Design and Implementation for Program Understanding Tool using DWARF2 Debug Information

Tomoya Suzuki (210051)

School of Information Science,
Japan Advanced Institute of Science and Technology

February 13, 2004

Keywords: DWARF2, debug information, XML, CASE tool, data integration.

C language is useful because using C improves significantly development efficiency and portability without degrading much execution time and memory in the case of demanding assembly language though it has some problems such as security. It is important to improve reliability of software which is described in C language and used for fundamental computer systems like OS or language processors. However, since access to fixed address of hardware using addressing operators is permited in C, checking is not thoroughly done in compile-time and run-time. Thus, for reducing bugs in C's code and making the system safer, it is necessary to provide CASE tools which assist efficient development and maintenance.

For efficient development of flexible CASE tools, data integration which defines a common format so that CASE tools can use the same data is supposed and some studies, e.g., PCTE and CDIF have been already done. But they are not widely used because their specifications are complicated.

Some studies, e.g., Sapid, ACML, GCC-XML, JavaML, have proposed to use XML as a data format. They enable developing opened flexible CASE tools effectively. On the other hand, it is difficult them to deal with software such as libraries with no source code and difference of C

processors. For example, Sapid and ACML cannot process GCC extensions such as asm syntax and __attribute__ syntax because they do not deal with all extensions. Thus, they cannot process programs which GCC can. Furthermore if they take care of each GCC version, it will cost very much.

Source code level data integration provides a common format for static source code information such as syntax, type and symbol information. Tt is becoming clearer that using XML as common formats is quite effective to reduce development cost of CASE tools.

However, each of these approaches has its own parser and analyser since they are all source code level in the sense that they extract information from source code.

To solve this problem, we introduce a new approach of applying binary code to CASE tools. There are many tools using binary level information. For example, Purify and PureCoverage are tools for inspecting or modifying binary code not source code. Using binary code information, we can cope with C extension problem because compilers not CASE tools, take care of C extensions, and also we can analyze libraries without their source code.

Therefore, we examine validity of binary level data integration and applicability of binary level information.

Binary codes have debug information, for example of local variables, user-defined types, line numbers, scopes and stack frames, which is primarily used for debuggers. We think that these information is useful for CASE tools.

DWARF2 is one of the debugging information formats, which is supported by a wide variety of language processors (e.g., GCC and GDB). This implies CASE tools using DWARF2 will cover large application area.

On the other hand, debugging information lacks some information such as expressions and sentences.

In this paper, using DWARF2, we designed and implemented a cross referencer as a program understanding tool, and we examined the applicability of debug information for CASE tools by comparison experiments with other cross referencers.

Moreover, to partially solve the problem of information lacking in debug information, we experimentally implemented a hybrid cross referencer of the cross referencer and GNU GLOBAL.

In the result, as long as this experiment, we developed a flexible and practical cross referencer in a man-month. Unfortunately, because of using XML and Ruby, the executing time of the tool is slower than the existing ones.