The Anatomy of Sustainable Open Source Community Building

The Cultural Point of View

Teemu Mikkonen & Tere Vadén

University of Tampere

{teemu.mikkonen & tere.vaden}@uta.fi

Abstract. Open source software (OSS) communities are too various to permit a one-size-fits-all solution to sustainability and successful development. In particular, the hacker ethic of volunteer communities and the salary-based work ethic of company-driven communities do not mix easily, and have very different conditions of sustainability. Based on earlier work on OSS community typology, we characterize three new cases of community (Wringer, IT Mill Toolkit, NoTA). These three communities are just being launched, and the typology suggests some possible bottlenecks of socio-cultural sustainability that the building communities may face.

1. Introduction

Based on OSCOMM (Building Open Source Communities) -project, the purpose of which is to develop guidelines on how to start and run sustainable open source projects, we discuss the issue of sustainability especially from the point of view of cultural and social sustainability. The goal of OSCOMM is to gain understanding of open source community building through a number of case studies. Here we consider three of those:

- NoTA (Network on Terminal Architecture) - a modular service-based architecture framework for embedded devices. First developed by Nokia and released as open source in 2008.

- IT Mill Toolkit - an open-source framework, providing widgets and tools for the development of Rich Internet Applications (RIAs). Originally developed by the company IT Mill. Released as open source in late 2007.
- Wringer - A User Interface scripting engine for embedded devices. Originally developed by Sesca Embedded Solutions. Will be released under a LGPL license in 2009.

Each of these communities has different starting points and different intentions for the future. In this work we are going to present the differences between the communities and evaluate the nature of these communities on the basis of an open source community typology developed earlier (Mikkonen et al. 2006) Our plan is to make a community typology which illustrates the differences and similarities with the older communities.

2. The Community Typology

Many if not all of the contemporary open source software communities are by nature hybrid, consisting of actors with both commercial and non-commercial interests, motivations and backgrounds (see e.g. Mikkonen et al. 2006 & 2007). The goals of different groups of collaborators - hobbyists, volunteers or paid workers - diverge while there is also considerable convergence with regard to the technical goals of a project.

Company participation and the work ethic it implies in communities present both dangers and opportunities for long-term sustainability. A company needs to be able to identify the systematic variation in motivation, values, ideology and practices between different communities. Consequently, a general framework for assessing the sustainability and conditions of success of a community of open collaboration will be useful in generating a strategy for interaction between companies and volunteer groups. To understand this strategy is important especially in gaining understanding of open source community building.

In the survey reported in (Mikkonen et al. 2006), two distinct types of community ideology and work ethic could be identified. What we call *the hacker ideology* is the traditional FOSS work ethic of freedom, fun and sharing of information, while the opposing ideology is the traditional, *salary-based work ethic*. These two types of ideologies correspond to certain kind of structures of power and authority. Therefore by “volunteer community” we mean those communities where the hacker work ethic is dominant, and by “company-based communities” we mean the communities where companies and business objectives have more importance and a large percentage of developers are paid for their contribution.

More detailed analysis and a typology of communities can be created by combining the voluntary/company axis with some other variables. In the following, three elements are investigated in tandem with the voluntary/company axis: the size of the community, how centralized/de-centralized communication and decision-making in the community are and the strength of the chosen license.

1. Size of the community. We assume that a larger community is always more efficient and sustainable but potentially increases problem complexity for company participation. The size of the community must also reach a certain minimum size in order to facilitate the open source effect.
2. Communication and decision-making structures of the community. Different systems of governance exist in free/open source software communities, including democracy, meritocracy and dictatorship. Here we look at how centralized communication is. This tells something about the governance structure, hierarchy and bottlenecks.
3. License. The type of free/open source software license chosen by the community potentially affects who will participate in the community. We classify licenses based on how strong copyleft effect they have. GNU General Public License, for example, is a strong copyleft license, while Eclipse Public License gives more freedom, and licenses like the BSD license are not copyleft at all.

When we combine these three elements with the volunteer/company axis, differences between communities can be identified as can be seen in table 1 (with examples).

Table 1. Community typology (Vainio et al. 2006)

Size / Hybridity	*Volunteer*	*Mixed*	*Company*
Small	Wordpress		MySQL, Laika
Medium	OpenBSD	Mozilla	OpenSolaris
Large	Debian	Linux (kernel), GNOME	Eclipse

Decision-making / Hybridity	*Volunteer*	*Mixed*	*Company*
Decentralized	Debian		Eclipse
Balanced		Linux (kernel)	
Centralized	GNU	Mozilla	MySQL

License / Hybridity	*Volunteer*	*Mixed*	*Company*
Non-copyleft	OpenBSD	Apache	
Weak copyleft		Mozilla	Eclipse, OpenSolaris, Darwin
Strong copyleft	GNU	Linux (kernel), GNOME	MySQL

In the classification above, we can see both differences and similarities between communities. Based on this analysis, some “ideal types” can be identified which characterize some of the most prominent differences between communities. Four ideal types can be identified:

1. Centralized, company-driven, small community (e.g. MySQL)
2. Large community, several companies, business work ethics (e.g. Eclipse)
3. Large community, several companies, hacker background (e.g. Linux kernel)
4. Volunteer, decentralized, large (e.g. Debian)

All of these ideal types have proven to be sustainable and successful over time. However, the four have different grounds for socio-cultural sustainability. For example, for the Debian community the strong volunteer ethos demands a strong copyleft license, which would not work for the Eclipse community. Correspondingly, introducing salary-based work ethics into the Debian community has proven to be hard, and potentially disruptive (witness, e.g., the "foot-dragging" caused by the dunc-tanc-discussion, see also Shuttleworth 2006). On the other end of the scale, the recent purchase of MySQL (the company) by Sun has created challenges for the sustainability of the community; it is hard to imagine any company purchase that would have a drastic effect on Linux kernel or Debian community.

3. How do the new communities fit in the typology?

The three case communities - NoTA, IT Mill Tookit and Wringer - are not in the same “phase” as the communities described above, all of which can be called mature communities. The three case communities are not yet complete communities. Not all of the relevant software has been released as open source, and the developer (and user) communities around the released applications are being built as we speak.

Table 2. The Nature of the Intended Community (Sirkkala et al.2009)

Software	**Community type**	**Core group**	**Intended future**
NoTA	Company based	Closed	Ecosystem of related communities
IT Mill Toolkit	Mixed	Closed	Independent Community
Wringer	Volunteer	Open	Closely connected with GTK+ bindings community

We asked the following questions from the persons responsible for the OSCOMM participation in the communities:

1. How many developers should there ideally be in the community?
2. What kind of structure is best for the community (e.g. "a benevolent dictatorship/meritocracy/democracy)?
3. What license could be the best one (e.g. LGPL)?
4. Should most of the developers in the community be volunteer or salary-based (work in some company which in some way benefits from the community)?
5. How intensive contacts should the community have with company partners?
6. How many company partners there should be to support the community (0, some, as many as possible)?

It is important to notice that these questions are more about the future intentions of the possible communities. The circumstances can change along the way and the goals may be different later. Anyway, it is possible to see certain analogies between the older communities and the intended or ideal case communities. Table 3. describes the three case communities in the typology discussed above:

Table 3. Community typology of the new communities

Size / Hybridity	*Volunteer*	*Mixed*	*Company*
Small			
Medium	Wringer		
Large		IT Mill Toolkit	NoTA

Decision-making / Hybridity	*Volunteer*	*Mixed*	*Company*
Decentralized			
Balanced			
Centralized	Wringer	IT Mil Toolkit	NoTA

License / Hybridity	*Volunteer*	*Mixed*	*Company*
Non-copyleft		IT Mill Toolkit	
Weak copyleft			
Strong copyleft	Wringer		NoTA

Like in the classification of the older open source communities, differences between the three communities can be identified.

1. Voluntary developers with centralized social structure and medium sized community. Intention to get more developers involved. Strong copyleft license. (Wringer)
2. Voluntary and salary-based developers (mixed) with “dictator” (IT Mill - company). Large community with many end-users. Intention to get profit and more contacts (customers) with companies. Non-copyleft license. (IT Mill Toolkit)
3. Company-driven ecosystem with centralized social structure. Intention to get communities to make applications on NoTA platform. Strong copyleft license. (NoTA)

With this framework it is possible to identify strengths and bottlenecks of socio-cultural sustainability described in Vainio et al. (2006).

In the case of *Wringer* the strengths could be the voluntary base of the medium -to-large community. Also the suitable license (LGPL) is an advantage to persuade new developers to commit to developing. However, if decision-making is centralized (i.e., company driven) the developers will have to have a strong internal motivation for collaboration. Possible bottlenecks could be the same identified in Debian. If the hacker-ethics clash with the salary-based ethics, there can be legal and social problems with the developers and their work.

With regard to *IT Mill Toolkit* community there are potentially different problems. The non-copyleft license permits certain kinds of business models (e.g., dual licensing), but at the same time makes building a large community harder. The centralized and dual-licensed MySQL community has always been relatively small, compared, e.g., to the GNOME and Eclipse communities. This kind of mixed and centralized structure can be problematic if the voluntary developers don’t agree with the company how to use and adapt the efforts they have given to community.

NoTA is crucially different from the others. Its openness is in the idea to offer a framework for developers (or companies) to develop their own software applications. The software framework, NoTA itself, is open also, but it is highly centralized because it must be stable enough to support new devices. The aim of NoTA is not to gather more developers for the framework, but to get users for the framework as a terminal architecture. This is why it is not easy to compare it to the communities mentioned before. However, the strengths and bottlenecks in NoTA as an ecosystem could be same kind as in the Eclipse community, which has also risen on the basis of a company driven effort (IBM) (see Mikkonen et al. 2007 & Vainio et al. 2006). Here it will be crucial to engage other companies to the framework.

4. Summary

Different types of OSS communities have different strategies of survival and success. By looking at the characteristics of long-lasting and fruitful OSS communities we can identify some of the prerequisites of sustainability. Even though each community is unique, some ideal types may be identified, and the building of new communities may take aim at these ideals. Based on a typology developed earlier, we suggest that the growth of the community (recruiting new volunteer developers) will be a bottleneck for the sustainability of the future Wringer community, while the IT Mill Toolkit community will, in addition, face problems of decision-making. In the case of NoTA, the challenge will be more on the side of companies, not so much individual developers.

References

Mikkonen, T., Vainio N. & Vadén T. (2006). Survey on four OSS communities: description, analysis and typology. Empirical Insights on Open Source Business. Helander N. & Mäntymäki, M. (eds.) Tampere: Tampere University of Technology and University of Tampere.

Mikkonen T., Vadén T., Vainio N.. (2007). The Protestant ethic strikes back: Open source developers and the ethic of capitalism. First Monday 12 (2), 1-12

Shuttleworth, M. (2006). Funding free software projects. <http://www.markshuttleworth.com/archives/4>, 06.04.2009

Sirkkala, P., Aaltonen, T. & Hammouda, I. (2009). Opening Industrial Software: Planting an Onion.

Vainio, N., Vadén, T., Oksanen, V., Seppänen, M. (2006). Elements of Open Source Community Sustainability. Helander N., Antikainen M. (eds.) Essays on OSS Practices and Sustainability. Tampere. (EBRC Research Reports 36)