Using Grounded Theory to Study the Human Aspects of Software Engineering

Rashina Hoda
Engineering and Computer Science
Victoria University of Wellington, New Zealand
rashina@ecs.vuw.ac.nz

James Noble
Engineering and Computer Science
Victoria University of Wellington, New Zealand
kjx@ecs.vuw.ac.nz

Stuart Marshall
Engineering and Computer Science
Victoria University of Wellington, New Zealand
stuart@ecs.vuw.ac.nz

ABSTRACT

Grounded Theory (GT) is increasingly being used to study the human aspects of Software Engineering. Unfortunately, the Grounded Theory method is still not widely understood in the Software Engineering discipline. We present an overview of the Grounded Theory method and discuss its use.

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management
; K.6.1 [**Management of Computing and Information Systems**]: Project and People Management

Keywords

Grounded Theory, Software Engineering, Human Aspects

1. INTRODUCTION

Grounded Theory (GT) is a research method developed by sociologists Glaser and Strauss in 1967 [8]. GT allows the researcher to systematically generate theory from data analyzed by constant comparison [6]. GT is particularly suited to study the human aspects of Software Engineering because (a) GT, used as a qualitative research method, allows us to study social interactions and behaviour—a key ingredient in the study of human aspects of Software Engineering and (b) GT uncovers the main concern of the research participants and how they go about resolving it. GT is increasingly being used to study the human aspects of Software Engineering [1, 2, 3, 4, 11, 10, 12, 13]. Unfortunately, the GT method is still not widely understood in the field of Software Engineering. In this paper, we present an overview of the classic GT method and discuss the use of GT as a valuable research method for studying human aspects of Software Engineering.

2. GROUNDED THEORY

Fig. 1 presents an overview of Grounded Theory. Examples of our use of the GT process are available here [9, 10, 11].

HAoSE at OOPSLA/SPLASH'10, October 17–21, 2010, Reno/Tahoe, Nevada, USA.

Initial Literature Review The researcher can start off with a light literature review—enough to carry on a conversation with his participants. The research question is formulated during the process of GT and not as a result of extensive literature review upfront. Extensive literature review can be done later in the process [6].

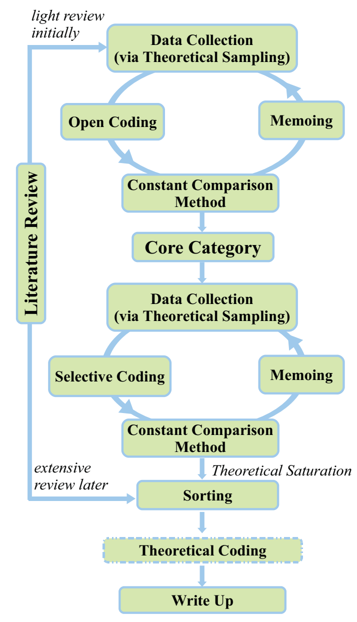

Figure 1: Overview of the Grounded Theory method

Data Collection in the field guided by a process called *Theoretical Sampling* which allows the researcher to collect, code, and analyze the data and then decide what data to collect next [6]. Interviews and observations are a couple of the most popular methods of data collection in GT.

Open Coding is the first step of analysis and starts by collating key points from raw data. These are then assigned a code—a phrase that summaries the key point in 2 or 3 words [5].

Constant Comparison Method The codes arising out of each interview are constantly compared against the codes from the same interview, and those from other interviews and observations. This is GT's *Constant Comparison Method* [5, 6] which is used to group these codes to produce a higher level of abstraction called concepts, and is repeated on these concepts to produce another level of abstraction called a category.

Memoing is the ongoing process of writing theoretical notes throughout the GT process. Memos capture the conceptual links between categories as the researcher notes down their reflections on different categories.

Core Category Several categories emerge as a result of data analysis amd the one that is able to account for most variations in the data and relates meaningfully and easily with other categories is called the *Core Category* [6].

Selective Coding Once the core category is established, the researcher ceases open coding and uses *Selective Coding*—a proceedure where they code for only the core category and those categories that are closely related to the core.

Theoretical Saturation When further data collection and analysis on a particular category leads to a point of diminishing results—no new insight into the category is generated—the category is said to have reached *Theoretical Saturation* [6]. The researcher can then stop collecting data and coding for that category.

Extensive Literature Review As the theory starts to emerge, the researcher can conduct extensive literature review to see how the literature in the field relates to their emerging theory.

Sorting Once the researcher has nearly finished data collection and coding is almost saturated, they can begin arranging the theoretical memos on a conceptual level or *Sorting*. Sorting results in an outline of the theory describing how the different categories relate to the core-category [6] .

Theoretical Coding Glaser lists several common structues of theories or theoretical coding families [7] which can be used as a framework to describe how the categories relate to each other as a hypotheses to be integrated into a theory. This is called *Theoretical Coding*.

Write up The final step in GT is writing up the theory, which follows the theoretical outline generated as a result of sorting and theoretical coding.

3. STUDYING HUMAN ASPECTS USING GT

Several aspects of GT lend themselves postively to the study of human aspects of Software Engineering. In GT, the main source of data collection are interviews conducted with human participants and observations of their interactions and behaviour during their every day software development acitvities. Incidents shared by participants and observed by the researcher become the basis for data analysis in GT as the researcher looks for common patterns among the sets of data. Through several levels of abstraction, the researcher is able to isolate the core category which captures the largest concern of the majority of the participants. Throughout the GT process, the human and social aspects remain the prime focus. Using GT as a research method, the researcher is not only able to discover the main concern of the participants but also the ways in which they go about resolving it. There are several examples of the application of GT to study the human aspects of Software Engineering [1, 2, 3, 4, 11, 10, 12, 13].

Our own research has used GT to study the self-organizing nature of Agile teams [9, 10, 11]. While self-organization is an important theoretical concept in Agile software engineering, how Agile teams self-organize in practice is not well understood. Using GT as a research method, we were able to interview and observe Agile practitioners in the field. As a result of GT analysis, we discovered six informal roles on Agile teams that specifically facilitate self-organization: Mentor, Co-ordinator, Translator, Champion, Promoter, and Terminator [10]. We also identified the partices of Agile teams that make them self-organizing: balancing freedom and responsibility, balancing cross-functionality and specialization, and balancing continuous improvement and iterature pressure [10]. Through the use of classic GT as a research method, we were able to understand and capture the main concern of our participants—becoming a self-organizing Agile teams—and how they go about resolving it—through informal self-organizational roles and practices.

4. REFERENCES

[1] J. Carver. The Impact of Background and Experience on Software Inspections *EMSE*, 9, 259–262, 2004.

[2] C.A. Crabtree, A.F. Norcio. Exploring Language in Software Process Elicitation: A Grounded Theory Approach In *ESEM*, 324–335, 2009.

[3] G. Coleman and R. O'Connor. Using Grounded Theory to Understand Software Process Improvement: A Study of Irish Software Product Companies. *Inf. Softw. Technol.*, 49(6), 654–667, 2007.

[4] B. Dagenais et. al. Moving into a New Software Project Landscape. In *ICSE*, 275–284, 2010.

[5] S Georgieva and G Allan. Best practices in project management through a grounded theory lens. *EJBRM*, 2008.

[6] B Glaser. *Theoretical Sensitivity: Advances in the Methodology of Grounded Theory*. Sociology Press, Mill Valley, CA, 1978.

[7] B Glaser. *The Grounded Theory Perspective III: Theoretical Coding*. Sociology Press, Mill Valley, 2005.

[8] B Glaser and A. L. Strauss. *The Discovery of Grounded Theory*. Aldine, Chicago, 1967.

[9] R. Hoda, J. Noble, S. Marshall. Balancing Acts: Walking the Agile Tightrope. In *CHASE* workshop at ICSE, ACM, 2010.

[10] R. Hoda, J. Noble, S. Marshall. Organizing Self-Organizing Teams. In *ICSE*, 285–294, ACM, 2010.

[11] R. Hoda, J. Noble, S. Marshall. Agility in Context. To appear in *SPLASH/OOPSLA*, 2010.

[12] A. Martin, R. Biddle, and J. Noble. The XP customer role: A grounded theory. In *Proceedings of Agile 2009*, IEEE CS, 2009.

[13] E. Whitworth and R. Biddle. The Social Nature of Agile Teams. In *Agile 2007*, IEEE CS, 2007.