, research collaboratories supporting bioinformation and genome researches are the well developed due to the gigantic volume of information involved. There are many successful research collaboratories for biotechnology researches. NCBI¹ is one of them. Other successful collaboratories include FlyBase², VectorBase³, EMBL⁴ and GenomeNet⁵.

3 Collaboratory Design

We designed our collaboratory using a three-tier hierarchy. These three tiers are:

1. Data tier (back-end data storage): Since the research collaboratory is a data-central system, a data repository is at the back end of the system to store all the related information. Normally, database is the implementation of the data repository at the back end. The major tasks of this tier include an appropriate schema, an appropriate backup strategy and an appropriate load-balancing technique. The design and implementation of this tier define the reliability, integrity and performance of the collaboratory.
2. Presentation tier (front-end web interface): This tier is responsible for formatting the information and deliver-

¹<http://www.ncbi.nlm.nih.gov>.

²<http://flybase.bio.indiana.edu>.

³<http://www.vectorbase.org>.

⁴<http://www.ebi.ac.uk/embl>.

⁵<http://www.genome.jp>.

ing it to the end user. It relieves the user of concerns in regard to data representation within the logic tier or the data tier. A web interface is one of the commonly used front-end tiers. The major tasks of this tier including providing various access methods to the backend repository and providing community support for the researchers. The design and implementation of this tier define the availability and usability of the collaboratory.

3. Logic tier (middle-layer service provider): This tier implements most of the processes and functions provided in the presentation tier and usually these processes and functions are based on the information stored in the data tier. Most of the programming language supporting networking can be used in this tier. The major functionality of this tier is implementing every function provided in the presentation tier. The design and implementation of this tier define the functionality of the collaboratory.

4 Implementation and Discussion

We built the research collaboratory for the Open Source Software Research based on the monthly database dumps we receive from SouceForge.net [2] under a data sharing agreement. We used the three-tier hierarchy to implement our collaboratory. Details about the hierarchy used in our collaboratory are shown in Figure 1.

We will explain details about these three tiers in our collaboratory in the rest of this section.

4.1 Data Tier

First of all, we will explain the data tier. As discussed in the previous section, there are three major tasks in designing the data tier – repository schema, backup strategy and load balancing.

Every month we receive a database dump from SourceForge.net. We need to design a repository schema supporting evolving data sets. So a multiple-layer data schema is used in the data repository. Timeline is the name of the top level – the database, where all the data dumps from SourceForge.net are stored. In the database, there are multiple schemas⁶. Every schema represents a single month, and stores the database dump from SourceForge.net of the corresponding month. These schemas are named from the month of the corresponding database dump. For example, schema

⁶A logic level between database and table.

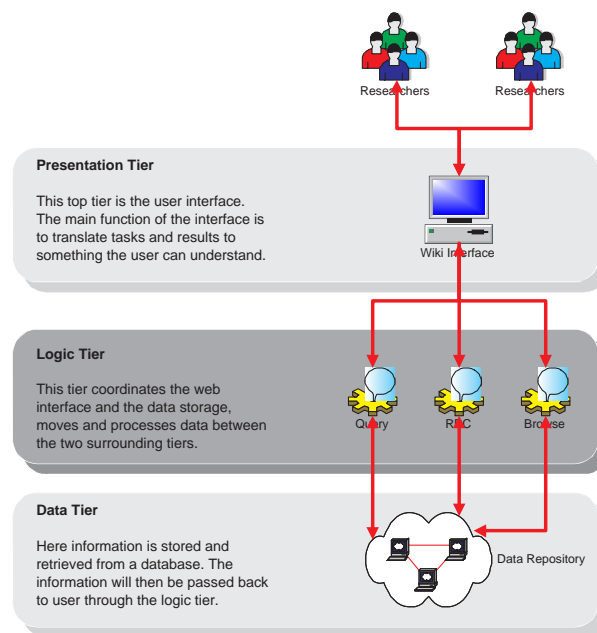


Figure 1. Three-tier Hierarchy for Our Collaboratory

“SF0103” stores the database dump of January 2003. In every schema, we copy the intact table definitions from the corresponding database dump as the table definitions.

Another important task of the data tier is the backup strategy. Since our repository should expand only every month, a new database dump is restored. A periodic (monthly) full repository backup is good enough to keep the repository fully recoverable. This backup action can be initialized by cron job or manually. In addition to this periodic full repository backup, we also keep a full backup of the database dump files in case there is potential error during data loading.

As to load balancing, we applied connection management only for our collaboratory. This is because currently our back-end database is only a single server hosting the “timeline” database and load balancing is not necessary. Instead, connection management is needed to relieve the back-end server from a possible heavy load of queries⁷. In order to mitigate the possible overloading issue for the database server, a connection pool mechanism is engaged. The connection pool mechanism is shown in Figure 2.

The connection pool entity acts like a daemon running on the database server, which keeps a predetermined number of persistent links to the database – “timeline.” This predeter-

⁷According to our statistics, in June 2006, we received 16,947 queries, returned 13,343 files in total size of over 26G from just 24 active users.

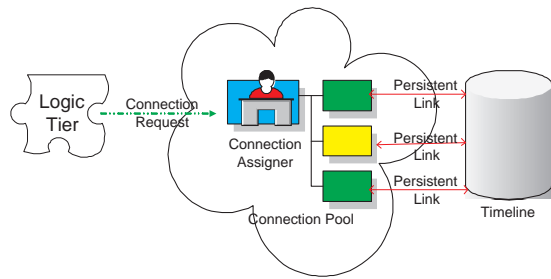


Figure 2. Connection Pool Design

mined number can be adjusted, according to the processing power of the database server. From the logic tier point of view, this connection pool entity works like a virtual back-end database. Every request to access the database will be submitted to the connection assigner, and the assigner will assign an idle link to the request procedure, if available, and recollect the link when the work is done. If every link is busy, the request will be queued at the connection assigner for the next available link to access the real back-end database. By engaging the connection pool mechanism and setting the appropriate number of persistent links, it is guaranteed that the database server won't be swamped by connections and every requests will be queued at the connection pool without suffering from bad performance or dropping. Also, by adjusting the number of persistent links, we can tune up the performance of the back-end database.

4.2 Presentation Tier

The presentation tier is the user interface of the research collaboratory. Community support and repository access are two of the major functions of the presentation tier. However, normally these two functions have different requirements for authorizations. For the repository access, tight authorization is needed to avoid information abuse. On the other hand, community support needs relatively loose authorization and, actually, sometimes anonymous users' contributions are vital to a "virtual" community (e.g., online bulletin board service). Thus, in our presentation tier, public community service and restricted repository access should be provided with layered authorization. All users, including anonymous users, can access the community; registered users are allowed to post on the community and only specially authorized users can access the back-end repository through the web interface or programmable interface.

For the public community service, we used wiki technol-

ogy⁸. The wiki engine (the collaborative software) we are using is *MediaWiki*, which is also the wiki engine used by wikipedia⁹. Figure 3 is a screen shot of the user interface of the collaboratory using the wiki technology.

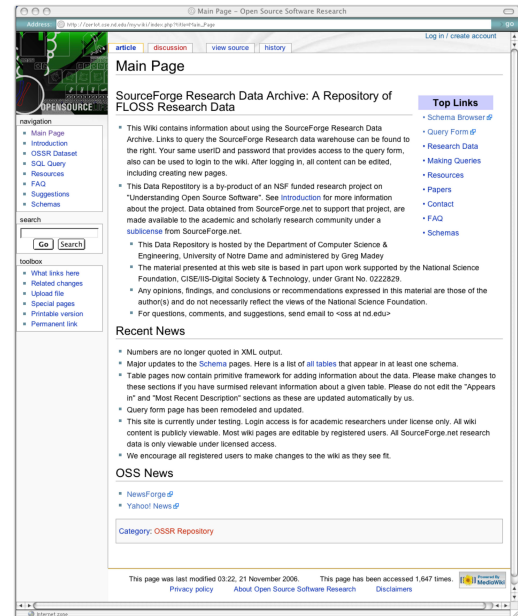


Figure 3. Screenshot of the Wiki Interface for "SourceForge Research Data Archive: A Repository of FLOSS Research Data" (<http://zerlot.cse.nd.edu/>).

Another major function of the presentation tier is the restricted repository access. The web interface module is implemented using Perl script and ODBC. There are two sub-interfaces provided in the web interface – database query interface and the schema browser interface. Authorized users can submit queries directly to the backed repository through the database query interface. And the schema browser provides a dynamic interface to browse the database schema of the repository, including available schemas and table definitions. Details about the implementation of these functions will be discussed in the next subsection about logic tier.

⁸A *wiki* is a type of website that allows visitors to easily add, remove, or otherwise edit and change some available content, sometimes without the need for registration. The ease of interaction and operation make a wiki an effective tool for collaborative authoring.

⁹<http://www.wikipedia.org>.

4.3 Logic Tier

Logic tier includes all the processes implementing every function in the presentation tier. In our implementation, access to the data repository is one of the major functions needed to be implemented in the logic tier. There are two kinds of access methods to the back-end repository provided in our collaboratory – web access and programmable access.

Web access is the simple method to access the back-end repository and it is used to implement the database query interface discussed before. In case the query can be described as a single query and submitted through a web page, web access is the preferred access method. In our implementation, authorized users can submit queries through a provided web page; the result will be saved in the user directory and be made available for download by the user. We use files instead of returning the result directly to the user to avoid unexpected network delays and other problems. Also, by saving the result in a file, we can avoid duplicate requests to mitigate the burden on the back-end repository. Figure 4 is a screenshot for the query page.

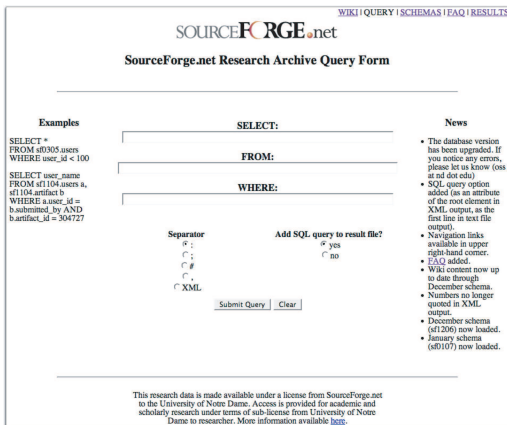


Figure 4. Screenshot of the Restricted Query Form for Our Collaboratory.

In our previous discuss, we mentioned that the schema of the data repository is evolving. It is difficult for us to keep publishing up-to-date schema on the wiki site and it is impossible for the user to query the data repository without the current knowledge about the schemas. To solve this problem, we provided a dynamic schema browser, which is simply another web query system about the data schemas only. By using the schema browser, users can pull the current schema directly from the back-end repository and then query the back-end repository according to the current schema.

With the schema browser, a typical user query can be divided into two steps. The first step is to retrieve the related table names and attribute names. The next step is to generate the query using the information retrieved in the first step and submit the query. This two-step approach provides a complete solution for users for submitting queries to access the dynamic evolving back-end repository.

Although most of the query for the data repository can be completed through the simple web access, there are some situations where simple web access is not enough. For example, if the user has a sophisticated request than cannot be described using single query or the user needs use procedures (queries with control structure like loop and condition) in the query, programmable access is needed. Currently only web access is open to our users, but web service functionality is in testing phase, which will allow programmable access to the back-end repository.

This collaboratory is implemented for the Open Source Software Research community. We provided access to the data repository of the SourceForge.net database dump and published our research about SourceForge.net in the collaboratory. Although it is a fairly new community, the statistics show that the usage of the collaboratory is promising. We expect this collaboratory could contribute to the research of Open Source Software.

5 Usage Statistics

The size of the database recently exceeded 450GB. There are over 80 registered users worldwide. In a typical month, 15 to 20 users will query the database. There are 27 schemas that make up the database. Since late June of 2006, the server has received approximately 488 queries per day on average. Figure 5 has query and download statistics by month for the year 2006. On the left y-axis is queries per month. On the right y-axis is data downloaded in gigabytes per month.

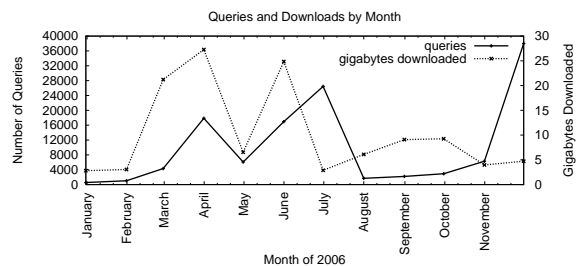


Figure 5. Usage statistics for the year 2006

6 Conclusion

We designed and implemented the research collaboratory for OSS related research. The collaboratory includes a repository for the research-related information and a community for the researchers to cooperate with each other in the research. The collaboratory provides a method to share information, publish the research results, discuss it with peer researchers and promote future research in the domain.

We implemented our research collaboratory using three-tier hierarchy. For the data tier, we designed a self-evolving data schema, periodic backup strategy and a connection control mechanism to ensure the reliability, integrity and performance of the collaboratory. For the presentation tier, we built a community service for the collaboratory using wiki technology and a web access module (restricted query and dynamic schema browser). We are also testing three web service protocols to implement the programmable access module. For the logic tier, we implemented the functions in the web access module, including restricted query and the dynamic schema browser, using Perl and ODBC. The design of the collaboratory should permit additional data set to be added, including those collected from other open source project sites and researcher's own contributed data sets.

Acknowledgment. This work was supported in part by the National Science Foundation, CISE/IIS-Digital Society & Technology, under Grant No. 0222829 [1, 2].

References

- [1] OSS research (<http://www.nd.edu/~oss>). 2006.
- [2] OSS research portal (<http://zerlot.cse.nd.edu>). 2006.
- [3] D. L. Cogburn. HCI in the so-called developing world: What's in it for everyone. *Interactions*, 10(2):80–87, 2003.
- [4] G. Chin Jr. and C. Lansing. The biological sciences collaboratory. *Mathematics and Engineering Techniques in Medicine and Biological Sciences*, 2004.
- [5] D. H. Sonnenwald. The conceptual organization: an emergent collaborative R&D organizational form. *Science Public Policy*, 30(4):261–272, 2003.