Open Access

CWS: An Awareness Tool to Support Starting Collaboration in Global Software Development

Ramón R. Palacio[1,2], Alberto L. Morán[3,4,*] and Víctor M. González[5]

[1]*Facultad de Ingeniería - Ensenada, UABC, México*

[2]*DES Navojoa, ITSON, México*

[3]*Facultad de Ciencias, UABC, México*

[4]*Ciencias de la Computación, CICESE, México*

[5]*Manchester Business School, University of Manchester, UK*

Abstract: Software development organizations are facing a paradigm shift towards Distributed Software Development. This shift introduces situations from which organizations may benefit (e.g. highly skilled human resources, development groups closer to client location, etc.); but also introduces challenges to which organizations have to adapt (e.g. coordination difficulties, inadequate knowledge management and communication, and lack of inter-virtual-team trust relationships). In this work, we particularly study the lack of timely adequate opportunities for informal interaction, which has been identified as an underpinning foundation to overcome coordination, communication and trust limitations. To achieve this, we introduce and define the concept of Collaborative Working Spheres (CWS), through which developers can obtain information related to the personal activities of their distributed colleagues. CWS allow identifying opportunities for collaboration in suitable moments both for the one making contact and the one being contacted. We notice that other examples of technologies, including the telephone and instant messaging are used by developers for starting collaboration; however, they do not provide enough information from the personal activity of the person being contacted. We argue that with CWS, software developers will be able to become aware about the status and progress their partners have achieved in some activity, and use this information to inform their starting collaboration. We illustrate this concept with the design of a CWS-based messenger tool that supports Collaborative Working Spheres for Distributed Software Developers. The results of an initial evaluation provide encouraging evidence on the perceived usefulness and ease of use of the proposed CWS-based messenger tool.

1. INTRODUCTION

Organizations dedicated to software development are increasingly facing a 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. Paradoxically, the shift also introduces negative aspects such as an increased risk of communication problems [2, 3].

In general, there are a number of characteristics that define scenarios for DSD. One of these is the distance between individual members or teams, which can vary from a few meters up to tens, hundreds or thousands of kilometers. A special case of DSD is where the distances are among cities from different countries, even continents, and this is known as Global Software Development or GSD [4].

The literature reports that DSD provides the following benefits: i) Software companies require highly skilled human resources and seek to meet this need by employing programmers in different cities and countries [5, 6]. ii) 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]. iii) Virtual development groups need to be created quickly to exploit new opportunities in this market [8]. iv) By working in different time zones, development groups can work continuously in critical projects [9, 10]. v) The reduction of costs for recruitment of human resources from places where labor is cheaper [11, 12]. However, it has been found that DSD also faces interesting challenges, such as: i) cultural differences [10, 13], ii) time differences [2], iii) inadequate communication [7, 14] and knowledge management [12], and v) difficulty to maintain trust relationships [15] between working groups.

A fundamental aspect of work in software development environments is that it is characterized by the high communication and coordination level among participants [16]. This is due to the need to achieve decision consensus among members of a group (e.g. requirements approval among customers and analysts, change acceptance by developers and change control engineers), decisions must be made with precision and expeditiously (e.g. the notification and delivery of new versions of a design document), interaction among team members is regular and frequent (e.g. information request and delivery of tasks progress reports), and is cooperative and collaborative work (e.g. programming

*Address correspondence to these authors at the Facultad de Ciencias, UABC, México; Tel: +52 646 1744560-ext. 115; Fax: +52 646 1744560-ext. 106; E-mail: alberto.moran@uabc.edu.mx

and collaborative work (e.g. programming in peer group review and creation of a design specification document).

However, dealing with these features is completely dependent on the way the development process is being performed. In the case of co-located development, project members are at sight or easily accessible, so that it is possible to see or know what they are doing, without a significant effort. One can even judge whether the current moment is adequate to interrupt what others are doing in order to establish communication and maintain a coordination effort. In contrast, in the DSD case, participants are located at distant sites, so that the contextual information existing in a co-located situation is not there anymore, which hinders the communication, coordination and production processes. For this reason, it is essential to know the state of the activities of the person one wishes or requires to contact, and thus try to find an adequate moment for both participants, in order to start an interaction while minimizing the negative effects of an interruption. That is, information is required to know the context of development of a participant to be contacted in order to potentiate collaboration [17, 18].

Given the need to diminish the impact of the disruption to the work being done by the person being sought, the first questions to guide this work emerge: What is needed to initiate collaboration in DSD? And how an appropriate time to do it can be identified? An additional question is which are the appropriate mechanisms to obtain, deliver and present the information to identify the appropriate time to initiate the collaboration?

Through empirical studies various researchers have investigated how workers are interrupted while executing their activities (e.g. [19]), showing how 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 characterize how workers behave while facing interruptions (interruption management). However, these studies do not provide enough elements on how to select the right time to start the interaction.

It has also been found that activities can be planned and organized through the use of project management, which helps to reduce uncertainty in the workplace [20]. However, project management despite being a common practice in these organizations, in the DSD context it is limited by the lack of informal communication [16], which generates low levels of trust between colleagues, and poor knowledge about their work and progress at remote sites [4, 21, 22].

To cope with these limitations, other works (e.g. [23]) propose to use instant messaging (IM) as support for informal communication, and thus to complement the management of individual work in collaborative projects by providing awareness information of multiple collaborators on multiple tasks through the IM interface. However, this proposal does not consider using awareness information to promote starting collaboration, and focuses mainly on providing information about the user's identity, project team membership, and general availability (offline, online, available, busy, etc.).

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 [24]. PAM is based on the analysis of processes and strategies that are involved with the way in which workers face the planning and management of their activities. PAM is related to the concept of Working Sphere, which explains how people as individuals organize their work, and is flexible enough as to represent the activities with the required degree of granularity [25, 26]. Using the concept of Working Sphere can be useful as it provides elements of activities and / or tasks performed by people (e.g. resources, repositories or related applications used by the individual) that can potentially be used to identify an opportunity to get into collaboration through the monitoring of context (e.g. through Potential Collaboration Awareness [17]). The concept of Potential Collaboration Awareness (PCA) is important because it enables finding the most appropriate time to establish interaction.

Thus, in this work we propose to integrate a PAM perspective of DSD workers with a PCA approach to provide support for starting or getting into collaboration 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 rest of the paper is organized as follows. Section 2 briefly presents the methodology followed to perform this work. Section 3 presents the results of a literature review that establish an initial set of features of DSD activities, while Section 4 presents an ensemble of design insights and features of Collaborative Working Spheres. Section 5 presents some of the design and implementation details of our proposed tool, while section 6 presents the main results of an evaluation of this prototype concerning the perception of ease of use and usefulness. Finally, section 7 presents our concluding remarks and directions of future work.

2. METHODOLOGY

Fig. (**1**) presents the methodology followed to pursue the objectives of this research.

Firstly, in phase 1 through a literature review we obtained an initial understanding regarding the activities and roles of DSD workers. This allowed us to characterize these activities by means of an ensemble of features which were central for collaboration initiation and coordination.

Secondly, based on the identified features of DSD, in phase 2 we envisioned how to support the needs previously identified. This was achieved through the use of scenarios [27] which allowed us to project our vision and further frame our understanding. This resulted in a set of design insights, and on a conceptualization of our proposed approach, the Collaborative Working Sphere (CWS).

Thirdly, to actually provide support to start timely suitable collaboration efforts in a DSD setting, and informed by the previously defined design insights and CWS concepts, we designed and implemented a prototype CWS tool in phase 3.

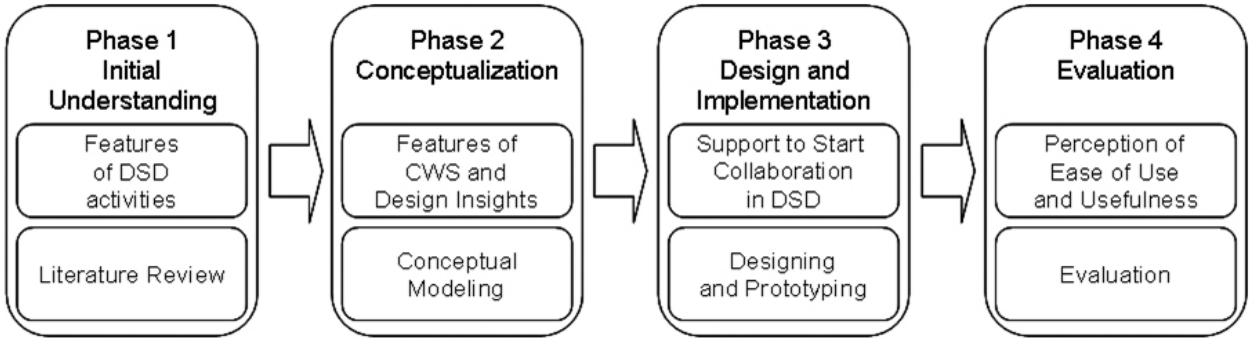

Fig. (1). Methodology for creating and evaluating a system to provide support to start collaboration in DSD.

Finally, in phase 4 we evaluated the system with a group of DSD workers in order to determine their perceived ease of use and usefulness regarding the proposed tool.

3. PHASE 1: FEATURES OF DSD ACTIVITIES

Software development is a complex task in itself, and thus, the characterization of its activities is therefore also a complex task due to the coordination problems that arise during its implementation. Unfortunately, these coordination problems are not only very common, but also inevitable [16].

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 organization. Accordingly, based on the proposals of [16] and [28], we established a set of elements through which to characterize DSD development activities. These are described as follows:

a) 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 organization and between organizations).

- Geographic distribution of the participants (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 the product size (small, medium, large and very large).

b) 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 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.

c) 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 her activities will be before being able to start them.

- Producer-consumer relationship: when a worker has to complete 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 her product available to another worker who needs it to perform her activities.

- Utility: when a consumer worker receives a product, she is dependent upon its degree of usefulness to perform 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.

d) 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 when in the search for partners related to the task being carried out, in taking or leaving messages for col-

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 about the progress status per work unit Control the project specifications Knowledge about the work units assigned to other people Knowledge about the general objective of the program Knowledge about the goals where the work units have an impact Knowledge about the program charter that drives the program
Interdependence	Awareness on 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 program Access to resources internal to the program 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 communications means

leagues, 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, and so forth. (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 others.

Based on the previous characterization, we identified some interaction needs that DSD workers face 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*: it has been previously discussed under the perspective of informal interaction and artificial proximity, due to the need to bring workers closer to each other through informal interactions. ii) *Uncertainty*: this perspective is given by project management, due to the need to know information related to the different projects under execution. iii) *Interdependence*: this perspective is based on proposals from informal awareness, Working Spheres and informal interaction due to the need for awareness of what involved workers are doing at a given moment to coordinate themselves through brief interactions. iv) *Communication*: this perspective is based on interaction, informal awareness and PCA due to their need to exchange information among involved workers.

The identified needs for interaction result from the characterization 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. As for formal communication, there are formal methods and commercial software applications that support it adequately. In contrast, despite informal interactions being rich in content, they are generally not regarded as an important asset within organization [16].

4. PHASE 2: COLLABORATIVE WORKING SPHERES

4.1. Design Implications

Based on the features identified for DSD activities, in Table **2** we identify some design implications needed to provide adequate support for starting collaboration between DSD workers. The table presents opportunity areas and matches them to some specific design implications. They are briefly described below:

i) Regarding *Scale*, it is required that technology provides communication services that allow interaction between people who are distributed in different locations, in order to guarantee participation of all those involved in a certain project.

ii) Concerning *Uncertainty*, it requires that technology provides a mechanism for sharing project information among colleagues or any other related people, as this will allow determining the progress status of activities assigned to people and of the project itself.

iii) Regarding *Interdependence*, it is required, firstly, a mechanism to ascertain the degree of progress of the work being done by each member of the project, this will mean increased information about what is being done in the context, and secondly, a mechanism that allows locating and

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 communications among people
Uncertainty	Knowledge about the progress status of colleagues on their assigned work units Knowledge about the progress status of the project	I2. Mechanisms to share and filter project information among colleagues of 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 locating and interacting with members of common or dependent work units
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 communications means	I5. Mechanisms that allow knowing the status of project members, as well as of the tasks they are performing I6. Mechanisms to identify when a user may interact with another one, based on the needs profile, status and activity under execution I7. Services for synchronous and asynchronous communication

interacting with members sharing a common or related tasks, so that they are aware and anticipate situations that affect the DSD activities.

iv) Concerning *Informal Communication*, it is required, firstly, a mechanism to ascertain the status of the members of the project and the work they are performing to have a better understanding of the convenience of attempting an interaction (i.e. interrupt the worker); secondly, a mechanism that identifies when a user may interact with another, based on the interaction need, status and the work being performed both of the worker who wants to contact and the one being contacted; and thirdly, synchronous and asynchronous communication mechanisms, to provide support for both real-time interaction, and for situations where messages would not be delivered immediately, but stored for later viewing.

4.2. Features of CWS

The features of DSD activities, the identified needs, and the proposed design implications have led us to adopt a conceptual unit of work that allows us to understand the management of an individual developer's work activities: the Working Sphere (WS) [29]. A WS is a concept that serves to describe the units of work that people use to organize 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 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 to analyze 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 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 and design insights of DSD activities. A large proportion of design insights are considered by potential collaboration spaces [18] in which collaborators are allowed to obtain a partial and personal view of the information related to the activities shared with other collaborators.

A CWS is a conceptual 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 offered a way for starting optimal 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 [17]. The conceptual model and characterization of CWS are herewith explained.

4.3. Conceptual Model

Fig. (**2**) depicts a conceptual model of a CWS. It shows two subjects working in some related activity within a project, which is represented by a set of elements that trigger the interaction with an activity. A working sphere (WS) is represented by a circle which contains the events, persons, activities, objectives, actions, and resources that define the way through which people achieve a particular activity.

Fig. (**2**) also includes three main tasks (represented as rectangles): (i) Identifying the required information of the activity of those involved (e.g. who is involved in this activ-

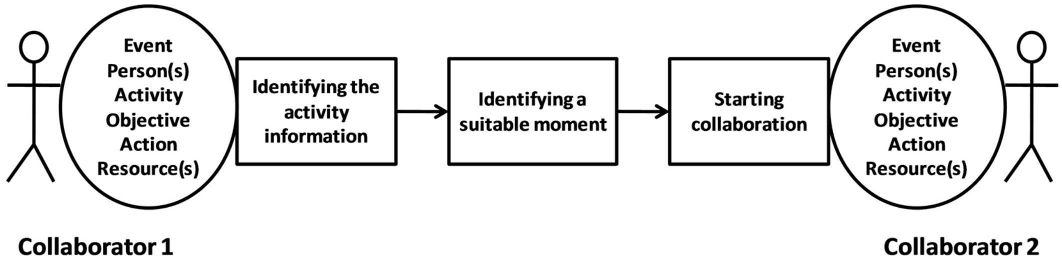

Fig. (2). Conceptual model of CWS.

Table 3. Features of CWS

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

ity?, which are my activities?, which activities are pending?) , (ii) 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 (iii) Initiating collaboration if the moment is right (e.g. who is talking to me?, what is my role?). This requires being able to monitor specific information from the collaborator´s WS so that the information on 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. These steps represent the potential for collaboration [17].

Once we have defined a conceptual model for CWS (presented in Fig. **2**), and along with the identified set of design insights, we established a set of features for a CWS system. Table **3** presents the resulting set of characteristics.

Regarding support for *remote initial interactions* (F1) which addresses design implications I1 and I6 (See Table **3**), it is necessary to include: i) a mechanism that allows interaction between remote colleagues which are focused on the work units previously assigned. This mechanism should allow to determine the need for or interest in initiating an interaction, based on whether it is (or when it is) the most appropriate time to begin an interaction.

Concerning 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, including 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, requires the inclusion of: iv) mechanisms to link the work performed individually to the work units proposed by the organization, and this 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 in it, their 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 to properly represent the different awareness levels of the potential for collaboration based on the work units attended by people 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, regarding 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 allow private communication between the collaborators involved in the work unit currently under way and which are unobtrusively available to potential collaborators.

5. PHASE 3: SUPPORT TO START COLLABORATION IN DSD

5.1. Designing a CWS Tool

A Multi-agent model. CWS require being able to monitor the activities of the collaborators and identifying specific information from the common work unit (e.g. the working sphere) so that information about the currently shared activity could become known to the group, and based on the obtained information determine whether this is the right moment for starting an interaction attempt. A technological solution is proposed in Fig. (**3**). In this figure we included the following phases: i) *Monitoring*: be alert to identify the work units in which collaborators are working. ii) *Identification*: links to the files that are being manipulated by the user with the work units that were assigned to her. iii) *Requests*: Perform requests for information of the current state of one or more users, based on the current state of the activity that occurs at a given moment. iv) *Formalization*: receive the requests from the previous phase to formalize it through the interpretation of the data that a request brings. It must be verified whether the request is valid. v) *Processing*: carry out the required queries to the organizational repository based on valid requests, and vi) *Notification*: publish the results of the petition.

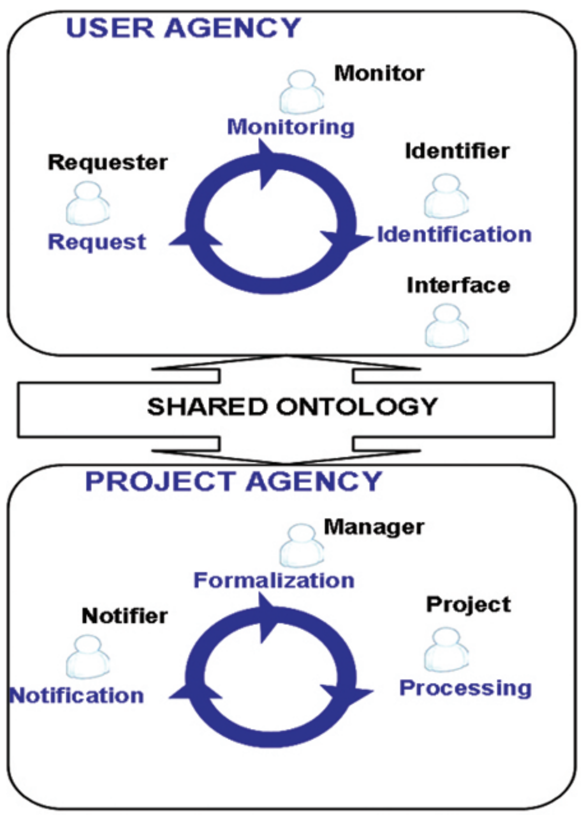

Fig. (3). Distribution of agents.

Agents are structured in two agencies: the User Agency and the Project Agency (see Fig. **3**). The User agency is responsible for providing support to requests 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 Interface Agent. On the other hand, the Project Agency aims to provide the information of the context of work based on information obtained from projects and users repositories of the organization. This agency comprises the Manager, Project and Notifier agents.

To satisfactorily address the previously mentioned phases we propose a multi-agent model (see Fig. **3**). Agents are involved in the following manner: i) Monitor Agent, it provides support in the monitoring phase. It is responsible for knowing the information (e.g. name, type, date, etc.) of the artifacts (e.g. files) manipulated by the user during 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.). ii) Identifier Agent, it provides support in the identification phase. This agent is responsible for identifying whether the file that is being manipulated by the user has to do with any activity and/or task of the organization's projects. To achieve this, the implementation of a proactive search process to identify and link the information generated during the monitoring phase is required. iii) Requester Agent, it provides support in the request phase. This agent is devoted to request information by executing a process to send data to the project instance through the use of a communication channel. iv) Manager Agent, it provides support in the formalization phase. This agent will be aware of valid projects. For this, a search process on the organization's project repositories is requested to help deciding whether the petition proceeds or not. If the request proceeds the process goes to the next phase. On the contrary, the reason for request rejection should be notified to the Interface Agent. v) Project Agent, it provides support for the processing phase. It is an agent responsible for the information of a specific project. It will request a search process for a particular project of the organization and the result will be sent to the Notifier Agent. vi) Notifier Agent, it provides support for the notification phase. It is responsible for reporting the request results to the Interface Agent (mediator between the user and agents).

In addition to Agencies, Fig. (**3**) shows a shared component ontology. This is necessary so that there is a consistent communication way between the agents of the different agencies.

5.2. A Prototype System

In order to test our model we are developing an extended IM Client with CWS support to be used by a set of DSD developers. To illustrate how the prototype works, we present the following scenario:

In a DSD organization a system designer accesses a UML file through a diagramming application. This file was sent to her as part of an interface design task. Usually, whenever the designer has a doubt about the contents of the UML file, she usually tries to contact the responsible analyst by any means of communication (e.g. telephone or an instant messaging application). In this way, the designer usually interrupts the activity of the analyst.

For the projected scenario we present a scenario diagram (see Fig. **4**), based on the INGENIAS methodology [30].

In the proposed scenario, at the time when the UML file is accessed (ApplicationEvent), a Monitor Agent observes the event (MonitoringFiles) and records it in its log (e.g. file name, document type, time, date, and state). Then an Identifier Agent verifies whether that file is related to a project (e.g. the highest level work unit). In this case, it verifies whether the file is associated to an activity or task of project "X" and updates the log by marking the file as valid to make a request for information. Such a request is detected by a

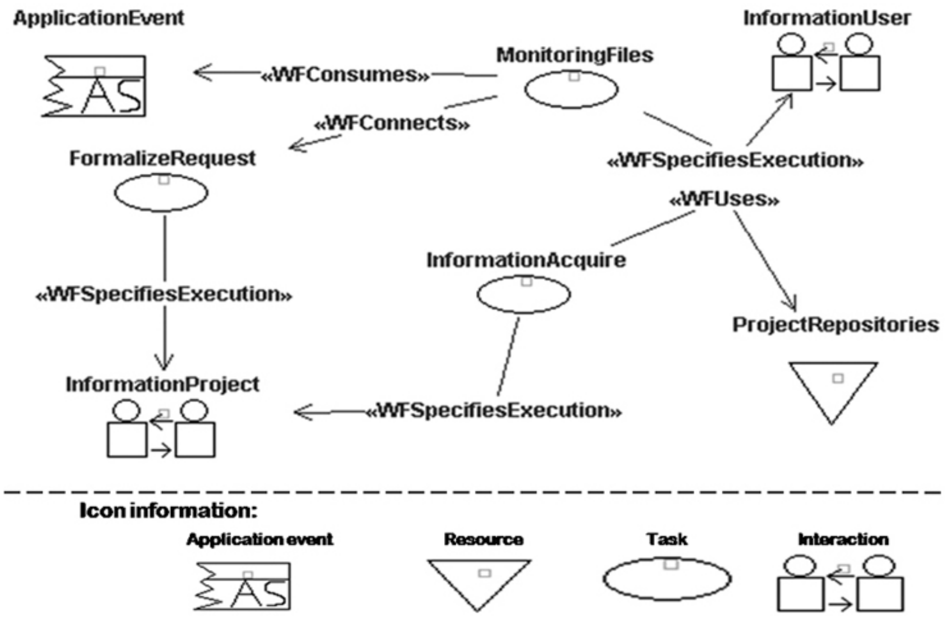

Fig. (4). Scenario diagram.

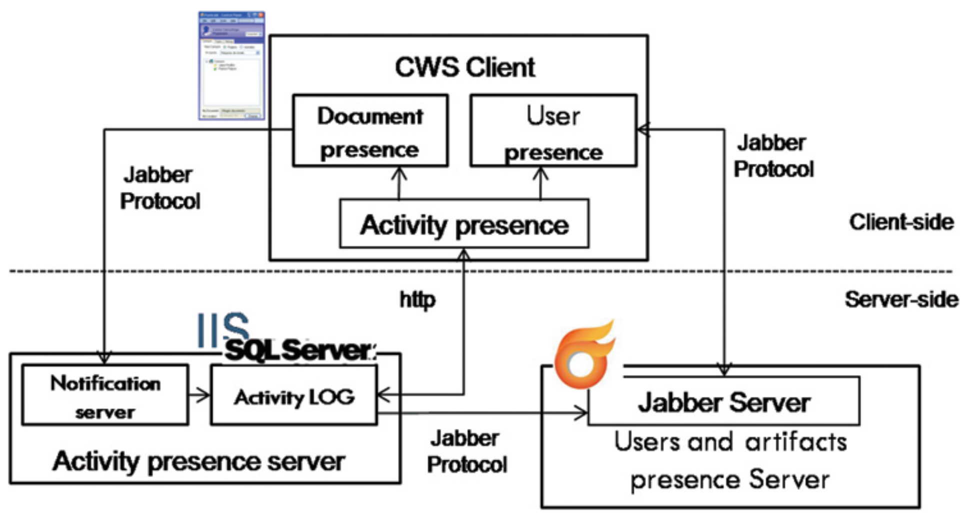

Fig. (5). Architecture of the proposed CWS-based messenger tool.

Requester Agent (InformationUser). After the interactions between agents of a User Agency, the Requester Agent makes a request to a Manager Agent, which formalizes the request validating that the Project Agent that corresponds to Project "X" is active (FormalizeRequest). In this case the project is active and the Manager Agent sends the information processing order to the Project Agent of the project in question (InformationProject). Upon receiving the order, the Project Agent searches in the repository (ProjectRepositories) for updated information on the projects and users associated to this task (e.g. developers involved, the state of developers, state of the task, etc.). On finding the information, it is packaged and sent (InformationAcquire) to an Interface Agent (InformationUser), which is responsible for updating the user interface with the received information.

In this case, the designer obtains information regarding the file in a rapid and seamless manner through the user interface of the CWS. The presented information refers to which collaborators were involved, and in which documents related to a project they were working. In turn, this information also allows the user to interpret the current state of collaborators (e.g. busy, available, not connected, etc.).

5.3. Implementation Aspects

The platform used is based on a client-server model. As shown in Fig. (**5**), CWS Client provides the interface to interact with the User presence and Activity server, and with the Users and Artifact server. This version of CWS was implemented using the C #. NET 2008 language, using the agsxmpp library [31]. This library provides the functionality required to connect to a Jabber server and retrieve the list of registered users and artifacts. Through this, the presence of users and artifacts is updated based on the activity reported from the activity log. The Activity Log is reported via a notification server, which has the information on the docu-

Table 4. Awareness Information Elements of the CWS-Based Tool and the Mechanisms Used to Obtain Them

Information Element	Type of Mechanism	Mechanisms to Obtain It
Presence	Implicit	Contact list per project or activity
Identity	Explicit	Information given from the register.
Authorship	Implicit	Information given from the individual resources of collaborator.
Teamwork	Implicit	Contact list created based on the project or activity that is being carried out.
Location	Implicit	Based on IP address from connection.
Resource view	Explicit	Based on shared resources per collaborator and project.
Gaze	Implicit	Information provided by notification history in base to activity.
Operation	Implicit	Based on presence of individual documents implied in common projects.
Purpose	Explicit	Information given by the goals of Project.
Role	Implicit	Information gotten based in the activity that collaborator is carrying it out.
To-Do List	Implicit	Activity list based on information given by assigned activities of projects.
Resource	Explicit	Linked documents based on assigned activities.
Privacy	Explicit	Based on personal shared documents.
Motivation	Explicit	Information specified in projects.
Notification	Implicit	Information created in base to success or events happened during the operations.
Expectations	Explicit	Information specified in projects.
History	Implicit	Based on event operations.
Articulation	Explicit	Group interactions based on messages and shared documents.

Table 5. GUI Components Associated to Mechanisms

GUI Element	Information Element
Contact list	Presence, role and articulation.
Identity Panel	Identity.
Resource Panel	Authorship, resource view, resources, privacy and articulation.
Contact view	Teamwork and activities.
Location label	Location.
Contact information ToolTip	Gaze y Operation.
Notification history	Operation, authorship, notification and history.
Task Panel	Purpose, motivation and expectations.

ment the user is working on. Finally, with this information, the document is validated to see whether it corresponds to a project and activity, and the result is notified through the Jabber server to all involved (connected) clients.

5.4. Designing an Instant Messaging Client as a Tool with CWS Support

Table **4** presents the information elements, the type of mechanism used (explicit or implicit) and the mechanism by which each element of information is obtained, in a similar way as in [18]. This was further used to determine the type of technological mechanisms (GUI elements) to be used to provide the information element in the CWS-based tool. Taking these into consideration, the actual mechanisms to provide each element of information were established as shown in Table **5**. These mechanisms and GUI elements were used to design and implement our prototype tool.

Table **4** shows the information elements associated to one type of mechanism (implicit or explicit) and the mechanism by which each information element is obtained. With this,

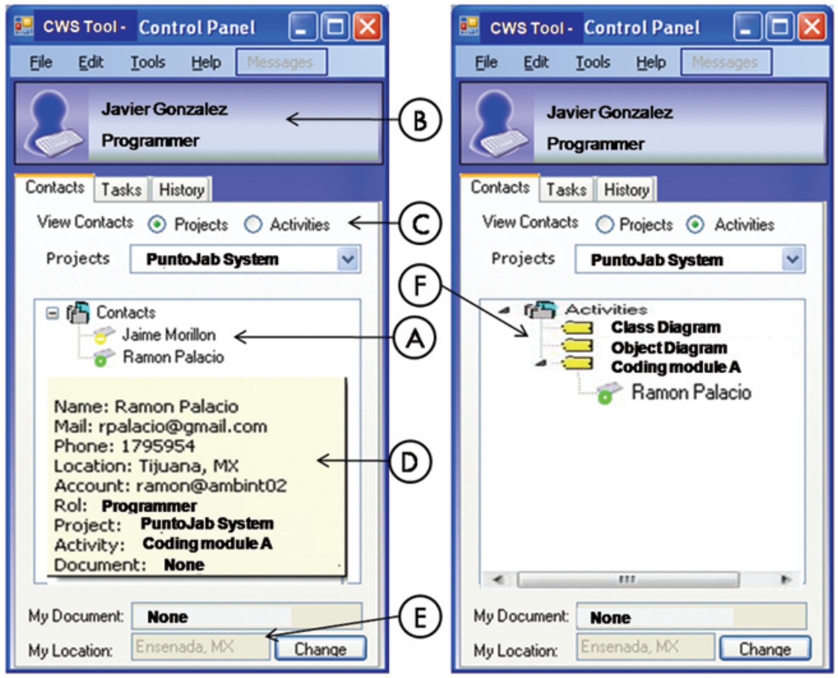

Fig. (6). The CWS IM Interface: Contact View.

the technological mechanisms (see Table **5**) that provide support to the social mechanisms (Information Element) for CWS were defined.

The mechanisms considered in this design help to improve the way in which knowledge of the project information flows in DSD processes [32]. According to the framework for analyzing information systems proposed in [33] our proposed tool could be useful to improve the transfer of tacit knowledge in DSD processes through the acquisition of information and support to for entering into collaboration between involved workers.

It is important to mention that the design aimed to create an unobtrusive tool, even though the tool is an IM client, by considering some rules for notifications, so that if a user requires contacting a collaborator, the tool tries to guarantee that the person that is being contacted is working on a common activity or project. The characterization of CWS suggests that collaborators require:

- Brief interactions where they can agree upon when is the moment to discuss about something;
- Knowledge of available potential collaborators according to the activity that is being carried out;
- Knowledge about personal activities assigned in an implicit manner where the collaborator is able to share personal resources with her collaborators and receive personal resources from them; and
- Remote communication between the collaborators in order to discuss and exchange information from different geographical locations.

5.5. CWS IM Interfaces

Our proposed solution to the lack of mechanisms that permit to decide whether the time for starting an interaction is appropriated for the person making contact, and for the person being contacted, considers the following features:

- Contacts are automatically clustered by project or activity according to the user's actual activity.
- Information about collaborators' individual activities, availability, identification, location and skills can be known at any given moment.

Information elements consider different levels of availability according to the common activity between collaborators, including:

- Personal list of assigned activities with project information.
- Unobtrusive notification history of events related to the assigned activities.
- Communication only with collaborators involved in a common activity in an unobtrusive manner.
- Information related to projects and collaborators is supplied in an implicit manner.

Figs. (**6**) and (**7**) present a CWS interface that includes these features. The CWS tool furnishes a special form to provide awareness of whether collaborators are connected, available and on which role (skill) they are assigned by means of a contact list. It is clustered based on the selected view. This interface also shows personal information about collaborators in order to identify what role a specific collaborator is taking in a given moment, what document she is working on, her location, etc.

To illustrate the functionality we present and describe the GUI components of the CWS Client (see Figs. **6** and **7**) that provides the selected CWS information elements and mechanisms, including:

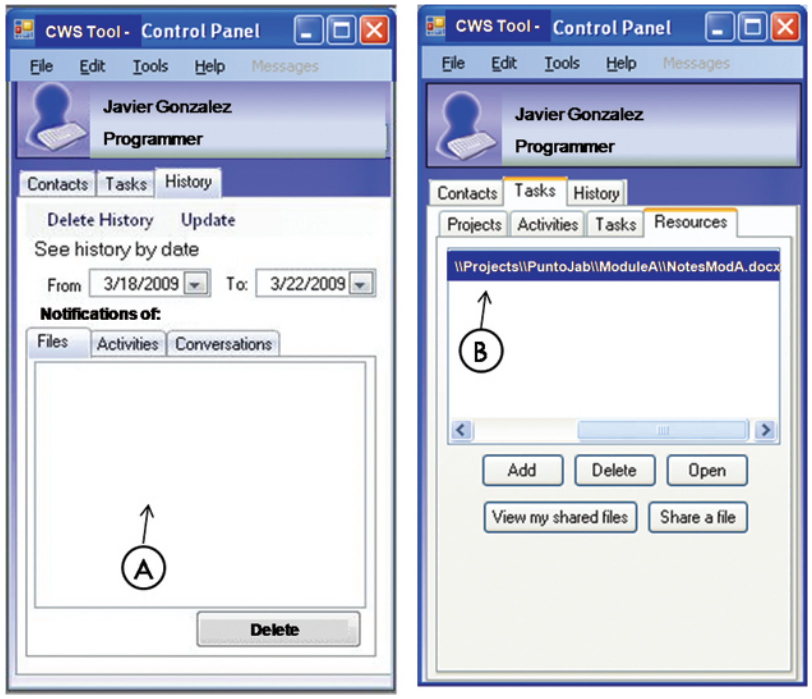

Fig. (7). The CWS IM Interface: Notification History and Task Panel.

Contact list (6A and 6F) presents a list of potential collaborators considering two possible perspectives according to the option selected in the contact view (6C). It can be shown using a Project or an Activity view. From this component it is possible to start a chat session with a group of collaborators or with one collaborator.

The Identity panel (6B) is used to present the actual collaborator's information such as name, role, and status.

The Contact information ToolTip (6D) is accessed through a right click over the desired contact. This component shows actual information about potential collaborators (e.g. actual location, active document according to the activity being carried out, actual role, etc.).

The Location label (6E) provides information about geographical location. This information is supplied based on the IP address assigned.

Notification history (7A) provides 3 categories of events: files (e.g. notification of shared files), chat (e.g. when there is a conversation concerning the group's activity) and activities (e.g. change a deadline activity).

Task panel (7B) presents 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.)

Furthermore, the CWS tool allows to start collaboration with target users in a more informed way by using the "selective availability" criterion, that is, it allows considering that "a user is available to collaborators whose activity is related to the work unit she is currently dealing with and not available to other collaborators". To achieve this, the status of a user is represented using a color code: if the status of the user is presented in green it means that the users are working in the same project and in the same activity; if the status is represented in yellow it means that the users are working on the same project, but in different activity; and if the status is presented in red it means that they are working in different projects. Thus, the CWS tool may allow creating an opportunity for interaction between users who are working in a different activity when the time becomes adequate for both users to do so. For example, if the status of a user is represented in red (working on another project), the CWS tool allows to "mark" this user so that when her status changes to green, the tool notifies that there exists a timely and convenient opportunity for both users to interact, as they are now working on the same project and on the same activity.

6. PHASE 4: EVALUATION

6.1. Experimental Design and Procedure

We conducted an experiment to explore the potential and limitations of the tool regarding the provided support for starting collaboration at timely adequate moments, both for the one making contact and the one being contacted.

To guide our research we established the following working hypotheses:

H1: Participants will perceive the use of the CWS-based tool as more useful than the traditional instant messaging application to perform the assigned tasks (perceived Usefulness).

H2: Participants will perceive the use of the CWS-based tool as easier to use than the traditional instant messaging application to perform the assigned tasks (perceived Ease of use).

The first hypothesis aims at establishing that a tool that considers the features of DSD workers and their activities could provide better support to perform the specified activities while working in a distributed manner when compared to traditional existing IM tools. The second hypothesis is established from our belief that the proposed tool could be easier to use to perform the specified activities than traditional and ubiquitous IM applications.

6.2. Participants

Participants were 16 workers from different companies in three different cities of the State of Sonora, Mexico. All of them (16) were participating or have participated in DSD projects. Their average age was 30.25 years old, and have at least three (3) years of DSD experience (7 of them) or less than three (3) years of DSD experience (9 of them). All of them (16) have concluded their bachelor (BSc) degree in computer science or related and one (1) graduated with a master (MSc) degree in computer science.

6.3. Setting

We performed a comparative study based on interviews and the presentation of current and projected scenarios of use. Condition 1, considered the use of a traditional Instant Messenger (IM) tool (current scenario) to perform three specific tasks in a DSD setting, and Condition 2, considered the use of the proposed CWS-based Messenger tool (projected scenario) to perform these same tasks.

The study was performed using a Within Subject paradigm (all subjects participated in both conditions of the study). The group was randomly divided into two groups as they showed up in the room where the study was performed. The first 8 participants started performing the tasks in Condition 1 (traditional IM tool), while the remaining 8 were assigned to the Condition 2 (CWS-based Messenger tool). After completing the tasks under Condition 1, the first group switched to Condition 2, conversely, group 2 switched to Condition 1 after completing the tasks in Condition 2.

6.4 Procedure

Three activities were performed by participants in approximately 45 minutes:

Activity 1: On-entry survey. Initially, participants were interviewed and asked about their demographic data (Name, Age, DSD and traditional software development experience, etc.)

Activity 2: Presentation of current and projected Scenarios of use. Later, a series of three scenarios for each condition were presented to participants. In these scenarios, they had to perform a task using the tool assigned to the condition, either the traditional IM tool, or the CWS-based messenger tool. The tasks consisted in i) search for a collaborating partner and determine his/her work status, ii) query the pending assigned activities, and iii) gather information on the activity being performed by the collaborating colleague (e.g. progress status, assigned resources, etc.).

Activity 3: On-exit survey. Finally, participants were interviewed and asked to respond standard Technology Acceptance Model (TAM) [34] questionnaires in order to obtain their perception on the usefulness and ease of use of the proposed CWS-based messenger tool.

The obtained data was processed using descriptive statistics (see Tables 6 and 7), in order to quantify the perception levels provided by the participants regarding usefulness and ease of use.

6.5. Results and Discussion

As stated earlier, we inquired participants about their general perception of usefulness and ease of use by using standard TAM questionnaires [34]. Perceived usefulness, and ease-of-use, which are important factors influencing user acceptance and usage behavior of information technologies were measured. Thus, the questionnaire consisted of two sections (with six questions each) as presented in Tables **6** and **7**. All questionnaire items were measured on a 7-pointLikert scale, ranging from 1 (“completely disagree”) to 7 (“completely agree”)

Tables **6** and **7** present the averages of the questionnaire items as evaluated by DSD workers. We calculated the averages for each factor which enabled us to make the following analysis. A brief presentation and discussion of the main results follows.

Table 6. TAM Results Regarding the Perceived Usefulness of the System

Usefulness								
Question	**IM**				**CWS**			
	Sc1	**Sc2**	**Sc3**	***Avg***	**Sc1**	**Sc2**	**Sc3**	***Avg***
Q1: Using the system would enable me to accomplish tasks more quickly	4	4.63	4.63	*4.42*	5.81	6.31	6.44	*6.19*
Q2: Using the system would improve my performance.	3.44	3.69	4.13	*3.75*	5.31	5.75	5.81	*5.62*
Q3: Using the system would make it easier to do my work chores.	3.81	4.13	4.19	*4.04*	5.69	5.63	6.38	*5.90*
Q4: Using the system would enhance my effectiveness at work.	3.44	3.56	3.81	*3.60*	5.25	5.63	5.88	*5.59*
Q5: Using the system display would increase my productivity.	3.19	3.50	3.81	*3.50*	4.81	5.44	5.88	*5.38*
Q6: I would find the system useful in my work.	4.81	5.13	4.38	*4.77*	6.13	6.38	6.69	*6.40*
	Condition Average			*4.01*	Condition Average			*5.85*

Table 7. TAM Results Regarding Perceived Ease of Use of the System

Ease of Use								
Question	IM				CWS			
	Sc1	Sc2	Sc3	*Avg*	Sc1	Sc2	Sc3	*Avg*
Q7: I find it easy to get the system to do what I want it to do.	4.13	4.63	4.44	*4.40*	5	5.25	5.75	*5.33*
Q8: My interaction with the system is clear and understandable.	5.44	5.13	5.56	*5.38*	5.88	6	6.25	*6.04*
Q9: Learning to operate them system would be easy for me.	5.75	5.69	6	*5.81*	5.75	6	6.31	*6.02*
Q10: It would be easy for me to become skillful at using the system.	6.25	6.06	6.31	*6.21*	6	6.19	6.25	*6.15*
Q11: I would find the system easy to use.	5.19	5.44	5.31	*5.31*	6.38	6.44	6.63	*6.48*
Q12: I would find the system to be flexible to interact with.	5.69	5.50	5.63	*5.61*	5.88	6.06	6.25	*6.06*
	Condition Average			*5.45*	Condition Average			*6.01*

The overall functionality of the CWS-based messenger tool was perceived as ease to use (6.01) and useful (5.85) by most participants. Regarding the comparison with the traditional IM tool, the CWS-based messenger tool was consistently perceived as more useful (mean average = 5.85) than the traditional IM tool (mean average = 4.01). This difference is statistically significant ($t_{.01}$=-1.915) and H1 is accepted.

Further, concerning ease of use, the average slightly favors the CWS-based messenger tool (mean average = 6.01) rather than the traditional IM tool (mean average = 5.45). This difference, however, is not significant ($t_{.01}$=7.093) and H2 is not accepted.

It is worth noting that although the traditional IM system was considered as slightly easier to become skillful at its use (Q10, average 6.21) than the CWS-based messenger tool (average 6.15), given that $t_{.01}$=-0.404 there is no significant difference. Finally, the CWS-based messenger tool was considered as easier to get it to do what the user wanted to do (Q7, average 5.33) than with the traditional IM system (average 4.44); given that $t_{.01}$= 3.61, there is a significant difference. A possible explanation for this is that the CWS-based messenger tool actually provides information required for the assigned tasks, while traditional IM systems lack this kind of information.

7. CONCLUSIONS AND FUTURE WORK

Software development organizations are facing a paradigm shift towards Distributed Software Development (DSD), i.e. the distribution of development process and teams. This shift introduces benefits and challenges to which organizations have to adapt. 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 labor rates. Challenges include limitations due to cultural differences, coordination difficulties due to different time zones, inadequate and problematic knowledge management and communication due to language and cultural differences and to geographical distribution, and lack of trust in inter-team relationships.

In this work, we 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. To achieve this, we introduced and defined the concept of Collaborative Working Spheres (CWS).

Based on a literature study, we identified a set of features of DSD activities, which were later used to establish an ensemble of design issues for a tool that aims at providing support to timely and adequately start interaction in a DSD environment. Informed by these results, we developed the concept of Collaborative Working Spheres (CWS), as well as a prototype tool based on this concept.

The CWS were mainly proposed to emphasize the need to exchange information between the members of a work group who are in different geographical locations in a lightweight and simple manner. Further, we included into CWS support to allow starting an interaction in a selective way according to a criterion that we refer to as "Selective availability", that is to consider that "I am available only to collaborators who are related to the work unit I am dealing with now and 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.

According to our evaluation results, a CWS-based tool can improve the way in which developers start collaboration compared to how this is done using current instant messaging tools. The main features to achieve this include that a CWS: i) distributes explicit organizational knowledge (e.g. Project status information) in real-time, ii) provides this explicit information through specialized mechanisms to specific project- and activity-related team members, iii) offers an individual perspective to the DSD group (e.g. availability information of each individual related to the project according to the assigned work and the activity currently under execution), as well as a group perspective to an specific team member (e.g. notifies whenever a developer is available for some project-related members, and unavailable for others).

Future work includes evaluating the proposed tool in actual use. We plan to start this evaluation by deploying our

CWS-based tool for a test trial of at least 3 weeks in a software factory distributed in three different cities in the state of Sonora, Mexico.

ACKNOWLEDGEMENT

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), and Competitiv-IT Strategic Solutions (Navojoa, Sonora, México) and the 16 participants for their valuable support and participation that made the evaluation possible. Also, authors 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 is partially supported 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.

REFERENCES

[1] R. Prikladnicki, J.L.N. Audy and J.R. Evaristo, "Distributed software development: toward an understanding of the relationship between project team, users and customers," *Proc. 5th Int. Conf. Enterprise Information Systems (ICEIS 03)*, pp. 417-423, 2003.

[2] D. Damian and D. Zowghi, "The impact of stakeholders'? Geographical distribution on managing requirements in a multi-site organization," *Proc. The 10th Anniversary IEEE Joint international Conference on Requirements Engineering*, pp. 319-330, 2002.

[3] L. Layman, L. Williams, D. Damian and H. Bures, "Essential communication practices for Extreme Programming in a global software development team," *Inf. Softw. Technol.*, vol. 48, no. 9, pp. 781-794, 2006.

[4] E. Ó. Conchúir, P. J. Ågerfalk, H. H. Olsson and B. Fitzgerald, "Global software development: where are the benefits?," *Commun. ACM*, vol. 58, no. 2, pp. 127-131, 2009.

[5] C. Ebert and P.D. Neve, "Surviving Global Software Development," *IEEE Softw.*, vol. 18, pp. 62-69, 2001.

[6] I. Richardson, V. D. Casey, D. Zage and W. Zage, *Global Software Development – The challenges* University of Limerick, Ball State University, 2005.

[7] D. Damian and D. Moitra, "Guest editors' Introduction: Global Software Development: How far have we come?," *IEEE Softw.*, vol. 23, no. 5, pp. 17-19, 2006.

[8] W. Lloyd, M. Rosson and J. Arthur, "Effectiveness of elicitation techniques in distributed requirements engineering," *Proc. The 10th anniversary IEEE Joint International Conference on Requirements Engineering, RE'02*, pp. 311-318, 2002.

[9] E. Carmel and R. Agarwal, "Tactical approaches for alleviating distance in global software development," *IEEE Softw.*, vol. 18, no. 2, pp. 22-29, 2001.

[10] J. Herbsleb and D. Moitra, "Guest Eds.' Introduction: Global software Development," *IEEE Softw.*, vol. 18, no. 2, pp. 16-20, 2001.

[11] S. Whittaker and H. Schwarz, "Meetings of the board: The impact of scheduling medium on long term group coordination in software development," *Comput. Supported Coop. Work*, vol. 8, pp. 175-205, 1999.

[12] E. Conchúir, H. Holmstrom, P. Agerfalk and B.Fitzgerald, "Exploring the assumed benefits of global software development," *Proc. The IEEE International Conference on Global Software Engineering (ICGSE'06)*, pp. 159-168, 2006.

[13] D. Damián, "Stakeholders in global requirements engineering: Lessons learned from practice," *IEEE Softw.*, vol. 24, pp. 21-27, 2007.

[14] U. Bellur, "An academic perspective on globalization in the software industry," *Proc. The 30th annual international computer software and applications conference (COMPSAC'06)*, pp. 52-54, 2006.

[15] M. Babar, J. Verner and P. Nguyen, "Establishing and maintaining trust in software outsourcing relationships: An empirical investigation," *Syst. Softw.*, vol. 80, no. 9, pp. 1438-1449, 2007.

[16] R. E. Kraut and L.A. Streeter, "Coordination in software development," *Commun. ACM*, vol. 38, no. 3, pp. 69-81, 1995.

[17] A.L. Morán, J. Favela, R. Romero, H. Natsu, C. Perez, O. Robles and A. Martinez, "Potential and actual collaboration support for distributed Pair-Programming," *Comput. y Sistemas* vol. 11, no. 3, pp. 211-229, 2008.

[18] A.L. Morán, J. Favela, A. Martinez and D. Decouchant, "On the design of potential collaboration spaces," *Int. J. Comp. Appl. Technol. (IJCAT)*, vol. 19, no. 3/4, pp. 261-271, 2004.

[19] M. Czerwinski, E. Horvitz and S. Wilhite, "A diary study of task switching and interruptions," *Proc. the SIGCHI Conference on Human Factors in Computing Systems*, pp. 175-182, 2004.

[20] A. Stinchcombe and C. Heimer, *Organization Theory and Project Management: Administering Uncertainty in* Norwegian *Offshore Oil*, A Scandinavian University Press Publication, 1986.

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

[22] R. Kraut, R. Fish, B. Root and B. Chalfonte, "Informal communication in organizations: Form, function and technology," *Proc. S. Oskamp & S. Spacapan, (Eds.), People's reactions to technology in factories, offices and aerospace, The Claremont Symposium on Applied Social Psychology, Sage Publications*, pp. 145-199, 1990.

[23] S. Fussell, S. Kiesler, L. D. Setlock and P. Scupelli, "Effects of instant messaging on the management of multiple project trajectories," *Proc. The SIGCHI Conference on Human Factors in Computing Systems*, pp. 191-198, 2004.

[24] V. González, L. Galicia and J. Favela, "Supporting the Planning and Organization of Multiple Activities in the Workplace," *Proc. INTERACT 2007*, pp. 235-238, 2007.

[25] R . Morteo, V. González, J. Favela and G. Mark, "Sphere Juggler: fast context retrieval in support of working spheres," *Proc. ENC 2004*, IEEE Press, pp. 361-367, 2004.

[26] J. Camacho, J. Favela and V. González, "Supporting the Management of Multiple Activities in Mobile Collaborative Working Environments," *Proc. CRIWG 2006*, pp. 381-388, 2006.

[27] J. Carroll, *Making Use*, MIT Press, 2000.

[28] I. McChesney, "Effective coordination in the software process - historical perspectives and future directions," *Softw. Quality Control*, vol. 6, no. 3, pp. 235-246, 1997.

[29] V. Gonzalez and G. Mark, "Constant, constant, multi-tasking craziness: managing multiple working spheres," *Proc. The SIGCHI Conference on Human Factors in Computing Systems*, ACM Press, pp. 2004, 2004.

[30] J. Pavón and J.J. Gómez-Sanz, "Agent oriented software engineering with INGENIAS," *Proc. The Multi-Agent Systems and Applications III: CEEMAS 2003 (LNAI)*, pp. 394-403, 2003.

[31] J. Clients, "The power of presence," 2008, http://www.jabber.com/CE/JabberClients.

[32] O. M. Rodríguez, A. I. Martínez, J. Favela, A. Vizcaíno and M. Piattini, "Understanding and Supporting Knowledge Flows in a Community of Software Developers," 3198, L. N. i. C. Science, ed., Springer, pp. 52-66, 2004.

[33] O. M. Rodríguez, A. I. Martínez-García, A. Vizcaíno, J. Favela and M. Piattini, "A framework to analyze information systems as knowledge flow facilitators," *Inf. Softw. Technol.*, vol. 50, pp. 481-498, 2008.

[34] F. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology," *MIS Quart.*, vol. 13, pp. 319-340, 1989.

Received: July 01, 2009 Revised: July 30, 2009 Accepted: May 03, 2010