

Hybrid Innovation: How Does the Collaboration Between the FLOSS Community and Corporations Happen?

Yuwei Lin
yuwei {at} ylin {dot} org

The paper will appear in the journal 'Knowledge, Technology and Policy' forthcoming in summer year 2005. Please refer your citation to that reference.

Abstract

Unlike innovation based on a strong professional culture involving close collaboration between professionals in academia and/or corporations, the current Free/Libre Open Source Software (FLOSS) development entails a global knowledge network, which consists of 1) a heterogeneous community of individuals and organisations who do not necessarily have professional backgrounds in computer science but competent skills to understand programming and working in a public domain; 2) corporations. This paper highlights the importance of the hybrid form of developing and implementing software, and also identifies several key factors shaping the collaboration between OSS firms and the community.

Author Bio

Yuwei Lin, Taiwanese, is a research fellow based at the Department of Information Systems, Marketing and Logistics at the Vrije Universiteit Amsterdam. She received her PhD in Sociology from the University of York (UK) in year 2004. Her PhD research investigated the heterogeneity and contingency in the Free/Libre Open Source Software (FLOSS) social worlds, which is based on a constellation of hacking practices, from the sociological perspective. Her principal research interests centre on FLOSS studies, Science and Technology Studies (STS), virtual communities and knowledge management. She can be reached at <ylin@feweb.vu.nl> or through her website at: <<http://www.ylin.org>>.

Introduction

With the maturation and the increased number of FLOSS-based products, the current FLOSS development denotes a hybrid innovation model, which takes the advantage of acquiring resources both from the communityⁱ and firmsⁱⁱ. The community offers an

almost boundless space for experimental projects and interactions between diverse actors, while firms stabilises and standardises the FLOSS-based products by incorporating them together and distributing in the market or by supporting related services. Unlike working in an informal innovation ambience where shared interests are the main concern for volunteer developers, after joining a firm one has to engage in the operation of a more institutionised group, working on specific projects, with certain colleagues. However, such a formalised and institutionalised working partnership does not mean that firm-based developers have terminated their connections with the community. By contrast, previous (informal) cooperation on parallel community projects remains of significance in these firm-based developers' daily practices.

The paper draws on data from my fieldwork undertaken in several OSS-related conferences across Europe, and 10 semi-structured interviews with FLOSS developers from the EU countries to show the dynamics of collaboration between public and private in the FLOSS innovation system. It seeks to answer the enquiry: How do firms collaborate with the community, and mobilise *social capital* (Lin *et al.*, 2001) emerged from this complex network to achieve their shared and/or divided goals? It raises some qualitative questions about the meanings and values of the collaboration between firms and the community in the FLOSS social world (Lin, 2004a). It will contribute to the mutual understandings between the private and the public sectors, namely firms and the community, and enhance the trust inter- and intra- users, developers and firms. The findings will also help practitioners and regulators to recognise, facilitate and utilise the new economy based on the hybrid innovation.

A Community of Open Source Practices (OSPs)

The history of the IT industry certainly is a wonderful vision of the evolution of an autocatalytic knowledge set, catalysed by the actions of a diverse set of agents, driven by a diverse set of motives, all resulting in an explosion of economic activity and an avalanche of creation and destruction. The heart of this process of creative destruction is the epistemic cycle of uncertainty, imagination and innovation.

(Jackson, Mandeville & Potts 2002: 329)

The FLOSS development works “at the organisational level, deciding includes issues of social justices, multiple interpretations, and adjudication of conflict across social boundaries” (Star *et al.* 2003: 242). These decision-making activities, usually cross-boundary, are based on a constellation of collective open source practice (for brevity, I refer to the open source practice as *the OSP*) shared amongst individuals, companies and other organisations in the FLOSS social world. This constellation of OSP, referring to

practices associated with both the FLOSS consumption and production, is based on my observation on innovation activities concerning the FLOSS development that are strongly based on the mentality of the interest in software problems and tackling them. Some common ones include sharing source code, trying out new releases or reporting bugs. The idea is reflected in some of the interviews:

The need to share information stems from the need to resolve problems caused by the multitude of proprietary systems and you need the code-breaking mentality to take on proprietary manufacturers.

(DY011202)

I think that sharing stuffs is not so important to me, but I think it's good to share software when you've written something important [...] I do it mostly because other people do it I have profit from that. So if everybody shares the code it's good for everybody. But other than that I don't see any personal gain of sharing a software.

(PC060209)

The FLOSS innovation system helps analyse the socio-technical dynamics of software innovation arising from the various cross-boundary activities. Allowing boundaries of diverse social groups to exist is crucial for sustaining the heterogeneity in the social world, which preserves diverse innovation resources, including social capital. However, boundaries should not stop members from travelling between groups or from communicating with each other. As a result, the soft boundaries encircling innovation networks in the FLOSS social world, on the one hand, allow each group to make their claims towards their collective practices and norms within the maintained demarcation, but on the other hand, the mobility of members and the fluidity of artefacts are prominent in order to keep the social world dynamic and energetic.

In this paper, I will analyse the innovation dynamics specifically of the mutual shaping of two different organisational cultures. While a contradiction seems to emerge from the encountering of the voluntary and the capitalistic cultures, I have observed some means through which some of the tensions between these two different ways of looking at knowledge and of organisation might be resolved. Through examining the collective practices of FLOSS developers residing in both the FLOSS community and OSS corporations and how the OSP is gradually institutionalised in the computer industry, this paper analysing the dynamics based on the socio-technical interactions will provide alternative perspective from ordinary forms of coordination built on costs and economic self-interest.

Working Practices in OSS Firms

The collaboration between the FLOSS community and OSS firms (see Bonaccorsi and Rossi, this issue), on the one hand, institutionalises the OSP, and on the other hand expands the FLOSS social network and brings more users into the social world. OSS firms operate as a combination of conventional companies bearing business strategies in mind and unconventional entities standardising the OSP. Having said that, the ubiquitous usage of ICTs and virtual communication in the FLOSS community remains the major channel for developers at OSS companies to communicate and to exchange innovation resources (social capitals, tools, ideas) within and outside companies. Albeit this attribute facilitates higher information flow than that in conventional firms, the corporate managerial yet demands developers within the system to do some bureaucratic works, such as attending regular meetings, dealing with paper work, or splitting iterative but necessary jobs such as debugging. Though these unavoidable routines are annoyed, however, working in an OSS company appears to be more liberal than in any other commercial sectors. Brian, an interviewee working in a leading OSS firm says,

When I do the routine work I don't always have to be at my desk. I get to decide when I get to do the things and also I have a lot of influences on what I do because we never get told 'you have to do this'. It's always we have these things that need to be done, who can do them? Then people can just volunteer. I mean, we always have plenty of things to do, so it's just the question of picking the one that is interesting to you, which is more or less the same thing you do when you work in your free time. So there's another freedom both in one I work and when I work home. But still then choices are of course made by my employer. I am not that just sponsor to sit and do whatever I want. There are some but not me.

(BO060203)

The innovation practices of developers at OSS companies follow the strategy which is a mixture of the conventional capitalist perspective on markets and customers, and the unconventional FLOSS innovation procedure. The innovation pattern in OSS firms therefore entails a hybrid one that pursues profits but is deeply linked to their interactions with the FLOSS community. In terms of the concern on profits, most interviewees have mentioned time pressure (strict deadline), influence of the investors/clients and available financial resources (loan) as crucial factors for firms to evaluate investment policies. Developers in this sense can no longer take their acts as "Just For Fun" (Torvalds *et al.* 2002). Unlike working in an informal innovation ambience where the shared interests are the main concern for them to work voluntarily on FLOSS projects, after joining a firm, FLOSS developers have to engage themselves in the operation of a smaller subgroup, working on specific projects, with certain colleagues. Regardless of that, many working contracts appear to be extended from developers' previous (informal) contacts in the FLOSS community. For instance, in the development of Ubuntu GNU/Linuxⁱⁱⁱ, the developers are brought together because they all work on the Debian project and know

each other well. Joining a firm for developers, therefore, signifies a formalised/institutionalised working partnership. In so doing, the trust and the tacit understanding between developers are enhanced. As many interviewees working in firms have said, if they have problems at hand, colleagues would be the first source they go after for resolutions, rather than through other means such as posting questions on newsgroups or discussion lists. This is probably because the shared goals are perceived better among colleagues working in the same team compared with the community. The institutional affiliation thus plays an important role in developers' daily programming practice. As Brian states:

It's very rare that I [post] question[s] on the Internet. Although I sign [up] for my local Linux community at home, usually I can find the answer to my question on the [firm's] internal mailing list.

(BO060206)

OSS companies thus often have close mailing lists for their employees. In setting up such an internal network of employed developers, the expertise in the firm has been centralised. The expertise of developers is concentrated in a bounded group and their interaction and relationship is fortified formally, in so doing to enhance the trust between colleagues and solve problems more efficiently. With such a strong developing team (expert-oriented), a firm is also able to convince their customers of the quality of their products and services.

It is also argued that clients and markets are the driving forces for firms to innovate. For instance, for firms distributing desktop systems, incorporating stable applications together for end-users is the main task. As Brian says,

[Un]like KDE developers just do it because they want to do it, we [company employees] have clients, the potential users out there. ... [T]he application probably needs to be able to work for a lot of different people. And obviously it must be pretty stable.

(BO0602)

As most users would not like to change their usage habit radically and tend to stay with existing interfaces, the key element to make a successful platform for users is to make them "feel at home". As Brian comments further,

It has been twenty years since we last saw a really new application. Everything else we do is just a little variation on what we did couple of years ago. So when you do word processor it has to feel like most of the word processors. If you do something it's very very differently, then you have to find something that really it's a much better way of doing it. Otherwise people would say 'no I can't figure

this out; I'll use something else'. That's the same for either graphical tool or command line tool. People have to feel at home. It is right [that] it's difficult to change from one operating system to the other because you don't feel at home suddenly. So you need a reason for going somewhere else.

(BO0602)

Although understanding the client's needs and communicating with them are vital for OSS companies to achieve in their marketplaces, if clients can understand the concept of open source, they would appreciate the products and services even more, and consequently fosters mutual-help between users (consumers) and developers (providers). Brian claims that it is important to let users understand the potential of open source software:

[We try to tell customers that] it's always better to have open source or free source because it gives [them] a power over the vendor that [they] can never have when it's close source. When it's close source then the vendor can lock [them] in as a user.

(BO0602)

The point that the relationship between clients and advisors would develop given their reciprocal understandings on the concept of open source is reflected on the conversation of James, a CEO in a FLOSS Small-Medium Enterprise (SME), about one of his clients:

He is a quite clever and open-minded. We understand [each other] perfectly and we know why we do that. He understands that it's better to share. In the case of ERP, he understands that proprietary ERP means you lose complete control on your information system and give to someone else; free software ERP means that you share many things with others that at least you can keep control on your information system. We both think the second case is more perfect.

(JP060202)

Where to find such a nice customer? Linux-related conferences or virtual platforms are good venues. Particularly at conference sites, developers or firms can spot their potential clients or co-workers easily, either through direct face-to-face interaction, or through snowballing contact. The social function of more informal conferences with less commercial atmosphere turns out to be even more useful for developers or firms to build up good relationships. In a relaxed environment, participants can establish informal personal contacts with people, among them some will become customers and the others will become colleagues. Paul, a founder of a OSS SME tells me, their customers are usually found through personal connections:

[Knowing people] is extremely important, especially for a small company. It's

very important that we know people in other companies who can give us projects or who can just give us contact to more important person in another company. That's essential because for small companies like us, it's very difficult to find entirely new customers. That's almost impossible.

(PC060205)

Conferences have become a conspicuous means for establishing social networking in the FLOSS innovation system. Social networking is seen to be crucial because with these contacts one is able to find whom s/he can turn to, whom s/he can cooperate with when problems come up. In addition, conferences act for a key mechanism within the FLOSS social world to reconcile various practices and assimilate diverse actors and organisations. In other words, OSS conferences and meetings secure social ventures for collaboration between firms and the community.

The Working Relationship Between the FLOSS Community and Firms

Hitherto, I have roughly described how OSS companies operate internally and their concerns on client-oriented issues. In the following, I will describe how OSS companies work with the FLOSS community in general.

The private sector is one of the stakeholders in the FLOSS social world and it co-exists with the FLOSS community. The mutually dependency between OSS firms and the community has been emphasised in many interviewees' conversation. Here are a couple of quotes from the interviewees based at OSS SME, which illustrate the celebrated community:

I think the company should go closer to the Linux community. We would like to keep some information confidential for business, but we should not forget the open source ideology. Because of open source, so there are we.

(SO060202)

The community is actually necessary. It is. Because that's the form to share interests and to communicate about to get information pass along, in a very broad sense.

(WP060202)

This interdependent relationship is endorsed and built on in a number of socio-technical mechanisms. Here, I will take discuss a number of them to explain the mutual-help model.

Bugs-reporting and Patches-contributing

Because software in a form of language has never been perfect and often altering when applied to different hardware systems or infrastructure, constant restoration and maintain are crucial to keep it in a feasible shape. Bugs-report, feature-request, patches-contributing and related mechanisms have emerged as practical solutions in response to this innate software problem. Since the FLOSS innovation system is more accessible than proprietary ones, actors can download newly-developed programmes/software, try out, spot bugs and report them back to authors or maintainers, or even write patches for them. These patches will be incorporated into new versions of software, if they are applicable. This mechanism of bugs-reporting and patches-contributing has evolved as a mutual-help culture in the FLOSS community. There is tendency that authors will make an announcement of the release of her/his *experimental* projects and invite downloading and feedbacks (i.e. bugs-reports or patches). I have seen a Debian developer using an unstable web browser *Galeon* which hangs often when opening up flash links. He explains to me that he uses this programme to report bugs. In so doing, he can help improve this imperfect programme and encourage innovation in web browser technology. Albeit this mutual-help pattern is helpful for developing FLOSS, it does contain some risks in innovation processes, peculiarly because of the informal working relationships. If volunteers get detracted in their lives, or the shared goals of the project diverge, their contributions will likely terminate.

It is estimated that around three billion dollars worth of hours have gone into this constant upgrade process for the heart of the Linux operating system. And this figure doesn't even take into account the millions of hours that have gone into the thousands of applications that run on Linux (Cancilla 2003). Working closely with the community thus can save firms time and cost in detecting bugs and writing patches (though they still have to do this as routines to ensure qualities of their products). Unlike proprietary software firms that deal with bugs and patches in close environments where only insiders get involved, OSS firms do welcome and rely heavily on bugs-reports from the community. As Paul says,

[The] Linux community makes enormous contributions to the formal of Linux distribution for example software libraries that we can integrate into our customer software. And that respect is very important for us. These libraries are very helpful on Linux distributions.

(PC060208)

However, OSS firms formalise this bug-reporting and patching approach in their practice while keeping channels to the innovation system open. It is even more standardised for firms dealing with patches with a series of guidelines, which simplify and standardise the

procedure of contributing patches. In so doing, it eliminates the risk of potential discontinuity owing to volunteer drop-outs or incompatible submissions and keeps software products in a consistent condition. Nevertheless, this also implies that prospect contributors have to be wary of the rules to be included. It is a process of *technological socialisation*.

Social Networking

The interdependent relationship between OSS firms and the FLOSS community is something that makes FLOSS innovation pattern different from other companies that deliver proprietary software. The fundamental software engineering techniques are not particularly distinctive in the FLOSS innovation. But as an interviewee states:

It's different in the way that [in the community] you certainly have people starting to contribute to the project that you never ask to do it, and that's interesting because they have the probability of giving you ideas you haven't thought of.

(BO060208)

In this sense, social capitals created and preserved in the FLOSS development appears to afford more innovation opportunities. It is recognised that brainstorming and experimental practice in the community (mostly taken place on the Web) stimulate and cultivate the FLOSS development. The social function of the community also highlighted in James' conversation:

[From] the social point of view on free software development, writing creative scripts and sharing them is basic. You get many friends when you do free software. And the relation is not just sharing things on the Internet but making friends with the whole communities of people.

(JP060204)

Identity Building

Apart from social networking, another social function of the community is to provide *socio-technical identity*. While employees working for firms get recognised by their companies' names, in the community, people identify each other by asking which projects one is working on. If they work on the joint projects, the familiarity is therefore strongly shared with the collective interests and common practices. The projects (the artefacts) symbolise one's identity and becomes metaphors in the community. If one is working for a specific prominent project, one is easier identified. If one contributes to more projects, s/he gets more credits and a stronger identity showing her/his level of expertise. When one is more famous, it is more likely for her/him to get employed by a firm, especially a big firm. Therefore, earning credits and gaining an identity become one of the incentives

for developers to participate in the community. This tendency of such a reputation game is portrayed in Eric Raymond's writing "The Cathedral and the Bazaar", and is also delineated in the FLOSS survey of developers in 2002 (FLOSS report 2002). As a security consultant working at Informatique CDC in France summarises his motivation to write free software as below, the reputation game and the support from the community are both well shown on his list.

- a- build something I need for my current projects, or tech watch
- b- be "fair" and share back, to thanks the community for past help... make the system work... (If I dont reward helps, soon there will be no help)
- c- get support from a community
- d- (to be honnest) get some "reconnaissance" (in french), ie just be known as competent and helpful 8)

(AC21202)

Licensing Scheme

Given the flourishing OSS business model, some software firms nowadays take adopting OSS licences as a business strategy. Various OSS licences are invented to meet individual organisations' commercial or technical requirements but still comply with the collective OSP. Unlike the purpose of General Public License (GPL), which is designed to engage users in the free software development (Lin, 2004b), many other OSS licences are designed to lessen the social constraint. In focusing mainly on the OSP, these licences are meant to be as flexible as possible to allow further manipulation.

Different licensing strategies embed the different meanings assigned to the idea of morality. For a CTO at an OSS SME developing portable operating systems, the licensing strategies they take and the legal artefacts they employ are based on practicality. He reasons the firm's policy behind the decisions of releasing the new version of software under OSS licences. He gives two main reasons. The first is the commercial:

I think [our product] is quite difficult now from the commercial point of view to compete as it is. It makes no sense that customers must pay quite serious money to get source code and they by no means may circulate it. They could just put examples on the web site, saying "here is our stuff, try." Well, they could put their own stuff but they wouldn't put that 'Here is the modified version of the driver'. In fact, if they have put it, I wouldn't worry too much about it. But the licence said that they could only send to other people when they have licence. So it was constrained. I think it's quite hard to compete, now, generally speaking, if the source code is not made available. So that's from the commercial aspect.

(CF060803)

The second is the market share to get as many users as possible.

Another one has to do with you've brought as many people as possible if you like. Well, you find that one of the reasons that those become popular even more popular than some other free software such as FreeBSD, is because universities start using it. There's also this counter culture I suppose to which might attract to universities or university students to use at the first place. But I think it's important in terms of having as many users as possible. You can do that by intermediary for such as experts done within years licences which said, "You could have this [new product] for research, education, and news." But you can't develop politics with it. Perhaps that will do as well. But I think what we got a decision to make certain things possible and I don't have very good reason to regret it. I think a right thing to do.

(CF060803)

While his morality is rather practical, the perspective of attracting more users de facto resembles one of the functions that many OSS licences provide. They serve as a useful mechanism to engage actors in the FLOSS innovation. In other words, they are invented and adopted mainly for encircling as many as actors as possible in their social networks to mobilise innovations, albeit some of them have put restrictions on appliance with proprietary software (e.g. GPL). Licensing thus serves as an important apparatus to bolster the collaboration between the FLOSS community and OSS firms.

Apart from the cooperative relationships listed above, there are many others that worth of exploring such as the *mutual-learning* between the public and the private sectors. The mutual-learning activity, which can not be measured in monetary terms, is attractive to OSS firms because of the rich *tacit knowledge* embedded in the doings. For instance, admitted his contributions to the community almost acquire no direct financial gain, a firm-based OSS developers Paul says,

It's just experience, something you cannot get from working with Windows or Microsoft NT servers. The most important thing is I think knowledge that comes from personal experience with other products. So you can see what you can do better than others.

(PC060206)

Tensions in the Hybrid Collaboration

While this type of hybrid innovation is more and more prominent in the FLOSS development, the collaboration process is never smooth and straight-forward. In the

following section, I will focus on different expectations of OSS firms and the FLOSS community on various issues to understand why tensions sometimes arise between these two counterparts and how are they managed in the collaboration process.

Normally, firms receive manpower and ideas from the community, and the members of the community get economic support from the firms, especially for big projects. The need of financial support is expressed in a firm-based OSS developers' words:

Open Source software is mostly developed by nonprofit organizations or individuals, there is usually no official support or guarantee provided. But this may not be a problem. Often, commercial support falls short or is too expensive.
(AS010401)

However, the economic reason might also endanger the existence of the community. Sometimes when the firms are dealing with business, complying with the OSP no longer fits into their agenda. The financial incentive will possibly force firms to move away from conducting the OSP and lock up their source code or concede with the proprietary software. Though proprietary software firms may not be moral enough to release full source codes, not all open-source developers would agree on a full-scale open source policy either. As William, a CTO at an OSS SEM says,

When you talk about open source, not every software can be open source. It would be Who will pay these people doing something very special? These people who do something very special need to communicate with their users and with people doing competitive stuffs. They need to exchange their ideas. But that doesn't necessary mean this thing has to be open source, why should it?
(WP060202)

When being asked does his conversation suggest that he does not support FLOSS, William says,

It's an interesting idea, and works well for some projects, doesn't work for other projects. And it doesn't work well for software company which would have a single product.
(WP060203)

While morality is continuously defined and redefined differently in the FLOSS social world, how to manage these different and sometimes contradicting voices with the FLOSS social world is the key to maintaining and sustaining the interdependent relationship within the community itself and also between the community and the commercial sector.

The Hybrid Identity of OSS Developers

As seen above, developers in OSS companies mostly remain working closely with the community, or deploy/employ the tools developed in the community. As the manoeuvre in the community is different from the one in the commercial sector, developers actually play double roles in their daily practices, which would give them hybrid identity in the FLOSS social world. Their practices at some time have to qualify the standard of the commercial sector for making profits and fulfil the requirements of the customers, and at the other time their contributions made to the community projects are obligated (morally, socially and technically). Having the hybrid identity in fact gives the developers flexibility to play around. They can have resources both from the commercial sector (e.g. institutional support and monetary capital) and from the community (e.g. social capitals and technical support). They can use the community as a ground to test out their new idea and make some experiments, but later on release their work formally on the market for further incorporation with other applications.

For example, a programmer at a German-based OSS SME, who mainly develops software for networking and system administration, still commits his spare time to writing scripts of detecting software vulnerability. The scripts he was writing later on would be applied to finding out security holes of the systems he develops for clients. The script did not work out fully as the way he expected, but after releasing it on the Web, a few bug reports directed him to modify the scripts and makes it better. The script now can detect security holes more efficiently. If he did not write this script to try out his idea, he would never find out “if certain things are possible or not” (PC060210). The experiment he has done after sharing with the community benefits from the feedback of other members in the community, and that turned out to be useful for developing commercial software. This experimental manner works well to both the community and firms, if no other pressures apply.

The private sector and the community however, represent two different manoeuvres and approaches in producing FLOSS. Developers have different motivations for residing in a hybrid world. Some firm-based OSS developers have shown their different motivations in working at firms. As Brian mentions,

In the corporate world you would need the mindset that I can make money on this because someone else needs it. That’s been driving a lot of applications, a lot of good software. It still does. I mean that’s the way we distributions compete against each other and write added value to the basic Linux part. It does not work on interested people or motivated people on personal level, but does attract some software developers.

(BO060214)

To support this view, the interviewee also explains his feeling on working with commercial companies:

It's a good feeling to have a project out and have people using it. That is something that drives a lot of the people, including me, doing user level application at least. Then it changes from being interested in solving a problem, to being able to give people something that people find useful, that is something I know that a lot of people find interesting and need it.

(BO060215)

Another developer explains his motivation to establish a commercial company as demonstrating his ability to have “the freedom of innovation” (JP060203). The incentive to have his own company is to be economically independent. Having a company he will not be exploited by big firms, and this also prevents his creativity being choked off from big firms' policy.

[Big firms] don't want to create new products for new markets. And I know many people they decided to put down a great job as an engineer in a great large corporation to create their own small company doing free software because they are tired of having ideas, which they could never put into practice and share with others. And so that's one of the reasons also why I am doing my company and the ERP, because if I have an idea, then I can write it as a software then it becomes quickly a product which I can share with others. And even in that new idea can be dangerous for a very old product, I can still do it. So it's like I have the freedom to innovate, put my innovation on the market, and share it with others. And that freedom does not exist in many large companies. Because there are so much innovation actually in free software, and there are so many people who go that way because that's probably the only way whenever you have an idea to see in practice used by many people and to improve it after it has been used.

(JP060203)

While some developers recognise the role of companies in the FLOSS development, other members in the community however hold a sceptical view. For instance, Stallman criticises the commercialisation in light of the development of GNU EMACS that

I don't think that anything like EMACS could have been developed commercially. Businesses have the wrong attitudes. The primary axiom of the commercial world toward users is that they are incompetent, and that if they have any control over their system they will mess it up. The primary goal is to give them nothing specific to complain about, not to give them a means of helping themselves. ... The secondary goal is to give managers power over users, because it's the managers who decide which system to buy, not the users. If a corporate

editor has any means for extensibility, they will probably let your manager decide things for you and give you no control at all. For both of these reasons, a company would never have designed an editor with which users could experiment as MIT users did, and they would not have been able to build on the results of the experiments to produce an EMACS.

(Stallman, The EMACS Full-Screen Editor)

Though firm-based innovation appears to provide a standardised and stabilised innovation practice, the community innovation de facto affords more dynamics than the firm-based innovation. As Stallman remarks,

[T]he standard EMACS command language was the result of years of experimentation by many user-maintainers on their own editors ... On the fateful day when I gave users the power to redefine their own editor, I didn't know that it would lead to an earthshaking new editor.

(Stallman, The EMACS Full-Screen Editor)

Giving Stallman's words a second thought, instead of speaking against OSS-oriented business, he actually highlights the non-replaced role of the FLOSS community. How to maintain and sustain the complex co-existence of the FLOSS community and firms remain an unanswered question.

The Heterogeneous FLOSS Social World

As open source software develops, we should expect that it becomes more and more like other dynamic, knowledge intensive industries. In that sense, the dynamics of software development are likely to rely on parallel processes of commercialisation and science [e.g. the early basic scientific and commercial uses of genetic engineering]. They will rely on both the overall production of public knowledge as well as on the closing off of parts of knowledge production within the firm in order to capture economic value.

(McKelvey 2001: 34)

Through out the paper, I have discussed the interactions between the FLOSS community and OSS firms, two of the most distinct fields in the wider FLOSS social world. The innovations in these two domains share the OSP. The hybrid innovation of FLOSS demonstrates the advantage of acquiring resources both from the community and the commercial sector. The community offers space for experimental projects and informal communications, while the commercial sector stabilises and standardises the development of these community projects by incorporating them together and putting into markets. Connecting the public and private sectors, the collective OSP have afforded diverse

actors to interpret and render the concept flexibly in various means (e.g. licences) and formed a FLOSS social world.

In this heterogeneous environment, the hybrid innovation also embodies hybrid cultures (or possibly multiple) in the FLOSS social world. Unlike a lot of open source avant-garde who has publicly claimed to be hackers, the identity qua hacker in the mundane FLOSS social world seems to be implicit. Some developers tended to consider themselves as pure programmers while some of them still regard themselves as hackers. The well-celebrated hacker culture in the 70s and 80s has become a branch of various self-expressed cultures. A pragmatist atmosphere has replaced with the hacker culture that is said to uphold the FLOSS development. As Brian says,

I don't like to word 'hacker', because the word has a bad opinion about it. And I am not one of those every time someone says hacker and really mean a cracker, then you say 'no, no, no...'. I don't care. It's just a word describing the hacker culture. And if you take the right meaning of the hacker, and yes, I am a hacker. But I would never use that word to a journalist who wouldn't get it. I would never do that because there is no reason to be religious about that stupid word. It's just a word. ... If I'd like to use a word to describe myself, I would say developer. I like writing software. And I like helping other people to do it, too, which is my motivation for going into management. Then it wouldn't be me writing the software but I would still be developing it by helping people. I am definitely a developer.

(BO0602)

Brian's words reflects the practical view prevailing in this type of hybrid innovation system -- because the most important thing is what you have done, rather than what you are. The essentiality of practices again is confirmed. Having said that, it does not mean that hacker culture has no longer been influential. The voices of good will hackers have never been silent in the FLOSS community. The pragmatist perspective however provides a pluralist domain to contain diverse actors. It is also the open and tolerant social structure of the FLOSS social world, where *Us* and *Them* can live together and cooperate, where *Others* have the right to express their views, that provide the highest affordance to the technological innovation. The FLOSS innovation takes place within such a heterogeneous field, where the local and tacit knowledge can be preserved and borrowed as innovation resources. It is also heterogeneity in the social world that affords the hybrid innovation coalescing the innovation resources of the firms and the community.

For future research, this paper has not yet dealt with different types of hybrid collaboration (Rossi 2003) considering various sizes and attributes of OSS firms, let alone cover all topics in the hybrid innovation in the FLOSS development. It is

particularly pivotal to understand whether the involvement of corporations in the community-based FLOSS projects influences (negatively or positively) the motivations and number of voluntary participants, and whether it matters on the degree of the corporate involvement. Further studies on hybrid innovation, both quantitative and qualitative, are urgently required in order to map out different types of OSS-based innovation patterns and the socio-technical dynamics in the FLOSS innovation system.

References

- Cancilla, J. 2003. Open Source Software for Windows. TechSoup.org.
<http://www.techsoup.org/howto/articlepage.cfm?ArticleId=523&topicid=2>
- Jackson, M. & Mandeville, T. & Potts, J. (2002) The evolution of the digital computation industry. *Prometheus*, Vol. 20 No. 4, p. 323-336.
- Lave, J. & Wenger, E. 1991. *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press.
- Lin, N. & Cook, K. & Burt, R. S. (Eds.) 2001. *Social Capital: Theory and Research*. New York: Aldine de Gruyter.
- Lin, Y. 2004a. *Hacking Practices and Software Development: A Analysis of ICT Innovation and the Role of Open Source Software*. Unpublished doctoral thesis. Department of Sociology, University of York, UK.
- Lin, Y. 2004b. 'Epistemologically Multiple Actor-Centered Systems: or, EMACS at work!'. *Ubiquity*, Volume 5, Issue 1, February 25 - March 2, 2004. URL: http://www.acm.org/ubiquity/views/v5il_lin.html
- McKelvey, M. 2001. "Internet Entrepreneurship: Linux and the dynamics of open source software", CRIC Discussion Paper no. 44. Centre for Research on Innovation and Competition, The University of Manchester & UMIST.
- Rossi, C. 2003. *The Economics of Open Source Software: Incentives, Coordination and Diffusion*. PhD thesis. SSSUP, Italy.
- Stallman, R. "The Emacs Full-Screen Editor". URL (consulted 27 November 2003): <http://www.lysator.liu.se/history/garb/txt/87-1-emacs.txt>
- Star, S. L., Bowker, G. C., Neumann L. J. 2003. "Transparency beyond the Individual Level of Scale: Convergence between Information Artifacts and Communities of Practice", in the book *Digital library use: social practice in design and evaluation*, edited by Ann Peterson Bishop, Nancy A. Van House, and Barbara P. Battenfield. The MIT press.
- Torvalds, L. & Diamond, D. 2002. *Just For Fun: The Story of an Accidental Revolutionary*. Texere Publishing.

-
- i In this paper, the term 'community' refers to the sense of 'a community of practice' which is parallel to the meaning of 'a social world' (see more discussion in the body text).
 - ii In this paper, 'OSS firms' are defined as those who develop, maintain/support or distribute FLOSS-based products or services, regardless of sizes, although it is possible that OSS SMEs might have different approaches towards collaborating with the FLOSS community compared with multinational companies.
 - iii <http://www.ubuntu.com>