Developers' Code Context Models for Change Tasks

Katja Kevic*

Department of Informatics
University of Zurich

1 Motivation

Software developers spend substantial time searching and navigating through code to understand relevant parts of a system for a particular change task [2, 6]. During this process of understanding and then changing code, developers implicitly build *code context models* that consist of the relevant code elements and the relations between these elements, often more generally referred to as task context. Since these models mainly stay implicit in developers' heads and are not persistent [3], developers have to continuously spend a significant amount of their time creating context models for newly assigned change tasks from scratch.

Researchers have suggested that more explicit context models for change tasks[1] can be used to support developers in their work [4]. To form these explicit context models, existing approaches have used methods ranging from a developer manually specifying the context (e.g., [5]) to automatically inferring the context from a developer's interaction with a development environment (e.g., [1]). While these approaches have been shown to support developers with change tasks, little is understood about the implicit code context models that developers build and their characteristics. With a better understanding of the characteristics of developers' code context models, we can help developers in the initial creation of these models for change tasks, saving time and effort.

2 Research

To investigate the characteristics that code context models exhibit for different change tasks, we conducted an exploratory study with twelve developers on three change tasks in open source projects and a second empirical analysis using data sets from several hundreds of change tasks and eighty developers working on open and closed source projects. In particular, we addressed the following three research questions:

(1) What are common characteristics that code context models exhibit?
(2) How do code context models vary based on different definitions of relevance?
(3) How do developers' code context models and navigation behavior vary for different tasks?

* This work was done in collaboration with T. Fritz and C. Bräunlich from UZH, and D. Shepherd and W. Snipes from ABB Research Raleigh and accepted as a full paper to ACM SIGSOFT FSE 2014.

[1] We use the term change task to refer to both modification task and bug.

Amongst other results, our studies show that developers' context models are highly connected, structurally and lexically, that the code navigation can differ substantially by individual even for the same change task, and that developers start change tasks using a combination of search and navigation and then frequently revisit code elements. Based on our findings we inferred design requirements to support developers in the creation of code context models. These design requirements span from the provision of personalised navigation support that combines structural and lexical navigation to tailoring the support based on the change task type. In our work, we discuss these requirements and how they would influence the design of an approach for the creation of code context models.

3 Contributions

In this work, we make the following research contributions:

(1) We identify important observations on the characteristics of code context models based on an exploratory study with 12 developers on three open source projects.
(2) We provide an empirical analysis of the findings on data collected from eighty developers working on a variety of change tasks on open and closed source projects.
(3) We identify design requirements and discuss the design of an approach to support developers in the creation of code context models.

This work represents a substantial step in better understanding developers' code navigation and implicit context models for change tasks that will help to provide better tool support with the potential for real-world impact on the development process, in particular on reducing the time and effort required for change tasks.

In my work on this project I contributed to all three parts, (1), (2) and (3), and focused a lot on the data analysis, the observations and the identification of design requirements.

References

[1] M. Kersten and G. C. Murphy. Using task context to improve programmer productivity. In *Proc. of FSE'06*, 2006.
[2] A. J. Ko, B. A. Myers, M. J. Coblenz, and H. H. Aung. An exploratory study of how developers seek, relate, and collect relevant information during soft. maintenance tasks. *IEEE Trans. on Soft. Eng.*, 32:971–987, 2006.
[3] T. D. LaToza, G. Venolia, and R. DeLine. Maintaining mental models: a study of developer work habits. In *Proc. of ICSE'06*, 2006.
[4] G. C. Murphy, M. Kersten, M. P. Robillard, and D. Čubranić. The emergent structure of development tasks. In *Proc. of ECOOP'05*, pages 33–48, 2005.
[5] M. P. Robillard and G. C. Murphy. Concern graphs: finding and describing concerns using structural program dependencies. In *Proc. of ICSE'02*, 2002.
[6] J. Singer, T. Lethbridge, N. Vinson, and N. Anquetil. An examination of software engineering work practices. In *Proc. of Centre for Adv. Studies on Collaborative Research*, 1997.