Collaborative Development Environments

Javier Soriano
Universidad Politécnica de Madrid (UPM), Spain

Genoveva López
Universidad Politécnica de Madrid (UPM), Spain

Rafael Fernández
Universidad Politécnica de Madrid (UPM), Spain

INTRODUCTION

More and more often organizations tend to behave like dynamically reconfigurable networked structures that carry out their tasks by means of collaboration and teamwork. Effective teamwork is an essential part of any non-trivial engineering process, and collaborative capabilities are an essential support for these teams. Software development is no exception; it is in itself a collaborative team effort, which has its own peculiarities. Both in the context of open source software development projects and in organizations that develop corporate products, more and more developers need to communicate and liaise with colleagues in geographically distant areas about the software product that they are conceiving, designing, building, testing, debugging, deploying, and maintaining. In their work, these development teams face significant collaborative challenges motivated by barriers erected by geographic distances, time factors, number of participants, business units or differences in organizational hierarchy or culture that inhibit and constrain the natural flow of communication and collaboration. To successfully overcome these barriers, these teams need tools by means of which to communicate with each other and coordinate their work. These tools should also take into account the functional, organizational, temporal and spatial characteristics of this collaboration. Software product users are now becoming increasingly involved in this process, for which reason they should also be considered.

In the context of the software development process, then, a collaborative development environment (CDE) can be defined as a safe and centralized solution conceived to optimize collaborative and distributed software development generally based on Internet standards.

This chapter introduces and defines the concept of CDE, while stressing the role these environments play in setting up a virtual space for negotiation, brainstorming, discussion, information and knowledge sharing, cooperation, coordination, development and management in engineering projects generally and especially software development projects. It then analyzes the collaboration-related points of conflict in the software development process. This conflict is motivated by issues, such as the space-time distribution of resources, which have a negative impact on both individual and team effectiveness and efficiency. On the basis of this analysis, we describe what essential purposes a CDE should serve, including: (a) the holistic integration of disparate collaborative processes and tools through a collaborative environment that represents a Web-accessible virtual project space, (b) the expansion of visibility and change control, (c) the centralization and administration of resources, and (d) the reinforcement of collaboration, creativity and innovation. We also examine what features and services a CDE should provide.

Then, we introduce the chief classification frameworks, according to which collaborative tools can be ranked by the needs that they satisfy, each one from a different viewpoint. Knowing and considering these frameworks, a team can contextualize the range of collaborative tools available, and compare them from different viewpoints and on the basis of assembled criteria sets to be able to make a grounded decision on what collaborative tools best meet its needs.

Finally, the chapter will refer to how CDEs are related within open source software communities. These communities have led to a change in how software development is viewed, and both communities and CDEs have been clearly influenced each other. A number of software and open source software develop-

ment support web sites that use CDEs to achieve their goals will be presented.

WHAT IS A CDE AND WHERE DO THEY COME FROM?

The issue of CDEs was perhaps taken up for the first time back in 1984, when Iren Greif and Paul Cashmand organized a workshop that brought together an influential of group of people to examine how to apply technology within a collaborative work environment. This meeting was the source of the "computer-supported cooperative work (CSCW)" concept (Grudin, 1994), which aimed to find an answer to how computer systems can support and coordinate collaborative activities.

A few years later, after further researching the concept of CSCW, Malone and Crowston (1994) introduced *coordination theory*, conceived on the basis of research in several different disciplines like computer science, organization theory, management science, economics, linguistics, and psychology, and according to which they defined coordination as a way of managing dependencies between activities. By characterizing the different types of possible dependencies between task activities, Malone and Crowston were able to identify and, consequently, manage the so-called coordination processes. This investigation identified some of the problems that future CDEs would have to deal with, such as, for example, resources allocation, as well as possible solutions.

Years later, when the technology was far enough evolved and after the Internet had materialized, these coordination processes and all the years of CSCW research led to collaborative tools capable of improving not only the development of software applications, but also the networked exchange of information and ideas from different branches of knowledge, with users who had possibly never worked together before and did not even know each other, based at geographically distant places, even overcoming time differences. This then led to the concept of *groupware* (Baecker, 1993), that is computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment, thanks to the enabling technologies of computer networking, software and services, materializing the ideas emerged from CSCW research (Engelbart, 1992).

Predictably, this activity yielded the first tangible definitions of CDEs. For example, "a CDE is a virtual space wherein all the stakeholders of a project, even if distributed by time or distance, may negotiate, brainstorm, discuss, share knowledge, and generally labor together to carry out some task, most often to create an executable deliverable and its supporting artifacts" (Booch & Brown, 2003). In this definition, the authors establish the key aspects to be taken into account in any CDE. In view of the importance that these environments have gained both in the open source context and the corporate environment with the upsurge of virtual and networked enterprises though, we believe that the definition falls short, as it only states what a CDE is and not how it works. It fails to come up with solutions for the challenges to be met by any CDE concerning the space-time distribution of resources. Therefore, we can add to the definition by saying that a CDE holistically integrates multiple collaborative tools and resources, thanks to which it offers a set of services to aid all the stakeholders in the software development area, including managers, developers, users, commercial software manufacturers and software product support enterprises, to communicate, cooperate and liaise. CDEs consider software development's social nature and assure that the people who design, produce, maintain, commercialize and use software are aware of and communicate about the activities of the others simply, efficiently and effectively, also encouraging creativity and driving innovation.

CHARACTERIZATION OF A CDE

Grady Booch and Alan W. Brown (2003) state that the purpose of a CDE is to create a foundation that minimizes the frictions that have an impact on the routine work of software developers, reducing both individual and group efficiency. The key points of friction are:

- **The cost of working space start-up and ongoing organization**. At the start of a project or when a new member joins, there will be a period of adaptation until the team finds the best tools to use, who to ask, the project status, and so forth.
- **Inefficient work product collaboration**. More than one person sometimes needs to work on the same document at the same time. When this is a critical document, a change control log needs to

be kept, specifying who changed what and why in order to rule out problems with simultaneous modifications, and so forth.

- **Maintaining effective group communication**. Negotiation and ambiguity management are critical tasks not related to programming. Team efficiency suffers to the extent that knowledge is inaccessible or communication mechanisms are defective.
- **Time starvation across multiple tasks**. There never seems to be enough time to do tasks.
- **Stakeholder negotiation**. This is the time it takes to reach consensus among individuals with different viewpoints so that the team can move on.
- **Stuff that doesn't work**. Although often ignored, any item that does not work (network crashes, software package errors, etc.) leads to an interruption and therefore a loss of efficiency

A CDE will help to redirect many of these friction points. Having a visual Web-based environment can help to minimize start-up costs. If this environment also offers a storage system integrating change management and the possibility of saving meta-information, teamwork-derived friction will drop substantially. Communication can be improved using discussion and meeting mechanisms. Time shortages can be counteracted by adding items that act as non-human team members executing scripts or tedious tasks. Negotiation can be improved by automating workflow. If the tool is in widespread use and is also open source, someone else is more likely to have detected and corrected the fault.

In any case, a CDE's worth lies in providing a work environment that minimizes these frictions, allowing the team to focus on its main mission: the production of useful and operational software.

Based on the definition of CDE given here, and also on the friction points previously mentioned, the key purposes a CDE should generally serve are:

- **The holistic integration of disparate collaborative processes and tools through a collaborative environment that represents a Web-accessible virtual project space**. The goal is to broaden the options for communication, cooperation and coordination, fill in missing information, and provide visibility for all resources needed by the team. Additionally, a simple way of capturing data and creating event logs should be provided for the purpose of improving project auditing and follow-up. All these tasks can be carried out by a single system, composed of subsystems providing different services.
- **The expansion of visibility and change control**. Changes will inevitably occur during project development, and the system has to be able to deal with such changes in a reliable and transparent fashion. A key point for distributed cooperation is a clear and exhaustive change control process. A centralized repository with easy access through a user-friendly interface is also essential.
- **The centralization and administration of resources**. The system should integrate and provide the tools needed for collaboration and for project management, providing methods for implementing the relations between teams, and for document, resources and activity sharing. This reduces isolation, maximizes accuracy and speeds up decision making. The system should also offer maximum usability through a generally Internet-accessible user-friendly interface.
- **The reinforcement of collaboration, creativity and innovation**. Process transparency and information availability have a very positive impact by encouraging a constructive attitude towards and motivating collaboration between teams. The ease with which information can be accessed and new ideas can be effectively shared is a source of inspiration for the creative process.

To further specify, if possible, the definition of a CDE, the following are in our opinion services that a CDE should provide.

Table 1. Services a CDE should provide

• Web hosting
• Web interface-based administration
• File persistence with version control
• Visibility control system
• Databases and directory services
• Fault reporting and monitoring system
• Bulletin boards or newsgroups
• Mailing lists
• Task organizers
• New feature request system

CLASSIFICATION FRAMEWORKS FOR CDEs

As we have seen, there are a number of collaborative tools that can be used by a team to collaboratively achieve its goals. However, a number of classification frameworks, each one based on a different set of characterizing parameters and criteria, have been proposed to rank tools by the needs they satisfy and allow a team to make a grounded decision on what collaborative tools best meet its needs. Knowing and considering the available frameworks, a team can contextualize the range of collaborative tools available and compare them from different viewpoints.

Making no claims to being exhaustive, some of the most representative frameworks that have been developed to date are concisely reviewed below.

- Conradi and Westfechtel (1998) provide a thorough taxonomy for comparing collaborative tools in a particular area.
- Grudin (1994) classifies collaborative tools based on their functionality, considering their adequacy for (a) the time mode in which communication takes place (real time, asynchronous), (b) team location (distributed, collocated) and (c) predictability or otherwise of this temporality and/or location.
- Nutt (1996), within the framework of workflow systems, defines a 3D domain space based on the underlying workflow model and more specifically on the mode in which the workflow model represents a work procedure. The resulting framework can classify models that represent just structured or explicit work, models conceived to deal with unstructured work, descriptive and analytical workflow models and conventional workflow models among others.
- Malone and Crowston (1994) identify the processes of coordination used by different disciplines to manage dependencies among activities and analyze their interdisciplinary nature. After identifying the processes, they create a taxonomy of process-based collaborative tools to provide support during software development.
- Van der Hoek et al. (2004) classify collaborative tools on the basis of their high-level approach to collaboration, and particularly depending on whether they take a formal process-based approach, an informal awareness-based approach or they combine both approaches.
- Booch and Brown (2003) classify tools on the basis of the capabilities offered, for which purpose they decompose the characteristics of a CDE into three categories of capabilities based on coordination, collaboration, and the community building nature of a CDE.
- Sarma (2005) classifies the tools depending on their impact on the effort required by users to collaborate effectively instead of focusing on functionality-related aspects and evaluates how sophisticated and automated the support they provide is. The framework classifies the expected user effort that is required to use a particular type of tool and collaborate effectively.

CDEs AND OPEN SOURCE COMMUNITIES

The software development industry has clearly undergone a change of paradigm due to the eruption of the open source phenomenon (Ghosh, 2002). The features distinguishing open source from proprietary software go beyond the merely technical points and stretch to philosophical viewpoints, new economic rules and different market models (Wynants ,& Cornelis, 2005). It also brings with it new development models, whose potential for success is well tried and tested, and which differ from the classical methodologies on several points. The chief feature of this new approach is that development is network focused, enabling people who are geographically far apart to collaborate using the Internet to communicate with each other and coordinate their activities. This networked development approach necessarily targets tools that are used during the process and means that the collaborative tools and environments to support open software development are strongly oriented to Internet use.

Organizations that decide to maintain a site to support collaborative project development and place it at the disposal of the open source software community do not do so for their own benefit or at least this is not their sole objective. The ultimate goal is to promote both development and the use of open source software, and one way to do this is to provide tools and resources to enable communication, cooperation and coordination between developers and users.

Almost all these sites host software projects, although there are others that accommodate no software at all and exclusively target information (Shah, 2005). Others accommodate software that shares some special feature or concerns some specific subject matter, whereas others offer their services for projects from many sources for different purposes. Some are very large and have thousands of visitors every day, whereas others are no more than an initiative run by a handful of enthusiasts. There are sites backed by enterprises and companies that have something to say in the open source world and others that are maintained by user associations or communities that come together around a common interest. Finally, some do not release their resources to the open source community, but use them for their own proprietary developments.

The first distinction then is between organizations that offer services to anyone who wants to use them to create an open source software project (provided they are kept under an open licence) and institutions that impose some additional conditions, generally concerning the project subject matter or even the license type. The first group includes, for example, SourceForge.net and Sofware-Libre.org, which host all sorts of projects provided they are governed by an open license. Most of the projects at Software-Libre.org have a GPL (general public license). SourceForge.net is larger and there is a wider variety of licenses, but most projects have an *open source initiative* approved and certified license, which means that they can be formally termed open source software. These two gateways also host projects on many different subjects, and there are practically no constraints apart from interest or utility.

Other organizations and associations maintain a web site to promote a particular product, stream or subject within the open source community. Alioth's aim is to host projects that are related to the Debian project. It promotes and facilitates the production of software that can ultimately be included in the Linux Debian distribution or serves the project's aims in some way, without placing any constraints on the subject matter of the hosted projects, because Debian is a general-purpose initiative. This improves the product (Debian) thanks to the cooperation of programmers that would probably not have been able to or would not have felt motivated to contribute without these free and accessible resources. The same applies to the Helix Community, the Blender Foundation and the PostNuke Development and Distribution Center. These are all gateways maintained by the creators of a specific project to produce a product. This product benefits from the related projects and the programmers of these related projects benefit because they have resources and tools at their disposal. This is a clear example of symbiosis. Real is the company behind the Helix Community. The Blender gateway is maintained by volunteers.

While the ultimate goal is to promote the development and use of open source software, some organizations pursue other specific goals not directly related to software development. Generally, these organizations aim to act as mediators between open source software-related information management and open source software organizations and interest groups, such as developers, users, commercial software manufacturers and open source software product support companies. This is a third type of community that covers gateways whose goals include providing a meeting and distribution point for documentation related to open source software products and are also a source of news on what is happening within the community. Another possible related goal is to offer developers and companies the possibility of making themselves known to the public, promoting themselves, and contacting sponsors and potential partners. Berlios is an example of this approach.

Another block includes sites, like Shavannah, whose motivations are a bit different. By providing a project host site, they aim to support, promote or improve a more general ideological project rather than a particular open source tool or product. Shavannah is the site hosting the GNU projects. GNU started up in 1984 with the goal of developing a UNIX-type operating system entirely based on open source software. The *Free Software Foundation (FSF)* is the key organization behind the GNU project. The FSF is for the most part financed by donations from sympathizers and aims to preserve, protect and promote the freedom to use, scrutinize, copy, modify and redistribute software and defend the rights of open source software users.

Finally, we should not forget that the collaborative development model associated with open source software is also very appealing to companies that do not consider the possibility of opening their resources to the community of open software users and developers or part of this community, but want to make private use of this collaborative development model and of the associated technologies and tools with a proven potential for success. It is a fact that many companies

use gateways for collaborative software development in their own internal networks to which their employees, business partners and/or customers have access. This way they benefit from the huge potential for resources communication and centralization that these gateways offer. These companies have their own needs that should be considered. Additionally, these companies may in time decide to release some of their proprietary developments. In this case, they often want to make public some parts and/or branches, while others are kept private.

CONCLUSION

Collaboration refers to the different processes wherein people, from small groups to larger collectives and societies, work together, possibly in ubiquitous environments like Internet. On the basis of the study of such processes and their distinctive properties, a number of useful and effective collaborative environments and methods have emerged and evolved to form collaborative development environments (CDE). We have defined a CDE as a virtual space wherein all project stakeholders, even if separated by time or distance, may negotiate, communicate, coordinate, brainstorm, discuss, share knowledge, and liaise to carry out some task, most often to create an executable deliverable and its supporting artefacts, holistically integrating multiple collaborative tools and resources. From this definition, the article has taken a step towards characterizing a CDE and has tackled the key purposes a CDE should serve and what services it should offer. The relationship there is between the rationale behind CDEs and research on CSCW and groupware has also been stressed. Next, a number of prominent classification frameworks have been listed with a view to enabling a team to make a grounded decision on what collaborative tools best meet its needs by contextualizing the range of collaborative tools available and comparing them from different points of view. Finally, we have discussed the role of CDEs in the development of open source communities and have shown how they influence each other.

REFERENCES

Alioth Website. Retrieved March 7, 2007, from http://alioth.debian.org

Baecker, R. (1993). *Readings in groupware and computer-supported cooperative work*. San Mateo: Morgan Kaufmann.

BerliOS Website. *The open source mediator*. Retrieved February 11, 2007, from http://www.berlios.de

Blender Website. *The free open source 3D content creation suite*. Retrieved February 11, 2007, from http://www.blender3d.org

Booch, G., & Brown, A. W. (2003). Collaborative development environments. In M. Zelkowitz (Ed.), *Advances in computers, 59*, San Diego, CA: Academic Press.

Conradi, R., & Westfechtel, B. (1998). Version models for software configuration management. *ACM Computing Surveys, 30*(2), 232–282.

Engelbart, D. C. (1992). *Toward high-performance organizations: A strategic role for groupware*. Bootstrap Institute. Retrieved from http://www.bootstrap.org/augdocs/augment-132811.htm

Ghosh, R. A. (Ed.). (2002). *Free/libre and open source software: Survey and study final report*. Berlin, Germany: International Institute of Infonomics, University of Maastricht, The Netherlands and Berlecon Research GmbH.

Grudin, J. (1994). CSCW: History and focus. *IEEE Computer, 27*(5), 19–27.

Helix Community Website. An open collaborative effort among multimedia enthusiasts. Retrieved from March 1, 2007, from https://helixcommunity.org

Malone, T. & Crowston, K. (1994). The interdisciplinary study of coordination. *ACM Computing Surveys (CSUR), 26*(1), 87–119.

NOC Postnuke Website. *Postnuke Network Operations Center*. Retrieved December 3, 2007, from http://noc.postnuke.org

Nutt, G. (1996). The evolution towards flexible workflow systems. *Distributed Systems Engineering, 3*(4), 276–294.

Sarma, A. (2005). *A survey of collaborative tools in software development* (ISR Technical Report UCI-ISR-05-3). Irvine, CA: University of California, Irvine.

Savannah Website. *A central point for development, distribution, and maintenance of GNU software*. Retrieved February 11, 2007, from http://savannah.gnu.org

Shah, S. K. (2005). Open beyond software. In C. Cooper & M. Stone (Ed.), *Open sources 2.0*. Sebastopol, CA: O'Reilly Media.

Software-Libre.org Website. *The free knowledge forge of the RedIRIS community*. Retrieved December 3, 2007, from http://www.software-libre.org

SourceForge Website. *The world's largest open source software development web site*. Retrieved February 11, 2007, from http://sourceforge.org

Van der Hoek, A., Redmiles, D., Dourish, P., Sarma, A., Silva Filho, R., & de Souza, C. (2004). Continuous coordination: A new paradigm for collaborative software engineering tools. In *Proceedings of Workshop on WoDISEE*, (pp. 29-36). Scotland.

Wynants, M., & Cornelis, J. (Ed.). (2005). *How open is the future? Economic, social & cultural scenarios inspired by free and open source software*. Brussels, Belgium: VUB Brussels University Press.

KEY TERMS

Collaborative Development Environment: A virtual space wherein all the stakeholders of a project, even if separated by time or distance, may negotiate, communicate, coordinate, brainstorm, discuss, share knowledge, and liaise to carry out some task, most often to create an executable deliverable and its supporting artifacts, holistically integrating multiple collaborative tools and resources.

Collaborative Tool: A software module conceived to assure that the people who design, produce, maintain, commercialize and use software are aware of and communicate about the activities of the others simply, efficiently and effectively, also encouraging creativity, driving innovation, and considering software development's social nature.

Collaboration: Refers to the different processes wherein people, from small groups to larger collectives and societies, work together, possibly in ubiquitous environments like Internet. A number of useful and effective collaborative environments and methods have emerged from the study of such processes and their distinctive properties.

Computer-Supported Cooperative Work: A field of study addressing the way collaborative activities and their coordination can be supported by means of software and computer systems commonly referred to as groupware, as well as their psychological, social, and organizational effects.

Coordination: The management of dependencies between activities (generally representing independent subtasks as a result of the division of a cooperative task) and the support of (inter) dependencies among actors involved in carrying them out.

Groupware: Computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment, thanks to the enabling technologies of computer networking, software and services.

Open Source: This concept describes practices in production and development that promote access to the end product's sources and allow for the concurrent use of different agendas and approaches to production. Some consider it a philosophy, and others as a pragmatic methodology. Open source has come to represent much more than software whose source code may be freely modified and redistributed with few restrictions imposed by the terms of its distribution license. Information, documentation, and other "sources" generally related to innovation and knowledge building and sharing processes, tend to fall under the open source umbrella.

Open Source Community: A loosely organized, ad-hoc community of contributors from all over the world who share an interest in meeting a common need, ranging from minor projects to huge developments, which they carry out using a high-performance collaborative development environment, allowing the organizational scheme and processes to emerge over time. The concept represents one of the most successful examples of high-performance collaboration and community-building on the Internet.