CLOSED AND OPEN SOURCE NEUROIMAGE ANALYSIS TOOLS AND LIBRARIES AT UNC

Martin Styner, Matthieu Jomier, Guido Gerig

Departments of Computer Science and Psychiatry, University of North Carolina at Chapel Hill, NC

ABSTRACT

The emergence of open-source libraries and development tools in the last decade has changed the process of academic software development in many ways. In medical image processing and visualization this change is especially evident, also because open source projects are actively furthered by grant funding institutions. This manuscript presents the use of such development tools and libraries at the UNC Neuro-Image Analysis Laboratory for open source applications and tools. We have also experienced in our research that the development of open source in academics raises the issue of access to unpublished methodology. The strategy at our laboratory is to combine all in-house libraries and applications into a single repository that consists of two parts: a fully open source part that is distributed under a Berkley-style license and a private, closed source part with unpublished tools and methods. Access to the open source part is unrestricted, whereas the private parts can only be downloaded via cvs user login. This setup solved our issues regarding unpublished methodology, as migration from the private to the open source part is very simple. Overall our experience with this development environment within the academic setting is very positive.

1. INTRODUCTION

Software development in medical image analysis and visualization is structurally quite different than it was less than 10 years ago due to the maturation of major open-source community libraries and tools. The field was always quite open for collaboration between different laboratories, but exchange of actual source code and implementation of methods were not common place, as most groups had their own software solutions which quite often were single-platform or based on proprietary software. Several changes in the community lead to the ongoing change towards openly exchanging source code. One major contribution were the maturation and stability of open source libraries like the visualization toolkit[1] and the Insight Segmentation and Registration Toolkit[2], as well as open source development tools such as CVS, CMake, Doxygen and others. As the research groups become more open regarding publication of their source code, the issue of access to unpublished methodology arose. This manuscript presents the current development environment at the UNC Neuro-Image Analysis Lab with its open and closed source tool and library repository UNC NeuroLib.

In the next section, we discuss the motivation and aims for our development process. Then, the basic set of tools, libraries and services that make up our development environment are presented, followed by the use of internet based communication for dissemination, training and support. Next, our view of open and closed source in academics, as well as of licensing and patenting issues are presented.

2. MOTIVATION AND AIMS

The structuring of our development environment was mainly motivated by the difficulties in managing and synchronizing multiple overlapping developments in our research projects. Inspired by the success of the development process in the open-source libraries the Visualization Tool Kit (VTK)[1] [1] and the Insight Segmentation and Registration Tool Kit (ITK) [2] [2], we aimed to mirror some of these processes in our laboratory. Both libraries are vital to the tool development in medical image analysis and visualization in many academic research groups due to their open source license, stability, and the availability of a large set of methods. Unlike these libraries, we focus on stand-alone tools that can be readily used in clinical neuro-image analysis research, e.g., morphometric studies of the human brain in neuro-degenerative or neuro-developmental diseases. The criterions listed below were used in the design of the UNC NeuroLib:

- Libraries: Methods are grouped into libraries for use in different research projects. All methods are continuously tested regarding compilation and functional aspects in order to facilitate maintenance.
- Automation: While many tools will always need direct user interaction, we aim to increase the level of automation such that the tools can be run in an interactive 'training' mode and an automatic mode with a set of prior 'trained' parameters. Quality assessment of the

This research is supported by the NIH Roadmap for Medical Research, Grant U54-EB005149, as well as NIH NIBIB grant P01 EB002779, NIH grant RO1 MH61696, and the UNC Neurodevelopmental Disorders Research Center HD 03110

[1] Visualization Toolkit (VTK), http://www.vtk.org

[2] Insight Segmentation and Registration Toolkit", http://www.itk.org

processing are computed both qualitatively (visualizations)and quantitatively to allow efficient inspection of the computed results.

- Modularity: We aim to develop small to mid-size applications each solving a clearly outlined problem. These applications can be serially applied and employed in pipeline software (e.g., the UNC NeuroLib tool "Imagine", or the UCLA LONI pipeline software [3]) or larger scale projects (e.g., Slicer [3]).
- Dissemination: In general, we want to develop processing and analysis tools that can be disseminated to clinical collaborators, collaborating research labs, as well as to the general research community.
- Stability: All tools are validated with clinical data and are continuously tested. The stability of both the functionality as well as the code base is highlighted when training developers.
- Versioning: While the development of a tool is progressing, it may already be employed in neuroimaging studies. In these studies we aim at using a single tool version throughout the whole study. This calls for a versioning process, in which different version of a tool can be released and minor bug fixes can be applied to different versions.
- Cross-platform development: The academic computing environment is constantly changing and cross-platform development allows for easy migration from one platform to another. It also facilitates dissemination.
- Support: Through supporting computing and information infrastructure and services, all the above mentioned criterions are aided and monitored.

3. STRUCTURE OF THE DEVELOPMENT ENVIRONMENT

The UNC NeuroLib development environment (see Fig. 1) depends on a large set of libraries of tools that are all stable, cross-platform and open-source projects. Thanks to the cross-platform nature of these toolkits, the UNC NeuroLib itself has been compiled for several Linux flavors, Solaris, Windows and MacOS X. The source code of the UNC NeuroLib is available via CVS [4] access. The CVS repository of NeuroLib consists of two main parts: a fully open source part that is distributed under a Berkley-style license and an internal, private part with unpublished tools and research libraries. This setup solves issues of open dissemination and protecting unpublished methodology (see Sec. 5). Central repository access also allows the use of the extreme programming paradigm.

[3] 3D Slicer, http://www.slicer.org

[4] Concurrent Versions System (CVS), http://www.nongnu.org/cvs

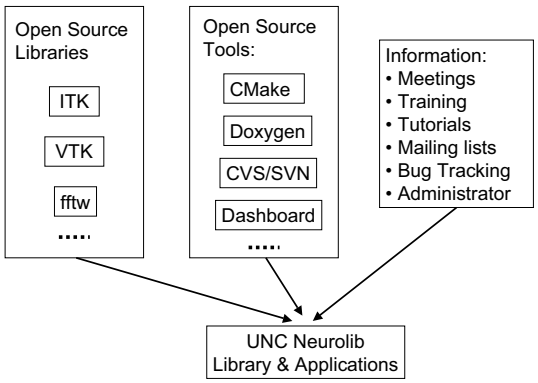

Fig. 1. Scheme of the development environment for the UNC NeuroLib. Open-source and cross-platform libraries and tools build the foundation of the NeuroLib, which is maintained in a single CVS repository. A set of supporting information are also needed for proper training, ongoing development, maintenance and dissemination.

Due to the modularity of the tools, only few projects involve a larger number of developers, e.g., the FiberViewer [4]. For those projects, frequent cycles of design, implementation and testing phases with continuous integration of new parts are the goal. Thus, all developers work on their copy of the source code, which is synchronized with the repository as often as possible. The guiding principle is "Release early, Release often" (Bill Lorensen, GE Research). This necessitates that all developers agree to keep the software as a whole defect free, as well as that a continuous monitoring of the repository is established. Several open-source tools facility our monitoring task: CVS [4] for access, CMake [5] for cross-platform compilation setup and Dart [6] for web-access compilation and testing quality dashboard. Additional open source tools solve cross-platform editing (SourceNavigator [7]), code documentation (Doxygen [8]), bug tracking and project management.

At the center of the methodological aspects of our development environment are a set of libraries handling input/output (x-medcon [9], ITK [2]), image processing (ITK [2]), signal processing (fftw [10]), and visualization (VTK [1], SOViewer [11]). Our group actively adapts and extends the source code of many of these libraries. Some of these extensions are part of the UNC NeuroLib and others made their way back to the

[5] Cross-platform Make, http://www.cmake.org

[6] Dart: Tests, Reports and Dashboard",http://public.kitware.com/Dart

[7] Source Navigator, http://sourcenav.sourceforge.net

[8] Doxygen Documentation System, http://www.doxygen.org

[9] (X)Medcon utility", http://xmedcon.sourceforge.net

[10] FFTW: Fast, Free C FFT Library, http://www.fftw.org

[11] SOViewer library", http://caddlab.rad.unc.edu/software/SOViewer

originating libraries. The use of this set of libraries considerably shortened the implementation time of new applications, even though it also demands a longer initial training phase.

The internal libraries are all that is needed for many of the command-line interface based tools in the UNC NeuroLib. However, few clinical collaborators can work with such tools without considerable training as they are unfamiliar with shell commands, command-line interaction and scripting. Most users of our tools are not engineers and thus a graphical user interface (GUI) is a necessity for most of our projects. This is achieved through GUI based pipeline tools (Imagine, LONI pipeline [3]) for command-line tools, as well as a host of generic and application specific stand-alone tools. Several GUI libraries are used in our tools: Qt [12], FLTK [13] and KWWidgets [14]. All these libraries offer similar functionality, but each one has its individual advantages and disadvantages regarding licensing (see Sec. 6), ease-of-use, and functionality.

4. INTERNET BASED COMMUNICATION

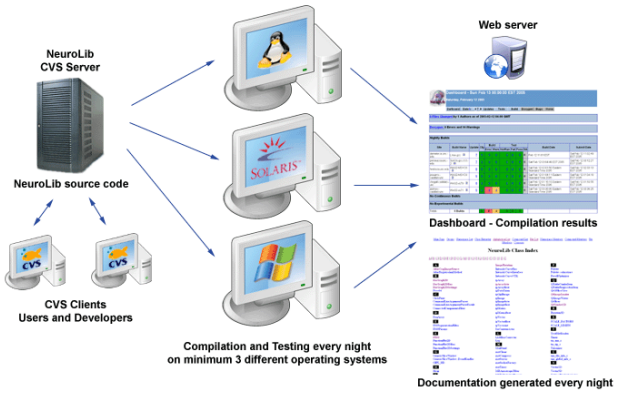

Fig. 2. Schematic visualization of maintenance support via server-based source revisioning (CVS/SVN), cross-compilation, module testing (both via Dart) and automatic documentation generation (Doxygen).

In order to facilitate training, ongoing development and dissemination with information services, our lab hosts a UNC NeuroLib devoted webpage (see Fig. 3). This page includes a download section for many tools with installers, as well as a bug database, project manager, online tutorials, mailings lists and discussion forums. The nightly cross-platform compilation/testing dashboards and the local Wiki documentation system complete the supported facilities. The webpage is located at http://www.ia.unc.edu/dev .

[12]Qt Application Framework: http://www.trolltech.com/products/qt/
[13]Fast Light Toolkit, http://www.fltk.org
[14]KWWidgets, http://www.vtk.org/Wiki/KWWidgets

The web-based setup allows incoming students and developers to do a larger part of their training by themselves, but face-to-face training, and regular meetings for problem solving and design reviews are also necessary. They are also enrolled in the mailing list and are introduced to the development support system. This support system entails the NeuroLib CVS repository and the nightly compilation of the repository on Linux, Windows XP and Solaris 9 machines (see Fig. 2). Most libraries and tools have additional testing programs, which are run automatically after the compilation in order to test the correctness of the compiled code. Results from both compilation and testing are visualized in the automatically generated dashboard. The online documentation is generated nightly using the Doxygen documentation system, which generates the documentation directly from the source code and comments within the code.

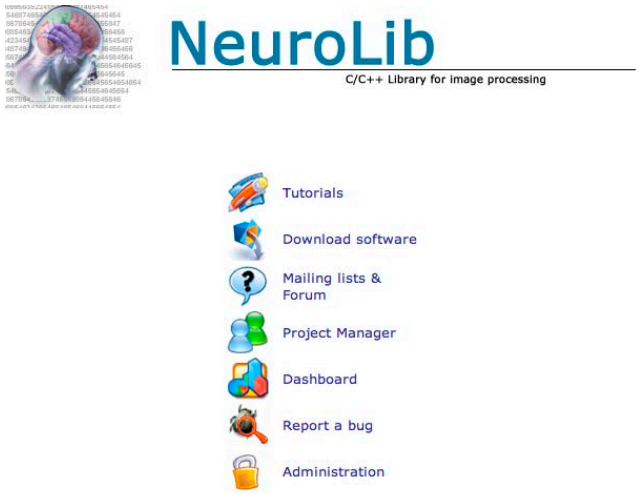

Fig. 3. Entry page of the Neurolib webpage. Several services are directly available from this webpage, most importantly the tutorials and the nightly dashboard.

5. CLOSED VS. OPEN SOURCE IN ACADEMICS

A few years ago, the National Institutes of Health (NIH) announced clear goals for software dissemination for many of their funding programs. These goals entail that software developed with NIH money should be freely available and permit the commercialization of enhanced or customized versions. In some projects, such as the NIH National Centers For Biomedical Computing, it is a necessity that researchers outside the center and its collaborating projects be able to modify the source code and to share modifications. This position in favor of open source research by the NIH differs somewhat from policies in academics that further patenting and commercialization of software and methodology. The position of

our laboratory is a clear commitment to open-source software without the need for software patents.

Academic researchers that are committed to open source often struggle with the need for the internal development of novel, unpublished methods, as well as the dissemination of established tools and libraries, which still need regular maintenance and bug fixing. Both unpublished and established research tools usually rely on the same libraries and the same development and testing framework. A common solution in the field is to release the source code for disseminated tools separately, often in a separate version, and keep the development repository fully private. This solution involves considerable overhead as extracting the source code of a specific tool including any dependent code located elsewhere in the repository is not trivial. This involves collecting the source code, defining a separate compilation environment and additional testing phase. This leads to a rather low frequency of releasing new versions.

Our proposal is a single source code repository and continuous release. In order to protect the source code that contains unpublished tools and libraries from dissemination, the repository is divided into 2 parts. The first part is open source and access to this part is unrestricted via CVS anonymous login. The other part is subject to restricted access via CVS user login. Thus, the download and update of the repository for both types of user access is the same only the CVS login is different. The open source part is fully independent of the private part, whereas the private part also depends on libraries located in the open source part. Once a novel method has been published in peer-reviewed literature, the corresponding source code is migrated from the private to the open source part. If a tool is to be released in a separate open source version, but the actual local development continues on the private parts, the necessary parts are copied rather than moved.

6. LICENSING AND PATENT ISSUES

In this section, we briefly discuss our experience with issues of licensing and patenting of internal and external libraries. Even though we mostly use external, open-source libraries, the licensing is quite different for some of these. Almost all common open source licenses are attractive to academic institutions from the viewpoint of free software development. This is not the case for commercial institutions, as a set of licenses such as the GNU General Public License (GPL) demand derivative work, e.g., via code inclusion or library linking, also to be free and open source. In case of collaborative research with commercial institutions, whether the research is funded by the NIH or industrial sources, the use of such licenses can be a problem even in academic institutions. Of the libraries used in our NeuroLib, this is the case for Qt, FFTW and Xmedcon. For many projects, we have thus moved to use alternatives (FLTK or KWWidgets instead of Qt, ITK instead of Xmedcon) with simpler, less "open-source-contagious" licenses, such as Berkley-style licenses.

For similar reasons of incompatibility with NIH funded industrial collaborations, we have encountered problems in using open-source software that contains patented methods. Our laboratory has not yet patented any of its methods and is not interested to do so in the future. Aside from the general argument that software is essentially mathematics, which in turn cannot be patented [5], industrial leaders in the open-source community have also argued that releasing patented methods in open-source libraries is akin to advertising, as the open source community popularizes the methods.

7. CONCLUSION

In this paper, we presented our UNC NeuroLib and our software development environment, which is based on open source libraries and tools. The UNC NeuroLib features a series of command-line and graphical user interface (GUI) tools established and validated in many neuro-imaging studies, as well as C++ libraries that are supporting these tools. We also discussed briefly our position and solutions to deal with issues of unpublished research and licensing.

There are differences between our development and industrial strength tools. As academic software development needs some degree of flexibility, API changes are not uncommon. This makes it more difficult for external developers to rely on the libraries offered in the NeuroLib repository. But our goal mainly is to offer a extensive development process to our developers leading to stable, validated tools, which are disseminated to clinical collaborators. In our experience, this setup fits this purpose excellently, as the development time is shortened, libraries and tools are well maintained, and research team interaction is enhanced. We are highly motivated and committed to stay with it.

8. REFERENCES

[1] W. Schroeder, K. Martin, and W. Lorensen, *The Visualization Toolkit, Third Edition*, Kitware Inc., 2004.

[2] L. Ibanez, W. Schroeder, L. Ng, and J. Cates, *The ITK Software Guide*, Kitware Inc., 2003.

[3] David E Rex, Jeffrey Q Ma, and Arthur W Toga, "The LONI Pipeline Processing Environment," *NeuroImage*, vol. 19, pp. 1033–1048, 2004.

[4] C. Goodlett, I. Corouge, M. Jomier, and G. Gerig, "A Quantitative DTI Fiber Tract Analysis Suite," *The Insight Journal*, vol. ISC/NA-MIC/MICCAI Workshop on Open-Source Software, 2005, Online publication: http://hdl.handle.net/1926/39.

[5] B Klemens, "Software Patents Don't Compute," *IEEE Spectrum*, July 2005.