Published in IET Software
Received on 21st December 2009
Revised on 23rd April 2010
doi: 10.1049/iet-sen.2009.0097

ISSN 1751-8806

Tool to facilitate appropriate interaction in global software development

R.R. Palacio[1,2] *A. Vizcaíno*[3] *A.L. Morán*[4] *V.M. González*[5]

[1]*Facultad de Ingeniería Ensenada, UABC, Ensenada, México*
[2]*DES Navojoa, ITSON, Navojoa, México*
[3]*ALARCOS Research Group, UCLM, Ciudad Real, Spain*
[4]*Facultad de Ciencias, UABC, Ensenada, México*
[5]*División de Ingeniería, ITAM, México, D.F.*
E-mail: rpalacio5@itson.mx

Abstract: Distributed software development is a new working philosophy that the software industry is currently facing. Organisations may benefit from the situations that this shift has created, although they must also confront new challenges related to them. In this study, the authors focused on the lack of timely adequate opportunities for informal interaction, which has been identified as an important issue to overcome coordination, communication and trust limitations. The authors attempted to confront this problem through obtaining information from the personal activities of remote colleagues. In this respect, the authors propose introducing and defining collaborative working spheres (CWS) because the authors argue that CWS permit the identification of opportunities for interaction at appropriate moments. This concept is illustrated with the design of CWS-instant messaging (IM), an extended IM tool that supports the CWS concept. This tool was tested by 16 distributed software development (DSD) workers during an initial scenario-based evaluation. The results show favourable evidence towards both the perceived usefulness and ease of use of CWS-IM.

1 Introduction

Organisations dedicated to software development are increasingly confronted with a working philosophy shift towards the distribution of processes and development teams. This is known as distributed software development (DSD) [1]. This change is owing to, among other things, the desire to extend working days, to benefit from the distribution of resources, to reduce costs and to be demographically closer to the target consumer. Paradoxically, the shift also introduces negative aspects such as an increased risk of experiencing communication problems [2, 3].

In general, DSD scenarios are defined by a number of characteristics. One of these is the distance between individual members or teams, which may vary from a few metres up to tens, hundreds or thousands of kilometres. A particular case of DSD is that in which the distances are between cities in different countries, or even continents, and this is known as global software development [4].

Literature reports that several have been the reasons that have led companies to adopt DSD:

1. Software companies require highly skilled human resources and seek to meet this need by employing programmers in different cities and countries [5, 6].
2. In order to be closer to the market and have a shorter response time, many companies have established development groups closer to the location of their clients [6, 7].
3. Virtual development groups are developed quickly to exploit new opportunities in this market [8].
4. By working in different time zones, development groups can work continuously on critical projects [9, 10].
5. Companies reduce their costs by recruiting human resources from places where labour is cheaper [11, 12].

In spite all these advantages, DSD is not exempt of problems. Developers distributed in different geographical locations have difficulties to interact and coordinate their activities. This is due to the nearly total absence of interaction among distributed workers, which is caused by distance [13, 14].

A fundamental aspect of work in software development environments is that it is characterised by a high communication and coordination level among participants [13]. This is owing to the following:

- The need to achieve decision consensus among members of a group: It refers to a decision process that not only seeks the agreement of most participants, but also aims to resolve or mitigate the objections of the minority to achieve more satisfactory decisions; for example, requirements approval between customers and analysts, change acceptance by developers and change control engineers.
- The decisions must be made expeditiously and with precision: that consists to keep as informed as possible to

the working group regarding the progress of the project. This fact is with the goal of that group members have elements to decide what and when to perform their assigned activities; for example, the notification and delivery of new versions of a design document.

- Interaction among team members must be regular and frequent as there is constant demand of communication between group members during the phases of software development project; for example, information request and delivery of tasks progress reports.
- The fact that the work is cooperative and collaborative: It refers to the developers that work side by side to jointly produce an artefact; for example, programming in peer group review and the creation of a design specification document.

However, the manner in which these features are dealt with is entirely dependent on the way the development process is being performed. In the case of co-located development, project members are on sight or are easily accessible, so that it is possible to see or know what they are doing, without any significant effort. It is even possible to judge whether that precise moment is appropriate to interrupt what others are doing in order to establish communication and maintain a coordination effort. In contrast, in the case of DSD, participants are located at distant sites, signifying that the contextual information existing in a co-located situation is no longer present, which hinders communication, coordination and control. It is, therefore, essential to know the state of the activities of the person that one wishes or needs to contact, and thus attempt to find an appropriate moment for both participants at which an interaction can be initiated while minimising the negative effects of an interruption. That is, information is necessary to discover the context of development of a participant to be contacted in order to potentiate collaboration [15, 16]. With this, we assume that a good coordination requires a good communication [14], and a good communication should be initiated at an appropriate time. This means, having the best conditions to respond to the request for interaction ('willing to interact') [17, 18].

Given the need to diminish the impact of disruption to the work being done by the person being sought, the first questions to guide this work emerge: What is needed to initiate interaction in DSD?, how can an appropriate time at which to do this be identified?, what are the appropriate mechanisms that can be used to obtain, deliver and present the information to identify the appropriate time to initiate the interaction? Compared with the mechanisms currently used by developers, an additional question is: What is the impact in terms of perceived usefulness and ease of use of these new mechanisms?

In order to answer the above questions, it is important to know how and why interruptions occur. In several empirical studies researchers have investigated how workers are interrupted while carrying out their activities (e.g. [19]). These studies have shown that the complexity of the task, its duration, the number of interruptions and the type of task have an impact on the difficulty of returning to the interrupted task (task switching). Thus, the results of those studies characterise how workers behave while confronted with interruptions (interruption management). Other works study the problems that an interruption may cause, such as remembering what it is necessary to do in a new task which was indicated in the interruption (e.g. prospective memory failure) [20, 21]. This effect increases with the number of interruptions and also when the number of tasks rises [21]. On the other hand, returning to an interrupted task is also rather difficult for employees [19], as they have to remember what the last thing they were doing in relation to this task was [22]. Another issue stated in [23] is that when the workers are performing a task they often tend to leave answering an interaction until later because they prefer to finish the task at hand rather than attending to the interaction. Additionally, nowadays there is an increase in interruptions caused by new technologies (e.g. email, instant messenger, mobile devices) [21]. Researchers are therefore attempting to study when these interactions are or are not irrelevant (e.g. [23]).

However, after reviewing the literature related to interruptions we did not find anything which would provide sufficient elements with regard to how to select the most appropriate time to initiate the remote interaction.

In order to deal with this, we propose the use of instant messaging (IM) and collaborative working spheres (CWS) as a support for informal communication, and through which the management of individual work in collaborative projects can thus be complemented by providing multiple collaborators with awareness information of multiple tasks through the IM interface. Fussell *et al.* [24] also use IM with a similar goal. However, this proposal does not consider the use of awareness information to promote the initiation of interaction, and focuses mainly on providing information concerning the user's identity, project team membership and general availability (offline, online, available, busy etc.).

Another relevant proposal is that of [25] in which a more personal approach to project management can be defined from the ideas presented as personal activity management (PAM), which offers a documented and informed perspective of the work that individual workers have to do. PAM is therefore based on the analysis of the processes and strategies that are involved in the way in which workers confront the planning and management of their activities. PAM also uses the concept of working sphere (WS), which explains how people as individuals organise their work, and is sufficiently flexible to represent the activities with the required degree of granularity [26, 27]. We consider that using the concept of WS would be useful in our work as it provides elements of activities and/or tasks performed by people (e.g. resources, repositories or related applications used by the individual) that may be used to identify potential opportunities at which to initiate interaction through the monitoring of context [e.g. through potential collaboration awareness (PCA) [15]]. The concept of PCA is important because it enables us to discover the most appropriate time at which to establish interaction.

Thus, in this work we propose the integration of a PAM perspective of DSD workers with a PCA approach to provide support in the initiation or commencement of interaction at appropriate and timely moments, not only for the person establishing contact, but also for the person that is contacted in the DSD processes.

The remainder of the paper is organised as follows: Section 2 outlines related work. After that, in Section 3 the WS and CWS concepts are described. Section 4 explains the features of DSD activities, whereas Section 5 describes the design requirements to develop a tool that considers those features of DSD that assist in detecting when it is most convenient to interact with another person. Section 6 explains how this tool was developed using software agents, and also shows a scenario to illustrate how the tool works. Section 7 describes the evaluation performed in order to discover whether our tool was more useful or easy to use than the traditional instant messenger. Finally, Section 8 presents our concluding remarks and the directions of our future work.

IET Softw., 2011, Vol. 5, Iss. 2, pp. 157–171
doi: 10.1049/iet-sen.2009.0097

2 Related work

Currently, there are technologies that support synchronous and asynchronous remote communication, which are widely used by distributed workgroups. However, the main barrier to using such tools is the limited understanding of what is being done. That is, remote users do not have relevant information to complete tasks or activities successfully. For instance, during the execution of project activities it is required to have relevant information at hand or to have readily access to a resource that has it.

For the latter case, communication tools represent an important means to acquire the information by providing access to a third-party that is the owner of the information (e.g. email, IM, the telephone etc.). However, actual access to the information requires to the one looking for the information to interrupt her own work to obtain the information, and to interrupt the work of the potential provider of the information. In addition, the communication tool may represent an additional source of interruptions by providing access to information that is irrelevant to the task at hand. These interruptions may cause unnecessary costs to the project [28], such as the additional production costs resulting from poor coordination of activities or tasks, maintenance costs of communication links, delays in products and those caused by re-work.

For these reasons, during software development activities, communication requires to be cost-effective regarding interruptions; that is, achieving a balance between having constant awareness and communication among colleagues for the benefit of the issuer and minimising interruptions and work fragmentation for the benefit of the receiver. Some aspects of this phenomenon have been studied by previous studies and technologies. In order to analyse their contributions we can divide them into two main areas: tools for awareness and tools for starting interaction.

Regarding the work on tools for awareness in DSD, there are several works focused on reporting the activity taking place within a development group. For example Palantir [29] is a tool that complements existing configuration systems by providing distributed awareness of the project's progress. This is done through a graphic display which shows measurements of severity and impact of the changes in the artefacts. ProjectWatcher [30] is a system designed to provide awareness support regarding two questions ('who is who in general?' and 'who works in this area of the code?'). MRAC [31] is a tool based on video, which permanently links a physical space with a virtual one. FASTDash [32] uses a representation of shared source code where the activities of other team members are highlighted; so that a developer could determine where others are working. Impromptu [33] allows users to share task information through displays environments. This enables group interaction with information focused on the problem to solve, which is shared through displays for discussion and reflection. YooHoo [34] is a system to keep developers informed about changes in the source code. This is done by custom notifications per user.

After the literature review we saw that with regard to the work on awareness, none of the tools provide a way to automatically reflect the change in status of a user through the project activities, neither status filtering according to the activity of both the issuer and the receiver. Furthermore, none of the cited works focus on informing the group with level of detail regarding the activity that individuals perform from their own workspace.

Regarding the work on starting interaction, there are different communication tools that have been proposed to work within software development environments. For example, Taskmaster [35] is a proposal based on the redesign of the email interface. This provides an easy option to control and display the tasks that users should attend. Rear view mirror (RVM) [36] is a presence-based instant messenger with features that help to support work teams. RVM offers the creation of groups. RVM is focused on managing working groups, but these groups have yet to be created manually. Hubbub [37] is an instant messenger that offers users the consciousness and access to potential conversations. This system allows users to associate a 'sound ID' that the contacts listen when the user is active. ProjectView-IM [24] suggests changes to the interface design of an IM application to provide both individual and project-related awareness information. This design facilitates the division of attention between multiple projects without penalising a primary task.

Concerning the work on starting interaction, none of the tools permit to automatically locate and create contact groups based on the assigned activities in the project management system. In addition, these tools for starting interaction were not designed to try to establish an interaction that is aligned to both the interests of the issuer and the recipient of the interaction. That is, the current communication mechanisms favour to the issuer of an interaction, since they do not consider the context of the receiver.

3 From WSs to CWS

A WS [38] is a concept that serves to describe the units of work that people use to organise and define their work in order to meet their responsibilities. A WS can refer to short-term tasks, such as fixing a software component, routine work such as the daily maintenance of equipment, events such as a vendors' exhibition, or long-term projects such as implementing a new infrastructure for a client. In spite of its usefulness in analysing the information work practices, and work fragmentation, 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 necessary. We therefore propose the introduction of the concept of CWS, which extends the concept of WS by considering the work characteristics, identified requirements and design insights of DSD activities.

A CWS is a conceptual combination of WSs and potential collaboration spaces that allow workers to detect, identify or create opportunities for interaction, communication and 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 interaction in a more informed way by means of the information obtained from the interaction that the collaborators have with their individual activities. Moreover, CWS provides collaborators with a meeting point at which they and their potential collaborators can initiate optimal interaction and can begin a work meeting with the collaborators involved. This meeting point also provides easy access to the work units involved,

IET Softw., 2011, Vol. 5, Iss. 2, pp. 157–171
doi: 10.1049/iet-sen.2009.0097

and consistently triggers actual collaboration [15]. The conceptual model and characterisation of CWS are explained as follows.

Fig. 1 depicts a conceptual model of a CWS. It shows two subjects working on a particular related activity within a project, which is represented by a set of elements that trigger the interaction with an activity. A WS is represented by a circle that contains the events, persons, activities, objectives, actions and resources that define the way in which people carry out a particular activity.

Fig. 1 also includes the potential for collaboration [15] (represented by an arrow). It allows users to identify opportunities for initiating interaction through the monitoring of the working context. To do this, potential collaboration is integrated in three steps:

- The first step is to identify the information required in the activity in which the protagonist is involved (e.g. who is involved in this activity?, what are my activities? and which activities are pending?).
- The second step is to identify a suitable moment to interrupt other collaborators (e.g. what is my partner doing?, what is my partner's role at this moment? and what document is s/he modifying).
- The third step is to initiate interaction if the moment is appropriate (e.g. who is talking to me? and what is my role?). This requires being able to monitor specific information from the collaborator's WS so that the information concerning the current activity can be passed to the group and, based on the information obtained, determining whether the moment is appropriate to initiate an interaction attempt.

4 Features of DSD activities

Software development is a complex task in itself, and the characterisation of its activities is therefore also a complex task owing to the coordination problems that arise during its implementation. Unfortunately, these coordination problems are not only very common, but also inevitable [13].

Having attempted to ascertain what the characteristics of DSD are, it would appear that no widely accepted description exists. 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 organisation. Inspired on the proposals of [13] and [39], we performed a literature review to establish a set of elements through which to characterise DSD development activities. These are described as follows:

1. Scale: this refers to the various values that software development can take in terms of:
- social substratum (individual, among a group of individuals, in a group, between groups, within an organisation and between organisations);
- geographic distribution of the participants (co-located, locally distributed and remotely or globally distributed); and
- duration of the development effort (days, weeks, months, years).

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

2. Uncertainty: this refers to the low certainty that developers may have with regard to knowing the actual progress of a task, goal or of the project itself. This is usually caused by communication and coordination problems associated with the project's scale (the greater the scale, 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.

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

- 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.

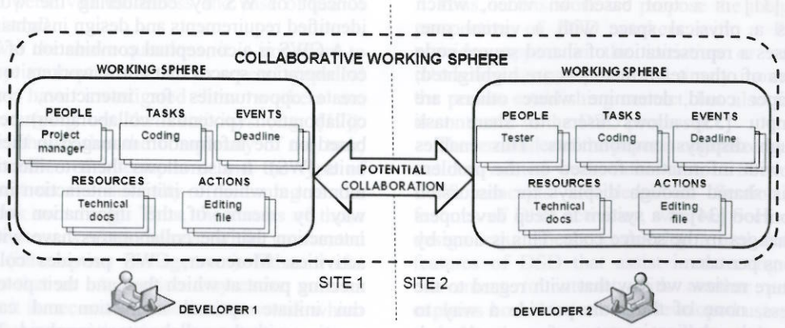

Fig. 1 *Conceptual model of CWS*

IET Softw., 2011, Vol. 5, Iss. 2, pp. 157–171
doi: 10.1049/iet-sen.2009.0097

- Prerequisite restrictions: when the worker 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 own activities.
- Utility: when a consumer 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.

4. 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 time nor the location of the interactions is planned, and the information exchange is short but rich and interactive in content (e.g. interactions in hallways). Informal communication is also used in the search for partners related to the task 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 with regard to 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, those situations in which a worker communicates to a group or the communication is not very interactive etc. (e.g. a meeting to report progress to the whole team). It serves to support the processes of delivering products to support the completion of certain tasks, to report contingencies (formats), to notify changes in the organisational structure (circulars), and to document products (manuals and/or charts), among others.

Considering the previous characterisation, and based on the results of the literature review of software development processes [40], we identified an ensemble of interaction needs that DSD workers experience while performing their activities. These needs are presented in Table 1. The first column refers to the features of the activities as previously defined. The second column shows the interaction needs that were identified among DSD workers.

The identified needs include (i) Scale: this has been previously discussed under the perspective of informal interaction and artificial proximity, owing to the need to bring workers closer to each other through informal interactions. (ii) Uncertainty: this perspective is provided by project management, owing to the need to know information related to the different projects under execution. (iii) Interdependence: this perspective is based on proposals from informal awareness, WSs and informal interaction owing to the need for awareness of what the workers involved are doing at a given moment to allow them to coordinate themselves through brief interactions. (iv) Communication: this perspective is based on interaction, informal awareness and PCA owing to the workers' need to exchange information with all those involved.

The identified needs for interaction result from the characterisation of DSD workers and their activities. It is important to note that in later sections of this work communication will focus mainly on informal communication. Formal communication can be suitably supported by both formal methods and commercial software applications. In contrast, and despite the fact that informal interactions are rich in content, they are generally not regarded as an important asset within organisation [13].

5 Design requirements for developing a tool to facilitate communication in DSD

As our goal is to design a tool to facilitate communication in DSD settings, in this section we describe how the different requirements of this tool were obtained by taking into account issues related to DSD needs, and the features of CWS.

5.1 Design implications regarding DSD

Based on the features identified for DSD activities (Table 1) we shall now identify some of the design implications that are necessary to provide appropriate support to initiate interaction between DSD workers. Table 2 presents areas of

Table 1 Features and identified needs of DSD workers and their activities

Features of DSD activities	Identified needs
scale	interaction among members of the work unit
	interaction with collaborators external to the work unit
uncertainty	knowledge regarding progress status per work unit
	control the project specifications
	knowledge regarding work units assigned to other people
	knowledge regarding general objective of the programme
	knowledge regarding goals on which the work units make an impact
	knowledge regarding programme charter that drives the programme
interdependence	awareness of the state of resources
	awareness of the status of members per work unit
	coordination in common or dependent work units among members of the work unit
communication	awareness of the status of people collaborating in the programme
	access to resources internal to the programme by people outside of it
	change control over unforeseen events of work units
	start an interaction with the right person at the right moment
	adequate and acceptable means of communication

IET Softw., 2011, Vol. 5, Iss. 2, pp. 157–171
doi: 10.1049/iet-sen.2009.0097

opportunity and matches them to certain specific design implications. These are briefly described below:

1. Regarding scale, technology must provide communication services that will permit interaction between people who are distributed in different locations, in order to guarantee the participation of all those involved in a certain project.
2. Concerning uncertainty, technology must provide a mechanism for sharing project information among colleagues or any other related people, as this will allow the progress status of activities assigned to people and that of the project itself to be determined.
3. Regarding interdependence, first, a mechanism is required to ascertain the degree of progress of the work being carried out by each member of the project, which will signify increased information about what is being done in this context. Second, a mechanism is required that will allow members who share common or related tasks to be located and to interact, so that they are aware of and anticipate situations that affect the DSD activities.
4. Concerning informal communication, first, a mechanism is required to indicate the status of the members of the project and the work they are performing in order to have a better understanding of the convenience of attempting an interaction (i.e. interrupt the worker). Second, a mechanism is required that identifies when a user may interact with another, based on the interaction need, status (busy, away etc.) and the work being performed both by the worker who wishes to make contact and the one being contacted; and third, synchronous and asynchronous communication mechanisms are necessary to provide support for both real-time interaction, and for situations in which messages are not delivered immediately, but are stored for later viewing.

5.2 Design requirements to support CWS

Once the requirements with regard to DSD upon which we wished to concentrate had been studied, we decided to study those requirements that were also necessary to support CWS. Table 3 presents the resulting set of requirements and also indicates those design implications (see Table 2) with which each new requirement coincides. A brief description of each feature is described as follows:

Table 3 Features of CWS

	Features (design implication)
F1	support for remote initial interactions (I1 and I6)
F2	support for PCA (I2 and I5)
F3	support for shared PAM (I3 and I4)
F4	support for remote communication (I7)

- Remote initial interactions (F1): it is necessary to include a mechanism that allows interaction between remote colleagues who are 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. This feature is aligned with the design implications I1 and I6 described in Table 2.
- PCA (F2): it is necessary to include a set of awareness elements dealing with the potential for collaboration, including information on the following categories: who (presence, identity and other collaborators); what (actions, intent, resources and skills); where (location, gaze and scope); why (motivation); when (event history, dependencies and expectations) and how (action history, resources and activities). Furthermore it would be advisable to include a set of mechanisms through which to share information resulting from the same work unit. This feature is aligned with design implications I2 and I5.
- Shared personal activities management (F3): this requires the inclusion of mechanisms to link the work performed individually to the work units proposed by the organisation, and this should take place through the use of digital artefacts or resources resulting from such individual work. Moreover, a set of services that offers information concerning work related to the workers involved in it, their work units and resources should also be included; a further requirement is to support mechanisms that implicitly present and adapt information to the context of work to be displayed and monitored by potential collaborators. Furthermore, mechanisms with which to correctly represent the different awareness levels of the potential for interaction based on the work units being worked on by people at a given moment should be added. Finally, 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 is also necessary. This requirement is aligned with design implications I3 and I4.
- Remote communication (F4): it is necessary to include synchronous and/or asynchronous communication services to enable the exchange of information between collaborators in an appropriate and simple manner, along with a set of technological tools that will permit private communication between the collaborators involved in the work unit currently under way and which are unobtrusively available to potential collaborators. This feature is aligned with design implication I7.

Table 2 Design implications for a tool to initiate appropriate and unobtrusive interaction in DSD settings

Features OF DSD activities	Opportunities	Design implications
scale	collaboration among colleagues collaboration with experts	I1. services that allow communication among people
uncertainty	knowledge regarding progress status of colleagues in their assigned work units knowledge regarding progress status of the project	I2. mechanisms to share and filter project information among colleagues from related work units
interdependence	awareness of the status of work units awareness of the state of resources coordination of common or dependent work units	I3. mechanisms that allow to know the progress level of the tasks that each member of the project is executing I4. mechanisms that allow members of common or dependent work units to be located and to interact
communication	awareness of the status of people collaborating in the program access to resources internal to the program by people outside of it start an interaction with the right person at the right moment adequate and acceptable means of communication	I5. mechanisms that allow the status of project members and the tasks they are performing to be known I6. mechanisms to identify when one user may interact with another, based on the needs profile, status and activity under execution I7. services for synchronous and asynchronous communication

It is important to mention that this work is an extension of a previous work [41] in which we proposed the characterisation of CWS. This characterisation represents how DSD workers can collaborate at a moment which is suitable for all involved participants. Therefore this paper aims to test whether the CWS model is feasible to design and introduce in DSD environment.

6 Designing a tool to initiate interaction in DSD

Based on the aforementioned ideas we have designed a tool to help users to determine when the moment for initiating an interaction attempt is appropriate. This tool has been implemented by using a multi-agent architecture, since agents offer autonomy and proactive features, which are useful for obtaining and disseminating information about the various tasks in a project without the users' intervention. To develop multi-agent system, we used the INGENIAS methodology [42], through which we guided our process of analysis and design and modelled each agent and the interactions among them. An example of agent model is illustrated in Fig. 2. In this case, the monitor agent carries out two tasks: (i) monitoring the files that the user is working on and (ii) report that a working file is in use. The figure also shows the role of this agent (activity monitor) and its specific goal (identify working files in use).

In Fig. 3 we depict an example of interaction model. In this figure we can see the goals of each agent and the interaction between them.

The agents were structured into two agencies: the user agency and the project agency (see Fig. 4). The user agency is responsible for supporting requests made for information by the interested user. It is composed of the monitor, requester and identifier agents. It is worth mentioning that there is an assistant agent in this agency, which is called the interface agent.

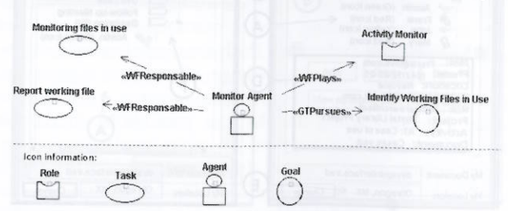

Fig. 2 *Monitor agent model*

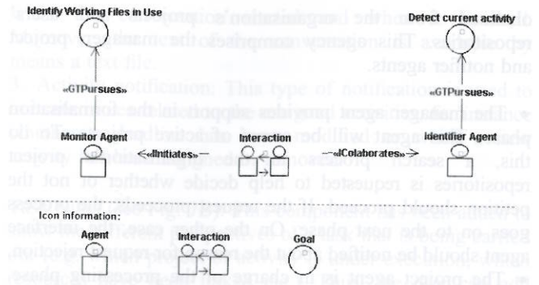

Fig. 3 *Interaction model between the agents monitor and identifier*

The roles of the different agents in the user agency are explained as follows:

- The monitor agent is in charge of supporting the monitoring phase. It is therefore responsible for knowing the information (e.g. name, type, date etc.) about the artefacts (e.g. files) manipulated by the user during his/her work. This requires the implementation of a proactive monitoring process to capture the information generated during the interaction between the user and the computer applications (e.g. text editors, programming languages, design applications etc.).
- The identifier agent provides support in the identification phase. This agent is responsible for identifying whether the file that is being manipulated by the user is connected with any activity and/or task of the organisation's projects. This is done through the implementation of a proactive search process to identify and link the information generated during the monitoring phase.
- The requester agent is devoted to requesting information about the projects from the project agency.
- The interface agent collects the information that is sent from the instance of a project with the goal of updating the information of the graphical user interface (GUI).

On the other hand, the project agency aims to provide the information of the context of work based on information

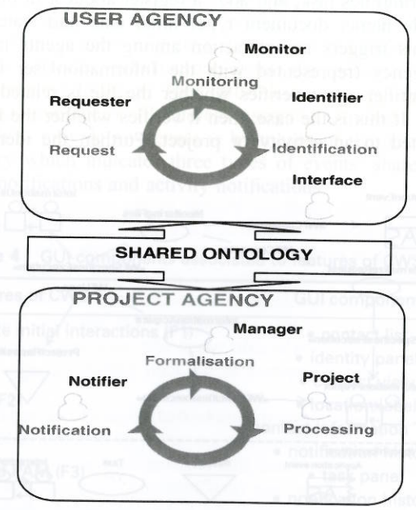

Fig. 4 *Distribution of agents: the multi-agent architecture*

obtained from the organisation's projects and users' repositories. This agency comprises the manager, project and notifier agents.

- The manager agent provides support in the formalisation phase. This agent will be aware of active projects. To do this, a search process in the organisation's project repositories is requested to help decide whether or not the petition should proceed. If the request proceeds, the process goes on to the next phase. On the other case, the interface agent should be notified about the reason for request rejection.
- The project agent is in charge of the processing phase. Therefore it is the agent responsible for the information on a specific project. It will request a search process for an organisation's particular project and the result will be sent to the notifier agent.
- The notifier agent supports the notification phase. It is responsible for reporting the requested results to the interface agent (mediator between the user and agents).

In addition to agencies, Fig. 4 shows a shared component ontology in order to have a consistent communication between the agents of the different agencies.

6.1 Scenario of work

In order to illustrate how the tool works, we present the following scenario:

In a DSD organisation a system designer accesses an UML file through a diagramming application. This file has been sent to her as part of an interface design task. Whenever the designer has a doubt about the contents of the UML file, she usually attempts to contact the analyst responsible by a particular means of communication (e.g. telephone or an IM application). The designer therefore usually interrupts the analyst's activity.

To explain how the multi-agent system (depicted in Fig. 4) works, Fig. 5 depicts a proposed scenario. A brief explanation follows.

At the time when the UML file (e.g. 'designInterface.vsd') is accessed (represented with the ApplicationEvent icon), a monitor agent observes the event by means of the MonitoringFiles task, and adds a register about it in the log (e.g. file name, document type, time, date and state). In turn, this triggers an interaction among the agents in the user agency (represented with the InformationUser icon): An identifier agent verifies whether the file is related to a project. If this is the case, then it verifies whether the file is associated to an activity or project. Further, the identifier agent updates the log by marking the file as valid to make a request for information to the project agency. This way, the user agency (see Fig. 4) detects that the user is working on a valid activity of a valid project (the 'designing GUI' activity of the 'accounting-project'). Furthermore, the user agency makes an information request to the project agency (represented by the InformationAcquire task icon). This request triggers the interaction among agents of this agency as follows: the request for information from the user agency is received by the manager agent. Then, the manager agent checks whether the 'accounting-project' is still active (e.g. there are people working on this project) and thus tests that the request is correct. Next, the located project agent searches for information related to this project (such as developers involved, status of the developers activity etc.). After finding this information, it is sent to the interface agent. In this way, the requested information about the project (e.g. 'accounting-project') is delivered and ready to be displayed to the end user.

As a result, Fig. 6 shows the information that the user can see with regard to the people who are involved in the same project on which s/he is working at that particular moment. Fig. 6F depicts the activities of this project that have been assigned to this person. This information is useful for the distributed employees since it enables them to discover what other developers are doing at that particular moment. Additionally, they can use this information to decide whether or not to interrupt the other developers, since they can see the activities on which those people are working and whether or not these activities are related to their work. So, when a person is working on a task in the same project and in the same activity as another, the tool automatically shows the icon that represents the user's status as green (for illustrative purposes the icon is indicated with the legend 'green icon' in Fig. 6). When the person is working on the same project but in a different activity then the icon appears in yellow (indicated with the legend 'yellow icon' in Fig. 6). Finally, when they are working on different projects this icon will be red (indicated with the legend 'red icon' in Fig. 6). At the same time, the interface provides information regarding the roles that his colleagues are currently playing by means of the same icon (e.g. check mark = tester, magnifying glass = analyst, keyboard = programmer, question mark = no specific role and eyeglasses = project manager).

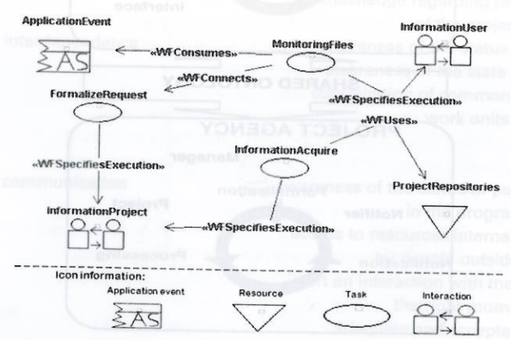

Fig. 5 *Scenario diagram*

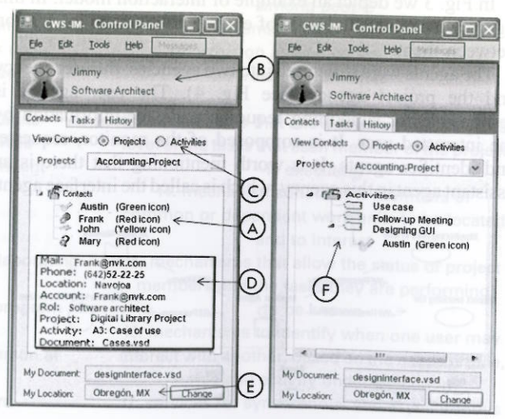

Fig. 6 *CWS IM interface: contact view*

IET Softw., 2011, Vol. 5, Iss. 2, pp. 157–171
doi: 10.1049/iet-sen.2009.0097

6.2 CWS-IM: current implementation of a CWS

This section illustrates how the information provided by the multi-agent system can be viewed by the end user, and describes how the previously detected requirements have been fulfilled when developing our tool. We named this tool CWS-IM.

Regarding GUI components, Figs. 6 and 7 show the actual look of the interfaces proposed for CWS-IM. A brief description of them is presented below:

Contact list: the GUI of the tool presents a list of potential collaborators and of their assigned activities of a particular project (see Figs. 6A and 6F). By using the contact list it is possible to start a chat session with a group of collaborators or with one collaborator. This component allows users to obtain the presence, role and teamwork information of the rest of the workers at a glance.

Identity panel: it is used to show the actual collaborator's information such as name and role (see Fig. 6B). This component allows users to obtain information about their own identity at a glance.

Contact view: Through this option users may choose which view they prefer (see Fig. 6C). It allows users to obtain a list of contacts per project (see Fig. 6A) or a list of the assigned activities of the user in the project (see Fig. 6F).

Contact information ToolTip: It is accessed through a right click over the desired contact (see Fig. 6D). This component shows current information about potential collaborators, such as authorship identity and current activity. The information of the potential contact is accessed only if they have common activities or projects.

Location label: This component provides information about the person's current geographical location (see Fig. 6E). This information is supplied based on the assigned computer's IP address. It allows providing information complementary to identity.

Notification history (see Fig. 7A): This component allows users to obtain the history of events regarding the projects of interest and notifications about assigned activities at a glance. It indicates three types of events:

1. Shared files: when a user shares a file with another user, the notification history logs a notification about this share.
2. Chat conversations: during a group conversation, the notification history indicates which people were involved, when the conversation started and when it finished. Moreover, the text of the conversation is available by means a text file.
3. Activity notification: This type of notification is used to indicate issues related to the assigned activities, for instance whether a deadline has been modified or when a new activity has been assigned or removed.

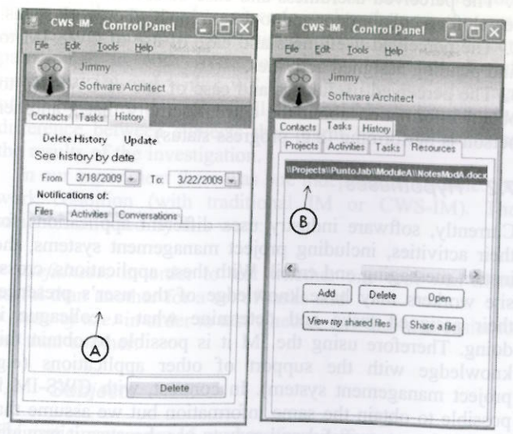

Fig. 7 *CWS IM interface: notification history and task panel*

Task panel (see Fig. 7B): This component has been added to show the different perspectives of a task that is being carried out (e.g. which project or activity is under execution, which resources have been linked etc.). Thus, the user can add documents that could be useful for a particular activity. This does not mean that these documents are reported as artefacts of the project. Rather, with this, the tool distinguishes which documents are associated with the assigned activities. That is, this component allows the association of the personal documents of the project with the assigned activities. By associating, the tool can determine that a particular task or activity is related to a specific project. Furthermore, this component allows sharing personal documents with a group of colleagues regardless of their availability, since these are stored in a shared space, which can be accessed through the notification history component.

Regarding the fulfillment of the requirements identified, Table 4 presents the features of CWS along with the associated GUI component of the prototype interfaces.

In order to fulfil the requirement related to remote initial interactions (F1), the GUI of the tool presents a list of potential collaborators and of the activities assigned to a particular project (contact list). Users can therefore choose what view they prefer by changing the option (contact view). By using the contact list it is possible to start a chat session with a group of collaborators or with one collaborator. On the other hand, the identity panel is used to show the actual collaborator's information such as name and role. All this information allows users to obtain the presence and role information about the rest of the workers at a glance. With regard to PCA (F2), the tool has a contact information ToolTip, which is accessed through a right click over the desired contact. This component shows current information about potential collaborators (e.g. location, active document according to the activity being carried out, current role etc.). There is also a location label which provides information about the person's current geographical location. This information is supplied based on the computer's IP address assigned. Another way in which to provide awareness is by means of the notification history which indicates three types of events: shared files, chat notifications and activity notifications.

Table 4 GUI components associated to features of CWS

Features of CWS	GUI component
remote initial interactions (F1)	• contact list • identity panel
PCA (F2)	• contact view • location label • contact information ToolTip • notification history
shared PAM (F3)	• task panel • notification history
remote communication (F4)	• contact list

IET Softw., 2011, Vol. 5, Iss. 2, pp. 157–171
doi: 10.1049/iet-sen.2009.0097

The notification history is also useful for sharing PAM (F3) as it allows users to share personal documents. Moreover, a task panel has been added to show the different perspectives of a task that is being carried out (e.g. which project or activity is under execution, which resources have been linked etc.).

Finally, a contact list is also useful to support remote communication (F4) as it makes synchronous communications among the collaborators possible. Moreover, a notification history receives the asynchronous communication, through which the users can get to know relevant information about assigned activities in an unobtrusive manner.

Furthermore, CWS-IM allows interaction with target users to be initiated in a more informed way by using the 'selective availability' criterion, that is, it allows users to consider that 'a user is available to interact with other collaborators whose activity is related to the work unit s/he is currently dealing with and is not available to other collaborators'. Thus, the CWS-IM tool may allow an opportunity for interaction between users who are working on a different activity to be created when the time is appropriate for both users to do so. For example, if the status of a user is represented in red (working on another project), CWS-IM allows this user to be 'marked' so that when his/her status changes to green, the tool notifies the user that a timely and convenient opportunity for both users to interact now exists, as they are now working on the same project and on the same activity.

It is important to mention that the aim of the design was to create a notification tool that was less obtrusive. To deal with this, we chose to provide notifications that were sufficiently prominent as to be noticed, but sufficiently subtle and peripheral as to be ignored if so decided. Furthermore, in order to allow issuers to start interactions when the current moment is more appropriate (e.g. the issuer and the receiver are working on the same activities or projects), we provide the issuer with information from the context of work of the receiver, so that the former could decide to proceed with the interaction attempt or to delay it. To achieve this, CWS-IM:

- Provides information of available potential collaborators according to the activity that is being carried out.
- Obtains information about the current activity of colleagues in a transparent way (without interrupting contacts).

Notifies information about personal activities assigned in an implicit manner, through which the user is able to share personal resources with his/her collaborators and to receive personal resources from them. In this case, the tool identifies if the resources belong to the current activity of the collaborator, when this is the case, the tool notifies the collaborator in a synchronous way. Otherwise, it does so in an asynchronous manner.

Interaction always begins when the recipient accepts the invitation of the issuer. That is, the chat window does not open on the receiver's side when she receives the petition to interact, but it appears as an icon (sphere icon) on the taskbar.

Finally, it is important to note that given the nature of our tool, an extended IM application, most of its features require that both the issuer and the receiver of the interaction be working at the same time (synchronous interaction) in order to take benefit of the proposed functionality (as when there is a time zone overlap between them). Regarding asynchronous interaction, the tool basically provides support for the notification and access of shared files, conversation logs and activity logs.

6.3 Implementation aspects

The architecture of this tool is based on a client–server model, which consists of three main parts: The CWS-IM client, the activity presence server and the users and artefacts presence server.

The CWS-IM client is the application on the client side, which was implemented in C #. NET 2008, and uses the agsxmpp library [43]. This library provides the functionality required to connect to an extensible messaging and presence protocol (XMPP) server (Openfire 3.5.1), to retrieve the list of registered users and artefacts, and to send and deliver instant messages.

The activity presence server receives notifications of documents related to some unit of work of the organisation. This server is implemented by the internet information server (IIS 5.1) using web services and the SQL server 2000 DBMS server.

The users and artefacts presence server provides the availability status of CWS clients based on the current activity. This server is implemented by the Openfire 3.5.1 XMPP server. This server uses the XMPP IM protocol.

It is important to mention that the multi-agent system was built using the following technologies: web services, XMPP protocol and simple object access protocol protocol, which are used for message exchange. Additionally, an XML-based ontology is also used, which provides a unique representation for the communication and understanding among agents.

7 Evaluation

This section describes a study to test the tool developed. We present the objectives and hypotheses guiding the study and then provide details on how it was conducted, as well as results and its limitations.

7.1 Objective

The following are the objectives that guided our study:

1. The perceived usefulness and ease of use of CWS-IM to search for a collaborator and to determine his/her work status.
2. The perceived usefulness and ease of use of CWS-IM to find pending assigned activities.
3. The perceived usefulness and ease of use of CWS-IM to obtain information about the current activity that a another person is performing (e.g. progress status).

7.2 Hypotheses

Currently, software industry uses different applications for their activities, including project management systems, the instant messaging and email. With these applications, cross-site workers may have knowledge of the user's presence, their assigned tasks and determine what a colleague is doing. Therefore using the IM it is possible to obtain this knowledge with the support of other applications (e.g. project management system). In contrast, with CWS-IM is possible to obtain the same information but we assume that it could be more useful and easy to use, because it provides an integrated and more functional interface. As a consequence, in order to frame our study we established the following working hypotheses:

H0: No differences exist between CWS-IM and a traditional instant messenger (IM) tool.
H1: The CWS-IM tool is more useful for searching for a collaborator and discovering his/her current status than a traditional IM tool (perceived usefulness).
H2: The CWS-IM tool is easier to use when searching for a collaborator and discovering his/her status than a traditional IM tool (perceived ease of use).
H3: The CWS-IM tool is more useful to discover the pending assigned activities than a traditional IM tool (perceived usefulness).
H4: The CWS-IM tool is easier to use to discover the pending assigned activities than a traditional IM tool (perceived ease of use).
H5: The CWS-IM tool is more useful for obtaining information about the current activity on which a collaborator is working than a traditional IM tool (perceived usefulness).
H6: The CWS-IM is easier to use to obtain information about the current activity on which a collaborator is working than a traditional IM tool (perceived ease of use).

7.3 Design of the study

We performed a comparative study in which the subjects had to work with a traditional IM tool (Windows Live Messenger 8.5 [44]) and CWS-IM. In both cases the subjects had to carry out these three specific tasks:

- Searching for a collaborator and discovering his/her current status.
- Discovering the assigned activities that a person should be performing.
- Obtaining information about the activity that a collaborator was performing at that moment.

In condition 1 the subjects used a traditional IM tool to perform these tasks, and in condition 2 they worked with CWS-IM. In both conditions, subjects had the option to access the project management system.

The study was performed using a within subject paradigm (all subjects participated in both conditions of the study) [45]. The within-subject design is the best way to ensure that the groups that work in the different experimental conditions are as similar as possible, since in reality they are the same subjects. In our case, this signified that all the subjects participated in both work conditions (they worked with traditional IM and with CWS-IM). This had the advantage of guaranteeing control of all the variables related to the difference between subjects that might have contaminated the results of the investigation.

In our experiment there was one independent variable: the work condition (with traditional IM or CWS-IM). The variables affected by this were:

- *usefulness:* in order to test whether one tool was more useful than another for a set of tasks and
- *ease of use:* in order to test whether one tool was easier to use than another.

7.4. Subjects

The participants were 16 workers from different companies in three different cities of the state of Sonora, Mexico. All of them (16) were participating or had participated in DSD projects. Their average age was 30.25 years of age, and seven of them had at least three years experience in DSD, whereas nine of them had less than three years experience in DSD. All of them (16) had concluded their bachelor (BSc) degrees in computer science or a related subject and one had graduated with a master (MSc) degree in computer science.

The subjects were divided into two groups. This division was made by taking into account their experience in DSD. Each group consisted of eight people, four of whom had three or more years experience in DSD and the rest of whom had less than four years experience. The situation of having more experienced people in one group than in the other was therefore avoided.

7.5 Procedure

The subjects attended an initial meeting (10 min approximately) in which they were informed about the tasks that they had to carry out. Later, in a first session which lasted 15 min, they solved three tasks using the IM (group 1) and using the CW-IM (group 2). In the second session which lasted 15 min, they solved the same tasks but using the tool that they had not used during the first session.

The tasks consisted of three scenarios:

1. Search for a collaborating partner and determine his/her work status: the subject is contextualised in the revision of a document (usecase.vsd), which was conducted by a colleague who has been previously identified. Then, the subject is questioned: what is she doing?. So the subject uses the tool and tries to determine the current activity of the colleague.
2. Query the pending assigned activities: the subject was contextualised in the need to know which activity a colleague should be performing at that time. For that, the subject uses the tool to obtain information about the pending assigned activities per project. In this scenario, subjects had the option to use the project management system.
3. Gather information about the activity being performed by the collaborating colleague: The subject was contextualised in the need to know if a colleague was able to complete its task in order to start a new shared activity. So that, the subject could use the tool, and optionally the project management system. In this scenario, the subject seeks to find a notification about a particular activity.

After finishing each scenario (three per session) participants were interviewed and asked to respond to standard technology acceptance model (TAM) [46] questionnaires in order to obtain their perception with regard to the usefulness and ease of use of the proposed tool.

Obtained data were processed using descriptive statistics (see Tables 5–7) in order to quantify the perception levels provided by the participants regarding usefulness and ease of use.

7.6 Limitations

The described case study and the methods used in order to evaluate the tool might have had weaknesses. The possible influence of these weaknesses on the results is explained below:

- The within subject studies has the problem of learning. We attempted to avoid this effect by showing the scenarios in

IET Softw., 2011, Vol. 5, Iss. 2, pp. 157–171
doi: 10.1049/iet-sen.2009.0097

IET Softw., 2011, Vol. 5, Iss. 2, pp. 157–171
doi: 10.1049/iet-sen.2009.0097

Table 5 TAM results regarding scenario 1

Question	IM, Sc1	CWS-IM, Sc1
Usefulness		
Q1: using the system would enable me to accomplish tasks more quickly	4 (1.549)	5.81 (0.655)
Q2: using the system would improve my performance	3.44 (1.504)	5.31 (1.302)
Q3: using the system would make it easier to do my work chores	3.81 (1.276)	5.69 (1.078)
Q4: using the system would enhance my effectiveness at work	3.44 (1.459)	5.25 (1.483)
Q5: using the system display would increase my productivity	3.19 (1.328)	4.81 (1.223)
Q6: i would find the system useful in my work	4.81 (1.109)	6.13 (0.806)
average	3.84	5.54
Ease of use		
Q7: i find it easy to get the system to do what I want it to do	4.13 (1.821)	5 (1.155)
Q8: my interaction with the system is clear and understandable	5.44 (1.365)	5.88 (1.025)
Q9: learning to operate the system would be easy for me	5.75 (1.291)	5.75 (0.856)
Q10: it would be easy for me to become skillful at using the system	6.25 (0.775)	6 (0.632)
Q11: i would find the system easy to use	5.19 (1.167)	6.38 (0.719)
Q12: i would find the system flexible to interact with	5.69 (1.195)	5.88 (0.885)
average	5.34	5.77

Table 6 TAM results regarding scenario 2

Question	IM, Sc2	CWS-IM, Sc2
Usefulness		
Q1: using the system would enable me to accomplish tasks more quickly	4.63 (1.025)	6.31 (0.793)
Q2: using the system would improve my performance	3.69 (1.493)	5.75 (0.856)
Q3: using the system would make it easier to do my work chores	4.13 (1.310)	5.63 (1.088)
Q4: using the system would enhance my effectiveness at work	3.56 (1.413)	5.63 (1.147)
Q5: using the system display would increase my productivity	3.50 (1.461)	5.44 (1.153)
Q6: i would find the system useful in my work	5.13 (1.668)	6.38 (0.500)
average	4.16	5.86
Ease of use		
Q7: i find it easy to get the system to do what I want it to do	4.63 (1.360)	5.25 (0.856)
Q8: my interaction with the system is clear and understandable	5.13 (1.688)	6 (0.816)
Q9: learning to operate the system would be easy for me	5.69 (1.352)	6 (0.894)
Q10: it would be easy for me to become skillful at using the system	6.06 (0.929)	6.19 (0.750)
Q11: i would find the system easy to use	5.44 (1.459)	6.44 (0.727)
Q12: i would find the system flexible to interact with	5.50 (1.211)	6.06 (0.772)
average	5.35	5.98

Table 7 TAM results regarding scenario 3

Question	IM, Sc3	CWS-IM, Sc3
Usefulness		
Q1: using the system would enable me to accomplish tasks more quickly	4.63 (1.893)	6.44 (0.512)
Q2: using the system would improve my performance	4.13 (1.500)	5.81 (0.834)
Q3: using the system would make it easier to do my work chores	4.19 (1.642)	6.38 (0.619)
Q4: using the system would enhance my effectiveness at work	3.81 (1.797)	5.88 (0.885)
Q5: using the system display would increase my productivity	3.81 (1.759)	5.88 (0.806)
Q6: i would find the system useful in my work	4.38 (2.094)	6.69 (0.602)
average	4.31	6.17
Ease of use		
Q7: i find it easy to get the system to do what I want it to do	4.44 (1.315)	5.75 (0.856)
Q8: my interaction with the system is clear and understandable	5.56 (1.263)	6.25 (0.577)
Q9: learning to operate the system would be easy for me	6 (1.095)	6.31 (0.704)
Q10: it would be easy for me to become skillful at using the system	6.31 (0.704)	6.25 (0.775)
Q11: i would find the system easy to use	5.31 (1.537)	6.63 (0.619)
Q12: i would find the system flexible to interact with	5.63 (1.147)	6.25 (0.683)
average	5.39	6.25

different order. However it is impossible to state whether this effect was totally controlled or avoided.

- People usually respond to questionnaires with what they think the experimenter wants to hear. We attempted to avoid this by asking the subjects not to write their name on the questionnaires. People are usually more sincere when they do not need to give their name.
- This study is focused on usefulness and ease of use of the data obtained from users' opinions. However, we did not have any other quantitative data to test, for example, whether users spent more time performing a task by using a particular tool. This is part of our future work.

7.7 Results

As stated earlier, we asked participants about their general perception of usefulness and ease of use by using standard TAM questionnaires [46]. The questionnaire thus consisted of two sections (with six questions each) as presented in Tables 5–7. All questionnaire items were measured on a 7-point Likert scale, ranging from 1 ('completely disagree') to 7 ('completely agree').

We used the SPSS tool version 15.0.1 for Windows [47] to analyse the data obtained from the questionnaires. The statistic used to compare means in independent samples is the *t*-Student, and this was chosen in order to compare whether there was significant difference between means. To decide whether participants preferred one tool over another for a given scenario, the *t*-test calculates a *p*-value based on the performance data. If the *p*-value is small enough (p-value < 0.05), we conclude that the difference is significant.

Therefore Table 5 shows the means for scenario 1. This table was useful in testing hypotheses 1 and 2 with regard to scenario 1 (searching for a collaborating partner and determining his/her work status).

In this case the mean average for usefulness when using the CWS-IM was 5.54, and 3.84 when using traditional IM. This difference is significant, since $t_{0.01} = 4.976$ and p-value $= 0.000$, since p is smaller than 0.05. These values therefore argue in favour of the proposition that the CWS-IM tool is more useful for searching for a collaborator and discovering his/her current status than traditional IM.

On the other hand, with regard to ease of use the mean average was 5.77 when using CWS-IM and 5.34 when using traditional IM. So in this case there was no significant difference, since the values obtained were $t_{0.01} = 1.559$ and $p = 0.130$ being $p > 0.05$. Hence it is not possible to state that the CWS-IM tool is easier to use than IM when searching for a collaborator and discovering his/her status.

Table 6 depicts the means for scenario 2 (query the pending assigned activities). These results helped us to test whether hypotheses H3 and H4 could be accepted.

In this case, for the usefulness factor the value of the mean average was 5.86 when using CWS-IM and 4.16 when using traditional IM. This is a significant difference, since $t_{0.01} = 5.501$ and $p = 0.000$. We can therefore state that the CWS-IM tool is more useful than IM in discovering the pending assigned activities. With regard to ease of use, the mean average when using CWS-IM was 5.98, and 5.32 in the case of traditional IM. As the results after applying the *t*-Student statistic were $t_{0.01} = 2.119$ and $p = 0.042$ it is possible to consider that the CWS-IM tool is easier to use to discover the pending assigned activities than traditional IM.

Table 7 shows the means for scenario 3 (gathering information about the activity being performed by the collaborating colleague), and these data were used to calculate whether hypotheses 5 and 6 were significant.

For the usefulness factor we obtained that the mean average was 6.17 when using CWS-IM and 4.31 when using traditional IM. The difference between both means is therefore significant, since $t_{0.01} = 4.814$ and $p = 0.000$. Consequently, it is possible to state that the CWS-IM tool is more useful than traditional IM in obtaining information about the current activity on which a collaborator is working.

For the ease of use factor the mean average was 6.25 when the subjects used CWS-IM and 5.39 when they used traditional IM. Again the difference between the means is significant, since $t_{0.01} = 3.203$ and $p = 0.003$. That signifies that the CWS-IM tool is easier to use than traditional IM to obtain information about the current activity on which a collaborator is working.

7.8 Discussion

Here we analyse each of results of testing the hypotheses and discuss the implications:

In the case of H1, we believe that the subjects considered CWS-IM to be more useful than IM for searching for a collaborator and discovering his/her current status because the first tool has a mechanism that allows workers to know on what particular task a person is working, the role that the person is playing at that precise moment and even the name of the document that the person is working on. Moreover, a person can search for people according to the project on which they are working.

In the case of H2 we were unable to accept this hypothesis. This could mean either that CWS-IM is as easy to use as traditional IM, or that traditional IM is easier to use than CWS-IM. A possible explanation for the latter case (worst-case scenario) is that when a person is used to working with a particular tool, starting to work with another one is often difficult and at first glance the second tool usually appears to be more complex. This might be one reason why the subjects did not believe that CWS-IM was easier to use than IM when searching for a collaborator and discovering his/her status. More experiments will, of course, be carried out in order to determine which of the two possibilities was actually considered. In case of the worst-case scenario, we will focus in determining which parts of the interface might have seemed difficult to use or how they could be improved in order to facilitate its use.

On the other hand, we were able to accept hypothesis H3 as the subjects considered CWS-IM to be more useful than IM in discovering the pending assigned activities. The possible reason for this is that CWS-IM has a panel on which the users' tasks are displayed and the user can see the assigned tasks, as well as consult the deadline by which they must be carried out.

Hypothesis H4 was also accepted as workers considered that it was easier to discover pending tasks by using CSW-IM. A possible reason for this could be that CWS-IM allows to automatically obtain the information related to the assigned tasks from the project repository. Therefore the user can easily access this information from their interface. In the IM case the user has to consult the project manager to obtain this information or consult additional tools, signifying that the user has to make an extra effort which is unnecessary when this information is at the GUI interface of the provided tool.

Finally, the fact that H5 and H6 were accepted indicates that our tool is useful and easy to use to obtain information regarding the current activity on which a collaborator is

working, in comparison to traditional IM. Given that our goal was to develop a tool that would help users to discover the best moment at which to initiate an interaction with another person, and that evaluation results provide evidence indicating that they could discover the status of a person as well as the activity on which a person is working, we believe that CWS-IM tool would be useful and well accepted in the context of DSD.

8 Conclusions and future work

Software development organisations are confronting a working philosophy shift towards DSD, that is the distribution of development processes and teams. This shift brings with it both benefits and challenges to which organisations must adapt. The benefits include access to highly skilled human resources, development groups closer to client location, 24/7 development cycles for critical projects taking advantage of different times zones and reduced recruitment costs at places with cheaper labour rates. The challenges include limitations caused by cultural differences, coordination difficulties resulting from different time zones, inadequate and problematic knowledge management and communication owing to language and cultural differences and to geographical distribution. Moreover, there is a lack of trust in inter-team relationships.

In this work, we have particularly addressed the lack of timely adequate opportunities for informal interactions, which has been identified as an underpinning foundation to overcome coordination, communication and trust limitations. We have attempted to tackle this problem by introducing and defining the concept of CWS.

We have used a literature study to identify a set of features of DSD activities, which has then been used to establish an ensemble of design issues for a tool that aims to support the timely and appropriate initiation of interactions in a DSD environment. Taking these as a basis, we have developed a tool implementing the concept of CWS.

CWS were mainly proposed to support the exchange of information between the members of a work group who are in different geographical locations in a lightweight and simple manner. Furthermore, CWS includes support to allow an interaction to be initiated in a selective manner according to a criterion that we refer to as 'selective availability', that is, to consider that 'I am available more willingly to interact with collaborators who are related to the work unit I am dealing with now and less (or not) available to other collaborators'. Furthermore, CWS could define how software developers' learned (tacit) and generated (explicit) knowledge can be of help to improve the DSD development process. Therefore the main features of the tool with CWS are that it:

- distributes explicit organisational knowledge (e.g. project status information) in real time;
- provides explicit information through specialised mechanisms to specific project- and activity-related team members;
- offers an individual perspective to the DSD group (e.g. availability of information regarding each individual related to the project according to the assigned work and the activity currently under execution), along with a group perspective with regard to a specific team member (e.g. it notifies whenever a developer is available to certain project-related members, and unavailable to others).

In order to evaluate our proposals a case study was performed. According to our evaluation results we obtained that CWS-IM can improve the way in which developers initiate interaction in comparison to how this is done by using current IM tools, since the proposed tool helps users to know what each person is working on at each precise moment. Therefore if one person wishes to initiate an interaction with another one, the first can wait until that other person starts to work on the same project, or preferably the same activity, on which the first person is working. Hence, it is assumed that the interaction is less disturbing as this will be related to his/her work at hand. Interactions with CWS-IM could consequently be more productive and less disturbing than those made when using traditional IM tools.

Further experiments will be performed in order to obtain additional results with which to validate our proposals. We are therefore planning to use a CWS-based tool in a software factory located in three different cities in the state of Sonora, Mexico.

9 Acknowledgments

The authors would like to thank Novutek, S.C. (Obregon, Sonora, México), Novutek, S.C. (Guaymas, Sonora, México), Celulosa y Corrugados de Sonora, S.A. de C.V. (Navojoa, Sonora, México), Competitiv-IT Strategic Solutions (Navojoa, Sonora, México) and the 16 participants for their valuable support and participation which made the evaluation possible. The authors also acknowledge the support of the British Council which, through the Research Exchange Program 2006–2007, provided the seed funding to establish this collaboration. This work was partially supported by CONACYT with a postdoctoral scholarship to the third author and by UABC under grant 0207 of the XIII Convocatoria Interna de Proyectos de Investigación. The first author is supported by scholarship PROMEP/103.5/06/3244. This work is partially supported by the ENGLOBAS (PII2I09-0147-8235) and MELISA (PAC08-0142-3315) projects, Junta de Comunidades de Castilla-La Mancha, Consejería de Educación y Ciencia, in Spain. It is also supported by PEGASO (Ministerio de Ciencia e Innovacion MICINN and Fondo Europeo de Desarrollo Regional FEDER, TIN2009-13718-C02-01) and the FABRUM project (grant PPT-430000-2008-063), Ministerio de Ciencia e Innovación, in Spain. This work has been also partially supported by Asociación Mexicana de Cultura A.C.

10 References

1 Prikladnicki, R., Audy, J.L.N., Evaristo, J.R.: 'Distributed software development: toward an understanding of the relationship between project team, users and customers'. Proc. Fifth Int. Conf. on Enterprise Information Systems (ICEIS 03), 2003, pp. 417–423
2 Damian, D., Zowghi, D.: 'The impact of stakeholders'? Geographical distribution on managing requirements in a multi-site organization'. Proc. Tenth Anniversary IEEE Joint Int. Conf. on Requirements Engineering, 2002, pp. 319–330
3 Layman, L., Williams, L., Damian, D., Bures, H.: 'Essential communication practices for extreme programming in a global software development team', *Inf. Softw. Technol.*, 2006, **48**, (9), pp. 781–794
4 Conchúir, E.Ó., Ågerfalk, P.J., Olsson, H.H., Fitzgerald, B.: 'Global software development: where are the benefits?', *Commun. ACM*, 2009, **58**, (2), pp. 127–131
5 Ebert, C., Neve, P.D.: 'Surviving global software development', *IEEE Softw.*, 2001, **18**, pp. 62–69
6 Richardson, I., Casey, V.D., Zage, D., Zage, W.: 'Global software development – the challenges' (University of Limerick, Ball State University, 2005)
7 Damian, D., Moitra, D.: 'Guest editors' Introduction: global software development: how far have we come?', *IEEE Softw.*, 2006, **23**, (5), pp. 17–19
8 Lloyd, W., Rosson, M., Arthur, J.: 'Effectiveness of elicitation techniques in distributed requirements engineering'. Proc. Tenth anniversary IEEE Joint Int. Conf. on Requirements Engineering, (RE'02), 2002, pp. 311–318
9 Carmel, E., Agarwal, R.: 'Tactical approaches for alleviating distance in global software development', *IEEE Softw.*, 2001, **18**, (2), pp. 22–29
10 Herbsleb, J., Moitra, D.: 'Guest Eds.' Introduction: global software development', *IEEE Softw.*, 2001, **18**, (2), pp. 16–20
11 Whittaker, S., Schwarz, H.: 'Meetings of the board: the impact of scheduling medium on long term GroupCoordination in software development'. Comput. Supported Cooperative Work, 1999, vol. 8, pp. 175–205
12 Conchúir, E., Holmstrom, H., Agerfalk, P., Fitzgerald, B.: 'Exploring the assumed benefits of global software development'. Proc. IEEE Int. Conf. on Global Software Engineering (ICGSE'06), 2006, pp. 159–168
13 Kraut, R.E., Streeter, L.A.: 'Coordination in software development', *Commun. ACM*, 1995, **38**, (3), pp. 69–81
14 Cataldo, M., Bass, M., Herbsleb, J.D., Bass, L.: 'On coordination mechanisms in global software development'. Proc. Int. Conf. on Global Software Engineering, 2007, pp. 71–80
15 Morán, A.L., Favela, J., Romero, R., Natsu, H., Perez, C., Robles, O., Martinez, A.: 'Potential and actual collaboration support for distributed Pair-Programming', *Comput. sistemas*, 2008, **11**, (3), pp. 211–229
16 Morán, A.L., Favela, J., Martinez, A., Decouchant, D.: 'On the design of potential collaboration spaces', *Int. J. Comput. Appl. Technol. (IJCAT)*, 2004, **19**, (3/4), pp. 261–271
17 Ågerfalk, P.J., Fitzgerald, B., Holmström, H., Lings, B., Lundell, B., Conchúir, E.Ó.: 'A framework for considering opportunities and threats in distributed software development'. Proc. Int. Workshop on Distributed Software Development (DiSD 2005), 2005, pp. 47–61
18 Ye, Y.: 'Supporting software development as knowledge-intensive and collaborative activity', 'Book Supporting software development as knowledge-intensive and collaborative activity' (Series supporting software development as knowledge-intensive and collaborative activity, ACM, 2006)
19 Czerwinski, M., Horvitz, E., Wilhite, S.: 'A diary study of task switching and interruptions'. Proc. the SIGCHI Conf. on Human Factors in Computing Systems, 2004, pp. 175–182
20 Bailey, B.P., Iqbal, S.T.: 'Understanding changes in mental workload during execution of goal-directed tasks and its application for interruption management', *ACM Trans. Comput.-Hum. Interact.*, 2008, **14**, (4), pp. 1–28; http://doi.acm.org/10.1145/1314683.1314689
21 Ellis, J., Kvavilashvili, L.: 'Prospective memory in 2000: past, present and future directions', *Appl. Cogn. Psychol.*, 2000, **14**, pp. 1–9
22 Brush, A.B., Meyers, B.R., Tan, D.S., Czerwinski, M.: 'Understanding memory triggers for task tracking'. Proc. SIGCHI Conf. on Human Factors in Computing Systems, 2007, pp. 947–950
23 Cutrell, E.B., Czerwinski, M., Horvitz, E.: 'Effects of instant messaging interruptions on computing tasks'. Proc. CHI '00 Extended Abstracts on Human Factors in Computing Systems, 2000, pp. 99–100
24 Fussell, S., Kiesler, S., Setlock, L.D., Scupelli, P.: 'Effects of instant messaging on the management of multiple project trajectories'. Proc. SIGCHI Conf. on Human Factors in Computing Systems, 2004, pp. 191–198
25 González, V., Galicia, L., Favela, J.: 'Supporting the planning and organization of multiple activities in the workplace'. Proc. INTERACT 2007, 2007, pp. 235–238
26 Morteo, R., González, V., Favela, J., Mark, G.: 'Sphere Juggler: fast context retrieval in support of working spheres'. Proc. ENC 2004, 2004, pp. 361–367
27 Camacho, J., Favela, J., González, V.: 'Supporting the management of multiple activities in mobile collaborative working environments'. Proc. CRIWG 2006, 2006, pp. 381–388
28 Espinosa, J.A., Carmel, E.: 'The impact of time separation on coordination in global software teams: a conceptual foundation', *J. Softw. Process Pract. Improv.*, 2004, **8**, (4), pp. 249–266
29 Sarma, A., Hoek, V.A.D.: 'Palantír: increasing awareness in distributed software development'. Proc. Workshop on Global Software Development (ICSE 2002), 2002, pp. 28–32
30 Gutwin, C., Schneider, K.A., Paquette, D., Penner, R.: 'Supporting group awareness in distributed software development'. Proc. EHCI/DS-VIS, 2004, pp. 383–397
31 Schnädelbach, H., Penn, A., Steadman, P., Benford, S., Koleva, B., Rodden, T.: 'Moving office: inhabiting a dynamic building', in 'Book Moving office: inhabiting a dynamic building' (Series Moving office: inhabiting a dynamic building, ACM, 2006)
32 Biehl, J.T., Czerwinski, M., Smith, G., Robertson, G.G.: 'FASTDash: a visual dashboard for fostering awareness in software teams', 'Book FASTDash: a visual dashboard for fostering awareness in software teams', (Series FASTDash: a visual dashboard for fostering awareness in software teams, ACM, 2007)
33 Biehl, J.T., Baker, W.T., Bailey, B.P., Tan, D.S., Inkpen, K.M., Czerwinski, M.: 'Impromptu: a new interaction framework for supporting collaboration in multiple display environments and its field evaluation for co-located software development', 'Book impromptu: a new interaction framework for supporting collaboration in multiple display environments and its field evaluation for co-located software development' (Series Impromptu: a new interaction framework for supporting collaboration in multiple display environments and its field evaluation for co-located software development, ACM, 2008)
34 Holmes, R., Walker, R.J.: 'Promoting developer-specific awareness', 'Book promoting developer-specific awareness' (Series Promoting developer-specific awareness, ACM, 2008)
35 Bellotti, V., Ducheneaut, N., Howard, M., Smith, I.: 'Taskmaster: recasting email as task management'. Proc. Position Paper for the CSCW'02 Workshop on Redesigning Email for the 21st Century, 2002
36 Handel, M., Herbsleb, J.D.: 'What is chat doing in the workplace?'. Proc. 2002 ACM Conf. on Computer Supported Cooperative Work, 2002, pp. 1–10
37 Isaacs, E., Walendowski, A., Ranganthan, D.: 'Hubbub: a sound-enhanced mobile instant messenger that supports awareness and opportunistic interactions', 'Book Hubbub: a sound-enhanced mobile instant messenger that supports awareness and opportunistic interactions' (Series Hubbub: a sound-enhanced mobile instant messenger that supports awareness and opportunistic interactions, ACM, 2002)
38 Gonzalez, V., Mark, G.: 'Constant, constant, multi-tasking craziness: managing multiple working spheres'. Proc. SIGCHI Conf. on Human Factors in Computing Systems, 2004, p. 2004
39 McChesney, I.: 'Effective coordination in the software process – historical perspectives and future directions', *Softw. Qual. Control*, 1997, **6**, (3), pp. 235–246
40 Pressman, R.S.: 'Software engineering: a practitioner's approach' (McGraw Hill, 2005), p. 873
41 Palacio, R.R., Moran, A.L., Gonzalez, V.M., Vizcaino, A.: 'Collaborative working spheres as support for starting collaboration in distributed software development'. Proc. 13th Int. Conf. on Computer Supported Cooperative Work in Design, 2009, 2009, pp. 636–641
42 Pavón, J., Gómez-Sanz, J.J.: 'Agent oriented software engineering with INGENIAS'. Proc. Multi-Agent Systems and Applications III: CEEMAS 2003 (LNAI), 2003, pp. 394–403
43 A. Software: 'agsXMPP SDK', 2000; http://www.ag-software.de/agsxmpp-sdk.html
44 Microsoft: 'Windows Live Messenger', 2007; http://windowslive.com/desktop/messenger
45 Cotton, J.W.: 'Analyzing within-subjects experiments' (Lawrence Erlbaum Associates, 1998), p. 331
46 Davis, F.: 'Perceived usefulness, perceived ease of use, and user acceptance of information technology', *MIS Q.*, 1989, **13**, pp. 319–340
47 S. Inc.: 'SPSS 15 for Windows' (SPSS Inc, 2006)