SHARING AND CREATING KNOWLEDGE IN OPEN-SOURCE COMMUNITIES - THE CASE OF KDE

Andrea Hemetsberger[a]
Christian Reinhardt[b]

[a]Department of Value-Creation Processes / Marketing
University of Innsbruck, Austria
andrea.hemetsberger@uibk.ac.at

[b]LUXMATE Controls GmbH, Austria
christian.reinhardt@luxmate.co.at

Session J-3

Abstract

Our research suggests that knowledge is shared and created in online communities of practice through the establishment of processes and 'technologies' that enable virtual re-experience for the learners at various levels. Three questions guided our research. The first one concentrates on how community members organize content with regard to their daily routines that potentially transforms into knowledge for other members. Secondly, we inquired how new members are enabled to accumulate the knowledge necessary for becoming a valued member. Thirdly, we asked how members co-create and conceptualize new ideas – create new knowledge – in absence of physical proximity. Re-experience is enabled by modular tasks and transactive group memory, rigid guidance of new members, openness and legitimate peripheral participation, asynchronous communication, and virtual experimentation. Empirical evidence is based on an ethnographic investigation of the KDE community – an open-source software project organized online.

Keywords: knowledge creation, open-source, communities of practice.

Sharing and Creating Knowledge in Open-Source Communities
The case of KDE

Andrea Hemetsberger[a]
Christian Reinhardt[b]

[a]Department of Value-Creation Processes / Marketing
University of Innsbruck, Austria
andrea.hemetsberger@uibk.ac.at
[b]LUXMATE Controls GmbH, Austria
Christian.Reinhardt@luxmate.co.at

Abstract

Our research suggests that knowledge is shared and created in online communities of practice through the establishment of processes and 'technologies' that enable virtual re-experience for the learners at various levels. Three questions guided our research. The first one concentrates on how community members organize content with regard to their daily routines that potentially transforms into knowledge for other members. Secondly, we inquired how new members are enabled to accumulate the knowledge necessary for becoming a valued member. Thirdly, we asked how members co-create and conceptualize new ideas – create new knowledge – in absence of physical proximity. Re-experience is enabled by modular tasks and transactive group memory, rigid guidance of new members, openness and legitimate peripheral participation, asynchronous communication, and virtual experimentation. Empirical evidence is based on an ethnographic investigation of the KDE community – an open-source software project organized online.

Keywords: knowledge creation; open-source; communities of practice
Suggested track: C oder E

Acknowledgements

We feel an enormous debt of gratitude to all KDE community members for their openness, helpfulness and their insightful comments regarding our work. May their spirit continue to enlighten their lives and those of others.

Sharing and Creating Knowledge in Open-Source Communities
The case of KDE

1 Introduction

The open-source movement has recently attracted increasing attention, mostly because its mere existence and the way it works contradicts existing theories and counteracts common business practices (Kogut & Metiu, 2001; Kollock & Smith, 1997; Kuwabara, 2000; Lerner & Tirole, 2001; von Hippel, 2002; von Hippel & von Krogh, 2003; Wayner, 2000). Open-source software is software for which the source code is distributed and accessible via the Internet without charge or limitations on modifications and future distribution by third parties (The Open Source Definition by the Open Source Initiative 1997). In open-source software projects expert programmers at different levels, supporters, and users voluntarily contribute to a collaborative software project that is administered via the Internet. They collectively develop software in a decentralized, self-directed, highly interactive, and knowledge-intensive process (Kogut & Metiu, 2001; Raymond, 1999).

The open-source network and its project communities constitute a real life, best-practice example for collective knowledge creation within an online community of practice. Due to its globally distributed developer force and the possibility to collaborate on a large scale basis open-source software projects enjoy extremely rapid code evolution and highest software quality (Cubranic & Booth, 1999). With products such as Linux, Apache, Perl, KDE and Gnome desktop, and many others open-source development is also highly successful in an economic sense. However, it is not the monetary incentive that drives developers' motivation but rather the cultural and the learning aspect (Hemetsberger & Pieters, 2001; Ye & Kishida, 2003).

Open-source projects are almost exclusively administered online. The use of knowledge requires the concentration of the knowledge resources at a certain space and time (Nonaka & Konno, 1998). Internet technology can be used in various ways to pool and archive knowledge resources (Haythornthwaite, Wellman, & Garton, 1998). The design of those online platforms provides the frame in which knowledge is concentrated and activated as a resource for creation. While Internet technologies are highly effective at facilitating the

transfer of codified knowledge, it is considered difficult to share and create tacit knowledge online and collaborate on tasks with high complexity (Nemiro, 2002). “Physical activities and face-to-face interaction are the key to sharing tacit knowledge.” (Nonaka, Reinmoeller, & Senoo, 2000).

Despite the increasing research effort into open-source communities of practice (Lanzara & Morner, 2003; Tuomi, 2001), existing literature nevertheless leaves us uninformed about how knowledge sharing and creation processes develop at the interface of technology and communal structures that effectively exploit the advantages of Internet technology and at the same time are able to overcome the problem of tacit knowledge transformation. The objective of our research is to enhance our understanding of the structures and processes that enable knowledge sharing and creation. To this end we applied a netnographic approach (Kozinets, 2002) to investigate a successful open-source project community – the KDE desktop developers.

2 Theoretical Background

We approach the research objective with a social view of learning and knowledge creation. This view promotes the idea that knowledge is deeply embedded in the technological and social context of a community that creates and reproduces knowledge (Nonaka & Konno, 1998; von Krogh, Ichijo, & Nonaka, 2000). According to social constructivist theory, people construct knowledge as they interact in a social context. Knowledge is information combined with experience, context, interpretation and reflection (Davenport & Prusak, 1998). Hence, collective knowledge creation comprises shared experience, shared context, and communication about interpretations and reflections on information and knowledge. Knowledge is dynamic, relational, and based on human action, thus, it depends on the situation and people involved rather than absolute truth or hard facts (von Krogh et al., 2000). If we agree with the social constructivist view, then knowledge creation is genuinely dependent on an enabling context – technological and social – where individuals form relationships, are acting together, collectively share and reflect on their individual knowledge and beliefs.

In developing an understanding of possibilities and difficulties in transferring and creating knowledge in an online context we have to distinguish between explicit and tacit knowledge. Explicit knowledge is formal and systematic hence, it can be expressed in words and figures. Therefore, given that individuals share some common understanding and thus are able to derive meaning from verbalized knowledge, explicit knowledge can efficiently be transferred on virtual platforms. Information systems can be used in various ways to enable routinized processes. Groupware applications enhance the informal transfer of knowledge within the community, workflow applications can deal with the more formalized steps of the tasks being performed (Ciborra & Andreu, 2001). Furthermore, boundaries of space and time in communication are blurred through tools that allow users being simultaneously present in 'cyberspace' and by storage of messages, allowing asynchronous, but still interactive communication (Haythornthwaite et al., 1998). Yet, especially when tasks require extreme levels of creativity and flexibility, research showed that high levels of virtuality appeared unproductive (Leenders, van Engelen, & Kratzer, 2003, Kratzer 2001, Nemiro 2002). It has been argued that factors such as the social nature of the innovation process and the tacit nature of knowledge limit the possibilities of the Internet (Feldman, 2002) and represent major challenges for online cooperation.

In his seminal work Polanyi (Polanyi, 1966) defined tacit knowledge as nonverbalizable, intuitive, and unarticulated. Tacit knowledge is context specific and personalized in nature. Spender (Spender, 1996) suggested that tacit knowledge be better construed as knowledge that is yet to be *abstracted* from practice. In the same vein, Nonaka and Konno (Nonaka & Konno, 1998) argued that the process of *externalization* of tacit knowledge in explicit concepts, like metaphors, hypotheses, or models is crucial to the formation of shared understanding and cooperation. A slightly different view is presented by Bechky (2003) who argued that knowledge is not simply transferred by means of metaphors, analogies, and so forth, but essentially *transformed* for the learner to be able to comprehend and 'see' things in the light of the sender. Bechky was able to show that decontextualizations or abstractions which are freed from context in order to *transfer* meaning were incomprehensible by others who didn't share the same work context. Transformation, on the other hand means, "placing that knowledge within her [a person's] own locus of practice" (Bechky, 2003), hence enabling her to see that world in a new light. The findings of her study demonstrate that this can be

realized through the use of tangible definitions, referring to examples that physically exhibit the problem.

These insights throw a different light on knowledge creation online. The question then should be how an online community of practice manages to provide tangible definitions of their work by means of technological tools and different forms of communication. In a virtual team space this means that team members, in order to gain shared understanding and co-create knowledge, must find a way of expressing a problem or idea that can be 'touched' in some way, and that makes apparent the differences in individual thinking. But how can individuals transform know-how, which is essentially tacit knowledge, into something 'tangible' online?

In conceptualizing ways how to enable sharing and creating knowledge online, we draw on the communities of practice literature (Brown & Duguid, 1991; Lave & Wenger, 1990; Wenger, 1998, 2000), the concept of double-loop learning by Argyris, and Schön's notion of 'the reflective practitioner'(Schön, 1999). Communities of practice embody knowledge that is often tacit in nature but visible and observable in the common practice of and interactions among competent practitioners. Schön describes such knowledge as *knowing-in-action*. This form of knowledge is also highly contextual and therefore, cannot be externalized and taught independent from its context. Schön's view of knowledge stands in sharp contrast to viewing knowledge as molecular, built up of basic units of information or skills which can be assembled together in complexes of more advanced information. Instead, *reflection-in-action* describes the knowledge or know-how we apply in action and spontaneously (Schön, 1999). According to Schön (1999), learning a professional skill is based on social interaction and competent use of technologies. Many key skills are tightly bound to the tools and material artifacts used by a professional community. *Reflection-on-action* is the intellectual work individuals have to do when they want to share and create know-how and skills with others through social interaction (Schön, 1999).

Lave and Wenger (1990) introduced the concept of legitimate peripheral participation that allows new members to move towards full participation in the socio-cultural practice of a community. This participation leads them to share a common understanding which is essential for collaborative work and knowledge creation. Central to this concept is that learning does not take place by being taught or instructed but by *becoming* a practitioner

(Brown & Duguid, 1991). During a first phase of learning, learners are granted legitimate access to the knowing of experts that can be observed and understood within its context. However, full participation is impossible due to the limited capabilities of the learner. On their way to becoming a full member and expert, learners observe, imitate and practice (Nonaka & Takeuchi, 1995). Moreover, they also learn through interacting and collaborating with others, experts as well as other learners. In order to achieve a convergence of meaning, knowledge has to be acquired by doing and experiencing – being a reflective practitioner. The role of the experts and more advanced learners is to provide access to their experiences and reflections. Helping others to experience, what oneself has experienced before is fundamental for knowledge creation (Maturana & Varela, 1992).

Creation of knowledge depends on whether a group is able to reflect their doing on a meta-level and think about their doing in a double-loop manner (Argyris, 1992). Most people define learning too narrowly as mere problem solving hence, they focus their attention on identifying and correcting errors. This is what Argyris (1992) described as single-loop learning. Highly skilled professionals are frequently very good at single-loop learning. They are quick problem solvers, because they can rely on previously acquired, embodied knowledge and mental models. However, knowledge gets internalized below the level of awareness into the unconscious mind. Although such compressed experience (Weick, 1995) increases the brain's cognitive capabilities, it also hinders cognitive reflection in a double-loop manner. Skilled incompetence is the result of learning focused exclusively on the *solution* of problems and the reflection of experiences as consequences of the action intended to solve the problem. In order to achieve double-loop learning attention must not be directed towards the solution of a problem, but primarily towards its *construction*. Consequently, reflection demands the questioning of the mental models and everyday theories that govern action. Double-loop learning goes beyond a mere interpretation of the consequences of action. It reveals and reflects on the unconscious mental models that initiate action in a specific way. A learning organization must therefore constantly strive to avoid defensive reasoning, causal attribution, and evaluation of thoughts and actions, but rather critically examine and change its own theories-in-use (Argyris, 2000). It is primarily a question of culture and communication processes that help organizations to foster double-loop learning, however, technology and its appropriate use can promote such a learning culture.

This article advances the perspective that knowledge in online communities of practice is shared and co-created through the establishment of processes and 'technologies' that indirectly enable re-experience. Three questions guided our research. Following the metaphor of the "Global Brain Study Group" we asked how such a global brain can work, grow, and think (Heylighen, 2000). Hence, the first question concentrates on how community members organize content with regard to their daily routines that potentially transforms into knowledge for other members. Secondly, as open-source communities depend on attracting and socializing new members, we inquired how new members are enabled to accumulate the knowledge necessary for becoming a valued member. Thirdly, we asked how members co-create and conceptualize new ideas – create new knowledge – in absence of physical proximity.

3 Methodology

At the initial stage of our research we defined four requirements in order to select an appropriate learning community for the investigation of online knowledge sharing and creation processes. We looked for a successful project that has developed organizational structures which fit the theory of communities of practice, and is above average in terms of (1) the period of time existing, (2) the number of members, and (3) the rate of innovation and diffusion. After an extensive exploration among open-source projects and careful consideration we selected the KDE (The K Desktop Environment) project as the most suitable case for our purposes.

3.1 The KDE project

KDE is a desktop environment for UNIX workstations, similar to those found under the MacOS or Microsoft Windows. A few years ago no such desktop environment was available for UNIX – the preferred computing platform for information technology professional and scientists since a long time, because of its superior quality in terms of stability, scalability and openness. However, the lack of an easy to use graphical user interface (GUI) has prevented UNIX from finding its way onto the computers of typical everyday users in offices and homes. With the increasing popularity of UNIX, in particular it's free and open source variant Linux, the need for such a GUI arose. In October 1996 Matthias Ettrich posted a message on an Internet newsgroup asking for help to create a desktop environment for Linux. Upon his call for help a few programmers responded positively.

Today a huge amount of developers constantly work on the creation of a good looking contemporary desktop environment with a consistent look and feel, standardized menus and toolbars, a centralized dialog driven configuration engine and a lot more features required by millions of users worldwide. KDE has achieved an enormous popularity and is now distributed with almost every Linux distribution, like e.g. Debian, Mandrake, Red Hat or SUSE. It is such a great success that a group of industry partners and KDE developers have founded the KDE League – an organization focused on facilitating the promotion, distribution and development of KDE. Among the founding members are leaders from a cross-section of the computer industries, e.g. Compaq, Corel, Fujitsu-Siemens, Hewlett-Packard and IBM.

3.2 The research method

After deciding to choose KDE as case for our research we asked a representative of the project for permission to observe the community and their communication. His answer revealed the cornerstone of their philosophy: “Our community’s communication is open and you can observe and read as much as you want.” During a four months period we thoroughly observed the project community in order to gain a deep understanding of their activity. In addition, the authors made themselves familiar with the KDE desktop environment, and tried to grasp the ‘work philosophy’ of its creators. In an attempt to further comprehend how online communities overcome the problem of physical distance, we analyzed which tools they use depending on the knowledge they share or co-create, and what forms of communication they choose to enable re-experience. We chose a research approach similar to, but not as participatory as Kozinet’s (1998, 2002) netnography, and in accordance with Glaser and Strauss’ (Glaser, 1978; Glaser & Strauss, 1967; Goulding, 2002) grounded theory approach.

During the observation phase memos were written, categories were developed and coded, and regularly discussed within the research team. A procedure of open, axial, and selective coding as described by Goulding (2002) was applied. We also included external open-source affiliates in our discussion, visited two ‘Linux Day’ conferences, one in Vienna, Austria and one in Ede-Wageningen, the Netherlands, and a two days workshop on free software. This was inevitable for us to have a chance to understand their culture, and to be able to grasp the meaning of their technical ‘insider’ language. After a first period of intense observation of their communication and the tools they used, the particular importance of mailing lists for the KDE developers’ knowledge sharing and creation became evident. Therefore we also executed an

in-depth analysis of 510 contributions to the general developers and core developers mailing list posted in March 2003. When theoretical saturation (Goulding, 2002) was achieved we integrated our findings into a coherent theory. To challenge our theory we asked for feedback from community members, a standard procedure of ethnographic research methodology called 'member check' (Belk, Wallendorf, & Sherry, 1989; Kozinets, 2002).

4 Findings

KDE is definitely one of the largest open source projects. More than 800 developers distributed all over the world work and communicate via the Internet. Up to date, their collaboration resulted in about 4 million lines of code. The core developers group consists of 35 programming experts who mainly code, but also constitute the managing group. They are responsible for important management tasks and democratically decide upon strategic and tactical questions (see also: http://www.kde.org/people/gallery.php). Additionally, thousands of volunteers are supporting KDE engaging in a variety of tasks like graphic design, writing documentation, or translation. The translation team alone consists of about 300 individuals who ensure the translation of KDE into more than 50 languages.

The tools they use for communication purposes are those provided by Internet technology. The KDE community maintains a number of homepages which are dedicated to inform the wider public and to provide the platform for different subgroups (e.g.: the games center at: http://games.kde.org). For work purposes common groupware and workflow applications are used that help keeping track of the current work status. Among those applications is a bug database where advanced users can report bugs and add missing features to the wish list. The Concurrent Versions System (CVS) is probably the most important tool for programmers who work simultaneously and must keep track of all the coding. Furthermore, also comments to parts of the code can be integrated into the CVS which should help others to understand the logic applied. A forum and newsgroups are dedicated mainly to advanced users who exchange thoughts and discuss important topics related to KDE in an asynchronous manner. Also Internet relay chat (IRC) is used as a synchronous tool for communication, mainly by advanced users. The most important communication tools for developers are the numerous mailing lists where developers reflect on their work and discuss strategic issues. All workflow and communication is archived and open for everybody to read and follow.

Within the three areas of knowledge sharing and creation depicted in the theoretical section, several processes have been identified that are fundamental for the community's brain to work, grow, and think. In the following we will describe those processes in detail.

4.1 Enabling re-experience by decreasing complexity and transactive group memory

Building up memory and organizing new content is the backbone of a community's knowledge system. In order to be able to digest the huge amount of knowledge technologies and task-related features are implemented that decrease complexity. This is, for instance, the bug reporting system, the modular structure of tasks, keeping track of code in a CVS repository, and shifting the locus of knowledge from individuals to a transactive group memory (Steinmueller, 2000; Thomson & Fine, 1999) where members know *where* to find information.

What the playful KDE community calls the bug reporting "wizard" is a good example to demonstrate how the community fosters single-loop learning with simple and user-friendly tools. As most of the tools used it builds on the specific characteristics of the Internet. With online asynchronous communication the boundaries of time, space and authorship get blurred. As soon as somebody found a problem and enters the findings in a database it becomes accessible to the whole community, and out of the huge developer community an expert gets his hands on fixing the problem (see also: (Mockuss, 2000 #156). The bug reporting tool mostly focuses on problems, not on solutions however, it also provides the opportunity of integrating users' wishes of most wanted features. It is the integrating nature of the bug reporting system that increases the knowledge base of the community and remarkably accelerates the innovation process.

Whilst user integration increases the knowledge base and helps evaluate the benefit of future software solutions, it also broadens the range of tasks and increases complexity. Task complexity is especially harmful when a huge number of developers are involved (Brooks, 1995). Hypermedia provides the potential for creating modular structures which, in the collaborative environment of KDE, are used to distribute tasks among members and thus, considerably reducing complexity without losing track.

Nonetheless, it has to be acknowledged that some applications consist of numerous different modules, which are connected among each other. Such connections may even exist beyond single applications, because specific modules are sometimes used within several other ones, as well. Consequently, it is not always negligible for developers how modules develop which they are not actually working on themselves. Therefore sophisticated coordination among developers is needed (see also: Metiu, 2001 #49). This is realized through efficient and effective communication using the CVS. It is not only a valuable tool that supports simultaneous work, but also helps developers to understand each others work. The notifications on changes attached to every version and the database supported retrieval of changes in the source code direct developers to the work done by someone else. To foster comprehension developers also add comments to their source code (reflection-on-action) which enables a re-experiencing process among the other community members.

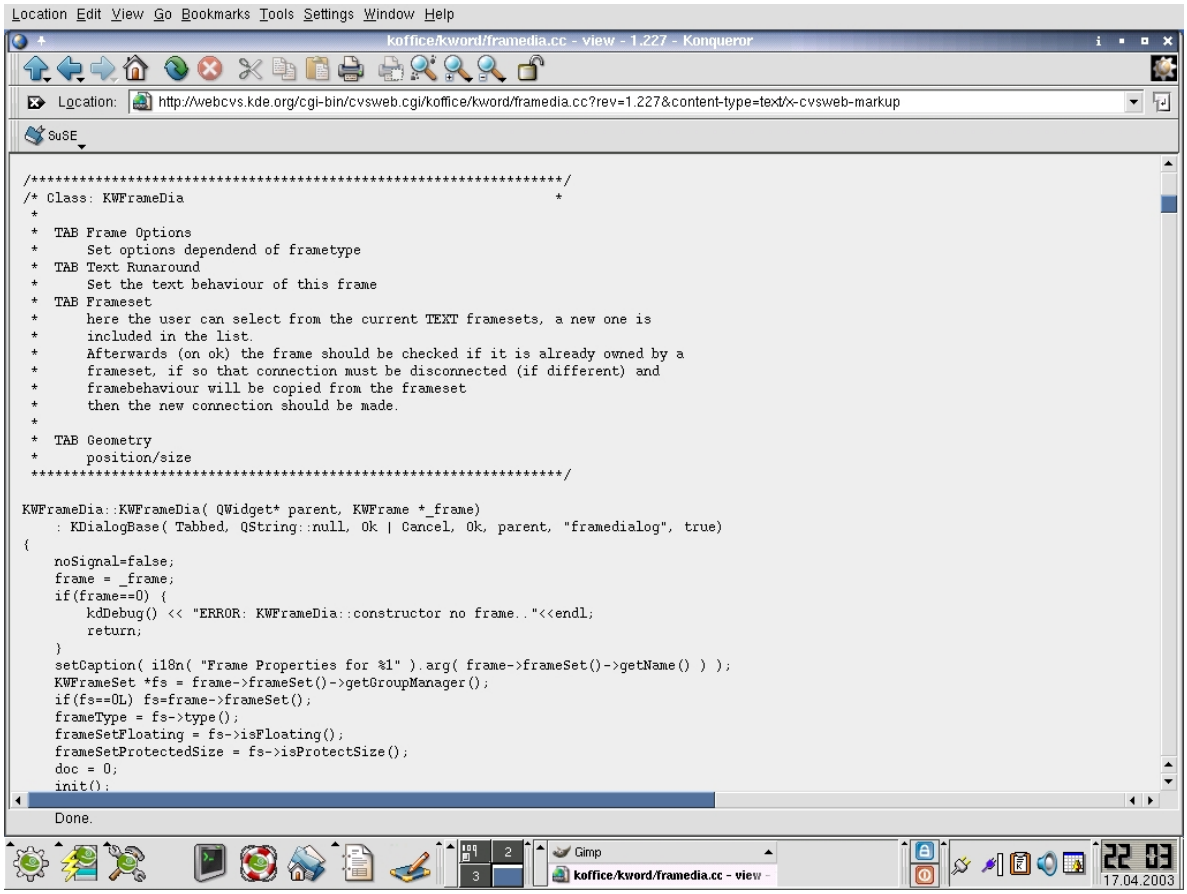

Fig. 1. screenshot of a commented source code

Commenting source code is not new, and the explanations to the code provide only a patchwork of small learning tasks. However, if learning is to take place within individuals expert programmers know that there has to be a challenge, an active and sometimes painful process of re-experiencing what others have elaborated before. Some expert programmers refer to this in their email signature quotes as follows:

> --
> Real programmers don't comment their code. It was hard to write, it should be hard to understand.

Together with the above mentioned tools the most important building block of the community's knowledge system consists of 81 mailing lists which are the platforms where discourse (Habermas, 1981) and open reflection (Senge, 1994) takes place and is archived as part of the transactive memory of the learning community. The following example shows how work done is reflected upon:

> On Sunday 30 March 2003 22:15, xxx1 wrote:
> > On Mon, 31 Mar 2003 01:44, xxx2 wrote:
> > > On Sunday 30 March 2003 11:47, xxx1 wrote:
> > > > The "Source File" is a confusing concept. Source of what?
> > >
> > > True that, but what could be used instead?
> > > "New file" ? - Naw, it could be older!
> > > "File that is going to replace the existing file" ? - Kinda long, isn't
> > > it? "File about to be copied" ? - Same as above, too long.
>
> IMO if there is something hard to describe with words, then visualize it :)
>
> [snip]
>
> > > Might it be helpful to drop the "existing" and "source"- label and make
> > > the "source" semi-transparent or some other effect to show that it's not
> > > existent at the destination folder rather than being about to be copied?
> >
> > Perhaps the concept should be "current document, rather than source file".
> > I did like the arrow too, although obviously the CPU load for the animation
> > would need to be tunable for low performance machines.
>
> When the idea of this arrow came to my mind i was indeed thinking of a
> non-animated one. but this is something for our kde-artists :)
> i believe even a non-animated arrow will make clear what happens to which file
> in which direction.
>
> xxx

All postings and reflections are archived and wait to be looked up by someone who wants to retrieve related information from the group memory. Transactive memory draws on the analogy of the mental operations of an individual and the processes of a group. It has been defined as a shared system for encoding, storing and retrieving information (Thomson & Fine, 1999), as a set of knowledge possessed by group members, coupled with an awareness of who knows what (Faraj & Sproull, 2000). The transactive memory system in the virtual realm turns to a collective mind and memory system that becomes independent from people, hence independent from 'knowing who knows what'. It is only necessary to know *where* to get in contact with the right people. Moreover, similar to human information processing, the KDE transactive memory also provides weekly 'digests' which give an overview of changes made. This is a good example of how information can be re-combined by the use of hypermedia technology.

4.2 Enabling re-experience by guidance, openness and legitimate peripheral participation

New members are integrated through a standardized entry and rigid guidance in the adoption of tasks and cultural norms. 'Newbies' are encouraged to *observe* common practice and communication in an attempt to foster re-thinking and re-experiencing processes before they are allowed to *become a practitioner*. Because of the reflective nature of the commented source code, the interactive tutorials, and the social interaction on asynchronous communication tools learning can even take place without person-to-person interaction. Openness to all tools and communication is the key to knowledge sharing; they constitute a 'hall of mirrors' (Schön, 1999) for the learners to reflect their doing.

At first, learners only peripherally participate in smaller and easier tasks. How they can contribute and what to do is briefly described in a 'How to help KDE' for potential contributors:

> "Are you looking for a way to start helping the KDE project?
> If so, this page might have something for you.
>
> Here you can find descriptions of available small projects involving coding, graphing, writing documentation or any other direct activities which the KDE projects would need but which aren't yet in the charge of a specific developer. Most of the jobs presented here require little time and various degrees of development skill.

> First of all, read here a small note about what KDE has already. This must help you decide yourself if yet another Integrated Development Environment (IDE), yet another Internet Relay Chat (IRC) client or yet another image viewer are necessary. Go to kde-apps.org and Freshmeat.NET and do a search for the application you are thinking of writing or just browse the lists there. If you find in these lists something of interest to you, you might want to contact the author(s) of the code and offer your help directly.
>
> Please note that you are free to choose. You may want to start another Integrated Development Environment (IDE) or Internet Relay Chat (IRC) client of your own, but this way you only make sure you lose one of the most important advantages of the free software / open source concepts: reuse of valuable code.
>
> If you have questions, feel free to email the How-to-Help Team."
> (http://www.kde.org/jobs/)

Within this excerpt of the introduction for 'newbies' one can detect clear rules that reflect a rigid guidance of potential new members on the one hand, and important cultural cornerstones, like the freedom to choose and do what one considers being fun, on the other hand. There are also explicit rights and rules where to read and post (see also: http://www.kde.org/info/faq.php). Rights as well as expectations are formulated in a crystal clear language, like in the following conversation taken from a posting of a newcomer to the developers list:

> Hello Everyone,
> I am totally new to KDevelop, please let me know what it is, when I saw it for the first time I found it as if it is for developing new applications for KDE ...
> Please tell me how to start with KDevelop ... if I want to develop some Applications like what we do with visual basic on Windows platform then what is the best (Let me know whether I can do something with QT Designer for Developing Applications to run on Linux ...)
> Anyone please help me in this regard ... I am very much interested to develop GUI applications for Linux ...
> Thanks & Regards

The poster received the following message:

> Maybe you want to wait for Visual Basic for Linux? Perhaps it is available in about 50 years. Since you are new to Linux I can give you the astonishing advice to read the documents which come with your system or are available at http://www.kde.org/. I do not believe that someone here has the time to write the documentation especially for you.
> I do not know where you come from. Perhaps you are used to Win systems. Obviously new users there get a short introduction to system and all software packages by Bill himself.
> You can believe me that I do not expect this mail to help you, but I could not resist.
> Sorry!

This answer gives the clear advice where newcomers are supposed to look for the first contact with the community – at the community's homepages. The harsh tone of the answer shows the disregard for not putting any efforts in solving one's own problems, before engaging in personal interactivity with community members. This harsh critique has an important function, namely to prohibit communication that violates cultural norms and does not serve knowledge sharing and creation.

Another important directive we found at the homepage is, for instance, the description of the pathway how to become a valuable contributor. Other forms of guidance are provided through FAQ's, HowTo's, and Tutorials. The technological tools used to guide learners are surprisingly simple, the methods of transformation of knowledge as well. Simple descriptions, screenshots, and cases are provided that guide the learner.

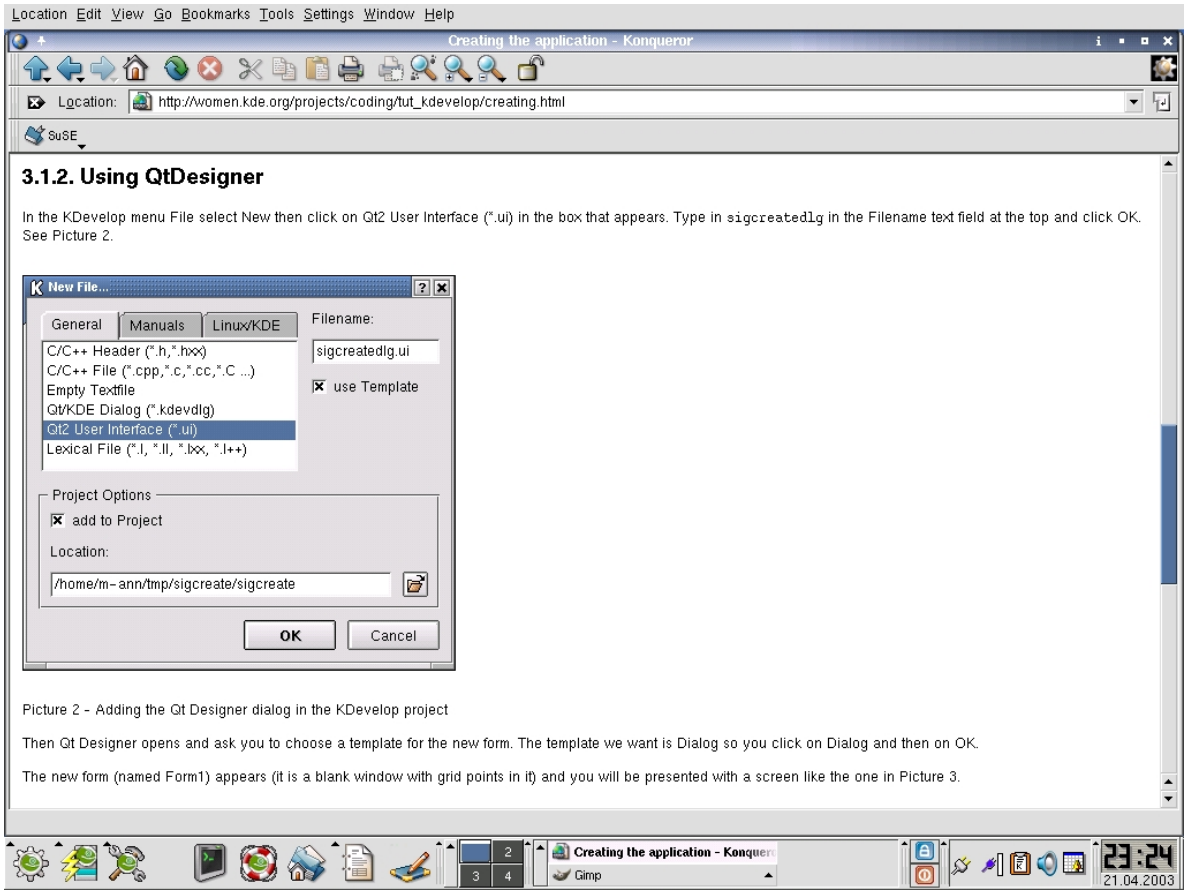

Fig. 2. screenshot – tutorial 'coding with KDE and Qt'

With the help of such tutorials learners can develop their own learning curriculum and set their own pace. Learning unfolds during human-machine interaction and collaboration with experts and other learners until a certain level of expertise is achieved that legitimates participation.

Open access to all past communication and code probably constitutes the most important function for knowledge sharing with new members. Learners can follow the entire streams of communication and processes of code development hence, they are able to re-experience those processes KDE members have undergone before. Within this dialectic of openness and rigid guidance an effective and lively learning culture can flourish.

4.3 Enabling re-experience by asynchronous communication and virtual experimentation

Asynchronous communication tools are also primarily used for new knowledge creation. Conceptualizing problems and new ideas are important creative processes in the KDE community. Using synchronous tools for other than coordination tasks or discussing solutions is actively avoided because this could hinder further reflection in a double-loop manner (Argyris, 1992) and thus, new knowledge creation. For joint conceptualizations KDE developers use programming language (e.g.: plain code; “what if, if then” arguments), analogies, and usage scenarios for collective reflection-on-action. Developers present their ideas, for instance for a new application or feature and ask others to comment on them. Upon such initial messages, lively interactive conversations occur with comments supporting and further elaborating on the idea, presenting a different perspective towards the problem, or pointing towards flaws or even errors in the presentation. Thus the conversation revolves around the construction of the problem itself. Knowledge creation occurs in a double-loop manner. A simple technique to double-loop learning in groups is by using analogies as is the case in the following conversation (analogy underlined):

> The fact that they are too lazy to create new folders and use them
> is similar to the fact that some people have 100 sheet of paper
> laying on their desktop in several levels: would you offer them
> an intuitive search engine to dig the wanted papers out, or
> wouldn't you rather give them a nice IKEA shelf with lots of
> big and small boxes for their papers? Boxes with small labels.

The answer reads as follows, also including an analogy (underlined):

> I kind of agree with you on this point. Some people who have no problems using PCs have very badly organised directories, and they are normally always the people who are badly organised in real life. A friend once told me that you could tell a lot about a person by checking if their virtual wastepaper basket was empty or full all the time.
>
> However, my mother who is really very bad with computers, can not handle the stress of organising files (too complicated), yet she can loves to surf the web and read email. I have tried to teach her the advantages of directories but she finds the whole thing to abstract to grasp -- a computer file is not a real document -- so the metaphor is completely lost on her. Yet surfing the web, something which only exists in cyberspace, makes complete sense to her. Why is that?

In that phase of idea generation it is not possible to draw on a sophisticated language developed for expressing technological details. Knowledge creation here is not a matter of technology but an issue of the greatest benefit derived from the realization of an idea. Hence, the developers place their thoughts into a context others are able to understand. Another common workaround to support the presentation of an idea is to describe the usage scenario and collectively re-think on the problem construction:

> The one [idea] you already gave looks like you have a computer in your living room that runs the whole time or most of the time (neglecting the consumpted power and the noise ;). It replaces the traditional stereoset and your television + video recorder. And some of your living fellows now and then do some work with it. Did I get it right?

Another KDE developer supports this perspective and explains how he handles the situation:

> This is exactly the scenarios we have got in our living room. I live at a student flat with 8 others. We have a computer running all day to browse the internet, play music or video and for some extra storage. Everybody has his own login (for access over ftp), and you can also use this login to start kde. There's also a general login for all users, without a password.

The dynamic process of such interaction is not foreseeable and sometimes ideas evolve which none of the developers has had before s/he engaged in the conversation. Such conversations are the origin of innovation. The group rather creates a ‘virtual’ world, a constructed representation of the future realization of their ideas. This ‘objectification’ of ideas provides the necessary ‘tangibility’ (Bechky, 2003) a team needs in order to be able to co-create a common understanding and imagination of their future action.

5 Discussion

This research demonstrates that online communities of practice successfully overcome the problem of tacit knowledge transformation through cultural norms and rules, technological tools, task-related features, individual and collective reflection, stories and usage scenarios. Highly sophisticated technological and social artifacts are built in order to ensure efficient and effective task-related communication and create a common understanding. We found that the way technical tools are adapted and used on the one hand, and cultural values, norms and rules applied on the other hand to be decisive for enabling re-experience which forms the basis for tacit knowledge transformation.

In particular, we found a culture of freedom, openness, and helpfulness to be one of the cornerstones for extensive knowledge sharing and creation in online communities of practice. This cultural heritage and the clear rules and norms that help following the suggested pathway of machine-interactive and human-interactive online learning are the building blocks for a sustainable community and a successful joint enterprise. The online concept of legitimate peripheral participation builds on re-experiencing methods, like open archives, FAQ's and tutorials that force new members not only to observe before practicing, but also to think and reflect on this experience before asking a community member for help. Although this rule of self-determined learning primarily serves to relieve the experts from extensive 'teaching', it obviously has also major advantages for knowledge sharing.

Enabling re-experience by observing social practice is dependent on a sophisticated archiving system where communication is stored as it really occurred and open access to all ongoing discourses. This allows the observer to follow the whole discourse and reflect upon it. By observing social practice the learner does not acquire tacit knowledge from a single individual, but from the social collective. The individuals engaged in the observed communication are not relevant because of who they are, but because of the roles they take, and the communicative behavior they show. The social tacit knowledge of a community of practice is predominately collective knowledge, not stored within any single individual's brain but within the entire global brain.

Because members of a virtual community of practice work individually at home and are only connected through computer-mediated communication they inevitably have to build on

language for task-related knowledge transfer. A hypermedia environment basically offers synchronous and asynchronous tools for one-to-many and many-to-many communication. Synchronous communication tools, for instance Internet relay chat, are not extensively used for task-specific communication. They mainly function as quick help forum where users ask advanced users for help and as media for social interaction. The advantage of those tools lies in the transgression of geographic and time boundaries which means that you always find someone to ask at any time. However it does not foster knowledge creation in a discursive manner for the reason that individuals give only quick answers and do not reflect on what they read and write. The main advantage of asynchronous, text-based communication is that reflection-on-action is fostered and knowledge is transformed by giving simple cases and examples, using analogies and usage scenarios that are loosened from the context of coding and software and put in a context of everyday experience.

Enabling re-experience is one thing that is enabled by asynchronous communication, but even more important is its function as a discursive platform where knowledge is transformed, reflected upon, and re-combined into something new by the collective mind. Probably the most intriguing communication processes are those when learners and experts collectively construct interesting ideas or problems. In order to overcome the difficulties of virtuality, successful communities of practice turn this problem into an art of acquiring a shared perspective of a problem. By constantly reflecting on the view of other members and one's own view communities gain a shared understanding of the problem but *not* of its solution, because this would hinder further reflection on the side of the individual who volunteers to solve the problem. By means of co-constructing ideas and problems to be solved, online communities of practice constantly support knowledge creation and dissemination not only for the current actors involved, but also for future generations.

References

Argyris, C. (1992). On Organizational Learning. Oxford: Blackwell.

Argyris, C. (2000). Flawed Advice and the Management Trap. New York: Oxford University Press.

Bechky, B. A. (2003): Sharing Meaning Across Occupational Communities: The Transformation of Understanding on a Production Floor. Organization Science, 143, 312-330

Belk, R., Wallendorf, M., & Sherry, J. (1989): The Sacred and the Profane in Consumer Behaviour - Theodicy on the Odyssey. Journal of Consumer Research, 16, 1-38

Brooks, F. P. (1995). The Mythical Man-Month: Essays on Software Engineering. Reading, MA: Addison-Wesley.

Brown, J. S., & Duguid, P. (1991): Organizational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning, and Innovation. Organization Science, 21, 40-57

Ciborra, C. U., & Andreu, R. (2001): Sharing knowledge across boundaries. Journal of Information Technology, 16, 73-81

Cubranic, D., & Booth, K. S. (1999). Coordination in open-source software development, Proceedings of the 7th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises.

Davenport, T. H., & Prusak, L. (1998). Working knowledge: how organizations manage what they know. Boston, MA: Harvard Business School Press.

Faraj, S., & Sproull, L. (2000): Coordinating Expertise in Software Development Teams. Management Science, 4612, 1554-1568

Feldman, M. P. (2002): The Internet Revolution and the Geography of Innovation. International Social Science Journal, 5447 - 56

Glaser, B. G. (1978). Theoretical Sensitivity. Mail Valley, CA: Sociology Press.

Glaser, B. G., & Strauss, A. L. (1967). The Discovery of Grounded Theory. New York, NY: de Gruyter.

Goulding, C. (2002). Grounded Theory - A Practical Guide for Management, Business and Market Researchers. London et al.: SAGE Publications.

Habermas, J. (1981). Theorie des kommunikativen Handelns. Frankfurt a. Main: Suhrkamp.

Haythornthwaite, C., Wellman, B., & Garton, L. (1998). Work and community via computer-mediated communication. In J. Gackenbach (Ed.), Psychology of the Internet (pp. 199-226). San Diego, CA: Academic Press.

Hemetsberger, A., & Pieters, R. (2001). When Consumers Produce on the Internet: An Inquiry into Motivational Sources of Contribution to Joint-Innovation Paper presented at the Fourth International Research Seminar on Marketing Communications and Consumer Behavior, La Londe.

Heylighen, F. (2000). The Social Superorganism and its Global Brain. The Global Brain Group. Available: http://pespmc1.vub.ac.be/SUPORGLI.html26th July 2002].

Kogut, B., & Metiu, A. (2001): Open-Source Software Development and Distributed Innovation. Oxford Review of Economic Policy, 172, 248-264

Kollock, P., & Smith, M. (1997). Communities and Cyberspace. New York: Routledge.

Kozinets, R. V. (1998): On Netnography: Initial Reflections on Consumer Research Investigations of Cyberculture. Advances in Consumer Research, 25, 366-371

Kozinets, R. V. (2002): The Field Behind the Screen: Using Netnography for Marketing Research in Online Communities. Journal of Marketing Research, 392, 61-72

Kuwabara, K. (2000): Linux: A Bazaar at the Edge of Chaos. First Monday, 53

Lanzara, G. F., & Morner, M. (2003). The Knowledge Ecology of Open-Source Software Projects.Unpublished manuscript, Copenhagen.

Lave, J., & Wenger, E. C. (1990). Situated Learning: Legitimate Peripheral Participation. Palo Alto, CA: Institute for Research on Learning.

Leenders, R. T. A. J., van Engelen, J. M. L., & Kratzer, J. (2003): Virtuality, communication, and new product team creativity: a social network perspective. Journal of Engineering and Technology Management, 20, 69-92

Lerner, J., & Tirole, J. (2001). Some Simple Economics of Open Source. L'Institut d'Economie Industrielle. Available: http://www.idei.asso.fr/Commun/Articles/Tirole/simpleeconomics-July24-2001.pdf25th June 2002].

Maturana, H. R., & Varela, F. J. (1992). The Tree of Knowledge: The Biological Roots of Human Understanding (2nd and revised ed.). Boston, MA: New Science Press.

Nemiro, J. E. (2002): The creative process in virtual teams. Creative Research Journal, 141, 69-83

Nonaka, I., & Konno, N. (1998): The Concept of 'Ba': Building a Foundation for Knowledge Creation. California Management Review, 403, 40-54

Nonaka, I., Reinmoeller, P., & Senoo, D. (2000). Integrated IT Systems to Capitalize on Market Knowledge. In T. Nishiguchi (Ed.), Knowledge Creation - A Source of Value (pp. 89-109). London and New York, NY: Macmillian Press.

Nonaka, I., & Takeuchi, H. (1995). The Knowledge-Creating Company. New York: Oxford University Press.

Polanyi, M. (1966). The tacit dimension. New York, NY: Anchor Day Books.

Raymond, E. S. (1999). The Cathedral and the Bazaar. First Monday. Available: http://firstmonday.org/issues/issue3_3/raymond19th June 2002].

Schön, D. (1999). The Reflective Practitioner - How Professionals Think in Action. New York, NY: Basic Books.

Senge, P. M. (1994). The Fifth Discipline. New York, NY et al.: Currency Doubleday.

Spender, J.-C. (1996): Organizational Knowledge, Learning, and Memory: Three Concepts in Search of Theory. Journal of Organizational Change, 9, 63-76

Steinmueller, W. E. (2000): Will New Information and Communication Technologies Improve the 'Codification' of Knowledge? Science and Technology Policy Research, 92, 361-376

Thomson, L., & Fine, G. A. (1999): Socially Shared Cognition, Affect, and Behavior: A Review and Integration. Personality and Social Psychology Review, 34, 278-302

Tuomi, I. (2001). Internet, Innovation, and Open Source: Actors in the Network. First Monday. Available: http://firstmonday.org/issue6_1/tuomi/index.html15th June 2002].

von Hippel, E. (2002). Horizontal innovation networks - by and for users. Available: http://opensource.mit.edu/papers/vonhippel3.pdf24th June 2002].

von Hippel, E., & von Krogh, G. (2003): Open Source Software and the Private-Collective Innovation Model : Issues for Organization Science. Organization Science, forthcoming

von Krogh, G., Ichijo, K., & Nonaka, I. (2000). Enabling Knowledge Creation. New York: Oxford University Press.

Wayner, P. (2000). Free for All: How Linux and the Free Software Movement Undercut the High-Tech Titans. New York: Harper Business.

Weick, K. E. (1995). Sensemaking in organizations. Thousand Oaks, CA: SAGE Publications.

Wenger, E. C. (1998). Communities of Practice: Learning, Meaning, and Identity. Cambridge: Cambridge University Press.

Wenger, E. C. (2000): Communities of Practice and Social Learning Systems. Organization, 72, 225-246

Ye, Y., & Kishida, K. (2003). Toward an Understanding of the Motivation of Open Source Software Developers Paper presented at the International Conference on Software Engineering (ICSE2003), Portland, OR.