Managing rights in free/libre/open source software

PRIN Workshop, 25 November 2005
University of Bologna

Rishab Aiyer Ghosh
rishab@dxm.org

MERIT, University of Maastricht
Based on research conducted under the EU FP5 IST FLOSS project and EU FP6 IST CALIBRE project

© 2005 Rishab Aiyer Ghosh
Distributed under the Creative Commons Atrrib.-NonCommercial-NoDerives Licence

Rights, protection and creativity

State granted temporary monopolies as an incentive for creativity and innovation: copyright, patents...

These protect *creations,* not necessarily *creativity*

Past creations are rewarded while future creations become more difficult to build

© 2005 Rishab Aiyer Ghosh

Mechanics of rights protection

A work is created. It is then exclusively appropriated by the creator, with a limited, temporary monopoly granted by the state.

Monopoly rewards creator.

Monopoly provides incentive for future creation.

© 2005 Rishab Aiyer Ghosh

Mechanics of rights protection

Copyright: monopoly is on reproducing work, preventing follow-on creation.

Patent: monopoly extends to prevent independent creation.

© 2005 Rishab Aiyer Ghosh

Collaborative creation

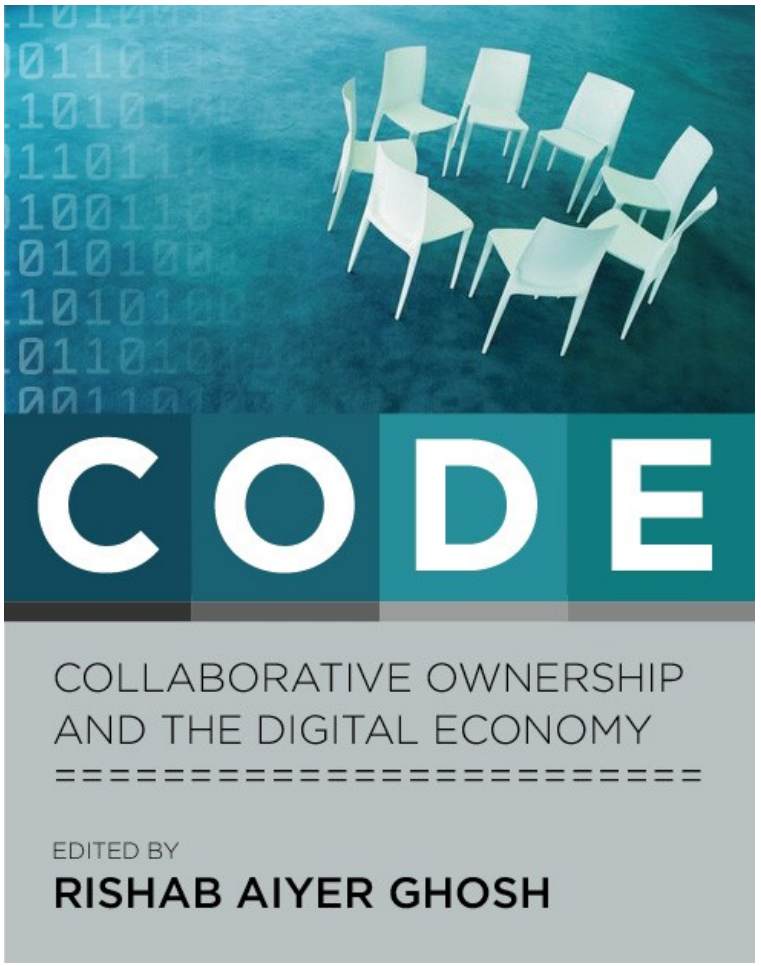
Exclusive appropriation is not a pre-requisite for innovation. Indeed, it may hamper innovation.

Many forms of "Collaborative ownership and development" in history, in different societies

Free/Libre/Open Source Software (FLOSS) most recent, headline-grabbing incarnation

© 2005 Rishab Aiyer Ghosh

Collaborative production



CODE: Collaborative Ownership and the Digital Economy

MIT Press, 2005
Rishab A. Ghosh, ed.

Philippe Aigrain, Yochai Benkler, Boatema Boateng, David Bollier, James Boyle, John Clippinger, Paul David, Cori Hayden, Tim Hubbard, Chris Kelty, James Leach, James Love, Fred Myers, Anthony Seeger, Richard Stallman, Marilyn Strathern

© 2005 Rishab Aiyer Ghosh

Extent of creativity in FLOSS

Debian 2.2 GNU/Linux (2001):

- Source lines of code: 55,201,526

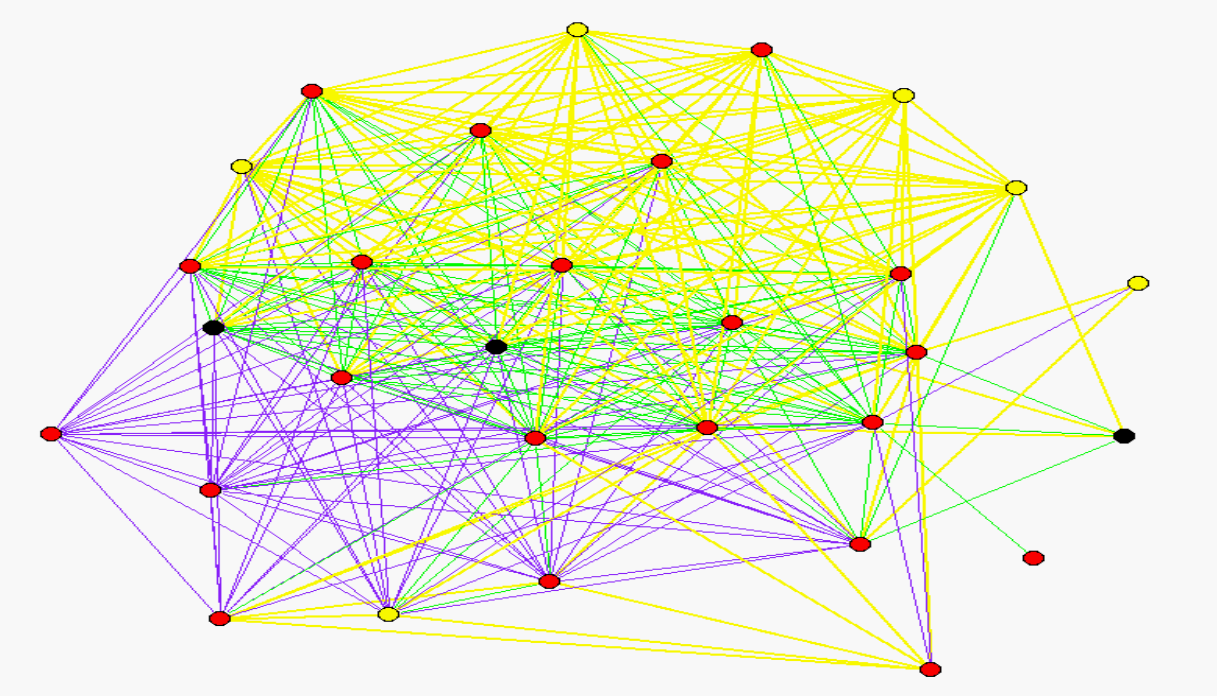
If Debian was written in a software company:

- Estimated effort: 14,005 person years
- Estimated schedule: 6.04 years (team of 2,318)
- Development cost: US$ 1,891,990,000

(Source: “Counting potatoes” by Gonzalez-Barahona et al)

© 2005 Rishab Aiyer Ghosh

Extent of collaboration in FLOSS

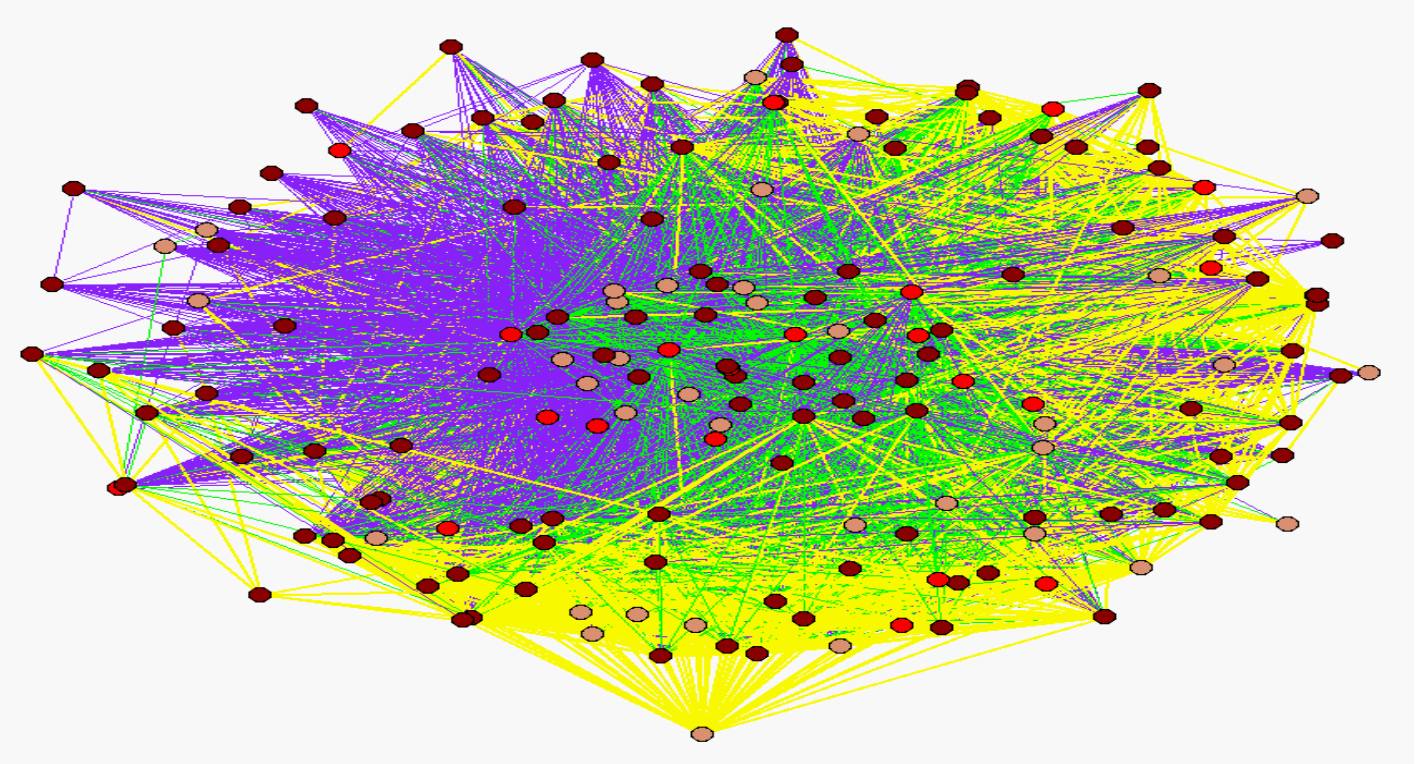


Linux Kernel v1.0. 1994. 158 authors. Nodes are 30 modules.
Arcs represent common authors, code dependencies, or both

(Source: "Nature and composition... ", Ghosh & David)

© 2005 Rishab Aiyer Ghosh

Extent of collaboration in FLOSS



Linux Kernel v2.5.25. 2002. 2263 authors. Nodes are 169 modules.
Arcs represent common authors, code dependencies, or both

(Source: "Nature and composition... ", Ghosh & David)

© 2005 Rishab Aiyer Ghosh

FLOSS: product innovation

Scripting languages (Perl, Python...)

Dynamic webservers (Apache)

Application development (Zope, Plone)

Multimedia (VideoLAN)

Clustered computing (Beowulf)

© 2005 Rishab Aiyer Ghosh

FLOSS: process innovation

Massively distributed development

User-driven development

Rapid prototyping

Quantifiable improvements (e.g. high proportion of “young” lines of code; ability to sustainably increase software complexity and team size)

© 2005 Rishab Aiyer Ghosh

FLOSS: rights

FLOSS is not public domain

Software with “Four Freedoms” (Stallman)

- Freedom to use
- Freedom to study
- Freedom to modify
- Freedom to share

© 2005 Rishab Aiyer Ghosh

FLOSS: rights

Software authors own their code

Software authors, as owners, can safely "give it away"

They can impose conditions

Conditions may perpetuate "Four Freedoms"

© 2005 Rishab Aiyer Ghosh

FLOSS: rights

Software authors own their code

- Software is automatically protected by copyright
- Copyright belongs to the *authors* (or employers)
- Authors have the sole right to license their software
- Software users must follow licence terms – otherwise they are infringing authors' copyright

© 2005 Rishab Aiyer Ghosh

FLOSS: rights

Software authors can safely "give it away"

- "Giving it away" is releasing to the public domain
- This is rare
- Two licensing models: *permissive* and *reciprocal*
- *Permissive:* close to public domain
- *Reciprocal:* not public domain, "protected commons"

© 2005 Rishab Aiyer Ghosh

FLOSS: rights

Software authors can impose conditions

- Proprietary licence terms add restrictions above the protections of copyright: restrictive contracts
- FLOSS licences only place conditions on the use of copyright: not necessarily contracts
- Without following conditions, users are infringing copyright

© 2005 Rishab Aiyer Ghosh

FLOSS: obligations

Permissive: terms close to public domain

...with liability and warranty disclaimers

Examples: BSD, Apache...

© 2005 Rishab Aiyer Ghosh

FLOSS: obligations

Reciprocal: terms to propagate "Four Freedoms"

- Derivative works must be distributed under the same terms: *works derived from FLOSS remain FLOSS*
- Examples: GPL, LGPL, Mozilla Public License

© 2005 Rishab Aiyer Ghosh

FLOSS: protected commons

Reciprocal licences ensure that development remains collaborative, cannot be exclusively appropriated

Relies on copyright for legal validity

Reciprocity conditions are enforceable (simply under copyright law)

Reciprocity provides incentive for new contributors, including firms

© 2005 Rishab Aiyer Ghosh

Reciprocity and incentives

60% of developers think the role of a licence is "To prevent others from appropriating the software we've created" (FLOSS-US survey)

Firms prefer GPL because "It allows to keep the code open and forbids competitors to turn it into proprietary." (Bonaccorsi & Rossi)

Reciprocity requirements encourage disclosure, collaboration and innovation

© 2005 Rishab Aiyer Ghosh

FLOSS misconceptions

FLOSS prevents commercial activity

FLOSS software can be “taken over” (by a firm)

GPL deprives authors of their rights

GPL deprives firms of their rights

FLOSS (GPL) prevents innovation

© 2005 Rishab Aiyer Ghosh

FLOSS misconceptions

FLOSS prevents commercial activity

- By definition (OSI, FSF), a licence that *prevents* charging for software is not a FLOSS licence
- *Source code* cannot be charged for
- But source code must *only* be given, under a licence allowing modification, to recipients of binary code (which can be charged for)
- Recipients can redistribute, which drives *sale price* to zero; commercial models are usually service-based

© 2005 Rishab Aiyer Ghosh

FLOSS misconceptions

FLOSS software can be “taken over” (by a firm)

- Yes, with permissive licences (BSD code in Windows)
- No, with reciprocal licences
- GPL ensures that even original copyright holders can only “take over” new versions of the software
- Old versions remain irrevocably FLOSS
- Linux kernel: >2500 individual copyright holders

© 2005 Rishab Aiyer Ghosh

FLOSS misconceptions

GPL deprives authors of their rights

- Original authors *choose* to "give away" their software;
- Authors are not bound by GPL terms for *licensees*
- New contributors have full rights to their own contributions – but they cannot build upon another work without respecting the rights of the first work's author.
- New authors have no "right" to infringe original authors' copyright – by violating GPL terms!

© 2005 Rishab Aiyer Ghosh

FLOSS misconceptions

GPL deprives firms of their rights

- As with authors' rights – firms can dual-license
- Firms have no “right” to software created by others
- Firms (developers) making significant contributions to FLOSS prefer reciprocal licences such as the GPL
- Firms (users) making minor contributions prefer the MPL/CDDL (with limited reciprocity requirements)
- Firms (users) that prefer permissive licences want to appropriate the work of others!

© 2005 Rishab Aiyer Ghosh

FLOSS misconceptions

FLOSS (GPL) prevents innovation

- GPL incentivises disclosure (> patent system)
- GPL prevents disclosure from leading to 3rd-party appropriation (> patent system)
- GPL encourages follow-on innovation (> patent system)

© 2005 Rishab Aiyer Ghosh

FLOSS misconceptions

FLOSS (GPL) prevents innovation

- FLOSS prevents monetisation only (partially) through sale price
- Packaged software is <30% of US market, <10-20% elsewhere. Most software innovation is monetised otherwise (customisation, integration, services)

© 2005 Rishab Aiyer Ghosh

FLOSS and patents

FLOSS rights management assumes that creators have sole rights to what they create

Copyright: independent creation allowed

Creators can be certain that they have sole rights to what they create, and safely "give it away"

© 2005 Rishab Aiyer Ghosh

FLOSS and patents

FLOSS rights management assumes that creators have sole rights to what they create

Copyright: independent creation allowed -

Patents: independent creation can infringe

Patents deprive creators of sole rights to their creation – so they *cannot* safely "give it away"

Patents are essentially incompatible with FLOSS

© 2005 Rishab Aiyer Ghosh

References

FLOSS survey, CALIBRE project (MERIT):
flossproject.org, www.calibre.ie

FLOSS-US survey (Stanford – Paul David et al):
www.stanford.edu/group/floss-us/stats/q7.html

Ghosh & David 2003, "Nature and composition of the Linux kernel",
Stanford SIEPR working paper, http://dxm.org/papers/licks1/

Gonzalez-Barahona et al, 2003, "Counting Potatoes", http://libresoft.urjc.es

Bonaccorsi, Andrea & Cristina Rossi
Licensing schemes in the production and distribution of Open Source software. An empirical investigation: http://opensource.mit.edu/papers/

© 2005 Rishab Aiyer Ghosh