

Doing business with free and open-source software

OSM2011
Kaido Kikkas

Auld lang syne

- MIT hackers: academics and hippies
- beginning of the 80s – most went into business, some crazy ones started to revolt (incl. RMS)
- Nevertheless, Stallman lived for some time for just selling software; paid services are OK
- 1989 – GNU GPL; no business restrictions
- Thus business is not the question, but how to do it. The business model is however different
- Free models fit well into the age of Internet

Eric S. Raymond

- „The Cathedral and the Bazaar“, Linux Kongress 1997
- Lots of feedback, incl making Netscape source code free and the rise of Open Source movement
- Later also „Homesteading the Noosphere“ (psychological/motivational aspect) and „The Magic Cauldron“ (economics/business)
- More readings, see <http://www.catb.org/~esr/>
- A good criticism:
http://fare.tunes.org/articles/about_esr.html

The Cathedral and the Bazaar

- Two approaches to software development:
 - Cathedral – a classical planned and managed project, fixed structure, team and schedule
 - Bazaar – community-based development, varying team, the work is completed „when ready“
- A note: while most proprietary software are cathedrals, the initial CatB focused only on Free Software – thus it does not mean Linux vs Windows! Emacs, GCC and the whole GNU were cathedrals too

...

- „Given enough eyeballs, all bugs are shallow“
- The Bazaar model needs a project seed (nucleus)
 - does not have to be large but should
 - work
 - demonstrate its potential
- NB! This is needed in our coursework too – someone from both teams could do the initial version (a very small one) of a given scenario (or it can be done in cooperation – as we use more fixed team roles)

„The Magic Cauldron“

- Analysis of the economic side of Open Source
- Sometimes software is considered similar to any other product by nature. But
 - Up to 95% of all software is not produced 'into the box' but instead made for in-house use
 - Likewise up to 95% of the IT staff time is used not to create new software but to maintain and develop the existing
- In both cases, the payment is actually for the service (labour force; not product) => proprietary software is a) actually a niche phenomenon, b) unfair in often letting the client pay twice

Business models

- Free product making the inroads for closed or hybrid products. E.g. Fedora, OO.o, SUSE (Novell)
- Hardware drivers – tendency seems to be towards open ones (e.g. Intel). On losing the 'IP', Raymond quotes Oleg Kalugin from KGB:
 - For instance, when we stole IBMs in our blueprints, or some other electronic areas which the West made great strides in and we were behind, it would take years to implement the results of our intelligence efforts. By that time, in five or seven years, the West would go forward, and we would have to steal again and again, and we'd fall behind more and more.

...

- „Give away recipe, open a restaurant“ - the main business model of FLOSS: commercial know-how on a free product. First used in 1989 by Cygnus Solutions, later by Red Hat and today by Canonical (Ubuntu)
- Complementary services – selling books and accessories. E.g. O'Reilly, various small firms
- Temporary closed license – e.g. turns into the GPL after a year. Ensures the continuity of the project

...

- Two more models considered „possible in principle“ by Raymond and being successfully used after the release of the CatB:
- selling the trade mark – offering certified services and compatibility/quality certificates for free software (IBM, Novell, Red Hat)
- selling the content - paid subscription service. E.g. Red Hat Network

OSI: why open source is better

- For developer
 - faster development (the first ones to join win the most!)
 - delegation (outsourcing) – a small firm can handle larger projects than with closed models
- For sellers
 - closeness to consumers – circles of developers and users are overlapping
 - wider market – can be ported to new platforms by basically everyone

Martin Fink

- HP Linux director
- *The Business and Economics of Linux and Open Source* , 2002. see also <http://www.linuxcio.com>
- Some thoughts:
 - in proprietary software, a new version is dictated by market needs and firm strategy. In free software, it will come when ready
 - release early, release often – hardcore testing in real conditions
 - in-house software is well-fitting but expensive. The mass product is cheaper but ill-fitting. FLOSS gives the strong points of both!

Brent C. Williams

- A stock analyst thinking like a hacker
- <http://stephesblog.blogs.com/presentations/BrentWilliamsEclipseConV02.pdf>
- A main point: defining the right task is vital!
 - Oracle Linux 2006: RHEL clone, similar services from another large corporation, price much lower. Result: in 90 days, the Oracle variant was downloaded about 9000 times. BUT Red Hat's free Fedora had 6 million downloads




Some points from „Perspectives on Free and Open-Source Software“

- A good collection of articles on FLOSS
- Motivation factors:
 - Intrinsic
 - enjoyment (fun)
 - obligation (giving back)
 - Extrinsic
 - salary (paid FLOSS)
 - scratching a personal itch
 - career advancement (job market signaling - I'm a pro!)
 - human capital (better skills)

...

- A provocative quote from the book:
[I]n every release cycle Microsoft always listens to its most ignorant customers. This is the key to dumbing down each release cycle of software for further assaulting the non-personal-computing population. Linux and OS/2 developers, on the other hand, tend to listen to their smartest customers. . . . The good that Microsoft does in bringing computers to non-users is outdone by the curse that they bring on experienced users (Nadeau 1999).



Sandeep Krishnamurthy on FLOSS business models


- The Distributor (Red Hat, Canonical)
 - product on CD
 - support services
 - upgrade services (esp. in corporate setting)
- Software Producer (Non-GPL)
 - incorporate a free product into a larger code base and create a new product
 - bundle an open source product with one of their own

...

- Software Producer (GPL)
 - accelerates innovation, due to more rapid feedback and input
 - greater inclusion of users builds relationships, and hence loyalty
 - user-built modifications available for the company
 - caution: open both ways, trade secrets are near impossible
- Difference between GPL and non-GPL model:
 - The GPL software producer expects an empowered user who is eager to engage in a two-way conversation
 - The non-GPL software producer wants the recipient of the software to simply use it and do nothing else

...

- Third-Party Service Provider
 - arguably higher-quality service (for pay)
 - local, on-site service (as opposed to global mailing lists etc)
 - especially selling turnkey free product plus service



Krishnamurthy: Strong and weak points

- Advantages:
 - robustness - security, reliability, survivability, availability
 - flexibility to user
 - local community support
- Disadvantages:
 - version proliferation (forks, distros etc)
 - can have usability problems (hackers to hackers only)

Krishnamurthy: Profit potential

- customer applicability - size of potential market
- relative product importance (OS > GUI > core application > peripheral application > small utility)

Combinations:

- Low CA, low RPI – low-profile niche projects (SF)
- Low CA, high RPI – high-profile niche project (Postfix)
- High CA, low RPI – mainstream utilities (archivers)
- High CA, high RPI – star projects (LibreOffice, Ubuntu)
- The last group is the primary candidate for profit

Krishnamurthy: profit factors

- Key factors that affect profit:
 - support from primary developer community (primary source)
 - presence of dominant competitive FLOSS products. OS products do compete for developers, distributors, and customers on two levels - product category level, e.g. Linux vs BSD, and distribution level (Linux distros)
 - presence of dominant competitive closed products - existing ones do have a resource advantage (established vs startup)
 - relative competitive position
 - need for marketing (category and distribution again - first Linux vs other OS, then MY Linux vs others)

BW: flexibility as the main point

- allows rapid deployment of 'product seeds' into existing areas
- allows 'innovation through integration', i.e. rapid construction of new products of existing parts (Lego style)
- allows niche producers to find a better position
- first comers have best chances, but the latter ones will get some too

Some more remarks

- mostly service model (see Raymond)
- in a right market and with a right approach, the price of the support may be quite high (Red Hat) – yet it does not need vendor lock-in
- support is much more valuable service (much more customisation options)
- in ideal case, a middle road between a turnkey solution and providing just 'a box of parts'
- allows to bring in providers of additional services

Conclusion: a new kind of economy?

- José Malaquias:
<http://www.malaquias.net/en/joseluis/articles/copyright.pdf> (can now be found at archive.org)
- Robert Theobald: *mindquake*
- Finite resources – a historical meme?
- Information as a new kind of resource
- See Yochai Benkler: <http://www.benkler.org/>

Final words

- Service-based business models may be fitting better to the Internet age
- Less chance for monopoly profit, yet more chance for just a good one => more winners
- Needs good formulation of tasks and good positioning (the example of Oracle Linux)

Sources & Reading

- Joseph Feller et al (eds). Perspectives on Free and Open-Source Software. MIT Press 2005.
<http://mitpress.mit.edu/catalog/item/default.asp?tid=10477&ttype=2>
- Eric S. Raymond. CatB, HtN and MC.
<http://www.catb.org/~esr/writings/cathedral-bazaar/>
- Martin Fink. The Business and Economics of Linux and Open Source. Prentice Hall 2003.

Homework

- Write a case study about three IT companies using Open Source as a part of their business strategy (one of them could do it a its main business, e.g. Red Hat).

Due: Monday, October 10



</business>
