Open source software can benefit imaging applications

Phillip Laplante

Open source imaging projects can reduce costs and improve quality provided they are properly evaluated.

Open source software (OSS) refers to programs whose source codes and executable files are made available on the Internet to the general public with very few or no restrictions. As a result, OSS is becoming widely used and transferred. There are thousands of on-going open source projects, including games, programming languages, tools, and enterprise (i.e., organizational)-level solutions. Clones of several well-known desktop and office applications are also available. Some of the best-known OSS projects are the Linux operating system, the Firefox Web browser, and the Apache Web server.

OSS is free to use or redistribute, provided that the terms of the license agreement are respected. This is usually understood to mean that any software derived from the open source code can only be redistributed along with the original source code while complying with its license. There are many types of licenses, and legal advice is necessary before incorporating any OSS in a product intended for release (though not for internal use).

The benefits of using OSS are numerous, starting with unrestricted access to sophisticated code and significant cost savings. Another potential benefit is defect transparency, highly dependent on the vitality of the project community. These open source communities consist of volunteers who contribute code or act as testers and beta users. Individuals get involved for a variety of reasons. Some are achievers who want to contribute something ‘important.’ Others want to be part of a close-knit group. As for companies, they tend to allow their employees to contribute code to open source projects, in some instances even turning proprietary projects to open source to benefit from having a range of people use and evaluate their product.¹

Using OSS in imaging applications

There are many open source imaging projects. A search of a popular repository, Sourceforge,² yielded 2656 hits on 1 January 2007. A few examples illustrate the diversity of the offerings.

Java Imaging Utilities,³ for instance, is a well-known and widely deployed library that includes operations for manipulating pixel images in various file formats. The Morphological Image Processing Project,⁴ written in Python, provides tools for reproducing most of the examples found in Hands-on Morphological Image Processing, by Dougherty and Lotufo.⁵ The Comparametric Toolkit⁶ is C- and Perl-based software for manipulating high-contrast images using known or generalized camera response functions. Finally, the Microscopy Image Processing Package,⁷ written in C++, is used to reconstruct 3D biological specimens obtained by transmission electron microscopy.

OSS can be tailored for individual purposes. For internal use, one need only download the software and use it as obtained. In the case of pure libraries, appropriate interfaces must be built. Alternatively, source code can be downloaded and the software modified. Users must decide whether to channel software modifications back into the project, which requires a vetting process. If the software is going to be released, licensing provisions need to be strictly followed. In some cases, only parts of the software are harvested from the project and introduced into a proprietary application or even combined with other OSS. Again, if the resulting product is intended for release and distribution, adherence to the license agreement is required.

Evaluating OSS

A commitment to using an OSS package can represent a major decision. Many company executives curtail or forbid the use of OSS for fear of losing intellectual control of the software in question. A second issue is that many OSS projects start up and then stall or halt entirely, while others are too buggy and unreliable. However, a little research before making the decision to commit to any OSS and ongoing monitoring of the project can significantly mitigate these risks. Assessing an OSS project should include the following considerations.

Longevity: How well established is the project? Does it have many (substantial) releases? Adoption: Is the software widely

Continued on next page
used? Are there commercial products that are also based on it? Bug diagnosis and repair: How regularly does the project community track and fix bugs? Quality of the development group: Is it diverse, cooperative, and well managed? This aspect can be hard to evaluate. Quality of the code: The software should improve, not decay with time.

Open source tools can also be used to mine information on project activity and investigate these metrics. Many of the repositories themselves now provide such capabilities. If need be, experts can be hired to provide guidance through the selection and monitoring processes.

Conclusions
OSS is not an option for everyone or all situations. Used properly, however, it can provide a wealth of cost-saving and reliable solutions. If care is taken before and during its adoption, OSS can and should be part of a company’s software portfolio.

Author Information

Phillip Laplante
Great Valley School of Graduate Professional Studies
The Pennsylvania State University
Malvern, PA
http://www.personal.psu.edu/pal11

Phil Laplante is a professor of software engineering at Penn State University. His research interests include quality improvement in imaging software and OSS. He holds BS, MEng, and PhD degrees from the Stevens Institute of Technology, and is a SPIE fellow.

References