A socio-cognitive analysis of online design discussions in an Open Source Software community

Flore Barcellini a,b,*, Françoise Détienne a, Jean-Marie Burkhardt a,c, Warren Sack d

a INRIA, Eiffel Team, “Cognition and Cooperation in Design”, Domaine de Voluceau, 78153 Le Chesnay Cedex, France
b Cnam, Ergonomics Lab, Research Center on Work and Development, 41 rue Gay-Lussac, 75005 Paris, France
c Université Paris 5, Laboratoire Ergonomics-Behaviors-Interactions, 45 rue des Saints-Pères, 75270 Paris, France
d University of California, Film and Digital Media Department, 1156 High Street, Santa Cruz, CA 95064, USA

Received 6 October 2006; received in revised form 8 October 2007; accepted 23 October 2007
Available online 17 November 2007

Abstract

This paper is an analysis of online discussions in an Open Source Software (OSS) design community, the Python project. Developers of Python are geographically distributed and work online asynchronously. The objective of our study is to understand and to model the dynamics of the OSS design process that takes place in mailing list exchanges. We develop a method to study distant and asynchronous collaborative design activity based on an analysis of quoting practices. We analyze and visualize three aspects of the online dynamics: social, thematic temporal, and design. We show that roles emerge during discussions according to the involvement and the position of the participants in the discussions and how they influence participation in the design discussions. In our analysis of the thematic temporal dynamics of discussion, we examine how themes of discussion emerge, diverge, and are refined over time. To understand the design dynamics, we perform a content analysis of messages exchanged between developers to reveal how the online discussions reflect the “work flow” of the project: it provides us with a picture of the collaborative design process in the OSS community. These combined results clarify how knowledge and artefacts are elaborated in this epistemic, exploration-oriented, OSS community. Finally, we outline the need to automate of our method to extend our results. The proposed automation could have implications for both researchers and participants in OSS communities.

© 2007 Elsevier B.V. All rights reserved.

Keywords: Open Source Software community; Distributed and asynchronous design; Online discussions; Quoting; Role

1. Background

The objectives of our work are twofold. First, we aim at developing a methodology to describe and understand OSS design process, which is an interesting case of distributed, mediated, and asynchronous collaborative design work. Second, on the basis of this methodology, we aim at analyzing the dynamics of the OSS design in a particular OSS community, the Python community.

Our work is focused on the discussion space (mailing lists, forums) (Sack et al., 2006), in which most of the design activity takes place (Mockus et al., 2002). We analyze two online discussions between Python developers engaged in a specific design process used to define and to specify revisions and improvements to the Python language. This process is called the Python Enhancement Proposal or PEP.

We develop a methodology, combining: (1) structural analyses based on the analysis of quoting in mailing list exchanges seen as a mean to maintain context and coherence in online discussions (Eklundh and Macdonald, 1994) and (2) content analyses of the message based on a coding scheme of collaborative design activities.
developed in our team (Darses et al., 2001). We have attempted to formalize our analyses in order to make it possible, in future work, to automate the procedures. We will adopt a quotation-based graph model to represent our online discussions (Barcellini et al., 2005), and as a basis to analyze three aspects of the online design dynamics: social, thematic temporal, and design dynamics.

The social dynamics characterizes the influence of participants in the design discussions. Indeed, OSS projects are organized in hierarchical communities with statuses, but some roles can emerge from interactions (Mahendran, 2002; Baker et al., 2003). We analyze the participants’ positions in the online discussions in order to investigate this question: Do participants with different statuses take part differently in the online discussions and the design process? Four statuses are identified: the project leader, the administrators of the project, the developers, and the Champion of the PEP (the one who proposed the PEP). To construct a social view of online discussions, we analyze the relationship between the statuses of the posters and their involvement in the discussions in terms of (1) the number of posted messages and three other measures; (2) the type of quotation structure (no quotation, one quotation, multiple quotation) used by participants (as a reflection of the richness of the content of a message); (3) the depth of a quotation (Is a quotation embedded in a subsequent quotation and then further embedded in one following that, etc.); and (4) the position in the discussion.

The thematic temporal dynamics characterizes how design themes emerge and evolve over time. Are asynchronous online design discussions framed by and focused on design or themes? Do they have some particular properties that differentiate their thematic coherence and temporal ordering from face-to-face exchanges? We construct the thematic temporal view by identifying the design themes (Olson et al., 1992) addressed by messages in order to characterize the organization and the temporal emergence of these themes during discussions. The degree of synchronism/asynchronism of discussions will be outlined on the basis of the posting date and the delay between a message post and its first quotation.

The design dynamics characterizes the dynamics of collaborative design activities in the online discussions. To what extent do OSS design situations correspond to face-to-face design in terms of collaborative design activities performed and in terms of sequences of activities that lead to the final design (d’Astous et al., 2004; Olson et al., 1992)? This design view is based on a content analysis of messages. Several kinds of activities are analyzed: proposal, evaluation, clarification, explicit decisions, co-ordination, and synthesis. In the following we present the main results of this three level analysis.

 Concerning the social dynamics, we show that roles emerge during discussions, and how they influence participation in the design discussions. In all of the discussions, the project leader and the champion of the PEP are frequent contributors and post messages that lead to multiple branches in the discussions (see Figs. 9 and 10). The champion is the only discussant who writes syntheses of what has been said. Our results suggest that complementary roles are occupied by the project leader and the administrators, the latter relying on the project leader when the theme is a crucial one for the Python language and filling in for him when he does not want (or cannot) participate in the discussion anymore. This confirms Mahendran’s (2002) findings in which he notes how power is distributed between the project leader and the administrators. Finally, we show that discussions are not “closed” for developers even if discussions are framed by the project leader, the administrators, and the champion of PEP. Developers do participate in the design process and their messages contain design proposals.

In our analysis of the thematic temporal dynamics, we examine how themes of discussion emerge, diverge and are refined over time. Design-oriented, online discussions are thematically focused, in contrast to previous findings from CMC research (e.g., Herring, 1999) where online discussions themes disappear over time. In our corpora, initial messages are PEP-related multi-thematic ones. After a series of exchanges, messages tend to become specialized and focused on a single theme of discussion. In some single-theme, multi-day discussions we have observed moments of quasi-synchronous interchange in which multiple messages are received within an hour’s time. These moments reflect implicit social rules of the design community (Conen, 2004; Coehendet et al., 2000; Stewart and Gosain, 2006): rules that concern what is considered “timely” for the community. A designer aiming to participate in the design theme discussion has to do it “in time” or risk being chastised by the project leader.

Concerning the design dynamics, we have seen that evaluation activity remains the most frequent activity (the two most frequent activities in face-to-face design meetings are (i) evaluation of design alternatives and criteria; and, (ii) clarification (Olson et al., 1992; d’Astous et al., 2004)). Clarification is also present in design-oriented, online discussions, but it seems to be largely controlled by the project leader who refers explanations – considered off-topic – to other threads of discussion. This suggests that specific sub-spaces exist in the discussion space: focused discussion design sub-spaces as compared to sub-spaces devoted to the construction of common ground between the participants (Olson and Olson, 2000; Carroll et al., 2003). We hypothesize that community members also elaborate common ground during face-to-face exchanges at international conferences (Barcellini et al., 2006). We also observed that few decisions are taken during the online discussions, but those decisions taken are explicit. In face-to-face design meetings many decisions are made in an implicit manner (Marty and Darses, 2001). This could be one of the main differences between face-to-face and distributed, online
design meetings. For the most part, a participant’s status in the project does not affect the kinds of design activities engaged in by the participant with the exception that certain activities are strictly tied to very specific statuses (for example, decisions are made only by the Project Leader). Despite the fact that discussions are informally moderated, there are real interactions between developers, administrators and the project leader about the design problems faced by the community. Patterns of interchange between designers exchanging messages in an asynchronous, online environment largely resemble the patterns of exchange observed in face-to-face design meetings (d’Astous et al., 2001, 2004).

Our results problematize some practitioners’ models of OSS development, such as the bazaar model (for OSS) vs. the cathedral model (for proprietary software) (Raymond, 1999). Major project, especially, have much more circumscribed – cathedral-style – design processes. Our research and the method we have developed are empirical means to explore the socio-cognitive mechanisms behind OSS design and to characterize how knowledge and artefacts (specifically, software) are collectively elaborated in an epistemic, exploration-oriented, OSS community.

To generalize these preliminary results, we need to repeat our work on other corpora. The type of analysis “by hand” we have performed for the current work is extremely laborious. Consequently, we propose to automate some of this analysis using a tool based on the Conversation Map (Sack, 2000), a piece of software designed and implemented by one of the co-authors for automatically analyzing and visualizing the semantics and social networks of large-scale online discussions. These types of automated tools may be useful for both researchers and for members of the design communities to provide them with design rationale tools (Buckingham Shum and Hammond, 1994; Concklin and Burgess Yakemovic, 1991; Moran and Carroll, 1996); and organizational memory tools (Sauvagnac and Falzon, 2003). An extension of this research will be to understand the entire design process in the broader community: from the emergence of a need of the user and/or developer communities, through its acceptance by the communities, its formalization in a PEP, and its implementation. We have, already, begun to investigate this design-use mediation process (Barcellini et al., 2006).

2. Introduction

The goal of our research is to investigate Open Source Software (OSS) design: a design process that is geographically distributed, communitarian, and asynchronous. OSS design is becoming increasingly widespread in the computer science world. There are thousands of OSS projects. Some of them are very successful, such as the Mozilla (www.mozilla.org) and Apache (www.apache.org) projects.

OSS design is an example of distributed and asynchronous collaborative design. Other kinds of designs – especially distributed, synchronous design and co-located collaborative design – have been more thoroughly researched (e.g., Détienne et al., 2004; Stempfle and Badke-Schaub, 2002; Olson et al., 1992). In these studies, design is seen as complex socio-cognitive process (Détienne, 2006).

Increasingly, Internet technologies (e.g., mailing lists, groupware, wikis, etc.) mediate the work practices of globally distributed design teams. Studying OSS design can provide relevant insights to understand how teams cope with the multiple discontinuities of distributed design (Watson-Manheim et al., 2002) such as distance, professional boundaries, organizational boundaries and temporal asynchronism. OSS also provides a field to investigate how mediation affects collaborative design processes and to propose enhancements to cooperative tools and methods; tools and methods that still suffer from weaknesses identified by research in Computer-Supported Cooperative Work (CSCW) and Human–Computer Interaction (HCI) (e.g., Olson and Olson, 2000). The larger software industry has become more and more distributed and global and tends to use OSS design tools and methods (Gutwin et al., 2004). Consequently, studying OSS is also a means to understand specific aspects of software design in general. Finally, we hope that our work contributes towards the elaboration of a common framework that might be used to compare and consolidate the results of empirical investigations of OSS design activities, dynamics, coordination mechanisms and work practices (Scacchi, 2001; Lerner and Tirole, 2002).

The objectives of our work are twofold. First, we aim at developing a methodology to describe and understand OSS design processes. Second, on the basis of this methodology, we aim at analyzing the dynamics of the OSS design in a particular OSS community, the Python community, as a case study. The Python community is a major and stable OSS project devoted to the design of the Python programming language. Because the major part of the OSS design process occurs online in a discussion space (Mockus et al., 2002; Sack et al., 2006), we analyze online discussions between Python developers engaged in a specific design process used to define and to specify revisions and improvements to the Python language. The process is called the Python Enhancement Proposal or PEP.

Finally, our case study will enable us to identify research directions to be explored in a larger scale study. Such a larger scale study will employ a set of analysis tools that automate parts of our methodology. With these tools we hope to be able to substantially expand our analysis beyond the confines of the case study described here.

In the following sections we first present the context of our study highlighting the main characteristics of OSS design activities. Then, we outline our research approach after a review of previous studies of OSS design practices and the collaborative activities of more traditional design projects. Finally, we present the methodology we have developed and discuss our results.
3. Theoretical framework

In order to understand the complex and hybrid form of OSS design, we employ methods and theories from various fields. Studies from software engineering, economics, and sciences and technologies help us to clarify social organization and regulation within OSS communities. Cognitive ergonomics and human-computer interaction allow us to clarify and distinguish between socio-cognitive collaborative design activities. Triangulating these different perspectives enables us to describe emerging roles that can be observed in OSS design.

3.1. Socio-organizational characteristics of Open Source Software design: communities, regulation and coordination

3.1.1. OSS projects as a hybrid form of communities

Chroniclers of OSS (e.g., Raymond, 1999; DiBona et al., 1999), as well as some scientific studies refer to OSS projects as communities (Bonaccorsi and Rossi, 2003; Scacchi, 2001; Dalle and Jullien, 2003). Specifically, an OSS project can be seen as an online community according to Preece’s definition (Preece, 2000; de Souza, 2003; Scacco, 2001). OSS projects are governed by the norms and principles of OSS (Stewart and Gosain, 2006). For example, OSS designers believe that OSS software design methods are more effective than a proprietary software approach. Moreover, activities in OSS communities are framed by implicit norms. Volunteer participation and evaluation of work by a peer-review mechanism (e.g., Raymond, 1999) are two such norms. Censorship of newcomers is a third example of an implicit norm: members need to be aware of the history of the project to avoid “spamming” mailing lists with already-posted and answered topics. Uninformed newcomers risk censure by other members if they do not know this history.

OSS projects can also be seen as epistemic communities (Cohen et al., 2000; Conein, 2004). Indeed, OSS members have the “meta” objective of producing and constructing knowledge about the artefact they develop for the benefit of the entire community. Their objective is not only to gain individual knowledge but also to co-construct and share knowledge.

Finally, OSS projects are also identified as meritocratic communities. Even though many major OSS projects are very hierarchical (Gacek and Arief, 2004; Mahendran, 2002; Crowston and Howison, 2006) it remains possible to rise within these hierarchies on the basis of apparent technical and discursive skills (Mahendran, 2002; Ducheneaut, 2005). Five different statuses are generally distinguished in major OSS projects; these differ according to the distinctive rights and power of participants. Some participants can modify the source code and participate directly in the design process and in the decisions concerning 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 maintain the code base, the documentation, and participate in the design and development of the software;
- The developers or contributors who participate to the evolution of the software 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 away 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 can be developers in another project;
- Other users are called passive users as they only use the software or lurk on the discussions and documentation spaces of the project (Preece et al., 2004).

It is possible to improve one’s status by proving one’s technical skills, and engaging and maintaining online discussions (Ducheneaut, 2005; Mahendran, 2002).

3.1.2. Organization and coordination within OSS design projects

OSS design is asynchronous and geographically distributed. OSS designers are not usually in the same place at the same time and various Internet tools mediate their activities. Thus, the OSS design process is distributed among three different activity spaces as described in Fig. 1. (Barcellini et al., 2005; Sack et al., 2006): a discussion space (mailing lists, forums, chat), a documentation space (source code and its development history1), and an implementation space (source code and its development history1). Most of the design activity occurs in the discussion space (Mockus et al., 2002), in which design choices, alternatives, and their rationales are elaborated and critiqued.

Explicit, design-related coordination mechanisms are elaborated, especially in large OSS projects (Mockus et al., 2002; Scacchi, 2001). These mechanisms are partially supported by tools. The history of a project’s implementation and its different versions is supported by versioning systems like CVS or Subversion. Bug report systems, such as in Bugzilla (Sandusky et al., 2004), can be also provided by default by OSS repository platforms such as Sourceforge.

The Python project, on which this work is focused, uses the Python Enhancement Proposal (PEP) process as an explicit means for proposing changes and extensions to the Python language. During the PEP process, community

---

1 Supported by the Concurrent Version System (CVS) or by Subversion.
input are collected in the discussion space and the chosen design decisions are documented. The term “PEP” refers both to the design document and the associated discussion process. The PEP document is written to describe a new language feature: it is intended to provide a concise technical specification of the feature, the rationale for the feature, and a reference implementation. Thus, the PEP design process includes activities of documentation, but also of discussion and implementation. Once the PEP editors (an administrator and a developer) have accepted a rough-draft PEP, the author of the PEP – called the champion – is responsible for posting the PEP to the community forums and mailing lists where the PEP is discussed. Archives of discussion, decisions regarding the PEP, and the different versions of PEP are kept in the documentation space. Information about the PEP and its status is, therefore, distributed between these two spaces. After a PEP has been accepted, it is given a final review by the leader of the Python project. Finally, a new piece of code is written to implement the PEP. This code is integrated into the project’s code archive: the implementation space.

3.2. Socio-cognitive characteristics of collaborative design

This section focuses on cognitive characteristics of design situations in general as identified by research in the cognitive ergonomics of design. Cognitive ergonomics studies cognition in work settings in order to optimize human well-being and system performance. Traditionally, it focuses on cognitive processes – e.g., diagnosis, decision-making, and planning (Green and Hoc, 1991). As outlined in the following section, it also describes collaborative, social and cognitive processes involved in design situations. The main basis of cognitive ergonomics’ methodology is to study the activity (Engeström et al., 1999) of subjects in order to produce knowledge about processes, strategies and constraints they employ in real world work environments. Its main goal is to produce recommendations to enhance socio-technical work settings (tools, organization).

3.2.1. Design as a collaborative activity

From a cognitive point of view, design has been considered mostly as an individual problem solving activity. This consists in specifying an artefact based on given requirements that indicate one or more functions to be fulfilled, and objectives to be satisfied by the artefact (Visser, 2004). Design problems are considered to be ‘ill-structured’ problems (Simon, 1973).

There has been, recently, a switch of focus in design studies: from individual to collaborative design (Détienne, 2006). The “‘pure” cognitive framework based on information-processing models is not sufficient to address the issues related to the collective nature of work. Thus, theoretical frameworks adopted in design studies have tended to integrate social and organizational aspects (Bucciarelli, 1988; Schön, 1988) together with the situated dimension of collaborative design.

Researchers in cognitive ergonomics have closely examined interactions between designers and/or design participants. Three modes of interaction can co-exist in a collaborative design project: face-to-face exchanges (co-presence and synchronous meetings); mediated and synchronous exchanges (e.g., video conferencing); and mediated and asynchronous interactions (e-mail, forums or groupware exchanges). Many studies have examined collaborative activities in face-to-face design meetings, especially in software design meetings (e.g., d’Astous et al., 2001; d’Astous et al., 2004; Herbsleb et al., 1995; Olson et al., 1992). A smaller number of studies have focused on mediated, distributed design teams (e.g., Olson and Olson, 2000; Détienne et al., 2004). We shortly review some previous studies of face-to-face and distributed design to enumerate the main characteristics of collaborative design activities.

3.2.2. Three types of collaborative design activities

Collaborative activities include activities in which designers are co-engaged to modify and control the object of the design and the design process itself in a common field of work (available artefacts and resources) (Schmidt, 1994).

One set of collaborative activities is related to the process of solving and evaluating various aspects of design problems. We refer to these as generation-evaluation activities. These activities concern the framing of the design problem (or theme) and articulation of its solution; e.g., elaboration of the problem, proposition and evaluation of alternatives of solutions.

Co-designers accomplish evaluation activities by arguing with each other. These arguments have a very specific form and can be characterized as sequences of moves or turns (d’Astous et al., 2004; Stempfle and Badke-Schaub, 2002).

A second type of activity concerns the construction of common references, or common ground, within a group of co-designers. These clarification activities take place when a group negotiates or constructs a shared representation of the current state of the problem, solution or design rules or criteria. Clarification activities (also called cognitive synchronization) are especially important for mediated, synchronous design (Détienne et al., 2004) as well as for
face-to-face design situations (d’Astous et al., 2004). They can constitute one third of the exchanges between co-designers.

Group management activities are a third kind of design activity. These activities are related to issues of the 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 or tasks – are another example of this kind of activity.

In the following section, we describe how role can emerge in OSS design situations based on the effective performance of these collaborative design activities, in particular in online discussions.

3.3. Emerging roles in OSS design

In most OSS projects, some participants have formally assigned administrative or managerial statuses, as described in 3.1. Previous studies of OSS (Conein, 2004; Mahendran, 2002) but also of other design situations (Baker et al., 2003; Dameron, 2002; Sonnenwald, 1996) and other online interactions (Cassell et al., 2005) suggest that a participant emerging role in an OSS project may be dependent upon his or her formal status, but is also contingent upon the participant’s actions in the design process and in the online discussions. This notion of role reflects the effective and emerging behavior of participants. In some cases, these activities may correspond to what is expected from a particular status.

Several studies of online discussions, and online communities (e.g., Cassell et al., 2005) have highlighted how participants actively construct their positions and their roles within an online community. These roles reflect the frequency of participation, and the content of the messages posted by a participant, and may partially depend upon the status of a participant. Maloney-Krichmar and Prece (2002) illustrate the emergence of roles in an online community: they show that community members create a mental model of the group dynamics and roles that serve as the basis for their involvement and participation in the community.

In an OSS project, roles may emerge from interactions between participants in the discussion space, the main place for exchanges and collaboration (Conein, 2004; Mahendran, 2002). ‘Bot, short for robot, is the nickname for one such role. ‘Bots gain their nickname by replying quickly and precisely in technical mailing lists. Roles emerge and are actively constructed. It is possible to change roles within the community by proving one’s value to the project (Ducheneaut, 2005; Mahendran, 2002; Jensen and Scacchi, 2005). A peer-review mechanism is the main basis for the evaluation of community members.

In other design situations, team members’ respective activities are differentiated and various roles thus appear in a project design group. These emerging roles support collaboration, integration and exploration of different knowledge and facilitate negotiation between members during the design process (Sonnenwald, 1996).

Three types of activity mutually frame participant’s roles in design situations (Baker et al., 2003): epistemic activities (i.e., contributions of knowledge to the project by the participant); argumentative (e.g., advancing or refuting an argument) and interactional activities (e.g., maintaining the topic of discussion); and enunciative activities (i.e., taking in charge of one particular stakeholder position).

Roles in OSS communities reflect the project’s organizational structure, the technical skills and activities exhibited by participants, and the participants’ contributions to the online discussions. We propose a method whereby we follow the participants in all of their different activities and weave our heterogeneous findings together to articulate each of the participants’ emerging roles.

4. Research strategy

The first sub-section motivates our choice of the Python project as a site of study. The second sub-section presents the quotation-based methodology we have developed to study OSS design discussions. Finally, the third sub-section presents the three dimensions of the OSS design dynamics that we investigate.

4.1. Why Python? An OSS project and its design-related discussion space

As the goal of our research is to understand OSS design process, we chose to focus on a single OSS project as a case study (Yin, 1994) rather than do a comparative study (e.g., Scacchi, 2001; Gacek and Arief, 2004; Nakakoji et al., 2002). Among the wide variety of OSS projects, we chose to focus on major exploration-oriented OSS, epistemic in nature, that “aim[s] at pushing the frontline of software development collectively” (Nakakoji et al., 2002). Projects that are epistemic and design-focused, are underpinned by a hierarchy of project leaders. They “must be developed and maintained by expert programmers, such as project leaders”. And so, the success of the project thus depends on the vision and skill of the project leaders. A canonical example of such a project is the Linux kernel project. Our focus, the Python project, is less studied than Linux but it has all the qualities of such a project. It comprises a stable group of less than a hundred developers dedicated to the design of the Python object-oriented programming language. Thus, the number of interactions and lines of code produced is quite reasonable compared to the Linux community with its hundreds of developers. The Python project has also a core group of leaders: the project leader himself, who has a strong authority over the project, and the core team of Python, composed of six administrators at the time of our study (Mahendran, 2002).

The Python Enhancement Proposal (PEP) process, we focus on, is quite similar to two design processes used in
conventional software projects: Request For Comments (RFCs) and Technical Review Meetings. RFCs have been practiced for decades to define standards for the Internet (especially by the Internet Engineering Task Force, IETF). Technical Review Meetings (d’Astous et al., 2004) have been practiced in many corporate and governmental settings. The PEP process is also similar to other design processes used by distributed design communities, like the XEP process (XMpp Extension Protocol) of the jabber community (www.jabber.org), and the PLIP process (Plone Improvement Proposal) (www.plone.org) of the Plone project. These formal processes are also close to the consensus-based decision-making of Apache (www.apache.org) in which design issues are discussed and chosen online. Because the PEP process closely resembles these other design processes, the Python project was also of interest to us insofar as it is representative of a larger number of other projects and their design processes.

Moreover, this research is built upon collaboration between researchers in cognitive ergonomics and socio-informatics that had already investigated some aspects of the Python project (Mahendran, 2002; Ducheneaut, 2005). For his Ph.D. dissertation, our colleague Ducheneaut investigated the evolution of links between people in two activity spaces, the discussion and implementation spaces. He showed how newcomers can (but sometimes cannot) be progressively integrated into the social and the technical structure of the Python project. Our colleague Mahendran’s ethnographic work investigated the social structure of the Python project and how power is distributed among the participants. In our current research we have endeavoured to extend Mahendran’s and Ducheneaut’s work.

Finally, this project has been less explored than other projects such as Linux (e.g., Hertel et al., 2003; Cohendet et al., 2000), Mozilla or Apache (e.g., Mockus et al., 2002; Franke and Von Hippel, 2003). We believe that it is essential to study OSS communities of different size, history, organization, and institutional background, in order to build model(s) of OSS development in general.

Since most of the design occurs in the discussion space (Mockus et al., 2002; Sack et al., 2006), we investigate the discussion space of the Python project. Specifically we have analyzed the python-dev mailing list, dedicated to the development of the Python language. This is the mailing list in which most of the design-related discussions are supposed to occur. Within the discussions, PEPs are usually referred to according to an assigned number and keyword. This facilitates, methodologically, our efforts to identify PEP-associated design discussions within a much larger corpus (frequently hundreds) of discussions, which take place on the mailing lists each month.

4.2. A methodology combining structural and content quotation-based analyses

Studies of OSS often employ structural analyses of the various logs and archives available (CVS, Bug Trackers, mailing lists), sometimes combining them with social network analysis (e.g., Lopez-Fernandez et al., 2004). Others have used ethnography (Mahendran, 2002) or virtual ethnography in which ethnography is combined with structural analysis (Ducheneaut, 2005). Large-scale surveys have also been done (Ghosh et al., 2002). Relatively few studies of OSS have employed content analysis (e.g., Ripoche and Sansonnet, 2006). Our approach is to develop a method that combines structural and content analyses of the online discussions, mainly based on quotation. In developing our method we have also attempted to formalize our analysis procedures in order to make it possible, in future work, to automate the procedures.

Previous studies investigating online discussions have been mostly conducted in the field of Computer-Mediated Communication (CMC). They have mainly focussed on “open” online discussions, such as public forums or chats. CMC studies have highlighted the lack of thematic coherence – how a turn connects to previous ones – in online discussions. A “turn” in an online discussion is constituted by a message from one of the participants. In online forums, a message can be separated both in time and place from the messages posted in response to it. Consequently, themes of discussions decay rapidly and conversation drifts (Herring, 1999), which can be confusing. Participants try to minimize this confusion by adopting compensatory conversational linking strategies, such as quoting, i.e., including portions of previous messages in a new one.

Quoting from previous messages creates the illusion of adjacency: it incorporates portions of two (or more) conversational turns into a single message by juxtaposing a comment with previous contributions. It maintains context in a single message (Herring, 1999), at least from the quoted’s point of view. It is an explicit linking strategy in contrast to implicit deictic or anaphoric references to previous messages (Eklundh and Rodriguez, 2004). This context-preserving mechanism is a widely used technique in online discussions where the majority of responders use it selectively. They perceive its use as contributing to the sense of conversation (Eklundh and Macdonald, 1994).

In the case of OSS design discussions, quotation is a common activity used by designers to maintain the context of the discussion. We assume that this activity is similar to the mechanism of turn-taking in face-to-face meetings. In a previous paper (Barcellini et al., 2005) we proposed an analysis based on the quoting activity in design-oriented online discussions as a mean to interrogate the dynamics of OSS discussions and their thematic coherence. We showed how the thematic coherence of online discussions is better represented by a quotation-based analysis than

---

2 Linking is the practice of referring to the content of a previous message (or previous messages as we will see in our analysis) in one’s response.

3 A deictic reference is understandable if the reader already knows the context, as in the sentence “it is here” here is unknown for one that come through the discussion. An anaphoric reference is a word or phrase referring back to something previously stated.
by the usual threading model based on the “reply-to” link between messages; i.e., what most email and newsgroup users would see as shared subject lines between messages. We made this comparison by constructing two graphs. Both graphs have nodes that represent messages. In the graph constructed according to the usual threading model, links between nodes are derived from the “reply-to” information in the headers of the messages. In the other graph, which represents a quotation-based analysis, two nodes are linked if one message contains a quote from another message. For an example, please refer to Section 6.2.1 of this paper. We found that the thread-based analysis incorrectly divided some theme-related messages into different threads and, furthermore, categorized as peripheral certain messages that were central contributions to the discussion. A quotation-based analysis did not exhibit these weaknesses: same theme messages are grouped together, there are more key relationships between messages and these key messages are linked to the main discussion.

In this paper, we will adopt this quotation-based graph model to represent our online discussions, and as a basis to analyze the dynamics according to the three views presented below.

4.3. Research questions: investigating social, thematic temporal, and design dynamics of OSS design discussions

Dynamics in OSS design community can be analyzed according to three views which, taken together, give one a picture of the emerging roles of participants.

4.3.1. Social dynamics

The social dynamics characterize the influence of participants in the design discussions. We analyze the participants’ positions in the online discussions in order to investigate this question: Do participants with different statuses participate differently in the online discussions and the design process?

To construct a social view of online discussions, we analyze the relationship between the status of the posters and their involvement in the discussions in terms of (1) the number of posted messages and three other measures: (2) the type of quotation structure (no quotation, one quotation, multiple quotation) used by participants (as a reflection of the richness of the content of a message); (3) the depth of a quotation (Is a quotation embedded in a subsequent quotation and then further embedded in one following that, etc.?); and (4) the position in the discussion.

4.3.2. Thematic/temporal dynamics

The thematic/temporal dynamics characterize how design themes emerge and evolve over time. Are asynchronous online design discussions framed by and focused on design problems or themes? Do they have some particular properties that differentiate their thematic coherence and temporal ordering from face-to-face exchanges?

To highlight this thematic temporal view of the discussions, we analyze the design themes addressed by each message in order to characterize the organization and the temporal emergence of these themes during discussions. The degree of synchronism/asynchronism of discussions will be outlined on the basis of the posting date and the delay between a message post and its first quotation. The temporal coherence will be outlined combining a content analysis of design themes and these formal criteria.

4.3.3. Design dynamics

The design dynamics characterizes the dynamics of collaborative design activities in the online discussions. To what extent do OSS design situations correspond to face-to-face design in terms of collaborative design activities performed and in terms of sequences of activities that lead to the final design?

To highlight this design view we complement the content analysis of messages by looking at design alternatives and the problems they address as well as the collaborative design activities identifiable in the quotations and their associated comments. Quotes and comments are chosen as the basic discursive units. This choice makes visible both the nature of design exchanges and the typical sequencing of activities. To complete the social view description we investigate the links between the social position and the nature of the design activities produced by participants.

5. Method

In this section, we describe how we collected and selected the data we have analyzed and the method we have developed to analyze these data according to the three views described above.

5.1. Collection and selection of the data

All PEP discussions are archived on the Web and publicly available on the python-dev archives (http://mail.python.org/mailman/listinfo/python-dev): the mailing list intended for Python design discussions. At the time of our study, there were 161 PEPs discussed in the Python project. Twenty-eight PEPs are meta and informational PEPs related to the PEP process itself and 133 PEPs concern new features for Python (51 accepted and implemented; 2 accepted but not yet implemented; 26 deferred, rejected, abandoned or withdrawn; 2 empty (abstract); 52 are still open and under consideration). We focus on design-related PEPs to investigate the OSS design process.

Our colleagues gathered data concerning the social structure of the Python project (Ducheneaut, 2005; Mahendran, 2002) for the first half of 2002. Thus, we chose to study PEP processes from March and April of 2002. We assume that the Python project had the same membership during this period.

To select the data the first step was to collect all the PEP-related discussions: the data was gathered by searching, by
hand, from the python-dev mailing list for the keyword PEP in the subject header of each message. Among the 342 different discussions (i.e., threads with different subject headers) occurring in March and April 2002, 53 discussions are dealt with 13 different PEPs (751 messages from 87 different participants). Five were meta and informational PEPs and eight were design-related PEPs: one was an open PEP (i.e., a PEP that had been opened but not yet implemented at the time of the study); one was an already rejected PEP; and six were ongoing PEP processes that ultimately led to an accepted PEP. Finally, among these six we chose PEPs that just had been created to be sure to capture the core of the design process and not just refinements of previous rationales. This led us to concentrate on three accepted, design-related, early PEPs. We chose to compare PEP 279 and PEP 285 because they generated the largest number of messages.

These two PEPs had the same status (accepted and implemented) and were at the same stage in the design process, but they differed according to

- The social positions of their respective champions: PEP 279 was proposed by a developer and PEP 285 by the project leader;
- The design themes addressed in the discussions: both PEPs proposed the introduction of new functionality. But, a “new built-in type” proposed in PEP 285 entailed a more general discussion of backward compatibility. No such meta-issues were broached in the discussions of PEP 279.

In order to be able to compare these PEP processes, we analyzed one main discussion for each of the two PEPs: the one that began with the first post by the champion of the PEP calling for community input. PEP 285 generated two main discussions of 129 messages by 22 authors. PEP 279 generated two main discussions of 228 messages by 22 authors. These initial contributions to the discussions comprised the majority of the design process. After the initial contributions, one or several design decisions had been taken. The sizes of the corpora we analyzed are as follows:

- PEP 279: 71 messages posted by 21 authors between March 28th and April 8th 2002; 3600 lines of text;
- PEP 285: 95 messages posted by 22 authors between March 29th and April 5th 2002; 3500 lines of text;

As a means to contextualize these two corpora, for the remaining PEP discussions, we completed analyses of status and involvement as presented below.

In what follows, we will use the quotation-based representation of these PEP discussions to support our analyses of social, thematic temporal, and design dynamics.

5.2. Social view

Our first step in constructing a social view of online discussions is to characterize the involvement of participants into the discussions according to their statuses. The second step is to identify the different quoting activities of the participants. Next, we examine more closely the messages’ structure and content.

5.2.1. Statuses and involvement of participants

The statuses of participants have been identified in previous work done by our colleague Mahendran (2002). Based on this work, we identify four kind of participants in the python-dev mailing list: (1) the project leader (PL); (2) the champion (C) of the PEP; (3) the administrators (A); and (4) the other developers (D) who are all other participants to the discussions.

We characterize the involvement for each participant in the online discussions according to the median number of messages posted. Combining the statuses of participants and their involvement leads us to distinguish between (1) HP-AID: administrators (including the project leader) and developers (including the champion) who sent more than two messages are High Participant Administrators (HP-A) or High Participant Developers (HP-D) and (2) LP-AID: those who posted two or fewer messages are termed Low Participant Administrators (LP-A) or Low Participant Developers (LP-D).

Quotation practices will be analyzed according to the combination of status and involvement, and positions in the discussion graphs (position and depth of quote).

5.2.2. Message quotation structures

After selecting the messages of interest, we assign them an arbitrary number. Within the texts of the messages, quotes are usually identifiable according to a simple syntax: greater-than (“>”) symbols precede each quoted line. But this syntax is not always reliable and so, for our analysis, we have also compared messages by hand to identify the message(s) quoted.

Each message is categorized according to its quotation structure and the source message(s) that is (are) quoted. The structure is defined according to the alternation of blocks of quoted material and blocks of commentary (new text) in a message:

- A text-only message is a message that does not contain any quotations.
- A single quote message is a message with one block of quotations (in italic) followed by a comment.
- A multiple quotes (Mq) message is a message containing alternating quotes and comments from the same message or from different ones (as in the example below).

```python
xxx writes:
> So now I'd like to choose between enumerate() and indexer(). Any closing arguments?
"indexer" is the name of the built-in full-text indexer, right? ;-)
```
5.2.3. Quotation patterns

We have investigated two aspects of the quotation patterns present in the discussions: the messages’ positions in the discussion graph and the depth of quotation exhibited in the messages. Three patterns of quotation are distinguished according to the position in the graph (see for instance Fig. 13 in the results section) of the messages posted by participants:

• Sequential structure: a message is quoted by only one message.
• Branching structure: a message is quoted by more two or more messages.
• Closing structure: a message is in a closing structure when it is not quoted at all.

The depth of quotation is defined as the distance between a source message and the last message that quotes it directly or indirectly (for instance, messages directly quoted by only one message have a depth of quotation equal to 1; those quoted one time directly and one time indirectly have a depth of quotation equal to 2, etc...).

5.3. Thematic temporal view

For our construction of a design-related view of the discussion dynamics we identify the design themes addressed by each message and characterize the temporal ordering of the discussions.

5.3.1. Design themes

We performed an analysis of the content of messages in order to identify the themes addressed by each message.

For the PEP 279 discussion, five themes (T) have been identified corresponding to the following technical design problems: T1 concerns the choice of a name for the proposed, new functionality of the language; T2 concerns alternative syntaxes for the proposed function; T3 concerns the utility of the proposed function and its potential usability; T4 concerns issues related to error handling: how will the new function signal an error to the user (or to other parts of the user’s program) if something goes wrong during the execution of the function? T5 is a thematic drift that concerns an orthogonal problem of name binding and the status of name spaces (i.e., two other technical issues).

For the PEP 285 discussion, there are six main themes corresponding to the following technical design problems: T1 deals with the consequences of a new built-in data type for the Python language (specifically the boolean type used to declare variables which will hold either the value true or the value false). Theme 1 is in fact a “meta-theme” that deals with possible implications of the PEP acceptance and how it might impact other, existing aspects of the Python language. Previous discussions about this theme have occurred in the project (they are mentioned in some messages). Theme 1 is controversial, it deals with the acceptability of the PEP, and is supported by the project leader; in that sense it is a “meta”-theme in the discussion. T2 deals with the specific function, the status of the type of the variable (boolean or integer) that is returned and complications for existing Python code. Can the function be introduced into Python in a backwards compatible form that will not force programmers to re-implement existing, working code? T3 concerns the name of the constant of the new built-in type; T4 concerns the elimination of non-boolean operations on booleans; T5 concerns a specific operator of Python and what it should return, a boolean or an integer; T6 concerns the inheritance relationship between two types (integer and boolean).

One main difference between the two discussions appears from this first analysis: PEP 279 discussion is focused on a clear and well-defined restricted design problem (the choice of the function name) whereas the PEP 285 discussion concerns large implications for the programming language.

5.3.2. Temporality

We extracted dates and server times from the headers of messages. Using this data we examined two aspects of the discussions’ respective temporalities:

• Quotation temporality: the delay between the time of a message’s posting and its quotation in a subsequent message. This characterizes the reactivity of participants and the degree of asynchronism of the discussion.
• Theme temporality: the amount of time between the posting of a first message addressing a design theme and the final message(s) addressing the same theme. This characterizes the longevity of a theme of discussion. Do discussants talk about a given theme for minutes, hours or days?

5.4. Design view

For our message content analysis we identify the design alternatives proposed and discussed and the design activities that appear in quotes and comments.

5.4.1. Design alternatives

Solutions to design problems, or alternative solutions, are proposed by developers according to the themes discussed. We identify as design alternatives every solution produced for each design problem or theme. For PEP 279, theme 1 (the choice of a function name), 23 design
alternatives were proposed; eight were proposed for theme 2; and three for theme 3. For PEP 285’s discussion, only 3 alternatives are proposed. In fact, this discussion was “closed”: developers were asked to give their point of view about the acceptance of the PEP in terms of its implications for the language (theme 1), or to evaluate some already existing alternatives (theme 3, 5, or 6).

5.4.2. Design activities identified from the content of quotes and comments

The first author has done the coding by hand, on the basis of a coding scheme developed in our research team to study other design situations (Darses et al., 2001; d’Astous et al., 2001, 2004; Détienne et al., 2004), this kind of coding scheme being use in other studies (Olson et al., 1992; Stempfle and Badke-Schaub, 2002). The following activities were distinguished in blocks of quotes and their following comments:

• **Proposal**: a developer proposes a new design theme or a new design alternative related to an existing theme. To identify this activity, we analyze expressions like “How about…?”, “Why not…?” and verbs like “to propose”, and “to add”. Here are extracts from several messages written by developers.

> I’d like to add (name_function)(...) What about doing it both ways…?

• **Evaluation**: a developer agrees or disagrees with a proposal or another proposition. This activity is identified by verbs such as “to like”, “to agree”, “to prefer”; or expressions like “yes”, “no”, “indeed”, “of course”, “great”, “sure”, or “±1” (a shorthand for voting for or against a proposal).

> -1. It looks like a noun to me

> Now, that’s a great idea

• **Group coordination**: a developer moderates the discussion or postpones a task.

> I think that #3 probably fits better in the new PEP you were gonna write for #4.

> When you send xxx your update to the PEP reflecting this, he can mark it Accepted.

• **Synthesis**: a developer recapitulates what has been discussed. Frequently, this takes the form of an enumerated list. In synchronous design situations we have analyzed in previous work, synthesis was a relatively unusual activity and so was not included in our prior coding scheme. But, in asynchronous design it occurs frequently and thus requires a category of its own.

> Okay, here’s what we have so far: Enum (…)

• **Clarification**: a developer provides an explanation or rearticulates a point to address another’s misunderstanding.

> As I said earlier in the msg, (...) I had in mind above

> Is this about the example below?

• **Explicit decision**: only the project leader has the power to make an explicit design decision. The examples below are extracted from a message from the project leader in which he approves of the proposal evaluations upon which a group of developers has reached consensus.

> > 3. gen exception passing --> deferred, needs case building
> OK (or you could give up now while you’re ahead :-).

• **Other activities**: a number of other activities fall outside our coding scheme. Humour, for example, is frequently exchanged between developers, but falls outside our analytical focus.

> Yeah, if I left Python’s design to xx, it would become quite the clever hack.

5.5. Descriptive statistical analyses

These analyses provide us with contingency tables presenting frequencies of our indicators coded as variables. The strength of the links between pairs of variables – as status and position in the graph for instance – can be described by two descriptive statistics: at a local level by relative deviation (RD) (Bernard, 2003) and at a global level by a co-efficient called $V^2$ Cramer.

RDs are calculated comparing each cell of the table – the observed frequencies – with calculated expected frequencies (i.e., those that would have been obtained if there was no association between the two variables). There is a positive association between two variables (an attraction), when RD is positive; and repulsion when it is negative. By convention, we report only attractions (and sometime repulsions) with values $>0.20$ ($<-0.20$). RDs highlight which modality of a variable ($v_1$), in a column, tend to be more associated to another variable ($v_2$), in a row: an attraction means that $v_1$ tends to be more associated with $v_2$ than other modalities. Conversely, negative RDs mean that $v_1$ and $v_2$ tend not to be associated. Thus, RDs are used here to reveal strength of associations between variables that can be difficult to outline in analyzing a contingency table. It is a complement to a frequency description of the contingency tables.

The co-efficient $V^2$ Cramer gives the global association between the two variables; it is a synthesis of all the local RDs. The association is insignificant for $0 < V^2 < 0.04$; intermediary if $0.04 < V^2 < 0.16$; and strong otherwise. Even if the global association is insignificant, local RDs can highlight some interesting associations.
6. Results

6.1. Social view of the OSS design process

6.1.1. Global distribution of statuses and involvement in the discussions

The distribution of participants, according to their statuses coupled with their involvement (Fig. 2), is quite similar in the two discussions ($\chi^2 = 0.32; \text{df} = 4; p = \text{n.s.}$).

The distribution of posted messages does not globally differ ($\chi^2 = 8.47; \text{df} = 4; p = \text{n.s.}$), (Fig. 3). In both of the discussions the project leader participated frequently; he posted between 18% (13/71) and 20% (19/95) of all messages. The champion for PEP 279 showed a similarly high level of participation with 14% (10/71) of the posted messages. This result was anticipated: both the champion and the project leader, according to their statuses in the project, are expected to be frequent discussants. However, a measure of relative deviation (RD) ($V^2 Cramer = 0.05$) showed that there is a local difference between the two PEP discussions. Administrators (HP-A in particular) participated more in PEP 279 discussions than in PEP 285 discussions. In contrast, developers participated more frequently in the PEP 285 discussions than in the PEP 279 discussions. This too might have been anticipated: the stakes are high for PEP 285; changes considered for this PEP could have long-term implications for large numbers of the Python community.

We verify that the distribution of participants in the two PEP discussions studied and in all the other PEP discussions is the same (Fig. 4) ($\chi^2 = 4.08; \text{df} = 4; p = \text{n.s.}$).

The distribution of posted message is different (Fig. 5) ($\chi^2 = 19; \text{df} = 4; p = 0.0001$) which is due to the fact that there is no LP-A in all the other PEP discussions; however, the involvement of other participants is similar.

6.1.2. Quotation activities

6.1.2.1. Quotation is a common activity. First of all, we verified that quotation is a common activity in our corpora since 94% (PEP 279)–98% (PEP 285) of messages are either one-quote messages or multiple quotes messages (Fig. 6).

Fig. 2. Distribution of participants according to their statuses and involvement in PEP 279 and PEP 285.

Fig. 3. Distribution of posted messages according to statuses and involvement of participants in PEP 279 and PEP 285.

Fig. 4. Distribution of participants according to their statuses and involvement in PEP 279 and 285 and in all other PEPs.

Fig. 5. Distribution of posted messages according to statuses and involvement of participants in PEP 279 and 285 and in all other PEPs.

Fig. 6. Distribution of quotations’ structures in discussions for PEP 279 and PEP 285.
Distribution of the various type of message structures does not differ between the two discussions ($\chi^2 = 3.98$, df = 2, $p = n.s.$). Furthermore, participants used quotation selectively since all quotes contained only a subpart of previous messages. This suggests that what is quoted is not determined by the default settings of the email clients used by the discussants.

6.1.2.2. Position within the discussion and quotation structure are not independent. We investigate the links between status and involvement within the discussions with the quotation structure of messages (Fig. 7). For PEP 279, there is an association of insignificant strength ($V^2$ Cramer = 0.02) between status and involvement and quotation structures. However, RD reveals a local difference: the PL tends to produce more multi-quote messages than other participants. For PEP 285, there is an intermediary association between these two variables ($V^2$ Cramer = 0.07). RD reveals that multi-quote messages tend to be produced by the PL and by low participants, administrators, and developers.

As there is no significant association between status and involvement, and quotation structures for the two discussions, we tested the association between quotation structures and position in the graph (Fig. 8). Interestingly, in the two PEP discussions, we found an intermediate association between these two variables ($V^2$ Cramer = 0.048 for PEP 279 and $V^2$ Cramer = 0.05 for PEP 285). RD reveals that branching messages are much more like to be multi-quote messages than text-only messages. Indeed, there are few text messages (3/71 for PEP 279 and 2/95 for PEP 285), which initiate a thread of discussion and thus in de facto first branching positions. RD also reveals that closing messages tend to be one-quote messages.

6.1.2.3. Position in the graph, statuses and involvement. Figs. 9 and 10 are visualizations of our quotation-based representation. They are discussion graphs wherein messages are labelled with the statuses and involvement of their posters (black circle for the project leader, star for the champion, triangle for administrators and square for the developers). Involvement in the discussions is symbolized using hashed-line for low participants and plain for high participants. In these figures, we visualize also ensembles of messages, which quote, directly or indirectly, a source message (encircled with a dotted line) and will discuss this last point in the next section.

Fig. 11 highlights the distribution of participants according to the position of messages they posted.

There is an intermediate association ($V^2$ Cramer = 0.06 for PEP 279, $V^2$ Cramer = 0.13 for PEP 279) between statuses and involvement, and the position in the graph. RD indicates that for PEP 279, a branching structure tend to be initiated by a message posted by either the project leader or the PEP’s champion. However, neither of them appears in the sub-discussion branch after message 63 – a thematic drift. However, in this thematic drift HP-A are much more present (which explain their high presence in branching positions). RD also reveals that developers tend to be in sequential positions. For theme 4 and theme 5 (the thematic drift), we note an alternation of developers and administrators. The project leader also tends to post closing messages. For instance, messages posted by the project leader close debates (message 50 for the thematic drift and message 2 which closes theme 2).

In the PEP 285 discussion, the project leader is very present (19% of messages posted) and the RD measures shows that he tends to initiate branching structures. HP-As are not very present (only in the meta-theme part of the discussion) and the RD shows that they tend to close the discussion. The major difference with the PEP 279 discussion is that the LP-D tends to initiate branching structures.

These differences can be related to the two different design problems of the discussions and the status of their champions. Indeed for PEP 279, the project leader reached a decision quickly. The remaining theme – theme 1 – concerns the choice of a name for the function (see the previous section of this paper on discussion analysis), which – in the end – is a decision left to the project leader. However, PEP 285 is a meta design problem that could have had important implications for the language. Previous debates about this PEP had already taken place and the developers may have felt especially invested in the outcome. Moreover, the project leader himself championed
Fig. 9. Statuses, involvement and position in the discussion for PEP 279. Gray zones symbolize depth of quotation.

Fig. 10. Statuses, involvement and position in the discussion for PEP 285. Gray areas represent depth of quotation.
this PEP so, most of the time, administrators and developers may rely on him to lead the debate.

6.1.2.4. Depth of quotation, statuses and involvement. The distribution of messages with a depth of quotation superior to one is quite similar for the two discussions ($\chi^2 = 4.84$, $df = 4$, $p = ns$) (Fig. 12).

RD reveals that for PEP 279, the messages, which are deeply quoted are more likely to be those from the project leader and the HP-A. Fig. 12 shows that the project leader and the champion are deeply quoted except in the thematic drift (message 64–49) where the HP-A is more deeply quoted. For PEP 285, developers are more deeply quoted and the RD reveals that is especially so for LP-developers.

Messages more deeply quoted are likely to be at the start of a branching structure (73%, 11/16, for PEP 279 and 70%, 12/17, for PEP 285). Less deeply quoted messages are in sequential positions.

6.2. Thematic temporal view

6.2.1. Theme specialization during the discussions

Figs. 13 and 14 are quotation-based graphs each representing one of the two discussions. Each message is represented by a shape: a circle, a square or a triangle. The different line styles (dotted, dashed, and unbroken) in combination with the various shapes indicate the themes addressed by the messages. The design themes and a specific pattern of discussion are visualized. This visualization makes clear the fact that first messages are frequently multi-thematic messages (represented as black circles) that are then usually followed by single-theme messages.

The number of single-theme messages is different for each of the themes (PEP 279, min = 3, max = 34, median = 13; for PEP 285 min = 1, max = 50, median = 7). Each of the two discussions has a particular, predominant theme addressed by participants (Theme 1, 34/71 for PEP 279 and 50/95 for PEP 285), and there are also themes addressed in only a few messages. For example, the project leader quickly rejects theme 3 in the first multi-thematic messages; and also rejects theme 4 discussed in several specialized messages. The project leader can also choose to stop a discussion and move it to another forum. This was the case for theme 5 – the thematic drift – that emerged from the theme 4 debates. The project leader stopped this drift (message 50), making clear to developers that the PEP 279 discussion is not the place for this question. The project leader can also support a parallel theme through a discussion. This is the case for theme 1 (the meta-theme), in PEP 285, which is present throughout the discussion associated with other themes. Previous discussions about this theme occurred in the project (they are mentioned in some messages). This theme is controversial and yet is supported by the project leader.

6.2.2. Temporality of online design discussion: degree of asynchronism and temporal coherence of discussions

We characterize the degree of asynchronism according to the delay between the time at which the server received a message and the time at which this message is quoted for the first time (Table 1). Half of the first quotes appear within less than 1 or 2 h after the post of the source message. This result suggests a small degree of asynchronism between participants. Message quotes appear relatively

Table 1

<table>
<thead>
<tr>
<th>Percentage of appeared quotes</th>
<th>PEP 279</th>
<th>PEP 285</th>
</tr>
</thead>
<tbody>
<tr>
<td>25% of 1st quotes</td>
<td>20 min</td>
<td>15 min</td>
</tr>
<tr>
<td>50% of 1st quotes</td>
<td>1 h</td>
<td>2 h 16</td>
</tr>
<tr>
<td>75% of 1st quotes</td>
<td>5 h</td>
<td>7 h 33</td>
</tr>
</tbody>
</table>

It is in fact a mixed-representation, because we use the reply-to link to put messages without quotes (text-only) in the thick of the discussion (3 messages for PEP 279 and 1 for PEP 285).

4 It is in fact a mixed-representation, because we use the reply-to link to put messages without quotes (text-only) in the thick of the discussion (3 messages for PEP 279 and 1 for PEP 285).

5 Except the theme 3 for PEP 279 and theme 3 and theme 4 for PEP 285 discussed in multi-thematic messages.
Fig. 13. Quotation-based representation of themes (PEP 279).

Fig. 14. Quotation-based representation of theme discussion themes (PEP 285).
Fig. 15. Temporal representation of PEP 279.

Fig. 16. Temporal representation of PEP 285.
quickly given the geographically distributed nature of OSS design community.

Figs. 15 and 16 are mixed-quoted-based visualizations of the temporal and the thematic coherence of the discussions. Messages that address the same theme are grouped together. They are chronologically sequenced: each day is delineated by a dotted line. We group together messages posted within 1 or 2 h of one another. (For PEP 279, 1 h was the median time until a message was quoted. For PEP 285, the median time was 2 h.). Discussions about the different design themes occur over different lengths of time. The themes of PEP 279 and PEP 285 are discussed for a minimum of 1 day, a maximum of 6 days, and a median of 2 days. The main themes for the two PEPs were the ones discussed for 6 days. Other themes (T2, T4, T5 for PEP 279; T1-6, T5 and T6 for PEP 285) were discussed for one or sometimes 2 days. These differences are largely due to the respective contributions to the discussion of the champions and project leader. In the discussion of PEP 279, for instance, theme 2 and the thematic drift were stopped by project leader (in messages 2 and 50, respectively).6 Both the champion and the project leader attempted to spur the community to reach a consensus. In the discussion of PEP 285, we observed a similar pattern: the project leader only intervened in the discussion for the first few days. Thereafter, the developers took over the discussion. Finally, at the end of the discussion, the project leader reappeared (in messages 64, 65, 66). We note the presence of quasi-synchronous moment of discussion (represented by the gray zone on the figures). These can be intra-thematic and correspond to a common reactivity of developers around the same design theme, or inter-thematic and may be related to quoting strategy of some participants who reply to all messages in the order at which they arrive.

6.3. Design view

The results discussed so far pertain to a structural analysis of the messages and how participants contribute to the discussions. We turn now to a content analysis of the messages. Our content analysis is focused on the design alternatives and the design activities of the project participants.

6.3.1. Design alternatives

In the PEP 279 discussion, 34 alternatives were proposed (23 for theme 1, 8 for theme 2, 3 for theme 5). Fewer alternatives (indeed, only three) were proposed in the PEP 285 discussion. We have investigated how the number of alternatives discussed within a message is correlated to several variables related to the dynamics of OSS design; specifically, position in the graph, the emerging role of the poster, and the structure of message. To examine these associations, we divided the set of messages according to the median number of alternatives discussed within the message. The median was 1.

There is a strong relation between the position in the graph and the number of alternatives discussed in a message ($V^2$ Cramer = 0.18). RD reveals that branching structures tend to be occupied by messages dealing with more than one alternative (9/20, 45%); and closing messages include only one or zero alternatives (23/38, 61%).

There is an association between the emerging role of the author and the number of messages posted dealing with alternatives ($V^2$ Cramer = 0.13). In PEP 279 discussion, the RD measurement shows that messages posted by the project leader (6/19, 32%), the PEP champion (5/19, 13%) and LP-D (4/19, 10%) contain the highest number of alternatives. The HP-Ds tend to mostly post messages with one or zero alternatives (11/39, 28%); e.g., HP-A (9/39, 23%).

These results confirm the importance of the project leader and the PEP champion in the design process: posting messages that deal with several alternatives suggests that they bring a broader vision to the problem. The presence of LP-D is interesting because it suggests that even low contributors can enrich the design process by proposing or discussing alternatives. This result has to be compared with the results presented above: for PEP 285, LP-Ds are much more present in begin branching structures (and are quoted several time by the others).

6.3.2. Design activities in quotes and comments

6.3.2.1. Activities in comments. The distribution of the activities in comments is different in the two discussions ($\chi^2 = 64, 6, df = 6, p < 0.001$) (Fig. 17). Nevertheless, for both the PEP 279 and PEP 285 discussion, the most frequent activity in comments is evaluation (124/218, 57% in PEP 279 and 89/190, 47% for PEP 285). Clarification seems to be much more important in the PEP 285 discussion (74/190, 39% as compared to 10/177, 13% in PEP 279). This is understandable when one considers the main theme of PEP 285, an issue with potentially far-reaching consequences for the whole Python language community. Thus, comments concerning this PEP need explanation so that their larger
consequences can be explored and considered by the discussants.

The main differences between the two discussions are as follows: compared to the PEP 279 discussion, the PEP 285 discussion contained fewer proposals and neither decisions nor syntheses. Fewer design alternatives were proposed because the design themes were de facto closed: developers were asked to choose between already existing alternatives or to address the meta-theme. Moreover, developers produced many arguments and explanations during the discussion, perhaps, implying that a synthesis was more complex to achieve. The lack of decisions taken in the PEP 285 discussion is due to the fact that the discussion stopped and then continued at a later date. In the second round of discussions, decisions were taken. A few, explicit decisions were taken concerning PEP 279. We have found that this is one general difference between electronically mediated discussions and face-to-face discussions. In face-to-face design situations, decisions are taken in a more implicit way and are marked, for example, by a change in the topic of discussions; participants thus infer that a decision has been taken.

Globally, there is no association between status and involvement and activities identified in comments ($\chi^2 = 0.04$ for the two discussions). However, the champion is the only provider of syntheses (e.g., message 70 of PEP 279; see extract below).

**Extract from PEP 279**

*And, the project leader is the only person who makes decisions. Both of these relations are obviously due to the status of the project leader and the champion.*

6.3.3. Sequences of activities

There are strong associations between activities in quotes and activities in comments ($\chi^2$ Cramer = 0.69 for PEP 285; and 0.23 for PEP 279). The RD measurements enable us identify the following meaningful sequences:

- An evaluation tends to be followed by another evaluation. An agreement (i.e., a positive evaluation) can be followed by a disagreement (a negative evaluation), which highlights diverging points between participants; or, can be followed by an agreement, which highlights converging points between participants.
- A proposal tends to be followed by evaluation or by other proposals. The latter sequence implies an implicit disagreement between participants.
- A clarification tends to be followed by other clarifications; this pattern characterizes the construction of common ground between designers.
- A synthesis tends to be followed by a proposal, which can be seen as an implicit disagreement between designers. A synthesis can also be followed by agreements. The former sequence suggests that syntheses are “design motors” which spur innovation in the form of alternative proposals.
- A coordination tends to be followed by another coordination activity.
- A decision tends to be followed by coordination, which corresponds to the task assignments implied by the decision. There are also sequences, such as “decision-proposal”, that can be explained as a late reply (an alternative proposal) from geographically distant participants. We have seen few explicit decisions that might explain this result.

All these sequences highlight the way that activities appeared online and characterize the way that knowledge is produced in this OSS design community.

7. Discussion

In this section, we primarily discuss the main results obtained concerning the social, thematic temporal, and design dynamics of the OSS design process in the Python community. We conclude this section by synthesizing the main contributions of this paper towards the development of knowledge about OSS design.

7.1. Social dynamics

7.1.1. The champion and the project leader

In all of the discussions, the project leader and the champion of the PEP are frequent contributors and post messages that lead to multiple branches. The project leader

![Fig. 18. Distribution of activities in quotes PEP 279 and PEP 285.](image-url)
tends to post multi-quote messages; to close discussions; and to make decisions. The champion is the only discussant who writes syntheses of sets of previously posted messages.

That the project leader and champion are important to the discussions is understandable: by definition, a champion is in charge of his PEP discussion; and the project leader acts as guarantor for the project. Mahendran’s previous results (2002) highlighted the project leader’s authority over the community. Our results confirm Mahendran’s findings. To complete this result, it may be interesting to investigate how the champion’s style might affect the design process.

7.1.2. Administrators and developers

In the PEP discussions, the messages posted by administrators tended to occur in the three different positions: (1) at a beginning branching position; (2) in linear sequences of exchanges with developers; and (3) in closing positions (PEP 285), when the project leader (or the champion for PEP 279) stopped participating in the discussion. In the PEP 285 discussion, administrators are only present in the meta-theme discussion. Such results suggest that complementary roles are occupied alternately by the project leader and the administrators, the latter relying on the project leader when the theme is a crucial one for the Python language and filling in for him when he does not want (or cannot) participate in the discussion anymore. This again confirms Mahendran’s (2002) findings in which he notes how power is distributed between the project leader and the administrators.

In the PEP discussions, the messages posted by developers tended to occur in two different positions: (1) in PEP 285, at a beginning branching position with deep quotations of exchanges with developers; and (2) in PEP 279, in linear sequences of exchanges with administrators and others developers. Discussions are not closed to developers even if discussions are framed by the project leader, the administrators, and the champion of a PEP. Developers do participate in the design process. Their messages contain design alternatives and they can initiate branching structures in the discussions.

This match between statuses and involvement in a design-oriented online community can be explained by the fact that the task faced by participants in the python-dev mailing list is very clear and focused. The more open mailing list, python-list, in which not only the language designers but all of the broader Python community is invited to participate would be interesting to analyze for the sake of comparison. Such a future analysis might make clear the emergence of roles enacted by members of the broader community who do not have formally assigned position in the project.

7.2. Thematic temporal dynamics

7.2.1. A general and selective use of quotation in design-oriented, online discussions

Our analysis of the PEPs online discussions between designers has shown how commonly quotations are used in design-oriented, online discussions. However, this use is clearly selective: designers select design-relevant quotes from previous messages and compose complex messages where quotes come from different messages. These results extend the ones from the Computer-Mediated-Communication field (Eklundh and Macdonald, 1994; Herring, 1999; Eklundh and Rodriguez, 2004) showing that quotation is a common strategy in open, online discussions. In the following, we will enumerate the major differences between open and design-oriented, online discussions.

7.2.2. Design-oriented, online discussions are thematically constrained

Design-oriented, online discussions are thematically focused. In contrast, CMC research (e.g., Herring, 1999) has shown that in open online discussions themes disappear over time.

The OSS design discussions we have analyzed have a characteristic pattern to them. First messages are PEP-related multi-thematic ones. After a series of exchanges, messages tend to become specialized and focused on a single theme of discussion. These single themes are then discussed in detail. We have, however, seen two variants that do not follow this general pattern: a meta-theme (PEP 285), and a thematic drift (PEP 279).

7.2.3. Design-oriented online discussions are quasi-synchronous discussions

In our study we have seen that themes of discussion persist longer if they are clearly related to some central aspect of a PEP. But even the themes of short persistence in the design discussions we observed do not dissipate and get replaced by non-design-oriented themes as they might in completely open, online discussions (Herring, 1999).

Single-theme design discussions are sometimes sustained for days. In these single-theme, multi-day discussions we have observed moments of quasi-synchronous interchange in which multiple messages are received within an hour’s time. These moments reflect implicit social rules of the design community (Conen, 2004; Cohendet et al., 2000; Stewart and Gosain, 2006); rules that concern what is considered “timely” for the community. A designer aiming at participating in the design theme discussion has to do it “in time” or risk being chastised by the project leader. These implicit rules might also be seen as pragmatic linguistic rules, similar to adjacency and relevance in face-to-face interactions: they imply the existence of acceptable moments to participate in design-theme discussions. Indeed, participants in online interactions develop mutual expectation for responsiveness. As set up by Tyler and Tang (2003) “people project responsiveness images to each other, and they calibrate their email behaviours to mirror rhythms of their correspondents” (p. 253).
7.3. Design dynamics

7.3.1. Activities in online design-oriented discussions

The two most important activities in face-to-face design meetings are (i) evaluations of design alternatives and criteria; and (ii) clarification (Olson et al., 1992; d’Astous et al., 2004). In distant and asynchronous discussions between designers, we have seen that evaluation activity remains the most important activity. Clarification is also present in design-oriented, online discussions, but it seems to be controlled by the project leader. We speculate that the project leader attempts to distinguish necessary explanations from explanations that are not crucial to the design process. He defers explanations considered as off-topic to other threads of discussion. This suggests that specific sub-spaces exist in the discussion space: focused discussion design sub-spaces as compared to sub-spaces devoted to the construction of common ground between the participants. Common ground between participants is crucial for the success of geographically distributed design (Olson and Olson, 2000; Carroll et al., 2003). It is likely that community members also elaborate common ground during face-to-face exchanges at international conferences (Barcellini et al., 2006).

In the two PEP discussions, few coordination activities were observed. We assume that this is the case because the PEP discussion rules presuppose that task allocations are accomplished in other spaces of exchange. We also observed that few decisions are taken during the online discussions, but those decisions taken are explicit. In face-to-face design meetings many decisions are made in an implicit manner (Marty and Darses, 2001). This could be one of the main differences between face-to-face and distributed, online design meetings.

For the most part, a participant’s status in the project does not affect the kinds of design activities engaged in by the participant with the exception that certain activities are strictly tied to very specific statuses. For example, only the project leader can make an explicit design decision and the Champion is the only provider of synthesis. Despite the fact that discussions are informally moderated, there are real interactions between developers, administrators and the project leader about the design problems faced by the community.

7.3.2. Sequence of activities in design-oriented online discussions

Patterns of interchange between designers exchanging messages in an asynchronous, online environment largely resemble the patterns of exchange observed in face-to-face design meetings (d’Astous et al., 2001, 2004). For example, an evaluation comment is often followed by another evaluation comment; a pattern that indicates that converging on or diverging from a solution. Propositions are frequently followed by evaluations in both online and face-to-face design discussions (e.g., Visser, 2004; Stempfel and Badke-Schaub, 2002). A clarification followed by another clarification was a pattern we observed in the thematic drift and the meta-theme subdiscussions. This pattern is a mechanism for constructing and sharing knowledge in the community. These observed, conversational patterns substantiate, in a more fine-grain fashion, previous work on the ways that knowledge is produced in an epistemic community (Cohendet et al., 2000; Conein, 2004).

7.4. Synthesis

This paper is a first attempt to characterize – at a fine-grain level – the collaborative design dynamics of an OSS project. To investigate the social, thematic temporal, and design dynamics we have developed a method combining structural analyses (position in the discussion, quotation structures, synchronism) with a content analysis of messages in design-oriented, online discussions.

The design-related discussions studied are focused and framed by design themes and have a specific temporality. Over the course of a design discussion, thematic coherence tends to be refined, rather than to drift as it does in many general, online, non-design discussions. The OSS design discussions we observed also frequently include syntheses that are constructed through a process of community consensus. Implicit communications rules (such as the expectation of responsiveness) frame the temporality of discussions. Some of these implicit rules of temporality lead to quasi-synchronous moments of interaction. Specific participants in the project do this framing: the leader and/or the administrators, and the champion of the design issue being discussed. Some of these participants performed specific roles online: for example, champions frequently authored discussion syntheses; the project leader often makes explicit decisions and moderates the discussion by disallowing off-topic posts.

Although discussions are actually framed by the “top hierarchy” of the project (project leader and administrators), they are not closed: other participants – so called “developers” – do enhance the design process by proposing alternative solutions; evaluating those solutions; and more generally, proposing Python Enhancements Proposals (PEPs). Our results problematize some practitioners’ models of OSS development, such as the bazaar model (for OSS) vs. the cathedral model (for proprietary software) (Raymond, 1999). Indeed, as other researchers have argued (e.g., Nakakoji et al., 2002), OSS projects are far from an open bazaar model. Major project, especially, have much more circumscribed – cathedral-style – design processes.

Our research and the method we have developed are empirical means to explore the socio-cognitive mechanisms behind OSS design and to characterize how knowledge and artefacts (specifically, software) are collectively elaborated in an epistemic, exploration-oriented, OSS community.

The specificity of OSS design process may not be in the frequently-invoked opposition of bazaar vs. cathedral styles of design, but rather in the opportunities given to
members of a project to participate in the design process in many different roles. Within the socio-technical environments of OSS we observed participants may create and perform many different roles throughout the design process.

8. Research implications

We consider how the results of our case study might be generalized by automating some parts of our methodology. We also examine how the research may be extended by taking into account not just the designers and developers but also the users of the Python programming language.

8.1. Automation of the method

We have analyzed two corpora of online OSS design discussions. The type of analysis “by hand” we have performed for the current work is extremely time-consuming. To generalize and extend these preliminary results, we need to repeat our work on other corpora of the Python community, but also in other OSS communities. Consequently, we plan to automate some of our structural analyses using a tool based on the Conversation Map (Sack, 2000), a piece of software designed and implemented by one of the co-authors for automatically analyzing and visualizing the semantics and social networks of large-scale online discussions. The tool will automatically extract quotations and build the discussion graphs employed in the current work. Our quotation-based representation might be enriched by several user-adaptable functionalities inspired by our three types of analysis. By automating some of the methods of our analysis, one might build a tool to display, for instance, quoted messages, messages with a depth of quote superior to 2, or beginning, branching messages that might be pivotal; e.g., those posted by the project leader, the champion, or the administrators; messages containing multiple quotations; messages that have been deeply quoted in discussions; etc. A tool designed to perform our temporal analysis might display only messages that have been quoted in the past day; or messages that have led to major synchronicity. All these characteristics can be found automatically in discussions.

These analyses can be enriched by social network analyses (SNA), especially in the comparison, for instance, of the social position observed in the quotation-based representation with the centrality of a participant, as measured according to techniques from SNA.

The content analysis part of our method categorizes message comments as atomic activities or predicates. This methodology could be extended by making explicit the arguments for or against design alternatives in the discussions. This could entail semi-automated or fully automated content analysis of the messages. A semi-automated approach would require researchers to tag messages to categorize the content expressed in the messages (Kirschner et al., 2003). The tool, then, could display sequences of quotations and comments that are linked to argumentation. The main shortcoming of this approach is that it creates an added task for the researcher. The second approach would be to construct an automatic discourse tagger to analyze automatically the themes of discussion and patterns of argumentation, an admittedly difficult task akin to rhetorical structure parsing (Marcu, 1997).

This automation would provide the means necessary to articulate the evaluative criteria employed by the designers and so lead to the reconstruction of a design rationale, i.e., the reasoning behind the design of a software artefact (Buckingham Shum and Hammond, 1994; Concklin and Burgess Yakemovic, 1991; Moran and Carroll, 1996). Design rationale tools aim at capturing the rationale behind the design choices, problems, solutions, and the design criteria employed to select or reject a solution. These types of automated tools may be useful for both researchers and for members of the design communities. Indeed, others have argued (Ripoche and Sansonnet, 2006; Gasser et al., 2003; Scacchi, 2001) that there is a lack of tools, especially content analysis tools, to support fine-grain research on online communities. Organizational memory tools (Sauvagnac and Falzon, 2003) could be based on visualizations of online discussions to facilitate their analysis and reuse. The goal of organizational memory tools is to capitalize and formalize knowledge in an organization. These kinds of tools could be useful for members of OSS communities—especially leaders or newcomers or lurkers—to find relevant information.

Design rationale tools could be used by researchers to analyze the collective reasoning and interactions of the community. By making their rationales explicit, the members of design communities could be able to keep track of past decisions and communicate these rationales to others outside the design team.

8.2. Future research

Beyond the design PEP discussions we have analyzed, we plan to analyze other types of PEP discussions (meta-PEPs, rejected PEPs, still open PEPs, PEPs with different champions, etc.) that took place during the same time period (i.e., the first half of 2002). We anticipate that by analyzing multiple corpora we will be able to more precisely characterize the relative importance of design activities in OSS design communities.

We also plan to analyze pre-PEP discussions in order to obtain a broader view of the design process and to better understand which parts of the design process are (or are not) captured by the PEP process: identification of needs, work preparation, specifications, rationale. This will help in positioning these interactions in the frameworks of team interactions (Rousseau et al., 2006).

In the present research, we focused on the discussions directly related to PEPs that took place on the python-dev mailing list; i.e., the developer-oriented community. Thus,
an extension of this research will be to understand the entire design process in the broader community: from the emergence of a need of the user and/or developer communities, through its acceptance by the communities, its formalization in a PEP, and its implementation. We have begun to investigate this design-use mediation process by focusing on a need that emerged from an application domain of Python and that was then relayed to the user and developer communities by a participant who took part in both the python-dev and python-list mailing lists (Barcellini et al., 2006). The python-list is concerned more with the needs of users rather than those of the developers of Python. Studying the design-use mediation process can lead both to increased usability of OSS (Nichols and Twidle, 2003) and also contribute towards new insights into participatory design methodologies (Schuler and Namioka, 1993). In particular, it can inform to what extent OSS’s socio-technical environments enable or inhibit various forms of participation. Finally, we intend to extend this research to the broader Python community and thus to explore the links between the core Python community and other service-oriented and utility-oriented OSS projects that build on or use the Python programming language. To reach these goals, we need to both automate some of our methods but also to extend our work to better understand a number of other user and design communities.

OSS project participants meet face-to-face as well as online (for example, in international conferences such as OSCON, PYCON, and Europython). In order to capture a part of these offline interactions, we plan to complement our methods with interviews of members of the Python community. The data we analyzed are only those captured in the online discussion spaces. Obviously, these data are only a part of the reality of OSS participant interactions (Paccagnella, 1997), even if they are an important part of that reality.

Acknowledgements

This study was supported by the France-Berkeley Fund; the French TCAN-CNRS program; and the National Science Foundation, Directorate for Computer and Information Science and Engineering, Division of Information and Intelligent Systems, Digital Society and Technologies Program, Award 0416353.

The authors thank the reviewers warmly for their helpful comments and advice.

References


Herbsleb, J.D., Klein, H., Olson, G.M., Brunner, H., Olson, J.S., Harding, J., 1995. Object-oriented analysis and design in software project teams.

Human–Computer Interaction 10, 249–292.


