GREGORIA ROMERO GRANDE

De: GREGORIA ROMERO GRANDE
Enviado el: jueves, 10 de septiembre de 2009 9:37
Para: Aurora Vizcaino Barcelo
Asunto: RV: un favor
Datos adjuntos: CSCWD09-3485-CWS_Rpalacio_etal[FinalPaper].pdf; CSIE09_942_PID796197_rpalacio_etal.pdf

Aurora,
Necesito también las paginas de portada y contraportada, la del Copywrite y el índice, puedes localizarlas por favor?.

Gracias,

Goyi Romero Grande
UCLM - ESI - Grupo ALARCOS
Pº de la Universidad, 4, 13071 Ciudad Real
Tlf.: 926 295 300 Fax: 926 295 354, ext.: 3747
e-mail: Gregoria.Romero@uclm.es

De: Aurora Vizcaino Barcelo [mailto:Aurora.Vizcaino@uclm.es]
Enviado el: martes, 01 de septiembre de 2009 14:46
Para: GREGORIA ROMERO GRANDE
Asunto: Re: un favor

Goyi, te mando dos artículos con sus referencias para añadir a la Alarnet. Gracias
Aurora

> Ramón R. Palacio, Alberto L. Morán, Víctor M. González, Aurora Vizcaíno, /"Providing Support for Starting Collaboration in Distributed Software Development: A Multi-agent Approach,"/ csie, vol. 7, pp.397-401, 2009 WRI World Congress on Computer Science and Information Engineering, 2009. Press ISBN: 978-0-7695-3507-4. Los Angeles, USA, March 31 - April 2, 2009.
>
>
> Ramon R. Palacio, Alberto L. Moran, Victor M. Gonzalez, Aurora Vizcaino, "/Col//laborative Working Spheres as support for starting collaboration in distributed software development/," cscwd, pp.636-641, 2009 13th International Conference on Computer Supported Cooperative Work in Design, 2009. Press ISBN: 978-1-4244-3534-0. Santiago, Chile April 22-April 24

Collaborative Working Spheres as Support for Starting Collaboration in Distributed Software Development

Ramón R. Palacio[1], Alberto L. Morán[2], Víctor M. González[3], Aurora Vizcaíno[4]
[1] *Facultad de Ingeniería Ensenada, UABC*
[2] *Facultad de Ciencias, UABC*
[3] *Manchester Business School, University of Manchester*
[4] *Alarcos Research Group, University of Castilla-La Mancha*
{rpalacio, alberto_moran}@uabc.mx, vmgonz@manchester.ac.uk, Aurora.Vizcaino@uclm.es

Abstract

One important tendency in software development has been the globalization of its industry. Software developers are frequently required to work in groups which are distributed throughout multiple geographic locations. People and processes (e.g. software design) are consequently distributed. Providing the support that will allow distributed software developers to collaborate at a convenient moment, is therefore extremely complicated both for the person making contact and the person being contacted. We therefore propose a characterization and a model of Collaborative Working Spheres (CWS) through which to understand the management of activities in distributed software development (DSD). Our work first aims to characterize DSD activities mainly through a survey of literature. We then describe a projected scenario to illustrate how DSD activities can be supported by technology and finally we propose a characterization of CWS through which to represent how DSD workers can collaborate at a moment which is suitable for all involved participants.

1. Introduction

Organizations that are dedicated to software development are facing an emerging paradigm shift towards the distribution of processes and development teams. This is known as Distributed Software Development (DSD) [1]. This change is due, among other things, to the desire to exploit broader working day schedules, to benefit from the distribution of resources, to reduce costs and to be demographically closer to the target consumer. Nevertheless, there are also negative aspects such as an increased risk of communication problems [1][2].

This new development paradigm poses several challenges for software design and development: the need for new techniques or extensions to those which already exist to support the inclusion of third-party services, and new processes, mechanisms and tools to deal with the fact that the development teams are geographically distributed. This has led to the situation of some organizations having to modify the way in which they conduct their processes (e.g. requirement analysis, design, management, etc.). These organizations had formerly been accustomed to traditional software development practices, which required all those involved in these processes, such as customers, users, developers, testers, project managers, etc to meet in one place.

In this development scenario, what was formerly a traditional development team which met regularly to discuss work face to face, has now become a virtual team, with each member possibly located in a different geographic location, which makes use of communication technologies in order to interact and carry out a shared task. DSD scenarios are defined by a number of characteristics. One of these is the distance between individual members or teams, which can vary from a few meters (teams working in separate but adjacent buildings), to tens, hundreds or thousands of miles (teams working in different cities). A special case of DSD occurs when the sites involved are in different countries or even continents (global software development or GDS) [1].

People who are not co-located therefore search for ways in which to contact with their colleagues in an informal and rapid manner (e.g. to ask for clarification or help, to obtain others' points of view, etc.) [3]. Typical examples of the technologies that are used in an attempt to solve these problems are the telephone and instant messaging tools. However, although these tools are easy to use, and are usually ubiquitously distributed in work environments, they may have negative results [4]. This is mainly due to the fact that they lack mechanisms that permit a balanced manner in which to decide whether the time for starting an interaction is suitable both for the person making contact, and for the person being contacted. A typical example of this problem is what is called "selective availability", i.e. the ability to establish one's availability according to a criterion, such as "I am available only to people who are related to the task I am dealing with now and am not available to other people". This problem typically presents itself in distributed work environments, where users do not usually have their partners in sight, and therefore do not know what activity is being conducted to determine whether the time is right for both of them to begin an interaction. This work thus aims to provide a model that integrates the individual perspective of DSD workers by means of a potential collaboration awareness [5] approach through which to identify suitable and appropriate moments to initiate collaboration, not only for the person making contact, but also for the person who is contacted in DSD processes.

The remainder of this paper is organized as follows. Section 2 describes the characteristics of DSD activities. Section 3 describes a projected application scenario using the proposed CWS model. Section 4 describes a characterization of Collaborative Working Spheres as a support for initiating collaboration in a timely, appropriate manner. Related work is described in Section 5. Finally, Section 6 presents some conclusions and directions for future work.

2. Features of DSD Activities

Software development is a complex activity and the characterization of its activities is therefore also a complex task due to the coordination problems that arise during its implementation, as these are not only very common, but also inevitable [6]. However, having attempted to ascertain what the characteristics of software development are it would appear that no widely accepted description exists although various proposals focus on the characteristics of the software itself (product), on the features of activities that are conducted as part of the development (process) or on the characteristics of the organization.

[6] proposes a set of elements through which to characterize DSD developers' activities, and these are described as follows:

Scale: this refers to the various values that development can take in terms of:

- Social substratum (individual, among a group of individuals, in a group, between groups, within an organization and between organizations).
- Geographic distribution (co-located, locally distributed and remotely or globally distributed).
- Duration of the development effort (days, weeks, months, years).

It is worth mentioning that the scale of development is related to product size (small, medium, large and very large).

Uncertainty: this refers to the low certainty that the developers may have with regard to knowing the actual progress of a task, goal, or even of the project itself. This is usually caused by coordination problems associated with the project's scale (the greater the scale is the higher the uncertainty tends to be) and the changing nature of the world (e.g. the user's specification needs and software change, the external world for which the software was designed changes, business needs change, etc.). Uncertainty therefore depends both on the nature of the real world, and on the technical possibilities that are available to address the problems that may be caused by this. Uncertainty is sensitive to the participants' perception, as it is very common that what represents a degree of progress in the development of a task for one individual is not necessarily perceived in the same way by another. This leads to the need to interact and share information in order to agree on the degree of progress for the task, thus allowing uncertainty to diminish.

Interdependence: this feature refers to the dependences that exist between the various activities undertaken by developers. These dependences may be due to:

- Shared resources: when a worker's activities depend on a resource that must become available before the task can continue.
- Allocation of tasks: when a worker is dependent on the project leader to specify what his/her activities will be before being able to start them.
- Producer-consumer relationship: when a worker has to complete his/her task and the product is then used to complete the pending task of another worker.
- Prerequisite restrictions: when the employee is given a task that is involved in a sequence of tasks that must be carried out first.
- Transference: the means by which the worker must make his/her product available to another worker who needs it to perform his/her activities.
- Utility: when a consumer worker receives a product, s/he is dependent upon its degree of usefulness to perform his/her tasks.
- Simultaneous restrictions: when two tasks are carried out in parallel by different developers to

achieve a common goal, they depend on each other in order to accomplish them.

- Tasks/subtasks relationship: when a task is divided into subtasks which are assigned to different developers.

Communication: this relates to the way in which information flows among members or participants of the project in order to provide information about or to communicate progress, achievements, problems, solutions to problems, justifications, and so on. Communication can be both informal and formal.

- Informal communication takes place when neither the date nor the location of the interactions is planned, and the information exchange is short but rich and interactive in content (e.g. interactions in corridors). Informal communication is also used in the search for partners related to the task that is being carried out, in taking or leaving messages for colleagues, in establishing review meetings between colleagues to work on contingencies, in releasing documents that support what is delivered as a product, in offering or obtaining help regarding technical issues or methodology, or providing news or progress on a task report.
- Formal communication includes interactions that are planned and have a preset agenda, a worker usually communicates to a group, it is not very interactive, etc. (e.g. a meeting to report progress to the whole team). It serves to support the processes for delivering products to support the completion of certain tasks, to report contingencies (formats), to notify changes in the organizational structure (circulars), and to document products (manuals and / or charts), among other things.

3. Projected scenario

An application scenario is as follows: in a DSD organization a software designer (*collaborator*) located at site S1 accesses a UML file through a diagramming application. This file has been sent to him/her by an analist (*collaborator*) located at site S2, as part of an interface design task.

In the current scenario, whenever the designer has a doubt about the contents of the UML file, he/she usually tries to contact the analyst responsible for it through a means of communication (e.g. telephone or an instant messenging application), but has no information regarding his/her current whereabouts. The designer therefore usually interrupts the analyst's activity.

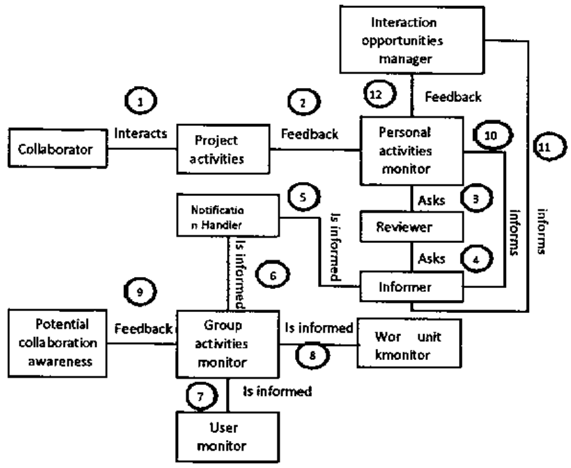

Fig. 1. The CWS model

A representation regarding how to deal with this scenario is shown in Figure 1. Its description follows. At the time when the UML file is accessed the *Personal activity monitor* observes the *Project activities* and records them in its log (e.g. file name, document type, time, date, state). The *Reviewer* then verifies whether that file is related to a project. In this case, it verifies whether the file is associated with a task of project "P1" and updates the log by marking the file as valid to make a request for information.The *Informer* subsequently receives an information request. A *Notification handler* formalizes the request by receiving the order and searching for the updated information concerning the projects and users associated with the task (e.g. the developers involved, the developers' state, the state of the task, etc.) in the *Group activities monitor*. Once the information has been found, it is packaged and sent to the *informer* in order to update the interaction opportunities manager and the Personal activities monitor.

In this case, the designer (*collaborator*) obtains information regarding the file in a rapid and seamless manner through the GUI in which the *Project activities* are presented and acesible. The presented information refers to which collaborators are involved, and on which documents related to a project they are working. In turn, this information also allows the user to interpret the collaborators' current state (e.g. busy, available, not connected, etc.).

4. Collaborative Working Spheres

The features of DSD activities have led us to adopt a practical work unit that allows us first to understand the management of an individual developer's work activities. This unit is the Working Sphere (WS) [7]. A WS is a unit of work that serves to describe work efforts that people pursue in practice in order to meet their responsibilities. A WS can refer to short-term tasks, such as fixing a software component, routine work such as daily maintenance of equipment, events such as a provider's exhibition, or long-term projects such as implementing a new infrastructure for a client. However, the WS concept is limited to a focus on the individual worker. In contrast, the DSD context demands a focus on the work of the group as a whole. Nonetheless, a focus on the individual activities of collaborators is still needed. We therefore propose the introduction of the concept of *Collaborative Working Spheres* (CWS), which extends the concept of WS by considering the work characteristics, identified requirements (see Table 1) and design insights (see Table 2) of DSD activities. A large proportion of design insights are considered by potential collaboration spaces [5] in which collaborators are allowed to obtain a partial and personal view of the information related to the activities which are shared with other collaborators.

Table 1. Identified requirements.

Characteristic of DSD Activities	Identified Requirements
Scale	R1: Interaction between members of work unit R2: Interaction with external members of work unit.
Uncertainty	R3: Knowledge of work unit's degree of progress R4: Knowledge of the work units assigned per person. R5: Knowledge of the overall objective of the program. R6: Knowledge of the goals which affect the work units. R7: Knowledge of the statutes (program charter) governing the program. R8: Control of project specifications.
Interdependence	R9: Awareness of the state of resources. R10: Awareness of the state of members per work unit. R11: Coordination in common or dependent work units among members of the work unit
Communication	R12: Awareness of the state of people who work in the same program. R13: Outsiders' access to program resources. R14: Agree to unforeseen changes related to the work units. R15: Contact the right person at the right time. R16: Acceptable communication media

Table 2. Design insights

Characteristic of DSD Activities	Design implications (and requirements aimed to be covered)
Scale	I1: Services that enable communication between people (R1 and R2).
Uncertainty	I2: Mechanism that allows sharing and filtering of project information among work unit related colleagues (R3, R4, R5, R6, R7 and R8).
Interdependence	I3: Mechanism that enables us to discover the degree of progress of the work being done by each individual member of the project (R9 and R10). I4: Mechanism that allows locating and interacting with members of related or dependent common tasks (R11).
Communication	I5: Mechanism that enables us to discover both the status of the project members and the job they are doing (R12 and R13). I6: Mechanism that permit us to identify when one user can interact with another, based on a need, state and/or performed task (R14 and R15). I7: Synchronous and asynchronous communication service (R16).

A CWS is a combination of working spheres and potential collaboration spaces that allows workers to detect, identify or create opportunities for collaboration (potential collaboration) between each other based on the information managed in their individual work units (WS). It also allows them to identify an appropriate moment at which to initiate collaboration in a more informed way by means of the information obtained from the interaction that the collaborators have with their individual activities. Moreover, CWS will allow collaborators to have a meeting point with their potential collaborators, where they are actually offered a manner in which to begin appropriate interaction and from which they can begin a work meeting with the collaborators involved, along with easy access to the work units involved, and to consistently trigger actual collaboration [5]. The process and characterization of CWS are herewith explained.

4.1 Process

Figure 2 depicts a process which includes three main tasks: (1) Identifying the required information of the activity of those involved (e.g. who is involved in this activity?, which are my activities?, which activities are pending?) , (2) Identifying a suitable moment to interrupt other collaborators (e.g. what is my partner doing?, what is my partner's role at this moment?, what document is s/he modifying), and (3) Initiating collaboration if the moment is right (e.g. who is talking to me?, what is my role?). This necessitates being able to monitor the collaborators' activities, identifying specific information from the common work unit (e.g WS) so that the information on the currently shared activity can be passed to the group and, based on the information obtained, determining whether the moment is appropriate to initiate an attempt at interaction.

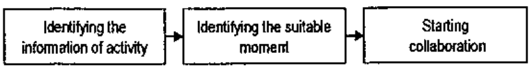

Fig. 2. Process of CWS.

4.2 Characterization

A literature review was used to identity a set of design insights that must be covered by CWS (see Table 2). This emerged from a characterization of the activities of DSD developers, as is shown in section 2. As a result, a set of features to address these design implications is now proposed (see Table 3).

As regards support for remote initial interactions (F1) which addresses design implications I1 and I6, it is necessary to include: i) a mechanism that permits interaction between remote colleagues which is focused on the work units previously assigned. This mechanism should allow us to determine the need for or interest in initiating an interaction, based on whether it is (or when is) the most appropriate time to begin an interaction.

With regard to support for potential collaboration awareness (F2) which addresses design implications I2 and I5, it is necessary to include ii) a set of awareness elements dealing with the potential for collaboration which includes information on the following categories: Who (presence, identity and other collaborators); What (actions, intent, resources and skills); Where (location, vision and scope); Why (motivation); When (event history, dependencies and expectations) and How (action history, resources and activities); and iii) a set of mechanisms through which to share information resulting from the same work unit.

Support for shared personal activities management (F3) which addresses design implications I3 and I4, necessitates the inclusion of: iv) mechanisms to link the work performed individually to the work units proposed by the organization through digital artifacts or resources resulting from such individual work; v) a set of services that offers information concerning work related to the workers involved, work units and resources; vi) mechanisms that implicitly present and adapt information to the context of work to be displayed and monitored by potential collaborators; vii) a set of mechanisms through which to suitably represent the different awareness levels of the potential for collaboration concerning the work units performed by employees at a given moment, and viii) a mechanism that allows the collaborator to transparently create contact groups according to the work unit being performed, with access to the mechanism through which to communicate with remote collaborators.

Finally, with regard to support for remote communication (F4) which addresses design implication I7, it is necessary to include: ix) synchronous and/or asynchronous communication services to enable the exchange of information between collaborators in an appropriate and simple manner; and x) a set of technological tools that permit private communication between the collaborators involved in the work unit currently under way and which are unobtrusively available to potential collaborators.

The following subsection shows a model of collaborative working spheres with a technological approach.

Table 3. Proposed features of CWS

Features (and design implication aimed for)	
F1	Support for remote initial interactions (I1and I6).
F2	Support for potential collaboration awareness (I2 and I5).
F3	Support for shared personal activities management (I3 and I4).
F4	Support for remote communication (I7).

4.3 The CWS Model

One way in which to represent a CWS with a technological approach is through an objects diagram (see Figure 1). This diagram represents the most important entities of the CWS and some of their relations. The principal aim of this approach is to support the requirements for a system. As Figure 1 shows, the *collaborator* is the worker (e.g. software designer or architect) that interacts (1) directly with the *project activities* through a graphical user interface (GUI), which is observed by a software technology (in the background) called the *personal activities monitor*, so that the GUI provides feedback (2) to the monitor with information concerning the files accessed from the applications. The individual activity monitor, in turn, requests an evaluation of each of the open files at a given time (3) from a software component called the *reviewer*, which relates the identified file with the work units assigned to the collaborator. If this is related, a request for information is sent to the other component called the *informer* (4). Otherwise, the application is dismissed. The *informer* requests information regarding work (5) from the *notification handler*, and this component will be aware of valid projects. To do this, a search process on the organization's project repositories is requested to help decide whether the petition should proceed or not. The latter is reported (6) through the *group activities monitor* which keeps track of the context through (7) a *user monitor* and through (8) a *work unit monitor*. An awareness model of the potential for collaboration provides feedback (9) to the *group activities monitor*. The *reviewer* reports information related to the work unit upon which the collaborator has focused its attention to the *personal activities monitor* (10) and to the interaction opportunity manager (11). Finally the *personal activities monitor* obtains information from (12) the *interaction opportunities manager*.

5. Related Work

Several research works that may contribute to the main tasks of properly iniating collaboration in DSD have been identified in literature. These tasks include: i) Identifying the right time to initiate collaboration. Potential Collaboration [5] is complementary to Actual

Collaboration. This refers to the possibility of collaboration and, as such, occurs while people are working on an individual basis, not necessarily in relation to a collaborative effort, mostly outside a shared space representation and usually in an asynchronous communication mode. Doc2U [8], offers an extended instant messaging service which provides support for potential collaboration through the presence of users, documents and specialized services. These works provide elements that may be used to create user services, task and resource presence. ii) Identifying information concerning DSD workers' activities. Working Spheres (WS) proposes a way in which to manage personal activities in the presence of interruptions [7]. Despite only containing information about individual activities, WS from multiple users can actually be grouped for group work. iii) Entering into collaboration. Project-View [3] suggests modifications to the UI of a traditional instant messaging application, considering three specific characteristics: awareness, user information and reminders. Taskmaster [9] proposes a redesign of e-mail, offering an easier option through which to manage the tasks that a user must perform. Both works provide a specilized UI with elements concerning the assigned work units that extend the way in which collaborators are able to interact with each other.

These studies were developed for purposes other than starting collaboration in DSD. Some of them do not even remotely consider work groups [8], others are only user interface proposals and/or design guidelines [3][4][7], and yet others were proposed as modifications to work environments outside the DSD domain [3][9].

6. Conclusions and Future Work

There is currently a great tendency towards developing software in a distributed manner, in order to benefit from the advantages that this kind of development brings. However, distributed development also implies that people do not have the opportunity for face to face interaction, which in turn introduces the possibility of coordination problems, disruptions at unwanted times, and the lack of informal communication. This may have an impact on the trust and quality of team member communication, and on project results. The CWS model aims to be aware of the managing of individual activities (WS) in DSD by providing data from both the user context and from the project context. Future work involves the validation of our model through the use of a prototype for distributed software developers.

7. Acknowledgements

This work is partially supported by FABRUM project, Ministerio de Ciencia e Innovación (grant PPT-430000-2008-063), the MELISA (PAC08-0142-3315) projects, Junta de Comunidades de Castilla-La Mancha, Consejería de Educación y Ciencia, in Spain., and by UABC under grant 207 of XIII Convocatoria Interna de Proyectos de Investigación. The first author is supported by scholarship PROMEP/103.5/06/3244.

8. References

[1] Layman L., Williams L, Damian D. y Bures H. "Essential communication practices for Extreme Programming in a global software development team", Information and software technology , 48(9), 781-794, 2006.

[2] Damian D. and Zowghi D., "The impact of stakeholders geographical distribution on managing requeriments in a multi-site organization", Paper presented at the IEEE Joint International Conference on Requeriments Engineering (RE'02), Los Alamitos, CA, USA, 2002.

[3] Fussell, S. R., Kiesler, S., Setlock, L. D., and Scupelli, P. "Effects of instant messaging on the management of multiple project trajectories". In Proc CHI '04. ACM, 191-198. 2004.

[4] Czerwinski M., Horvitz E., Wilhite S. "A diary study of task switching and interruptions", Proc CHI 175-182, ACM Press. 2004.

[5] Morán A., Favela J., Romero R., Natsu H., Perez C., Robles O., Martinez A, "Potential and actual collaboration support for distributed Pair-Programming", Computación y sistemas Vol. 11 No. 3 pp 211-229. ISSN 1405-5546, 2008.

[6] Palacio R., Morán A., González V., "Soporte para la entrada en colaboración en el desarrollo distribuido de software: implicaciones de diseño", Mexican International Conference On Computer Science (ENC-SES'08); Mexicali, BC, México. 2008.

[7] Gonzalez V. and Mark G. "Constant, Constant, Multi-tasking Craziness": Managing Multiple Working Spheres. Proceedings of CHI 2004, ACM Press. 2004.

[8] Morán A. "Potential and Actual Collaboration Support in the design of distributed Groupware", Institut National Polytechnique de Grenoble, PhD Thesis. 2005.

[9] Bellotti, V., Ducheneaut, N., Howard, M., and Smith, I. "Taskmaster: recasting email as task management." Position paper for the CSCW'02 workshop on Redesigning Email for the 21st Century. 2002.