Barcellini, F., Détienne, F., Burkhardt, J.M. (to appear). User and developer mediation in an Open Source Software Community: boundary spanning through cross participation in online discussions. *International Journal of Human-Computer Studies.*

User and developer mediation in an Open Source Software Community: boundary spanning through cross participation in online discussions

Flore Barcellini* [(1),(2)], Françoise Détienne [(1)], Jean-Marie Burkhardt [(3)]

** Corresponding author*

(1) INRIA Eiffel2 Group "Cognition et Coopération en Conception", Domaine de Voluceau 78153 Le Chesnay Cedex France. Phone: 0033139635552 (5522), Fax: 0033139635995

(2) Cnam, Laboratoire d'Ergonomie, Centre de Recherche sur le Travail et le Développement, 41 rue Gay-Lussac 75005 Paris France Phone: 0033144107851

(3) Université Paris 5, Laboratoire Ergonomie-Comportement-Interactions, 45 rue des Saints-Pères 75270 Paris France. Phone: 0033142862136

Flore.Barcellini@inria.fr, *Francoise.Detienne@inria.fr*, *Jean-Marie.Burkhardt@univ-paris5.fr*

Abstract. The aim of this research is to analyse how design and use are mediated in Open Source Software (OSS) design. Focusing on the Python community, our study examines a "pushed-by-users" design proposal through the discussions occurring in two mailing lists: one, user-oriented and the other, developer-oriented. To characterize the links between users and developers, we investigate the activities and references (knowledge sharing) performed by the contributors to these two mailing-lists. We found that the participation of users remains local to their communities. However, several key participants act as boundary spanners between the user and the developer communities. This emerging role is characterized by cross-participation in parallel same-topic discussions in both mailing-lists, cohesion between cross-participants, the occupation of a central position in the social network linking users and developers, as well as active, distinctive and adapted contributions. The user championing the proposal acts as a key boundary spanner coordinating the process and using explicit linking strategies. We argue that OSS design may be considered as a form of "role emerging design", i.e. design organised and pushed through emerging roles and through a balance between these roles. The OSS communities seem to provide a suitable socio-technical environment to enable such role emergence.

Keywords. Open Source Software Community, Cross-participants, boundary spanners, distributed design, role emerging design

1. Introduction

Open Source Software (OSS) design is characterized by a communitarian and a distant, asynchronous and mediated design process. This new way of designing is becoming increasingly widespread in the computer science world: there are

thousands of OSS, some of which are highly successful, like Mozilla (www.mozilla.org) or Apache (www.apache.org), and they are supported by communities of tens to hundreds of developers and millions of users (Gacek and Arief, 2004).

Mainly mediated by Internet tools (Sack et al., 2006; Mockus et al., 2002), OSS design is a paradigmatic case of distant and asynchronous collaborative design, which has thus far been less investigated than distant and synchronous, or co-located collaborative design (e.g. Détienne et al., 2004; 2005; Stempfle and Badke-Shaub, 2002; Olson et al., 1992). Studying OSS is also of particular interest to gain insights into supporting the changing nature of the software industry, which is increasingly making use of OSS design's tools and methods as it becomes more and more distributed and global (Gutwin et al., 2004).

OSS design can also be considered as a continous form of distributed participatory design: new functionalities can always be proposed and discussed at any step in the project (Gasser et al., 2003), forms of participation in OSS projects are supposed to be « open » in time and for different kinds of participants whatever their stake in the project (developers or users of the OSS). Thus, users of OSS can potentially be involved in all the phases of the design process (elicitation of needs and requirements, design and implementation), at least if they have the skills to do so. This is often the case as, in OSS, users can be highly skilled in computer science (Ducheneaut, 2005), as well as in particular application domains (e.g., education, biology, scientific computing, etc..). Moreover, the participation of users is considered to be the major strength of the OSS design process compared to proprietary ones: most bugs are detected and fixed because "there are many eyeballs looking at the problem" (Raymond, 1999).

As far as we know, there has been no research that aims at obtaining a global understanding of the OSS design process, and of the position actually occupied by users proposing new functionalities. Mediating design and use in this distant and asynchronous design setting can be of particular interest given the usability problems of OSS software, which are mainly due to the lack of Human-Computer Interaction methods in OSS communities (Twidale and Nichols, 2005).

The aim of this research is thus to understand the ways in which members of OSS communities, and especially users, participate in the design process and to identify whether or not some key participants may act as boundary spanners to link the user and the developer communities. This research is focused on a major OSS project, Python, which is an object-oriented programming language (www.python.org).

In the following sections, after a review of our theoretical framework, we set out our research questions and strategy. Then, we present the results and discuss the perspectives of this research.

2. Theoretical framework

To build our theoretical framework, we need to refer to OSS studies that establish which statuses and roles can emerge in OSS communities. We also refer to organizational science and design studies which point out that some key-participants, boundary spanners, act as mediators between users and designers. Finally, we refer to the cognitive ergonomics of collaborative design to understand what activities participants perform during a design process.

2.1. Roles, statuses and user participation in OSS communities

OSS projects are seen as online epistemic communities (Preece, 2000; Cohendet et al., 2000). Their members form a group of people connected together on the Internet with a common goal- to develop software - with the "meta"- objective of producing and constructing knowledge about the artefact they develop for the benefit of all the community. Their activities are framed by implicit and explicit rules: volunteer participation or evaluation of work by a peer-review mechanism of works for instance (e.g. Raymond, 1999).

Major OSS projects are highly hierarchical and meritocratic communities (Gacek and Arief, 2004; Mahendran, 2002). Five different statuses are generally distinguished in these projects, according to the distinctive rights and power of the participants. Some participants can modify the source code and participate directly in the design process and in decisions regarding the software:

_ the *project leader* (generally the creator of the project such as Guido Van Rossum for Python, or Linus Torvalds for Linux);
_ the core team or *administrators*, who have to maintain the code base, the documentation;
_ the *developers* or contributors who participate in the evolution of the OSS and maintain some of its parts.

Others participants are called *users*. In an OSS context, users may be highly skilled in computer science, and thus far from the classical notion of "end-users".

_ They are called *active users* if they participate in mailing-list discussions as informants for newcomers, by reporting or correcting bugs with patches, and by proposing new modules. These *active users* in a particular OSS project may be developers in another project
_ Other users are called *passive users* as they only use the software or lurk on the discussion and documentation spaces of the project (Preece et al., 2004).

It is possible to evolve between these statuses by acquiring and proving one's technical skills and ability to engage and maintain online discussions: that is to say that roles emerge and are actively constructed within the community (Ducheneaut, 2005; Mahendran, 2002). This notion of role reflects the effective

and emerging behaviour of participants. In some cases, these activities may correspond to what is expected from a particular status.

The literature on OSS clearly identifies that active users take part in the evaluation phase of design (bug reporting and patching, e.g. Ripoche and Sansonnet, 2006) and that the project leader, administrators and developers participate in the design process itself, i.e. generating and evaluating solutions and taking decisions (Barcellini et al., 2005). Open issues are still to characterize the role of users regarding the design process itself and the role of all the active participants (project leader, administrators, developers and active users) during the elicitation of the needs and requirements phase. Despite the idealistic picture that users may intervene freely in the process, we will question whether users who are neither administrators nor developers in the core Python community can really have an impact on the design choices and decisions. In particular, we will focus on the design-use mediation process: how use and design are articulated when new functionalities are proposed, solutions are generated and evaluated; and what are the links between users and developers in OSS communities.

2.2. Collaborative design activities in OSS

Studies of face-to-face design meetings, especially in software design meetings (e.g. d'Astous, et al. 2004; Herbsleb et al. 1995; Olson et al., 1992) or on mediated, distributed design teams (e.g., Olson and Olson, 2000; Détienne et al., 2004) highlight three main subsets of collaborative activities:

- G*eneration-evaluation activities* related to the process of solving and evaluating various aspects of design problems.
- C*larification-reformulation activities,* concerning the construction of common references, or common ground, within a group of co-designers.
- Group *management activities* related to issues of process. Project management activities that concern the coordination of people and resources -- e.g., the allocation and planning of tasks – are of this kind. Meeting management activities – e.g., the ordering and postponing of topics of discussion – are another example of this kind of activity.

In previous work (Barcellini et al., 2005), we have studied collaborative design activity of Python's developers engaged in a formal design process, the Python Enhancement Proposal (PEP), used to propose language evolutions. This previous study focused mainly on the discussion space and on the *python-dev* mailing-list in which most of the PEP design activity is expected to occur. Some of the design choices and alternatives, together with their rationales are set out and discussed in this space.

We showed that the analysis of quotes and their associated comments - as a link to reconstruct the argumentation process - is useful to study the design process. We found that online design discussions are focused and framed by specific

members of the project, especially the project leader, and that the proportion of the various design activities differs in online interactions from face-to-face interactions. For instance the clarification activity occurs as in face-to-face design situations but seems to be framed by the project leader and limited to specific locations in the discussion space.

The goal of the research presented in this paper is to extend these results by investigating the users' forms of participation in the design process. This implies examining design discussions in another mailing list, that is more open to users than the one we focused on in the previous part of this research.

2.3. Boundary spanners role as mediator between users and designers

In Cognitive Ergonomics and Human-Computer Interaction studies it is well established that users' and designers' representations of the artefact being designed do not match. Designers might never be able to completely satisfy users needs and requirements and incorporate into the artefact features anticipating usage constructions. Thus, different kinds of methodologies (user-centered design, participatory design) have been developed to span the gap between the users' and the designers' worlds. However, both organizational sciences and design studies point out that some key-participants may act as boundary spanners or mediators (Sonnenwald, 1996; Bansler and Havn, 2006) or as brokers between user and developer groups (Wenger, 1998).

Boundary spanners are literally persons who bridge the gap between their organization and external ones (Sarant, 2004). The role of boundary spanners is defined as "*communication or behaviour between two or more networks or groups*" (Sonnenwald, 1996). They move among different teams transferring information about the state of the project.

This role has been studied in various situations, for instance in research and development (e.g. Sarant, 2004), and in design situations (Grinter, 1999; Krasner et al., 1987). This is not a formal role but rather a role that emerges through interaction and practices. Becoming boundary spanners implies having developed skills and competences in the different fields that are spanned. Boundary spanners are well aware of all practices and have achieved legitimacy and credibility in the domains they span. Thus, they tend to occupy "high" hierarchical positions (Sarant 2004; Sonnenwald, 1996).

Boundary spanners appear to be of particular importance in collaborative design situations as they tend to produce innovative solutions to non-routine problems, i.e. to design problems (Sonnenwald, 1996). They are also key participants who can enhance coordination through informal communications (Krasner et al., 1987). They reduce the amount of information lost or miscommunicated between different phases of design development and different development teams. In non-

design situations, it has been shown that they can also protect against external pressure or represent their organization (Sarant, 2004)

As far as we know, in OSS design, and more generally in mediated, asynchronous and distant design situations, the role of boundary spanners has not yet been investigated. Our hypothesis is that design-use mediation may be supported by boundary spanners whose roles emerge from interactions between the user and developer communities of an OSS project. We plan to investigate this issue through a cognitive ergonomics approach, by focusing specifically on participants' activity, reflecting their "effective role" – as distinct from their formal roles or statuses- in the design process (Baker et al., to appear). Although the effective role is partly dependent upon the formal position of the participant in the community, i.e. the power he/she has on the artefact being designed, it is also contingent upon the participant's actions in the design process – his/her activities- and in the online discussions (Mahendran, 2002; Sonnenwald, 1996).

3. Research questions and strategy

In this section, we present our research questions and describe the research strategy we adopt to characterize forms of participation and design-use mediation in one OSS project.

3.1. Research questions

Given the limitations of the various studies outlined in the above section our objective is to investigate the following research questions:

- *What forms do the users' contributions take in the design process?*
- *Are there key participants acting as boundary spanners between the user and developer communities?*
- *What forms do the contributions of these key-participants take in terms of management and knowledge sharing activities?*

Following a cognitive ergonomics approach, these research questions will be investigated by analysing the activities that participants actually perform during the design process. We will identify interactions between users and developers through the activities occurring in the discussion space, in particular in a developer-oriented list and a user-oriented list.

3.2. Research strategy

A focus on the Python project

The Python project is dedicated to the design and use of the Python object-oriented programming language, known for its ease of use and the clarity of its syntax. We chose to focus this research on the Python project for the following reasons:

- This project has a large community of users in various application domains (web development, scientific computing, gaming, finance etc…) that might lead to innovation in different ways.
- It is a stable project with a core group of around 60 developers. Thus, the amount of interactions and lines of code remain quite reasonable compared to the Linux community and its hundreds of developers.
- The Python developers used a formal design process, the Python Enhancement Proposal (PEP), to discuss and specify new evolutions of the language. This PEP process is similar to other design processes used by distant design communities, like the Request for Comment used by the Internet Engineering Task Force, and the XEP process used by the jabber community (www.jabber.org). This formal process is close to the consensus-based decision making of Apache.

The Python galaxy and its subcommunities

The Python project can be seen as a “galaxy” in which users and developers participate (Figure 1) (Barcellini et al., 2006).

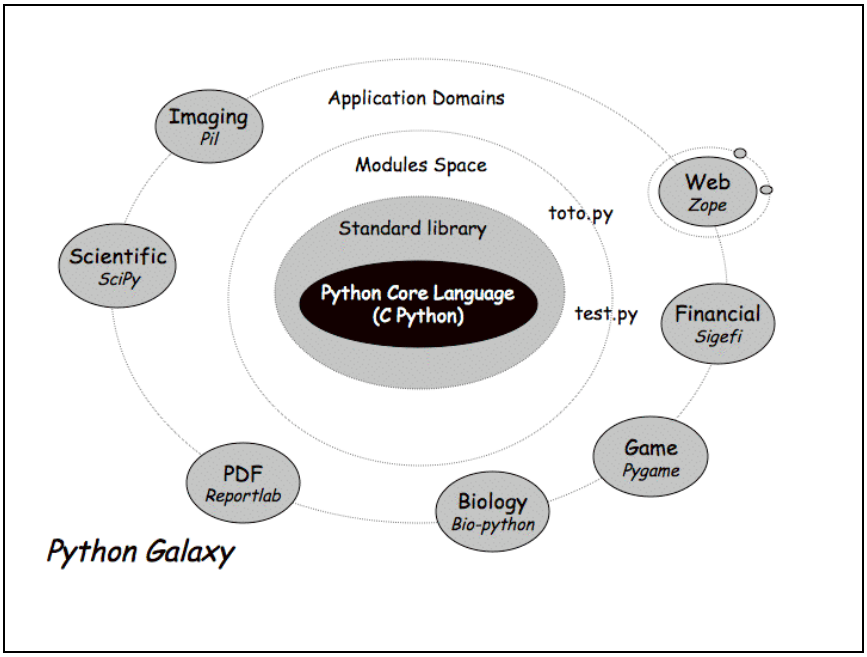

Figure 1. The *Python galaxy*

The Python core language and its standard library occupy the core of this galaxy. This space is the world of *Python's developers* (administrators and developers) that are responsible for the design of the Python programming language itself and its standard library. On the basis of this core language, *active users* of the Python programming language can develop modules based on Python (*modules space)*. *Active users* can also ask for language evolutions, using the *Python Enhancement Proposal* for major ones. This is this kind of process that we are going to follow.

Around this first core, sub-communities related to different application domains in which users can be involved are identified. Some of them are well-structured projects with their own project website and groupware. In this space we find: *(1) active users* of the Python programming language who are *developers* of application software based on Python, *(2) active users* of the application software, *(3) end-users* of these applications based on Python (web development, scientific computing, biology, and finance etc…).

This research is focused on the Python core language's space. We want to investigate how needs from various application domains are relayed and can impact the design of the Python programming language. Thus, the participants we focus on are: Python language developers and active users who may be developers or active users in other application domain subcommunities.

Cross-participation between user-oriented and developer-oriented mailing-lists

Most OSS design occurs in a discussion space composed of different mailing-lists or newsgroups (Sack et al., 2006; Barcellini et al., 2005; Mockus et al., 2002).[1]. As all discussions are publicly available and archived online, they constitute rich traces of the OSS design process.

To study the mediation between users and developers, we will analyse the interactions occurring in two mailing lists, a user-oriented mailing list, the *python-list*, and a developer-oriented mailing, the *python-dev*:

- The *python-list* mailing-list deals with general discussions and questions about Python. Posting to this usage-dedicated mailing-list is the main way an active user can participate in the Python project. Most discussions on *python-list* are about developing *with* Python, not about developing the Python interpreter itself. *Python-list* can be seen as a cross-community mailing-list bridging the gap between various users and the developers of Python.

- The *python-dev* mailing-list is for work on developing Python (fixing bugs and adding new features to Python itself). This is the heart of Python's development. Anyone may subscribe to *python-dev*, though his/her subscription will have to be approved but the list address accepts e-mail from non-members.

In these mailing lists, we will in particular investigate the activities of *cross-participants*. We define *cross-participants* as persons who take part in same-topic discussions, occurring in parallel in both mailing-lists. This notion is an extended notion of cross-posting defined as "the practice of posting the same message to multiple newsgroups" (Kollock and Smith, 1996) and as "broadcast interactions to multiple newsgroups" (Whittaker et al., 1998). Our hypothesis is that cross-participants may be key participants, or boundary spanners between user-oriented and developer-oriented communities.

4. Method

The first step of our study was to identify and select some successful "pushed-by-users" design proposals using field interviews, and to collect traces of exchanges regarding one of these proposals, in *python-dev* and *python-list*. The second step

[1] Members of the Python community can interact in these different "online" spaces, but also in physical interaction spaces where they meet face-to-face, such as at annual international conferences (EuroPython, RMLL, PyCon...)

was to identify the participants and cross-participants involved and their statuses. The final step was to analyze interactions, design and boundary spanning activities, in particular those of users and cross-participants.

4.1. Identification and collection of data

Field interviews to identify "pushed-by-users" design proposal

We interviewed 13 participants in the Python project: 11 active users of the Python language, the project leader and one developer. These people were selected according to two criteria: their involvement in the Python community and / or their participation in Python's mailing-lists. Six of these 11 active users were working in different application domains: two in computer science, two in biology, and two in chemistry/nuclear institutions. Five were working in firms providing web services or scientific computing around Python.

As the Python community uses the *Python Enhancement Proposal* (PEP) process to formalize a new design proposal, interviewees were asked which PEPs had been most significant for them, or for some other users in the community according to them.

The interviews were then transcribed and two "pushed-by-users" design proposals transformed in PEP (PEP 327 and PEP 308), i.e. successful design proposals, were outlined among the 161 PEPs of the Python project.

Data Collection

We selected the design proposal formalized in PEP 327 on the basis of several criteria. The champion of this PEP, i.e the person who originally proposed it, was a user. The PEP is related to one end-user application domain of Python (financial), whereas PEP 308 deals with programming in Python[2]. Finally, we were able to collect most (as far as we know) of the traces related to this PEP on the two mailing-lists.

As all discussions are publicly available and archived online, the data was gathered by searching manually from the *python-list* and the *python-dev* mailing-lists, for the keywords: decimal, money, currency, PEP 327 and the name of the *user-champion* (the user who was actually proposing the PEP and championing it). The search was performed from the first message posted by the user champion in October 2003 to May 2006. Each message, which was returned was read by the first author of this paper in order to ensure that it was indeed a message dealing with the design issue we are interested in, as in O'Shea and Exton (2005).

[2] Adding conditional expressions to Python as in other programming languages

In the user-champion's weblog, we also found five articles referring to Python conferences where issues about the design proposal were discussed.

The *PEP 327 corpus* is composed of 52 discussions in *python-list* and *python-dev* from the 17th October 2003 to the 23th May 2006, enriched by these five weblog articles.

Participation and identification of cross-participants in the PEP 327 design process

To select relevant discussions, we need to have an overview of the temporal organization of the 52 discussions in the two mailing-lists. To characterize the temporal organization we ordered the discussions according to their beginning and ending date in parallel in the two mailing-lists (Figure 2). The *python-dev* and the *python-list* discussions are represented in parallel. Each discussion is represented by a symbol. Conferences in which there were some interactions relating to the design process are represented by a grey vertical line. Cross-participation between parallel same-topic discussions in *python-list* and *python-dev* is labelled using dotted vertical lines.

This representation helps us to identify five parallel same-topic discussions (Table 1; in black in Figure 2) between the two mailing-lists, in which cross-participants are present. For a more global description of this design process see Barcellini et al. (2006).

These discussions dealt with the reformulation of the initial proposal of the future user-champion, i.e. the introduction of a *Money Data Type* in Python: it appears that creating a money data type requires work on the decimal type of Python first of all and the proposal evolved to PEP 327 on *decimal data type.*[3]

[3] We excluded some messages (PEP 327 news and Money Module) when there was only one message posted on python-list not followed by any answer, i.e. no discussion emerged.

Barcellini, F., Détienne, F., Burkhardt, J.M. (to appear). User and developer mediation in an Open Source Software Community: boundary spanning through cross participation in online discussions. *International Journal of Human-Computer Studies.*

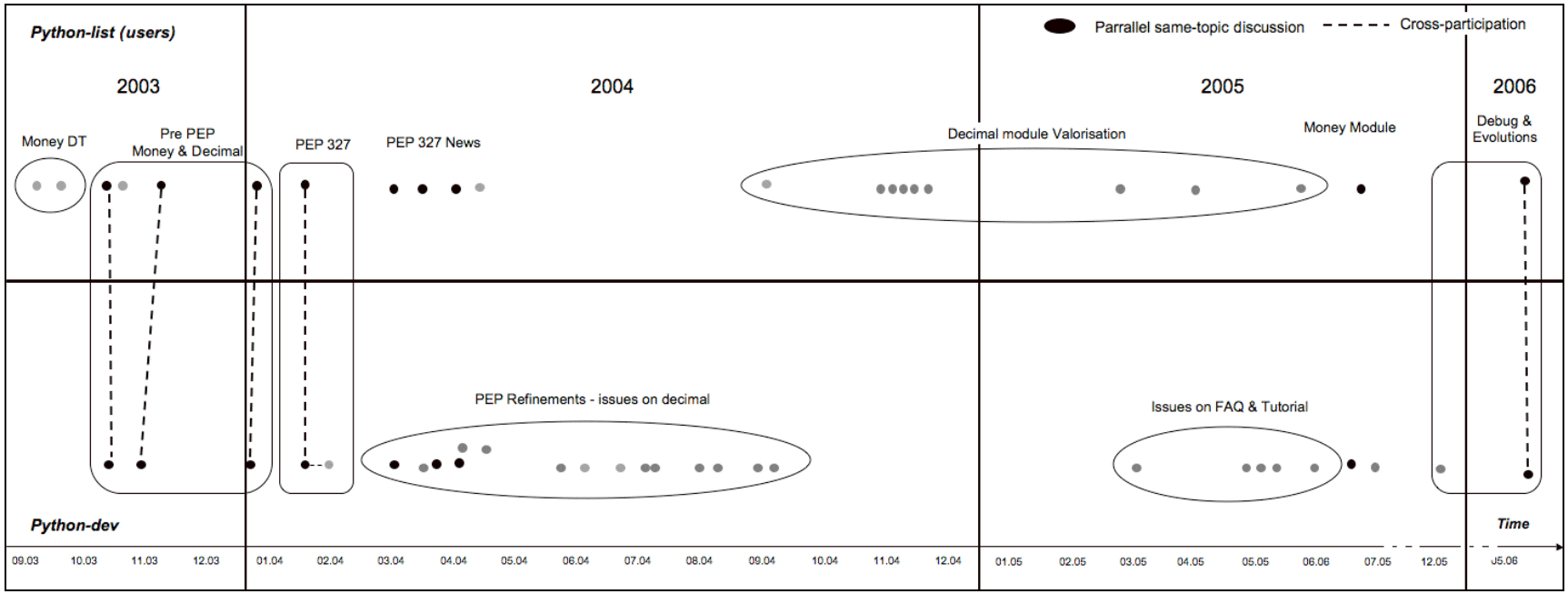

Figure 2. Temporal organization of discussions related to PEP 327 design process

Table 1. Date, number of messages and participants in our corpus of five parallel discussions on the *python-dev* and the *python-list*.

Discussions	***Date***		**Messages**		**Participants**	
	Py-list	*Py-dev*	*Py-list*	*Py-dev*	*Py-list*	*Py-dev*
PrePEPMoney	17 to 22 oct. 03	17 to 22 oct. 03	51	23	7	8
PrePEPDecimal	31 oct. to 9 nov 03	31 oct. 03	77	2	14	2
PrePEPDecimalV0.2	5 to 6 jan. 04	5 jan. 04	4	2	3	2
PEP327	31 jan. to 10 feb. 04	31 jan. to 5 feb. 04	34	16	14	7
Decimal-Expo	19 to 23 may 06	19 to 23 may 06	7	3	4	2
Total			*173*	*46*	*42*	*21*

As some participants post messages in several discussions, there are only 28 different participants in *python-list* and 13 different participants in *python-dev*.

In this corpus, five people were identified as cross participating in parallel same-topic discussions: the user-champion (he was not formally defined as a developer at the beginning of the process and was the project leader of a financial project), one administrator (CP-A1) who is a recognized expert in the decimal domain; two developers (CP-D1 and CP-D2), one of whom had already worked on a decimal module; and one user4 (CP-U1).

4.2. Analysis of a "pushed by users" PEP design process

A quotation analysis to highlight participants' interactions

Quotation, i.e. integrating parts of previous messages in another one, is a compensatory linking strategy used by participants to maintain the context in online discussions (Herring, 1999). In a previous paper (Barcellini et al., 2005), we showed that the organization of messages according to the quoting link is useful to reconstruct the thematic coherence of online discussions and to understand the interactions between participants in terms of verbal turns. In the following, we outline interactions between participants using this quotation analysis. Blocks of quotation are identified in each message by the "greater than" symbol ">". Then, we search manually in the corpus which message and which author this quote comes from to obtain a table of "who is quoting whom".

[4] We employ the term "users" for those who are not clearly identified as administrators or developers on the project webpage.

Analysis of activities and references in mailing-lists

The coding of activities is inspired by previous studies on collaborative software design activities (d'Astous et al., 2004; Détienne et al., 2004; Olson et al., 1992; Stempfle and Badke-Schaub, 2002) and by the coding scheme developed in our previous paper (Barcellini et al., 2005), which we have extended. We code collaborative design activities, reflecting the functions of the turns in the design discussion (e.g. making a proposal, a reformulation, an argumentation…), the reference categories reflecting the knowledge that is shared in the discussion (e.g. knowledge about use).

The corpus obtained is segmented into units, corresponding to comments or sub-units of comments. All messages contain sequences of quotes and comments. On the basis of our previous study (Barcellini et al., 2005), we consider that a comment corresponds to an individual turn in an online discussion. Each unit is categorized using the coding scheme composed of two types of categories: activities and references. In this paper, we will only describe and use: group management (coordination, synthesis, decision), social relationship activities (Table 2), and referencing activities (Table 3) to highlight the roles of boundary spanners.

Table 2. Definition and examples of boundary spanners related activities used in our coding scheme

Categories of activity	**Activities**	**Description**	**Examples**
Coordination	*Coordination*	Tasks management, announce of "To do" work	*I suggest that you register with the PEP editor right away to get a PEP number*
	Synthesis	Recapitulation of what has been discussed, often followed by an enumeration	*Okay, here's what we have so far : Enum (…)*
	Decision	Explicit decision (only by the Project Leader)	*Not observed in the corpus*
Social Relationship	*Social relationship*	acknowledgement, recognition of previous work, work done, explicit Help Proposal	*The code of the Money class is based in large part on the code of yy' FixedPoint: thank you for your (very) valuable ideas.*

Table 3. Definition and examples of references used by participants

Category of reference	References	Description	Examples
Application domain	*Decimal Domain*	Information related to the decimal standard	*The Decimal data type (…) must comply the decimal arithmetic ANSI standard X3.274-1996.*
	Financial Domain	Information related to law and rules of the financial domain	*Found it -- article 5 of the Council Regulation which established the Euro" a few years ago is titled "Rounding" and specifies (…)*
Computer science	*Programming Rules*	Evocation of rules and knowledge about coding	*You can control something based on the type of a destination only via augmented assignment.*
	Python rules/project	Reference to the specificities and the rules of the programming language or the project	*Also, making mutable numbers (…) is not very Pythonic.* *As the Zen of Python puts it (…)*
	Other Programming languages	Comparison to other programming languages (Java, C++, Cobol…)	*For example, for multiplication Java's BigDecimal class (…)*
Examples and code	*Examples and code*	Reference to an application example for proposals or requirements; proposals of code	*because for example with a precision of 9, Decimal(35) + 1.2 is OK but Decimal(35) + 1.1 raises an error)*
Usage-Experience	*Personal Experience*	Reference to a personal experience at work, in the financial domain, in implementation	*Having had to deal with monetary calculations , laws and accountant. (…).* *The toy Decimal I'm playing with*
	End-User	Evocation of end-users	*it IS what accountants *DO* -- and if one writes accounting software one should really play by their rules*
	Programming-user	Evocation of programming-users	*I think that expert programmers will have little difficulty parameterizing their operations.*
	Generic users	Evocation of users without any precision on the types of these users (end-users, programming-users)	*Naive users will always _believe_ that they're getting "good" precision*
	Usage-usability	Usability issues	*The resulting decimal type, however, may not be highly usable for some kinds of monetary computations.*
	Scenario	Simulation in "real world" conditions/constraints	*Would it be acceptable to carry around the "exact" amount and then say (..) to a customer who owns 1000 units of the stock, that he must pay a charge of 6.168209 dollars or euros*
Explicit linking	*Person*	Explicit references to the name of stakeholders.	*This wouldn't have been possible without the help from yy, xx (…)*
	Same/Others discussions	Reference to a post in the same discussion or to a post in a previous discussion or in another mailing-list	*I've followed up to xx's similar post on Python-dev* *This subject has been already discussed in c.l.p (python-list)*

This content analysis was completed by an interview with the user-champion of this PEP.

5. Results

5.1. Global participation distribution

Table 4 sums up the distribution of references in the two lists according to the status of the participants and their cross participation role.

Concerning the referencing activities, CPs are the greatest contributors as they are responsible for 54% (297/551) of references in *python-list* and more than 80% (104/130) in *python-dev*.

There is a strong relationship between the level of references and the list (V^2 Cramer=0.29): the UC, AD and the PL tend to participate by more references in *python-dev* than in *python-list*, whereas users tend to participate by more references in *python-list* (they represent only 2% (3/130) of references in *python-dev*). The project leader (PL) does not participate at all in *python-list* and participates less (4.3%) in *python-dev* than in other discussions studied in our previous paper (Barcellini et al., 2005). Indeed, the PL declared in a post that he was not an expert in this area. On the other hand, the administrator who cross-participates is a well-known expert in decimal. This result confirms the findings of our previous paper in which we observed that the effective roles of administrators and the PL tend to be complementary.

Table 4. Distribution of references used by participants in *python-dev* and *python-list*

Status	**Distribution of References**	
	Python-list	*Python-dev*
PL	0	7 (5%)
AD	3 (1%)	16 (12%)
Users	251 (45%)	3 (2%)
UC	67 (12%)	59 (46%)
CP *(without UC)*	230 (42%)	45 (35%)
Total	*551(39%)*	*130 (40%)*

5.2. Social network of users and developers communities

The attraction graph[5] below (Figure 3) represents the interactions between participants, showing who tends to quote whom in the two mailing-lists. There is an intermediary relationship (V^2 Cramer=0.07) between the status of the participant who is quoting and the status of the participant who is being quoted.

Figure 3 displays a kind of social network in which the links are based on quoting rather than reply-to links. Firstly, it shows that the cross-participants as a whole tend to quote each other: the UC tends to quote and be quoted by the CPs. This result is coherent with the interview with the user-champion who stated that he received technical, social and discursive help or support from other CPs, in particular from three of them: CP-D1, CP-D2 and CP-A1. Secondly, this figure outlines the fact that all the cross-participants (both CPs and the UC) are the relay between the user community (U) and the developer community (PL and AD). They are central in the social network linking these two communities. On the user side, the cross-participants (CPs only) tend to quote and be quoted by the users (U) in the two mailing-lists. On the developer side, the UC who is a specific cross-participant tends to quote and be quoted by the PL and the AD in the *Python-dev* list.

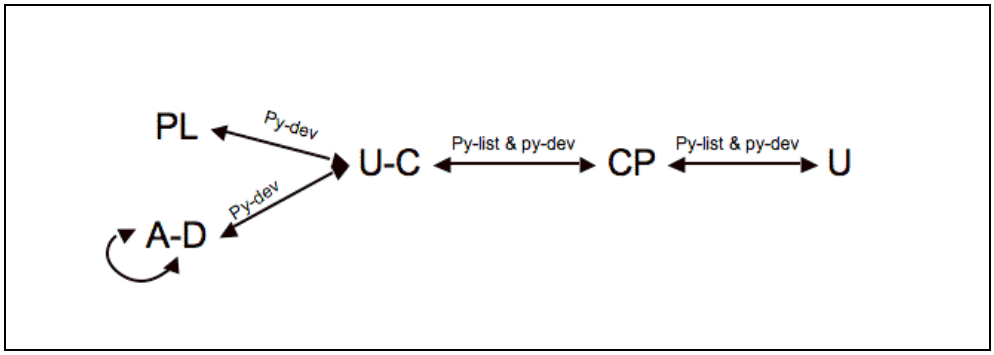

Figure 3. Attraction graph highlighting quoting between participants

5.3. Global activities and references distribution

Group management and social relationship activities are more important in *python-dev* (resp. 55/ 324, 17% of activities; 20/324, 6.2% of activities) than in *python-list* (resp. 76/1415, 5.4%; 32/1415, 2.3%). As analysed discussions took place at the beginning of the PEP process, participants in *python-list* are mostly evaluating and debating the requirements proposed by the PrePEPs and PEP; whereas *python-dev* is dedicated to the coordination of the project and the

[5] This graph is based on the relative deviation (RD) analysis. By convention, we represent only the attractions between variables (here the participants) > 0.2.

recognition by the user-champion of both previous work and work in *python-list* (social relationship).

Table 5 sums up the distribution of types of references in *python-list* and *python-dev*. Globally, *computer science* references (190/681, 28%) and *usage / personal experience* (151/681, 22%) are the most important references used by participants. Although the relationship between the type of references and the list is low (V^2 Cramer=0.01), the analysis of local RD[6] reveals that *usage/personal experience* and *computer science* references tend to occur in *python-list* and that *explicit linking* strategies tend to occur in *python-dev*.

Table 5. Distribution of references used by participants in *python-dev* and *python-list*

References	*Python-list*	*Python-dev*	**Total**
Computer science	162 (29%)	28 (21%)	190 (28%)
Usage / personal experience	128 (23%)	23 (18%)	151 (22%)
Example and code	113 (22%)	32 (25%)	145 (21%)
Explicit linking strategies	74 (13%)	28 (21%)	102 (15%)
Financial and decimal domains	74 (13%)	19 (15%)	93 (14%)
Total	*551 (81%)*	*130 (19%)*	*681*

Table 6 gives the distribution of references inside the *usage / personal experience* category. The most frequent references in this category are *scenario* (40/109, 37%) followed by *programming-users* (20/109, 18%) and *end-users* (20/109, 18%). Analysis of RD (V^2 Cramer=0.06 meaning an intermediary relationship) reveals that reference to *programming users* and *usage/usability* issues tend to occur more in *python-dev* and that reference to *end-users*, *generic users* and *scenario* tend to occur in *python-list*.

Table 6. Distribution of usage/personal experience references by participants in *python-dev* and *python-list*

References	*Python-list*	*Python-dev*	**Total**
Scenario	36 (39%)	4 (24%)	40 (37%)
Programming users	15 (16%)	5 (29%)	20 (18%)
End-users	19 (21%)	1 (6%)	20 (18%)
Usability/usage	12 (13%)	6 (35%)	18 (17%)
Personal experience	29 (24%)	5 (23%)	34 (14%)

[6] RDs measure the association between two nominal variables. They are calculated on the basis of a comparison between observed and expected frequencies (i.e. those that would have been obtained if there had been no association between the two variables). There is attraction when the RD is positive, and repulsion – when it is negative. By convention, we reported only attractions with values >.20.

Generic users	10 (11%)	1 (6%)	11 (10%)
Total	*92 (85%)*	*17 (15%)*	*109*

5.4. Emerging roles through effective coordination and referencing activities

We investigate whether there is a link between cross participation, status and effective activities/references of the participants. First, we make a quantitative analysis of referencing activity according to cross participation and status. Then we analyse the attraction between types of participants and activities.

As concerns cross-participants, we clarify the nature of the references they provide in both lists:

- In *python-list*, CPs provide 54% (40/74) of *decimal-financial* references, 46% (74/162) of *computer science* references, 42% (31/74) of *explicit linking* references and 40% (52/128) of *usage and personal experience* references. The user-champion alone (UC) provides 26% (29/113) of *code and examples*, and 22% (16/74) of *explicit linking*.
- In *python-dev*, CPs provide 68% (13/19) of *decimal-financial* references and 64% (18/28) of *computer science* references. The UC alone provides 78% (25/32) of *code and examples* and 61% (17/28) of *explicit linking*.

Concerning the users, they mainly provide references in the user-oriented list, the *python-list* (only 2% of references in *python-dev*): 52% (67/128) of *usage and personal experience* references, 49% (80/162) of *computer science* references, and 43% (49/113) of *code and examples*.

Figure 4 and Figure 5 complete this global description. They display attraction graphs between the participants and the activities/references that the participants tend to perform the most. Figure 4 represents the attraction graph between participants, coordination activities and references in the two mailing-lists. There is an intermediate relationship between participants and the types of references and activities they perform in *python-dev* (V^2 Cramer=0.12 for references and 0.05 for activities) and a low relationship (V^2 Cramer=0.03 for references and 0.02 for activities) in *python-list*.

Cross-participants (CPs) tend to provide more references about the application domain of the PEP (financial and decimal domains) than other participants in both mailing-lists. In *python-dev*, they also tend to provide knowledge about the computer science domain, especially about programming and Python community rules. These referencing activities are one way to link user-oriented and developer-oriented communities.

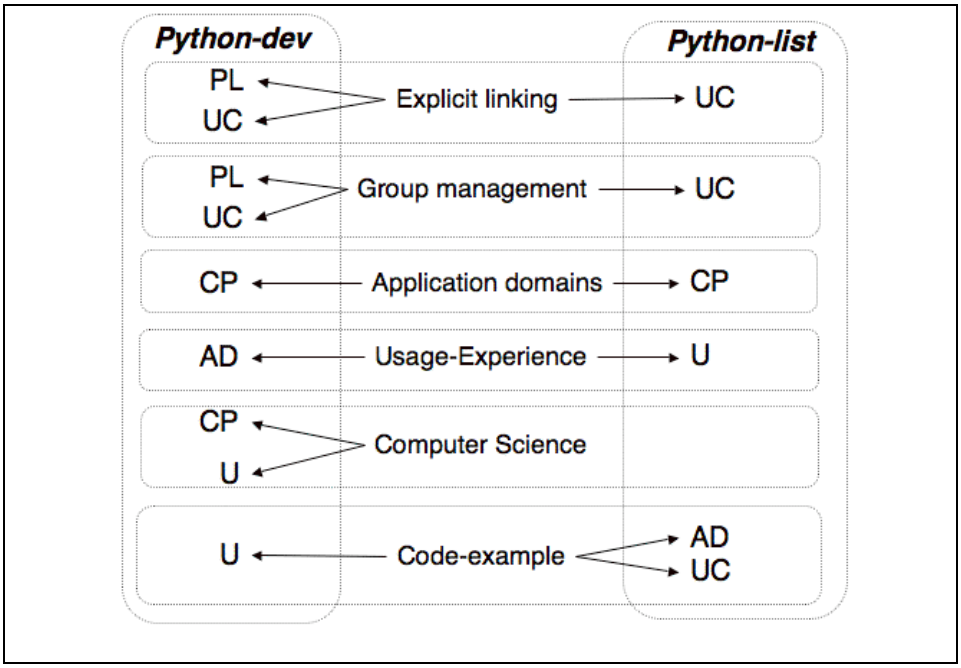

Figure 4. Attractions graph of the relationship between participants, activities and references

Regarding the user-champion (UC) alone, he tends to manage the group more than other participants in both lists. He also tends to enhance social relationship by explicit linking strategies to other people's work, and to participation in other discussions. In this way, he applies the social rules of OSS communities: the recognition of other works. He does so by crossing the boundaries between the two communities. In the activities of both management and explicit linking, the UC is supported by the project leader (PL) in the developer-oriented list.

Interestingly, we note asymmetry between users and Administrators-Developers (AD): ADs tend to refer to usage in *python-dev*, whereas users (U) tend to talk more about usage in *python-list*.

Figure 5 focuses on *usage/personal experience* references: it displays the attraction graph between participants and the *usage/personal experience* references subcategories they tend to use the most. There is a strong relationship between the participants and the types of usage references in *python-dev* (V^2 Cramer=0.22) and an intermediate relationship in *python-list* (V^2 Cramer=0.05).

Figure 5 shows that the cross-participants tend to refer to the end-users in the *Python-dev* list and to the programming users in the *python-list*. They tend to adapt their referencing activity according to the target community they are addressing: transferring referencing about end-users to the developer community, transferring references about programming users to the user community.

The UC and the PL tend to provide usage-usability references in a complementary way, respectively in p*ython-dev* and the *python-list*.

The users tend to provide more personal experience references and end-user references than others in p*ython-list*.

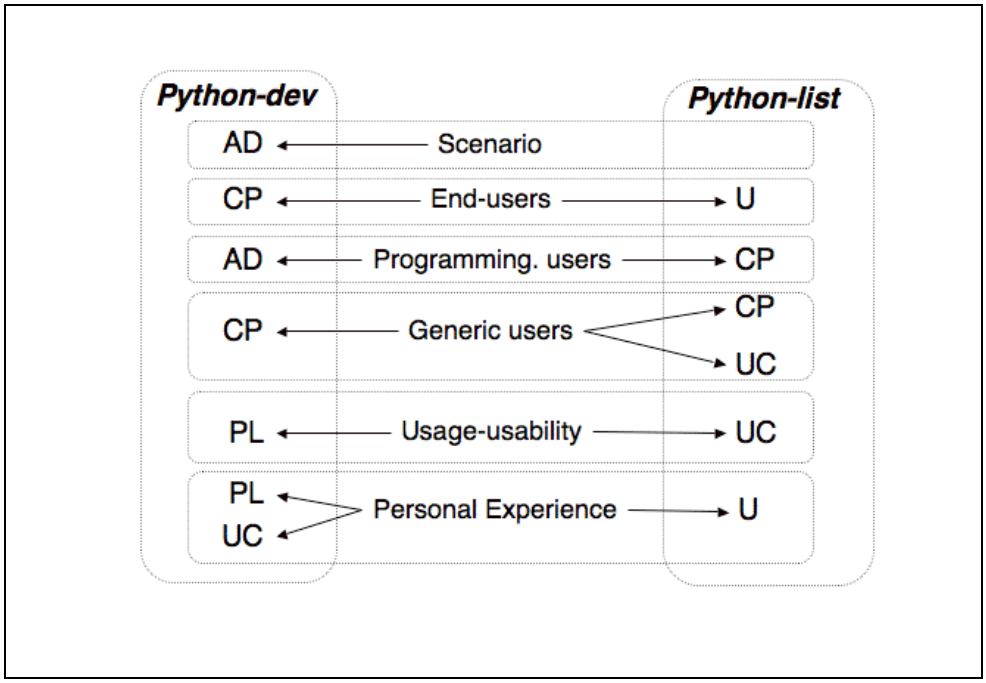

Figure 5. Attractions graph of the relationship between participants *usage/personal experience* references

6. Discussion and further work

6.1. Users' contribution

We found that users mostly tend to participate in the user-oriented list, *python-list*. Here, they provide references mostly on *usage and personal experience*, *computer science*, and *code and examples*. They also tend to provide more *personal experience* and *end-user* references than others in *python-list*. Even if the users' contribution seems important in order to specify usage needs, their participation remains local to the user-oriented community and does not guarantee that these needs will be taken into account in the actual design.

6.2. Cross-participation and boundary spanning role

We found that cross-participants perform an emerging role of boundary spanners, which guarantees that usage is linked to design and that the boundary between the user community and the developer community is crossed. This emerging role is characterised as follows:

- Cross-participation in both communities: the cross-participants post messages in same topic discussions occurring in parallel, in mailing lists directed toward users or developers.
- Cohesion between cross-participants: the cross-participants tend to quote each other as is shown in a social network analysis, based on quotation links. They tend to support each other: in our study, the user-champion received some technical, social and discursive support from other cross-participants.
- Central position of cross-participants: the cross-participants are central in the social network linking the user community and the developer community. On the user side, the cross-participants tend to quote and be quoted by the users. On the developers side, the user-champion, a specific cross-participant, tends to quote and be quoted by the administrators (including the project leader) and the developers of Python.
- Active contribution: the cross-participants are the main contributors and the main providers of references in both lists. Moreover, for the whole corpus (the 52 discussions), we found that the cross-participants are active throughout the design process.
- Distinctive contribution: the cross-participants provide and share knowledge about both the user-oriented application domain and the developer-oriented programming domain. In this way they cross the boundaries between the two communities. In our analysis, we found that they provide about half of the references to the application domain (financial and decimal) and to the computer science domain.
- Adapted contribution: the cross-participants tend to adapt their referencing activity according to the target community they address: transferring references about end-users to the developer community; transferring references about programming-users to the user community. Furthermore they are complementary to users as they are the main providers of *usage and personal experience* references in the developer-oriented list whereas users are the main providers of this type of knowledge in the user-oriented list. Thus, they also maintain a strong focus on usage in the developer community.

6.3. Key roles of the user-champion and the project leader

The user-champion tends to enhance social relationships by explicit linking strategies, to other people's work, and by participating in other discussions. He refers to the previous work of other or other discussions through explicit linking strategies (Herring, 1999), and thus enhances harmonious social relationships, respecting the rules of the community, such as recognition of work. By doing so, he may prevent conflicts by crossing the boundary between the two communities. He is also a coordination agent, managing the group more than other participants in both lists, summarizing and posting news about the state of the PEP design

process in the user-oriented list. At the end of the process, he becomes an official developer and is responsible for presenting the new module at the Python Conference. In both activities of management and explicit linking, the user-champion is supported by the project leader in the developer-oriented list.

6.4. Role emerging design

Our study provides insights into how design and use are articulated in OSS design, a distant and asynchronous community-based design process. According to our results, OSS design does not seem to be participatory in the strict sense of the definition, i.e. user involvement in "design" activities. Even if users of OSS may potentially be involved in all the phases of the OSS design process (elicitation of needs and requirements, design and implementation), we found that their participation remains mostly local to the user community in the PEP process we analysed.

We found that the design-use mediation is supported rather by a number of key participants who act as boundary spanners and who are not necessarily users themselves: two of them were users but the three others were administrators and developers.

In the literature, it is highlighted that these roles emerge in collective activities based on the technical, discursive skills and interest of the participants. In OSS communities, the participants are highly skilled, often in a different field of expertise. The status of user or developer becomes completely relative to the context in which their skills can be expressed. Our position is that the context of OSS design can enable these skills to be expressed as needed, and in this way, enables "*role emerging design*", i.e. design organised and pushed through emerging roles and through a balance between these roles. The Open Source communities, and more general online communities, seem to provide a socio-technical environment enabling role emergence and role complementarity, thereby constituting "enabling environments" (Falzon, 2005) for the participants.

6.5. Further work

To complement this research, we plan to carry out in-depth interviews with cross-participants and some users to gain a better understanding of their positions in this design process, and to make a deeper analysis of boundary spanning roles in distributed design communities.

Moreover, the user-champion received some technical, social and discursive (he was a non native English speaker) support by cross-participants to get his idea accepted; he evolved and enhanced his competences during this design process. We plan to investigate a PEP on the same theme, that was proposed and rejected several months before in order to highlight both the necessary conditions and the

barriers to get a "pushed-by-users" PEP accepted, and more generally to enhance role emergence in this kind of distributed design community.

We found that using a formal criterion such as cross-participation is a powerful means to identify boundary spanners in parallel mailing-lists. In order to further investigate the notion of "role emerging design", the identification of cross-participants, as well as social network based on quoting, could be automated and applied to other online communities.

7. Acknowledgments

The authors wish to thank the reviewers for their helpful comments and suggestions, and all the members of the Python community. This research is funded by the French National Research Department, via the Cnam and INRIA.

8. References

d'Astous, P., Détienne, F., Visser, W., and Robillard, P. N. (2004). Changing our view on design evaluation meetings methodology: a study of software technical evaluation meetings. *Design Studies, 25*, 625-655.

Barcellini, F, Détienne, F., Burkhardt, J.M., and Sack, W. (2005). Thematic coherence and quotation practices in OSS design-oriented online discussions. In K. Schmidt, M. Pendergast, M. Ackerman, et G. Mark (Eds.) *Proceedings of the 2005 International ACM SIGGROUP* (pp 177-186). New York, USA: ACM Press.

Barcellini, F., Détienne, F., Burkhardt, J.M. (2006). Users' participation to the design process in a Free Open Source Software Online Community ». In P.Romero, J.Good, S.A Bryant & E. Chapparo (Eds.) *Proceedings of the 18th workshop Psychology of Programming PPIG'06*, pp-99-114.

Baker, M., Détienne, F., Lund, K., and Séjourné, A. (to appear). Analyse épistémique et argumentative de la conception collective en architecture: étude des profils interactifs. In F. Détienne et V. Traverso (Eds) *Méthodologies d'analyse de situations coopératives de conception: corpus MOSAÏC*. Nancy: PUN.

Bansler, J.P. and Havn, E. (2006). Sensemaking in Technology-Use Mediation: Adapting Groupware Technology in Organizations. *Journal of Computer Supported Cooperative Work, 15 (1)*, 55–91.

Cohendet, P., Creplet, F. and Dupouët, O (2000). Organizational innovation, communities of practice and epistemic communities: the case of Linux. In A Kirman & JB

Zimmermann (Eds) *Economics with Heterogeneous Interacting agents*. The Netherlands: Springer.

Détienne, F., Boujut, J-F., & Hohmann, B. (2004) Characterization of Collaborative Design and Interaction Management Activities in a Distant Engineering Design Situation. In F. Darses, R.. Dieng, C. Simone, M. Zaklad (Eds) *Cooperative Systems design*. IOS Press, 83- 98.

Détienne, F., Martin, G., and Lavigne, E. (2005). Viewpoints in co-design: A field study in concurrent engineering. *Design Studies*, *26 (3),* 215–241.

Ducheneaut, N. (2005). Socialization in an Open Source Software Community: A Socio-Technical Analysis. *Journal of Computer Supported Collaborative Work, 14*, 323-368.

Falzon, P. (2005). Ergonomics, knowledge developpement and the design of enabling environments. In Conference on Humanizing Work and Work Environment, Guwahati, Inde.

Gacek, C., and Arief, B. (2004). The Many Meanings of Open Source. *IEEE Software, 21*, 34-40.

Gasser, L., Scacchi, W., Ripoche, G., and Penne, B. (2003). *Understanding Continuous Design in F/OSS project*. Communication at ICSSEA-03, Paris, France, December 2003.

Grinter, R., 1999. Systems Architecture: Product Designing and Social Engineering, San Francisco, CA, USA

Gutwin, C., Penner, R., and Schneider, K. Group Awareness in Distributed Software Development. In Proceedings of CSCW 2004 (pp72-81). New York, USA : ACM press.

Herbsleb, J., Klein, H., Olson, G. M., Brunner, H., Olson, J. S., and Harding, J. (1995). Object-oriented analysis and design in software project teams. *Human-Computer Interaction, 10*, 249-292.

Herring, S. (1999). Interactional Coherence in CMC. In Proceedings of the *32nd Hawaii Conference on system sciences* (13 p.). Maui Island, Hawaï, USA, 5-8 January 1999.

Kollock, P., and Smith, M. (1996). Managing the Virtual Commons. In S.Herring (Ed.) *Computer-Mediated Communication: Linguistic, Social, and Cross-Cultural Perspectives* (Pp. 109-128), Amsterdam, The Netherlands: John Benjamins.

Krasner, H., Curtis, B., Iscoe, N. (1987). Communication breakdowns and boundary spanning activities on large programming projects. In G. Olson, S. Sheppard, and E.

Soloway, E. (Eds.) *Empirical Studies of programmers:*Second Workshop, Ablex, pp. 47-64.

Mahendran, D. (2002). *Serpents and Primitives: An ethnographic excursion into an Open Source community*. Master's Thesis, School of Information Management and Systems, University of California at Berkeley.

Mockus, A., Fielding, R. T., and Herbsleb, J. (2002). Two Case Studies of Open Source Software Development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology, 11(3)*, 309–346.

Olson, G.M., Olson, J.S., Carter, M. R., and Storrosten, M. (1992). Small Group Design Meetings: An Analysis of Collaboration. *Human-Computer Interaction, 7*, 347-374.

Olson, G. M., and Olson, J. S. (2000). Distance Matters. *Human-Computer Interaction, 15*, 139-178.

Preece, J. (2000) *Online Communities: Designing Usability and Supporting Socialbilty*. New York, USA : John Wiley and sons.

Preece, J., Nonnecke, B., Andrews, D. (2004). The top five reasons for lurking: improving community experience for everyone. *Computer in Human behavior,20*, 201-223.

Raymond, E. S. (1999) *The cathedral and the bazaar* [web page] **http://www.tuxedo.org/esr/writings/cathedral-bazaar/** [June 20 2005].

Ripoche, G. and Sansonnet, J.-P. (2006). Experiences in Automating the Analysis of Linguistic Interactions for the Study of Distributed Collectives. *JCSCW, 15(2-3),* 149-183.

Sack, W, Détienne F, Burkhardt, J.M., Barcellini F, Ducheneaut, N, and Mahendran D. (2006). A Methodological Framework for Socio-Cognitive Analyses of Collaborative Design of Open Source Software. *Journal of Computer Supported Collaborative Work, 15 (2-3)*, 229-250

Sarant, S.A. (2004). *The role of* organizational *boundary spanners in industry/university collaborative relationship*. Doctor of Philosophy in Psychology Dissertation Thesis. North Carolina State University, 2004.

o'Shea, P., Exton, P. (2005). The Role of source code within program summaries describing maintenance activities. In P. Romero, J. Good, E. Acosta Chaparro and S. Bryant (Eds) *Proceedings of PPIG 17*, pp160-172.

Sonnenwald, D.H. (1996). Communication role that support collaboration during the design process. In Design Studies, 17, 277-301.

Stempfle, J., and Badke-Schaub, P. (2002). Thinking in design teams - an analysis of team communication. *Design Studies, 23*, 473–496.

Twidale, M.B., Nichols, D.M. (2005). Exploring usability discussions in Open Source development. In Proceedings of HICSS '05, pp198c- 198c.

Wenger, E. (1998). Communities of practice : learning, meaning and identity. New York, USA : Cambridge University Press.

Whittaker, S., Terveen, L., Hill,W., and Cherny L. (1998). The dynamics of mass interaction. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, p257-264.