2010 International Conference on Global Software Engineering

Virtual Open Conversation Spaces: Towards Improved Awareness in a GSE Setting

Kevin Dullemond
Delft University of Technology
IHomer
K.Dullemond@TUDelft.nl

Ben van Gameren
Delft University of Technology
IHomer
B.J.A.vanGameren@TUDelft.nl

Rini van Solingen
Delft University
of Technology
D.M.vanSolingen@TUDelft.nl

***Abstract*—Conversations between colleagues in collaborative software engineering are important for coordinating work, sharing knowledge and creating knowledge. Overhearing conversations of others is useful as well since this: (i) provides access to the information discussed in the conversations, (ii) offers the possibility of joining the conversations and (iii) provides insight in the communication structure of the project team. When working in a GSE setting, specialized tooling is required to be able to have conversations and to know what conversations others are having. In this paper we discuss how conversations support collaborative software engineering and how this can be supported by technology in a GSE environment. To do this, we introduce Communico: a virtual open conversation space which features: (i) initiating conversations by selecting people to converse with, (ii) sharing information regarding the involvement of project members in these conversations and (iii) having access to persistent conversations with an explicit status indicating whether they are ongoing.**

I. Introduction

Global Software Engineering (GSE) is becoming increasingly interesting due to the globalization of business [1], [2], [3], [4], [5], [6]. In GSE the software development process is distributed between several geographically dispersed locations [7], [3], [8]. Advantages of GSE include: market-proximity [9], [10], [3], reducing time-to- market by working around the clock [1], [11], [12], [3], flexibility with respect to business opportunities [1], [11], reducing costs by delegating work to countries with low labor cost [13], [3] and being able to fully utilize available resources [2], [9], [3]. Besides being beneficial, GSE introduces a number of challenges in relation to communication, coordination and control of the development process [13]. Examples are: lack of informal communication [1], [11], [2], [14] reduced hours of collaboration [15], [16], [17], [6], communication delay [14], [18], [4], [7] and loss of cohesion [1], [19], [4].

In collaborative work it is essential to have knowledge about the context in which you are working to properly cooperate with others [20], [21]. With information about the context we mean information about the other members in the project team, their activities, information about the state of the project and so on. This information is essential because this knowledge is necessary for coordinating actions, managing coupling, discussing tasks, anticipating others' actions, and finding help [20], [21], [22]. The complexity and interdependency of software systems (e.g., [23]) suggest that this is also the case for collaborative software development. In scientific literature the term 'awareness' is often used to denote this [20], [24]. Dourish et al. use the following definition: *"An understanding of the activities of others which provides a context for your own activity"* [24].

Awareness information is distributed among the members of the project team as follows: Actors display information on a shared medium while other actors monitor the medium and acquire information from it [20]. In this process, both monitoring and displaying are activities that are not necessarily conducted with the full attention of the actor. Often when expressing this varying degree of attention dichotomies are used, such as: explicit versus implicit, deliberate versus automatic, conscious versus subconscious, focused versus unfocused and active versus passive. However, as emphasized by Schmidt [20], the distinction between these notions is not categorical but merely one of degrees.

When team members are not sharing a physical work environment they are outside of sensory range of each other. Therefore information exchange between them becomes infeasible without some kind of technological support. This can be dealt with by providing other ways of acquiring the required information, like using the telephone or email to ask a question. However, in general, such solutions are inferior to the way contextual information is shared in a traditional co-located setting, in the sense that in comparison it (i) takes more effort because the communication is more intentional [25], (ii) is more obtrusive [26], (iii) happens less frequently [19], [27], [28] and (iv) contains less information [25], [11], [29]. As such we can conclude that sharing awareness information is more difficult in a distributed setting. Due to the nature of the challenges associated with GSE, it is plausible to assume these challenges originate from having insufficient access to information regarding the work context: a lack of awareness. The research presented in this paper continues upon this insight that a lack of awareness is the origin of the challenges faced in GSE. It

978-0-7695-4122-8/10 $26.00 © 2010 IEEE
DOI 10.1109/ICGSE.2010.36

is part of the ASPIC[1] research program. The goal of this program is to develop solutions to the problems caused by the difficulties with acquiring and maintaining awareness in GSE. In this research the focus will lie on making the sharing of information a more passive activity because (i) this will likely lower the effort to share awareness information, (ii) cause this information to be more recent and (iii) improve the quality of the information as well.

In this paper we will focus on how knowledge about the conversations between the members of a development team can improve collaboration in GSE. The main research question will be: *"How can awareness about conversations within a development team support collaborative software engineering and how can this be facilitated by technology in a GSE setting?"*

This paper is structured as follows: In section two we give a formal definition of a conversation within the context of GSE. In section three we discuss the advantages of conversations, the advantages of overhearing conversations and introduce the concept of an open conversation space. Following this in section four we discuss what existing supporting technologies support an open conversation space in a GSE setting. Subsequently in section five Communico, a virtual open conversation space we developed in this research, is discussed. This is done by comparing it with existing tooling, discussing the user interface and briefly discussing the technical implementation. In section six an evaluation of the use of Communico in a practical setting is presented. Finally we discuss limitations and future work in section seven and conclude upon our research in section eight.

II. Conversations

There are many definitions of the word conversation. The Oxford English dictionary for example defines it as: *"An informal spoken exchange of news and ideas between two or more people"* [30]. The Merriam-Webster's collegiate dictionary uses the following definition: *"oral exchange of sentiments, observations, opinions or ideas"* [31]. Finally the Cambridge advanced learner's dictionary defines conversation as: *"(a) talk between two or more people in which thoughts, feelings and ideas are expressed, questions are asked and answered, or news and information are exchanged"* [32]. These definitions seem to agree on the fact that, conversations:

1) Use verbal communication
2) Are an exchange of information of various origin between two or more people

We find the confinement to verbal communication too strict however. This confinement presumably originated from how people have held conversations for a very long time, namely by being at the same place at the same time and talking to each other. We feel, however, that the confinement to verbal communication is meant to convey something else altogether. Firstly, an exchange of information between people is only a conversation when the communication can be considered synchronous[2]. Because the communication should be synchronous for an information exchange to be a conversation, a form of communication that is comprehensible by humans in real time should be used. To help distinguish it from *conversations*, we propose to use the term *correspondence* for the exchange of information between people using asynchronous communication[3]. Secondly, when people are part of a conversation they are directing their communication at one or more specific people. So, broadcasting information, like for instance an announcer working at a train station, at a football stadium or at the market place, is not a conversation. This can however be a way to initiate a conversation. Another way to initiate a conversation is by directly starting to communicate with one or more specific people. Summarizing, we define a conversation in the context of GSE as: *"An exchange of information between one or more people where those participating use synchronous communication directed at the other participants"*.

III. Open Conversation Space

In collaborative work conversations have various uses. For one, conversations help people to integrate and coordinate their work, by discussing their past, current and future activities [34], [22], [35]. An example of this is when a developer (d_1) tells a colleague (d_2) he is currently working on a certain work item and what work item he is planning to do next. Because developer d_1 informs developer d_2 of his current and planned future activities, developer d_2 can both adapt his own current and planned activities as well as influence those of developer d_1. Secondly conversations are a powerful tool to share knowledge about the actual work [36], [37]. An example of this is when a project member asks a colleague to explain some technical aspect of the work he is doing. Finally, conversations are ways of creating new knowledge [38], [36], [37]. Examples of this are having a discussion with someone to come up with a solution to some issue in the project and having a brainstorm session to identify the requirements of a system to be built.

Besides taking part in conversations, also conversations of others are useful. For one, overhearing the conversations of others is a source of information since this provides access to the information which is discussed and/or concluded in the conversation [22]. This refers to both technical project

[1]Awareness-based Support Project for Interpersonal Collaboration in Software Engineering, http://aspic.ewi.tudelft.nl

[2]Communication is *'synchronous'* when the sending and receipt of messages between actors communicating can be regarded as instantaneous [33]

[3]Note that some tools, for example Gmail, use the term conversation for what we call correspondence

information and information regarding the current, past and future activities of other project members. Secondly, having insight in the ongoing conversations provides the opportunity to join a conversation [39]. Having joined a conversation, being part of that conversation provides the same uses as discussed earlier for conversations in general, namely: integrating collaborative activities, knowledge sharing, and the creation of new knowledge. Finally, by having access to the communication frequencies between colleagues, the insight into the communication structure of the project team is increased [40], [41], [42]. This information is important to be able to address communication issues [43], [44], [45]. An example when this information can be useful is the following: Say you need to ask someone a question but you cannot reach him. If you know who that person frequently communicates with, you could attempt to reach this person instead and see if he can assist you with contacting the right person or with resolving the issue itself.

Having discussed the uses of conversations in general and the uses of having access to the conversations of others, we can conclude both being able to have and overhear conversations is important in collaborative software engineering. To refer to an environment in which this is possible, we define an open conversation space: *A space in which (i) conversations are possible between the actors in that space and (ii) these conversations are visible to other actors in that space*. An example is a normal office setting. In such a setting members of the project team converse by means of spoken natural language (among other means) and such conversations are audible by other people in the setting. Such an instantiation of an open conversation space also causes some disadvantages. Examples of these are: distractions, interruptions and a lack of privacy. However, working in a space in which the members of the project team are frequently able to both see and hear each other is so advantageous that it is one of the main reasons for working in a co-located setting [42]. Therefore we propose the creation of a *'virtual'* open conversation space: *"An open conversation space which is applicable in a distributed setting"*.

IV. Existing supporting technologies

Arising from the need to communicate when people are not physically together, technologies were developed which support telecommunication. The most used example of these is the telephone [46]. With the widespread adoption of the internet many more solutions which support remote communication were developed. Examples are: Instant Messaging, Internet Relay Chat, E-mail, Forums and more recently Twitter [47] and Google Wave [48]. Even though these solutions support remote communication, not all of them are suitable to have conversations. To support conversations the tools facilitating the communication should support both synchronous communication and directing the communication at specific people. So, Twitter is not suitable to have a conversation since the communication it supports is asynchronous, and the communication is mostly directed at the world in general. For the same reasons forums are not suitable as well. E-mail, finally is also not suitable because the communication it supports is also asynchronous. The other solutions we mentioned above (telephone, Instant Messaging, IRC and Google Wave) do support directed synchronous communication and thus having conversations.

Having discussed the requirements of a tool to be able to support conversations, we turn our attention to tool support for virtual open conversation spaces. To be classified as a virtual open conversation space a tool should (i) support having conversations and (ii) make it possible for people using the tool to also *'see'* the ongoing conversations they are not part of. Taking note of these requirements, it is straightforward to see telephone and Instant Messaging are not virtual open conversation spaces because, using these tools, it is impossible to perceive what conversations are going on, unless you are part of the conversation. Google Wave also cannot be classified as a virtual open conversation space because all the participants need to be explicitly selected by the current participants in the Wave. People who are not part of the Wave do not know of its existence unless they are invited[4]. This leaves Internet Relay Chat since it fulfills both requirements necessary to be classified as a virtual open conversation space.

Internet Relay Chat is however not the most suitable implementation of a virtual open conversation space. For one, it does not make the concept of a conversation explicit. All messages are shown in sequential order irrespective of what conversation they belong to. This limits the creation of a structured and logical layout for group discussions [49]. In a co-located setting, conversations are also sequential in nature, but other indicators help to more easily identify conversations as such. Examples of indicators of conversations are the placement of individuals in the room (people standing in close proximity talking are likely to be having a conversation) and the people at whom people are looking. When only the sequential ordering of conversations is known, all that is left to identify conversations is semantics. Because of this it requires more effort to be aware what conversations are going on. Therefore tooling, supporting open conversation spaces in a distributed setting, should provide for easier identification of ongoing conversations. One method of accomplishing this is making conversations explicit entities within the virtual open conversation space. Several existing specialized tools which provide a virtual open conversation space, do make the conversation explicit.

[4]By applying a work-around, Google Wave can be made open in a restricted way. This is done by adding a specific bot to a wave which makes the respective wave searchable and open to all users of Google Wave (the entire world). Google Wave might evolve towards a full virtual open conversation space as it is currently under development.

Examples are: GroupBanter [50], Babble [37], Loops [51], SWIM [49], ReachOut [52] and Threaded chat [53]. All of these however initiate conversations by *creating a topic*. Comparing this to the co-located setting, an example of this is asking whether anyone knows something about a certain topic by asking it out loud in a group. Most conversations are however initiated in a different fashion. In *'real-life'* conversations are mainly started by starting to speak to a specific person. Birnholtz et al. [54] report doing something like this with respect to their tool *'OpenMessenger'*. We could however not retrieve how they made conversations explicit as their work focuses on the gradual initiation of interactions.

V. Communico

To validate the theory devised in the research for this paper we developed a tool called *Communico*[5]. Communico is a virtual open conversation space which utilizes the communication functionalities provided by the Instant Messaging program Office Communications Server[6]. First we will discuss how it differs from existing tools which provide an open conversation space in a distributed setting. Subsequently we will discuss the tool itself by illustrating its use and we will conclude by briefly discussing the technical implementation.

A. Difference with existing tooling

Communico differs from the existing tools which provide open conversation spaces in a number of ways. Firstly, because it uses an Instant Messaging program for the communication functionalities, it allows for participant based conversation initiation: conversations can be started by selecting one or more participants and no a priori subject or topic is required. We feel initiating conversations in this fashion is quite *'natural'*, as it happens frequently in a co-located setting. Therefore it seems beneficial for a tool supporting a virtual open conversation space to offer this kind of functionality.

Secondly, we provide information regarding the involvement of the various actors with respect to the conversations. A number of sources [53], [55] report the importance of this although they mostly focus on the importance of knowing who are currently viewing the ongoing conversations and do not specify various levels of involvement. Erickson et al. [37] do consider different levels of involvement in a conversation and use a visual representation they call a *'social proxy'* to visualize this in their tool Babble. They depict people as marbles in and around a circle which represents a certain conversation. Marbles depicting active people are depicted in the center of the circle and slowly drift to the border during inactivity. Activity is defined by how recently someone has directly contributed to the conversation or interacted with the conversation window using the mouse. People logged on to Babble but active in a different conversation are depicted as marbles outside of the circle.

In Communico we propose to visualize involvement differently. In a co-located setting the degree of involvement in a conversation of someone depends on how aware he is of the conversation and whether or not he participates in the conversation. Therefore we suggest that a tool supporting a virtual open conversation space should indicate per conversation (i) who is aware of it and (ii) who participates in it. To do this, we define three levels of involvement and depict these in figure 1:

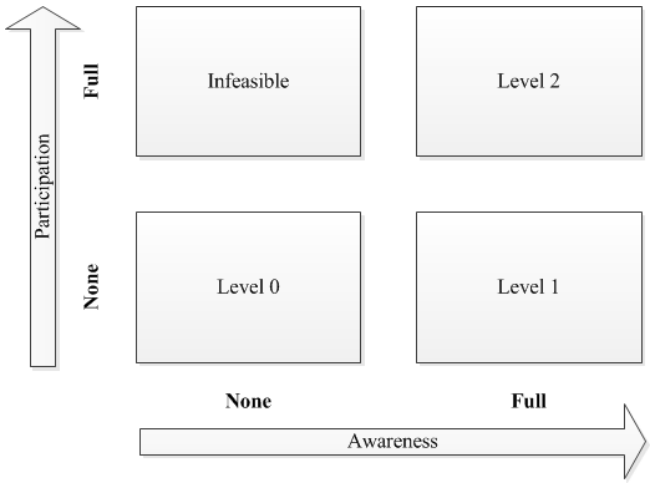

Figure 1. Initial model of conversation involvement

Level 0. The actor has no awareness of the conversation and does not participate.

Level 1. The actor is aware of the conversation and knows things like: the content, the participants, the running time and/or the topic.

Level 2. The actor participates in the conversation: the entire history of the conversation is available and the actor is able to contribute.

Using the social proxy used in Babble, it is not possible to categorize the degree of involvement of someone in a conversation using these levels. When the marble representing a specific individual is not shown in the social proxy of a certain conversation, this does not mean this individual does not know of its existence: he is either involved in the conversation on level 0 or level 1. Likewise, when a marble is depicted in the center of a social proxy circle, it is impossible to determine whether the person represented by that marble is actually a participant in the conversation: it is not clear whether he is involved in the conversation on level 1 or level 2.

In Communico it is possible to differentiate between the three levels. When a user is not logged in he does not

[5]Communico was chosen as a name since it means 'to share' in Latin and the word closely resembles the verb *'to communicate'* in English

[6]http://www.microsoft.com/communicationsserver/

know of the existence of any conversation and so, for each conversation, his level of involvement is level 0. When a user logs in he is not part of any conversation. He can, however, see the list of conversations and thus be aware of them. So, his level of involvement in all conversations is of level 1. Finally, when the user becomes a participant of a certain conversation, either by being invited or by requesting to join, his involvement for that specific conversation increases to level 2. When, in turn, the user leaves the conversation, his involvement of that conversation drops back to level 1. To indicate the level of involvement Communico shows: (i) what people are logged in, (ii) what people are not logged in (iii) per conversation what users participate in it.

The third way in which Communico differentiates itself from existing tooling is with respect to persistence. By making conversations persistent it becomes possible to access the knowledge created by having these conversations, both by the participants of the conversations and by others. Several tools we looked at [37], [53], [52], [51], [49] support making conversations persistent, however they do not differentiate between different states of conversations. So when everyone leaves a conversation and joins it again the next day, they can continue the discussion and this is recognized as being the same conversation. In our view this is not a correct representation of what actually happened due to the synchronous nature of a conversation. In reality a second conversation about a similar subject occurs. To reflect this in the tooling we do provide persistence but also assigned them a state to indicate whether the conversation is still active. Conversations that are active are ongoing and so open for further contributions. Conversations that are no longer active are past conversations and no longer open for contributions.

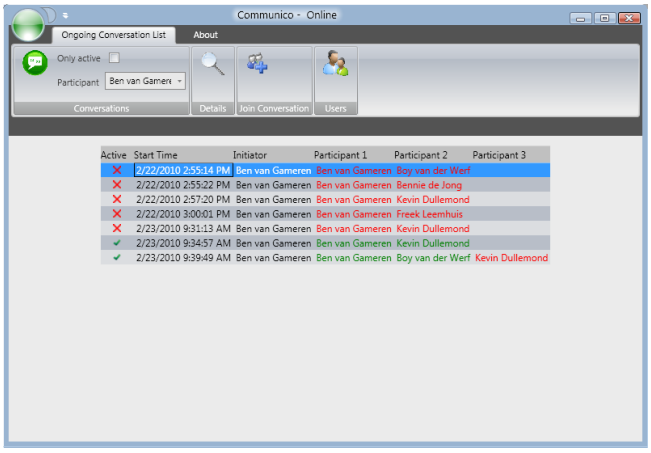

Figure 2. Conversations view

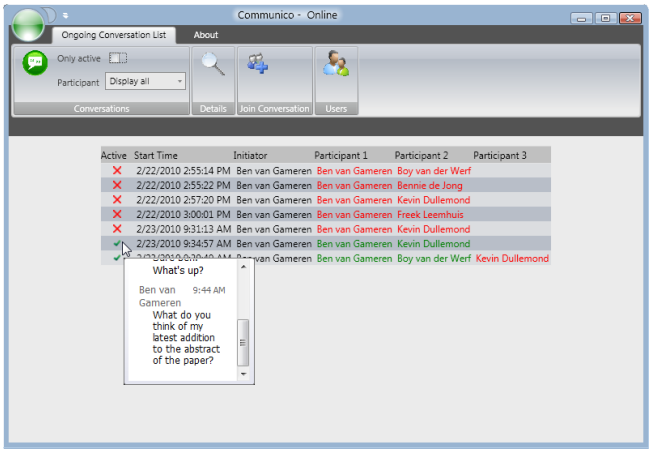

Figure 3. Conversation tooltip

B. User Interface

Having discussed what differentiates Communico from existing supporting technologies for virtual open conversation spaces, we continue by illustrating the tool itself. Communico uses the communication functionalities of Office Communications Server to allow for communication between its users and it accesses and displays information about the conversations that are going on, as well as those that have ended. The main screen of Communico can be seen in figure 2. In this view the conversations are displayed and updated in real time. Of each conversation it is possible to see the participants currently in the conversation (in green), the participants that have already left the conversation (in red), the start time, the initiator and whether the conversation is ongoing or not. When the mouse is moved over a certain conversation the last part of the conversation is displayed in a tooltip box to give an indication of the content of the conversation. This is shown in figure 3. In this view it is also possible to apply filters to the conversations that are displayed to aid the user. In this version of Communico it is possible to filter to show only conversations with a particular participant and/or only conversations which are active. Having chosen a certain conversation, one can chose to go to the detailed view, depicted in figure 4, either by double clicking the conversation or by selecting a conversation and clicking details. In the details window the complete content of the conversation is shown and updated in real time. Both from the detailed view and from the conversation overview described earlier, it is possible to request to join a conversation by clicking the appropriate button. The effect of this is that the initiator of the conversation is informed that you wish to join and he can decide whether or not to allow this. If he chooses to allow it, you are added to the conversation automatically. Finally, there is also a view showing which users are currently logged in to Communico. This is depicted in figure 5.

C. Technical Implementation

Finally, after the discussion of both the differences with existing tooling and showing the user interface, we will

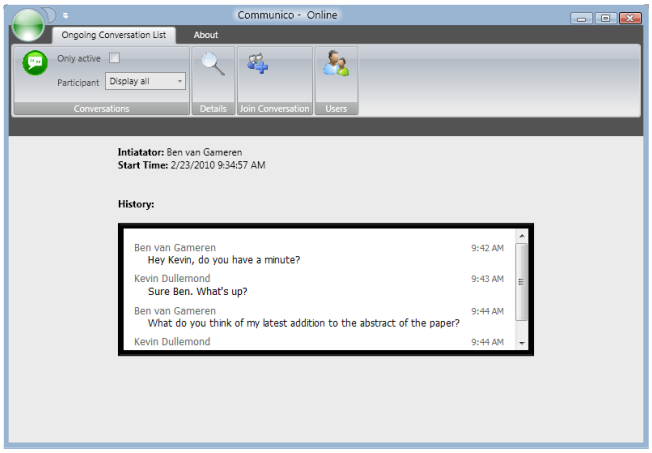

Figure 4. Conversation details view

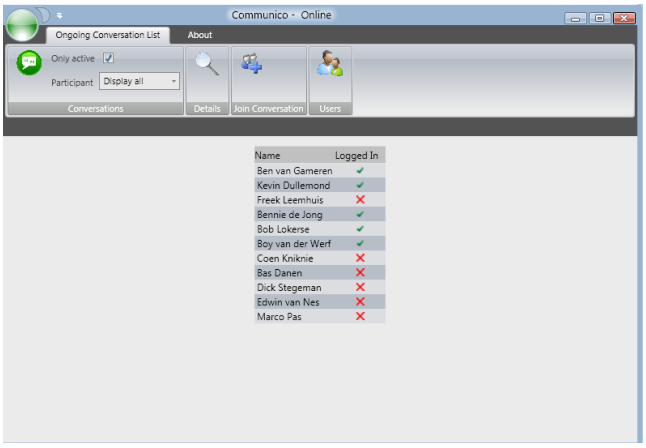

Figure 5. Users view

briefly discuss the technical implementation of Communico. When using Office Communications Server as a communication platform, a central server routes all communication and every user of the system runs a client (in this case Office Communicator 2007 R2) to use the functionality this server offers. The functionality includes things like IM, audio chat, video chat and screen sharing. Communico runs alongside Office Communicator on the machine of every user of the system. The functionality of Communico is threefold. Firstly, Communico uses the Office Communicator Automation API to gather data about users and conversations and stores this in a central database. When performing these actions, each instance of Communico cooperates with the instances of Communico running on the machines of the others users to determine which instance is responsible for what data and make sure all data is consistent. Secondly, Communico shows the data from the central database server and displays it in its graphical user interface which we discussed in the previous section. Finally, Communico also offers the possibility of joining a conversation. To do so, it writes a join request in the database and the instance of Communico that owns the conversation will handle the join request by asking the user of that machine whether or not to allow this. A global overview of the entire system is displayed in figure 6.

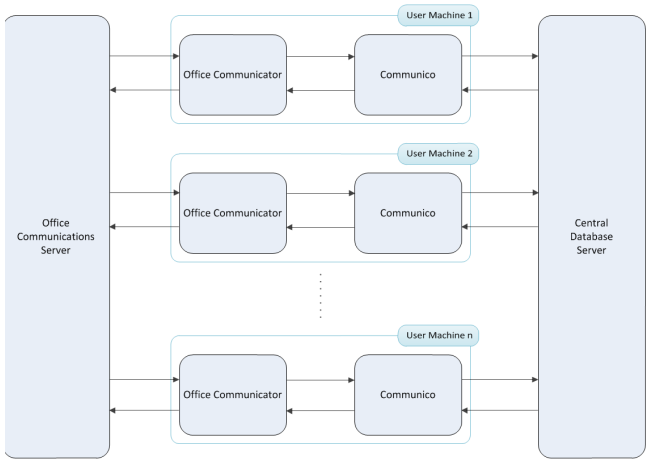

Figure 6. Overview of the Communico system

VI. Evaluation

To be able to reflect on the value of Communico in practice we conducted an informal evaluation to gather first impressions about its real use. The main purpose of this was not to validate Communico but to gather feedback from users and develop ideas for extensions and improvements. The evaluation was done over a period of two months in a small Dutch software engineering company, with senior developers who are used to working in GSE projects. Before the evaluation the employees of this company already used Office Communications Server to communicate while working on projects. During the evaluation they continued to do so, but five of the employees also used Communico, so they could benefit from the features it offers.

During the period of use we interviewed the users to evaluate Communico. In general the users of Communico found the added functionality beneficial and the overall consensus was that the tool is useful and the research direction highly promising. Firstly they enjoyed being able to see conversations others are having at the moment while working distributed. They reported that it helped them feel more connected. Secondly they also liked that they could actively join these conversations and did not have to wait passively to be invited. Finally they reported that being able to search through past conversations was beneficial as well, since this provided an insight in what conversations have taken place, as well as give the opportunity to access conversations which occurred when they were not working themselves. Besides being beneficial, the users also reported some limitations of Communico. Firstly, they did not know

when new conversations are started in the open conversation space. This is because they do not have the main screen of Communico on their display at all times and Communico does not provide another means of notifying its users that a new conversation has started. Secondly, they could not see who were watching a conversation they were having and they would like to be aware of this, just like they would be in a co-located setting. Thirdly, knowing what participants are in a conversation combined with the last sentence of a conversation is not sufficient to determine if a conversation is useful for them. Therefore they had to go to the detailed view of every conversation to determine this, which took time and effort. Fourthly, the filters that are available to search through the past conversations are insufficient as the list of past conversations gets substantial quite fast. Finally, using Communico all conversations are public and the users reported they would like to have the possibility to have private conversations as well.

VII. Limitations and Future Work

In the previous section we discussed some limitations of Communico that came up in the evaluation. In this section we will start by discussing possible improvements. Firstly, people did not know when new conversations were initiated in Communico. A solution to this is to notify people when new conversations are initiated. This should however be done non-obtrusively to minimize the interference with the other activities of the user. An example would be to notify the user of the start of a new conversation using a desktop alert: a small semi-transparent message in the bottom of the screen which automatically fades out after a short period of time.

Secondly, the users reported they would like to see who were actually watching the full content of a conversation they were having. Therefore, for a specific conversation, it should be visible what users are watching the details and what users are only viewing general information about the conversation. In order to accomplish this, we propose to extend the model depicting the levels of involvement, shown in figure 1, by defining a third level of awareness of a conversation, rated between no awareness and full awareness. In this third level one is aware of *general* information about the conversation. The extended model is shown in figure 7. When a user sees a conversation in the list of conversations he has general awareness about it, and thus is involved on level 1a. When a user goes to the full detailed view of a conversation he is fully aware about the conversation and is involved on level 1b. The other levels of involvement have remained the same. The levels of involvement in the extended model should be visible in a future version of Communico.

Thirdly it should be investigated what general information about a conversation is required to determine what conversations are useful to look at in more detail. For example automatic topic recognition or cloud tagging could

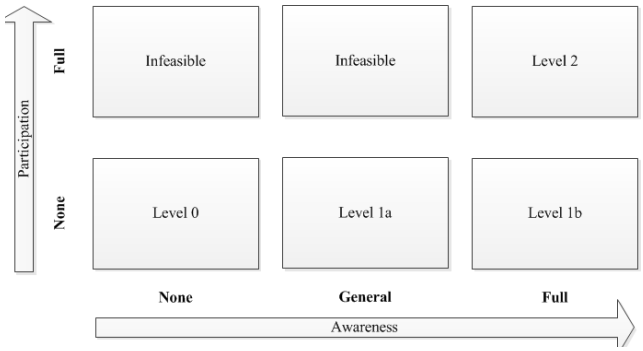

Figure 7. Model of conversation involvement

be investigated. Another seemingly simple solution would be to research manual tagging. Carrol et al. [56] however state: *"one of the striking lessons from 20 years of research on computer-supported collaborative work (CSCW) is that people will not, and do not want to share explicit intentional status information"*. Therefore an automated solution is likely to be the best alternative. The solution to the fourth limitation reported in the evaluation is likely to be related because it is similar in nature. This limitation concerned that the filters available to search through the past conversations are insufficient. Therefore it should be researched how these could be extended. Examples of possibilities of such an extension are filtering based on topic or date, or filtering on multiple participants instead of just one.

The final limitation reported in the evaluation concerned that all conversations are public in Communico. The users would like to also be able to have non-public conversations and this could be implemented by giving participants in a conversation the possibility of hiding it from the other people using Communico. In a normal office setting this is possible as well. In such a setting it is understood that what is discussed is public information [22]. It is, however, also possible to have a conversation in private by having the conversation in a separate office with closed doors. It will be interesting to see how much this functionality will actually be used in practice when it is added to Communico.

Following the possible extensions to Communico which arose from its evaluation in practice, we also think it is useful to research the addition of topic based conversation initiation to Communico. Some existing virtual open conversation spaces report the benefits of topic based conversation initiation and in the co-located setting this functionality is available as well. Therefore it seems interesting to research the use of a virtual open conversation space supporting both participant- and topic-based conversation initiation. Furthermore there has not been a formal evaluation of sufficient size to draw conclusions regarding the validity of the concepts Communico is based on. Therefore, after implementation of the improvements, we will validate Communico in an indus-

trial setting. In this validation we will research the extent in which Communico supports collaborative software development teams. We will do this by setting up an industrial case of sufficient size in which we will both monitor usage and interview users of Communico. This study will focus on the identification of the value Communico provides to its users, and the impact the use of Communico has on the perceived distance between dislocated team members. Finally, we will also continue our research in the ASPIC research program: focusing on increasing awareness in distributed collaborative software engineering by sharing relevant information about the work context.

VIII. Conclusion

In this paper we have answered the following research question: *"How can awareness about conversations within a development team support collaborative software engineering and how can this be facilitated by technology in a GSE setting?"* We answered the first part of the research question, by defining the concept of a conversation, discussing the reasons for having conversations and discussing the advantages of overhearing the conversations of colleagues. These advantages are the following: (i) it provides access to the information discussed in the conversations, (ii) it offers the possibility of joining the conversations and (iii) it provides insight in the communication structure of the project team. This led us to conclude that a work environment should both provide the possibility of having conversations with colleagues and make the conversations going on in the work space visible. We called a space which fulfills these requirements an open conversation space and discussed that in a GSE environment it is not automatically possible to have and overhear conversations. Therefore, explicit tooling is required to create an open conversation space in a GSE setting: a *virtual* open conversation space.

We answered the second part of the research question by: (i) examining the use of existing supporting tools for dislocated communication in an open conversation space, (ii) examining existing specialized tools which support transparent group communication and (iii) reporting on Communico, a virtual open conversation space we developed, and its evaluation in a practical setting. From this research we concluded that in a virtual open conversation space it is important to: (i) support participant based conversation initiation, (ii) make it transparent for the users how involved others are in conversations they are having and (iii) have access to persistent conversations with an explicit status indicating whether they are ongoing.

To summarize, the contributions made in this paper are the following:

- The identification of the insight that a lack of awareness is the origin of the challenges faced in GSE.
- Definitions of: conversation, correspondence, open conversation space and virtual open conversation space.
- An discussion of the extent in which existing tools support the concept of a virtual open conversation space.
- The development of Communico and discussion of its main characteristics.
- An evaluation of Communico in practice, resulting in ideas for extensions and improvements.

References

[1] E. Carmel, *Global software teams: collaborating across borders and time zones*. Upper Saddle River: Prentice Hall PTR, 1999.

[2] J. Herbsleb and D. Moitra, "Guest Editors' Introduction: Global Software Development," *IEEE Software*, vol. 18, no. 2, pp. 16–20, 2001.

[3] D. Damian and D. Moitra, "Guest Editors' Introduction: Global Software Development: How Far Have We Come?" *IEEE Software*, vol. 23, no. 5, pp. 17–19, 2006.

[4] J. Herbsleb, "Global Software Engineering: The Future of Socio-technical Coordination," in *Proceedings of the IEEE 2007 Workshop on the Future of Software Engineering*. IEEE Computer Society Press, 2007, pp. 188–198.

[5] R. Prikladnicki, J. Audy, D. Damian, and T. de Oliveira, "Distributed Software Development: Practices and challenges in different business strategies of offshoring and onshoring," in *Proceedings of the IEEE 2007 International Conference on Global Software Engineering*. IEEE Computer Society Press, 2007, pp. 262–274.

[6] B. Fitzgerald, P. Ågerfalk, H. Holmström Olsson, and E. Conchúir, "Benefits of Global Software Development: The Known and Unknown," in *Proceedings of the 2008 International Conference on Software Process*. Springer, 2008, pp. 1–9.

[7] E. Conchúir, H. Holmström Olsson, P. Ågerfalk, and B. Fitzgerald, "Exploring the Assumed Benefits of Global Software Development," in *Proceedings of the IEEE 2006 International Conference on Global Software Engineering*. IEEE Computer Society Press, 2006, pp. 159–168.

[8] R. Sangwan, M. Bass, N. Mullick, D. Paulish, and J. Kazmeier, *Global Software Development Handbook*. Auerbach Publications, 2007.

[9] R. Grinter, J. Herbsleb, and D. Perry, "The geography of coordination: dealing with distance in R&D work," in *Proceedings of the ACM SIGGROUP 1999 International Conference on Supporting Group Work*. ACM Press, 1999, pp. 306–315.

[10] J. Herbsleb, A. Mockus, T. Finholt, and R. Grinter, "Distance, dependencies, and delay in a global collaboration," in *Proceedings of the ACM 2000 Conference on Computer Supported Cooperative Work*. ACM Press, 2000, pp. 319–328.

[11] J. Herbsleb and R. Grinter, "Architectures, coordination, and distance: Conway's law and beyond," *IEEE Software*, vol. 16, no. 5, pp. 63–70, 1999.

[12] C. Ebert and P. De Neve, "Surviving global software development," *IEEE Software*, vol. 18, no. 2, pp. 62–69, 2001.

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

[14] P. Ågerfalk, B. Fitzgerald, H. Holmström Olsson, B. Lings, B. Lundell, and E. Conchúir, "A Framework for Considering Opportunities and Threats in Distributed Software Development," in *Austrian Computer Society*, August 2005, pp. 47–61.

[15] R. Battin, R. Crocker, J. Kreidler, and K. Subramanian, "Leveraging resources in global software development," *IEEE Software*, vol. 18, no. 2, pp. 70–77, 2001.

[16] L. Kiel, "Experiences in distributed development: a case study," in *Proceedings of the 2003 International Workshop on Global Software Development*, 2003, pp. 44–47.

[17] H. Holmström Olsson, E. Conchúir, P. Ågerfalk, and B. Fitzgerald, "Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance," in *Proceedings of the IEEE 2006 International Conference on Global Software Engineering*. IEEE Computer Society Press, 2006, pp. 3–11.

[18] J. Herbsleb, D. Paulish, and M. Bass, "Global software development at siemens: experience from nine projects," in *Proceedings of the IEEE 2005 International Conference on Software Engineering*. ACM Press, 2005, pp. 524–533.

[19] J. Herbsleb and A. Mockus, "An empirical study of speed and communication in globally distributed software development," *IEEE Transactions on Software Engineering*, vol. 29, no. 6, pp. 481–494, 2003.

[20] K. Schmidt, "The Problem with 'Awareness': Introductory Remarks on 'Awareness in CSCW'," *Computer Supported Cooperative Work*, vol. 11, no. 3-4, pp. 285 – 298, 2002.

[21] A. Syri, "Tailoring cooperation support through mediators," in *Proceedings of the 1997 European Conference on Computer Supported Cooperative Work*. Kluwer Academic Publishers, 1997, pp. 157–172.

[22] S. Greenberg and C. Gutwin, "A descriptive framework of workspace awareness for real-time groupware," *Computer Supported Cooperative Work*, vol. 11, no. 3-4, pp. 411–446, 2002.

[23] R. Kraut and L. Streeter, "Coordination in software development," *Communications of the ACM*, vol. 38, no. 3, pp. 69–81, 1995.

[24] P. Dourish and S. Bly, "Portholes: supporting awareness in a distributed work group," in *Proceedings of the ACM CHI 1992 Conference on Human Factors in Computing Systems*. ACM Press, 1992, pp. 541–547.

[25] C. Gutwin, R. Penner, and K. Schneider, "Group awareness in distributed software development," in *Proceedings of the ACM 2004 Conference on Computer Supported Cooperative Work*. ACM Press, 2004, pp. 72–81.

[26] J. Fogarty, S. Hudson, C. Atkeson, D. Avrahami, J. Forlizzi, S. Kiesler, J. Lee, and J. Yang, "Predicting human interruptibility with sensors," *ACM Transactions on Computer-Human Interaction*, vol. 12, no. 1, pp. 119–146, 2005.

[27] T. Allen, *Managing the flow of technology*. MIT press, 1977.

[28] R. Kraut, J. Galegher, and C. Egido, *Intellectual teamwork: Social and technological foundations of cooperative Work*. Hillsdale: L. Erlbaum Associates Inc., 1990.

[29] J. Olson and S. Teasley, "Groupware in the wild: lessons learned from a year of virtual collocation," in *Proceedings of the ACM 1996 Conference on Computer Supported Cooperative Work*. ACM Press, 1996, pp. 419–427.

[30] C. Soanes, Ed., *The Oxford compact English dictionary*. New York: Oxford University Press, 2003.

[31] F. Mish, Ed., *Merriam-Webster's collegiate dictionary.* Springfield: Merriam-Webster Inc., 2003.

[32] K. Woodford, Ed., *Cambridge advanced learner's dictionary*. Cambridge: Cambridge University Press, 2008.

[33] K. Dullemond and B. van Gameren, "Technological support for distributed agile development," Master thesis, Delft University of Technology, 2009.

[34] J. Espinosa and E. Carmel, "The impact of time separation on coordination in global software teams: a conceptual foundation," *Software Process: Improvement and Practice*, vol. 8, no. 4, pp. 249–266, 2003.

[35] Y. Ren and R. Kraut, "A Simulation for Designing Online Community: Member Motivation, Contribution, and Discussion Moderation." Manuscript in preparation: Retrieved from: http://www.cs.cmu.edu/∼kraut on February 24^{th} 2010.

[36] A. Webber, "What's so new about the new economy?" *Harvard Business Review*, 1993.

[37] T. Erickson, D. Smith, W. Kellogg, M. Laff, J. Richards, and E. Bradner, "Socially translucent systems: social proxies, persistent conversation, and the design of babble," in *Proceedings of the SIGCHI 1999 Conference on Human Factors in Computing Systems*. ACM Press, 1999, pp. 72–79.

[38] E. Wynn, "Office conversation as an information medium," Ph.D. thesis, University of California, Berkely, 1979.

[39] S. Greenberg and M. Rounding, "The notification collage: posting information to public and personal displays," in *Proceedings of the SIGCHI 2001 Conference on Human Factors in Computing Systems*. ACM Press, 2001, pp. 514–521.

[40] M. Sosa, S. Eppinger, M. Pich, D. McKendrick, and S. Stout, "Factors that influence technical communication in distributed product development: an empirical study in the telecommunications industry," *IEEE Transactions on Engineering Management*, vol. 49, no. 1, pp. 45–58, 2002.

[41] K. R. McCord, "Managing the integration problem in concurrent engineering," Master thesis, Massachusetts Institute of Technology, 1993.

[42] R. Kraut, R. Fish, R. Root, and B. Chalfonte, "Informal communication in organizations: Form, function, and technology," in *Human reactions to technology: Claremont Symposium on Applied Social Psychology*. Sage Publications, 1990, pp. 145–199.

[43] K. Ehrlich, G. Valetto, and M. Helander, "Seeing inside: Using social network analysis to understand patterns of collaboration and coordination in global software teams," in *Proceedings of the IEEE 2007 International Conference on Global Software Engineering*. IEEE Computer Society Press, 2007, pp. 297–298.

[44] M. Cataldo, P. Wagstrom, J. Herbsleb, and K. Carley, "Identification of coordination requirements: Implications for the design of collaboration and awareness tools," in *Proceedings of the 2006 ACM Conference on Computer Supported Cooperative Work*. ACM Press, 2006, pp. 353–362.

[45] K. Crowston and J. Howison, "The social structure of free and open source software development," *First Monday*, vol. 10, no. 2, 2005.

[46] A. Huurdeman, *The worldwide history of telecommunications*. Wiley-IEEE Press, 2003.

[47] A. Java, X. Song, T. Finin, and B. Tseng, "Why we twitter: understanding microblogging usage and communities," in *Proceedings of the 2007 Workshop on Web Mining and Social Network Analysis*. ACM Press, 2007, pp. 56–65.

[48] G. Trapani and A. Pash, "The complete guide to Google Wave," 2008, Retrieved from: http://completewaveguide.com on February 24^{th} 2010.

[49] M. Tran, Y. Yang, and G. Raikundalia, "SWIM: an alternative interface for MSN messenger," in *Proceedings of the 2007 Australasian Conference on User Interface*. Australian Computer Society, Inc., 2007, pp. 55–62.

[50] K. Inkpen, S. Whittaker, M. Czerwinski, R. Fernandez, and J. Wallace, "GroupBanter: Supporting Serendipitous Group Conversations with IM," in *Proceedings of the 2008 International Conference on Collaborative Computing: Networking, Applications and Worksharing*. Springer, 2008, pp. 485–498.

[51] T. Erickson, W. Kellogg, M. Laff, J. Sussman, T. Wolf, C. Halverson, and D. Edwards, "A persistent chat space for work groups: the design, evaluation and deployment of loops," in *Proceedings of the ACM 2006 Conference on Designing Interactive systems*. ACM Press, 2006, pp. 331–340.

[52] A. Ribak, M. Jacovi, and V. Soroka, "Ask before you search: peer support and community building with ReachOut," in *Proceedings of the ACM 2002 Conference on Computer Supported Cooperative Work*. ACM Press, 2002, pp. 126–135.

[53] M. Smith, J. Cadiz, and B. Burkhalter, "Conversation trees and threaded chats," in *Proceedings of the ACM 2000 Conference on Computer Supported Cooperative Work*. ACM Press, 2000, pp. 97–105.

[54] J. Birnholtz, C. Gutwin, G. Ramos, and M. Watson, "OpenMessenger: gradual initiation of interaction for distributed workgroups," in *Proceedings of ACM CHI 2008 Conference on Human Factors in Computing Systems*. ACM Press, 2008, pp. 1661–1664.

[55] M. Tran, Y. Yang, and G. Raikundalia, "Supporting awareness in instant messaging: an empirical study and mechanism design," in *Proceedings of the 2005 Australian Conference on Computer Human Interaction: Citizens Online: Considerations for Today and the Future*. Computer-Human Interaction Special Interest Group of Australia, 2005, pp. 1–10.

[56] J. Carroll, M. Rosson, U. Farooq, and L. Xiao, "Beyond being aware," *Information and Organization*, vol. 19, no. 3, pp. 162–185, 2009.