CSCI-4470 Open Source Software

Summer 2022

Rensselaer Polytechnic Institute

Department of Computer Science

Licensing

Licensing

Our Lecture Slides have greatly benefited from the previous classes taught by Dr. William Schroeder and Dr. Luis Ibanez at RPI,

And by Professor Mukkai Krishnamoorthy who succeeded them in teaching the course

Figure 1: image

Reading Material

Reading Material

- The Cathedral and the Bazaar by Eric Raymond
  - http://www.unterstein.net/su/docs/CathBaz.pdf
  - Sections 1,2,3, 4 and 9
- Free Culture (Introduction and Chapter 4)
  - https://github.com/rcos/CSCI-4470-OpenSource/blob/master/Resources/freeculture.pdf
- A (Brief) History of REGEX
  - https://whyisthisinteresting.substack.com/p/the-regular-expression-edition

Why Open Source?

Open Source Issues

- Why Open Source?
- Why not Open Source?
- Difference between Open Source and Free Software
  - http://askubuntu.com/questions/78958/is-there-a-difference-between-free-software-and-open-source-software

Why Open Source?

- Fun
  - Form Communities
  - Engage in a hobby
  - Learning Experience
- Profitable and Successful Business Models
  - Red Hat, Inc.
    - Cygnus Solutions
  - Kitware
  - Service Oriented Business Model
- Altruism – serve the planet – If the world improves, you improve
- Open Science and Engineering
  - Open – Access to Data
  - Open – Access to Source Code
  - Collaboration oriented
- Intellectual Freedom
  - Idea is a property – if and when an idea gets patented, you no longer have the freedom to use it
- Open Medicine
  - Data
  - Chemical Compounds
  - Effectiveness of Treatment
- Scalable Software Development
  - Eric Raymond's The Cathedral and The Bazaar
    * "Open source peer review is the only scalable method for achieving high reliability and quality"

Why Not Open Source?

- Intellectual property concerns
- Chaotic development environment
  - Volunteer based
  - Distributed
  - No clear authority
- Hard to change code
  - Public API visible
  - Internal structure visible
- Benefits are a function of community size

- Proprietary business model
  - Better understood
  - Greater potential for $$$

The Open Source Model

The Cathedral and the Bazaar

- Cathedral Model (commercial world)
  - Development by a single person or by a chosen committee
- Bazaar Model (linux world)
  - Contribution by people – but used in alpha, pre alpha stage by a lot of people – Release early and release often
- Every Good Work of Software starts by scratching a developer's itch. - Most students projects tend to be on games!
- Good Programmers know what to write; Great ones know what to rewrite and reuse!
- Plan to throw one away; you will anyhow (Fred Brooks, "The Mythical Man Month")
- To solve an interesting problem, start by finding a problem that interests you.
- If you have the right attitude, interesting problems will find you (be part of a community)
- Release early, Release often. And listen to your customers.
- Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging.
- When you lose interest in a program, your last duty is to hand off to a competent successor.
- Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix will be obvious to someone
- Smart data structures and dumb code works a lot better than the other way around.
- If you treat your beta testers as if they are your most valuable resource, they will respond by becoming your most valuable resource.
- Any tool should be useful in the expected way, but a truly great tool lends itself to uses you never expected.
- When your language is nowhere near Turing-complete, syntactic sugar can be your friend.

Software Management Functions

Software Management has five functions.

1. Define goals and keep every one pointed in the same direction.
2. To Monitor and make sure critical details do not get skipped.
3. To motivate people to do boring and drudgery work.
4. To organize the deployment of people for best productivity.

5. To marshal resources needed to sustain the project.

Getting Started

Establishing An Open Source Project

- Create a clear vision (requirements doc) – Technical domain, Software Stack/Tools
- Involve team oriented people (big egos are big problems)
- Identify leadership/management structure (Methods to break conflicts)
- Establish an effective software process
- Define Communication protocol
  - Chat room
  - Developer mailing list
  - Periodic face-to-face meetings
- Avoid Pitfalls
  - Establish Core architecture early
  - Start development with a few key people
  - Start testing early
  - Use version control
  - Lock up language.

Use External Tools

- Use external open source tools and libraries
- Redevelopment is a waste of time (most of the time)

Licensing

- Understand licensing and use a license
- Select the software libraries that use similar (compatible) licenses
- Think about a commercialization strategy
  - Pure support
  - Open toolkits but closed applications
  - Open standards closed implementations
  - Open platforms/closed plug-ins)

Questions and Discussion