The Eclipse Development Process for Committers

The Eclipse Development Process

eclipse.org/projects/dev_process

Agenda

- Definition of an Eclipse Project
- The Eclipse Development Process
- Working as a Committer
- Releases and Reviews
- Project Management Interface
- Other Stuff You Should Know

What is an Eclipse Project?

TL;DR

An Eclipse Project…

- Conforms to the EDP and Eclipse IP Policy;
- Calls itself an “Eclipse Project” and conforms to Eclipse Foundation Branding Guidelines;
- Operates independently from any specific vendor;
- Uses infrastructure provided by the Eclipse Foundation for **core content**

Eclipse Development Process

Defines…

- The Open source rules of engagement
- Governance structure and definitions
- Relationships and roles

Provides a framework for project lifecycle management

- Releases and reviews
- Interaction between projects and the community

The Eclipse Intellectual Property Policy

The Eclipse IP Policy is concerned with establishing practices to mitigate legal risks associated with building products based on Eclipse software.

- Licensing
- Copyright
- Provenance tracking

Trademarks and Branding

The Eclipse Foundation holds all project trademarks on behalf of the community.

- Trademarks must be managed to remain valid
- Cycle of value
  - The Eclipse brand adds value to project trademarks
  - Project trademarks add value to the Eclipse brand

Source Code is Core Content

Source code must be *actively developed* on infrastructure provided by the Eclipse Foundation.

- Official project Git repositories must be the central authority
- Committers must push regularly to official project repositories
- Test code is source code

Level playing field; open to all comers.

Documentation is Core Content

Documentation is a first-class project resource.

- Documentation maintained using the same rules as source code
- Must be deployed on EF infrastructure
  - EF website, downloads, documentation server, …
- May *also* be deployed elsewhere
  - e.g. Community website, ReadTheDocs

Issues/Bugs/Tickets are Core Content

Issues must be tracked using infrastructure provided by the Eclipse Foundation.

- Bugzilla
- GitHub Issues

Project Dev List is Core Content

The Project Dev list is the main means of contact between the Eclipse Foundation (staff and automatic services) and project team.

- Formal EF process must happen on the devlist
  - practical: votes, etc.
- Other project communication must be captured in a public channel

Non-Core Content

Other content must be transparent and open but not necessarily owned/operated by the EF

- User forum discussions, presentations
- Builds and downloads
- Examples, blogs, social media handles, …

“The Keys” Must be Kept Safe

Core infrastructure must be *immune* from domination by any single individual or organization.

- The EF must hold the keys for core infrastructure
- Pick up and carry on

Ownership of non-core infrastructure must be shared

- Must be able to survive departures of team members

The Eclipse Development Process

- Open source rules of engagement
- Governance, structure, definitions, reviews
- General framework for projects
- Day-by-day development rules/process is defined by the project

Open Source Rules of Engagement

- Transparent
  - Project discussions, minutes, deliberations, project plans, plans for new features, and other artifacts are open, public, and easily accessible
- Open
  - The same opportunity to all
  - Everyone participates with the same rules
- Meritocracy
  - The more you contribute the more responsibility you will earn
  - Committer/Project lead elections

Vendor Neutrality

- Open to all comers, including competitors!
- Vendor neutral resources and services
  - Project website
  - Issue tracker
  - Mailing lists and forums
  - Downloads
  - VCS/Git
- Committers must work in the open
- Freedom of action
  - GitHub services backed up on EF servers

Three Communities

- Users
  - Users are, well... users
- Adopters
  - Individuals, groups, organizations
  - Build products, extensions, based on your project
- Developers
  - Contributors, committers

Writing code is fun, but...

- Open source rules of engagement
  - Transparency, openness, meritocracy
- Have project-specific diversity goals
  - Building diversity takes work
- Actively court contributors
- Be responsive when they do come
  - “Kill with kindness”

Project Leadership

- Project lead(s)
  - Management, technical leadership
  - Primary liaison with PMC and EMO
- Project Management Committee (PMC)
  - Technical approval of IP contributions
  - Approval of releases
  - Project oversight, general responsibility
- EMO
  - Overall responsibility to manage processes and maintain services
  - Eclipse Foundation Staff, Architecture and Planning Councils

Working as a Committer

Committers and Contributors

- Committers
  - Have write access to project resources (Git repos, downloads, etc.)
  - Must complete committer paperwork
    - Individual or Member Committer Agreement (ICA/MCA)
- Contributors
  - Contributions must be accepted by a committer
  - Must electronically sign the Eclipse Contributor Agreement (ECA)
  - May be invited to join the project as a committer

Eclipse Contributor Agreement

- Contributors must digitally sign the ECA
  - Confirms that contributor agrees to comply with the DCO
- Developer’s Certificate of Origin
  - You have authored 100% of the content
  - You have the necessary rights
  - Provided under the project license(s)
  - Your contributions are public
- Contributors must “sign-off” on each commit

https://eclipse.org/legal/ECA.php

Contributions via Git

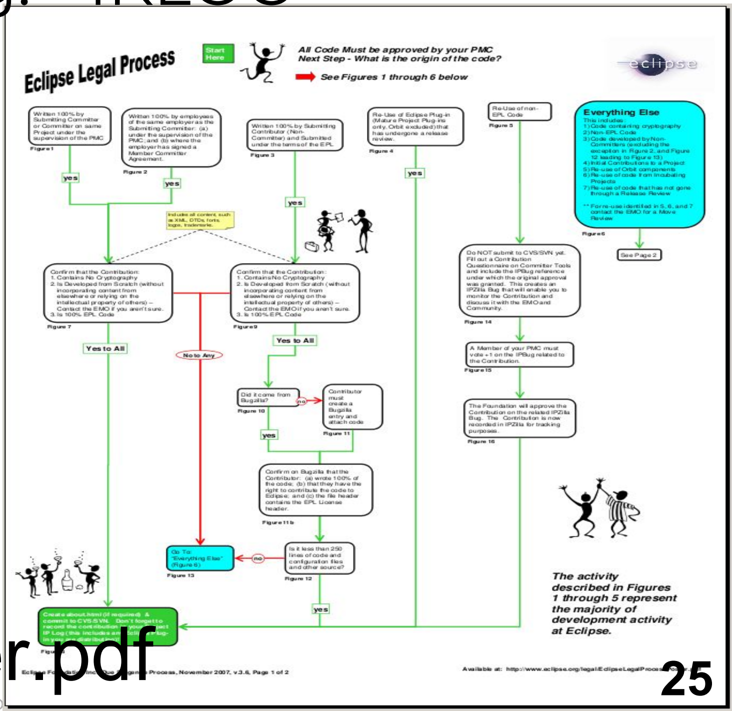
```
commit d6cf52411377a039fc2906378711091a26e932cb
Author: Some Body <somebody@somewhere.com>
Date:   Wed May 29 16:17:36 2013 +0200
    [123456] Hide unwanted common navigator action bar items

    This change hides unwanted 'Link with Editor' and 'Customize View...'
    items from the local toolbar and the view menu.

    See bug 409722 for restoring the feature of activating the editor when
    changing the selection using 'Link with Selection'.

    Bug: https://bugs.eclipse.org/123456
    Change-Id: Ia2bd5091303d1b0a738157effc24e4dac5a7d0c7
    Also-by: Some Bodyelse <somebodyelse@nowhere.com>
    Signed-off-by: Some Body <somebody@somewhere.com>
```

Handling Git Contributions

- Git/Gerrit will check for ECA/sign-off
- GitHub will warn if ECA or sign-off is missing
- Engage with IP Team if required, e.g.
  - Contribution has >1 KLOC
  - You’re not sure of the provenance
  - Not certain that the ECA/DCO terms are met

Intellectual Property
Due Diligence Process

- All code in project VCS or downloads
  - All third-party libraries must be vetted by the IP Team
  - Initial contribution
  - Many contributions must be vetted as well, e.g. >1KLOC
- Vetting
  - Known pedigree
  - Permissive license; compatible with project license (e.g. EPL)
- Due diligence process defined



Eclipse Legal Process

Start Here

All Code Must be approved by your PMC
Next Step - What is the origin of the code?

See Figures 1 through 6 below

Written 100% by Submitting Committer or Committer on same Project under the supervision of the PMC

Figure 1

Written 100% by employees of the same employer as the Submitting Committer: (a) under the supervision of the PMC; and (b) where the employer has signed a Member Committer Agreement.

Figure 2

Written 100% by Submitting Contributor (Non-Committer) and Submitted under the terms of the EPL

Figure 3

Re-Use of Eclipse Plug-in (Mature Project Plug-ins only, Orbit excluded) that has undergone a release review.

Figure 4

Re-Use of non-EPL Code

Figure 5

Everything Else
This includes:
1) Code containing cryptography
2) Non-EPL Code
3) Code developed by Non-Committers (excluding the exception in Figure 2, and Figure 12 leading to Figure 13)
4) Initial Contributions to a Project
5) Re-use of Orbit components
6) Re-use of code from Incubating Projects
7) Re-use of code that has not gone through a Release Review

** For re-use identified in 5, 6, and 7 contact the EMO for a Move Review

Figure 6

See Page 2

Includes all content, such as XML, DTDs, fonts, logos, trademarks.

Confirm that the Contribution:
1. Contains No Cryptography
2. Is Developed from Scratch (without incorporating content from elsewhere or relying on the intellectual property of others) – Contact the EMO if you aren't sure.
3. Is 100% EPL Code

Figure 7

Confirm that the Contribution:
1. Contains No Cryptography
2. Is Developed from Scratch (without incorporating content from elsewhere or relying on the intellectual property of others) – Contact the EMO if you aren't sure.
3. Is 100% EPL Code

Figure 9

Did it come from Bugzilla?

Figure 10

Contributor must create a Bugzilla entry and attach code

Figure 11

Confirm on Bugzilla that the Contributor: (a) wrote 100% of the code; (b) that they have the right to contribute the code to Eclipse; and (c) the file header contains the EPL License header.

Figure 11b

Is it less than 250 lines of code and configuration files and other source?

Figure 12

Go To: "Everything Else" (Figure 6)

Figure 13

Do NOT submit to CVS/SVN yet. Fill out a Contribution Questionnaire on Committer Tools and include the IPBug reference under which the original approval was granted. This creates an IPZilla Bug that will enable you to monitor the Contribution and discuss it with the EMO and Community.

Figure 14

A Member of your PMC must vote +1 on the IPBug related to the Contribution.

Figure 15

The Foundation will approve the Contribution on the related IPZilla Bug. The Contribution is now recorded in IPZilla for tracking purposes.

Figure 16

Create about.html (if required) & commit to CVS/SVN. Don't forget to record the contribution in your project IP Log (this includes any Eclipse Plug-in you are distributing!)

Figure 8

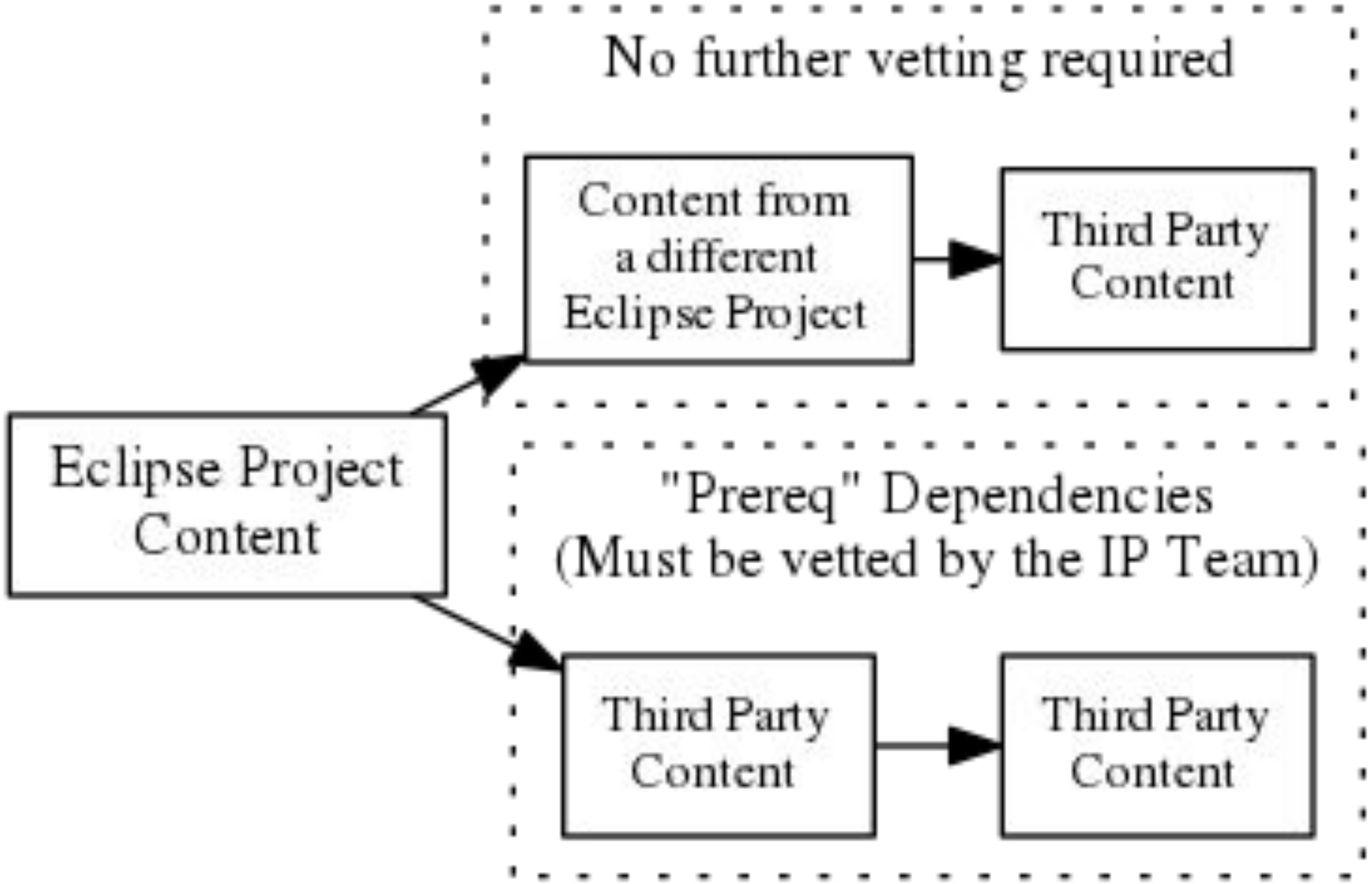
The activity described in Figures 1 through 5 represent the majority of development activity at Eclipse.

Eclipse Foundation, Inc. Due Diligence Process, November 2007, v.3.6, Page 1 of 2

Available at: http://www.eclipse.org/legal/EclipseLegalProcessPoster.pdf

http://eclipse.org/legal/EclipseLegalProcessPoster.pdf

Dependencies


No further vetting required

Content from a different Eclipse Project

Third Party Content

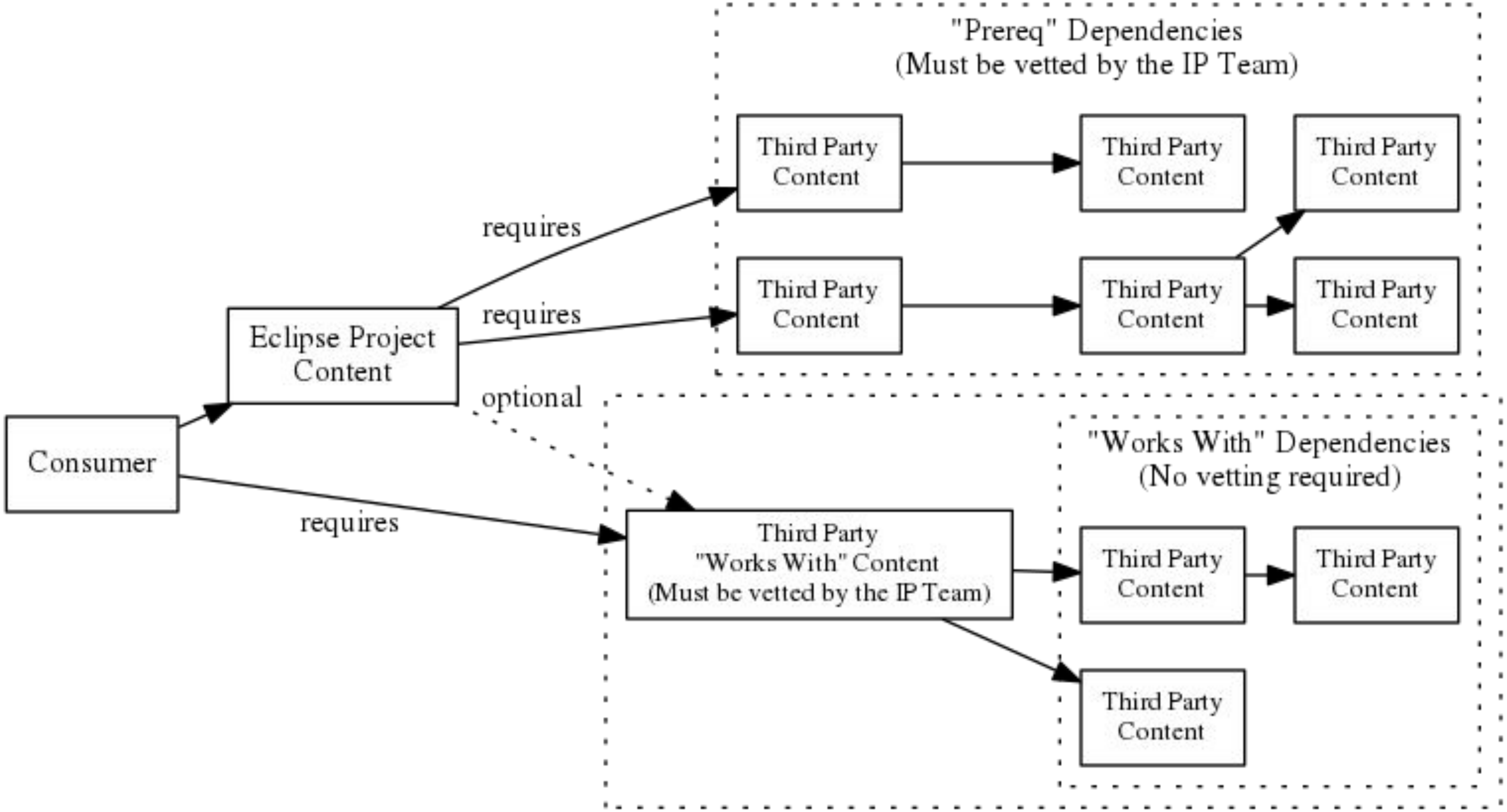
Eclipse Project Content

"Prereq" Dependencies (Must be vetted by the IP Team)

Third Party Content

Third Party Content

Prerequisites and “Works With”



Legal (and Other) Documentation

```
git.eclipse.org/project.git
 ├── LICENSE[.md]
 ├── NOTICE[.md]
 ├── CODE_OF_CONDUCT[.md]
 ├── CONTRIBUTING[.md]
 ├── ...
```

Contribution Guide

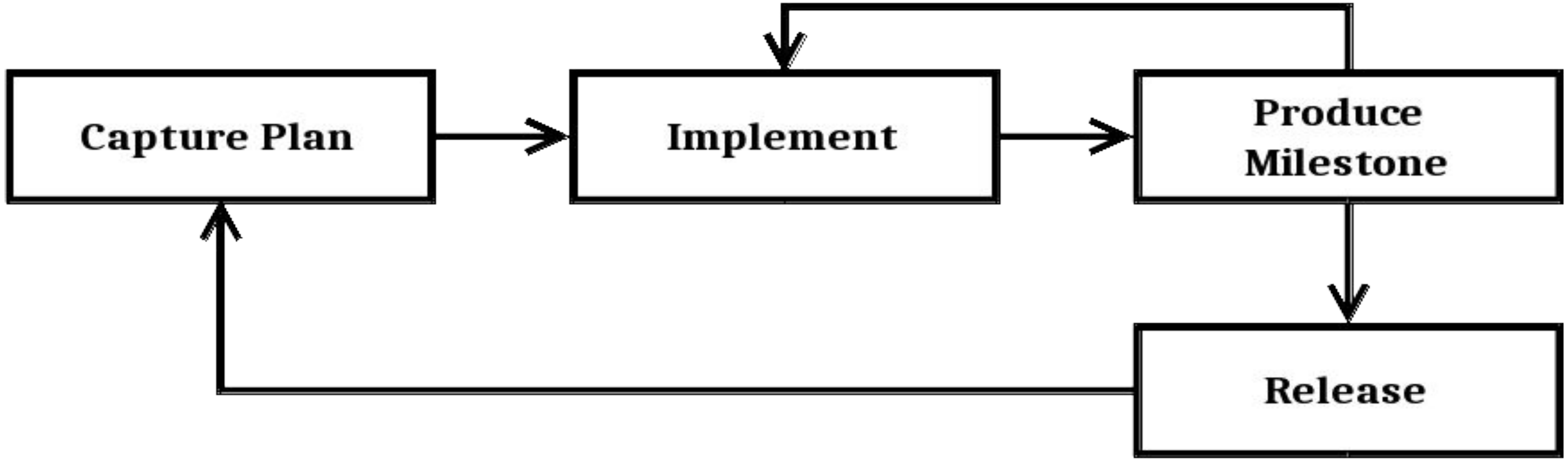
- Git, Gerrit, GitHub, …
- Issue tracking (Bugzilla)
  - “Help wanted” issues
- Project plan
- How to build
- How to engage (mailing lists, forums, issues)
- Spend time with the community

Keep Project Information Up-to-date

- Project and release metadata, website, downloads
- Project code must be buildable
- Maintain a contribution guide
- Keep project communities informed

Releases and Reviews

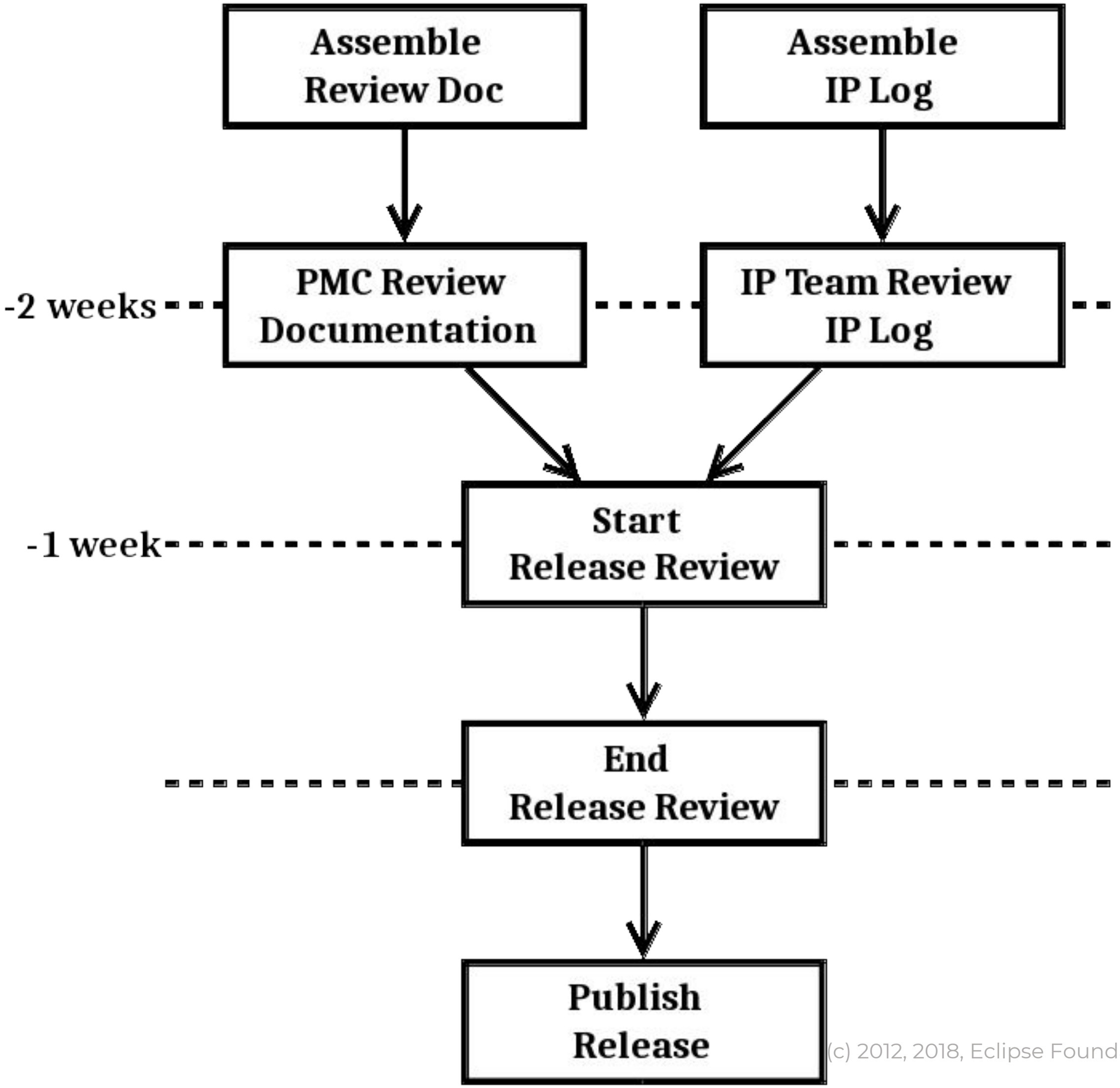
Releases



Release Review

- Major/minor releases
  - Release review
  - IP Log approval
  - Plan to spend time planning/documenting release
- Service/Bugfix-only releases
  - No review
  - No IP Log approval

Release Review

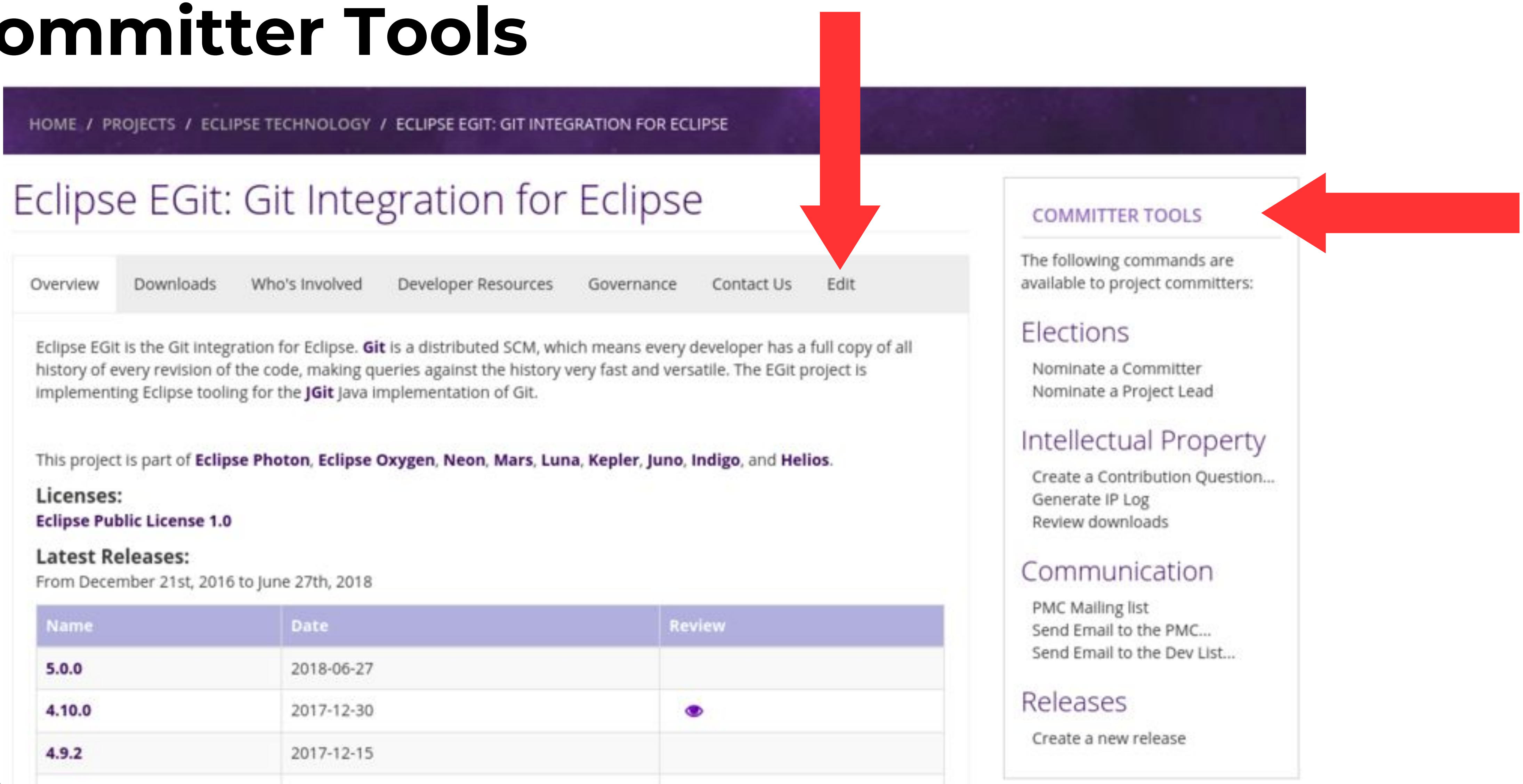


Project Management Interface (PMI)

Project Metadata

- Used for discovery and drives various processes and systems
  - e.g. IP Log generator, notices to members and committers
- Description, scope, logo, technology type
  - “Elevator pitch”
- Source code, builds, issue tracker
- Releases, relationships to other projects, build technologies

Committer Tools



HOME / PROJECTS / ECLIPSE TECHNOLOGY / ECLIPSE EGIT: GIT INTEGRATION FOR ECLIPSE

Eclipse EGit: Git Integration for Eclipse

Eclipse EGit is the Git integration for Eclipse. **Git** is a distributed SCM, which means every developer has a full copy of all history of every revision of the code, making queries against the history very fast and versatile. The EGit project is implementing Eclipse tooling for the **JGit** Java implementation of Git.

This project is part of **Eclipse Photon**, **Eclipse Oxygen**, **Neon**, **Mars**, **Luna**, **Kepler**, **Juno**, **Indigo**, and **Helios**.

Licenses:
Eclipse Public License 1.0

Latest Releases:
From December 21st, 2016 to June 27th, 2018

Name	Date	Review
5.0.0	2018-06-27	
4.10.0	2017-12-30	●
4.9.2	2017-12-15	

COMMITTER TOOLS

The following commands are available to project committers:

Elections

- Nominate a Committer
- Nominate a Project Lead

Intellectual Property

- Create a Contribution Question...
- Generate IP Log
- Review downloads

Communication

- PMC Mailing list
- Send Email to the PMC...
- Send Email to the Dev List...

Releases

- Create a new release

http://projects.eclipse.org/projects/<projectId>

Other Stuff You Should Know

IP Terms

- Initial Contribution
- Contribution Questionnaires (CQs)
- Piggyback CQ
- IPZilla
- Parallel IP

IT Services

- Connect with the Webmaster for IT support
  - Git repositories, builds, downloads
  - Permissions
- Connect via Bugzilla (Community)
- webmaster@eclipse.org

Eclipse Project Handbook

- All-in-one guide for project leads and committers
- Starting a new project
- Forge-specific variations:
  - https://www.eclipse.org/projects/handbook
  - https://www.polarsys.org/documentation/handbook

Incubation Mailing List

- Connect with other incubating projects
- Connect with mentors
- http://dev.eclipse.org/mailman/listinfo/incubation
- incubation@eclipse.org

Things that We Didn't Talk About

- Committer and Project Lead Elections
- Graduation from Incubation Phase
- Committer Paperwork
- Starting an open source project
- IP Logs

Links

- Eclipse Development Process
  - http://www.eclipse.org/projects/dev_process
- Eclipse Project Handbook
  - http://www.eclipse.org/projects/handbook

The End