Analysing Communication Trends in Pair Programming Videos using Grounded Theory

Mark Zarb
School of Computing
University of Dundee
Dundee, Scotland
markzarb@computing.dundee.ac.uk

Janet Hughes
School of Computing
University of Dundee
Dundee, Scotland
jhughes@computing.dundee.ac.uk

John Richards
IBM T.J. Watson Research Center
Hawthorne
New York, USA
ajtr@us.ibm.com

Abstract: Communication, much of it subtle and non-verbal, is important for programmers engaged in pair programming tasks. Little is known about the nature of this communication, for example how it changes over the course of a programming project or how it might differ according to programming skill and experience with paired programming. This paper presents an initial investigation of 60 videos of expert-expert pair programming, analysed in detail using grounded theory. The aim was to make general observations about the various ways communication occurs within the programming pair and to identify any patterns in behaviour or style of communication in the context of pair programming. As a result, a general coding scheme has been created. Preliminary data analysis shows that it may be possible to detect certain communication patterns by using transcription code occurrence and duration. This could be used to further explore communication topics and trends within pair programming.

Pair programming. Communication. Video analysis. Grounded theory. Collaboration. Coding.

1. INTRODUCTION

Pair programming is one of the twelve key practices of Extreme Programming (XP), which is a method for software development "that favours both informal and immediate communication over the detailed and specific work products required by any number of traditional design methods" (Williams et al., 2000).

Pair programming is primarily a coding activity during which two programmers collaborate continuously on the same program, usually at the same computer. The pair takes on different roles: one, the driver, has full control of the keyboard, whereas the other (known as the navigator) is in charge of reviewing the code and performing continuous analysis of the design (Williams and Kessler, 2000). It is common practice for partners to switch roles frequently. Ultimately, pair programming encourages programmers to talk to each other and to themselves – this 'pair pressure' adds benefits such as greater enjoyment and increased knowledge distribution (Bryant et al., 2006).

2. PROJECT OVERVIEW

Existing studies show that pairs who exhibited a high level of communication between partners did not necessarily experience a high level of satisfaction or mutual compatibility, and does not guarantee a high level of confidence regarding the finished product (Choi et al., 2009). This indicates that it might not enough for the partners to communicate between themselves – it is also important that the communication stays on-topic and that it is used to drive the work forward.

As part of an ongoing series of investigations into the nature of pair programming, the present paper reports on a preliminary study using audio, video, and screen-capture data, and examines the evolution of intra-pair communication over the course of a moderately large software project.

2.1. Collection Methods

Prior to this study, a series of videos were independently created by two software engineers in order to introduce agile software development to a wider audience. These videos were broadcast live at the time of filming, and are now archived online (Marcano and Palmer, 2009). The videos generally consist of a screencast with live code updates, live audio commentary, and a smaller webcam-captured feed at the bottom of the screen showing the two programmers interacting. The three actions (audio, video and code) are synchronized.

To date, the programmers have uploaded sixty pair programming videos showcasing 'live' pair programming, made publicly available on

vimeo.com under the name *pairwith.us*. The team have agreed to allow the first author to analyze these materials in a preliminary investigation into communication and pair programming.

At the beginning of each video, the pair programmers start a 25-minute countdown timer, which signals the end of their coding session. The video ends after the timer has counted down, at which point the video is stopped, and the programmers take a short break. Upon their return, they start recording the next video in the sequence using the same procedure.

3. QUALITATIVE METHODOLOGY

At this stage, our investigation is concerned with analysing the content of the videos collected from the pairwith.us team, and examining patterns and topics of the verbal and non-verbal communication that occurs between the two participants. Due to the natural, unstructured setting of these videos, and the investigation's concern with the words, actions, and points of view of the participants, a qualitative research method was chosen for this study.

Following a careful review of methodologies such as ethnography and phenomenology (Wertz et al., 2011; Ritchie and Lewis, 2003; Bryman, 2012), Grounded Theory was selected for data analysis during this investigation: it allows data coding to be used to generate evidence, as well as promoting constant refinement and comparisons of the data gathered.

3.1. Grounded Theory

Grounded theory is a systematic methodology that has become the most widely-used framework for the analysis of qualitative data (Bryman, 2012). It involves analysis of data through observations, interactions and materials gathered by researchers. Whereas most other qualitative methods allow researchers free rein to follow up on potentially interesting data, grounded theory provides methods which have explicit guidelines that indicate how the research should proceed. Furthermore, these methods can complement other approaches to qualitative data analysis, rather than stand in opposition to them (Charmaz, 2006). According to Myers (2008), grounded theory is "very useful in developing […] descriptions and explanations of organizational phenomena".

The original defining components of grounded theory include the following (Glaser and Strauss, 1967; Dick and Zarnett, 2002):

- Simultaneous involvement in data collection and analysis;
- Constructing analytic codes and categories from data, not from preconceived logically deduced hypotheses;
- Using the constant comparative method, which involves making comparisons during each stage of the analysis;
- Advancing theory development during each step of data collection and analysis;
- Sampling aimed toward theory construction, not for population representatives.

Grounded theory has a number of advantages over other qualitative research methods. It provides a systematic approach to analysing text-based transcripts created from raw video data, allowing researchers to use coding to generate evidence, which backs up the generated theory. It is an intuitive process for novices, and due to the early data coding stage, encourages constant refinement of the theory through frequent comparisons between data collection and the data analysis (Myers, 2008; Lazar et al., 2009).

4. INVESTIGATION

All sixty pairwith.us videos were reviewed to identify any unsuitable for analysis due to various problems which could hinder the investigation. As a result, the videos with bad audio-visual quality, no visual feed, and a prominent, distracting echo during the recording were rejected. The 31 remaining videos were recorded in a time-span of three months, between May and July 2009.

The remaining videos were analysed using the Grounded Theory approach.

4.1. Transcribing Videos

Videos provide very rich material that can be a great resource due to the wealth of layers embedded in the information – the researcher has access to not only the participants, but to the setting, their gestures, their speech and their activity. The richness of the available data can be reflected by noting down gestures or events that are thought to be of importance as well as by transcribing the conversations.

Transcription has the advantage that the researcher can quickly identify key themes and become aware of any similarities and differences within the data (Bryman, 2012). Two methods for transcription were considered:

(i) Scratch Notes are very brief notes that are taken down at the time of the activity; they can be used to jog the researcher's memory at a later date, allowing him or her

to recall a full account of an event, which can be expanded later. Scratch notes are considered to be a first step from field perception to paper (Sanjek, 1990), and are mostly used "in the field", when taking full notes would be too time-consuming.

(ii) A full transcription emphasizes certain features of talk beyond merely *what* is said simply by recording them. The transcript can be used to highlight features that the analyst deems to be meaningful; e.g. how people speak, sounds that are not words, etc. Transcribing videos isa highly time-consuming process, even relative to audio recordings. It can take anything from two to ten hours per hour of video, at a minimum (Chong et al., 2005) and is also subject to human error.

The first author began by fully transcribing two of the videos – a process which lasted over seven hours.

Wetherell et al. (2001) state that working from recordings (audio or visual) is a good alternative to transcription, "especially for preliminary analysis". These circumstances, as well as the researcher's wish to minimise the degree of error, led to the decision to watch the videos with minimal pausing or stopping, using scratch notes to note down interesting codes and information – a process known as open coding. This gave the researcher an over-arching sense of the different themes and types of communication that the two programmers exhibited during the course of the videos.

4.2. Observations through Open Coding

The following were identified through open coding and field notes as the most frequently occurring trends in the programmers' communication after an initial viewing of all 31 videos:

(i) The mouse pointer was used to attract attention to a specific area in the code.

(ii) Several types of communication between the programmers were non-verbal, consisting of actions such as deep inhalations of breath, tutting, scoffing, or pointing at sections of the code on screen as a prompting device.

(iii) Utterances or short phrases were used by the navigator to communicate their feelings to the driver:

(a) 'Mmhmm', 'that's right', 'yeah', 'OK', or a simple repetition of what the driver had just said indicated approval;

(b) 'Don't know', 'hmmm', 'no', 'actually…', well…', 'except…' and 'but' frequently indicated disapproval.

(iv) There is a constant awareness and very frequent discussion as to why the current task is being carried out and what is expected to happen following this task; i.e. is the code expected to compile, or break? What shall be achieved, following these actions?

(v) There is a constant awareness and regular discussion as to what the immediate future steps will be, and what are the aims behind them.

(vi) Both programmers can be simultaneously silent. This seems to indicate concentration, distraction, uncertainty, or a lengthy coding period.

(vii) The driver generally tends to speak out or mutter what they are currently typing.

Several tendencies were also identified in the programmers' behaviour, which could have a direct impact on their communication:

(i) The driver tends to ask the navigator for confirmation before proceeding.

(ii) The navigator generally dictates what the driver should do, and is mostly concerned with future-proofing the code.

(iii) Both programmers are concerned not just by the fact that the code works/compiles, but also by its logic and its aesthetic.

(iv) The 25-minute timer is mostly ignored, as the programmers are concerned with getting the task at hand to work, rather than to stick to the time limit.

(v) Errors are used as prompting devices for tasks.

(vi) The switch from driver to navigator generally occurs after the driver completes a set test or task, but the navigator can request control of the keyboard to explain their particular point through code, rather than speaking it out.

(vii) Jokes and off-topic conversations seemed to be used more frequently in earlier videos.

4.3. Code Development

The data collected resulted in a need to run more in-depth analysis to gain a clearer understanding of the communication trends that the researcher was witnessing.

The video field notes were compared to the observations listed above to generate a list of key codes that could be used to describe each instance of communication within the videos.

A random sample of five pairwith.us videos was next selected, transcribed and coded using Transana[1]. At this stage of the analytic process, the researcher felt that the degree of precision that a full verbatim transcript would provide would lead to a more complete coding schema.

A set of analytics codes and categories were identified. These codes evolved and changed shape throughout this process – for example, the code for 'Silence' was eventually split into two: 'Working Silence' and 'Muttering', and a new code was generated for 'Suggestions'.

5. CURRENT WORK

The five coded videos provide over two and a half hours of rich data which will be used for further analysis and discovery. Currently, the code duration and occurrence rate within these videos is being used to examine potential communication patterns within the pair.

Throughout the duration of this investigation, the researcher noted an increasing awareness of the importance of the navigator's role within the pair.

The current data is limited as it focuses on the same pair – in future studies; we expect to collect more footage of pair programming from multiple programmers in order to validate the coding scheme. This shall then be applied to novice pairs. Following this, further experiments shall be run to explore any benefits of communication within pairs, as well as to investigate the various roles the navigator plays within such a setting.

6. ACKNOWLEDGEMENTS

The researcher thanks pairwith.us for allowing the use of their videos in this research project.

The research work disclosed in this publication is funded by the Strategic Educational Pathways Scholarship (Malta). The scholarship is part-financed by the European Union - European Social Fund (ESF) under Operational Programme II - Cohesion Policy 2007-2013, "Empowering People for More Jobs and a Better Quality of Life".

[1] http://www.transana.org/

7. REFERENCES

Bryant, S., Romero, P. & du Boulay, B. 2006. The Collaborative Nature of Pair Programming. *In:* Abrahamsson, P., Marchesi, M. & Succi, G. (eds.) *Extreme Programming and Agile Processes in Software Engineering.* Springer Berlin / Heidelberg.

Bryman, A. 2012. *Social Research Methods*, Oxford University Press.

Charmaz, K. 2006. *Constructing grounded theory: a practical guide through qualitative analysis*, SAGE.

Choi, K. S., Deek, F. P. & Im, I. 2009. Pair dynamics in team collaboration. *Computers in Human Behavior,* 25**,** 844-852.

Chong, J., Plummer, R., Leifer, L., Klemmer, S. R., Eris, O. & Toye, G. 2005. Pair programming: When and why it works. *In:* 17th Annual Workshop of the Psychology of Programming Interest Group, 2005 Brighton, UK. 43-48.

Dick, A. J. & Zarnett, B. 2002. Paired programming and personality traits. *XP2002, Italy*.

Glaser, B. G. & Strauss, A. L. 1967. *The discovery of grounded theory: Strategies for qualitative research*, Aldine de Gruyter.

Lazar, J., Feng, J. H. & Hochheiser, H. 2009. *Research methods in human-computer interaction*, Wiley.

Marcano, A. & Palmer, A. 2009. *pairwith.us* [Online]. Available: http://vimeo.com/channels/pairwithus [Accessed 31 July 2012].

Myers, M. D. 2008. *Qualitative research in business & management,* London, Sage Publications Ltd.

Ritchie, J. & Lewis, J. 2003. *Qualitative research practice: a guide for social science students and researchers,* London, Sage Publications Ltd.

Sanjek, R. 1990. A vocabulary for fieldnotes. *Fieldnotes: The makings of anthropology***,** 92-121.

Wertz, F. J., Charmaz, K., McMullen, L. M., Josselson, R. & Anderson, R. 2011. *Five Ways of Doing Qualitative Analysis: Phenomenological Psychology, Grounded Theory, Discourse Analysis, Narrative Research, and Intuitive Inquiry*, Guilford Press.

Wetherell, M., Taylor, S. & Yates, S. 2001. *Discourse as data: A guide to analysis*, Sage Publications Ltd.

Williams, L., Kessler, R. R., Cunningham, W. & Jeffries, R. 2000. Strengthening the Case for Pair Programming. *IEEE Software,* 17**,** 19-25.

Williams, L. A. & Kessler, R. R. 2000. All I really need to know about pair programming I learned in kindergarten. *Communications of the ACM,* 43**,** 108-114.