Producing and Interpreting Debug Texts

An Empirical Study of Distributed, Parallel Debugging in Open Source Software Development

Thomas Østerlie
Norwegian University of Science and Technology
Sem Sælands vei 7-9, 7491 Trondheim, Norway
thomas.osterlie@idi.ntnu.no

Abstract. This paper presents preliminary findings from an ethnographic study of distributed, parallel debugging in an open source software (OSS) community. Focusing on the OSS developers' daily activities, I propose the concept of making software debuggable. In so doing, I see a somewhat different story than common narratives of debugging in current OSS research, which describes distributed, parallel debugging as a set of highly cohesive tasks within loosely couple groups. I find that parallel, distributed debugging is rather a closely coupled collective process of producing and interpreting debug texts with high cohesion between the activities of reporting, finding, and understanding bugs.

1 Introduction

Parallel debugging is identified as one of the key characteristics of OSS development processes [1], and "is the site of claims of effectiveness made for [OSS] projects" [2]. While empirical research show that defects are found and corrected rapidly with parallel debugging [3][4], explanations for these findings remain inconclusive. It has been proposed that OSS is more maintainable than commercial software. However, no difference is found in the maintainability between commercial and OSS software [3]. Another proposed explanation is that successful OSS projects exhibit a specific social structure [5]. Yet, research has shown the structure varies among projects and that different successful OSS projects may exhibit different social structures [2].

In my research I seek to explore an explanation to the success of parallel debugging that lies in the everyday activities of debugging. Existing studies of parallel debugging tells us little about what OSS developers do on a day-to-day basis. The key question raised in my research is therefore: what are OSS developers daily activities in parallel debugging?

My research is based on materials collected during ten months ethnographic studies in the Gentoo OSS community. The Gentoo community develops, maintains, and operates a system for distributing and installing third-party OSS on various Unix variants, along with their own GNU/Linux distribution. Gentoo releases its software for parallel debugging by the community as part of a formalized process.

Please use the following format when citing this chapter:
Østerlie, T., 2006, in IFIP International Federation for Information Processing, Volume 203, Open Source Systems, eds. Damiani, E., Fitzgerald, B., Scacchi, W., Scotto, M., Succi, G., (Boston: Springer), pp. 335-336

2 Preliminary findings

Mockus et al. [4] find that "most of the effort in bug fixing is generally in tracking down the source of the problem". I find that tracking down the bug need not be all that simple in practice. It need not be obvious what the bug "really is". Rather, it is subject to interpretation. To make sense of failures reported in bug reports, the developers discuss a number of possible sources for the failure. Of these possible explanations, I find that none are dismissed on conclusive evidence. Instead, alternative explanations for reported failures are made more or less plausible by producing new debug texts, trying to reproduce the bug, and drawing on external texts like installation scripts, source code, documentation, and change logs.

Wherein previous studies seek to explain the success of debugging in OSS as a function of qualities with the software product [3], my observation is that the success of debugging may be found in the daily activities of OSS users and developers. Finding the source of a bug is a process where the person reporting the bug and those trying to understand make the bug debuggable by working together to find relevant pieces of information and producing new debug texts. Making the software debuggable can therefore be interpreted as a collective process including both the person submitting the bug report, those trying to understand and resolve the problem, as well as the tools involved in producing the various debug texts being interpreted. It is by iteratively producing debug texts and extracting pieces of from these texts into meaningful combinations that bugs are made debuggable.

3 References

1. J. Feller and B. Fitzgerald, *Understanding Open Source Software Development* (Pearson Education Limited, Harlow, 2002)
2. K. Crowston and J. Howison, The Social Structure of Free and Open Source Software Development, *First Monday* **10**(2), 2005.
3. J.W. Paulson, G. Succi and A. Eberlein,, An Empirical Study of Open Source and Closed-Source Software Products, *IEEE Transactions on Software Engineering* **30**(4):246-256 (2004).
4. A. Mockus, R.T. Fielding amd J.D. Herbsleb, Two Case Studies of Open Source Software Development: Apache and Mozilla, *Transactions on Software Engineering and Methodology* **11**(3), 309-346 (2002).
5. J.Y. Moon and L. Sproull, The Essence of Distributed Work: The Case of the Linux Kernel, *First Monday* **5**(11), 2000.