SUMMARY OF Ph.D. DISSERTATION

School School of Science for Open and Environmental Systems	Student Identification Number	SURNAME, First name Toshihiko Koju

Title

Design and Implementation of a Framework
for Controlling and Monitoring Program Execution for Debuggers

Abstract

Debugging is a tedious but unavoidable part of the software development process. Unfortunately, basic functionalities provided by traditional debuggers are insufficient to debug increasingly larger and more complex software. Thus, more sophisticated functionalities that can assist software developers in analyzing their target programs more deeply have been proposed. Examples of such functionalities include program slicing and reversible execution.

Debuggers equipped with such sophisticated functionalities need to perform special kinds of tasks of controlling and monitoring program execution. In our research, we propose a new debugger development environment, DbgStar, which provides a basis for these tasks to debugger developers. Our contributions are as follows:

- **Applying SDT to debuggers:** In our research, we apply the technique of SDT (Software Dynamic Translation) to debuggers. SDT is a technique which executes programs on a virtual machine, while simultaneously performing code translations. Using SDT, we can instrument programs (insertions of code for monitoring) (1) during execution, and (2) to only those parts which are actually executed. This alleviates issues in existing approaches such as:
  - **Overhead:** Since only parts, that are actually executed, are instrumented, no additional costs for redundant instrumentations are required. In addition, instrumented code are executed directly on the real CPU. Therefore, overall overhead is relatively low.
  - **Affinity:** Since executables generated by existing compilers are used, modifications to debuggee's source code are basically not required. Also, no specialized hardware is required. Therefore, high affinity with existing software development environment can be achieved.
  - **Flexibility:** The VM allows dynamic adjustments of details of code translations. This means the granularities of monitoring can be changed depending on the debugging situations, which enables minimal monitoring. Therefore, very flexible monitoring can be achieved.
- **Debugger development environment:** Controlling and monitoring of program execution are fundamental tasks for various kinds of debugger's functionalities. However, implementing these tasks is not trivial, since program execution is affected by many different-level elements, such as hardware architectures, OS, and compilers. Therefore, debugger developers can greatly reduce implementation efforts by using DbgStar.

In this paper, we introduce a debugger which was developed using DbgStar. We integrated some sophisticated functionalities to our debugger, such as program slicing and reversible execution. We describe debugging scenarios which show usages of our debugger, using real faults reported for open source programs (ProFTPD, GNU Awk, and Apache HTTP Server). Evaluation results showed that DbgStar satisfied the three perspectives of overhead, affinity, and flexibility.