Application Development

Jed Moffitt
Lori Bowen Ayre

Evergreen International Conference
Indianapolis, Indiana
April 26, 2012

How it works

1. Verify no one else working on the idea
2. Write requirements
3. Find developer
4. Introduce developer to EG dev community
5. Flesh out dev requirements into dev plan
6. Track progress
7. "Accept" code from your developer
8. Submit code to community devs
9. Keep fingers crossed that new code will make it into the next release

Where to Find Developers

Companies doing development work on Evergreen already (besides Equinox of course!)

- http://edoceo.com – Main contact is David Busby who has been working with KCLS

- http://libtech-consult.com - Main contact is Kyle Banerjee who has been working with Michigan on NCIP support for Evergreen

- http://catalystitservices.com/ - Jeffrey Bond introduced himself on general mailing list.

Other people I've talked to:

- Thomas Berezansky (tsbere@mvlc.org)

- Scott Prater (sprater@gmail.com)

- Vicent Mas (uvemas@gmail.com)

Search resumes at http://Craigslist.org

Post job listing at http://perl.meetup.com/

or http://mysql.meetup.com/

or http://jobs.perl.com/.

Attend **Open Source Bridge** (Portland, June 26-29, 2012)

or **LITA National Forum** (Columbus, OH October 4-7, 2012)

Computer Engineer Barbie®

Evergreen Developer Skill Set

Main programming languages

- Perl 5
- JavaScript
- PL/pgSQL and PL/Perl
- HTML and CSS

Other languages and frameworks used:

- C
- XUL
- Dojo 1.3 (framework)
- Template Toolkit

Source: http://evergreen-ils.org/dokuwiki/doku.php?id=dev:contributing_code

Are you sure this is a new idea?

- Share your development idea with the community via mailing lists (-dev and –gen)
- Search Launchpad – anything like it there already?
- Once you've verified that no one else is already working on this idea….proceed to next step.

Create a Requirements Document

DEVELOPMENT NAME

Background Information:

[Library: Provide any background information about the existing software, library environment or situation that will help the developers understand the purpose of the development project]

Use Case:

[Library: Describe the reasons this development is needed and how it will be used by the end users]

Requirements:

[Library: Describe the functional requirements of the development in detail]

Getting your Developer Started

1. Review the Requirements Document
2. Provide any additional background info
    - Explain what an ILS does?
    - Review Use Cases
    - Discuss acceptance testing tests and methodology
3. Facilitate introductions with EG developer community – find a mentor.
4. Have developer write a Development Plan based on your requirements document
5. Draft a Development Plan and Contract (based on Plan)

Requirements Document → Development Plan
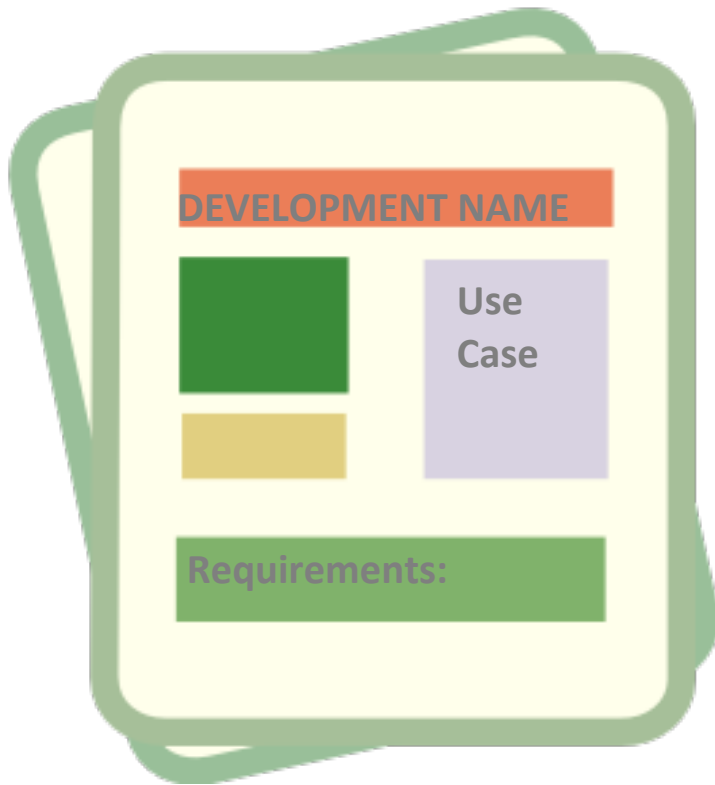
Add these sections to your Requirements Document:

Objects Affected:

[Library and Developer: List all Evergreen components that will be affected by code changes]

Acceptance Criteria:

[Library and Developer: Describe the tests that will be performed on the supplied code to determine that the project has been successfully completed]

Copyright Issues

- Agree on how you'd like to copyright the code
- Valid options:
  - In developers name
  - In library's name
  - In developer and library's name
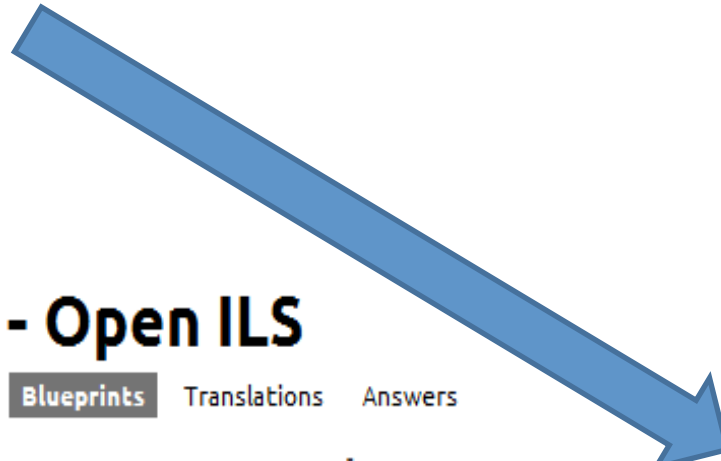- Make sure all work is copyrighted as "GPL v2 or later"

Submit Development Plan to Community

Once it is clear that the development idea is new, submit a Development Plan…

What Developer Needs to Do	Example
Add the development information to the wiki in the dev:proposal name space.	http://evergreen-ils.org/dokuwiki/doku.php?id=dev:proposal:patron_statistical_categories
Add a Blueprint entry at Launchpad with the basic overview. Link the Blueprint entry to the wiki page.	https://blueprints.launchpad.net/evergreen/+spec/enhanced-patron-stistical-categories
Send the basic overview and links to the Blueprint and wiki page to the Evergreen development mailing list with a subject line beginning with Feature Proposal.	http://libmail.georgialibraries.org/pipermail/open-ils-dev/2011-November/007700.html

# Subscribe to your Development in Launchpad

Git 101
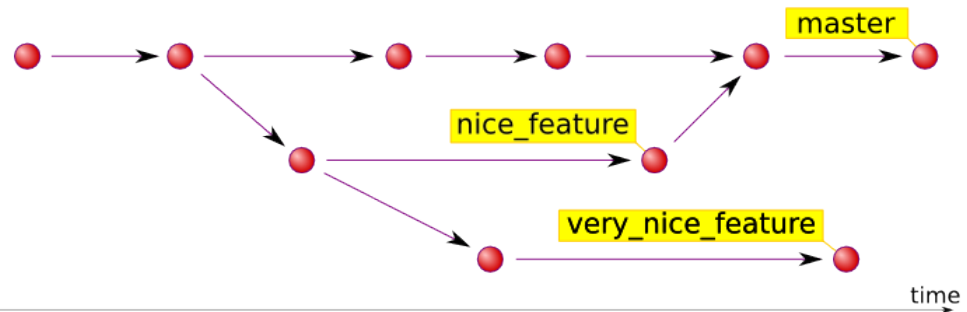
- Version control system
  for Evergreen code and documentation

- Git repositories contain chunks of code arranged in "branches"

- "Master" refers to the branches of code currently staged for the next release

- Developers make their work available as "topic branches" which can be pushed and pulled for testing against master

Your Developer and Git

- Developers may have their own Git repository (e.g. GitHub)

- Testing of the code being developed for you will come from your developer's "repo"

- When the code is approved by Library, developer submits code to Dev Community
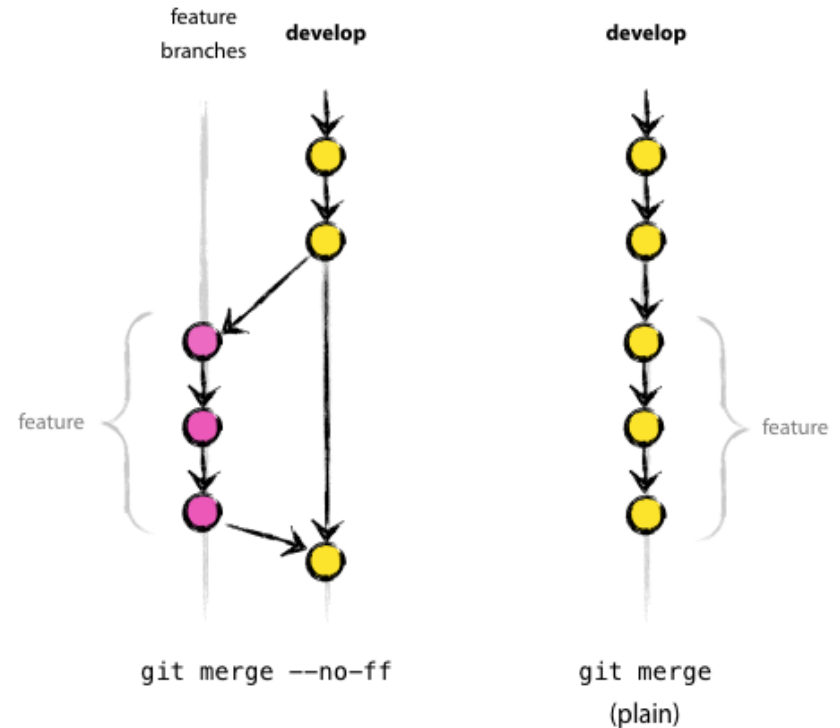
Evergreen Code Repositories

- Evergreen Git Repositories
  - Evergreen http://git.evergreen-ils.org/?p=Evergreen.git;a=summary
  - OpenSRF http://git.evergreen-ils.org/?p=OpenSRF.git;a=summary
  - Documentation https://github.com/rsoulliere/Evergreen-DocBook

- Other public Git repositories
  - Equinox
  - Sitka
  - MVLC
  - Your developer!

Source: http://evergreen-ils.org/dokuwiki/doku.php?id=dev:code_repositories

Protocol for Contributing Code

1. Request working branch access from one of the Git server administrators
    - Thomas Berezansky
    - Galen Charlton
2. Publish working branch - raise awareness of available branch and link to Launchpad bug ticket for more info.
3. Dev's mentor (and others in the Community) will sign off on changes, or make other final suggestions.
4. One of the core committers will eventually merge code into master.
5. Celebrate your contribution to Evergreen!

Source: http://evergreen-ils.org/dokuwiki/doku.php?id=dev:git