

Building Open Source Communities

Ir. Robert VISEUR

Faculté Polytechnique de Mons (FPMs)

Centre d'Excellence en Technologies de l'Information et de la
Communication (CETIC)

robert.viseur@fpms.ac.be

Linuxdays 2007

CRP Centre Henri Tudor

Luxembourg, 01 February 2007



ACADÉMIE UNIVERSITAIRE WALLONIE-BRUXELLES



What is Open Source Software ?

(1)

- Two definitions given by Free Software Foundation (freedoms : edit, study, modify and distribute) and Open Source Initiative (10 criteria)
- A set of technologies
 - Languages : Perl, Python, PHP, C/C++ (GCC), Java,...
 - Libraries and frameworks : Qt, PEAR, Zend,...
 - Applications : Open Office, Firefox, ERP

What is Open Source Software ?

(2)

- A set of economic features
 - Specific business models
 - Specific licenses
 - Strong link between business models and licenses
 - Direct influence on business models
- A hope about mutualisation of development process
 - Community building process \leq my theme
 - Real added value of Open Source

Rules about Building Community Process

- Rules about co-creation management
- Rules about community management
- Rules about software management

Create an Open Source community : it's easy...

- Few writings about Open source Software Management
- In general : some weak principles and... good luck !
- Example : Eric Raymond
- What we can read about this :
 - Open development process is better than structured process (see debates about bazaar vs cathedral)
 - Release soon as possible
 - KISS principle : "Keep it Simple, Stupid"
 - Software architecture is not important

Some examples as starter

- Failures
- Success

Some famous examples of "failures"

- Failures ?
 - *When something works wrong in the Open Source process*
- Examples
 - Mozilla
 - PHP Nuke
 - About forks

Mozilla – History (1)

- < 1998 : Netscape used iterative development
- 1998 : Netscape launched Open Source project (Netscape 5 => Mozilla)
- 1998 : Heavy loss of market shares for Netscape
- 2000 : Netscape 6.x failed (Beta software)
- 2002 : Netscape is dying out
- 2003 : Phoenix (Firefox) Project

Mozilla – History (2)

- 2005 :
 - Mozilla Corporation is created
 - Mozilla realized a revenue of \$52,9 millions in 2005
- 2006 : Firefox has more or less 20% market shares in Europe

Mozilla – Major causes of failure (1)

- Poor quality of first Open Source release
 - Source code too complex to modify, thus, few contributors.
 - Many bugs in source code inherited from Netscape 5
 - => development of a new clean code (1998-2000)
 - So loss of time (IE wins)
 - But, at the end, quality, usability and popularity (with Firefox)

Mozilla – Major causes of failure (2)

- Two branches developed in parallel, only one really used
 - Loss of energy
 - Lack of incentive : contributors want to view their contributions in the software they use.

PHP Nuke

- One of the first popular CMS
- Many forks (the Nukes) : Xoops, eXoops, Postnuke, NPDS,... with more or less clear technical orientation (focus on OO, on security,...)
- Origins :
 - Poor technical knowledge of leader Fransisco Burzi (one consequence : security flaws)
 - Closed development process
 - Temptation for proprietary come back (due to problem of business model)

About forks (1)

- A way for new experimentations or specialization
 - Specialization of *BSD OS (simplicity for FreeBSD, portability for NetBSD, security for OpenBSD)
 - More disruptive evolutions in Samba-TNG (with sharing of codes)
- But also sometimes a wind of change, and extinction of the forked project
 - XFree86 forked in X.org (Keith Packard) with global approbation of users (including companies)

About forks (2)

- But also sometimes a loss of energy
 - Claroline forked in Dokeos for brand policy problem.
 - Emacs forked in Xemacs first for license application conflict

Some famous success

- Apache
- Eclipse

Apache (1)

- Strong market shares with Apache HTTPD
- New recognized projects : Tomcat, Struts, Cocoon,...

Apache (2)

- Some reasons :
 - Based on already popular NCSA server
 - Real mutualization organized by lead users
 - Modular architecture adapted to mutualization
 - Growth by "acquisition" (Lucene, Nutch, Spamassassin,...) and ecosystem (PHP,...)
 - Freedom for innovation in the incubator (good for creativity and future growth)
 - Structured but opened and democratic management structure

Eclipse

- Strong market share as IDE (Java world oriented)
- Some reasons :
 - Circumstantial value ("Eclipse the sun")
 - One of the first movers in the domain
 - Project promoted by lead Open Source actor (IBM)
 - Organization of "coopetition" (and next independence of community) with BEA, Borland, etc
 - Modular architecture (Eclipse is also a toolkit)

Open Source as co-creation

- Co-creation means "*value creation between consumer and producer of value*".
- Examples : Open Source, Web 2.0 (*user generated contents, affiliation,...*), etc.

Some general rules for co-creation (1)

- Define the objectives
 - What are the outlines of the project ?
- Good choice of co-creators
 - Do I collaborate with "long tail" or professionals or specific users (i.e. lead users) ?
- Define the rights and duties of each ones
 - Which license do I choose ?
 - What is my business model (linked to license) ?

Some general rules for co-creation (2)

- Control the communication ways
- Outsource the management (if necessary)
 - Some public tools such as Sourceforge, some software available such as GForge but...
 - Management of co-creation (so Open Source projects) needs specific competences
- Simplify the co-creation
 - Are there well known tools ?
 - Can I help the knowledge transfer to newbies ?
- Warning : shortsightedness of user

Open Source, community and identity

- Community = set of people close by common interests or opinions.
 - Here (Open Source) : common project
- Identity
 - Some "symbols" : name, logos, colors, mascots,...
 - Some values
 - Complementarity with official rules
- Physical meetings can be good social events (examples of sprint in Python world)
- Challenge to traditional "authority" of the company

What can you ask your contributors to do ? (1)

- They do with pleasure :
 - Point out a bug
 - Make a translation
 - Give feedback
- They do sometimes :
 - Write a new function, a new feature
 - Fix a bug

What can you ask your contributors to do ? (2)

- But following ones are boring :
 - Professional testing
 - Writing specifications
 - Writing documentation

Reminder about choice of co-creators and community (1)

- Continuum of choices
- Three ways, three examples
 - IDX-PKI : softwares for PKI infrastructure, edited by IdealX (France). Co-development in group combination of clients and IdealX (discussion about the needs). Publication on public website from time to time.
 - Eclipse : strong partnerships with companies (sometimes competitors).
 - Mozilla : free contributions from "long tail".

Reminder about choice of co-creators and community (2)

- Don't forget :
 - Need of a core team
 - Rights : who can update the sources ?

Some specific rules about Open Source / Conclusions (1)

- Importance of the architecture
 - Well engineered project is compatible with heterogeneous needs and simplify the mutualisation
 - Don't let you be crunched by specifications papers but specifications are important for long term success for the project
 - Always think about collaborative development process (simplicity, homogeneity of tools, modularity, etc) and internationalisation

Some specific rules about Open Source / Conclusions (2)

- Importance of an Open Management
 - Decision can be taken in concertation with all the contributors
 - Interest of institutional partnerships
 - Open Source is a way to stimulate collaboration without the heaviness (and the risks) of traditional commercial contracts
 - Open Source = good tool for coopetition (coopetition = cooperation between competitors)
 - Institution = companies or non profit (Objectweb, Apache,...)

Some specific rules about Open Source / Conclusions (3)

- Importance of an Open Management
 - ...
 - Make links with others communities
 - Better mutualisation
 - External growth factor
 - Notion of ecosystem

Some specific rules about Open Source / Conclusions (4)

- For companies : promote communication between employees and community members
 - Often : leader moderator in company
 - Rules in corporate communication in company
 - Need of more human communication
 - What are the boundaries of the openness ("private life" of the company)

Some specific rules about Open Source / Conclusions (5)

- Promote creativity and education
 - Simplify the transfer of competences : documentation, tutorials,...
 - Warning : there are several types of clients
 - => simple users (to download and to use), contributors (to download and to participate : translation, bug fixes,...)
 - => heterogeneity in competences
 - Leave a place for experimentation (as sandbox, incubators)

Ingredients for success ?

**« Good product, good place,
good management »**

Questions ?