OKCon 2011

What kind of a commons is free software?

Miguel Said Vieira

PhD student (Philosophy of Education),
University of São Paulo / Scientiae Studia
Epidemia Collective

Outline

1. Commons studies
2. Immaterial commons
3. Proposal; two layers of analysis (free software)
    1. Use
    2. Development
4. Examples, some final thoughts

Commons studies

- Commons: community sharing things
- Debate regarding free software [FS]: open access, or managed commons?
  - *open access*: freedoms in FS licenses
  - *managed*: empirical studies show communities are structured, and follow some principles and norms
- Importance of the question: most traditions consider open access as “non-commons”

Neoinstitutional approach

- Most renowned / successful approach
  - Elinor Ostrom: Nobel Prize in Economics
  - strong influence from neoinstitutional economics
- Ostrom disproved Hardin's "Tragedy of the Commons":
  - commons are not doomed to overuse
  - "soft" methodological individualism
  - empirical studies; led to design principles
  - open access: lack boundaries, rules; thus, should not be sustainable

Immaterial commons

- Ostrom's work: small-scale, material commons; what about the sharing of knowledge?
- Economists' typology of goods: immaterial = public good
  - not easily *excludable*, also not *rival*; (material commons: not *excludable*, but *rival*)
  - Nina Paley's *Copying Is Not Theft*
  - that could explain why open access works here.
- But... there's always a “but”. :-)

Rivality, excludability: intrinsic characteristics?

- Rivality, excludability:
  - are not binary variables, but a continuum
  - are also not absolute givens
- Change in time and space (for the same good)
  - *time*: a software now and 30 years ago (TeX, e.g.)
  - *space*: a software in Silicon Valley and in Africa
- Historical and social codetermination
  - Methodological individualism coupled with essentialist approach to material world: problems

Alternative approaches; *commoning*

- Peter Linebaugh, marxist historian: “there's no commons without commoning”
  - focus on the social bonds and political struggles from which commons (or enclosure) arise
  - less mechanistic view of community / goods relation
- Broader view; might “scale” better to analyse larger commons, or their fit in capitalist society
- But: still not as systematic as the neoinstitutional approach (far from it)

Two layers of analysis

1. Use in general
2. Development

- They're interdependent
- There is some flow (and overlap) between them
  - but also differences, from a commons perspective

Dual nature of (free) software

- Source code / machine code
  - Machine code *performs* the software's functions
  - ...but software is *developed* in source code
- My proposal for looking to free software from a commons perspective mirrors this duality

First layer: use in general

- Wider layer
  - use is central; may include modification practices, but not in a systematic way
- *Community*: everyone that uses the software
- *Resource pool*: all pieces of software under FSD/OSD licenses
  - forms a single commons
- *Governance*: mostly based on the freedoms granted by the licenses
  - freedom 0 (to use) as a baseline

First layer: what kind of commons?

- For the most part, this layer is open access
  - there are rules “only” when there's redistribution (but: redistribution is prerequisite to sharing)
- However, abiding by rules depends only on the will of the commoner-to-be
  - membership is not refused based on ad-hoc rules, or on limits to the size of the community
    - *intensional* definition of community (vs. *extensional* definition, in material commons)

Second layer: development

- Expanded notion of development:
  - documenting, evangelizing, bug-testing, translating
- Still, *communities* are subset of previous layer's
- *Resource pool*: each individual FS project
  - multiple commons (and drivers, needs, principles)
- *Governance* less based on licenses;
  more on other systems of rules (formal or not)
  - Debian Social Contract / Constitution
  - informal rules at play all the time; meritocracy, i.a.

Second layer: what kind of commons?

- Managed commons; effective participation can be restricted (criteria vary in each commons)
  - many different levels of participation and authority
- Values and principles underpinning those rules also can vary a lot
  - Debian mixes meritocracy, democratic procedures and formal authority

Android as example of closed management (2nd layer)

- Software stack, includes Linux and non-free sw
- “explicitly open source (as opposed to FS)”
  - but even source is not always open...
- Project is quite open for apps' developers, but very closed for handset producers
- Google employees in gate-keeping positions
  - criteria for accepting code is not only meritocracy, but also “alignment with Android strategy”
  - Conflict with interests of the 1st layer's community?

Forking: example of interdependence

- Right to fork is formally in the 1st layer's rules
- But its legitimacy is determined on the ground, according to 2nd layer's rules
  - 2nd layer community might refuse to cooperate (severe split between communities)
- In a way, what's pooled in the 2nd layer is also developer's hours of work
  - much more rival than software!

Final thoughts

- Methodological remark: “nested enterprises”?
  - but what Ostrom refers to is somewhat different
- FS as a commons is open access for use
- but excessive focus on this can overshadow importance of the development layer
  - defines projects' directions, and decides whose interests will be cared for
  - must be gradually, cyclically “taken over”, or we'll reinforce a problematic user-producer divide that mirrors other inequalities and power asymmetries

WIP: comments highly appreciated!

Thanks / Obrigado

msaid@usp.br
http://impropriedades.wordpress.com/
[in Portuguese]