Starting a State Government Open Source Project

"Officers of government are trustees and servants of the people and it is in the public interest to enable any person to review and criticize their decisions even though such examination may cause inconvenience or embarrassment." 1 V.S.A. § 315

Tom Cort <tom.cort@state.vt.us>
Systems Developer II
Vermont Department of Taxes

What is Open Source Software?

- Free Redistribution
- Source Code
- Derived Works
- Integrity of The Author's Source Code
- No Discrimination Against Persons or Groups
- No Discrimination Against Fields of Endeavor
- Distribution of License
- License Must Not Be Specific to a Product
- License Must Not Restrict Other Software
- License Must Be Technology-Neutral

What is State Government Software?

- Developed and maintained “in house” by one or more state agencies or departments
- Developed and maintained by state employees or state contractors
- Used by one or more state agencies or depts
- “Owned” by the state (no 3rd party copyright agreements or patent claims)
- Development and maintenance funded with public money (taxes)

What is the Gateway Project?

- Implements Streamlined Sales Tax web services.
- Provides a web interface.
- Provides client applications.
- Designed to be an extensible framework upon which future tax software can be built.
- Modernized eFile (MeF) coming soon!

- Project Website: http://gateway.sf.net/

Where to begin

- Like IT infrastructure changes, development process changes are usually rolled out gradually.
- Be realistic about expectations.
- Start out with one project and see how it goes.
- Some people may have never heard of OSS. How you handle this will influence how they think and feel about open source software.
- Educate everyone on the team about what open source is and what it means. See ProducingOSS.com for a free eBook.
- Try to get at least one person on the project team who has open source experience.
- Be prepared for skepticism and criticism.

What kind of project makes a good candidate for open source?

- Not written in a proprietary language like VB.net. Do compilers exist for GNU/Linux?
- Does not depend solely on proprietary tools like Visual Studio. Can it be developed and built with open source tools?
- Does not depend on proprietary libraries. Are all the dependencies open source?
- Does not depend on proprietary software to run. Are there any closed databases or closed apps that are needed to run the software?
- Generally useful. Would at least 3 other people find some or all of the software useful?

In the beginning...

- Gateway started out as a closed source application.
- Built using open source tools (subversion, eclipse, apache axis, saxon, mysql, etc).
- Small development team.
- Rumors that one day the code *might* be open.

Getting approval vs just do it

- You can either ask management for permission to release the code as open source software or you can just release it on your own and hope no one gets mad or fires you.
- Both ways work!

- Getting formal approval is best for business critical applications.
- Just releasing the code is best for “one off” projects and patches.

- The gateway was approved by management.

Arguments for the Gateway

- Cost - It takes very little effort and doesn't cost the state anything to publish the source code, we use a free service called Source Forge. If we receive source code contributions from outside developers it will actually result in a savings because the contributed code is code that Vermont didn't have to pay developers to write.

Arguments for the Gateway

- To be a good Neighbor - Many states are implementing Streamlined Sales Tax and other XML-based electronic return systems. Sharing the code with fellow states would be a nice thing to do because it helps them accomplish their goals while reducing their costs. Additionally, other Vermont agencies and departments could adapt and enhance the code to meet their needs.

Arguments for the Gateway

- Education - Vermonters who are studying Java software development could learn by studying our code. Our code has a lot of interesting pieces in it that aren't directly related to taxes (our event logging system, authentication system, unit test setup, etc). Developers of open source software could legally copy and use the code in their projects.

Arguments for the Gateway

- Better Software - With more people using our code and more developers looking at our code, bugs would be more easily spotted and fixed. Sharing the code with the transmitters who send us data allows them to test changes to their software and debug problems more easily. They could examine exactly what our code does with their input and setup their own private testing environment.

Security Concerns

- Open source code means anyone can download, inspect, and test your software for security vulnerabilities.
- Anyone could help you fix a problem and anyone could exploit a problem.
- Both closed source and open source applications have security holes that can be found and exploited by anyone.
- IMHO, neither methodology is “better” or more secure. Always code with security in mind.

- If the software is internet accessible, think about what data lives on that server.

Security Disclosure Policy

- Decide on what your security disclosure policy will be before you find out about a security issue.
- Full Disclosure
- Full Disclosure with Patch
- Staggered Disclosure
- Limited Disclosure
- Silent Fix

Choosing a license

- OpenSource.org
- www.fsf.org/licensing/licenses/index_html

- Read all of the licenses at OpenSource.org
- Figure out what you want the license to do (copyleft or not, notice files or not, GPL compatible or not, etc).
- Look at what other like software uses for its license. (Example, Java usually is Apache 2.0)
- Stop license proliferation, choose an existing license that meets your criteria.
- Don’t start a war over the license.

Determine copyright information

- Find out the exact text to use for the copyright line.
- Usually the website for your department or agency has a copyright message you can copy.

- Decide if contributors must assign copyright.

Choosing a development model

- Core team
- Committers and contributors
- Voting/Polling
- Benevolent Dictator
- Define your own?

- Document this for later. It is important that external devs know how to submit patches.

Preparing the source code

- Read every line of code.
- Make sure there is a copyright header in every source file.
- Make sure no passwords, private keys, host names, or other private/semi-private information exists in the code.
- Make sure the code looks good (non-vulgar comments, indentation, no ugly hacks, etc)

- Tip: use svn:externals in subversion to allow you to keep a set of private configuration and password files.

Version Numbering

- A lot of numbering schemes exist.
- Single number: udev-115
- Major, Minor: gateway-3.0
- A.B.C[.D]: linux-2.6.22.6

- The exact scheme doesn’t matter
- Pick one that you’ll stick to
- Document the scheme on your download page
- Make it clear what code is production ready and what is beta/alpha/rc.

Preparing the documentation

- Must explain what the software is and what it does.
- Must describe how to install and run the software.
- Must explain how to build from source.

- If possible, it should describe the source code (javadoc, doxygen).

- Is best with screen shots, commands, and examples.

Preparing the website

- You need a website!
- Should include all documentation.
- An FAQ to answer repeat questions.
- A download page so people can get the software.
- Project News so people know what’s happening.
- A project road map so people know what’s going to happen.
- A guide for people interested in contributing.
- Links to other project resources (bug tracker, continuous integration, mailing lists, svn/cvs).
- A clear point of contact for discussion, getting help/support, suggesting features/improvements and reporting bugs.

Preparing other services

- IRC – chat room. May not be the best idea for state employees to be chatting. See Freenode.net
- Mailing lists. Make sure all project members are subscribed and read the incoming mail.
- Source control. Educate all of the developers on the best practices and common features. All developers should know how to add code, remove code, create a patch, apply a patch, revert a change, create a branch, merge code from two branches, and resolve a merge conflict.
- Wiki. Be careful that it isn't interpreted as the official word of the state. Watch out for spam.

Preparing the first release

- Make sure your copy of the code is up to date.
- Make sure the documentation is up to date.
- Create a source package.
- Optionally create 1 or more binary packages.
- Packages should be in standard formats (no 7z please). Windows installer should be .exe or .msi. Linux packages should be .rpm, .deb, .ebuild, etc. Linux sources should be .tar.gz or .tar.bz2. Windows sources should be .zip.
- Test the software packages you just made and the documentation before uploading them.

Finding a Project Hosting Provider

- 4 Options: current agency/dept website, self-managed infrastructure, canned OSS hosting, custom OSS hosting.
- current agency/dept website is usually too limited or requires too much paper work to get setup.
- self-managed infrastructure requires hardware and someone with enough time to maintain it.
- canned hosting is free and gives you complete control over everything you need to run a project.
- custom OSS hosting can be hard to get, but its much more flexible than the canned stuff.

Free Canned OSS Hosting

- Source Forge - sourceforge.net
- Savannah - savannah.nongnu.org
- Gna! - gna.org
- berliOS - berlios.de

Custom OSS Hosting

- OSU OSL - osuosl.org

Being a good community member

- Report bugs and submit patches to other projects.
- Join mailing lists and help people.
- Develop in the open.
- Acknowledge contributors and developers when they contribute. Add a note in the change log and commit log, thank them by e-mail or on a public mailing list, thank users for submitting bug reports, etc.

Implementing Open Standards

- A standard exists for just about everything.
- Search for standards before implementing new storage or data transfer functionality.
- Choose only open standards.
- Implement all required parts of the standard correctly.

Gateway Demo

- Demo Website: gateway-demo.osuosl.org
- Gateway SSTP Client GUI
- SSTPong Plus - http://sstpong.sf.net/