Incubating Open Source Communities

J Aaron Farr
馮傑仁

JadeTower Corporation
Apache Software Foundation
farra@apache.org

Me

- Member of the ASF
- On the Incubator PMC
- Mentored OFBiz through Incubator
- Mentor of Apache Heraldry (OpenID)
- Software Startup
- Living in Hong Kong

Apache Software Foundation

- US Non-Profit
- Provides legal and infrastructure support
- Completely volunteer organization
- 1000+ committers, 200+ members
- 30+ open source projects

You

- Want to join an existing community
- Want to grow your own community
- Considering entering the ASF Incubator
- You're in the ASF Incubator

Should Feel Free To Ask Questions

Incubating Open Source Communities

J Aaron Farr

Communities

Communities

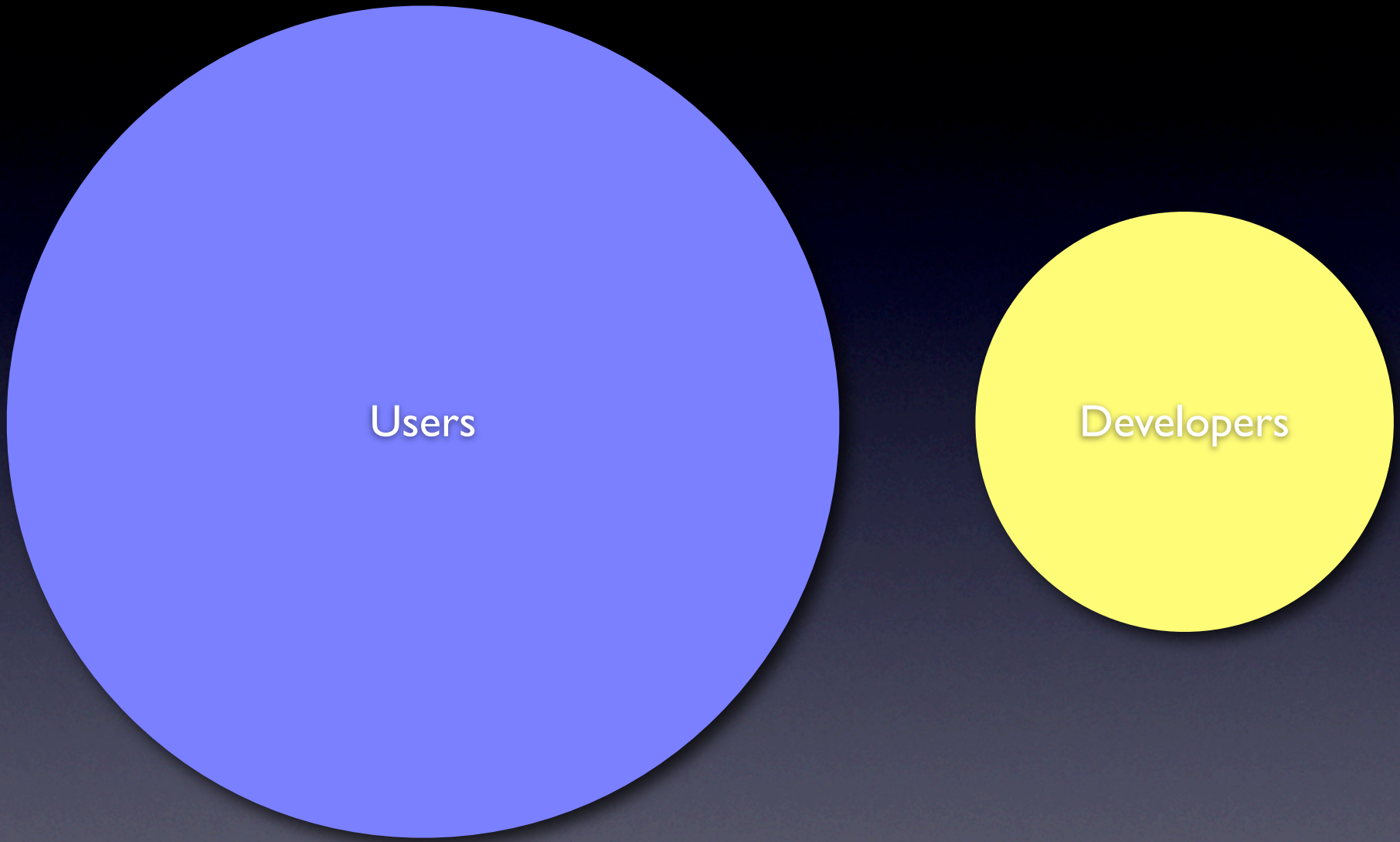
Which

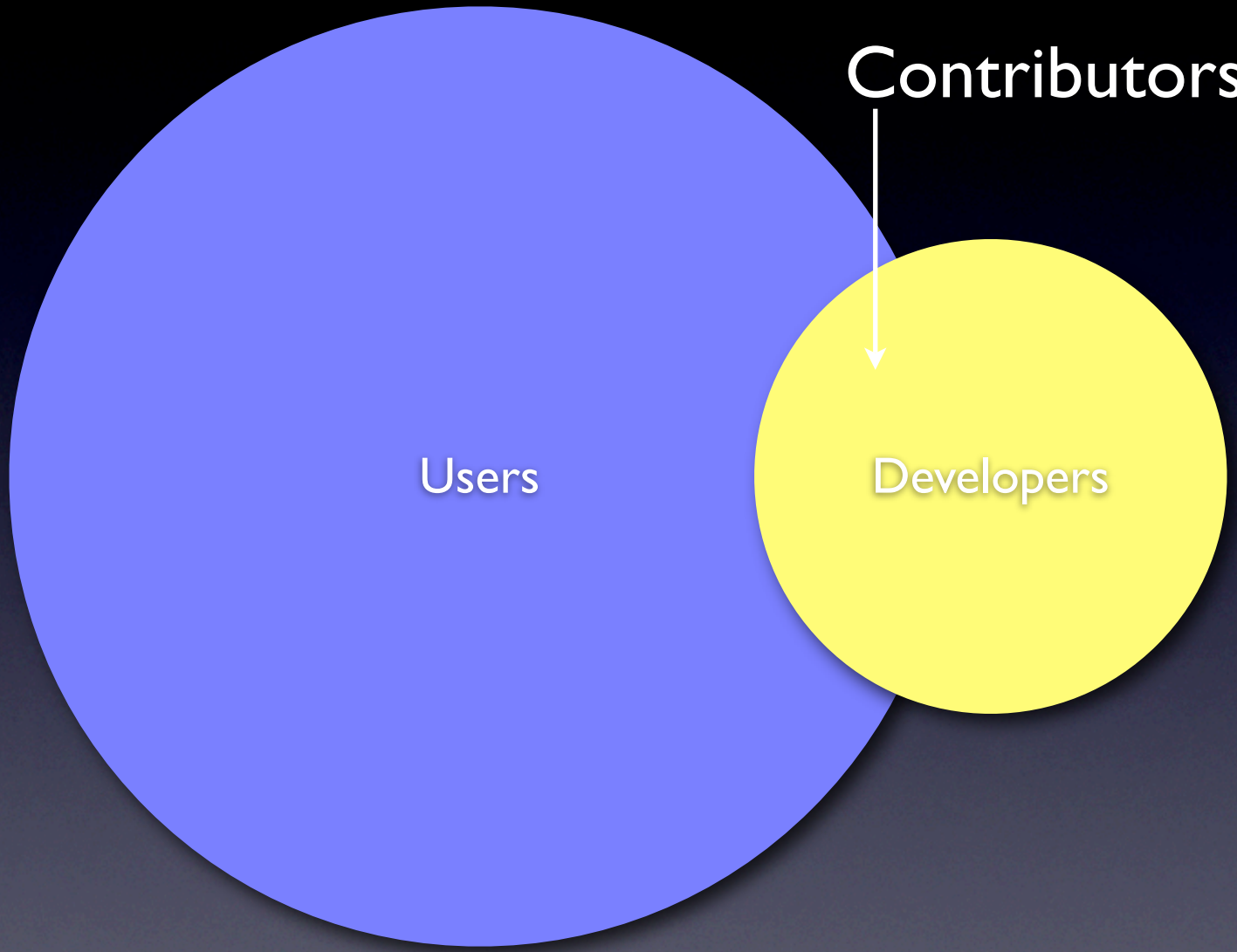
Why

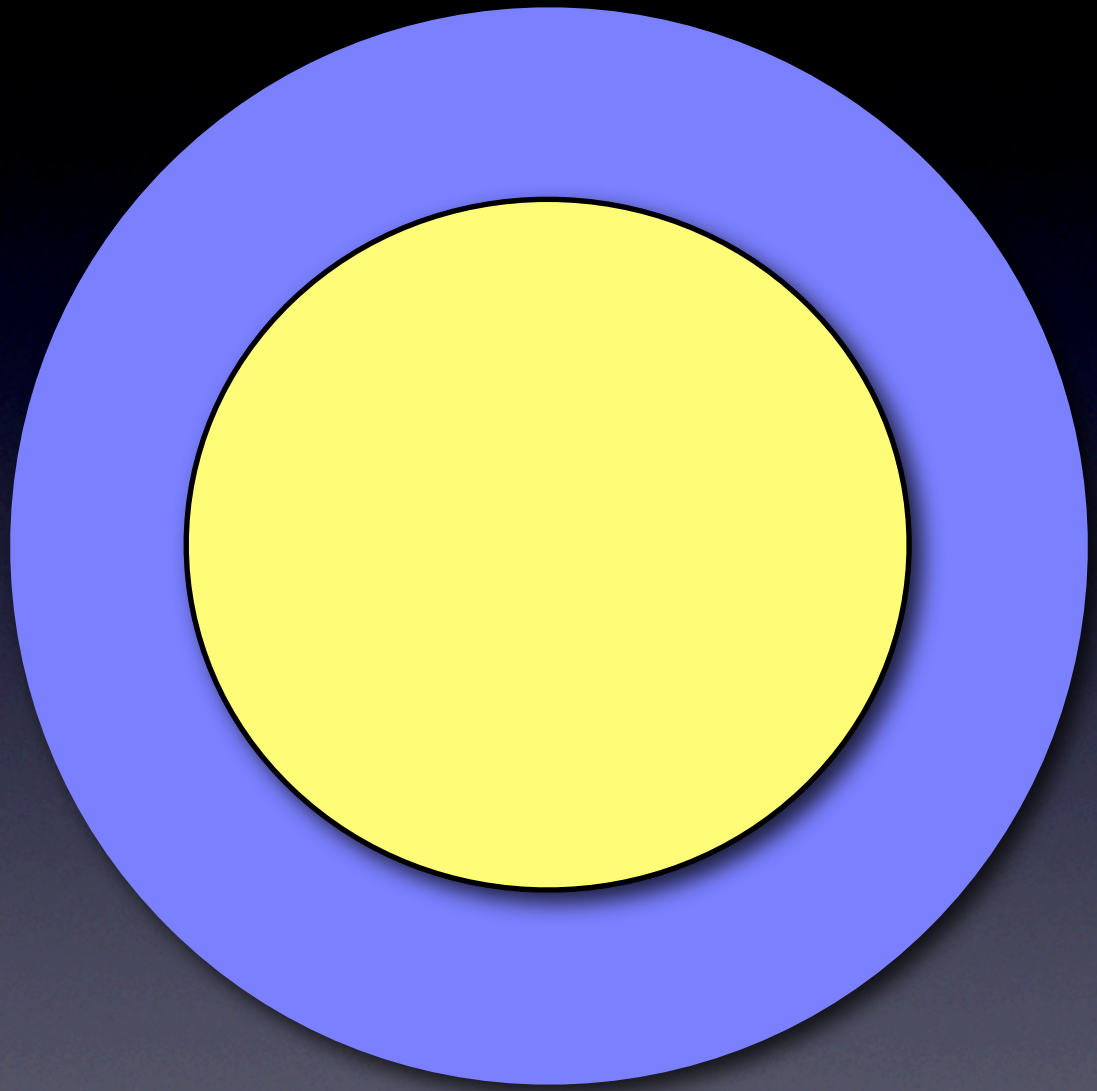
What

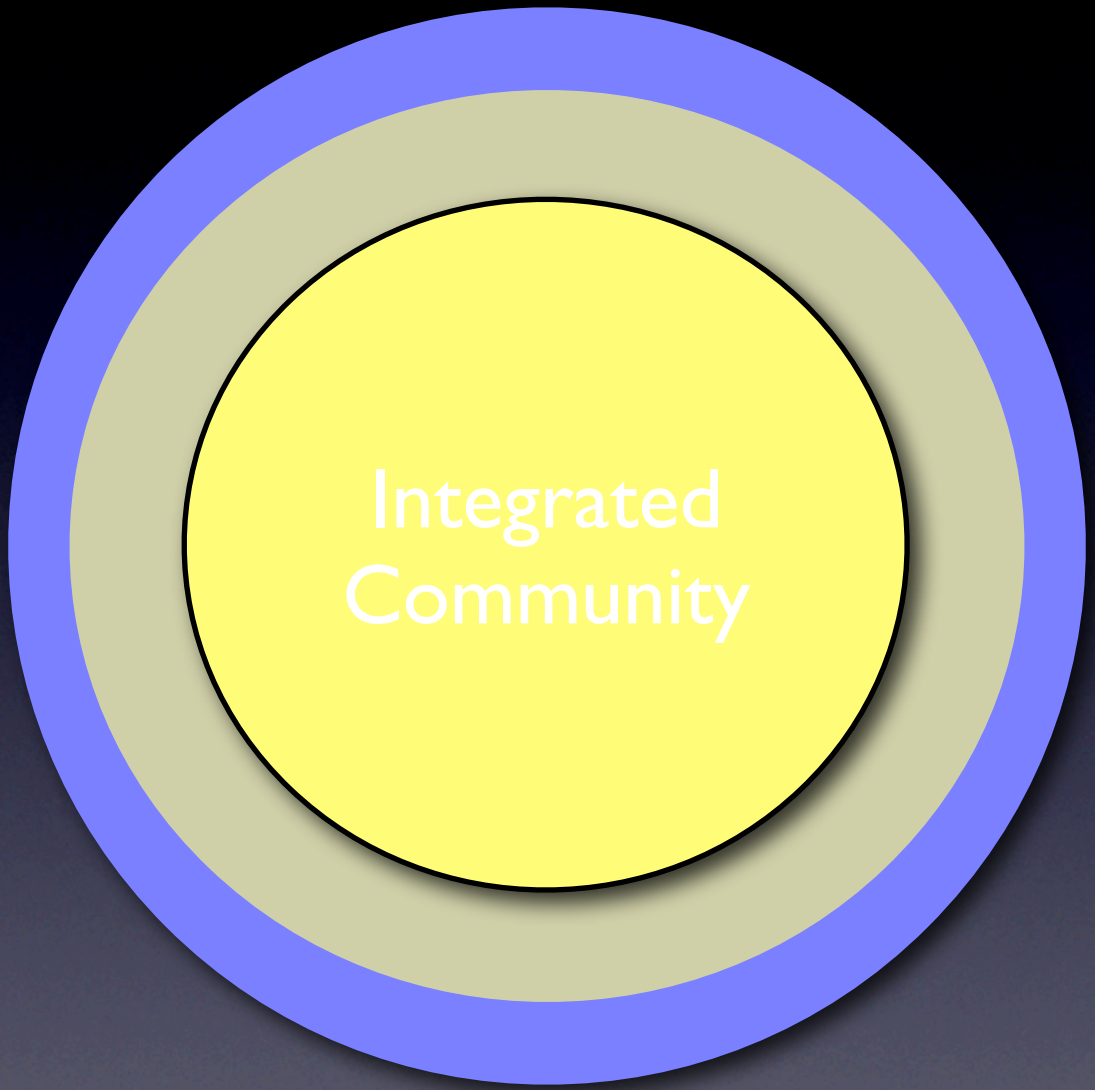
How

Which Community


Users
Developers


Contributors
Users
Developers




Integrated
Community


Integrated
Community

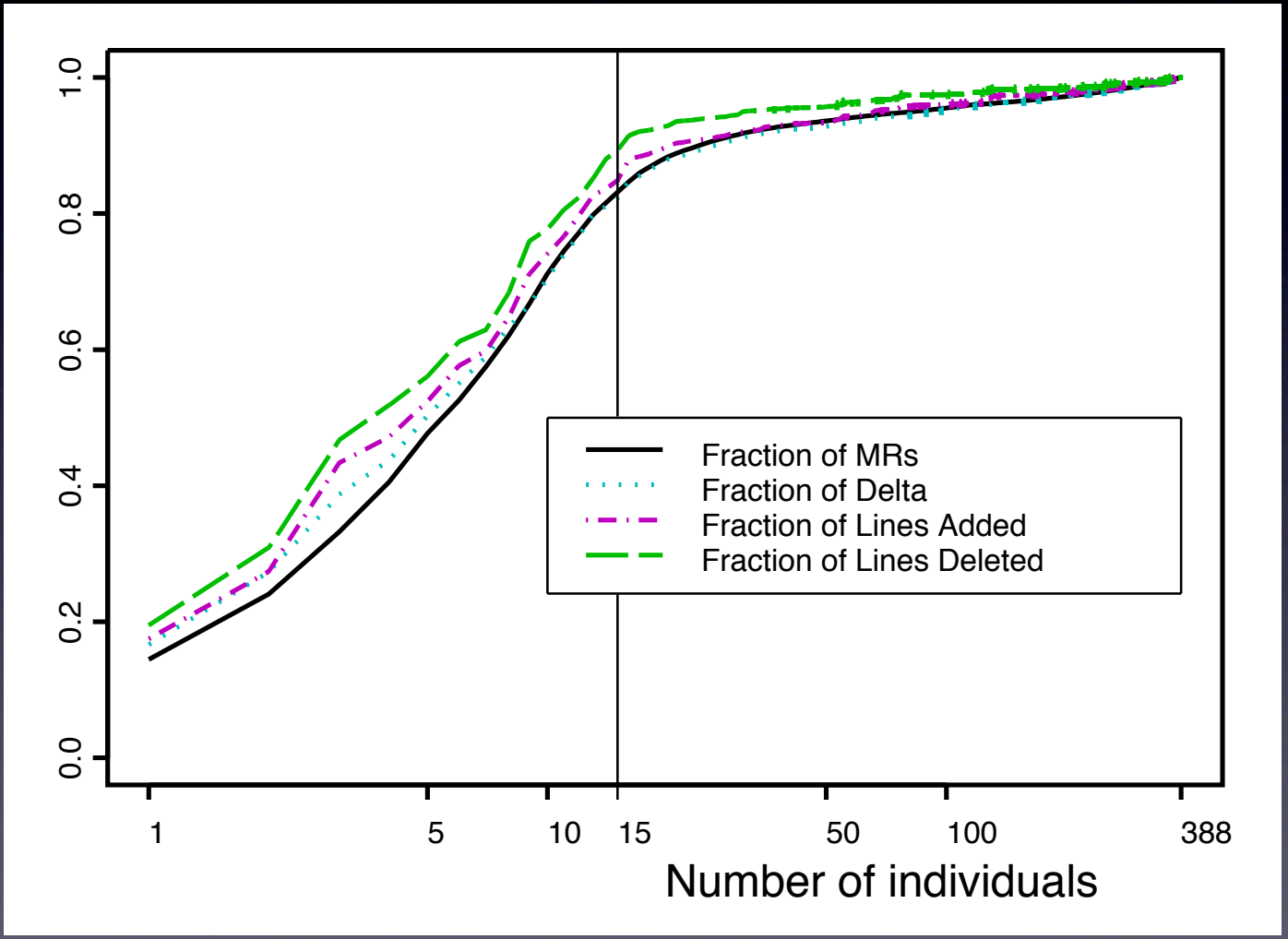
At The Core

The community begins with the **developers** and **contributors**. The **larger** and **more open** the developer community, the greater chance for **sustained activity** and **growth.**

A Note on Size

- Most developer communities are small
- Even with large developers, most of the development is usually by a small number of developers
- Apache requires minimum of 3 independent votes for a release

Apache's Long Tail



Feilding, Herbleb & Mockus, 2000

http://opensource.mit.edu/papers/mockusapache.pdf

Linux 2.6.20

50% of the changes where made by
2.5% of the developers

Who Wrote 2.6.20?
http://lwn.net/Articles/222773/
by corbet

What Does That Mean?

It means a **huge** number of contributors (741) who made **thousands** of small contributions

Communities

Which

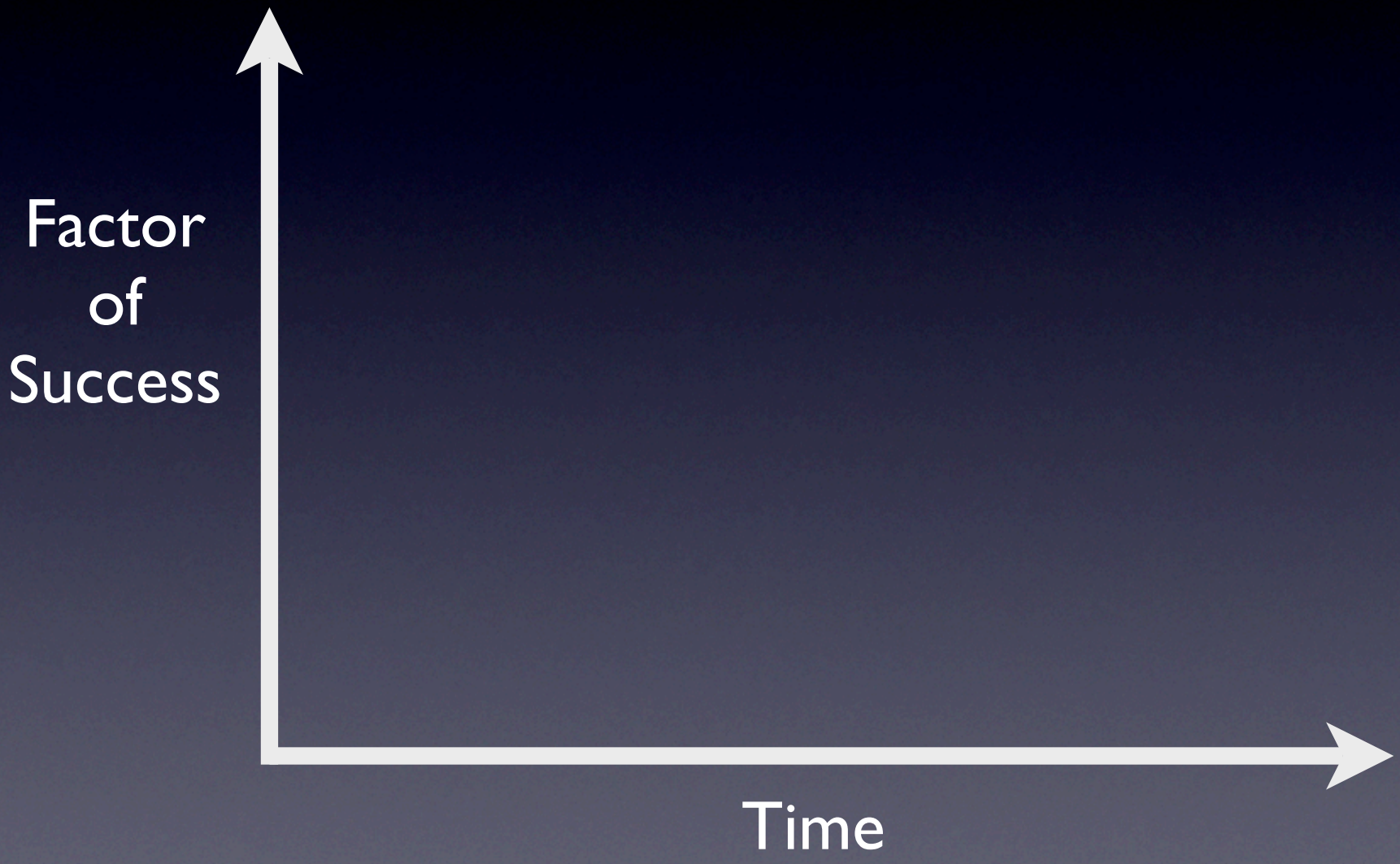
Why

What

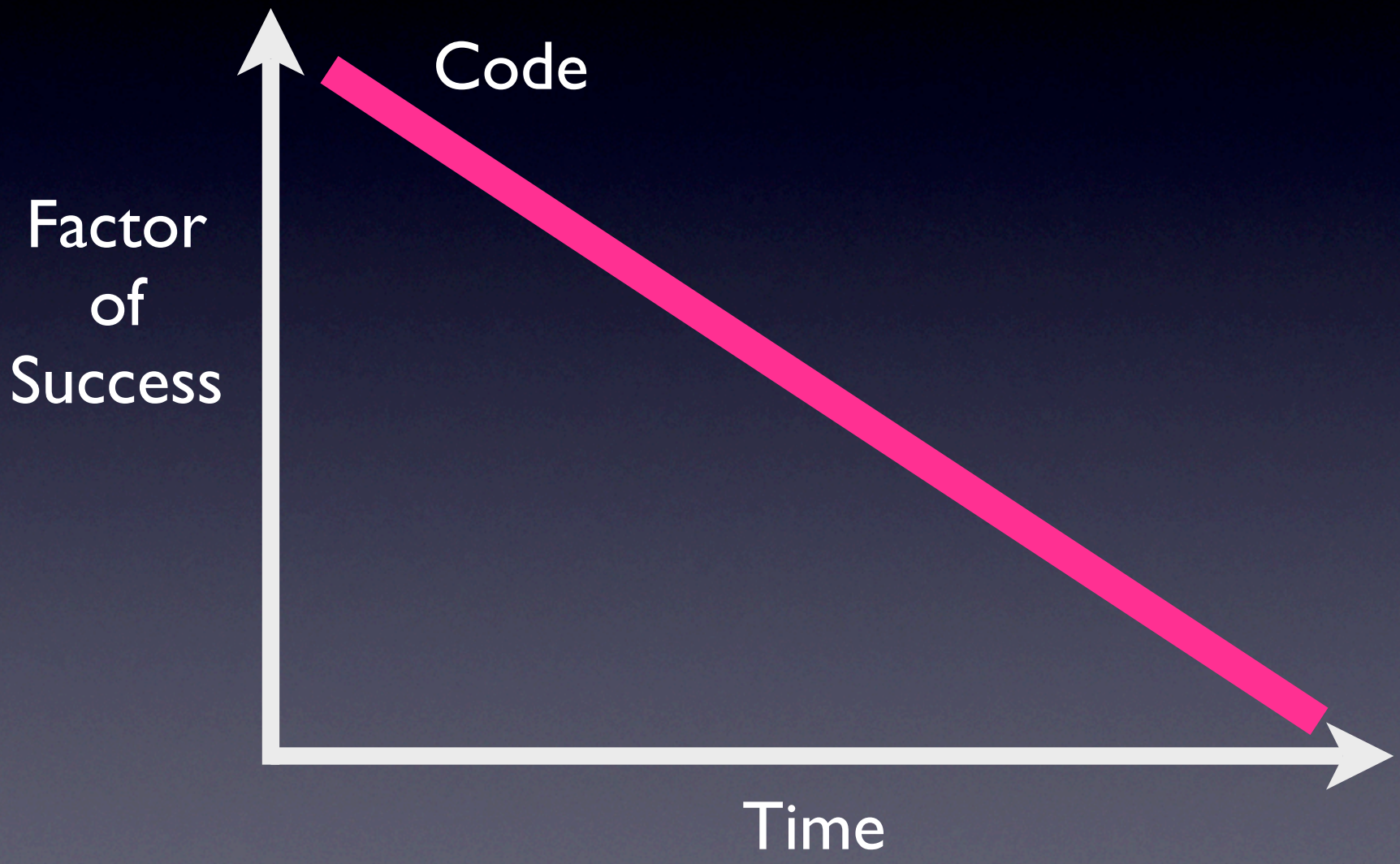
How

Community > Code

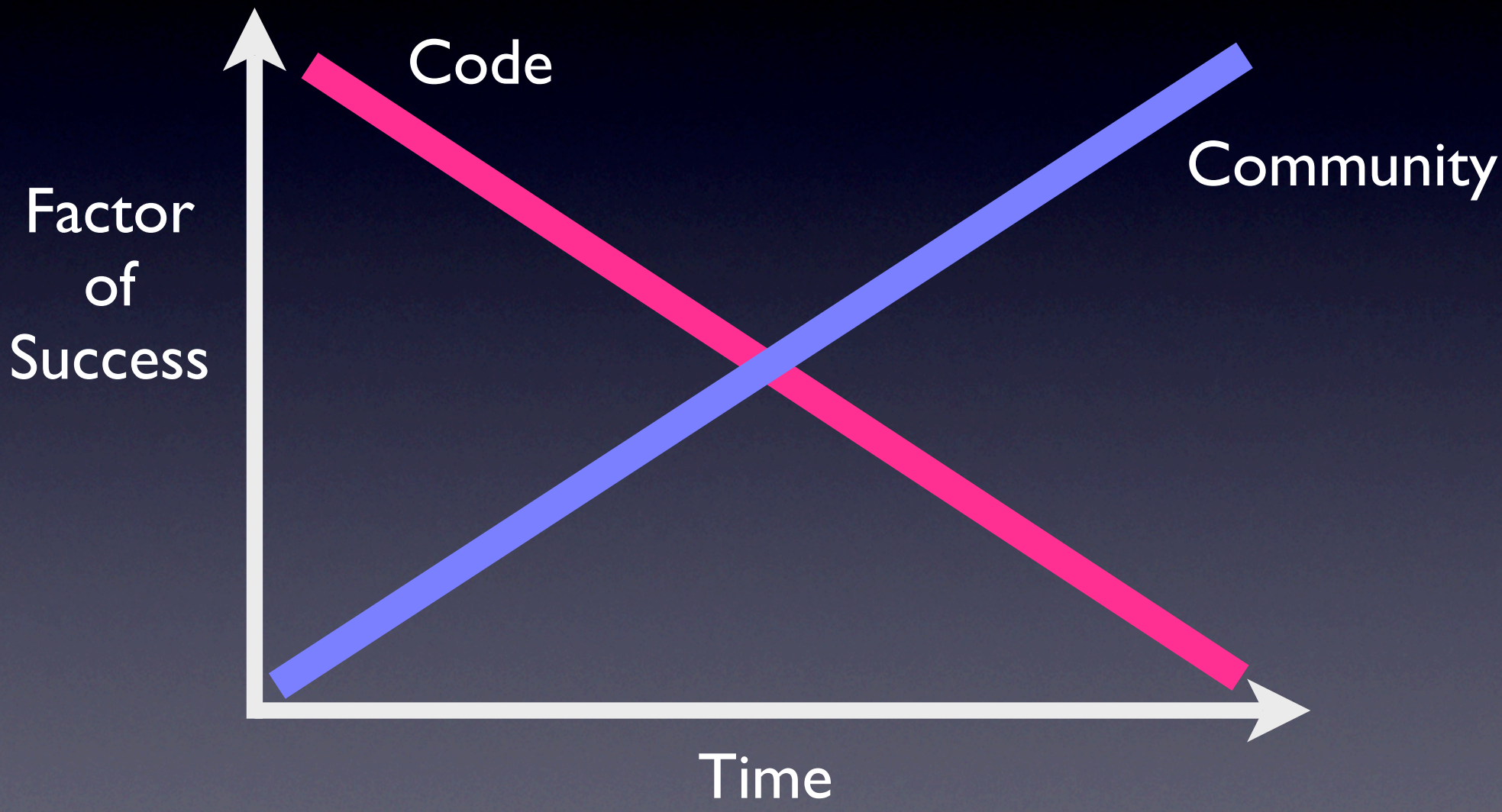
Why Community



Why Community



Why Community



Community First

Communities

Which

Why

What

How

Community

community |kəˈmyoōnitē| |kəˌmjunədi| |kəˌmjuːnɪti|

noun (pl. **-ties**)

1 a group of people living together in one place, esp. one practicing common ownership : *a community of nuns.*

- all the people living in a particular area or place : *local communities.*
- a particular area or place considered together with its inhabitants : *a rural community.*
- (**the community**) the people of a district or country considered collectively, esp. in the context of social values and responsibilities; society : *preparing prisoners for life back in the community.*
- [as adj.] denoting a worker or resource designed to serve the people of a particular area : *community health services.*

2 [usu. with adj.] a group of people having a religion, race, profession, or other particular characteristic in common : *Rhode Island's Japanese community | the scientific community.*

- a body of nations or states unified by common interests : [in names] *the European Community | the African Economic Community.*

3 a feeling of fellowship with others, as a result of sharing common attitudes, interests, and goals : *the sense of community that organized religion can provide.*

- [in sing.] a similarity or identity : *writers who shared a community of interests.*
- joint ownership or liability : *a commitment to the community of goods.*

4 Ecology a group of interdependent organisms of different species growing or living together in a specified habitat : *communities of insectivorous birds.*

- a set of species found in the same habitat or ecosystem at the same time.

Oxford American Dictionaries

Community

community |kəˈmyoōnitē| |kəˌmjunədi| |kəˌmjuːnɪti|

noun (pl. **-ties**)

1 a group of people living together in one place, esp. one practicing common ownership : *a community of nuns.*

- all the people living in a particular area or place : *local communities.*
- a particular area or place considered together with its inhabitants : *a rural community.*
- (**the community**) the people of a district or country considered collectively, esp. in the context of social values and responsibilities; society : *preparing prisoners for life back in the community.*
- [as adj.] denoting a worker or resource designed to serve the people of a particular area : *community health services.*

2 [usu. with adj.] a group of people having a religion, race, profession, or other particular characteristic in common : *Rhode Island's Japanese community | the scientific community.*

- a body of nations or states unified by common interests : [in names] *the European Community | the African Economic Community.*

3 a feeling of fellowship with others, as a result of sharing common attitudes, interests, and goals : *the sense of community that organized religion can provide.*

- [in sing.] a similarity or identity : *writers who shared a community of interests.*
- joint ownership or liability : *a commitment to the community of goods.*

4 Ecology a group of interdependent organisms of different species growing or living together in a specified habitat : *communities of insectivorous birds.*

- a set of species found in the same habitat or ecosystem at the same time.

Oxford American Dictionaries

Community

People
Customs
Ownership
Shared Goals

Community

Communicates

Community

People
Customs
Ownership
Shared Goals
Communication

Culture

Shared meanings used by a group to approach the problems of life

Communities

Which

Why

What

How

The Path To Contribution

Path to Contribution

Types of Contribution

- Documentation
- Bug Fixes
- Answers on mailing lists
- Promotion
- Money

Providing a Path

- *How to Get Involved* Documentation
- Low Barrier To Contribution
  - Wiki Documentation Site
  - Bug Tracker
- Recognize Contributors (Release Notes)

Community

People
Customs
Ownership
Shared Goals
Communication

People

Individuals

Not Business

Not Corporations

Not Organizations

People

- Meritocracy
- Clearly define:
  - Roles
  - Expectations
  - Qualifications

People

Roles

The meritocracy enables various roles:

`user` | `developer` | `committer` | `PMC member` | `ASF member`

User

`user` is someone that uses our software. They contribute to the Apache projects by providing feedback to developers in the form of bug reports and feature suggestions. Users participate in the Apache community by helping other users on mailing lists and user support forums. The passive users are also known as `lurkers`.

Developer

`developer` is a user who contributes to a project in the form of code or documentation. They take extra steps to participate in a project, are active on the developer mailing list, participate in discussions, provide patches, documentation, suggestions, and criticism. Developers are also known as `contributors`.

Committer

`committer` is a developer that was given write access to the code repository and has a signed Contributor License Agreement (CLA) on file. They have an apache.org mail address....

People

Apache CLA

The Apache Software Foundation
Individual Contributor License Agreement ("Agreement") V2.0
http://www.apache.org/licenses/

Thank you for your interest in The Apache Software Foundation (the "Foundation"). In order to clarify the intellectual property license granted with Contributions from any person or entity, the Foundation must have a Contributor License Agreement ("CLA") on file that has been signed by each Contributor, indicating agreement to the license terms below. This license is for your protection as a Contributor as well as the protection of the Foundation and its users; it does not change your rights to use your own Contributions for any other purpose. If you have not already done so, please complete and send an original signed Agreement to The Apache Software Foundation, 1901 Munsey Drive, Forest Hill, MD 21050-2747, U.S.A. If necessary, you may send it by facsimile to the Foundation at +1-410-803-2258. Please read this document carefully before signing and keep a copy for your records.

Customs

Oral Traditions

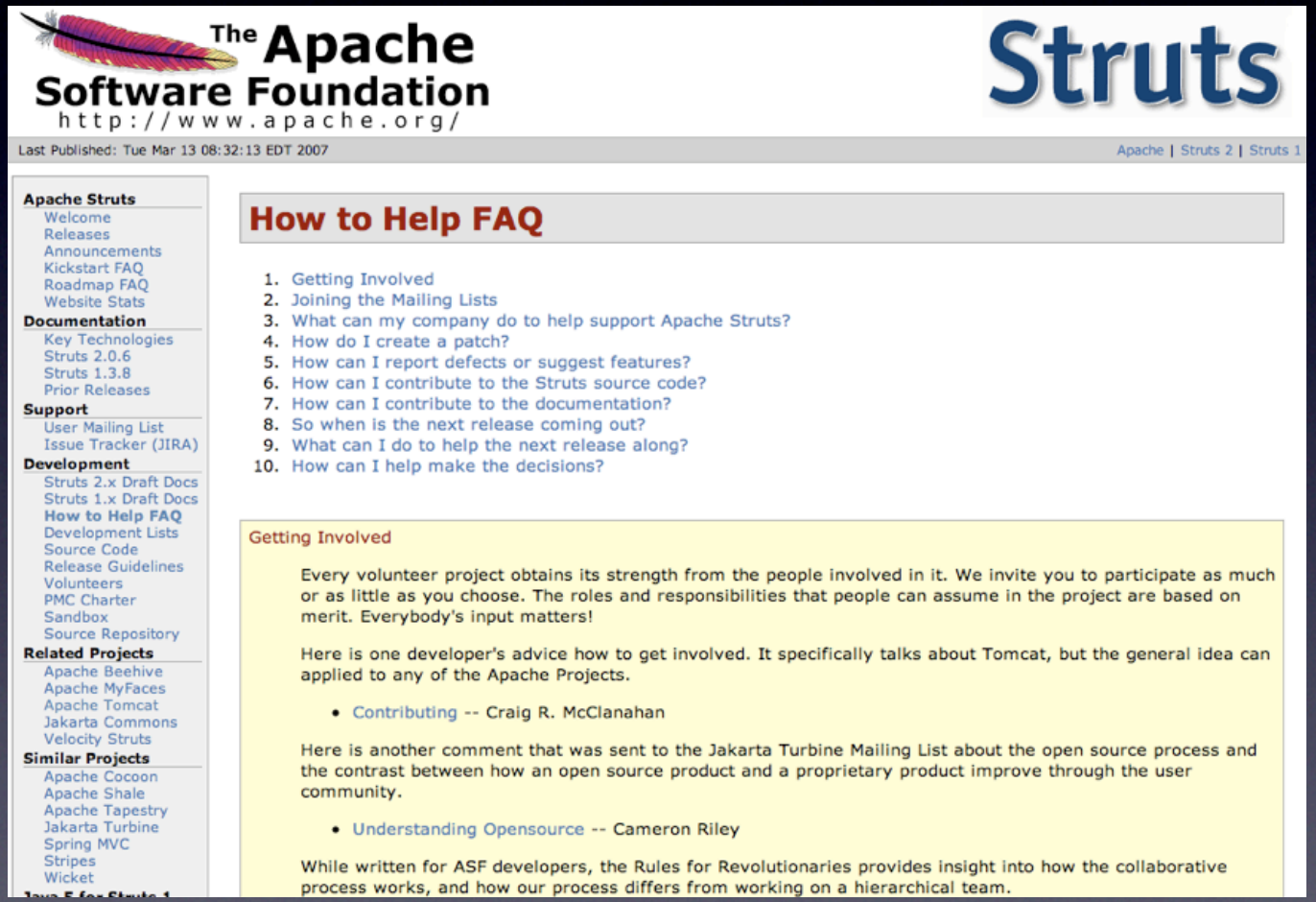
Rules of Participation

Decision Making Processes

Customs

- Clearly define project practices
  - Bug submission process
  - Release process
  - Voting process
- Coding Standards
- Mailing List Etiquette

FAQ


The Apache Software Foundation
http://www.apache.org/
Struts
Last Published: Tue Mar 13 08:32:13 EDT 2007
Apache | Struts 2 | Struts 1
Apache Struts
Welcome
Releases
Announcements
Kickstart FAQ
Roadmap FAQ
Website Stats
Documentation
Key Technologies
Struts 2.0.6
Struts 1.3.8
Prior Releases
Support
User Mailing List
Issue Tracker (JIRA)
Development
Struts 2.x Draft Docs
Struts 1.x Draft Docs
How to Help FAQ
Development Lists
Source Code
Release Guidelines
Volunteers
PMC Charter
Sandbox
Source Repository
Related Projects
Apache Beehive
Apache MyFaces
Apache Tomcat
Jakarta Commons
Velocity Struts
Similar Projects
Apache Cocoon
Apache Shale
Apache Tapestry
Jakarta Turbine
Spring MVC
Stripes
Wicket
How to Help FAQ
1. Getting Involved
2. Joining the Mailing Lists
3. What can my company do to help support Apache Struts?
4. How do I create a patch?
5. How can I report defects or suggest features?
6. How can I contribute to the Struts source code?
7. How can I contribute to the documentation?
8. So when is the next release coming out?
9. What can I do to help the next release along?
10. How can I help make the decisions?
Getting Involved
Every volunteer project obtains its strength from the people involved in it. We invite you to participate as much or as little as you choose. The roles and responsibilities that people can assume in the project are based on merit. Everybody's input matters!
Here is one developer's advice how to get involved. It specifically talks about Tomcat, but the general idea can applied to any of the Apache Projects.
Contributing -- Craig R. McClanahan
Here is another comment that was sent to the Jakarta Turbine Mailing List about the open source process and the contrast between how an open source product and a proprietary product improve through the user community.
Understanding Opensource -- Cameron Riley
While written for ASF developers, the Rules for Revolutionaries provides insight into how the collaborative process works, and how our process differs from working on a hierarchical team.

Ownership

Open Source License

Access Permissions

Responsibilities

Ownership

- Prominently display the license
- Openly identify business interests
- Non-locking source control
- No Author Tags
- Avoid Dictatorships or “one man shows”

Commit Bits

- When to give commit access?
  - Depends on the project
  - When the contributor has shown consistent commitment with patches
- The ASF averages 15 new committers a month, but most of that is through new projects entering.
- Many projects only add 2 committers a year

Shared Goals

Group Identity

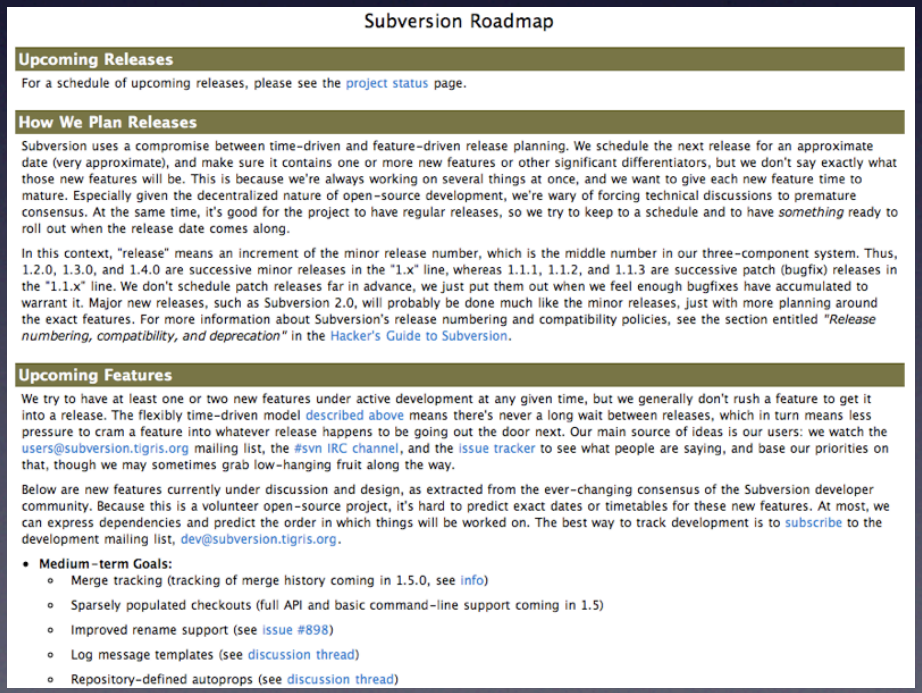
Clear Purpose

What We're Not

Subversion

Why does this project exist?

To take over the CVS user base. Specifically, we're writing a new version control system that is very similar to CVS, but fixes many things that are broken. See our front page.



Subversion Roadmap

Upcoming Releases

For a schedule of upcoming releases, please see the project status page.

How We Plan Releases

Subversion uses a compromise between time-driven and feature-driven release planning. We schedule the next release for an approximate date (very approximate), and make sure it contains one or more new features or other significant differentiators, but we don't say exactly what those new features will be. This is because we're always working on several things at once, and we want to give each new feature time to mature. Especially given the decentralized nature of open-source development, we're wary of forcing technical discussions to premature consensus. At the same time, it's good for the project to have regular releases, so we try to keep to a schedule and to have *something* ready to roll out when the release date comes along.

In this context, "release" means an increment of the minor release number, which is the middle number in our three-component system. Thus, 1.2.0, 1.3.0, and 1.4.0 are successive minor releases in the "1.x" line, whereas 1.1.1, 1.1.2, and 1.1.3 are successive patch (bugfix) releases in the "1.1.x" line. We don't schedule patch releases far in advance, we just put them out when we feel enough bugfixes have accumulated to warrant it. Major new releases, such as Subversion 2.0, will probably be done much like the minor releases, just with more planning around the exact features. For more information about Subversion's release numbering and compatibility policies, see the section entitled *"Release numbering, compatibility, and deprecation"* in the Hacker's Guide to Subversion.

Upcoming Features

We try to have at least one or two new features under active development at any given time, but we generally don't rush a feature to get it into a release. The flexibly time-driven model described above means there's never a long wait between releases, which in turn means less pressure to cram a feature into whatever release happens to be going out the door next. Our main source of ideas is our users: we watch the users@subversion.tigris.org mailing list, the #svn IRC channel, and the issue tracker to see what people are saying, and base our priorities on that, though we may sometimes grab low-hanging fruit along the way.

Below are new features currently under discussion and design, as extracted from the ever-changing consensus of the Subversion developer community. Because this is a volunteer open-source project, it's hard to predict exact dates or timetables for these new features. At most, we can express dependencies and predict the order in which things will be worked on. The best way to track development is to subscribe to the development mailing list, dev@subversion.tigris.org.

- **Medium-term Goals:**
  - Merge tracking (tracking of merge history coming in 1.5.0, see info)
  - Sparsely populated checkouts (full API and basic command-line support coming in 1.5)
  - Improved rename support (see issue #898)
  - Log message templates (see discussion thread)
  - Repository-defined autoprops (see discussion thread)

Apache Avalon

JAN 1999

The [Avalon] project is an effort to create, design, develop and maintain a common framework for server applications written using the Java language. This framework will not be a standalone product, but will allow existing and yet to be created server applications to fit into a common platform and to share code, design and human resources.

Apache Avalon

FEB 2003

The Avalon project is an effort to create, design, develop and maintain a common framework and set of components for applications written using the Java language.

Having said that, what Avalon 'is', is a framework that allows components of varying scale to be created, managed via a specific set of lifecycle methods, and used in an application. While Avalon is geared towards server-side applications, it is not limited to such, and is quite flexible.

Apache Avalon

Apache Avalon provides a complete platform for component programming including a core framework, utilities, tools, components and containers. By using key design patterns such as Inversion of Control (IoC) and Separation of Concerns (SoC), Avalon achieves a number of advantages over traditional object oriented programming frameworks:

DEC 2003

Apache Avalon

Apache Avalon provides a complete development platform for component and container programming utilizing key design patterns such as Inversion of Control and Separation of Concerns.

NOV 2004

Society → Community

The difference between a
society and a **community** is
a "**unity of wills**"

Community and Society: Gemeinschaft und Gesellschaft, Ferdinand Tönnies

People Customs Ownership Shared Goals Communication

Communication

Open

Consistent

Archived

Communication

- Website (with documentation)
- Mailing Lists
- Issue Tracking
- Source Repository
- Instant Message / Chat

Communities

Which

Why

What

How

Communities

Code

Code

- Directly useful to developers (itch)
- Builds
- Common standards
- Composable
- Consistent
- Improvable

Composability

... codebases that are more modular or have more option value increase developers' incentives to join and to remain involved in an open source development effort; and decrease the amount of free-riding in equilibrium.

The Architecture of Participation:
Does Code Architecture Mitigate Free Riding in the Open Source Development Model
Baldwin and Clark, 2005

http://www.people.hbs.edu/cbaldwin/DR2/BaldwinArchPartAll.pdf

Community

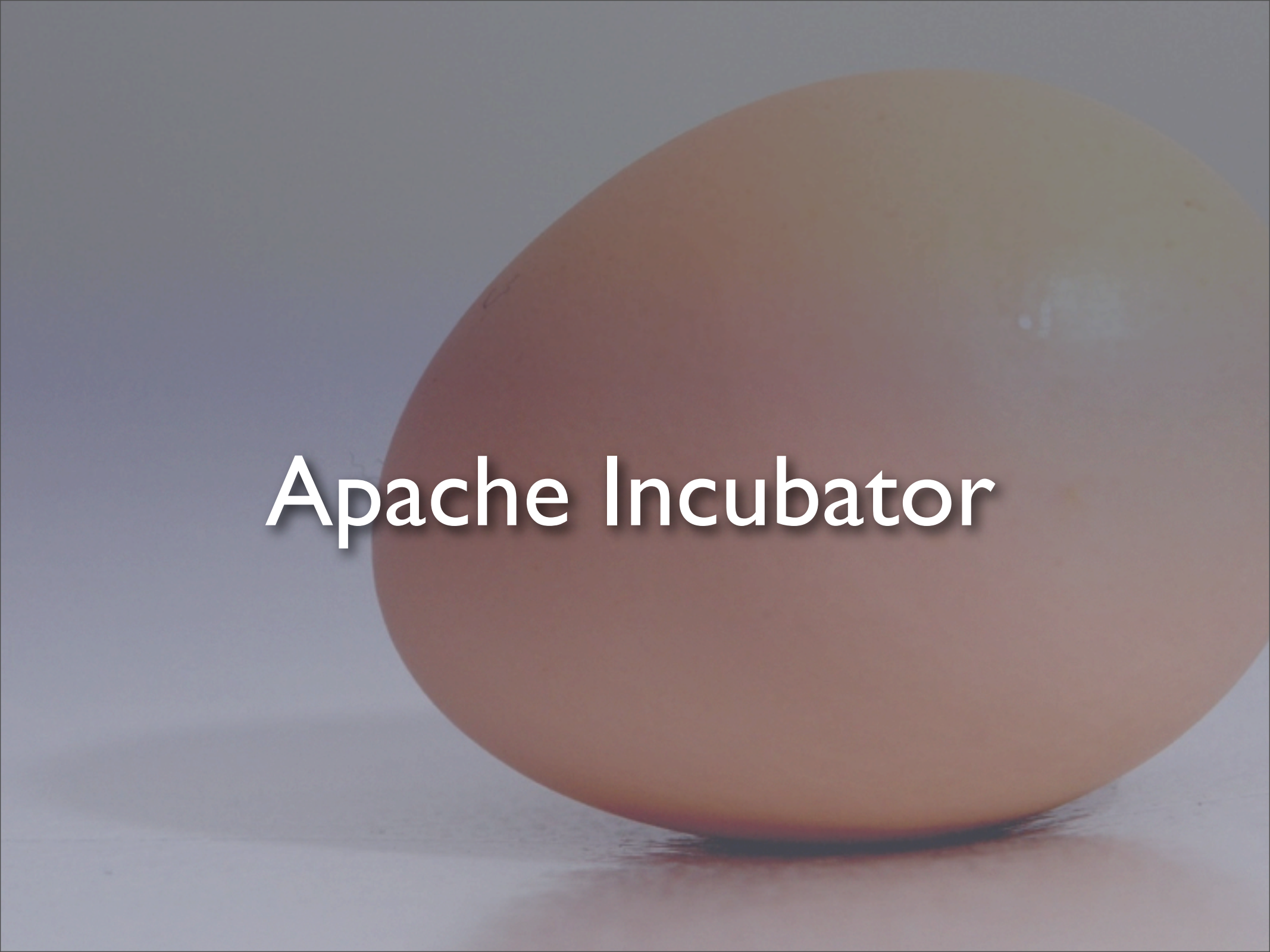
Community

People
Customs
Ownership
Shared Goals
Communication

Community

People
Customs
Ownership
Shared Goals
Communication

Provide A Pathway To Contribution


Apache Incubator

Non-Apache Project

↓

Incubator

↓

Apache Project

Apache Incubator

- Legal

  Ensure donations and projects are in accordance with legal standards

- Community

  Develop communities that adhere to ASF guiding principles

Graduated

ActiveMQ

Apollo

Beehive

XMLBeans

Spam Assassin

Geronimo

Pluto

Hermes

Jackrabbit

JaxMe

jUDDI

Merlin Dev.

Muse

Tapestry

Tobago

Lenya

log4cxx

http-CLI

Directory

iBatis

MyFaces

Nutch

Derby

JDO

WebWork2

Harmony

OFBiz

Cayenne

Synapse

Solr

Felix

Incubating

Abdera
CeltiXfire
FtpServer
Graffito
Heraldry
Ivy
JuiCE
log4net
log4php
Lokahi
Lucene.net
mod_ftp
NMaven
Ode
OpenEJB
OpenJPA
Qpid
River
Roller
ServiceMix
stdcxx
Tika
Trinidad
TripleSoup
Tuscany
UIMA
wadi
Wicket
Wooden
WSRP4J
XAP
Yoko

Roles

- Incubator PMC (IPMC)
- Podling
- Podling PMC (PPMC)
- Champion
- Sponsor
- Mentor
- Committers

Mentor

An Incubator PMC member who assists the podling through Incubation. Acts as a buffer between the ASF and the podling, ensures all policies and being followed. Podlings can have more than one mentor (3 is recommended). At least one mentor should be an ASF Member.

Incubation Process

Question

The incubator process starts by asking questions:

- Do we want to join Apache?
- Why?
- How?

Search

Then begins the search for the sometimes elusive **champion** and **sponsor**.

Commit

Now the writing of the **proposal** begins.

The Proposal

The proposal captures the **intent** of the podling. It also checks for certain **critical signs** of successful projects.

The Proposal: Status

- Meritocracy
- Community
- Core Developers
- Alignment

The Proposal: Known Risks

- Orphaned Products
- Inexperience with Open Source
- Homogenous Developers
- Reliance on Salaried Developers
- Relationships with other Apache products
- Excessive Fascination with Apache brand

The Proposal: Important Details

- IP Submission Plan
- External Dependencies
- Cryptography

Committed

With the proposal complete, the document is posted to the Incubator mailing list with a call for review and then **vote**.

If the proposal passes the vote, then the podling begins its new life in the Incubator.

Establish

Upon a successful vote, the Apache Infrastructure team will setup a website, mailing lists, subversion repository and issue trackers.

Join

All committers will need to sign and submit a **Contributor License Agreement** (CLA).

Businesses which assign employees to a project should provide a **Corporate CLA**.

Clear

Incoming code must complete an IP Clearance process. **Software grants** are required for code written outside of Apache and donated to the podling. The process must be completed by an ASF member or officer.

Learn

The long haul begins:

Incubation involves learning Apache policies and practices. The mentors will assist in this education process.

Adjust

The podling may have to modify licenses, dependencies, code names, and documentation.

The podling will also be adjusting to new committers, new practices, and new pressures.

Grow

Podlings should seek to grow their **developer community** by adding new committers. A diverse community is a requisite for graduation.

Report

The podling must maintain a **status** report on its website. **Monthy** and then **quarterly** reports to the IPMC are required.

Perform

The podling should perform one or more **releases** during incubation. This ensures podlings understand ASF release practices and encourages community growth.

Podling releases must adhere to all ASF release guidelines -- license compliance, signed, hashed, and so on.

Leave

When the podling has fulfilled all **exit requirements**, it may motion for a vote for graduation. The podling then joins the sponsoring Apache project or becomes a top level project (TLP) of its own.

Exit Requirements

- The podling is a worthy and healthy project
- It truly fits within the ASF framework
- It *gets* the Apache Way

Exit Requirements: Legal

- All code under the Apache License and follows Apache Licensing guidelines
- IP Clearance complete
  - Software grants
  - CLAs
  - Trademark check

Exit Requirements: Community

- Active and diverse community
- No single company or entity as sponsor
- ASF style consensus building and voting practices
- Can tolerate and resolve conflict
- Successful release process

Exit Requirements: Alignment

- Use of other ASF products
- Developing relationships with other ASF projects

Exit Requirements: Infrastructure

- All code in subversion
- Mailing lists active and archived
- Documentation available on website
- Issue tracker created and maintained
- Releases signed

Incubation Process

- Question
- Search
- Commit
- Committed
- Establish
- Join
- Clear
- Learn
- Adjust
- Grow
- Report
- Perform
- Leave

Incubation

Over **30** projects have graduated from the Apache Incubator. The average time is 9 months, but varies from 5 to 18 months. During that time, the average project adds 2 - 3 committers.

Incubation Challenges

There will be mistakes, frustrations, conflicts and challenges.

Dealing With Policy

The Apache Software Foundation has plenty of **rules** and **policies**. These represent the collective wisdom of Apache developers. As such, they are **not static**. Policy may (and probably will) evolve during a podling's incubation.

Suggestion

have patience and work closely with your mentor

Dealing With the IPMC

The IPMC holds the keys to releases and graduation. It is a large and diverse group that discusses policy in open forums.

Suggestion:
submit your reports on time!

Existing Users

During incubation, podlings must do a lot of "cleaning house." The user community of an existing product should be kept informed of the incubation process.

Suggestion

Publish a roadmap and tools for migrating to the new Apache version of the product

Open On-List Communication

All communication should be on-list, not on-site. This can be a problem for projects with committers from a single employer.

Suggestion

All offline communication **must** be reported online

Releasing

Releases must follow all ASF guidelines. Podlings should have completed the IP clearance process before a release. The release requires three positive votes from the IPMC.

Suggestion

Run the ARAT tool on your code before a release vote

Is the Incubator For Us?

incubator.apache.org

Thank You
謝謝

J Aaron Farr

farra@apache.org
www.jadetower.com

www.cubiclemuses.com/cm/files/incubating_communities.pdf