Empirical Study of Ad Hoc Collaborative Activities in Software Engineering

Sébastien Cherry[1], Pierre N. Robillard[1]

[1] Software Engineering Research Laboratory, Computer Engineering Department, École Polytechnique de Montréal, C.P. 6079. succ. Centre-Ville, Montréal, Canada

Abstract. This paper presents empirical research on ad hoc collaborative activities found in an industrial software engineering setting. We believe that a better understanding of these activities and their content will help us to propose software development process enhancements and also provide some insight into the tools needed to support communications in a distributed software development environment. Further details of our motivations are included, followed by a discussion on our research methodology, and, finally, some results of a preliminary analysis confirming the significance of our data and the importance of the observed phenomenon.

1 Introduction

It is well supported in the literature that some problems encountered in software development are not attributable to technical factors, but rather to the human aspects of software engineering [2], [4], [5], [7], [11], [13], [14], [15]. While some aspects, such as "communication" [7], [15], "coordination" [4], [5] and "collaboration" [2], [13], are gaining recognition in the research community, some methodological challenges emerge. Human factors, for example, have been overlooked in the past for many reasons, but principally because of the difficulty in measuring these facets quantitatively [11]. Nevertheless, empirical research in software engineering is growing in popularity and beginning to be adapted to studying this new topic of interest, namely people, and methods and techniques are being borrowed which were formerly used in the human sciences such as psychology and sociology. Like many researchers, we think that this domain will offer research opportunities for years to come.

This paper presents in-progress empirical research in the context of a case study in the industry, and explores collaborative work in software engineering; specifically, the ad hoc collaborative activities that take place during the software development process. By ad hoc collaborative activities, we mean activities which are not formally prescribed, and which occur between two or more developers working on a specific project task and which happen informally and spontaneously. They can take many forms, such as in peer-to-peer conversations, electronic mail exchanges, and so on.

Details of our motivations are to be found in the next section of this paper, followed by a discussion of the research methodology used, including data collection methods, and the need to place greater emphasis on the analysis techniques that will

Cherry S. and N. Robillard P. (2004).
Empirical Study of Ad Hoc Collaborative Activities in Software Engineering.
In *Proceedings of the 1st International Workshop on Computer Supported Activity Coordination*, pages 116-125
DOI: 10.5220/0002682701160125

be used to explore the large quantity of amassed data. Finally, some results of a preliminary analysis are revealed, which confirm both the relevance of the collected data and the importance of the observed phenomenon.

2 Motivations

2.1 Why collaborative work?

As previously mentioned, a growing number of researchers support the view that many of the problems that arise during software development could be imputable to human factors associated with the software engineering process. Perry, Staudenmayer and Votta (1994) [11], among others, believe that too much attention is given to the technological aspects of software engineering. They say that one of the reasons frequently mentioned is the difficulty of measuring these human factors quantitatively. The same comments are also supported by Seaman (1999) [16].

Many approaches have been envisaged to study the human aspects of software engineering. Some researchers have examined the communication occurring during software development [7], [15], while others have studied the coordination aspect [4], [5] and still others were interested in the collaborative work [2], [13].

With regard to collaborative work, Robillard and Robillard (2000) [13] have empirically identified four types of collaborative activities performed during the software engineering process. They defined “ad hoc” collaborative activities, for example, as the work carried out simultaneously by teammates on a particular task of the project and which is not prescribed by a formal process. One of the conclusions of this research was that ad hoc collaborative activities can play a major role in team communication dynamics, accounting for 41% of this dynamics during the case study. Furthermore, these activities constitute the longest of the working sessions, and in addition seem to have an important impact on individual activities, since they often precede long individual working sessions.

Also, Perry, Staudenmayer and Votta (1994) [11] found during another case study that informal communications take up an average of 75 minutes per day per software developer. Seaman (1996) [15] also supports the view that this type of communication is a non-negligible element to be taken into account during a development process, and which is essential if developers are to carry out their tasks adequately.

Because they monopolize quite a considerable part of a software project and constitute an important element of it, as established above, exploratory research is essential to understanding the content of these ad hoc collaborative activities and the communication that ensues, and to measuring the impact, both positive and negative, on the rest of the development process. We believe that such research will help us to subsequently propose software engineering process enhancements which will be better adapted to the human and empirical realities of software development.

By contrast, collaborative software development, also known as “distributed software development”, is an increasingly fashionable domain of research these days.

Both these expressions refer to software development distributed over time and over relatively long distances, something that has become quite common business practice nowadays in cases where it occurs.

However, according to recent research in this domain [5], the distances between the members of virtual teams tends to obstruct informal communications, resulting in problems of coordination. This is another important reason for undertaking research in this field. It will also provide some insight into the tools needed to support communications in a distributed software development environment.

2.2 Why an empirical study?

Empirical research based on the experimental method has been conducted for a long time now, in many of the human sciences, such as psychology and sociology. It is, moreover, very often considered to be the only valid scientific method accepted in these domains. Although empirical research has been conducted in software engineering for many years, it is on a much smaller scale and only quite recently has it seen an increase in popularity. One reason for this is the growing interest in the human aspects of software engineering [16].

Further arguments supporting this new tendency, and strengthening the evidence for it, have been expressed by Tichy (1998) [17] in his paper, "Should Computer Scientists Experiment More?" However, those who uphold this practice in software engineering believe that, since the quantity of empirical research is on the increase, its quality should increase as well [10], [18].

3 Research Protocol

3.1 Problem Statement

Research Objectives. As discussed previously, the importance and the necessity for ad hoc collaborative work and the communications that ensue in software development are widely supported by many authors [2], [4], [5], [6], [7], [11], [13], [15]. Although some research has quantified the importance of the phenomenon, there has been no known attempt to determine and describe the content of that work. These considerations led us to define the following research objectives:

- To observe the collaborative work taking place in a case study in the industry to design a conceptual model and distinguish some patterns of exchanges.
- To characterize the ad hoc collaborative activities found, the communications that ensue, and to identify and describe their content.
- To generate a series of hypotheses which emerges from the results of this research, and which could later be validated by confirmatory research.

Theoretical Relevance of the Research. The fact that there has been little or no empirical study of ad hoc collaborative activities gives this research theoretical relevance. It will make it possible to establish a model of these activities and to gain a sense of the cognitive aspects involved from which we will be able to generate a series of hypotheses to create a theoretical base in this domain.

Practical Relevance of the Research. Based on the fact that good collaboration is an indispensable condition of a software development team working effectively to make a quality product which meets the needs of the user, in the time required and at the expected cost, this research is relevant in practice because it will potentially pave the way to the proposal of improvements to software engineering processes. Also, as previously mentioned, it will allow us to better understand the informal collaboration and communication aspects of software engineering, and provide some insight into the tools needed to support communications in a distributed software development environment.

3.2 Research Methodology

General Approach. The research is carried out by means of participant observation within the framework of a case study in an industrial environment. This type of approach is suitable in our case because this is exploratory research. Also, as Jorgensen (1989) [8] and Babbie (2001) [1] have stressed, study in the field combined with participant observation are appropriate when it is not a question of empirically verifying hypotheses formulated in advance, but rather of inductively generating theories from observations and from the empirical data collected.

Target-setting. The setting in which the chosen software development team works is a large enterprise which produces software for commercial purposes. It is a well-established, mature concern which has been in operation for several years, and where there exists a clearly defined development process. Nevertheless, even if the observations are done in a large company, this last also contains some attributes of smaller organizations since the development of software components is divided into small teams.

Also, based on a common-sense judgment (face validity) [1], we can say that the chosen team of eight individuals is representative of the majority of development teams, with a wide range of ages, amounts of schooling, years of experience in software development and length of service in the company.

Data Collection. The following data collection methods were identified from a preliminary ethnography period within the chosen team which lasted several months.

An initial data collection phase, which took place in the autumn of 2003, is now completed and was spread out over 8 weeks. The results presented in this paper were produced from these earliest data. The purpose of this collection was to gather the

maximum amount of information from the beginning to the end of the development of an update (patch) of a given version of the software produced.

The data collected during this first phase includes:

- 185 hours of audio-video recordings of working sessions over 37 workdays
- The capture of a total of 2496 e-mails exchanged by the 8 teammates
- A daily backup of the source code and other documents and artifacts found

E-mails were captured automatically, by means of triggers defined in the messaging software used in the company. This capture included both e-mails received and those sent by teammates, in order to permit cross-validation.

The daily backup of the source code, and the various documents and artifacts, was available for potential use for subsequent content analysis.

Data Analysis. One of the techniques that will be used for the analysis is Exploratory Sequential Data Analysis (ESDA) [3]. This technique is suited to exploratory research, where the objective is to find answers to research questions or to find patterns among the empirical data and to describe them using, for example, simple statistical representations.

ESDA allows researchers to define, from these descriptions, hypotheses which are subsequently verified by means of confirmatory research using statistical inference methods. However, the important feature of ESDA is that it applies more specifically to research where the sequential integrity of the data must be preserved.

Of the eight operations proposed by ESDA, encoding is certainly the most important. This involves labeling each sequence of data by means of a code formed using a particular syntax and contained in an exhaustive, exclusive and relatively restricted category list, and doing so to decrease the variability of the data, as well as to facilitate its subsequent manipulation. This encoding makes it possible to transform qualitative data into quantitative data, on which it is then possible to perform statistical analysis [3], [16].

The ESDA process, such as proposed by Fisher and Sanderson (1996) [3], is an iterative one, involving the definition of a series of concepts stemming from research questions of interest. The process will subsequently drive what should be observed among the collected raw data and what manipulations should be made to obtain derived data on which it is possible to generate theories or define hypotheses. It is iterative because it is often necessary to revisit certain steps; for example, to add, remove or redefine concepts or categories that are sometimes found intuitively and validated by their statistical representations.

Research Validity. To satisfy the validity criterion for the research, the empirical measure must faithfully translate the empirical reality of the measured phenomenon [1]. To enhance the validity of our research, particular attention is directed to the definition of the concepts or categories chosen to encode the data. These concept definitions, which must arise from the ESDA traditions that concern us [3], as well as from the context of the research field, ensure a degree of representativeness of the phenomenon under study by common-sense validity (face validity) [1], [8].

The concepts or categories under which the data will be encoded, as well as the number of categories chosen, will also be very important as far as content validity is

concerned [1]. This aspect of validity refers rather to the coverage of the meanings encompassed by the concepts. Furthermore, the validity of the connections or the relations (construct validity) [1] should be assured among the concepts forming the theoretical model appearing from the data. This can be done by means of certain correlation measures or statistical associations.

Finally, a data triangulation will be made between qualitative and quantitative data, as well as of data resulting from various sources [16], [18]. Concerning this last point, other phases of data collection are to be anticipated.

4 Preliminary Results

The results presented in this section are the product of a preliminary analysis concerning four of the eight developers in the team who were observed over a period of 8 hours. The choice of these individuals was not made by means of a sampling method, but from direct observations in the field: they had been identified as being likely to work more collaboratively than the others. This choice is justified because the objective of this research is not to find a magic number indicating the time spent on ad hoc collaborative work, but rather to investigate the content of these collaborative activities. It should also be noted that the results below do not take into account e-mail interactions, but only the peer-to-peer conversations and telephone exchanges.

As can be seen from Figure 1, 51% of the time is spent on ad hoc collaborative work, as against 49% for the other types of activities. This result tends to corroborate the observations made in the field that suggested the importance of the phenomenon.

Fig. 1. Distribution of time spent on ad hoc collaborative activities in comparison with other types of activities

Figure 2 indicates the percentage of time spent on ad hoc collaborative activities by the subjects observed. As was noted in the field, subjects MS2 and MS3 seem to have spent a large amount of their time collaborating and communicating in a spontaneous way with their colleagues. This can be explained by the nature of the work performed by these subjects. MS2 occupies the position of project manager in the team, and one of his functions is to circulate relevant information needed by the developers on his team. When we examine more closely the interactions in which MS2 is involved, we note that, for 78.13% of the time, his colleagues initiate the interactions. We can suggest hypothetically that MS2 constitutes a source of the information his colleagues require. However, it was noted in the field that a great deal of the information passed on by MS2 to his teammates is in the form of e-mails. It would be interesting to investigate this method of communication. MS3 is, for his part, responsible for the infrastructure of the software built, and often the individual consulted to

resolve problems. He manages several tasks at the same time, which brings him into communication with some of his colleagues more often.

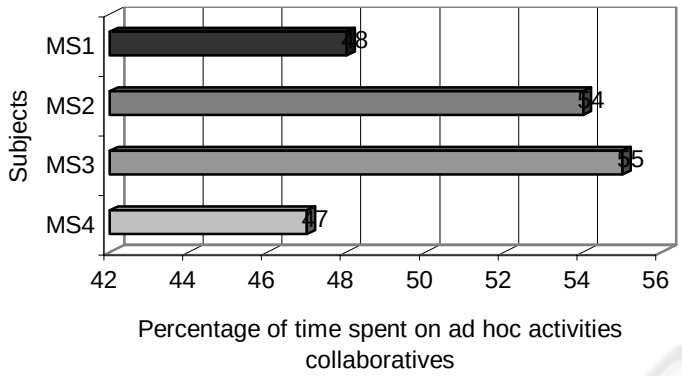

Fig. 2. Percentage of ad hoc collaborative activities per subject

By contrast, the average duration of the interactions analyzed by the four developers is 6:31 minutes, and these interactions involve, on average, 2.3 stakeholders. We should remember that an interaction is defined as a communicative unit which presents an evident internal continuity, while it breaks with what precedes it and what follows it [11]. Moreover, these results were based on a total of 82 observed interactions.

Figures 3 and 4 give an initial outline of a distribution as a percentage with regard to the number of occurrences and time spent on the various categories of ad hoc collaborative activities identified. “Cognitive synchronization” exists when two or more developers exchange information to ensure that they share the same knowledge or the same representation of the object in question. “Problem resolution” occurs when two or more developers are aware of the existence of a problem and attempt by different means to solve the problem or to mitigate it. “Development” occurs when two or more developers contribute to the development of a new feature or component of the software. “Management” is the result of two or more developers coordinating and planning activities such as meetings, common working sessions or setting schedules. “Conflict resolution” is the process of two or several developers taking part in discussions to resolve a difference of opinion. Ad hoc collaborative activities under the “not relevant” category group together all interactions which do not concern the project or the software built.

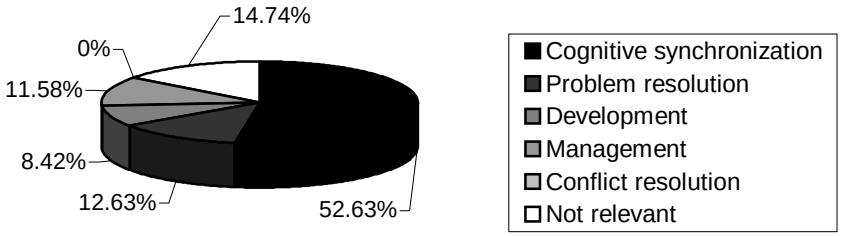

Fig. 3. Distribution in number of occurrences of ad hoc collaborative activities identified

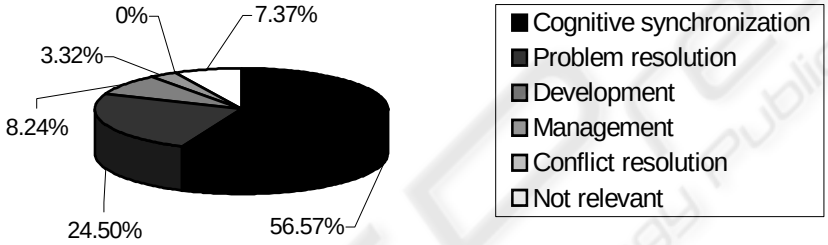

Fig. 4. Distribution in terms of time spent on ad hoc collaborative activities identified

As Figure 3 suggests, 52.63% of the ad hoc collaborative activities that arose are forms of cognitive synchronization. This agrees with the direct observations made in the field. The figure is not surprising when we consider that the exchange of information and knowledge constitutes an essential element in software development, which is to crystallize [16] all the information required in quality software to meet the needs of the user. As shown in Figure 4, cognitive synchronization occupies 56.57% of the time spent on ad hoc collaborative work, which also supports the previous finding.

The other significant category, problem resolution, does not seem important in Figure 3 in terms of number of occurrences. Figure 4, however, suggests that this activity monopolizes almost a quarter of the time spent on ad hoc collaborative work by the four subjects observed. It demonstrates that perhaps, while problem resolution activities are relatively few in number, when they do arise, they monopolize a rather considerable amount of time. An analysis of the mean time spent by sequence as a function of ad hoc collaborative activity tends to confirm this, showing that problem resolution takes up to 9:48 minutes when it occurs, as opposed to the interaction average of 6:31 minutes.

The results relative to management activities also reveal an interesting finding. Unlike problem resolution activities, they are relatively numerous considering the fairly short time that they occupy. This may tend to confirm the theories of certain authors [7], who maintain that informal communications are necessary in order that the members of a team can coordinate their activities effectively.

5 Conclusion

It is clear, and widely supported, that good collaboration and communication are an essential condition of the successful delivery of a quality product by a software development team, one that meets the user's needs in a timely fashion and at the expected cost.

It was revealed by previous research that ad hoc collaborative activities and informal communications occupy a considerable portion of the time that a developer spends on a software project. However, no research has attempted to describe the content of these activities, which leaves a vast field open for exploration.

The empirical research described in the present paper suggests the importance of investigating this field, because the authors believe that understanding how people collaborate will make it possible to propose practices to enhance collaboration and communication within a development team, as well as improve software development processes.

This article has briefly presented the methodology used to meet our research objectives. It was influenced by previous empirical research which was also aimed at investigating the human aspects of software engineering, but which was, however, adapted to the context of the field on which this study focuses.

The embryonic results that were partially presented in this paper, and that arise from a tiny portion of the considerable quantity of data that was collected, are very interesting. Although more analysis is needed, a model of data and patterns already seems to be emerging which will allow us to subsequently form hypotheses which can be validated by other, confirmatory research, thereby forging a knowledge base in this, as yet unknown, domain of software engineering.

6 Acknowledgments

This research would not have been possible without the agreement of the company in which it was conducted, and without the generous participation and patience of the software development team members from which the data was collected. To all these people, we extend our grateful thanks.

References

1. Babbie, E.: The Practice of Social Research, 9th ed., Wadsworth Publishing Company, Belmont, CA (2001)
2. D'Astous, P., Robillard, P.N.: Les aspects de l'échange d'information dans un processus de génie logiciel, Rapport interne, École Polytechnique de Montréal, EPM/RT-97-06 (1997)
3. Fisher, C., Sanderson, P.: Exploratory Sequential Data Analysis: Exploring Continuous Observational Data, Interactions, Vol.3, No. 2, Mar. (1996)
4. Grinter, R.E., Herbsleb, J.D., Perry, D.E.: The geography of coordination: Dealing with distance in R&D work, GROUP'99: International Conference on Supporting Group Work, Coordination and Negotiation, (1999) 306-315

5. Herbsleb, J.D., Grinter, R.E.: Splitting the Organization and Integrating the Code: Conway's Law Revisited, Proceedings, International Conference on Software Engineering, Los Angeles, CA (1999) 85-95
6. Herbsleb, J.D., Moitra, D.: Guest Editors' introduction: Global software development. IEEE Software, Vol.18 No.2, March/April (2001) 16-20
7. Herbsleb, J.D., Mockus, A.: An empirical study of speed and communication in globally distributed software development, IEEE Transactions on Software Engineering, Vol.29, No.6, June (2003) 481-494
8. Jorgensen, D.L.: Participant Observation: A Methodology for Human Studies, Applied Social Research Methods Series; v.15, Sage Publications, Newbury Park, CA (1989)
9. Kerbrat-Orrecchioni, C.: Les Interactions Verbales, Armand Colin, Paris (1998)
10. Kitchenham B. et al., Preliminary guidelines for empirical research in software engineering, IEEE Transactions on Software Engineering, Vol.28, No.8. (2002) 721–734
11. Perry, D.E., Staudenmayer N., Votta, L.G.: People, Organizations, and Process Improvement, IEEE Software, July (1994)
12. Robillard, P.N.: Études des aspects cognitifs applicables au génie logiciel, Rapport interne, École Polytechnique de Montréal, EPM/RT-96/18 (1996)
13. Robillard, P. N., Robillard, M.P.: Types of Collaborative Work in Software Engineering, The Journal of System and Software, No.53 (2000) 219-224
14. Robillard, P. N., Kruchten, P., D'Astous, P.: Software Engineering Process with the UPEDU, Addison Wesley, Pearson Education (2002)
15. Seaman, C.: Organizational Issues in Software Development: An Empirical Study of Communication, PhD Thesis, Computer Science Department, University of Maryland, Technical Report CS-TR-3726, UMIACS Technical Report UMIACS-TR-96-94 (1996)
16. Seaman, C.: Qualitative methods in empirical studies of software engineering, IEEE Transactions on Software Engineering, Vol.25, No.4 (1999) 557–572
17. Tichy, W.: Should computer scientists experiment more? Computer, Vol.31, No.5 (1998) 32–40
18. Walker, R.J., Briand, L.C., Notkin, D., Seaman, C.B., Tichy, W.F.: Panel: Empirical Validation–What, Why, When, and How, Proceedings, International Conference on Software Engineering, Portland, Oregon (2003) 721-722