How The ASF Views Open Source
-or-
“The Apache Way”

Jim Jagielski
Agenda

• Introduction
• What is the ASF
• What exactly is “Open Source”
• How the ASF Does It
• How The ASF Views It
Introduction

- Jim Jagielski
  - Longest still-active developer/contributor
  - Co-founder of the ASF
  - Member, Director and President
The ASF

• ASF == The Apache Software Foundation
• Before the ASF there was “The Apache Group”
• The ASF was incorporated in 1999
The ASF

- Non-profit corporation founded in 1999
- 501(c)3 charity
- Volunteer organization
- Virtual world-wide organization
- Exists to provide the organizational, legal, and financial support for various OSS projects
The ASF - then

- Started with 21 members
- 2 projects
- All servers and services donated
The ASF - now

• We have 330 members...
• >70 TLPs
• ~25 Incubator podlings
• Tons of committers (literally)
  – (Over 2500 people)
• Very large and growing infrastructure
The ASF’s Mission

• Provide open source software to the public free of charge

• Provide a foundation for open, collaborative software development projects by supplying hardware, communication, and business infrastructure

• Create an independent legal entity to which companies and individuals can donate resources and be assured that those resources will be used for the public benefit
The ASF’s Mission

- Provide a means for individual volunteers to be sheltered from legal suits directed at the Foundation’s projects
- Protect the ‘Apache’ brand, as applied to its software products, from being abused by other organizations
- Provide legal and technical infrastructure for open source software development and to perform appropriate oversight of such software
How We Work

• The Apache Software Foundation provides support for the Apache community of open-source software projects. The Apache projects are characterized by a collaborative, consensus based development process, an open and pragmatic software license, and a desire to create high quality software that leads the way in its field. We consider ourselves not simply a group of projects sharing a server, but rather a community of developers and users.
How We Work, Take 2

• Community created code
• Our code should be exceptional
Structure of the ASF - dev

• Volunteer Driven Organization
• Software Projects are managed by Project Management Committees (PMCs)
• PMCs vote in new PMC members and committers
• At the end of the day: People / Individual focused
Structure of the ASF - legal

- Member-based corporation - individuals only
- Members nominate and elect new members
- Members elect a board - 9 seats
- Semi-annual meetings via IRC
- Each PMC has a Chair - eyes and ears of the board (oversight only)
ASF “Org Chart”

- Development
  - PMC Members
  - Committers
  - Contributors
  - Patchers/Buggers
  - Users

- Administrative
  - Members
  - Officers
  - Board
Issues with Dual Stacks

• Despite clear differentiation, sometimes there are leaks
  – eg: PMC chair seen as “lead” developer
• Sometimes officers are assumed to have too much power if they venture into development issues
  – “hats”
Why Open Source?

• Access to the source code
• Avoid vendor lock-in (or worse!)
• Much better software
• Better security record (more eyes)
• Much more nimble development - frequent releases
• Direct user input
The draw of Open Source

- Having a real impact in the development and direction of IT
- Personal satisfaction: I wrote that!
- Sense of membership in a community
- Sense of accomplishment - very quick turnaround times
- Developers and engineers love to tinker - huge opportunity to do so
Open Source FUD

- No quality or quality control
- Prevents or slows development
- Have to “give it away for free”
- No real innovation
What is Open Source?

• Open Source Licensing
  – OSI Approved
• Free Software
  – As in Free Speech, not Free Beer
• Open Source Development (secondary)
  – ala, the Apache Software Foundation
What is Open Source?

• Basically, it’s a “new” way to develop, license and distribute code
• Actually, there was “open source” even before it was called that
• The key technologies behind the Internet and the Web are all Open Source based
True Open Source

• For software to be Open Source, it must be under an OSI approved Open Source License
• At last count, over 70 exist
Open Source Licenses

• Give Me Credit
  – AL, BSD, MIT
• Give Me Fixes
  – (L)GPL, EPL, MPL
• Give Me Everything
  – GPL
The Apache License (AL)

• A liberal open source software license - BSD-like
• Business friendly
• Requires attribution
• Includes Patent Grant
• Easily reused by other projects & organizations
Give Me Fixes

• MPL / EPL / (L)GPL
  – Used mostly with platforms or libraries
  – Protects the licensed code, but allows larger derivative works with different licensing
  – Still very business friendly
Give Me Everything

- GPL (copyleft)
  - Derivative works also under GPL
  - Linked works could also be under GPL
  - Viral nature may likely limit adoption
  - GPL trumps all others or else incompatible
License Differences

• Mainly involve the licensing of derivative works
• Only really applies during (re)distribution of work
• Where the “freedom” should be mostly focused: the user or the code itself
One True License

• There is no such thing
• Licensing is selected to address what you are trying to do
• In general, Open Standards do better with AL-like license
The Apache Way

- Although the term is deprecated, “The Apache Way” relates to how the ASF (and its projects) work and operate
- Basically, the least common denominators on how PMCs operate
Basic Memes

- Meritocracy
- Peer-based
- Consensus decision making
- Collaborative development
- Responsible oversight
• "Govern by Merit"
• Merit is based on what you do
• Merit never expires
• Those with merit, get more responsibility
Peer-based

- Developers represent themselves - individuals
- Mutual trust and respect
- All votes hold the same weight
- Community created code
  - Healthy communities create healthy code
  - Poisonous communities don’t
Look Familiar?

• These concepts are not new or unique
• Best practices regarding how the Scientific and Health community works
Publish or Perish

- In Open Source, frequent releases indicate healthy activity
- What is collaborative s/w development other than peer review?
- Think how restrictive research would be w/o open communication
Why Community -> Code

• Since we are all volunteers, people’s time and interests change
• A healthy community is “warm and inviting” and encourages a continued influx of developers
• Poisonous people/communities turn people off, and the project will die
• End result - better code, long-term code
Consensus decision making

- Key is the idea of voting
  - +1 - yes
  - +0 - no real comment
  - -1 - veto

- Sometimes you’ll also see stuff like -0, -0.5, etc…
Voting

- The main intent is to gauge developer acceptance
- Vetos must be justifiable and have sound technical merit
- If valid, Vetos cannot be overruled
- Vetos are very rare
Commit Process

• Review Then Commit (RTC)
  – A patch is submitted to the project for inclusion
  – If at least 3 +1s and no -1s, code is committed
  – Good for stable branches
  – Ensures enough “eyes on the code” on a direct-to-release path
Commit Process

• Commit Then Review (CTR)
  – A patch is committed directly to the code
  – Review Process happens post commit
  – Good for development branches
  – Depends on people doing reviews after the fact
  – Allows very fast development
Commit Process

• Lazy Consensus
  – variant of RTC
  – “I plan on committing this in 3 days”
  – Provides opportunity for oversight, but with known “deadline”
  – As always, can be vetoed after the fact
Collaborative Development

- Code is developed by the community
- Voting ensures at least 3 active developers
- Development done online and on-list
  - If it didn’t happen on-list, it didn’t happen
Collaborative Development

• Mailing lists are the preferred method
  – Archived
  – Asynchronous
  – Available to anyone - public list
Collaborative Development

• Other methods are OK, if not primary
  – Wikis
  – IRC
  – F2F

• Always bring back to the list
Responsible Oversight

- Ensure license compliance
- Track IP
- Quality code
- Quality community
The Apache Incubator

• Entry point for all new projects and codebases
• Indoctrinates the Apache Way to the podling
• Ensures and tracks IP
Contributor License Agreement

• aka: iCLA (for individual)
• Required of all committers
• Guarantees:
  – The person has the authority to commit the code
  – That the ASF can relicense the code
• Does NOT assign copyright
Success Stories - HTTPD

• Apache HTTP Server ("Apache")
  – Reference implementation of HTTP
  – Most popular web server in existence
  – Found in numerous commercial web servers
    • Oracle, IBM,...
  – Influenced countless more
Success Stories - HTTPD

• By having a “free” and open source reference implementation, the drive to create a separate proprietary version was reduced.

• “Why spend time and money, when we can use this”

• This allowed HTTP (and the Web) to grow and STAY usable (compare to the old browser wars)
Success Stories - Tomcat

• Apache Tomcat (Servlet Container)
  – The default standard servlet container
  – Each version maps to a specific spec.
  – Bundled with numerous Java apps out there
  – Likely a major influence on the diminishing relevance of JEE
Success Stories - Others

- Apache Geronimo (Server Framework - JEE)
  - Not “just” a JEE server
- Apache Maven (Java project build/management tool)
  - Another industry standard
- Apache Logging, Axis, Struts, ...
Internal Advantages

• Much tighter community
  – better communication (esp. with “users”)
  – tighter feedback loop
• Ready made audience
• Don’t need to “sell” the aspects of Open Source which scare/confuse IT people
Governance Gotchas

• Trust and Merit are earned, which implies time
• The available pool might be relatively small
• How to encourage external involvement – “scratching your own itch”
• Mentoring: lurking on lists vs. direct involvement
Some Concluding Thoughts

• Trust your developers AND your users
• Communication is key
• Open Source is NOT the Good Housekeeping Seal Of Approval
• But don’t believe in all the FUD either
• Success is not measured in market share, but in adoption
Some Concluding Thoughts

• Open Source should have a viable business or emotional reason
• Give some thought to licensing
• Make it easier for developers and users to “join”
• Give them a reason to
Helpful links

• The Apache Software Foundation
  – www.apache.org

• Want to help a great organization?
  – www.marylandstateboychoir.org
That’s It

- Thank you!
- Any questions?