Virtual communities as a resource for the development of OSS projects: the case of Linux ports to embedded processors

S.L. Toral[a]*, M.R. Martínez-Torres[b] and F.J. Barrero[a]

[a]*Departamento de Ingeniería Electrónica, Universidad de Sevilla, Sevilla, Spain;* [b]*Departamento de Administración de Empresas y Comercialización e Investigación de Mercados (Marketing), Universidad de Sevilla, Sevilla, Spain*

Open source software (OSS) projects represent a new paradigm of software creation and development based on hundreds or even thousands of developers and users organised in the form of a virtual community. The success of an OSS project is closely linked to the successful organisation and development of the virtual community of support. The main objective of this article is to analyse the activity of virtual communities. Social network analysis is employed to analyse Linux ports to embedded processors as a case study to achieve this aim. The obtained results confirm the necessity of structuring the virtual community with a selection of active developers and core members to promote community activity and attract peripheral users, expanding the impact of the underlying software. The obtained result will be useful for the software industry migrating to the open source software paradigm.

Keywords: open source software; virtual communities; social network analysis; regression analysis

1. Introduction

Open source software (OSS) is software whose source code is available to users and can be distributed by third parties with few limitations on modifications and distributions. This term has been exactly defined by the Open Source Initiative (1999) using 10 key requirements. In particular, OSS projects are developed and released under some sort of 'open source' licence that allows inspection and reuse of the software's source code (Crowston and Scozzi 2002). Each OSS project is supported by a few, dozens or even hundreds of geographically distributed developers, organised as an Internet-based community, who voluntarily collaborate to develop the underlying software. The success of OSS projects has been attributed to their speed of development and the reliability, portability and scalability of the resulting software (Hallen *et al*. 1999, Trung *et al*. 2005). These claimed advantages of OSS development are due to the fact that the source code is open to the Internet community. As everybody can access and review anybody else's work, developers can learn from each other and improve their overall software development skill (Lussier 2004). Also, instead of using huge financial resources to put the software through extensive testing and quality assurance, like a proprietary vendor will do, the open source projects have the community as a resource (Lakhani and Hippel 2003, Gruber and Henkel 2006). Finally, several studies claim that OSS is developed faster, cheaper, and the resulting systems are more reliable than the proprietary software (Wu and Lin 2001, Mockus *et al*. 2002).

The literature on OSS has been focused on several topics related to the successful OSS development (Perkings 1999, Trung *et al*. 2005, Henkel 2006), motivation of programmers and developers (Bonaccorsi and Rossi 2003, Hertel *et al*. 2003, Wu and Tsang, 2008), the benefits of OSS (Kogut and Metiu 2001, Spinellis and Szyperski 2004, Spinellis 2006) and its implications for the public sector (Applewhite 2003), or public domain licensing (Valimaki and Oksanen 2005, Gambardella and May 2006). This article will be focused on successful OSS development, but instead of analysing the development in terms of the source code produced, we will focus on the social relationships among virtual community members. Virtual communities have become an important new organisational form and yet relatively little is known about the conditions that lead to their success. The idea of OSS communities organised in a certain structure in which their members perform different roles according to their degree of involvement has been reported in the literature (Mockus *et al*. 2002, Wasko *et al*. 2009). This study goes beyond this idea defining to what extent the structure and the different members' roles have an incidence on the successful development of the OSS community. Social network analysis (SNA) techniques will be used for this purpose. SNA models

*Corresponding author. Email: toral@esi.us.es

the community as a group of participants considering the links among them (Wasserman and Faust 1994, Nooy *et al*. 2005). From this model, several indicators measuring some features of the community can be derived. A case study based on the Linux ports to different processors will be used to measure the proposed indicators and to test their influence on the development of the community. The obtained results could be applied to design new OSS projects or to improve an existing community.

The rest of the article is organised as follows. The next section analyses the role of virtual communities in the development of OSS projects using the notion of communities of practice (CoP) as the theoretical background. After that, the methodology based on SNA techniques is presented. Then the case study is introduced and the proposed methodology is applied. Finally, the conclusions are detailed.

2. Virtual communities in OSS projects

OSS communities have been referred to as virtual communities (Crowston and Scozzi 2002). However, open source research is often viewed as its own track and is not frequently associated with the study of virtual communities. This may be due to the fact that much of this open source research addresses software/technical/work process issues (Crowston *et al*. 2005) and not social/community issues.

A virtual community-based approach is an effective way of sharing knowledge that facilitates informal sharing of knowledge available from experienced and skilled people (Lee *et al*. 2003). Wikis and weblogs can be considered also as virtual communities supporting interaction and collaboration by people geographically distributed around the world. However, they exhibit notable differences. For instance, OSS communities make an intense use of forums and mailing lists, engaging users in a question-and-answer process, whereas weblogs encourage first-person storytelling and commenting, and wikis are focused on incremental knowledge exchange (Lin 2008). Although all of them are based on open content creation, OSS communities exhibit a different governance structure and require a certain level of expertise (Wagner and Prasarnphanich 2007).

A typical OSS project web site provides forums and mailing lists where participants and contributors can report software improvements, needs or bugs, and share and discuss solutions to posted messages. For new members, bug fixes or problems exposition is usually the way to start contributing. If somebody has found a problem, it becomes accessible to the whole community. As members of the community browse through the OSS project tools, it is very likely that someone will consider the problem, and will jump right into action providing solutions or alternatives using his or her concrete experience. One of the most interesting features of forums and mailing lists consists of enabling re-experience by collective reflection and virtual experimentation (Hemetsberger and Reinhardt 2006, Stefanone and Gay 2008). The notion of re-experience must be understood in the sense that meaning is negotiated among community members leading to new knowledge. As a difference to bug reporting databases or Concurrent Version Systems, both forums and mailing lists have the advantage of showing explicitly the sequence of discussion as they allow the possibility of being organised through threads of discussion. Threads are groups of messages sharing the same subject. A thread is initiated by someone who posts a message asking for help, suggesting some improvements, or just considering some new idea. Then people start answering this initial message, posting possible solutions, sources of information or just extending posted considerations. Some members of the community become engaged in a process of conceptualisation, leading to some collective innovation and new knowledge. The result is a list of related messages where the sequence of reflections is detailed, so newcomers can follow expert reasoning step by step.

2.1. Communities of practice

Virtual communities can be studied from the perspective of CoP developed by Lave and Wenger (1991). This concept refers to the process of social learning that occurs when people who have a common interest in some subject or problem collaborate over an extended period to share ideas, find solutions and build innovations. The basic assumption underlying the theory of CoPs is that engagement in social practice is the fundamental process by which we learn (Wenger 1998). Learning in a community should be viewed as an integral constituent of participation in the CoP, and as a process of constructing knowledge through social interaction with other members of the community, of changing relationships with other members of the community and of transforming roles and establishing identities from a journeyman to a master in the community (Yunwen and Kishida 2003, Rohde *et al*. 2007). The community-based model is rather different from the firm-based model typical of proprietary software. For instance, knowledge is a public asset which can be distributed and membership is open to everybody in a community-based model whereas in the firm-based model, knowledge is private and membership is restricted to the size of the firm (Lee and Cole 2003). When interactions take place using

electronic media, these communities are often referred to as virtual communities or networks of practice (Johnson 2001).

The process underlying the construction and nurturing of knowledge in CoPs is called legitimate peripheral participation (LPP) (Lave and Wenger 1991, Zhang and Stork 2001). LPP is the process by which newcomers become full members by learning from more competent practitioners and by being allowed to participate in certain tasks that relate to the practice of the community (Kimble *et al*. 2000). Learners' experience does not appear as a result of being taught, but through direct engagement in the social, cultural and technical practice of the community. At first, newcomers can only peripherally participate in small and easy tasks. Through the LPP process, learners create their own learning curriculum by developing a global view of the community and what there is to be learned. Through interaction and collaboration with competent practitioners and other learners, learners gain knowledge and become competent in undertaking more important roles, changing their relationships with other members and transforming their roles in the community. Gradually, peripheral participants move towards the centre of the community and eventually establish their identities as competent practitioners in the community. The dual process is called reification, which means giving concrete form to something that is abstract. It is the process underlying the construction of explicit content (Hildreth and Kimble 2002). This article is focused on the participation process that will be analysed through SNA techniques. The content analysis is out of the scope of this article, but it could be incorporated as a future research using latent semantic analysis techniques (Blei *et al*. 2003).

2.2. *Community structure*

OSS communities are typically initiated by an individual (or group of individuals) who provides systems and development components, or their access, as well as communication infrastructure. Participants are usually volunteers and contributors are not normally motivated by traditional economic incentives, but rather by instrumental factors associated with fulfilling a need, and by intrinsic factors such as enhanced reputation, expertise development (learning), self-fulfilment, as well as basic fun and enjoyment (Lerner and Tirole 2002, von Hippel and von Krogh 2003, Leimeister *et al*. 2004). The individuals who participate in OSS projects are often described as comprising a community.

These communities have been described as having an onion-like structure, with a central core of highly active individuals, surrounded by other layers of progressively less active individuals. One example of this is presented by Ye *et al*. (2005) in which the central core is composed of the project leaders and core members, with five outer layers containing active developers, peripheral developers, bug reporters, passive users and stakeholders, respectively.

It has been demonstrated that much of the OSS development is realised by a small percentage of individuals despite the fact that there are tens of thousands of available developers. Such concentration is called 'participation inequality' (Kuk 2006), and it can be explained by the different user profiles of open source communities. Consequently, the structure of OSS communities is not completely flat as it was claimed by the bazaar model of full participation (Raymond 1998). This argument is based on the logic that a highly participative community may lead to richer discussion, better flow of ideas, efficient code development, faster bug finding and fixing, and, hence, faster and efficient project growth (Weber 2004). However, contrary to the bazaar view of OSS projects, many empirical studies have found that only a small number of developers contribute a large percentage of code and discussion in OSS projects. This concentration is clearly shown in Figure 1, which corresponds to the situation of the ARM Debian Linux community during the year 2007. The percent of the community versus the number of contributions has been represented. It can be concluded that more than 60% of the community is integrated by peripheral participants. They are not making any contributions, they are just learning from more competent practitioners. The remainder (40%) is made up of individuals who can be categorised according to the degree of their contributions, as shown in Figure 1. In this case, peripheral and active developers are more or less balanced as it is a specialised community. The number of peripheral users is typically much bigger for non-specialised communities (Zhang and Stock 2001).

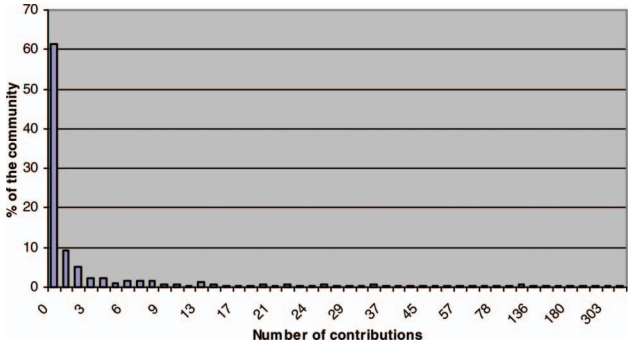

Figure 1. Distribution of contributions of ARM Debian Linux community during 2007.

Participation inequality allows the categorisation of OSS community members into three groups (Mockus *et al*. 2002, Xu *et al*. 2005):

- Core members. They are responsible for guiding and coordinating the development of an OSS project. They are usually involved with the project during a long period of time and have made significant contributions to the development and evolution of the system. Moderators and leaders are included in this group.
- Active developers. They regularly make contributions to the project.
- Peripheral developers. They occasionally contribute with new features to the existing system. This contribution is irregular, and the period of involvement is short and sporadic.

Notice that information and participation are open to everybody in virtual communities. No level of expertise is mandatory to post a message, which is the way in which newcomers begin their participation. However, a certain level of expertise is required to answer questions and participate in topics under discussion. The amount of time required to achieve this level of expertise depends on the involvement of each individual participant. Some studies reveal that there is a correlation between level of expertise and time online (Maybury 2001).

Obviously, there are also some limitations to virtual communities. One of them is the presence of free riders (Wasko and Faraj 2005). They can be defined as members who enjoy the benefits of the collective good without contributing to its establishment. Although they are tolerated, an excessive presence of free riders can constitute a threat for the community. A percentage of peripheral members is expected to grow into becoming full members in order for the community to survive. Free riders will not grow towards full members. Therefore, an excessive presence may damage the evolution of the community in the end. Some other limitations refer to the necessity of motivating network members to participate in the community and to share valuable knowledge with each other openly (Dyer and Nobeoka 2000). Knowledge sharing requires the consideration of knowledge as a public good owned by the community.

3. Research framework: hypotheses

The central question of this article consists of identifying which factors affect the successful development of OSS virtual communities from the perspective of SNA, such as the degree of interactions, the topology of the community and the different profiles of the participants. From a social network viewpoint, individuals and their actions are interdependent because individuals are embedded in networks of relationships (Wasserman and Faust 1994). Several previous works have applied SNA techniques to discover patterns of interactions in virtual communities. For instance, Wasko *et al*. (2009) defines electronics networks of practice and then describes how theories of collective action are relevant for understanding the social and structural characteristics of these electronic networks, and Jung (2009) applies SNA techniques to peer-to-peer networks to detect the fraud and adversarial information drained out from the malicious peers. More specifically related to OSS communities, SNA techniques have been applied to the identification of knowledge brokers (Sowe *et al*. 2006), to analyse the social status of the participants (Bird *et al*. 2006) and the internal structure of concurrent versioning system (CVS) repositories (Lopez-Fernandez *et al*. 2004). This article also makes use of SNA techniques, but the focus consists of extracting the internal structure of the community that contributes to its successful development. First, it should be defined what a successful development is. Success in a virtual community could be manifested through the level of participation, which can be understood as the number of participants and the number of messages posted in the community (Preece 2001). Hinds and Lee (2008) define success as the level of community activity and quantity of community work output. It is crucial for the community survival that there is a certain level of activity around the emergent topics of discussion. Users usually consult communities when they need solutions or alternatives to a particular problem. If the posted question doesn't generate any activity, a user will be frustrated and he or she will never be engaged as a regular participant. Activity can be measured through the number of authors, messages and threads.

However, not all the participants exhibit the same level of participation. Consequently, the idea of activity should be analysed taking into account antecedents such as the degree of interactions. Interactions in OSS virtual communities take place through threads of discussions. Core or active developers answer posted questions and sometimes a debate arises around a certain topic. Whenever an author is answering a posted question, an interaction between them emerges. In particular, when someone is posting a message to a thread, he is actually answering all the previous posted messages, as the answer must be consistent with the previous history of the debate (Knock 2001). Degree of interactions refers to the extent in which the interactions take place. It is expected that the activity of virtual communities will

be promoted if they are able to attract and motivate more active participants (Hinds and Lee 2008) leading to more interactions. Therefore, we hypothesise:

> *Hypothesis H1: A high degree of interactions in virtual communities supporting OSS projects will have a positive effect on the activity of virtual communities.*

Open source communities produce software systems that are not only intended for use by themselves but also for external clients. These external users – i.e. users that are not actively involved in the open source online community – can be made active developers that modify the source and regive it to the project (Kindsmüller *et al.* 2005). The reciprocities, the interlinkages and penetrations among interdependent messages constitute the conversational interactivity embedded within a thread. Participation in virtual communities occurs through the process of connecting seekers and responders through knowledge. Some authors claim that exchange pattern in virtual communities is generalised in nature rather than dense or reciprocal, because people typically do not know each other and participation is discretionary (Wasko *et al.* 2009). By creating a network of contacts, the chances of getting to relevant information sources increases dramatically (Bergquist and Ljungberg 2001), and the quality of information has been pointed out as one of the motivations to participate in virtual communities (Lin and Lee 2006). The more diverse the views and ideas expressed in a discussion thread are, the higher the level of conversational interactivity is (Kuk 2006). The presence of active developers is crucial for the activity of the virtual community and they constitute one of the major development forces in OSS projects (Yunwen and Kishida 2003). This leads to the second hypothesis:

> *Hypothesis H2: A high number of active developers will have a strong positive effect on the activity of virtual communities.*

Nevertheless, the proportion of active developers should remain low as compared with the number of peripheral developers. A community with a majority of active developers would not benefit the accessibility of newcomers through LPP processes (Lave and Wenger 1991). Besides, too many developers will inevitably increase the coordination and communication costs (Brooks 1995). As in a 'real' community, information overload definitely plays an important role in the formation of virtual community. Jones *et al.* (2001) observed that as the number of interactive posters in forums increases, the number of interactive messages decreases. This observation is related to the cognitive abilities of people to digest huge amount of information (Rafaeli *et al.* 2004). In accordance with them, Huang and Liu (2005) say that many open source developers do not contribute a great deal to open source development as they only do relatively minor work, such as fixing non-critical bugs, and do not make major contributions to the development process. Thus it would be said that:

> *Hypothesis H3: A high proportion of active developers in the community will have a negative effect on the activity of virtual communities*

Core members and active developers should strive to create an environment and culture that fosters the sense of belonging to the community and mechanisms that encourage and enable newcomers to move towards the centre of the community through continual contributions, improving community activity (Yunwen and Kishida 2003). OSS projects would benefit from people of the active developers group being nominated to the core group. The core team represents the most productive group, and consequently, they improve the community activity (Xu *et al.* 2005). This subset of active contributors has been labelled 'critical mass', referring to the idea that a certain threshold of participation or action has to be achieved for a social movement to arise (Oliver and Marwell 1988). They define the critical mass as 'a small segment of the population that chooses to make big contributions to the collective action while the majority do little or nothing' (Oliver *et al.* 1985, p. 524). Therefore, theoretically we can expect that collective action in virtual communities is sustained through the efforts of a minority of individuals who constitute a critical mass. Thus:

> *Hypothesis H4: A high proportion of core members in the active developers group will have a positive effect on the activity of virtual communities.*

Because software development is associated with various complex tasks including code development and debugging, studies have suggested that sufficient human resources are essential for the success of the software team and the final product (Mockus *et al.* 2002). With higher degree centrality, the project team is able to spread complex tasks and cognitive strains over a larger number of developers, resulting in higher user orientation (the team is able to attend to users' needs and requirements more promptly and more efficiently) and higher productivity (faster and better technical improvements) (Wang 2007). Non-central developers only make relatively minor contributions. Developers with high centrality values play important

roles, whereas those with lower values play peripheral roles (Huang and Liu 2005). Ahuja *et al.* (2003) found that an individual's network centrality in an electronic R&D group was positively associated with helping behaviour and performance. Thus, we hypothesise that:

> *Hypothesis H5: Active developers' centrality will contribute positively to the activity of virtual communities.*

4. Methodology

The roles and their associated influences in OSS communities can be realised only through contributions to the community. In this article, we analysed the mailing lists because they allow the collective reflection and community discussions, and activities are not just confined to software development or coding alone (Sowe *et al.* 2006). Interactions are usually structured in threads of discussion, which facilitates their analysis.

The simplest way to classify threads is using their length, i.e. the total number of posts they contain. Posts per thread – how densely packed posts are in a collection of threads – turns out to be a reliable metric to determine the degree of 'conversational concentration' of an author in a given group (Bonacci 2004). Nevertheless, this kind of data does not provide any information about the social structure of the community, or about the relationships among authors. In this article, social networks will be extracted from threads of discussion, and SNA techniques will be applied to provide new insights in the community organisation (Cho *et al.* 2005, Stefanone and Gay 2008). A social network can be represented as a graph $G = (V,E)$ where V denotes a finite set of vertices and E denotes a finite set of edges such that $E \subseteq V \times V$. Some network analysis methods are easier to understand when graphs are conceptualised as matrices:

$$M = (m_{i,j})_{n*n} \quad \text{where} \quad n = |V|,$$
$$m_{i,j} = \begin{cases} 1 & \text{if}(v_i, v_j) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

In case of a valued graph, real valued weight function $w(e)$ is defined on the set of edges, i.e. $w(e) = Ex\Re$, and the matrix is then defined as:

$$m_{i,j} = \begin{cases} w(e) & \text{if}(v_i, v_j) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

In the context of threads of discussion, V is given by all the authors posting messages and E is given by the successive answers among authors inside a thread, which is the basic unit considered (Jones *et al.* 2004). The use of discussion threads as the basic unit of analysis is highly valid, considering that the epistemic interactions in support of OSS development often take place in discussion threads where individual postings provide the context to encourage participation (Kuk 2006). In contrast to a reply to a single message, it is more cognitively complex to reply to a threaded discussion, because the ebb and flow of earlier postings must be taken into account to develop a coherent answer (Knock 2001). That is the reason why an author posting to a thread will be tied to all the authors who have previously posted to the same thread when constructing the social network. The resulting graph will exhibit the following features:

- It will be a directed graph. Usually, the word edge is reserved for undirected lines, while arc is the notation used for directed lines. The direction of the arc is given by the flow of information between two authors. That means that a sender (the tail of the arc) is answering a receiver (the head of the arc) inside a thread of discussion.
- It will be a valued graph. An author is able to participate several times inside a thread or can answer to the same authors in different threads. Consequently, the function $w(e)$ is a measure of the strength of the relationship between two authors.

Figure 2 illustrates a graph example corresponding to the ARM Debian Linux community. Vertices represent community members, and they are linked with arcs which represent the flow of information between two vertices. Vertices labels and arc values have been omitted for clarity.

Networks can be partitioned using some discrete characteristics of vertices. For instance, several classes of vertices can be obtained using the function $w(e)$, that

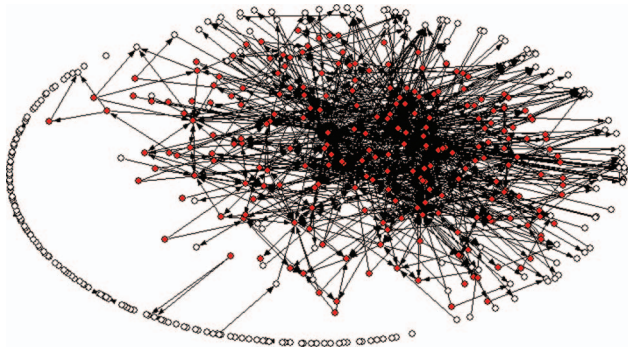

Figure 2. Community structure.

is, the strength of arcs. In the case of OSS projects, these kinds of partitions should highlight the core/periphery (C/P) structure of the community. A C/P structure divides vertices in two distinct subgroups: vertices in the core, densely connected with each other, and vertices on the periphery, not connected with each other, only nodes in the core. In network analysis, density is a measure of the cohesion of the network. More ties between people yield a tighter structure, which is, presumably, more cohesive. Density can be defined as the number of lines in a simple network, expressed as a proportion of the maximum possible number of lines. Consequently, maximum density is found in a network where all pairs of vertices are linked by two arcs, one in each direction. However, network density is not very useful when dealing with valued networks. In this case, it is better to look at the number of ties in which each vertex is involved. This is called the degree of a vertex. As we are involved with a directed network, we will actually use the concept of out-degree of a vertex, that is, the number of arcs it sends. Therefore, the average out-degree of all vertices could be used to measure the structural cohesion of a network independently of the network size. Figure 2 uses the average out-degree value as the threshold value to distinguish between active and peripheral developers. The latter are shown as blank vertices placed on the periphery of the community while active developers are the coloured vertices placed on more central positions of the community. This threshold value has been chosen to compare different virtual communities, independently of the size of the community. Notice that there is no explicit criterion to distinguish peripheral, active and core developers. That is the reason why a size independent threshold has been arbitrarily chosen. Following this reasoning, a core group of active developers can be also extracted from the group of active developers. Instead of using an absolute threshold value which would be dependent of the community size, the average out-degree plus the standard deviation of the out-degree distribution will be used as the extraction criterion of core members of the community. Using these general guidelines, the following variables can be extracted from the SNA of the virtual community:

- Messages, threads and authors: these variables (V1, V2 and V3) can be easily extracted in OSS projects because posts are archived and sorted attending to them. The three of them represent a measure of the community size and activity.
- Community out-degree: out-degree of a social network represents the degree of interactions in threads of discussion. Consequently, average and standard deviation out-degree values (V4 and V5) will be obtained to be used as a threshold to distinguish among peripheral, active and core developers.
- Active developers: the absolute value of active developers (V6) and their percentage with respect to the whole community (V7) will be evaluated to consider the specific weight of this group.
- Betweenness: it is a measure of centrality that rests on the idea that a person is more central if he or she is more important as an intermediary in the communication network (Nooy *et al*. 2005). The centrality of a person depends on the extent to which he or she is needed as a link in the chains of contacts that facilitate the spread of information within the network. The more a person is a go-between, the more central his or her position is in the network. If we consider that the shortest path between two vertices (geodesic) is the most likely channel for transporting information between actors, an actor who is situated on the geodesics between many pairs of vertices is very important to the flow of information within the network. The betweenness centrality of a vertex is the proportion of all geodesics between pairs of other vertices that include this vertex, and betweenness centralisation of the network is the variation in the betweenness centrality of vertices divided by the maximum variation in betweenness centrality scores possible in a network of the same size (Nooy *et al*. 2005). Two values of betweenness will be considered: the betweenness of the whole network (V8) and the betweenness of the sub-network of active developers (V9). Notice that variable V8 includes peripheral participants while variable V9 just includes active developers. The value of V8 is expected to be always lower than the value of V9, since the consideration of the whole network affects the denominator of the betweenness centrality definition.
- Core developers: the absolute value of core developers (V10) and their percentage with respect to sub-network of active developers (V11) and the whole community (V12) will be evaluated to consider the specific weight of this group. The importance of core developers has been highlighted in several studies (Toral *et al*. 2009), and the three considered variables are trying to measure the extent in which their presence is important.
- Active and core developers out-degree: the average out-degree value of the sub-networks of active developers (V13) and core developers (V14) are measures of participation inequality. The relative importance of the core will be

measured evaluating the percentage of the out-degree because of the core members of the community (V15). These variables are trying to measure the extent in which participation inequality is important for the community development.

5. Case study: Linux ports to embedded processors

Linux is a PC-based operating system that has been developed as OSS along the structure of the UNIX operating system, and it is one of the most prominent examples of OSS projects. The same as Windows is the most prominent operating system released under a proprietary software licence, Linux is the most prominent operating system released under a free licence such as GPL (Hertel 2003). Although Linux started as a hobby in 1991, it represents today a serious threat to Microsoft Windows' market dominance in operating systems (Cusumano and Selby 1997, Spinello 2003). Nevertheless, the proposed case study will be focused on Linux ports to other processor architectures not intended for desktop or personal computer market. There are several reasons for this choice. First, Linux is firmly in first place as the operating system of choice for smart gadgets and embedded systems (Henkel 2006). Second, in contrast to other typical open source projects or even desktop Linux project, most contributions in this field do not come from volunteers or hobbyists, but from commercial firms, many of which are dedicated embedded Linux firms. Third, there are a lot of communities supporting each one of these Linux ports, and this is an excellent opportunity for analysing a big group of more or less 'homogeneous' communities.

Up to 11 virtual communities have been considered. They are listed in Table 1. Nine of them are Debian Linux ports to different processor architectures. The Debian Project is an association of individuals who have made common cause to create a free operating system called Debian GNU/Linux, or simply Debian for short (Robles *et al*. 2005). The other two virtual communities are specific Linux ports to ARM and PowerPC processors. They have been considered because of the special importance of these two families of processors.

Variables V1–V15 have been measured for each virtual community. As communities are not static entities and they are continuously evolving, variables have been measured per community and per year. The period of time analysed goes from 2001 to 2007, as this is the common period in which all considered communities were active. When accounting authors, it is necessary to consider the fact that they are identified

Table 1. Virtual communities considered in the proposed case study.

	URL	Description
The ARM Linux project (ARM)	http://www.arm.linux.org.uk/	ARM Linux is a port of the successful Linux Kernel to ARM processor based machines.
Debian port to ARM (D-ARM)	http://lists.debian.org/debian-arm/	ARM port for Debian GNU/Linux. Debian fully supports a port to little-endian ARM.
Linux PPC port (PPC)	http://penguinppc.org/	PowerPC Linux is the Linux kernel running on a PowerPC processor.
Debian port to PowerPC (D-PPC)	http://lists.debian.org/debian-powerpc/	PowerPC port of Debian GNU/Linux. The PowerPC architecture allows both 64-bit and 32-bit implementations.
Debian port to m68k (D-68k)	http://lists.debian.org/debian-68k/	Motorola 68k port of Debian GNU/Linux. Debian currently runs on the 68020, 68030, 68040 and 68060 processors.
Debian port to Alpha (D-Alpha)	http://lists.debian.org/debian-alpha/	The purpose of this project is to assist developers and others interested with the ongoing project to port the Debian distribution of Linux to the Alpha family of processors.
Debian port to MIPS (D-MIPS)	http://lists.debian.org/debian-mips/	MIPS port of Debian GNU/Linux, able to run at both endiannesses.
Debian port to BSD (D-BSD)	http://lists.debian.org/debian-bsd/	This is a port of the Debian operating system, complete with apt, dpkg, and GNU userland, to the NetBSD kernel.
Debian port to HPPA (D-HPPA)	http://lists.debian.org/debian-hppa/	This is a port to Hewlett-Packard's PA-RISC architecture.
Debian port to Hurd (D-HURD)	http://lists.debian.org/debian-hurd/	The GNU Hurd is a totally new operating system being put together by the GNU group.
Debian port to SPARC (D-SPARC)	http://lists.debian.org/debian-sparc/	This port runs on the Sun SPARCstation series of workstations, as well as some of their successors in the sun4 architectures.

using aliases. Aliases usually correspond to a unique email address, but sometimes several email addresses are used by the same person. Therefore, header of messages should be processed to check there is no duplicity of aliases or emails (Bird *et al*. 2006).

Figure 3 compares the selected communities in terms of average number of messages, threads and community members. Data are displayed as a bar diagram following the list of communities detailed on the right hand of the figure. Although the considered communities exhibit different sizes in terms of messages, threads and authors, it can be observed that these variables are highly correlated.

Figure 4 compares the selected communities in terms of the different user profiles which integrate the whole community. Again, the ratios among members and active and core developers remain approximately constant through the considered communities.

6. Results

A stepwise regression screening procedure was carried out in order to come up with as few variables as possible while still predicting the dependent variable. Stepwise regression is designed to find the most parsimonious set of predictors that are most effective in predicting the dependent variable. Variables are added to the regression equation one at a time, using the statistical criterion of maximising the R^2 of the included variables. The process of adding more variables stops when all of the available variables have been included or when it is not possible to make a statistically significant improvement in R^2 using any of the variables not yet included.

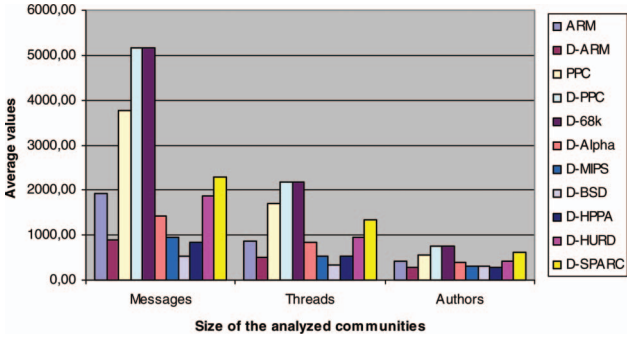

Figure 3. Average size of the analysed communities during the period 2001–2007.

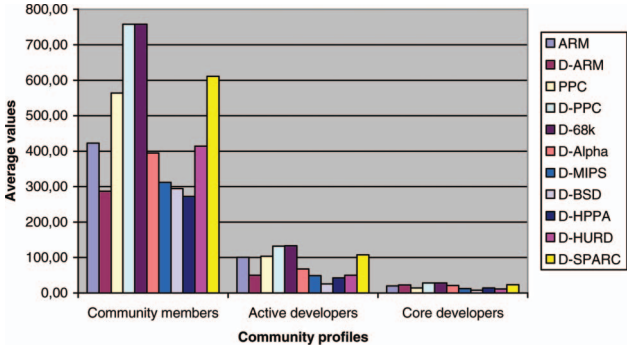

Figure 4. Community profiles of the analysed communities during the period 2001–2007.

The dependent variable is the activity of the community, which can be represented by the number of messages, threads or authors. As these variables are highly correlated, the number of threads has been chosen as the dependent variable representative of the community activity. This is also consequent with the fact of using threads as the basic unit of analysis in which interactions among participants take place. The best subset of independent variables obtained using the stepwise screening procedure were variables V4, V6, V7, V9 and V11. The rest of the variables were dropped from the list of predictors as they couldn't improve the R^2 of the included variables. There are several reasons to explain why these variables were dropped:

- Variable V8 is the betweenness centrality including the whole network. Obviously, the contribution of peripheral vertices is scarce. Consequently, the variability of V8 is much lower than variability of V9, which is working better as a predictor.
- Variables V10–V12 are measuring the presence of core developers. Again, their variability as an absolute value or as a percentage of the whole community is much lower than their variability as a percentage of active developers. That is the reason why just V11 is obtained as a predictor of the community activity.
- Finally, variables V13–V15 were not considered as predictors because participation inequality is a common issue among all the considered communities. Consequently, this concept is not relevant for explaining the community activity.

The obtained predictors have been used to verify the proposed hypothesis. All of them have been measured per year for the 11 considered communities. This leads to 77 cases.

As a previous stage, it is necessary to check assumptions of normality and linearity, and detecting outliers.

Table 2 tests the assumptions of normality and linearity. The skewness and kurtosis of each distribution should be between -1.0 and $+1.0$ to satisfy the criteria for a normal distribution (Bai and Ng 2005). Variables V4 and V7 do not satisfy this criterion.

Therefore, a logarithmic transformation was applied to resolve non-normality. Table 3 also shows the correlations between dependent and independent variables. Correlations for V4 and V6 are significant whereas variables V7, V9 and V11 exhibited a weak relationship with the dependent variable.

The transformed variables were incorporated in another regression analysis to identify outliers as standardised residuals. In multiple regressions, an outlier in the solution can be defined as a case that has a large residual because the equation did a poor job of predicting its value. Detection of outliers in the regression solution is based on an analysis of the residuals, or prediction error not accounted for by the regression (Hutcheson and Sofroniou 1999). Standardising the residuals means we can detect outliers with large residuals using the same z-score criteria we used for univariate outliers (± 3.00). Table 3 shows that all standardised residuals were less than ± 3.0 by looking at the minimum and maximum standardised residuals. Consequently, there are no outliers.

The results of running the regression analysis including transformed variables are shown in Table 4.

Table 2. A sample table.

	Assumption of normality		Correlations
	Skewness	Kurtosis	Threads
V4	1.412	1.451	0.697[a]
Log(V4)	0.010	−0.301	0.690[a]
V6	0.565	−0.550	0.875[a]
V7	1.023	2.732	0.170
Log(V7)	−0.222	0.498	0.203
V9	0.049	−0.232	0.126
V11	0.168	0.575	−0.220

[a]Correlation is significant at the 0.01 level (2-tailed).

Multiple regressions assume that the errors are independent and there is no serial correlation. No serial correlation implies that the size of the residual for one case has no impact on the size of the residual for the next case. The Durbin–Watson statistic is used to test for the presence of serial correlation among the residuals. The value of the Durbin–Watson statistic ranges from 0 to 4. As a general rule of thumb, the residuals are not correlated if the Durbin–Watson statistic is approximately 2, and an acceptable range is 1.50–2.50 (Ott and Longnecker 2001). The Durbin–Watson statistic for the regression of Table 2 is 1.897, which falls within the acceptable range. Coefficients of regression analysis are shown in Table 5.

Multicollinearity is a problem in regression analysis that occurs when two independent variables are highly correlated. According to Table 6, the tolerance values for all of the independent variables are larger than 0.10. Consequently, multicollinearity is not a problem in this regression analysis.

On the basis of the ANOVA table for the standard multiple regression ($F(5, 71) = 109.858$, $p < 0.001$), there was an overall relationship between the dependent variable and independent variables. The Multiple R for the relationship between the combined set of independent variables and the dependent variable was 0.941, which would be characterised as a very strong relationship.

Finally, the proposed hypotheses were checked using the β coefficient of Table 5. The significant positive value of the β coefficient for variable V4 confirms hypothesis H1: the average out-degree has a positive impact in the community activity. The highest significant β value corresponds to V6, showing the strong relationship among the number of active developers and community activity, as it was claimed by hypothesis H2. The negative β value of V7 confirms

Table 3. Residuals statistics.

	Minimum	Maximum	Mean	Std. deviation	N
Predicted value	−232.0600	2684.3601	1081.5325	687.45133	77
Residual	−727.29041	752.96051	0.00000	247.15551	77
Std. predicted value	−1.911	2.332	0.000	1.000	77
Std. residual	−2.844	2.945	0.000	0.967	77

Table 4. Regression including transformed variables: model summary.

					Change statistics					
Model	R	R^2 square	Adjusted R^2	Std. error of the estimate	R^2 change	F change	$df1$	$df2$	Sig. F change	Durbin–Watson
1	0.941[a]	0.886	0.877	255.71012	0.886	109.858	5	71	0.000	1.897

[a]Predictors: (Constant), V11, LogV4, LogV7, V9, V6.

Table 5. Coefficients of regression analysis.

Model		Unstandardised coefficients		Standardised coefficients			Collinearity statistics	
		B	Std. error	β	*t*	Sig.	Tolerance	VIF
1	(Constant)	667.916	432.907		1.543	0.127		
	LogV4	601.888	120.330	0.245	5.002	0.000	0.670	1.493
	V6	15.483	1.006	0.,895	15.398	0.000	0.478	2.094
	LogV7	−1173.971	266.373	−0.222	−4.407	0.000	0.637	1.569
	V9	47.974	236.176	0.010	0.203	0.840	0.667	1.500
	V11	7.316	3.936	0.102	1.859	0.067	0.532	1.879

Table 6. ANOVA for multiple regression.

Model		Sum of squares	*df*	Mean square	*F*	Sig.
1	Regression	35916789.020	5	7183357.804	109.858	0.000[a]
	Residual	4642524.149	71	65387.664		
	Total	40559313.169	76			

[a]Predictors: (Constant), V11, LogV4, LogV7, V9, V6.

H3: a low proportion of active developers with respect to peripheral contributors will guarantee that the last group can benefit from LPP processes to become insiders. Hypothesis H4 is weakly confirmed with the positive β value coefficient of variable V11. The β value is the lowest among the significant β values and even the significance is higher than 0.05. Finally, hypothesis H5 cannot be confirmed because the β coefficient for variable V9 is not significant.

The obtained results confirm the important role the different user profiles play on the activity and the development of virtual communities. A certain structure of the virtual community is required to achieve a successful result. Although the activity is essentially concentrated on the active developers group, they should represent a low percentage of the community members. This result highlights the importance of peripheral contributors. On one hand, they represent the active or core contributors of the future, as they will increase their knowledge through the interaction with regular contributors. On the other hand, they represent a measure of the interest of users in the underlying software project. If an OSS project is not able to attract people to the discussion forums, that means the software has a limited interest among potential users, and finally the project will be abandoned.

7. Discussion and implications

The obtained results show that OSS communities are developed under the participation inequality scheme in which the structure is given by the level of involvement of community members. Peripheral users are necessary to expand the underlying software and to nurture the group of mature developers through LPP process. Core members and active developers should strive to create an environment and culture that fosters the sense of belonging to the community and mechanisms that encourage and enable some newcomers to move towards the centre of the community through continual contributions (Yunwen and Kishida 2003). In this sense, the presence of active and core developers has been shown to be the most important factor affecting the successful development of OSS communities in the proposed analysis (hypothesis H2). This asseveration is supported by the generalised exchange pattern typical of OSS communities (Wasko *et al.* 2009) and their hierarchical structure (Mockus *et al.* 2002).

However, theoretically it has also been more difficult to sustain collective action in large groups as contributions are more likely to go unnoticed or seen as unnecessary (Hardin 1982). In accordance with Oliver–Marwell studies, larger groups are less likely to engage in collective action than smaller ones. This issue is solved in OSS communities through the distinction between active and peripheral users. Although the community is open to everybody, even free-riders, just a small fraction of the community is responsible for the collective action (Wasko *et al.* 2009). This fact explains the negative influence of a high proportion of active developers tested by hypothesis H3.

The degree of interaction is a measure of the network cohesion. Our analysis indicates the positive influence of the network cohesion in the activity of the community (hypothesis H1). This is consistent with

the findings of other studies, providing further support for the theoretical argument for the advantages of cohesive networks (Gulati 1995, Hagedoorn and Duysters 2002). For instance, they are conducive for developing trust and enhancing reputation mechanisms which are one of the main motivations of core developers (Lerner and Tirole 2002, von Krogh and Spaeth 2007). Moreover, they are also beneficial for building absorptive capacity, i.e. the ability to discover, evaluate, exploit and dissimilate novel and distant knowledge (Hagedoorn and Duysters 2002, Gilsing and Nooteboom 2005).

A high level of interaction is also desirable so new threads are answered and doubts and problems can be solved in a reasonable time. Newcomers are motivated by a virtual community if they feel the time they spend is useful. When a posted message does not receive any answer, a newcomer could feel frustrated and decide to abandon the community. That is why the role of active and core developers is so important. People who want to start an OSS project should consider the necessity of planning a community, and choosing a selection of developers to maintain and promote it. This result is consistent with the findings of Mockus *et al*.'s (2002) case study which suggests that a critical mass of core developers is important for the success of Apache and Mozilla.

Finally, no evidence about the influence of active developers' centrality has been obtained. A possible explanation is that centrality should only be considered relevant for a particular profile of participants, the knowledge brokers, and not for the whole group of active developers. Some studies (Sowe *et al*. 2006) suggest that knowledge brokers bridge the gap between expert software developers and user communities.

The main contribution of this study is the identification of social structures of communities supporting OSS projects. The obtained results will contribute to OSS managers' knowledge on how they should structure their teams and nurture their social network resources in order to produce software products that have a greater chance of being adopted and deployed. For instance, OSS managers should try to attract active developers to sustain the collective action attending to their motivations. Two kinds of motivations are frequently reported in the literature: intrinsic and extrinsic motivations (Osterloh and Rota 2007). Intrinsic motives include benefits gained from enjoyment and pro-social motives. Extrinsic motives include benefits gained from extended functionality and reputation (Hars and Ou 2002). Interactions should also be promoted as the mechanism to guarantee that some of the peripheral users will become future experts as a process of social learning. In this sense, the core group must be responsible for helping the underlying project to engage in a discourse and co-learning experience with their user communities and promoting the training action (Sowe *et al*. 2005).

8. Conclusions

This research addresses an emergent area of interest in the software market sector like OSS projects. The ways in which these projects are developed are in stark contrast with traditional proprietary software. For instance, OSS projects are based on virtual communities supporting the code development and the interactions between knowledge seekers and knowledge providers. Although it could seem that virtual communities are chaotically organised, the reality is that they are well structured and different user profiles can be distinguished. To go more deeply in this field, this article proposes to analyse the activity of virtual communities using SNA techniques. Data from several virtual communities related to Linux ports to different processor architectures have been collected. The obtained results confirm the necessity of structuring the virtual community with a selection of active developers and core members to promote community activity and attract peripheral users, expanding the impact of the underlying software.

There are several limitations in the current research. First, the sample is constrained to Linux ports communities. Although this may limit the generalisability of the results, the advantage of using a homogeneous group of communities in terms of size is avoiding the distortion caused by very different communities. Second, the study has extracted several variables related to SNA, but some other variables could be used to measure the interactions or the centrality of the community, leading to new hypotheses and new results. Third, the considerations of threads as the unit of analysis is not taking into account cross relations between threads, threads that are redundant over time or the content of threads. A semantic analysis of the content in conjunction with SNA techniques could be carried out to consider this issue.

Acknowledgements

This work has been supported by the Spanish Ministry of Education and Science (Research Project with reference DPI2007-60128) and the Consejería de Innovación, Ciencia y Empresa (Research Project with reference P07-TIC-02621).

References

Applewhite, A., 2003. Should governments go open source? *IEEE Software*, 20 (4), 88–91.

Ahuja, M., Galletta, D., and Carley, K., 2003. Individual centrality and performance in virtual R&D groups: An empirical study. *Management Science*, 49 (1), 21–38.

Bai, J. and Ng, S., 2005. Tests for skewness, kurtosis, and normality for time series data. *Journal of Business & Economic Statistics*, 23 (1), 49–60.

Bergquist, M. and Ljungberg, J., 2001. The power of gifts: organizing social relationships in open source communities. *Information Systems Journal*, 11 (4), 305–320.

Bird, C., *et al*., 2006. Mining email social networks. *In*: *Proceedings of the International Workshop on Mining Software Repositories,* MSR'06, 22–23 May 2006, Shanghai, China. New York: ACM, 137–143.

Blei, D.M., Ng, A.Y., and Jordan, M.I., 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3, 993–1022.

Bonacci, D., 2004. Towards quantitative tools for analysing qualitative properties of virtual communities. *Interdisciplinary Description of Complex Systems*, 2 (2), 126–135.

Bonaccorsi, A. and Rossi, C., 2003. Why open source software can succeed. *Research Policy*, 32, 1243–1258.

Brooks, F.P., 1995. *The mythical man-month*. Anniversary ed. Addison Wesley.

Cho, H., *et al*., 2005. Development of computer-supported collaborative social networks in a distributed learning community. *Behaviour & Information Technology*, 24 (6), 435–447.

Crowston, K., *et al*., 2005. Effective work practices for FLOSS development: a model and propositions. *In*: *Proceedings of the 38th Hawaii international conference on system sciences*, *HICSS '05*, 3–6 January, Hawaii, 197.

Crowston, K. and Scozzi, B., 2002. Open source software projects as virtual organisations: competencyrallying for software development. *IEE Proceedings – Software*, 149 (1), 3–17.

Cusumano, M. and Selby, R., 1997. How microsoft builds software. *Communications of the ACM*, 40 (6), 53–61.

Dyer, J.H. and Nobeoka, K., 2000. Creating and managing a high-performance knowledge-sharing network: the toyota case. *Strategic Management Journal*, 21, 345–367.

Gambardella, A. and May, B.H., 2006. Proprietary versus public domain licensing of software and research products. *Research Policy*, 35, 875–892.

Gilsing, V.A. and Nooteboom, B., 2005. Density and strength of ties in innovation networks: an analysis of multimedia and biotechnology. *European Management Review*, 2 (3), 179–197.

Gruber, M. and Henkel, J., 2006. New ventures based on open innovation – an empirical analysis of start-up firms in embedded Linux. *International Journal of Technology Management*, 33 (4), 356–372.

Gulati, R., 1995. Does familiarity breed trust? The implications of repeated ties for contractual choice in alliances. *Academy of Management Journal*, 38 (1), 85–112.

Hagedoorn, J. and Duysters, G., 2002. Learning in dynamic inter-firm networks: the efficacy of multiple contacts. *Organization Studies*, 23 (4), 525–548.

Hallen, J., *et al*., 1999. Linux in the workplace. *IEEE Software*, 16 (1), 52–57.

Hardin, R., 1982. *Collective action*. Baltimore, MD: Johns Hopkins University Press.

Hars, A. and Ou, S., 2002. Working for free? Motivations for participating in open source projects. *International Journal of Electronic Commerce*, 6 (3), 25–39.

Hemetsberger, A. and Reinhardt, C., 2006. Learning and knowledge-building in open-source communities. A social-experiential approach. *Management Learning*, 37 (2), 1350–5076.

Henkel, J., 2006. Selective revealing in open innovation processes: the case of embedded Linux. *Research Policy*, 35 (7), 953–969.

Hertel, G., Niedner, S., and Herrmann, S., 2003. Motivation of software developers in open source projects: an internet-based survey of contributors to the Linux kernel. *Research Policy*, 32, 1159–1177.

Hildreth, P.M. and Kimble, C., 2002. The duality of knowledge. *Information Research*, 8 (1), 1–17.

Hinds, D. and Lee, R.M., 2008. Social network structure as a critical success condition for virtual communities. *In*: *Proceedings of the 41st Hawaii international conference on system sciences*, *HICSS '08*, 7–10 January, Big Islands, Hawaii.

Huang, S.-K. and Liu, K.-M., 2005. Mining version histories to verify the learning process of Legitimate Peripheral Participants. *In: International conference on software engineering*. *Proceedings of the 2005 international workshop on mining software repositories*, 17 May, St Louis, Missouri, 1–5.

Hutcheson, G. and Sofroniou, N., 1999. *The multivariate social scientist: introductory statistics using generalized linear models*. Thousand Oaks, Calif: Sage Publications.

Johnson, C.M., 2001. A survey of current research on online communities of practice. *Internet and Higher Education*, 4 (1), 45–60.

Jones, G., Ravid, G., and Rafaela, S., 2004. Information overload and the message dynamics of online interaction spaces: a theoretical model and empirical exploration. *Information Systems Research*, 15 (2), 194–210.

Jones, G., Ravid, G., and Rafaela, S., 2001. Information overload and virtual public discourse boundaries. *In*: *Proceedings of eighth IFIP conference on human–computer interaction*, 9–13 July, Tokyo, Japan.

Jung, J., 2009. Trustworthy knowledge diffusion model based on risk discovery on peer-to-peer networks. *Expert Systems with Applications*, 36, 7123–7128.

Kimble, C., Hildreth, P., and Wright, P., 2000. Communities of practice: Going virtual. *In*: P.M. Hildreth and C. Kimble, eds. *Knowledge networks: innovation through communities of practice*. Hershey, PA: Idea Group Publishing, 220–234.

Kindsmüller, M.C., Leuchter, S., and Urbas, L., 2005. Online communities and online community building. *In*: M. Khosrow-Pour, ed. *Encyclopedia of information science and technology*. Hershey, PA: Information Science Publishing, S. 2203–S. 2208.

Knock, N., 2001. Compensatory adaptation to a lean medium: An action research investigation of electronic communication in process involvement groups. *IEEE Transactions on Professional Communication*, 44 (4), 267–285.

Kogut, B. and Metiu, A., 2001. Open source software and distributed innovation. *Oxford Review of Economic Policy*, 17 (2), 248–264.

Kuk, G., 2006. Strategic interaction and knowledge sharing in the KDE developer mailing list. *Management Science*, 52 (7), 1031–1042.

Lakhani, K. and Hippel, E., 2003. How open source software works: 'free' user to user assistance. *Research Policy*, 32 (6), 923–943.

Lave, J. and Wenger, E., 1991. *Situated learning: legitimate peripheral participation*. Cambridge University Press.

Lee, F.S., Vogel, D., and Limayem, M., 2003. Virtual community informatics: a review and research agenda. *Journal of Information Technology Theory and Applications*, 5 (1), 47–61.

Lee, G.K. and Cole, R.E., 2003. From a firm-based to a community-based model of knowledge creation: the case of the Linux kernel development. *Organization Science*, 14 (6), 633–649.

Leimeister, J.M., Sidiras, P., and Kremar, H., 2004. Success factors of virtual communities from the perspective of members and operators: an empirical study. *In*: *Proceedings of the 37th annual Hawaii international conference on system sciences HICSS '04*, 5–8 January, Big Islands, Hawaii.

Lerner, J. and Tirole, J., 2002. Some simple economics of open source. *Journal of Industrial Economics*, 50 (2), 197–234.

Lin, H.F. and Lee, G.-G., 2006. Determinants of success for online communities: an empirical study. *Behaviour & Information Technology*, 25 (6), 479–488.

Lin, H.F., 2008. Determinants of successful virtual communities: contributions from system characteristics and social factors. *Information and Management*, 45, 522–527.

Lopez-Fernandez, L., Robles, G., and Gonzalez-Barahona, J.M., 2004. Applying social network analysis to the information in CVS repositories. *In*: *26th international conference on software engineering ICSE '04*, 23–28 May, Edinburgh. IEEE Computer Society Press, 101–105.

Lussier, S., 2004. New tricks: how open source changed the way my team works. *IEEE Software*, 21 (1), 68–72.

Maybury, M., 2001. Collaborative virtual environments for analysis and decision support. *Communications of the ACM*, 44 (12), 51–54.

Mockus, A., Fielding, T., and Herbsleb, D., 2002. Two case studies of open source software development: Apache and Mozilla. *ACM Transactions of Software Engineering and Methodology*, 11 (3), 309–346.

Nooy, W., Mrvar, A., and Batagelj, V., 2005. *Exploratory network analysis with Pajek*. New York: Cambridge University Press.

Oliver, P., Marwell, G., and Teixeira, R., 1985. A theory of the critical mass. I. Interdependence, group heterogeneity and the production of collective action. *American Journal of Sociology*, 91 (3), 522–556.

Oliver, P. and Marwell, G., 1988. The paradox of group size in collective action: a theory of the critical mass. II. *American Sociological Review*, 53, 1–8.

Open Source Initiative. 1999. *Open Source Definition,* [online]. Available from: http://www.opensource.org/osd.html [Accessed 14 October 2008, last updated 24 July 2006].

Osterloh, M. and Rota, S., 2007. Open source software development – just another case of collective invention? *Research Policy*, 36 (2), 157–171.

Ott, R.L. and Longnecker, M., 2001. *An introduction to statistical methods and data analysis*, 5th ed. Pacific Grove, CA: Duxbury.

Perkins, G., 1999. Cultural clash and the road to world domination. *IEEE Software*, 16 (1), 23–25.

Preece, J., 2001. Sociability and usability in online communities: determining and measuring success. *Behaviour & Information Technology*, 20 (5), 347–356.

Rafaeli, S., Ravid, G., and Soroka, V., 2004. De-lurking in virtual communities: a social communication network approach to measuring the effects of social and cultural capital. *In*: *Proceedings of the 37th Hawaii international conference on system sciences, HICSS '04*, 5–8 January, Big Islands, Hawaii.

Raymond, E., 1998. *The cathedral and the bazaar*. Sebatopol, CA: O'Reilly.

Robles, G., Gonzalez-Barahona, J.M., and Michlmayr, M., 2005. Evolution of volunteer participation in libre software projects: Evidence from Debian. *In*: M. Scotto and G. Succi, eds. *Proceedings of the first international conference on open source systems*, 11–15 July 2005. Genova, 100–107.

Rohde, M., *et al*., 2007. Reality is our laboratory: communities of practice in applied computer science. *Behaviour & Information Technology*, 26 (1), 81–94.

Sowe, S.K., Karoulis, A., and Stamelos, I., 2005. A constructivist view of knowledge management in open source virtual communities. *In*: A.D. Figueiredo and A.P. Afonso, eds. *Managing learning in virtual settings: the role of context*. Hershey, PA: Idea Group Inc., 290–308.

Sowe, S., Stamelos, I., and Angelis, L., 2006. Identifying knowledge brokers that yield software engineering knowledge in OSS projects. *Information and Software Technology*, 48 (11), 1025–1033.

Spinellis, D. and Szyperski, C., 2004. How is open source affecting software development? *IEEE Software*, 21 (1), 28–33.

Spinellis, D., 2006. Open source and professional advancement. *IEEE Software*, 23 (5), 70–71.

Spinello, R.A., 2003. The case against microsoft: an ethical perspective. *Business Ethics: A European Review*, 12 (2), 116–132.

Stefanone, M.A. and Gay, G., 2008. Structural reproduction of social networks in computer-mediated communication forums. *Behaviour & Information Technology*, 27 (2), 97–106.

Toral, S.L., Martínez-Torres, M.R., and Barrero, F., 2009. Modelling mailing list behaviour in open source projects: the case of ARM embedded Linux. *Journal of Universal Computer Science*, 15 (3), 648–664.

Trung, T., Dinh-Trong, and Bieman, J.M., 2005. The FreeBSD project: a replication case study of open source development. *IEEE Transactions on Software Engineering*, 31 (6), 481–494.

Valimaki, M. and Oksanen, V., 2005. The impact of free and open source licensing on operating system software markets. *Telematics and Informatics*, 22, 97–110.

Von Hippel, E. and von Krogh, G., 2003. Open source software and the "private-collective" innovation model: issues for organization science. *Organization Science*, 14 (2), 209–223.

von Krogh, G. and Spaeth, S., 2007. The open source software phenomenon: characteristics that promote research. *The Journal of Strategic Information Systems*, 16 (3), 236–253.

Wagner, C. and Prasarnphanich, P., 2007. Innovating collaborative content creation: the role of altruism and Wiki technology. *In*: *40th annual Hawaii international conference on system sciences, HICSS '07*, Big Islands, Hawaii. IEEE Computer Society Press, 18–28.

Wang, J., 2007. *The role of social networks in the success of open-source software systems: a theoretical framework and an empirical investigation*. Thesis (PhD). Kent State University Graduate School of Management.

Wasko, M. and Faraj, S., 2005. Why should I share? Examining social capital and knowledge contribution in electronic networks of practice. *Management Information Systems Quarterly*, 29 (1), 35–57.

Wasko, M., Teigland, R., and Faraj, S., 2009. The provision of online public goods: examining social structure in an electronic network of practice. *Decision Support Systems*, 47 (3), 254–265.

Wasserman, S. and Faust, K., 1994. *Social network analysis*. Cambridge University Press.

Weber, S., 2004. *The success of open source*. Boston, MA: Harvard University Press.

Wenger, E., 1998. *Communities of practice: learning, meaning, and identity*. Cambridge: Cambridge University Press.

Wu, J.-J. and Tsang, A.S., 2008. Factors affecting members' trust belief and behaviour intention in virtual communities. *Behaviour & Information Technology*, 27 (2), 115–125.

Wu, M. and Lin, Y., 2001. Open source software development: an overview. *Computer*, 46 (6), 33–38.

Xu, J., *et al*., 2005. A topological analysis of the open source software development community. *In*: *Proceedings of the 38th annual Hawaii international conference on system sciences*, *HICSS '05*, 3–6 January, Big Islands, Hawaii, IEEE Computer Society Press, 188–198.

Ye, Y., *et al*., 2005. *In*: S. Koch, ed. *The co-evolution of systems and communities in free and open source software development*. *Free/open source software development*. Hershey, PA: Idea Group Inc (IGI), 59–82.

Yunwen, Y. and Kishida, K., 2003. Toward an understanding of the motivation of open source software developers. *In*: *Proceedings of the 25th international conference on software engineering, ICSE '03*, 3–10 May, Portland, OR, IEEE Computer Society Press, 419–429.

Zhang, W. and Storck, J., 2001. *Peripheral members in online communities* [online]. AMCIS, Boston, MA. Available from: http://opensource.mit.edu/papers/zhang.pdf [Accessed 1 July 2009].