Intersubjectivity in Open Source Software Development: Expansive Grounding in Distributed Work

Pål Fugelli
Research Computing Services
University of Oslo
Oslo, Norway
pal.fugelli@gmail.com

Leif C. Lahn
Department of Educational Research
University of Oslo
Oslo, Norway
l.c.lahn@ped.uio.no

Anders I. Mørch
InterMedia
University of Oslo
Oslo, Norway
a.i.morch@intermedia.uio.no

ABSTRACT
In this paper we address intersubjectivity, which is of particular relevance to those who wish to understand how learning is produced within complex group work. A conceptual framework for intersubjectivity is developed based on a review of past research with a starting point in the work of Rommetveit, a Norwegian social psychologist. The framework is applied to the analysis of a mature open source software project (mod_perl module of Apache Web server) with a focus on how the ‘tacit’ or otherwise hidden information in the central and peripheral regions of the developer network may contribute to understanding intersubjectivity as an expansive process. Based on observation of 215 participants’ contributions on the project’s mailing list over a period of 6 months, we explore how processes of intersubjectivity evolve around shared objects of development. We conclude with a discussion of how vague and imprecise utterances in post- and reply messages may be significant and trigger the co-construction of shared understandings across the developer network.

Categories and Subject Descriptors
H.5.3 [Information Interfaces and Presentation]: Group and Organizational Interfaces – asynchronous interaction, evaluation/methodology. K.4.3 [Computers and Society]: Organizational Impacts – computer-supported collaborative work

General Terms
Documentation, Human Factors, Theory.

Keywords
Collaboration, empirical study, grounding, intersubjectivity, prolepsis, shared objects, software development.

1. INTRODUCTION
The open-source model in software development is considered by Miettinen (2006) and others (Hippel and Krogh, 2003) to be a relatively new form of community-based, distributed knowledge creation activity, characterized by high complexity and demarcation between core members and peripheral participants.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Conference’10, Month 1–2, 2010, City, State, Country.
Copyright 2010 ACM 1-58113-000-0/00/0010...$10.00.

The complexity of creating software together at a distance makes learning a necessity, both for experts and newcomers. Newcomers must learn to get into the project (Edwards, 2001), and the experts (core members) must learn to sustain common understandings. An example of open source development (OSD) is the Apache mod_perl project, consisting of a worldwide network of more or less dedicated collaborators. The software object to which they are committed is the development of a module for the Apache Web server that integrates Perl programming with the Apache Application Programmer Interface (API). This module allows much flexibility for web developers and programmers. It is popular and constantly changing and unfolding as new patches are submitted and upgrades are released by the project at a high frequency. Since its first release in April 1995, Apache has become the most popular server software of the Web. The project is described as “a collaborative software development effort aimed at creating a robust, commercial-grade, featureful, and freely-available source code implementation of an HTTP (Web) server” (Apache.org 2006).

In order to interact and work on a common set of objects in such a distributed software development community, it can be argued that the participants must share “some kind” of understanding and that intersubjectivity is involved in the process. We find the expression “some kind” in the previous sentence, a rather appropriate description, since the research literature on collaborative work and learning often leans on general models of human communication that do not address questions like: How much and what kind of intersubjectivity is needed in order to get things going? Is consensus a prerequisite for joint efforts? How is intersubjectivity achieved, and what role does symbolic mediation play? The latter must be addressed when we study knowledge creation in distributed work and the previous work we survey below and our main assumption for further exploration is that new models must depart from models of face-face interaction and take into account the unique requirements of working at distance. Researchers have acknowledged this is CSCW and computer-supported collaborative learning (CSCL). For example Stahl, Koschmann, and Suthers (2006) suggest that the problem of intersubjectivity is of particular relevance for those who wish to understand how learning is produced in interaction, advocating for more in-depth interdisciplinary research, and when studying interaction in a large virtual world (Second Life), Shami, Erickson & Kellogg (2011) found that implicit grounding, i.e. common ground that is established before conversation actually begins, can be supported by affordances built into the virtual environment that extends the face to face situations that participants are familiar with (e.g. a virtual poster session with avatars that look like people they want to meet in their example).
The paper is organized as follows. We develop a conceptual framework of intersubjectivity based on a review of two theoretical positions that have broken with the classical logistic (Shannon-Weaver) model of human communication (sender-message-receiver), but at the same time are challenged by the distributed character of online work. The first is Rommetveit’s (1974; 1979) notion of “prolepsis,” which is central to our discussion of collaboration in open-source development networks. We compare it with the concept of grounding as described within the “extended contribution model” proposed by Clark and Brennan (1991). In the empirical section we present data representing a six-months logging of online communication among subscribers of the mod_perl open source project mailing lists.

2. CONCEPTUAL FRAMEWORK
2.1 Intersubjectivity and Prolepsis

Nearly 40 years ago Ragnar Rommetveit, a Norwegian social psychologist, presented a phenomenological reinterpretation of the Shannon-Weaver sender-receiver model of communication (Rommetveit, 1974) as a reaction to the dilemma of distinguishing private and shared worlds (Rommetveit, 1979; p. 94). In essence, it says that communicators’ perspectives of the world can only be partial, never complete. They are considered to draw on a presupposed, shared reality. Between the ‘I’ and ‘You’ in Figure 1, potential states of intersubjectivity can emerge along spatial-temporal coordinates in three dimensions. It is an expansive space of communication within which intersubjectivity can emerge (drawing on past and future events, and the surrounding spatial context). This allows us to consider not only the objects of speech acts (explicit knowledge) but also their embeddings in the communicators tacit knowing, along two dimensions: here/there and before/afterwards. Utterances get their meaning through a process of joint construction that draws on this tacit three-dimensional space, which in Rommetveit’s (1979) words is described as mutually accessible social realities that each of the participants bring to the situation.

Figure 1. The spatial-temporal-interpersonal coordinates of the act of speech defining the emergence of intersubjectivity (Rommetveit, 1979)

Rommetveit argues, “the temporarily shared social world is in part based upon premises tacitly induced by the speaker” (Rommetveit, 1979; p. 87). The ‘tacitly induced’ in this context means that some elements are left out of the conversation and the listener is subsequently invited to step into a partially shared space of intersubjectivity in order to make sense of the left out elements. Rommetveit goes as far as saying the process entails the listener as an insider of a tacitly expanded and enriched ‘here and now’. To put words on this process he introduces the notion of *shared prolepses* (Rommetveit, 1979), which is a communicative act among participants in conversation where the speaker presupposes or takes for granted something that has not yet been discussed. The term prolepsis dates from ancient Greek writings, meaning preconception or anticipation. In the framework above, this term was applied to explain how intersubjectivity ‘here and now’ includes a blend of experience (before) and anticipation (afterwards) brought to the situation, and visualized by the curved lines (expansive space) in Figure 1. In this manner, shared prolepsis can provide an important concept for understanding grounding in distributed, asynchronous collaboration environments, as it is both future oriented and based on past events.

Moreover, the formation of prolepsis is seen as a dynamic and recursive process in the sense that the actors continually fine-tune their assumptions and anticipations in reaction to the others’ contributions and feedback (Matusov, 2001). Rommetveit (1974) offers the following example of prolepsis in a personal letter received from a friend:

"Today, I walked with one of the psychologists here past the Mayflower cinema in Eugene, where Bergman’s latest film movie is being shown. He asked me whether I had seen it. I said no, and asked if he had. He said yes, he had. I asked him how he liked it, and he said ‘I liked it very much, but Mary Ann did not’; without ever explicitly having ‘made known’ to me that he is married and that his wife’s name is Mary Ann, that they went to see the film together, and a lot of other things – and (if I am correct) without assuming that I knew all this. His utterance was proleptic in that it triggered a search on my part for a shared social reality, which in turn would provide a basis for understanding the sentence. Incidentally, it would have been barbaric and pedantic to say, ‘Oh, Mary Ann is your wife’. To be precise, prolepsis here served to establish a relationship between his wife and me as persons who should at some time get together. My comment would have been a crude rejection of that implication” (Rommetveit, 1974, p. 87-88).

In this story the fact that the friend has a wife and her name is Mary Ann is presupposed but not introduced. This may challenge the listener and “trigger anticipatory comprehension, and what is made known will hence necessarily transcend what is said” (p. 88). As an example of prolepsis it may challenge the receiver and trigger the construction of new understandings of the intended meaning. Prolepsis can then serve as a catalyst for making sense of not-yet-provided information, in this case to prepare for a future meeting of three people. Its role in intersubjectivity is to provide prompts or cues for building a special kind of common ground, whereby the issuer of utterances “presuppose shared knowledge that has not yet been introduced into the interaction, but which is essential to making utterances interpretable” (Cole, 2002, p. 312). In this way, proleptic instances may enhance the progression towards greater intersubjectivity by providing access to each other’s private worlds (Rommetveit, 1974; 1979), which again may direct the construction of common understanding.

The process of omitting information combined with implicit cues invites the listener to participate in the co-construction of an expanded intersubjective space (Rommetveit, 1974). As a consequence, the receiver of proleptic utterances can actively participate by “filling in” the missing pieces and thus co-produce the information needed to achieve a sufficient level of common...
understanding. Assisted by a more competent other, or perhaps a computer system, this process goes on in parallel with communication. If successful, the ‘private worlds’ of participants may expand so as to intersect and form a greater common space, a partially shared intersubjective state. Using proleptic states as building blocks for conversational grounding, has not been explored in the type of intersubjectivity proposed by Clark and colleagues, which are the most frequently cited work on grounding in CSCW and CSCL. We describe the related work below and compare it with Rommetveit's approach. We will later use the conceptual framework to study processes of constructing a shared knowledge object in the mod_perl open source development network. We claim the proleptic approach to intersubjectivity provides new insight for understanding the mechanics (sub processes) of grounding in distributed work.

2.2 The Extended Contribution Theory

Clark and Schaefer (1989) proposed a collaborative model for human communication based on grounding, referred to as the contribution model. The model revises a more traditional sender/receiver perspective on communication first proposed by Shannon and Weaver (1949), broadening the analytic frame from single utterances to contributions developed in interaction. Clark and Brennan (1991) extended this model and postulated that “all collective actions are built on a common ground and its accumulation” (p. 222). To coordinate the content and process of human conducts, participants must update their common ground moment by moment – in what they refer to as grounding, and the grounding processes vary according to the context of interaction.

Clark and Brennan (1991) claim that different media put different constraints on grounding processes. For example, real time communication media like videoconferencing, offer visibility and co-temporality, whereas e-mail or personal letters lack the same features, causing delays in turn taking for proper understanding. They suggest, “people should ground with those techniques available in a medium that leads to the least collaborative effort” (Clark & Brennan, 1991). According to Rommetveit and his notion of prolepsis, such a shared understanding is only partially attained and depends on historical information (common pre-understanding).

Baker et al. (1998) applied Clark and colleagues (1989; 1991) grounding model to understand grounding and intersubjectivity in collaborative learning tasks. They used aspects of cultural historical activity theory (CHAT) and Matusov’s (1996) notion of disagreement to inform the analysis. They define intersubjectivity as coordination of contributions in joint activity rather than as overlap of different understandings. Central to sociocultural theory and CHAT is the idea of collective subjectivity as essentially object mediated, which we claim is closer to the notion of interobjectivity as defined by (Latour, 1996).

Suthers (2006) proposes intersubjective learning as a thematic agenda for the future of CSCL. He developed a conceptual framework for collaborative knowledge construction that integrates Clark and Brennan’s notion of “uptake” with the framework of knowledge building (Scardamalia & Bereiter, 1994; Stahl, 2000). Suther’s calls it collaborative knowledge construction, which he defines as “accretion of interpretations of a dynamically evolving context”, i.e. one participant takes up another’s contribution and does something further with it. This can take place in a discussion forum when the participants reuse and extend a shared space of contributions. For example, in a knowledge-building environment FLE, the participants make use of existing postings to clarify their ideas by questions, personal answers and scientific explanations (Fugelli, 2004). In OSD, the contributors rely on latest versions of software to propose new patches to improve upon them. Our work attempts to extend the previous work on intersubjectivity by adopting the notion of prolepsis and nesting to propose an alternative to the cumulative process of uptake (Clark & Brennan, 1991; Suthers, 2006).

2.3 A Distributed Model of Shared Prolepsis

Computer mediated communication introduces aspects that were not taken into consideration in the framework for intersubjectivity proposed by Rommetveit. In the revised framework we present here and depict in Figure 2, the x-axis is a path for the evolution of a shared object of attention, which we refer to as “knowledge object” (or “CSCL” for short), and the y-axis a distributed space and time dimension (“CSCW” for short).

![Figure 2. Framework for intersubjectivity and shared prolepsis in mediated expert networks (Fugelli, 2010)](image)

The horizontal axis of this model represents the creation of common understanding within a group. By distinguishing a pre-proleptic stage from a proleptic instance, we differentiate this dimension further. Based on the previous work in OSD (see below), two groups of developers participate in OSD projects: novice developers (peripheral participants) and expert developers (core members). Therefore, we suggest that shared prolepsis ought to pass through two phases: 1) pre-proleptic stage, involving communication between novice and expert developers, and 2) proleptic instances, communication among expert developers. For the pre-proleptic utterances to evolve into proleptic instances, they have to be taken up by more competent developers. This presupposes some kind of shared understanding about some problematic issue identifiable by the novice developers, but not yet introduced in the expert forum. This process is represented as a ‘spot of shared attention,’ moving along the horizontal axis in Figure 2, from more or less vague utterances towards clarification. The latter is manifest as anticipatory comprehension among the more competent participants, realized as prompts that challenges and boosts the construction of a greater intersubjective space.

The intersection of the two axes, indicated by the intersection of the dotted lines in the lower left and upper right quadrangles, define the expansive intersubjectivity space. In here, the coordinates ‘I and You’ in Rommetveit’s original framework are...
replaced by “issuer of initial Problem” (iiP) and “more competent Participant” (mcP). This is further discussed in section 5.

2.4 Open Source Development

Our application domain is open source development (OSD). There is little research that explicitly links OSD and “intersubjectivity.” We survey previous work related to interaction patterns and the process of achieving collaboration and intersubjectivity. The latter line of research addresses the issue of using different resources to accomplish work.

Bergquist and Ljungberg (2001) conducted an empirical study of open source developers for a period of two years and identified a phenomenon they labeled “gift giving” culture. They found that developers communally gave and appreciated gifts, and that the open source developer community relies on gift giving as a way of “getting new ideas and prototypes out into circulation.” Gifts were not physical commodities but associated with knowledge, source code and peer reviews (e.g. abstract entities or cultural artifacts). They claim that the sharing of these resources provides a key for understanding the open source community’s cultural foundations as well as its social organization (Bergquist & Ljungberg, 2001).

Network-level studies of open source development projects have reported that interaction patterns among participants vary depending on their roles. Most notably are the roles of core developers vs. peripheral participants (Crowston et al., 2006; Edwards, 2001; Long, 2006). In one study, Long and Stiau (2007) applied social network analysis to three projects accessible in SourceForge.net. They report that the interaction pattern within these projects evolves from a single hub at the beginning of a software project’s life cycle into a core-periphery model as the project matures. In our study, another approach is proposed while using the core-periphery pattern as a network-level contrast for analyzing message content targeting processes of intersubjectivity.

In a study on how software engineers’ communicate in geographically dispersed teams by Cataldo and Herbsleb, a core group of developers emerged as the liaisons (or gatekeepers) between formally defined teams (Cataldo & Herbsleb, 2008). Their findings revealed that “individuals in the core not only perform a critical communication role but also they are the top contributors to the actual development effort” (p. 1). The general core-periphery pattern in software developers’ communication networks is also corroborated by our study. However, our analytic attention is rather on how core- and peripheral participants take on different roles in the construction of shared understandings rather than measuring communication roles and levels of productivity.

Edwards (2001) studied the transition from peripheral to core participation in OSD using “community of practice” as a theoretical framework. He found that situated learning is essential for newcomers to become core members. However, it was not clear from his study how many peripheral members were in fact able become core members. He also found that all contributors, but especially the core members, rely on a common frame of reference for coordinating their joint effort. Much of this was not a conscious effort but a consequence of maintaining membership in the community. The peripheral participants contributed to user support mailing lists (e.g. how to download and install a system), whereas the core members contribute to the technical discussion lists and propose patches to improve the software. The learning curve was steep for newcomers, and the motivation of core members to sustain membership was associated with the joy of solving software related problems (Edwards, 2001). Our study reveals a clear separation of core and peripheral participants but little transition from periphery to nucleus.

There are studies of global online teams other than OSD projects that use organization structures that compare with OSD, like management of large scale engineering projects. They reveal that distributed teams have structures that may easily dissolve because of high demands of coordination and “centrifugal forces” that pull teams apart. Lanzara and Morner (2005) and Hemetsberger and Reinhardt (2009) contrast these findings with the persisting cohesion typical of open source collaboration. They attribute the success of coordination in OSD to a number of technological artifacts that define the process. The source code is the main artifact and serves as a coordination mechanism by providing a modular (part/whole nesting) structure for organizing the mailing lists, and formulating contractual rules in licensing agreements. The combination of a well-defined agenda for what to do and a task structure that facilitate the recruitment of newcomers ensures stability and progress. Approaches to enroll newcomers in the activities of core members are for example core members reviewing the progress of newcomers performing “junior jobs” like bug fixing (Hemetsberger & Reinhardt, 2009).

These studies show how open source development projects are collaborative, knowledge intensive, and require learning, both for newcomers and seasoned programmers. The interaction is organized around artifacts, and there are two main types: primary and secondary. Primary artifacts are source code, executable releases for download and mailing lists, and secondary artifacts are the tools needed for operating on the primary artifacts: programming environments, programming languages, web browsers, debuggers, version control systems, and so on. This infrastructure provides a system of resources that are often implicitly used and brought into the intersubjective field of open source projects. Our goal is to explore how the concept of “shared prolepsis” can be used to better understand open source development processes. To what extent does interaction in such contexts draw on the above mentioned resources as part of a shared background, and what are the “shared objects” that serve as cues and markers to monitor and guide crucial operations in the environment? We also address how peripheral participants contribute to building common understanding by expanding the intersubjectivity space.

3. METHODOLOGICAL APPROACH

We used a mixed methods approach to address our research questions, combining quantitative (social network) and qualitative (message content) analysis. All data were obtained from the mod_perl project mailing lists related to the development of the mod_perl core, which are publically available as a text-based archive from http://mail-archives.apache.org/mod_mbox/perl-dev/. Over a period of six months the first author observed the text-mediated communication among subscribers of these mailing lists, reading through previous postings and following the discussion threads without disturbing the ‘naturally occurring’ communication in the project. Mann and Stewart (2000) refer to such a practice as lurking and suggest that it as an advantage of Internet research; though ethical issues may arise when collecting online communication as the research involves human subjects. Researchers posting or ‘lurking’ in online communities might be perceived as intruders by the participants (Eysenbach & Till, 2001). With respect to the transparency and public nature of the
mod_perl project, we decided to send an informative letter introducing ourselves and explaining the study’s purposes (Fugelli, 2010).

3.1 Participants and Procedure
Since Gisle Aas released the first version of mod_perl on March 25, 1996, thousands of individuals have contributed to its current state of development (http://perl.apache.org/about/history.html). In order to strike a good balance between a manageable and rich data set, we decided to include all members within a time frame of six months. Another reason for this choice is that in a global open-source project of this magnitude, over time, new developers tend to join and others leave. By including all members within a fixed time interval, we get a ‘snapshot’ of participation in this period. The study period lasted from 01.01.2008 to 30.05.2008, and included 215 individual contributors. The number of postings during this period was 1154, with a mean of 6.2 postings per contributor.

3.2 Data Analysis
At the first descriptive level of analysis, formal measures associated with Social Network Analysis (SNA) (Wassermann & Faust, 1994; Scott, 2000) were applied to render the overall communication among participants in the mod_perl project visible. The main focus on this level of analysis was to organize the message exchange into sub-groups to serve as contrasts for a more detailed content analysis of individual postings. Are there a central core and peripheral sub-groups present, we asked? The computer program UCINET for Windows supports our calculations. Subsequently, the e-mail correspondences among the sub-groups were content analyzed for processes of intersubjectivity and shared prolepsis using thematic coding with NVivo qualitative data analysis software.

Based on the related work we surveyed, we selected communicational ties among contributors on the mod_perl mailing list for users and developers as relation and the basic units for constructing the network. All of the communicational ties among the selected nodes were studied to provide a census for a full network analysis (Hanneman & Riddle, 2005). The adjacency matrix generated from the SNA software (UCINET) included information about each participant’s communicational ties with all other participants in the population. The rationale for this selection was to collect information about all communicational dyads in the network and to construct an inclusive overview and provide parameters to calculate centrality and density in communication. In this way, sub-groups of core and peripheral contributors are identified for further content analysis.

The mailing lists consist of post and reply messages. Our focus on interaction led us to ignore posts that were not responded to. The presence of interaction requires as minimum one post and reply sequence, and this was coded as ‘1,’ whereas non-interacting dyad’s were coded ‘0’. Consequently, messages that are not responded to are unaccounted for in our analysis. This left us with a standard asymmetric binary network matrix to perform basic operations, which simplified network level analysis. However, one can argue that potentially valuable information is lost by not using a higher level of measurement (e.g., ordinal or interval level). For the descriptive and sorting purposes of providing contrast groups for content analysis, we do not regard the strength of ties as significant. Actually, we believe the simplicity of handling binary data may be worth the information lost.

4. DATA AND RESULTS
We have organized our results into three parts, reflecting the different steps in the research process: 1) The overall network structure, which is an inclusive overview of the communication dyads forming the social network, 2) centrality and density measures, which are computed in order to obtain a general impression of the communication patterns in the mod_perl project, and 3) subgroups within the network, which are clusters of participants in the structural center and peripheral regions of the network. After the network level results are presented, we analyze the processes of intersubjectivity and the role of shared prolepsis by zooming in on the message exchange among participants in the sub-groups, focusing on the content of the messages and identify proleptic instances in the message flow.

4.1 Density and Centrality
The density of the whole network was simply calculated by summarizing the number of communicative connections between members (N=584) in the adjacency matrix and dividing it by the number of possible connections. The density score ranges from 0-1, describing the strength of linkage among the actors in the network. In our network the score is 0.0127, which indicates a rather low density (a value closer to 1 would be high). In other words, the points in the graph represent a rather loosely knit structure. This means that relatively few participants in the mod_perl project communicate directly and frequently with each other. The relative number of communicational ties among actors is only a small fraction of all possible ties. However, network density provides only a rough estimate as it describes the general level of cohesion in the overall network. To explore if this cohesion is organized around particular focal points, centrality measures are more informative. Freeman (1979) developed a technique for computing degree centrality, which shows the overall network activity of individuals. Borgatti (2005) describes degree centrality as “the number of ties incident upon a node” or “the sum of each row in the adjacency matrix”.

The above degree centrality values indicate that dev-12 is the most central member in the social network. As this is a directed graph, it is also relevant to assess OutDegree and InDegree measures. An Out/InDegree of 62/40 means that dev-12 has established 62 communicational ties (his number of postings) and received 40. A high centralization score indicates that this participant has a greater number of connections than others. In an asymmetric expert network such as the mod_perl community, a high InDegree may indicate centrality in the sense that other members consult them more often. High OutDegree centralization could mean that the participant is influential in doing the majority of consulting in the project.

To obtain a visual impression of the connectedness of dev-12, we extracted his ego network from the overall network. This is displayed in Figure 3. Note that node sizes are weighted according to degree centrality.
The sociogram of dev-12 depicts a relatively clear star structure, which indicates that this developer is one of the most active participants in the social network. It also emerges from the same sociogram that the developer is connected to other central developers (e.g., dev-29 and dev-81 in top-left region). However, these measures with corresponding visual representations only signify centrality around the most central points in the network. This does not tell us whether the central points also cluster in sub-groups (cliques) around the structural center. For this we need to compute so-called ‘group conceptualizations’ and combine it with the analysis. The identification of possible clusters or cliques in the central and peripheral regions of the network is probed further in the next section.

4.2 Subgroups within the Network

Considering the population size (N=215) of the network, it might be difficult for each developer to have communicational ties with all the other developers. It was also indicated above that the overall density is at the lower end of the scale (0.0127). A potential consequence of a loosely knit large expert network may be the emergence of smaller sub-structures within the network. In the following paragraphs, we explore the presence of sub-groups or cliques to be used in comparison and contrast with content analyses when we zoom in on processes of intersubjectivity.

Initially, we tested algorithms operating on a strict definition of cliques. This meant that we looked for fully connected sub-graphs in the network where N actors have all possible ties present among them. For the smallest group to be considered a clique, we applied N>=5. The following four cliques were found:

1: dev-12 dev-29 dev-40 dev-81 dev-92
2: dev-12 dev-29 dev-40 dev-81 dev-144
3: dev-12 dev-20 dev-22 dev-23 dev-65
4: dev-3 dev-12 dev-38 dev-39 dev-40

There are thus four maximally complete sub-graphs present in the network. The overlapping relations in the four sets of cliques are displayed in Figure 4.

From the clique sets in Figure 4, we get an impression of central participants’ co-membership in these sub-groups. For instance, dev-12 and dev-40 share membership in three of the four groups. On the other hand, dev-3, dev-20, dev-22, dev-23, dev-38, dev-39, dev-65, dev-92 and dev-144 are more isolated from the rest, since they belong in only one of the cliques. dev-12 and dev-81 share memberships in two of the four groups. We zoom in on the details of their interactions in the next section.

Recalling the last section, we see that dev-12, dev-29, dev-40 and dev-81 also have the highest scores with respect to centrality measures. From the model above, we see that the most central participants are overlapping co-members in the cliques. It follows that these actors form a central sub-group around the structural core in the network. Given that the most central nodes in the network are co-members in clearly defined cliques, we find it plausible to regard them as the structural center in the network. The well-bounded, central position of this sub-group is corroborated by a less strict clique concept called K-plex (Seidman & Foster, 1978). K-plex is an algorithm characterized by each member of a K-plex group of a specific size N has N-K ties to other members. In the data set, a total of four 1-plexes were found (N=5) that confirmed the patterns shown in the previous calculation. In summary, based on social network analysis we have identified a sub-group of 4 central participants operating in the structural center of the network, namely dev-12, dev-29, dev-40 and dev-81. These will be referred to as the nucleus group in the content analysis that follows.

The next problem we faced was to identify a cluster of peripheral participants to serve as a contrast to the nucleus. A distinguishing feature emerging from the overall network structure is the relatively high number of peripheral participants in the population. A simple dichotomized core-periphery model based on density measures indicates that 202 of 215 participants are peripheral members. Moreover, the low-density scores of this group point towards a different communication pattern in the peripheral region of the network than in the core. In the core group we obtain an intragroup (core group with own group) with a score of 0.436 and an intergroup (core group with other groups) score of 0.053, whereas the similar scores for the peripheral regions were much lower: 0.070 and 0.005. A density value of 0.005 indicates a very low prevalence of ties among peripheral members in the network. On the other hand, a density value of 0.436 indicates a high density, or frequent direct communication among participants in the core region. Almost half of the possible ties are present.

However, due to the high number of peripheral participants in the project, some kind of data reduction was needed prior to the content analyses we describe below. Our practical solution to this problem was to randomly extract a selection from the total population of peripheral participants. As a starting point, we used the simple core-periphery model and randomly selected ten
participants from among the 202 who belong to the class of peripheral participants. The UCINET software was applied to
categorize the participants in the two classes according to the
core-periphery model (Borgatti & Everett, 2002). In the upcoming
content analysis, the ten random participants in the peripheral
region are referred to as the peripheral group representing the
participants with low centrality and density measures. In
comparison with the nucleus group, they contribute and interact
infrequently on the project’s mailing lists. The two contrasting
(nucleus and peripheral) groups’ distinguishing features are
displayed in Table 2.

Table 2. Contrasting features of nucleus and periphery in the
mod_perl network

<table>
<thead>
<tr>
<th>Group</th>
<th>Participants</th>
<th>Characteristics</th>
</tr>
</thead>
</table>
| Nucleus | dev-12, dev-29,
| N=4     | dev-40, dev-81 | High centrality and density. Densely knit around the
|         |              | structural center of the graph. Frequent contributors on
|         |              | the project mailing list. |
| Peripheral | dev-19, dev-30,
| N=10    | dev-33, dev-95,
|         | dev-131, dev-142,
|         | dev-153, dev-169,
|         | dev-186, dev-207 | Random participants with low
|         |              | centrality and density measures. Sparsely knit in the peripheral
|         |              | regions of the network. Infrequent contributors on the
|         |              | project mailing list. |

4.3 Processes of Intersubjectivity and the Role of Shared Prolepsis

In the following paragraphs, we present and discuss the content of
selected message exchanges between nucleus group participants
and peripheral participants that involve active sense making
concerning some aspect of their joint software object. The
organization of this section is to first present a sequence of
interaction data from the mod_perl mailing list, followed by an
explanation in “common sense” terminology, and later to discuss
it with the conceptual framework of intersubjectivity introduced in
the beginning of the paper. The extracts show a sequence of post
and reply messages spanning five days. It starts by a problem
raised by a peripheral participant (dev-186) that subsequently
triggers a ‘sense-making’ discussion, involving two members of
the nucleus group (dev-12 and 81). Extract 1 is displayed below:

1-2 Hello @all,
3-5 we are just transferring our Intranet Server which used
Apache 1.3 and mod_perl1 from SLES8 to SLES10
with Apache2 and mod_perl2.
6-7 The switch has to go quick and we can’t redesign all of
our old mp1 programs to mp2, so we try to use the
compat module. But I have no luck with it.
8-10 I tested my config with SELS10, OpenSuse 10.3 and
Ubuntu but i hit everytime the same problem. I searched the
Mail Archivie and the Documentation, but no luck.
11-13 [code]
14 I get the following output. 
15-16 Software error:
17 Can’t locate object method "request" via package
18 "Apache" at /usr/lib/cgi-bin/test1.pl line 4.
19-21 For help, please send mail to the webmaster
(webmaster@localhost), giving this error message and
the time and date of the error.
22-23 It seems, he can’t find the requested methods. How
can i validate that the compat module is loaded and
working?

When switching to an upgraded intranet server, dev-186
experiences problems with his earlier mod_perl1 programs (mp1).
When searching the project’s message archive and documentation,
he found no adequate solution to the problem. In order to assist
mod_perl peers in understanding this problem, a piece of code is
posted along with the message (line 11-13) and the software error
that appears when running the system is displayed in line 15-20.
Following Rommetveit’s (1979) framework for intersubjectivity,
the sender of utterances is in a privileged position to control what
features of the shared object “to enter the field of shared
attention” (p. 95). If we interpret source code as the main joint
software object attended to in the forum, the message sent by dev-
186 points out a specific feature of the object to penetrate the
shared space; namely, the “compat module” (lines 7 and 23) that
is used when transferring server programs written for mod_perl
1.0 to an upgraded mod_perl 2.0 system. In the problem
formulation, he also refers to several different server operating
systems (SLES10, OpenSuse 10.3 and Ubuntu) that are not part of
the mod_perl development project. However, a partial shared
understanding of how these systems integrate with mod_perl may
be important premises for making sense of dev-186’s posting.
What shared information can be tacitly assumed or presupposed
by the receivers of this message? Dev-186 explains that prior to
sending this message, he searched the mail archive and the project
documentation (line 8-10). The previous postings stored in the
mail archive and the project’s documentation can be seen as
knowledge resources that are mutually accessible and monitor the
joint efforts of the participants. However, we also need to observe
how the receivers of the message tune in.

As the initial message is addressed to all subscribers of the
mailing list (Hello@all, line 1), the potential ‘listeners’ or
receivers of utterances are numerous. They are dispersed all over
the developer network, ranging from novices to experts. It follows
that ‘the listener’ to be invited into a partial shared intersubjective
space is not a single individual, as in Rommetveit’s framework,
but rather a vast group of peers. This implies that the sender
somehow has to adapt the message content to a potentially large
number of receivers. A member of the nucleus group (dev-81)
replies to the problem introduced by dev-186 a few hours later on
the same day. It appears that he has experienced “the same kind of
problem” at an earlier point in time (data extract withdrawn for
space reasons).

Dev-81 indicates a shared understanding with dev-186 regarding
the problem of loading the ‘compat’ module, but it is incomplete.
Dev-81 is able to draw a parallel to his own problem, while
pointing out the main differences between them. The latter is
located in a script called "startup.pl," which dev-81 assumes is
part of dev-186 common base (preunderstanding). He goes on to
indicate that the differences between them are in the
configurations of the startup.pl script’s “use” statements. However
there is no further specification of these differences and the
question is left open. It turns out that the lack of details in dev-81’s
reply caught the attention of the nucleus group. The sense-making
discussion that followed is shown in extract 2. It involves dev-12,
dev-19, and dev-81:
[Line] Reply from dev-81, 13 Feb 2008 22:00:56 GMT
1-5 Of course, what I forgot to mention below - and sorry if you know that already - is that whichever perl modules you pre-load in your main Apache server config via the startup.pl script, you do not need to "use" anymore in all your perl scripts or Apache/mod_perl handlers. [...] 
6 Reply from dev-19, 14 Feb 2008 14:13
7-11 This is the first time that NOT using "use" because it was preloaded is mentioned. In fact, how would the modules compile (while testing for example)? Wasn't preloading supposed to do the exact opposite? [...] I'm at a loss here. 
12 Reply from dev-12, 14 Feb 2008 15:51:22 GMT: [...] I think that was just a wording mistake. When the process forks, the loaded modules are shared by the operating system's copy-on-write feature. I believe that's what he was trying to say. 
17 Reply from dev-81, 18 Feb 2008 09:40
18-21 Well, it was not exactly what I was trying to say, but undoubtedly this reflects my incomplete personal understanding of the process anyway, and I stand ready to be educated. 
22-30 What I have until now believed is that perl "code" is in fact "data" for the perl interpreter, and that as such it cannot really be "shared". What I mean is that, as soon as some bit is changed in a "page" of any perl module, that "page" is dirty and must be copied and made private to the one child process. And since there is (in my understanding) not such a clear separation as to which parts in "perl code" are data and which are code, after a while one ends up with a full duplicate in each child anyway. Probably badly explained, but not so in the general sense? 
31-36 P.S. What I really meant originally, is that if the speed to make it work was of the essence, it might be easier to (find/grep) and remove the various use Apache-x() from the multiple modules or cgi scripts, and put them all in the startup script. Then later one could go back and refine things, if it makes a difference. 
37 Reply from dev-12, 18 Feb 2008 19:55:03 GMT 
38-40 Your technical understanding is correct, but in practice most pages remain shared. You can help this by using a tool like Apache::SizeLimit that kills off processes after a while. 

In the above replies, the general and intended functionality of the startup.pl script's "use" statement is the subject of shared attention between two nucleus group members (dev-81, dev-12) and another semi-central participant (dev-19). With reference to previous postings, dev-19 notes that the statement's preloading function is intended applied in the reverse manner (line 7-11). It appears that the incomplete utterances concerning the "use" statement are not sufficiently understood by all of the "experts" and a temporary communication breakdown occurs among nucleus members. A central nucleus group member (dev-12) attempts a repair sequence, suggesting that this is only a wording mistake (line 13), but dev-81 who initiated the reply to dev-186 turns down dev-12s rendering of his explanation. He goes on to admit some uncertainty about his "understanding of the process" before elaborating this point, until he returns to his previous references to "differences." At the end of the excerpt dev-81s explanation is approved by dev-12, who is one of the most central participants in the nucleus group. At this stage possibly the common understanding is brought to a new level, making the previous implicit and partly mistaken view and anticipated information in dev-81s earlier postings explicit. The new understanding is nested in what was partially known in advance, and thus provides both an expansion of the intersubjective space and a clarification of the partly misunderstood feature (Rommetveit, 1974; 1979).

How can we understand the peripheral participant's (dev-186) role in the development of partial intersubjectivity in the above message exchange? And how is the extended discussion that follows related to collaborative construction of knowledge (Stahl, 2006; Sutgers, 2006) as advocated in collaborative learning scenarios? As the initial sender, dev-186 was in a privileged position in controlling what aspect of the shared software object to penetrate the intersubjective space and in the subsequent exchanges he is invited to attain his understanding to those in the nucleus group. However we find no traces in the postings of dev-186 reactions in the follow-up communication, which takes place among participants in the core (we assume he has read the messages).

According to Clark and Brennan (1991), two key factors ‘shape’ grounding processes: purpose and medium. By purpose is meant that a sufficient condition for continuing the collective activity is that communicators mutually believe that the receivers “have understood what the contributor meant to a criterion sufficient for the current purpose” (p. 223). In the case of dev-186, reply messages provided by nucleus group members were perhaps sufficient to carry on with his local problem solving. However, as the message thread continues into a discussion among more competent participants, the initial purpose of the communication thread seems to change. In the second wave of replies shown in extract 2, the sense-making discussion moves beyond the initial problem raised by dev-186. It triggers nucleus group members to discuss the functionality and semantic meaning of a specific feature of the configuration of their software related to the “use” statement. This again leads dev-81 to fine-tune his understanding of his own software installation (referring to Perl scripts). In the final reply message, dev-12 provides positive evidence that the procedure was understood sufficiently for the current purpose (to install the latest version of the software). The information can thus be added to the nucleus group members’ common ground.

In summary: The complete message thread can perhaps be seen as a gradual movement from a local problem experienced by a peripheral actor, towards an expansion of intersubjectivity in a distributed expert network. It thus demonstrates efforts of nesting on the basis of proleptic cues in two different occasions, one triggering the other: first between peripheral and nucleus members and then within the nucleus among the experts. The main finding from this analysis is development or movement along two axes: 1) problem solving to improve the software and 2) knowledge creation to improve the shared understanding. According to Matusov (1996), processes of intersubjectivity transcend specific and time-limited joint activities and are “not only the basis and derivative of the joint activity but also the social glue of different sociocultural activities” (p. 30). The praising of peers, asking to be educated, and reassuring that one’s understanding of the shared object is correct, as indicated by dev-12’s last reply to dev-81, may perhaps be significant for sustaining participation in the open-source development project.
5. GENERAL DISCUSSION

We group the discussion of findings into the dimensions of distributed space and time and evolution of shared knowledge object. Our findings are tentative as we provide examples from one case study. We compare our findings with the results reported in the literature that we introduced in section 2 (conceptual framework).

5.1 The Role of Shared Prolepsis in Distributed Problem Solving

This dimension is for simplicity referred to as “distributed problem solving” as it reflects the time/place matrix commonly used in CSCW to characterize groupware. In the mod_perl study peripheral actors, we assume, create triggers for a more elaborate problem solving process, which in turn may prepare the conditions for building shared understandings. The peripheral participants contributed by posting initial problem formulations based on their experience as users of the software in their local practices (e.g., questions regarding downloading and installing the Apache server). However, proleptic instances that challenged the shared background occurred only during the second wave of replies, among the more competent participants situated in the core regions of the network. Other studies has pointed out that core-participants in open source development groups are often loosely knit and have a history of a few buddies starting the whole thing and keeping the control. We have several instances in our data material that confirms such findings.

5.2 The role of Shared Prolepsis in Collaborative Knowledge Creation

This dimension is for simplicity referred to as “collaborative inquiry”, a process in CSCL that aims to involve students in answering questions that do not have simple answers, but multiple alternatives that need to be clarified by elaboration. The goal is to clarify the situation and create common understanding (Stahl, 2006; Moen, March & Paavola, 2012). The literature on OSD describes paths for newcomers to become active contributors in the activities taking place in the networks. For example Edwards (2001), suggest that situated learning and apprenticeship (techniques from community of practice) do apply in OSD projects. Our own study corroborates previous findings of socialization, but we did find a clear demarcation between insiders and outsiders in terms of participation. Presenting a problem may be one possible path to enter into the core of a project, but a problem-proposer also face the risks of flaming as a result of issuing a poorly formulated question. The development of shared prolepsis suggests that the iiP is in a privileged position in controlling what feature of the joint object should enter the field of shared attention. However, this control may only be temporary and potentially constrained by the iiP’s ability to formulate an adequate description of the local problem that is tuned to the common goal of the project and be worthwhile picked up by a mcP.

For the pre-proleptic stage to evolve into an instance of shared prolepsis, our data suggest that the local problems issued by peripheral participants need to be picked up by more competent central developers and re-defined into assumptions about a piece of shared knowledge. The original local problem formulation may then contribute to the construction of new but partially shared understandings around the structural core of the network. The threaded message extracts discussed in the paper point towards the conclusion that peripheral participants issuing iiPs are not yet carriers of the culturally taken for granted that make prolepsis effective in enrolling outsiders in the project. Our preliminary findings indicate that the former have first of all a pragmatic role in providing the pre-proleptic utterances that serve as the ‘raw material’ or triggers for making new knowledge for their own efforts (see Figure 2). Further studies ought to investigate this relationship in more detail.

Some open issues for investigation based on this research are: How does the move by dev-186 deviates from a participant in an e-mail exchange where a question is posed to “all”? Does communication in mod_perl project in any significant way differ from other forms of asynchronous interaction? And what can we learn from these studies to apply to OSD to ease transitions among the two groups and boost intersubjectivity in interactions.

6. SUMMARY AND CONCLUSIONS

We have studied communication patterns and message content between the central and peripheral regions of the mod_perl network, and shown how a mediated intersubjective space emerged along multiple dimensions. The relative density of interactions among participants from different regions reflects the transition from problems experienced in a local context towards experts’ expanded intersubjectivity. Namely, peripheral participants tend to introduce preliminary problems that serve as building blocks for initiation the (co)construction of a greater intersubjectivity among central participants. As the initial problems become shared with a large number of other participants, an important implication is that the constructive utterances should induce or hint towards some lacking feature in the joint knowledge object that has not yet been introduced into the shared space. The inducing content of such initial problems posed to a large number of participants may allow more competent peers to “fill in” the uncompleted information (i.e. explicate tacit or hidden information) and creatively reformulate these utterances on the basis of ‘proleptic instances’ that not only are more complete in terms of resolution here and now but also foresee further development of the project’s joint software object.

Our findings indicate that there is a shift in communicational functions and structure when going from the peripheral to the core regions of the mod_perl network. The peripheral participants tend to issue initial problem descriptions, whereas the nucleus group is more likely to follow up with advisory and educational replies and contribute to more extended sense making, sometimes spanning several days. Similarly this coincides with a progression towards a more extended space of intersubjectivity. Yet, the phenomenological account of intersubjectivity that takes the ‘here and now’ of a conversation as starting point and suggested in Rommetveit’s (1974; 1979) framework, appears of limited explanatory value in order to explain how these processes unfold across the multiple dimensions of distributed software development.

We proposed a new framework for intersubjectivity that takes into account computer-mediated communication by incorporating a modified (distributed) notion of time and space along the vertical axis (Figure 2). In order to account for the intersections between a local work situation and distributed expert networks, the model also suggests a reformulated continuum along the horizontal axis, from problems experienced in a local work context towards expanded intersubjectivity in mediated expert networks. An important implication of this is that the effects of shared prolepsis serve as a catalyst for collaborative knowledge creation. Therefore, the vast number of peripheral participants in the
network creates an advantage when they ‘feed’ the central core with pre-proleptic utterances (e.g. problems). As pragmatic elements in a ‘collective learning system’, the swarm of actors on the outer edges of the developer network monitors the functionality of the shared knowledge object and contributes with prompts and cues that might trigger further development cycles in the core region. However, for pre-proleptic utterances to progress into the project’s space of common knowledge, the study suggest that the local problems shared by peripheral participants need to be re-defined among the more competent, central developers into assumptions about a piece of shared knowledge that has not yet been introduced into the intersubjective space. This implies that pre-proleptic utterances are often vague and sometimes off-track, suggesting some lacking feature in the knowledge object that may invite re-formulation and further productive discussions among more competent peers.

7. ACKNOWLEDGMENTS
To be completed …

8. REFERENCES

Note: Some of the references have been temporarily withdrawn from the Reference section out to fit the paper into 10 pages!