SCLIP: Empowering Real-time Bioinformatics Collaborations via Social Networks

Gil Alterovitz[1,2,3,4§], Kshitij Marwah [5], Paresh Malalur[2], Anshuman Aggarwal[5], Marco F. Ramoni[1,3,4]

[1]Division of Health Sciences and Technology, Harvard University and Massachusetts Institute of Technology, Cambridge, MA, USA

[2]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA

[3]Children's Hospital Informatics Program, Boston, MA, USA

[4]Center for Personalized Genetic Medicine, Partners HealthCare, Boston, MA, USA

[5]Department of Computer Science and Engineering, Indian Institute of Technology, Delhi, India

[§]Corresponding author

Email addresses:

GA: gil@mit.edu

KM: kshtj22@gmail.com

PM: pareshmg@mit.edu

AA: anshuman.iitdelhi@gmail.com

MFR: marco_ramoni@harvard.edu

Abstract

Background

Bioinformatics is a highly interdisciplinary field. Generally, researchers in this field specialize in their own computer language and work with vast code libraries in their own community. This make collaborative project management in bioinformatics a very complex process involving sharing of code functionalities between users and developers. Social networks can serve both as a bridge and a source of computational power to allow effective collaborations between researchers. Currently, there is no such framework that can allow for cross-language communication using the immense computational power available to us via social networks.

Results

SCLIP is a framework that allows for real-time communication between different languages over a web-based interface using social networks to facilitate seamless collaborations between researchers. It combines a facebook application which provides an interface for sharing methods and a daemon service running on various systems for seamless code sharing and distributed execution using social networks. It supports all the currently used programming languages in bioinformatics over all existing platforms. Moreover, the framework also supports an interactive editor and graph viewing functionalities for real-time analysis.

Conclusions

We have developed a framework designed to facilitate collaborations by combining languages and platforms in real-time and using the computational power and resources available in social networks. The framework is open source and can be downloaded from http://bcl.med.harvard.edu/proj/SCLIP.

Background

At its core, bioinformatics is essentially a collaborative endeavor, bringing together researchers from disparate backgrounds. By it's nature it involves interdisciplinary work between biologists, mathematicians, engineers and computer scientists. When it comes to programming languages, these users and developers of bioinformatics algorithms tend to specialize in different language variants based on their training, background, and researcher experiences. Each knows and comes with their own language but perhaps more importantly can access vast libraries of code within his or her speciality's community written in their native language. In addition, some specialities use one platform e.g. Mac OS X while others use different systems e.g. PC Windows terminal, UNIX server for calculations.

Also, social networking web sites, such as facebook, provide a powerful framework and resources for enabling collaborative endeavours amongst researchers. Due to their inherent distributed nature, they serve as a source of immense computational power and can facilitate sharing of code functionalities across users. This feature of social networks can be utilized for integrating knowledge, methods and data from different disciplines. .
Currently available, collaborative code sharing project management tools typically allow various researchers to work on a project based on a single programming language in a centralized environment. This works well when there is a closely knit team of scientists working in a common language and executing code on a single machine. Also, current programming models for cross-language communication such as SWIG, (http://www.swig.org) use a high level language for prototyping and low level one for optimized code. This is suited to situations in which when there is a team of researchers who typically use one language, say Matlab and want to interface it with say, C for speed. Other programming models [2] use an interface definition [1] and a language or platform description of a software library. These are then processed to generate glue-code which is used for cross-language communication. This approach requires significant familiarity with the tools, which greatly impairs diffusion of its use.

Various software tools that allow for distributed computing like Globus [3] have their own toolkits for specific languages and computing facilities for deployment of grid-based projects [6].

There is a need for a platform which can enable researchers to share and execute codes and functionalities in various languages across disparate platforms. Moreover, use of social networks as a distributed source of computational power, resources and as a means of sharing of methods across users can be beneficial to the bioinformatics community.
We have therefore developed SCLIP to allow cooperation between developers by allowing seamless code sharing and distributed execution over a web-based framework using social networks.

Implementation

SCLIP is composed of two parts: 1) a facebook application providing a web-based framework for sharing methods and 2) a daemon running as a service on various systems for allowing cross-language communication (see Figure 1).

The facebook application is implemented in Facebook Markup Language (FBML), using the facebook Application Programming Interface (API) and AJAX. AJAX allows asynchronous loading of data without the need to reload the page thus providing a desktop like application behaviour.

The daemon service is graphical based, implemented using Python, a platform independent object-oriented programming and scripting language.
The facebook application provides an interactive framework for the user to view, share and execute various methods deployed by him and his or her friends. This functionality is broadly implemented under three distinct interfaces in the facebook application.

The first interface, View Methods allows the user to query the available shared methods by his or her friends. The second interface, Editor enables real time programming and execution of methods, with a graphical window to view or download results. The base language used for writing programs in the editor is Python. The third interface, Multi-Mode Console combines the first two interfaces enabling rapid prototyping and testing. This application also allows the users to register their daemon service so as to enable their shared methods to be accessed by their friends.

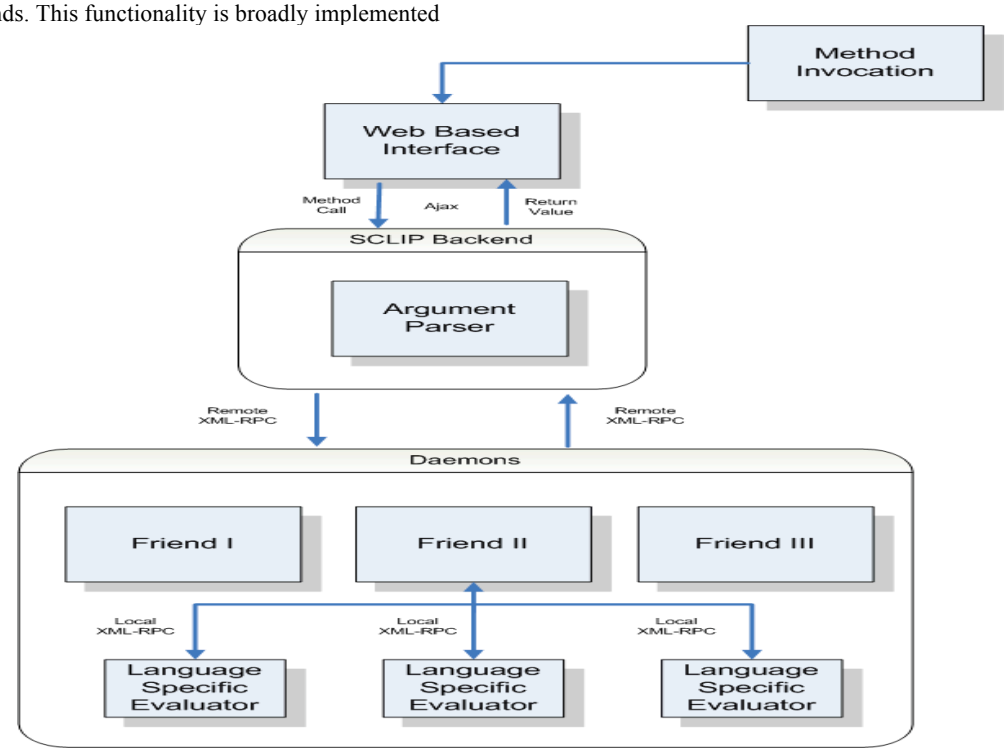

Figure 1

Flow Chart. This figure illustrates the overall SCLIP communication flow diagram.

Each user in the facebook application works in his or her own address space containing the source and the data files for each user. File handling mechanisms are provided for uploading and downloading of source and data files.

The daemon software enables users to integrate their code functionalities written in various programming languages. It provides a very intuitive Graphical User Interface, which allows users to add methods in all major languages used in bioinformatics. The automatic detection and integration mechanism in the daemon makes it very easy to use and deploy. The daemon comes with the latest precompiled binaries for all the languages it supports for various platforms, so the user does not need to install.

For up gradation of libraries or compilers the user just needs to install the updated versions in the daemon directory. The daemon then uses the new versions for all further compilations and executions.

Protocol between the Facebook Application and Daemon

SCLIP uses XML-RPC, which is a simple yet a very robust remote procedure calling mechanism for interaction between the application and the daemon. This protocol uses the Hypertext Transport Protocol (HTTP) as the transport vehicle and Extensible Markup Language (XML) for encoding, allowing complex data structures to be transmitted, processed and returned [13]. The data types supported range from primitive data types such as integers, doubles and strings to complex data types such as arrays and structures.

The use of XML-RPC enables SCLIP to make remote procedure calls over the Internet and we preferred this for it's simplicity, minimalism, easy

of use and that most programming languages support libraries for this protocol.

Interplay among the Facebook Application and Daemon

SCLIP uses methods or functions in programming language as atomic constructs to be shared and executed amongst users.

A user who wants to share his methods with his friends, adds them to be shared by him using the daemon software. Integrating methods from various language is made vey intuitive and easy by the daemon. The videos and the manual available on the web-site provide a detailed explanation on this process. The user then registers his daemon service using the facebook application so as to enable their sharing and execution across his friends.

The user's friends can now select functions offered by him. For accessing shared methods there is a special syntax

>> clip.<Method Name>(<Arguments>)

When the code is executed, interspersed with these shared methods, the requested method is parsed by the backend of the facebook application to extract out the function name and the arguments. Next, the associated system with the function is directed to execute the function. The system containing this function spawns a language specific XML-RPC service, and evaluates the function. The result is sent back to the facebook application for further computation or to be displayed to the user.

Results and Discussion

The SCLIP daemon currently supports sharing of code functionalities for Python, Java, Lisp, Matlab, R, Perl, C, C++ and Ruby. It has functionality for automatically detecting and integrating XML-RPC capabilities to any given software library.

An example of the functionality of SCLIP is demonstrated here using an Affymetrix gene expression dataset [12]. This study involved two types of acute leukemias: Acute Lymphoblasic Leukemia (ALL) and Acute Myeloid Leukemia (AML). Clustering analysis using partitioning algorithms like pam (partitioning around medoids) was performed on the dataset and silhouette plots were produced using SCLIP (Figure 2).

In this example, the user specified the code for R-implemented algorithm via the facebook application and this request was directed to a friend's computer for execution. The R function was executed, the results were returned in a non-language specific format, and the output was then plotted by the facebook application (see Figure 2). Further analysis can be done by calling methods implementing algorithms in other languages. Moreover, the code and the plots can be shared between friends for further work.

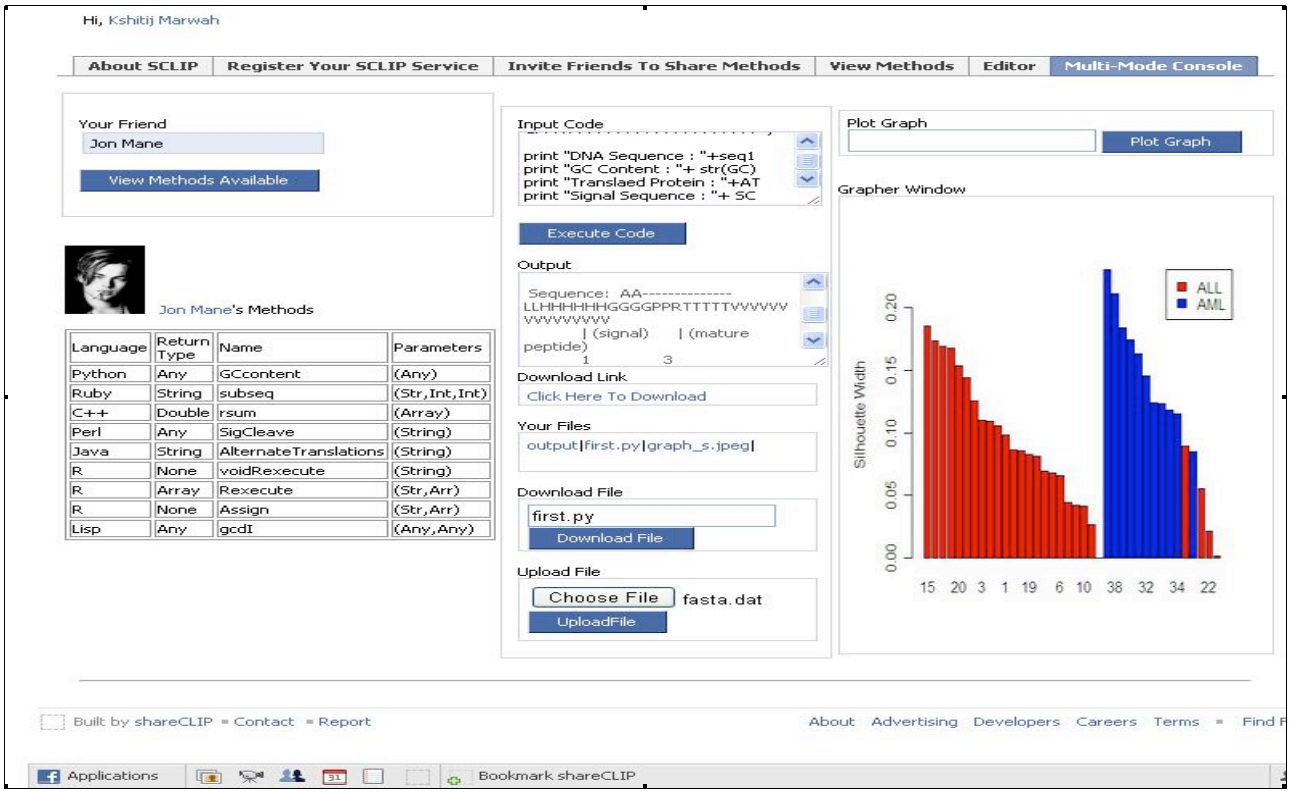

Figure 2
Multi-Mode Console. Silhouette plot for Acute Lymphoblastic Leukemia (ALL) and Acute Myeloid Leukemia (AML) datasets calculated using SCLIP.

This example also highlights various methods in disparate languages shared by the user's friends. A sample output of the execution of those methods is also displayed.

We have developed an integrated platform for enabling seamless collaborations within an interdisciplinary workspace. The uniqueness of this framework is in the intuitiveness and ease in which it enables programming language interoperability combined together with the novel use of social networks for enabling distributed computation and assimilation of results.

Other existing tools for programming language interoperability, are either inadequate for the needs that arise in bioinformatics or have a very steep learning curve thus rendering their limited use. This framework certainly tried to overcome these limitations that exist in the currently available infrastructure promising fast prototyping and deployment of interoperable code. We have supported all the major programming languages in use by the bioinformatics community.

Thus, this framework provides a novel way to seamlessly integrate analysis within an interdisciplinary and disparate workspace across researchers.

Conclusions

SCLIP allows bioinformatics researchers to collaborate in real-time by using social networks as a means for cooperation and sharing of resources. Most of the languages used in bioinformatics are currently supported, allowing researchers all over the world to communicate seamlessly. Given this incorporation of various languages in use by researchers in bioinformatics combined with the enormous potential of social networks in terms of the computational resources it provides, the developed framework could be of great interest and use to the bioinformatics community.

Availability and Requirements

- Project name: SCLIP
- Project home page: http://bcl.med.harvard.edu/proj/SCLIP
- Operating System: Microsoft Windows, Linux, Mac OS X
- Programming Language: Python, FBML, Ajax
- Other requirements: Web-browser, Facebook Account, 800 MB of free hard drive space with minimum 256 MB of RAM.
- Licence: GNU-GPL
- Any restrictions to use by non-academics: GNU-GPL Licence

Installation of the SCLIP daemon is provided through an installer and should be complete within half and hour. Step-by-step instructions are provided in the manual available on the web-site. The Facebook application is browser based, facebook specific and only requires the necessary rights to access your profile details.

Authors' contributions

GA conceptualised the idea, design of the application and helped in drafting the manuscript. KM designed the application and drafted the manuscript. He was responsible for the implementation of the daemon and the facebook application. PM gave valuable suggestions on the usability of the framework. AA participated in implementing intermediate functionaries for the application. MFR was responsible for the overall project coordination. All authors give final approval of the version to be published.

Acknowledgements

References

1. Jacobsen A: **Programming language interoperability in distributed computing environments.** In *Proceedings of the IFIP WG 61 International Working Conference on Distributed Applications and Interoperable Systems II*. 1999: 287 – 300.

2. S. Kohn GK, J. Painter, C.J. Ribbens: **Divorcing language dependencies from a scientific software library.** In *Proceedings of the Tenth SIAM Conference on Parallel Processing for Scientific Computing; Philadelphia, PA*. 2001

3. Foster I: **Globus Toolkit Version 4: Software for Service-Oriented Systems.** In *FIP International Conference on Network and Parallel Computing*. 2006: 2 – 13

4. Rob Armsrtong DG, Al Geist, Katarzyna Keahey, Scott Kohn, Lois McInnes, Steve Parker, Brent Smolinski: **Toward a Common Component Architecture for High-Performance Scientific Computing.** In *Proceedings of the 8th IEEE International Symposium on High Performance Distributed Computing*. 1999: 13.

5. E-Kai Shen SM, Istabrak Abdul-Fatah: **High performance adaptive middleware for CORBA-based systems.** In *Proceedings of the nineteenth annual ACM symposium on Principles of distributed computing*. 2000: 199 - 207.

6. Raffaele Montella GA, Daniele Mastrangelo, Milena Menna: **A globus toolkit 4 based instrument service for environmental**

data acquisition and distribution. In *Proceedings of the third international workshop on Use of P2P, grid and agents for the development of content networks; Boston, MA, USA*. 2008: 21 - 28.

7. Remi Bastide PP, Ousmane Sy, David Navarre: **Formal specification of CORBA services: experience and lessons learned.** In *Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications; Minneapolis, Minnesota, United States*. 2000: 105 - 117.

8. Vinoski S: **CORBA: Integrating Diverse Applications Within Distributed Heterogenous Environments.** *IEEE Communications Magazine* 1997.

9. Maria Toro TTH, Jinsong Zhu, Kangming Liu, Victor C. M. Leung: **CORBA based design and implementation of universal personal computing.** *Mobile Networks and Applications* 2003, **8:**75 - 86.

10. S. Brunett KC, S. Fitzgerald, C. Kesselman, I. Foster, S. Tuecke, A. Johnson, J. Leigh: **Application Experiences with the Globus Toolkit.** In *Proceedings of the 7th IEEE International Symposium on High Performance Distributed Computing*. 1998: 81.

11. Gary Kumfert DEB, Thomas Epperly, James Kohl, Lois Curfman McInnes, Steven Parker, and Jaideep Ray: **How the Common Component Architecture Advances Computational Science.** In *Proceedings of Scientific Discovery through Advanced Computing*. 2006

12. T.R. Golub DKS, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, E.S. Lander: **Molecular classification of cancer: class discovery and class prediction by gene expression monitoring.** *Science* 1999, **286:**531 - 537.

13. Allman M: **An evaluation of XML-RPC.** *ACM SIGMETRICS Performance Evaluation Review* 2003, **30:**2 - 11.