An Open Source Work Shop

Luciano Resende (lresende@apache.org)
Haleh Mahbod (hmahbod@gmail.com)

Aug. 2008

Topics

- General knowledge about open source
  - Importance of Open Source
  - What is Open Source
  - License
  - Infrastructure
  - Community

Why open source?

It's "impossible to avoid"

Gartner 2007 Study:

By 2011, 80% of all commercial software will contain open source code

-**Open source impossible to avoid, Gartner says", Network World**

-**http://www.networkworld.com/news/2007/092007-open-source-unavoidable.html**

Forrester 2008 Study:

Study of 2,252 North American and European Software decision makers done by Forrester: 66% are interested in open-source software…. Open source seems to be a tactic for achieving the high priority initiatives. .. And

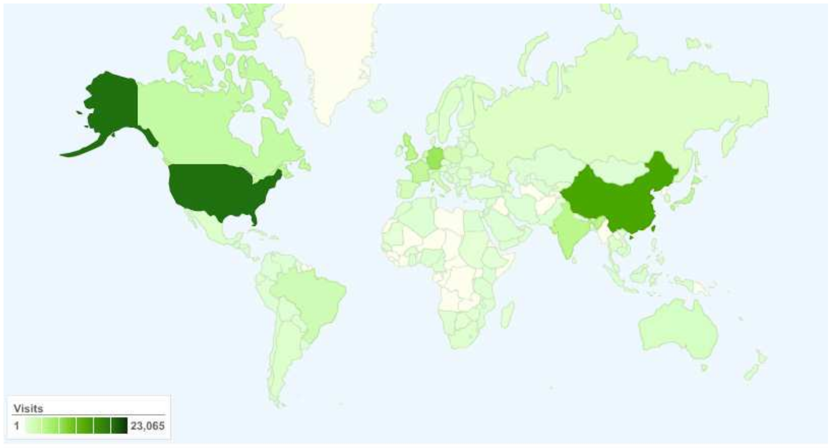
Web 2.0 technologies (such as blogs, wikis and RSS) and

Service Oriented Architecture among their major software initiatives.

-**"Open source software: Just a Means to an End", CIO magazine, March 2008**

Why is Open Source important ?

- Can be a major source of innovation!
  - It unites perspectives from a host of disciplines and brings ideas together from all around the world to
    - Rapidly solve business issues
    - Accelerate technological advancements
    - Stimulate economic growth
    - Enable new business models
- OSS is a good approach for driving emerging open standards
  - Popular open source projects can become the common implementations
- IT can benefit
  - Increased choice and flexibility
  - Lower costs
  - Quick response



What Is Open Source?

- open source refers to software that is published under licenses that defines how source code can be made available to everyone to inspect, change, download, and explore as they wish.

- Open source is not a software methodology
  - It is a way of developing ideas and software collaboratively in the open

- Open source is about software license
  - Defines how the developers develop the software
  - Defines how the users consume the software and contribute back
  - Defines how the community interacts with one another

What characterizes an open source?

- The community of software users and developers interested to develop some idea in the open create the open source project
  - The makeup of each open source project is identified by it's philosophy, it's culture and it's character defines an open source project.
  - Community Philosophy: The software license used for the open source project captures defines its philosophy
  - Community Culture: Where the open source project is hosted can define its community culture. For example Apache promotes a different culture than Eclipse.
  - Community Character:
    - Each project can run differently even within a given open source embodiment. The community decides on how to create, build and maintain the project.
    - The technology offered by a project can also form the character of the project. For a project that develops applications will be different than one that develops an infrastructure.

Open Source License

- Open Source license defines the community philosophy
  - open source refers to software that is published under licenses that defines how source code can be made available to everyone to inspect, change, download, and explore as they wish.

- There are upward of 71 open source licenses which fall into 3 families:
  - Give me credit
  - Give me fixes
  - Give me everything

Open Source License Type – Give Me Credit

- Examples: AL, BSD, MIT
- Typical Characteristic:
  - Derivatives can sub-license
  - May have some conditions
    - No warranty
  - Credit to original authors required
  - Limited control by any one entity
  - Allows for commercial product development
  - Allows for competing services

Open Source License Type – Give Me Fixes

- Examples: Mozilla (MPL), Eclipse (EPL/CPL), LGPL
- Typical Characteristic:
  - Single entity control, still ‘business friendly’. If you modify the code, you need to make the modification available
    - File or derivative based conditions
    - Original author may have special rights
    - Differentiate between source and binary
    - Larger works can be under a different license
    - Encourages incorporation of code into larger works
    - Ensures direct development benefits all

Open Source License Type – Give Me Everything

- Examples: GPL (**GNU General Public License**)
- Typical Characteristic:
  - Also referred to COPY LEFT. If you use it, everything must be under this license
    - Derivative works remain under the license
    - Linked works may also remain under the license
    - Ensures all ‘down stream’ have the same rights
    - All direct development is contributed back
    - Contributors assured code remains open source
    - Encourages a full free software economy
    - Copyright holder retains much control
    - Limits commercial adoption: Forbids distribution for profit
- Software with this type of license cannot be included in Apache projects

Open Source: Most common Licenses

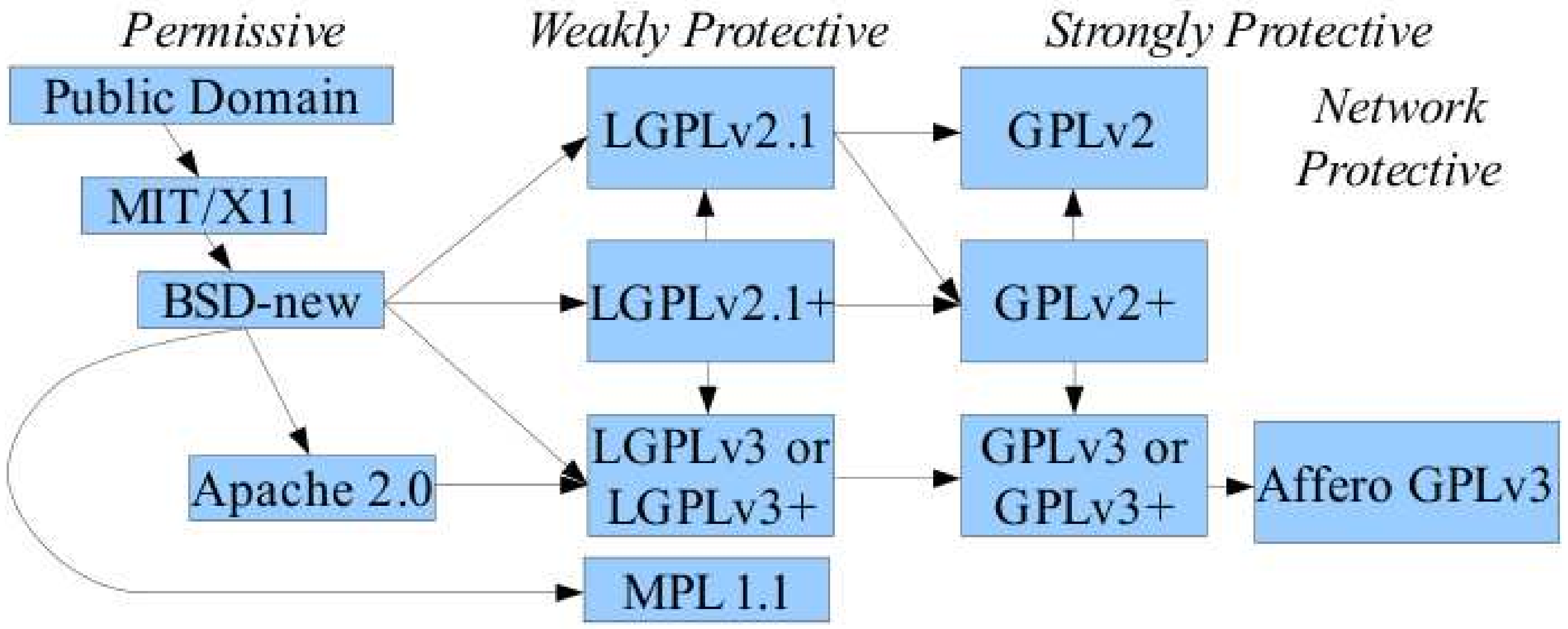
- Apache License, 2.0
- BSD licenses
- GNU General Public License (GPL)
- GNU Library or "Lesser" General Public License (LGPL)
- MIT license
- Mozilla Public License 1.1 (MPL)
- Common Development and Distribution License
- Eclipse Public License
- Artistic Licenses

License Compatibility

- **License compatibility** refers to the problem with software licenses which can contain contradictory requirements, rendering it impossible to combine code from such packages in order to create new software packages.
- Let’s consider the following scenario:
  - Code distributed with license A says :"*modified versions must mention the developers in any advertising materials*“
  - Code distributed with license B says : "*modified versions cannot contain additional attribution requirements*"
- These two licenses are considered **license-incompatible**. If someone combine a software package which uses license A, with a software package which uses license B, it would be impossible to distribute the combination because the two requirements cannot be simultaneously fulfilled.

License Compatibility (wikipedia): http://en.wikipedia.org/wiki/License_compatibility

License Compatibility



The Free-Libre / Open Source Software (FLOSS) License Slide (by David A. Wheeler) : http://www.dwheeler.com/essays/floss-license-slide.html

Apache License and third-party licenses

- **Authorized Licenses**
  - Apache License 2.0
  - Apache Software License 1.1
  - BSD (without advertising clause)
  - MIT/X11
  - University of Illinois/NCSA
  - W3C Software License
  - X.Net
  - zlib/libpng
  - FSF autoconf license
  - DejaVu Fonts
  - Academic Free License 3.0
  - Service+Component+Architecture+Specifications
  - OOXML XSD ECMA License
  - Microsoft Public License (MsPL)
  - Creative Commons Attribution (CC-A)
  - Python Software Foundation License

- **Excluded Licenses**
  - BCL*
  - Special exceptions to the GNU GPL (e.g. GNU Classpath)*
  - GNU GPL 1, 2, 3
  - GNU LGPL 2, 2.1, 3
  - Affero GPL 3
  - NPL 1.0/NPL 1.1
  - QPL
  - Sleepycat License

Apache third-party licensing policy : http://www.apache.org/legal/resolved.html

Open Source Project Hosts

- Project umbrellas host the open source projects and **can influence the overall culture of a community**

- Hosts provide infrastructure for open source project referred to as PRIM:
  - P (portal), R (repository), I (Issue tracking), M (mailing list)
  - Can provide legal governance

- There are three main types of public open source hosts:
  - Pure infrastructure
  - Vendor collaboration
  - Community focused

Project Host Type 1: Pure Infrastructure

- Examples: SourceForge, CollabNet, Codehaus, googlecode

- Provide the infrastructure
  - Sets overall rules (e.g. type of license permitted)
  - Each project governs itself

- Often many small projects
  - One or two developers, although may have lots of users
  - Fairly small codebase

- Provide an incubator role
  - Projects start under incubation and move to a more formal community when they grow

Project Host Type 2: Vendor Collaboration

- Examples: Eclipse, ObjectWeb, Mozilla
- Allows companies to collaborate
  - Specifically acknowledge the role companies have
  - Consider corporate needs
- Closer to commercial software development roadmap
  - More planning and oversight
    - (e.g.) have a architectural steering committee
    - (e.g.) have an official project management committee
  - Can be more conservative
- Eclipse recently created “Eclipse Project Incubator” as a place for innovation and investigation of new and alternative ideas.

Project Host Type 3: Community Focused

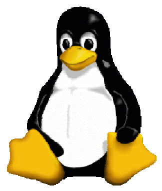
- Example: Apache Software Foundation
- Non-profit corporation
  - No staff, all volunteer
  - Elected membership
- Primary goal is to foster open source communities
  - Provide technical infrastructure
  - Provide legal oversight
  - Projects start under incubation and once they demonstrate they can run as a healthy Apache type project they graduate into an Apache top level project.
- Technocratic Meritocracy
  - People earn status by what they do
- Project communities are very independent
  - Project Management Committee (PMC) is a legal construct
    - Binding decisions e.g. to release software
  - Community decides direction and priorities

Examples of Open Source projects

- Apache
  - Web Server
  - Community initiated (apache.org)

The Apache Software Foundation
http://www.apache.org/

- Linux
  - Operating System kernel
  - Individual initiated (Linus Torvalds)



- Eclipse
  - Universal Integration platform
  - Extensible application framework supporting solution based plug-ins
  - IT Vendor initiated (IBM and others)

eclipse

- Mozilla
  - Browser and client technology
  - Hybrid (Netscape + community)

mozilla FOUNDATION

Mozilla Firefox

What characterizes an Open Source?

This Characterizes each open source project

Community Philosophy

Is influenced by License type License

3 families of licenses:

- Give me credit
    - Give me fixes
    - Give me everything

✓Watch out for license incompatibility

Community Culture

Is influenced by where the Project is hosted

There are three main types of public open source hosts:

- Pure infrastructure
- Vendor collaboration
- Community focused

Community Character

Is influenced by the people who support the project

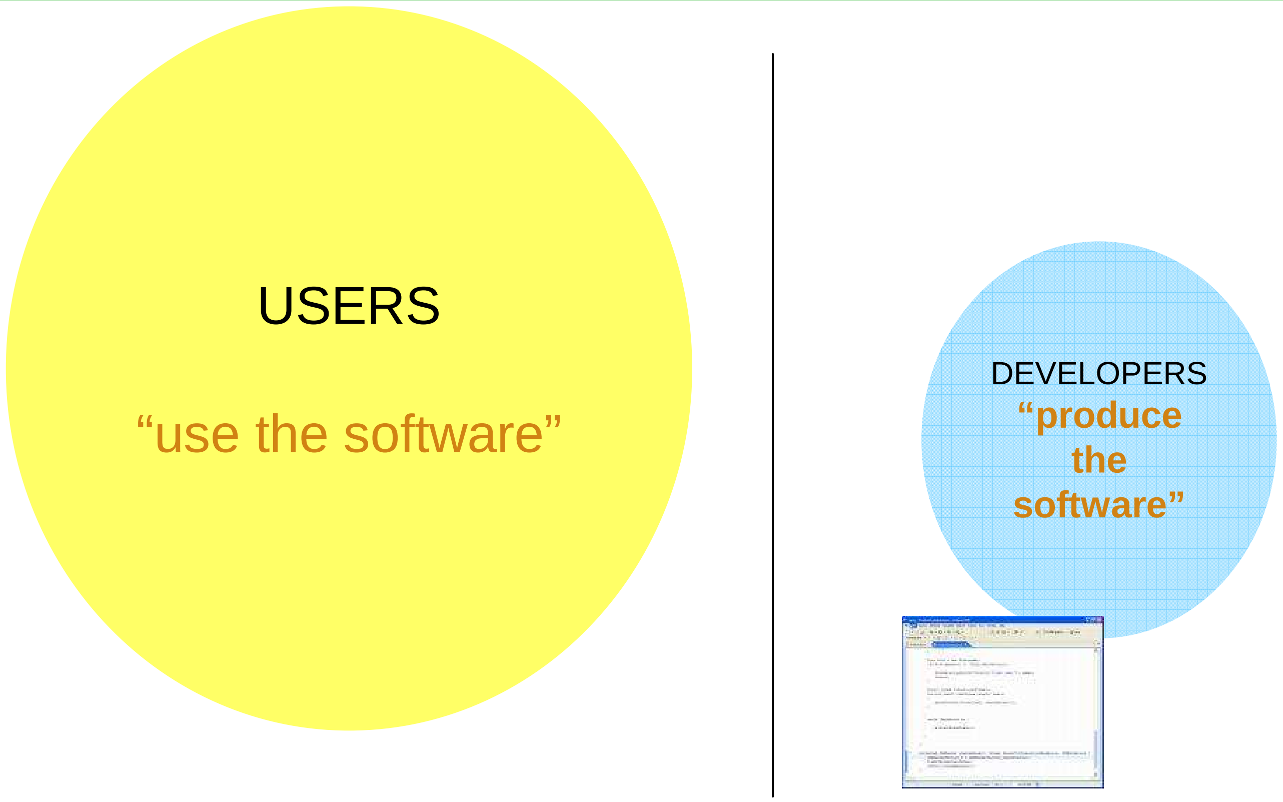
Each project has its own character:

- Community decides on how to create, grow and maintain the project.
- Technology supported by the project can influence
    - Who gets drawn to the project
    - How the project should be run to be successful

Open Source Community

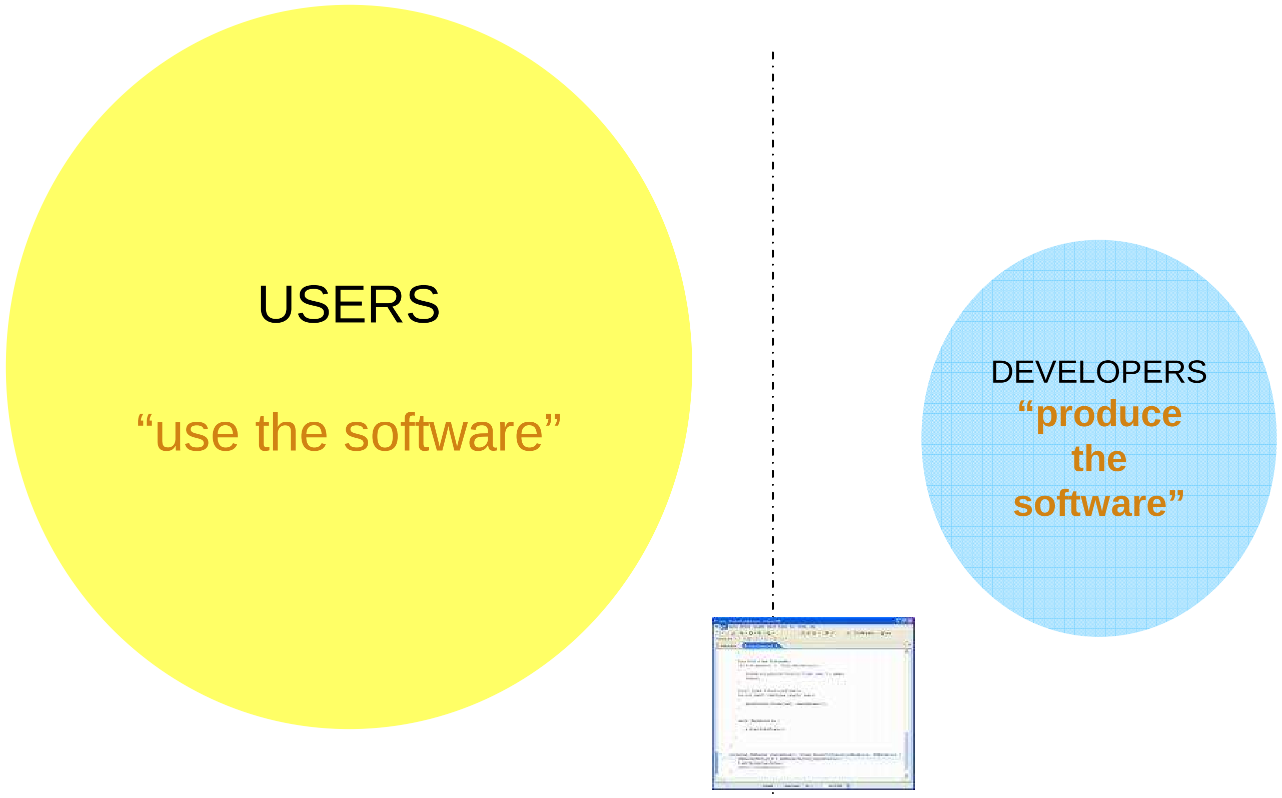
Heart of an open source project

Traditional Commercial Software



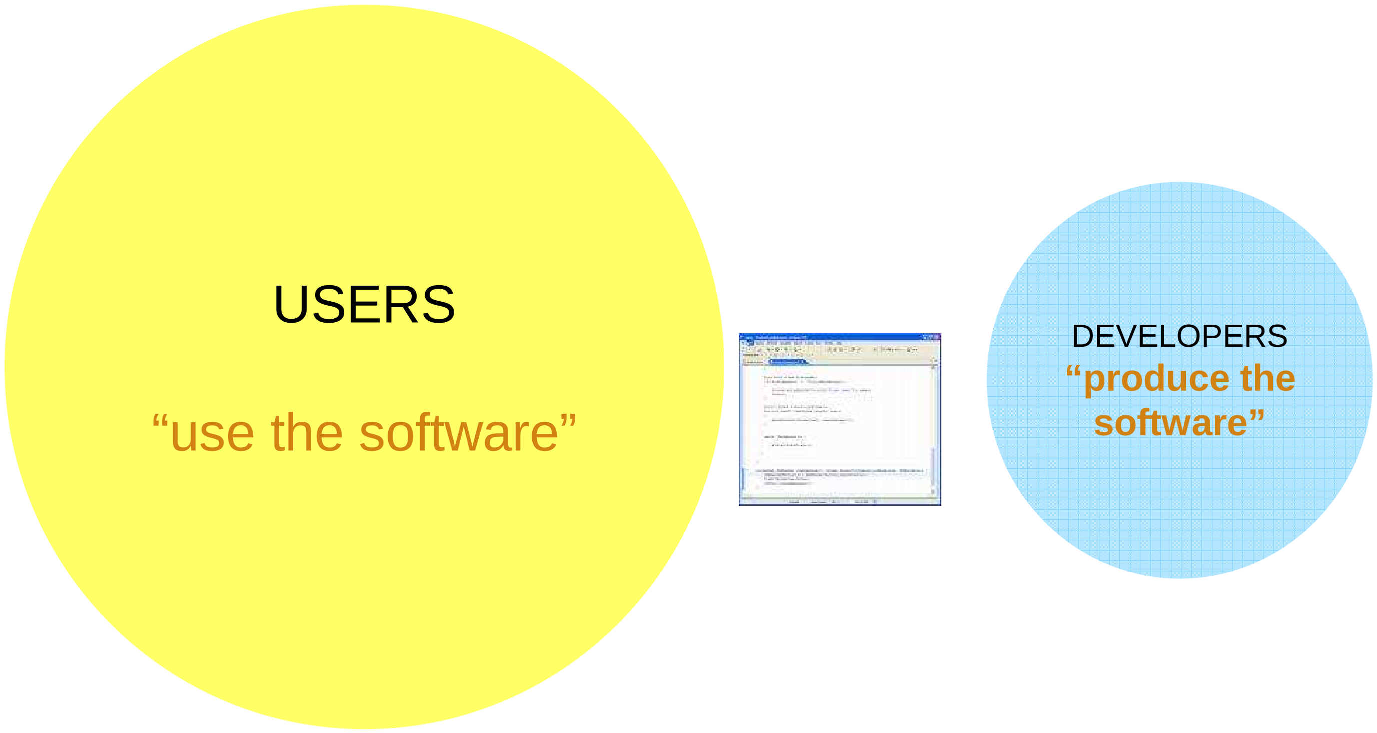
- Separate User and Developer Community
- Feedback provided through Beta and early adaptor programs
- Agile development model can help to provide more frequent feedback

Mixed Commercial – Share Code



- Code Is developed in the open and can be used readily
- Users provide feedback, but can’t modify the main repository
- Software is typically not free or a more advanced form of it is licensed

Open Source – Collaborative Environment



- Code Is developed in the open and can be used readily
- Users can become involved in the development of the software

Start of an Open Source Project

USERS
DEVELOPERS
"produce the software"

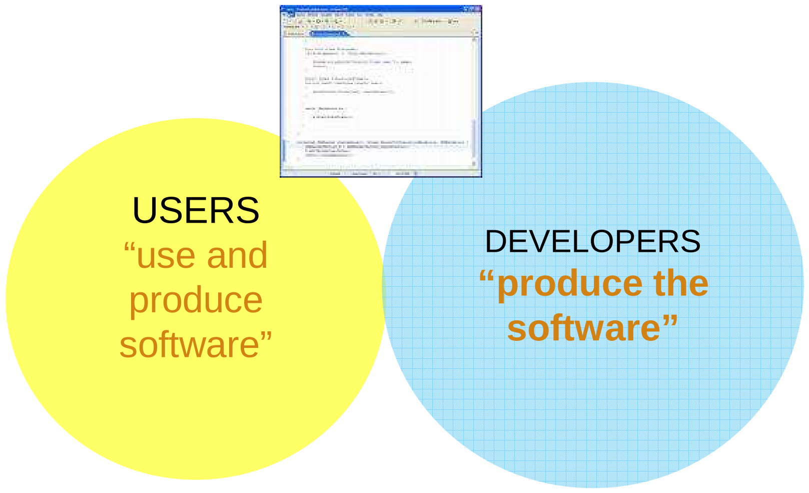
- Open Source projects typically start with
  - A smaller developer community who have an idea to develop
  - One or more users who can use the software

Who are the developers?

- A group of people interested to develop some idea in the open collaboratively.
  - Because ..
    - They want to create software that solves their business problems
    - Want to test new ideas in the market
    - Want To participate in new technology development
    - Want To test validity of standards through collaboration with users
    - Maybe they get paid by an employer, it's their job.
    - Maybe it is just interesting to get involved in new technology!
      - Participation in a successful project can open the door to fame and recognition
    - …….
    - There is no magic answer. This makes it good challenge to analyze how to attract developers for each project

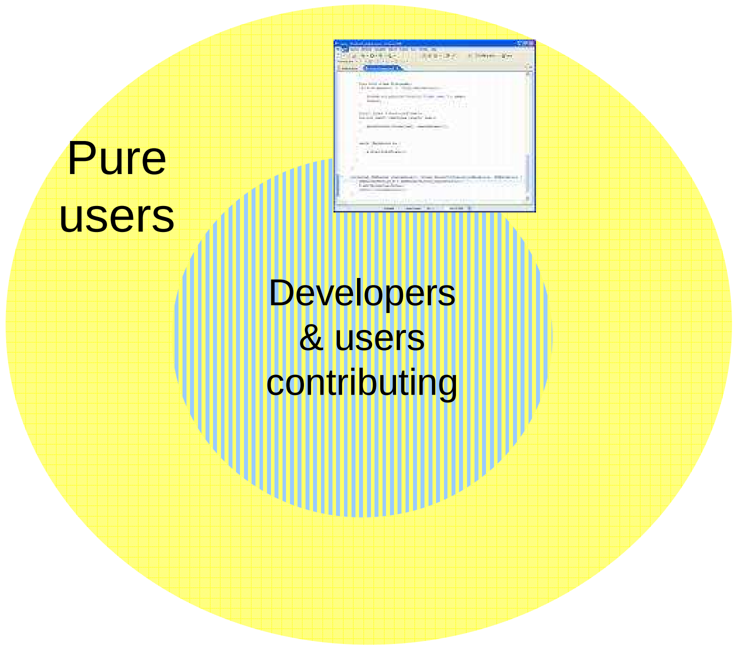
  - **IMPORTANT: Anyone interested can join in to help with the development**
    - **No invitation is required!** Just get involved in what is interesting to you and share your thoughts with the community.
    - People from different backgrounds participate in open source. The key is to not be shy about not having a high command of a given language. Source Code and technology ideas become the common language.

Open Source Project



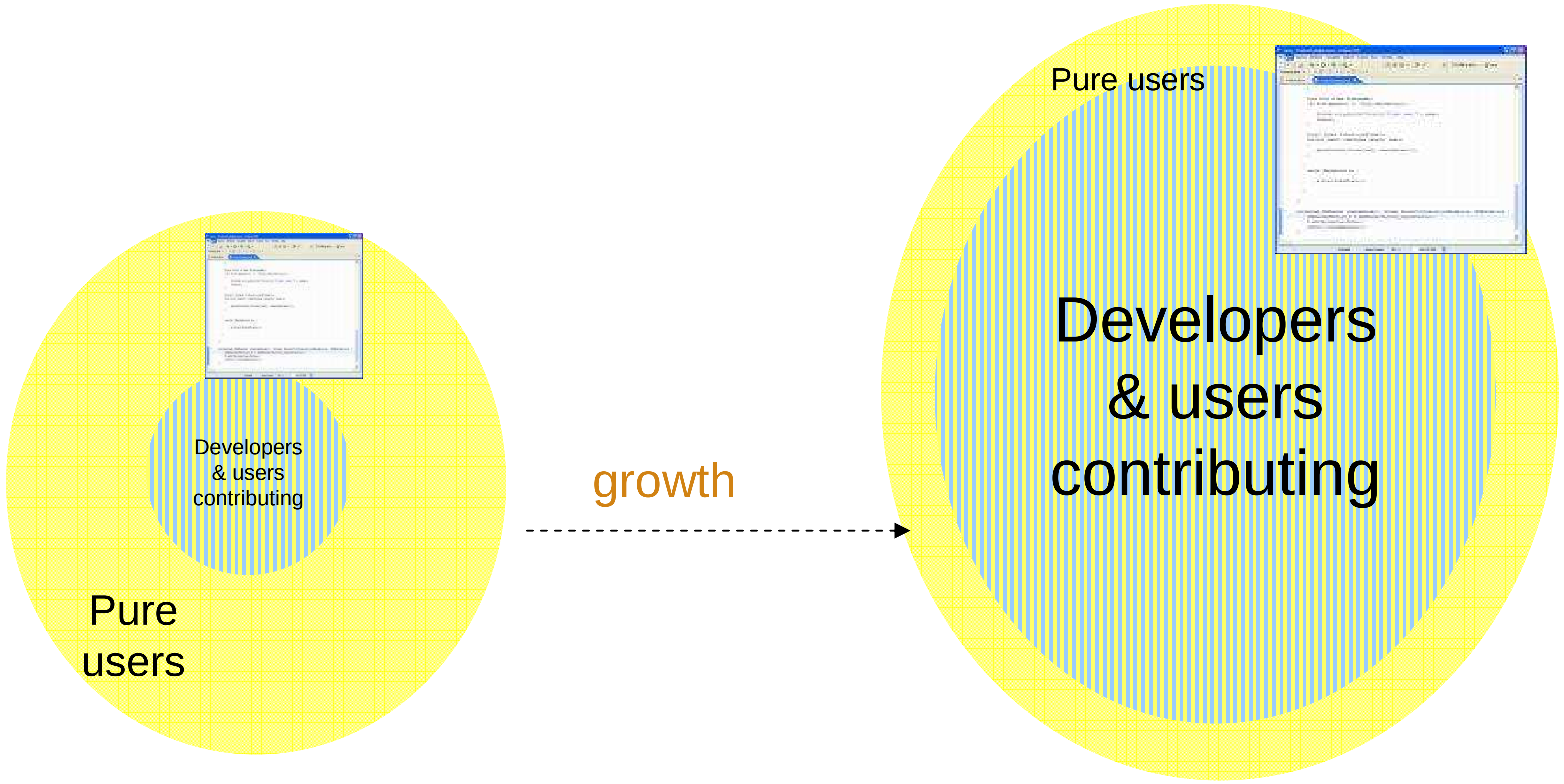
- Open Source Brings Users and Developers together to
  - Invent, develop, share experience, improve
  - Overtime, users join the developers community to influence what they use (care about).

Open Source Project



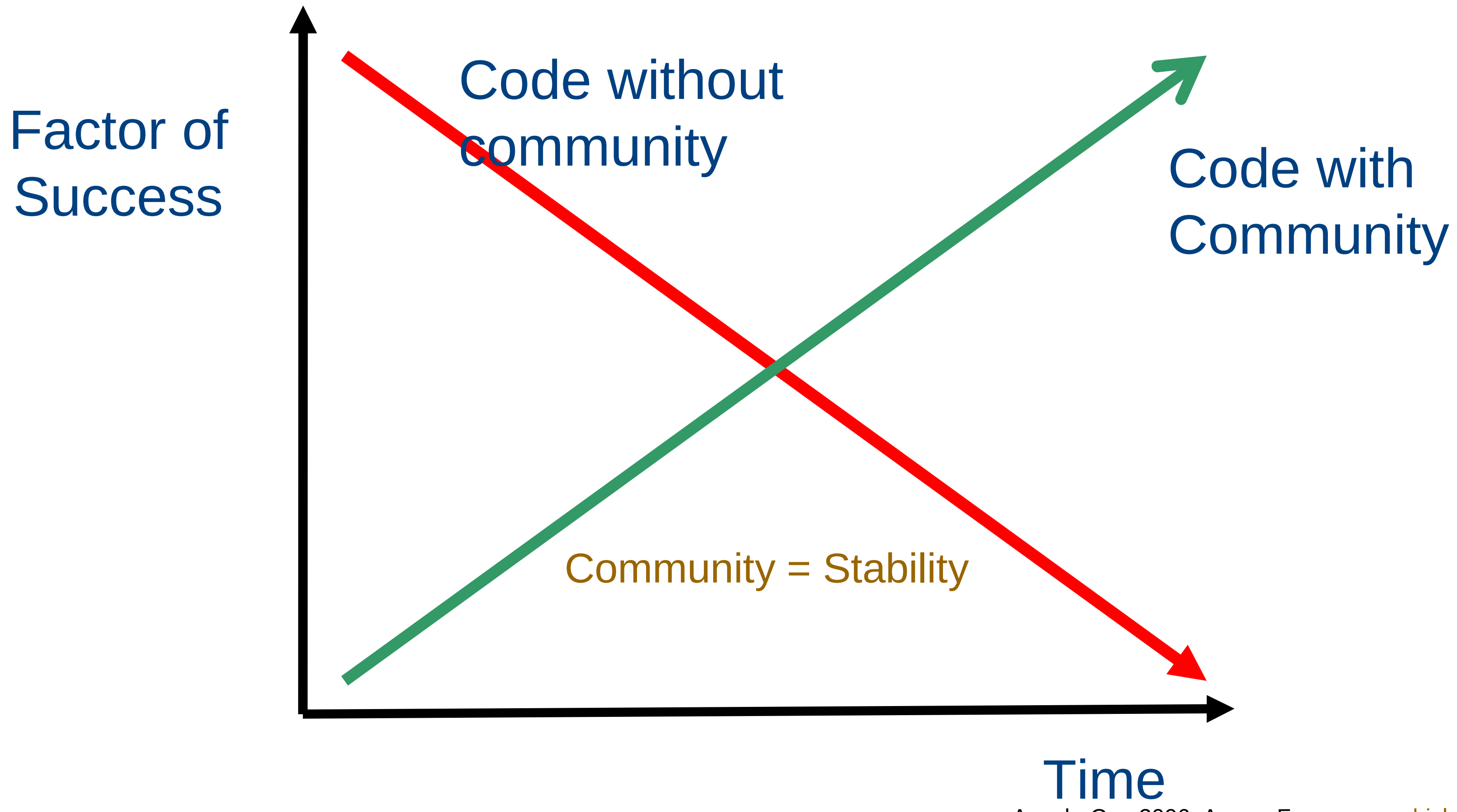
- A successful Open Source Community is an integrated user and developer community who together they
  - Invent, develop, share experience, improve

Open Source Project



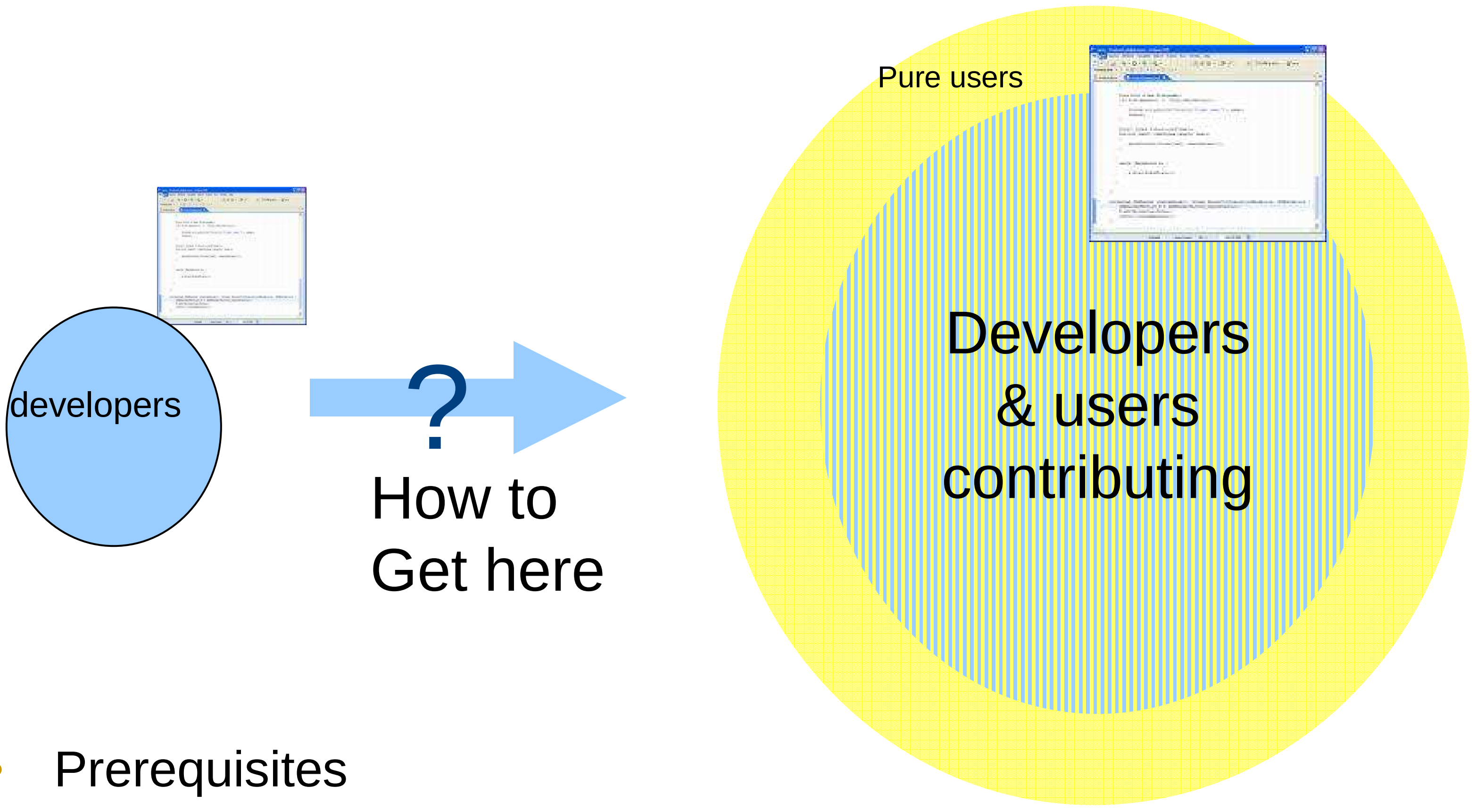
- Open source project grows as users get more involved and new developers join the project.
- Growth and adoption makes the project more stable and brings in innovations.

Why is Community Important?



ApacheCon 2006, Aaron Farr, www.cubiclemuses.com

How to Grow a Community?



- Prerequisites
  - Attractive Code Qualities
  - Communication Infrastructure
  - **And, people who believe in the project and help it grow**

Attractive Code Qualities for Starting OS projects

- Directly used by and useful to developers
- Builds
- Follows common standards where applicable
- Modular and flexible
- Consistent
- Enough documentation to help new developers to get started
- Enough test examples to help new developers to get started and be confident to test their changes
- Can be improved
  - Does not have to be functionality complete (Tough concept to grasp)
    - Incremental, smaller checkins accompanied with discussions with the community get the community more involved and enables others to participate in building the software.

Communication Infrastructure

- Source code repository
- Issue Tracking
  - Defines how problems can be reported.
  - Provide a way to organize handling of many different ideas, feedbacks, etc.
- Website
  - First impression of the project is from the website. It is important to have a good website which clearly states objectives and introduces the visitor the to project
    - Most developers don’t find this a fun thing to do! Find ways to make it happen.
- Documentation
  - User documentation : Focuses on how to use the project
  - Developer documentation: Focus on how to get involved
    - Caution: ‘code talks’ does not work 100%. It is worth to spend time to share information
- Mailing list, newsgroup, or forum
  - Communication should be open
    - Engages everyone and solicits new ideas and participation
  - Communication should be archived
    - Archived information can be used to search for problems that were discussed before
    - Provides a reference for decisions that were made

How to attract a community?

- Good product and frequent releases
- Good documentation
- Examples of how to use the software
- Modular and flexible architecture when it makes sense
  - Lowers the barrier to entry for developers. Let’s them focus on areas that they are interested in
  - Facilitates adoption by allowing users to pick and choose what they need. Lower footprint.
- Have an open, inviting environment
- Mentor new people to learn the project and feel comfortable to contribute
- But, that’s not enough!
  - People need to know about your project to download and use it or come to the site to participate.
    - Talk about the project and how it solves the given business problem through Conferences, Forums, Articles, …

Summary

- Open Source brings Users and Developers from all around the world together to invent, develop, share ideas, …
- Community is the heart of Open Source
    - Community = stability
- Starting and growing of an open source project requires
    - Code
    - Infrastructure
    - And, dedication and effort to build the community
- Participation in open source is open to all. No invitation is required!