Performance Measurement/Analysis Tool “mevalet”

KAWAMURA Kantou, SUZUKI Kazuaki, HORIKAWA Takashi, YAMASHITA Toshiaki, SAKAKI Daiya

Abstract

Recently embedded software development has noticeably been increasing in scale and complexity, and reductions in the delivery term and improvements in development efficiency have now become critical management issues. To deal with them, NEC has developed a performance measurement/analysis tool called “mevalet,” which is compatible with Linux running on ARM CPU, and features an added data save function using a miniSD card instead of a data transfer function that uses the network. This tool has improved the development efficiency significantly, making it possible to solve a performance issue that has not previously been solvable however hard we tried. This paper is intended to introduce this tool together with a description of cases in which it was actually applied in in-house embedded software development.

Keywords

performance, performance measurement, analysis, embedding, development environment

1. Introduction

Embedded equipment such as the cellular phone, car-mount devices and home electric appliances is experiencing rapid advancement of networking, improvement of functions, diversification of user needs, shortening of product lifecycles and reduction of product prices. These changes in trends are enhancing the requirement for the embedded software for use in embedded equipment to increase in scale, handle more complicated targets and shorten the delivery term. Thereby reductions in the delivery term and development costs have become critical management issues. In addition, the necessity for the observance of related laws including the PL law and for improved reliability to meet the advance in safety awareness of the users is forcing the software in general to offer a higher quality than before including improvements in ease of use and an increase in processing speed. To meet these challenges, it is required to improve software development productivity based on a full understanding of the development processes in the field, including both distributed and concurrent development.

2. Approach for Preventing Performance-Related Problems

In order to prevent a performance-related problem before it occurs, the software development process should be conscious of implementing performance from upstream. We developed the Performance Engineering Framework (PEF) shown in **Fig. 1** as a technological system to realize this purpose. The PEF is composed of mevalet, which is a performance measurement/diagnosis tool developed using NEC-original technology with performance prediction theory and templates, etc. at its core. In the design phase, performance may be verified in advance by checking the method-level performance requirements and behavior of software using existing software products and past software assets. In the evaluation phase, the actual software is run and measured using mevalet to verify the implementation accuracy and to confirm that the performance requirements verified in the upstream process are met. It also discovers bottlenecks at an early stage, evaluates the performance improvement measures and checks their effectiveness. The performance can be implemented in this way, by checking the performance requirements set in the upstream process at every turning point of the design and development.

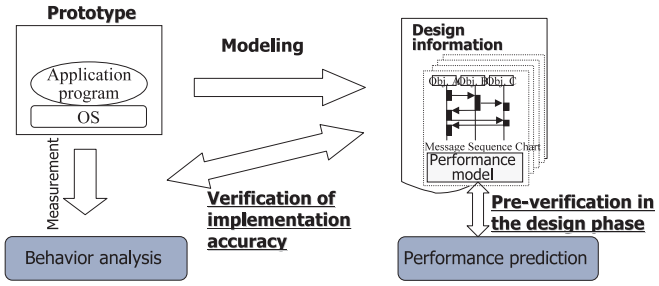

Fig. 1 Performance Engineering Framework.

3. Outline of mevalet, Its Application to Embedded Systems and Its Effects

3.1 Outline of mevalet

Fig. 2 shows the architecture of mevalet. It installs hook points inside the OS of the measurement target embedded system to measure the performance and analyze the behaviors of any application with a low overhead of some tens of nanosec. to µsec. This is done independently from a specific product or language and without the need of modifying the application. The events that can be collected with mevalet include the waiting, start, completion of CPU usage in kernel or user mode, the inter-task interfacing and interrupts.

When the user performs an operation on the PC, mevalet starts collection of events from the kernel and stores them in a buffer. After the measurement, it transmits the collected data to the analysis PC to let it analyze them, and also displays the results on TraceView that is a mevalet-original display screen. The menu displayed by right clicking on mevalet can display information including the total CPU usage time per process and the order of system call generation. For the Java applications, we developed the Java Probe for distinguishing the behavior of Java VM and applications. The Java Probe can automatically be inserted between the components by defining the information for Java Probe insertion in advance. This makes it possible to analyze the behavior and performance of Java applications without modifying the applications.

3.2 Eclipse Compatibility

As the development environment, we also developed the mevalet TraceView plug-in that runs on Eclipse, which is the *de facto* standard. **Fig. 3** shows an example of behavior and performance analyses on Eclipse. This plug-in makes it possible to display the behaviors of an application running on the embedded system as a sequence chart on Eclipse. Additionally, the system behavior can also be displayed integrally by switching the view to per machine, per process, per class or per method even with an embedded system that uses multiple machines, and performance bottleneck analysis with a top-down approach is also available. These user interfaces not only make an advanced performance analysis technology unnecessary and allow anyone to solve expert-level performance-related problems, but also enable us to “visualize” the causal associations between performance-related problems.

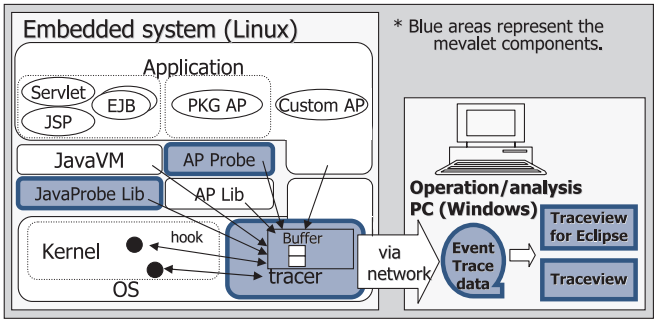

Fig. 2 mevalet architecture.

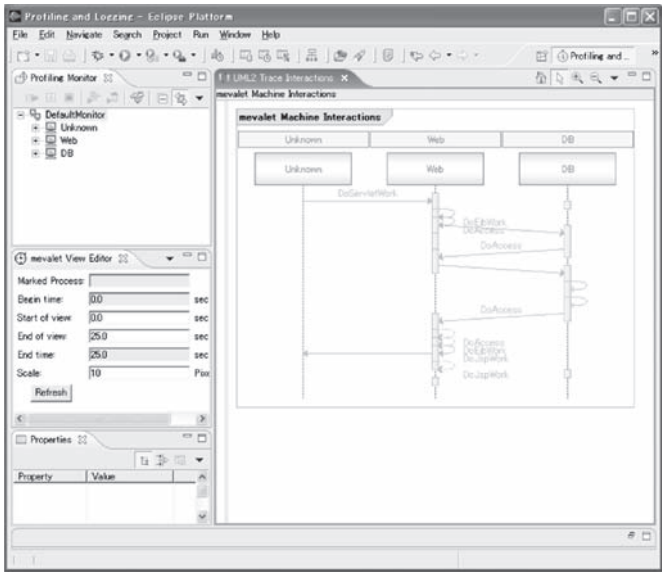

Fig. 3 Example of behavior/performance analyses on Eclipse.

3.3 Application to Development of Linux-Based Embedded Software That Runs on ARM CPU and Its Effects

Since the hardware architectures are not unified in an embedded system, the current mevalet required customization in most cases because evaluation and verification of application are necessary per hardware. Specifically, confirmation is required on the modification of the part dependent on the CPU architecture, the absence of the network for trace data transfer and the method of time acquisition.

Fig. 4 shows details of the modification of mevalet for developing Linux-based embedded software that runs on an ARM CPU. In this case, mevalet is modified to collect the necessary events from the Linux kernel. In addition, the function for collecting trace data in the SD card is added to compensate for the absence of a network communication function. The use of

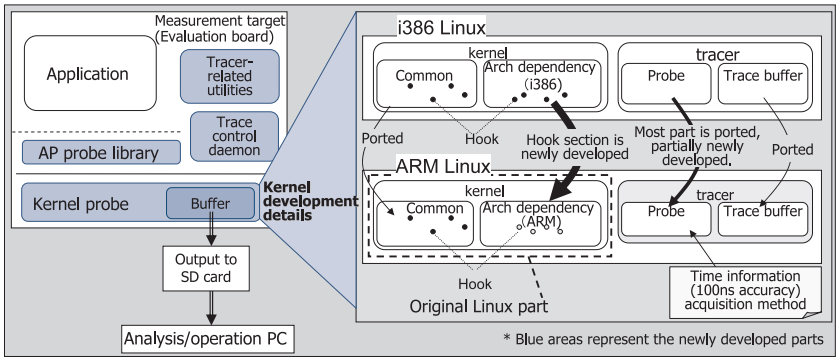

Fig. 4 mevalet modification details.

mevalet in the development of this embedded software has made it possible to execute the primary analysis of performance data quickly and easily. Furthermore, the possibility of intuitive confirmation of threads and operations including interrupts has facilitated detection and identification of problems, significantly improving the productivity in solutions for the issues related to Java performance and those of response degradation due to increased CPU load. The man-hours have also been greatly reduced, for example by enabling the solving of problems, the cause of which has previously been untraceable even after a month of survey work.

4. Present Status, Future Perspectives

We recognized a large number of positive quantitative effects brought about by mevalet in the development of embedded software for Linux running on an ARM CPU, and are thus currently deploying it in all of the development bases of specific departments of the NEC Group inside as well as outside Japan. In the future, we will promote further development by adopting a new mevalet architecture that does not require modification of the OS and by expanding the applicable OS range for use in network and car-mount devices. We will also enhance differentiation of mevalet by packaging various feedback and expertise obtained in-house as a result of its use.

* ARM is a registered trademark of ARM Limited in the UK and other countries.
* Linux is a registered trademark of Linus Torvalds in the USA and other countries.
* Other brand names and product names mentioned in this paper are trademarks or registered trademarks of their respective owners.

Reference

1) "Performance Engineering Framework NO KAIHATSU (Development of Performance Engineering Framework)," NEC GIHO, Vol. 58, No. 3. http://tj.nepas.nec.co.jp/techrep/journal/g05/n03/t0503a01.pdf

Authors' Profiles

KAWAMURA Kantou
Senior Manager,
Software Development Environment Engineering Division,
Systems Software Operations Unit,
NEC Corporation

SUZUKI Kazuaki
Manager,
Software Development Environment Engineering Division,
Systems Software Operations Unit,
NEC Corporation

HORIKAWA Takashi
Senior Principal Researcher,
Common Platform Software Research Laboratories,
NEC Corporation
Member of IPS (Information Processing Society of Japan)

YAMASHITA Toshiaki
Assistant Manager,
Advanced Technology Solution Division,
NEC Informatec Systems, Ltd.

SAKAKI Daiya
Assistant Manager,
Third Solution Business Division,
NEC Software Hokuriku, Ltd.

●The details about this paper can be seen at the following.
Related URL: http://www.nec.co.jp/cced/mevalet/