STUDI PERBANDINGAN REAL TIME JVM (OPEN VIRTUAL MACHINE & JAVA REAL TIME EXTENSION)

M Hendra Herviawan[1], Fazmah Arief Yulianto[2], -[3]

[1]Teknik Informatika, Fakultas Teknik Informatika, Universitas Telkom

Abstrak

Dalam tugas akhir ini akan di lakukan pengujian kecepatan, keakuratan, keterprediksian dan kesesuaian atas implementasi Real Time Specification forJava (RTSJ). Pengujian akan dilakukan terhadap Open Virtual Machine (OVM) yang dikembangkan oleh Purdu University serta Java Real Time Extension (JRATE) yang di kembangkan oleh Washington University. Pengujian akan dititikberatkan kepada keterpredikasian dari OVM dan JRATE. Selanjutnya akan dilakukan juga pengujian terhadap unjuk kerja dari OVM, JRATE dibandingkan dengan Sun JVM 1.4.2 melalui simulasi automatic cruise control system (ACCS). Hasil dari pengujian menunjukan bahwa JRATE memiliki kecepatan yang relative sama dengan Sun JVM 1.4.2, sedangkan OVM lebih lambat sampai dengan dua kali dibandingkan JRATE dan Sun JVM 1.4.2. OVM dan jRate memiliki keakuratan yang relatif sama. Apabila melihat dari keterprediksiannya, JRATE lebih dapat diprediksi dibandingkan dengan OVM untuk simulasi yang membutuhkan keakuratan sedangkan untuk simulasi yang membutuhkan kecepatan OVM dan jRate memiliki keterprediksian yang relatif sama

Kata Kunci : Real time system, Real time speciation for java, RTSJ, JSR-1, real time JVM.

Abstract

In this final project will be tested speed, accurasy, predictability and compatibility implementation of Real Time Specification for Java (RTSJ). Testing will be conducted to Open Virtual Machine (OVM) from Purdue University and Java Real Time Extension (JRATE) from Washington University. Testing will be pointed to predictability of OVM and JRATE. Furthermore testing will be conducted to OVM, JRATE compared to Sun JVM 1.4.2 at automatic cruise control system (ACCS). Result from simulation show that JRATE have speed equal to Sun JVM 1.4.2, mean while OVM two time more slow then JRATE and Sun JVM 1.4.2. OVM and JRATE have same accurasy. If looking at predictability, JRATE more predictive then OVM for simulation that need of accurasy, for simulation that need speed, OVM and JRATE relative have same predictability.

Keywords : : Real time system, Real time speciation for java, RTSJ, JSR-1, real time JVM.

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Real time specification for java (RTSJ) yang dirancang oleh *java expert group*, memungkinkan bahasa pemrograman *java* digunakan sebagai bahasa pemrograman dalam membuat *real time system*. RTSJ bukanlah pengembangan dari *java*, RTSJ hanyalah kumpulan *packet – packet* yang dapat digunakan untuk membuat *real time system* pada bahasa pemrograman *java*. RTSJ sangat berbeda dengan *java* specification yang lain, untuk dapat mengimplementasikan RTSJ, *Java Virtual Mechine* (JVM) harus dimodifikasi agar dapat melakukan *real time programming*.

Dalam mengimplementasikan RTSJ, masing – masing memodifikasi JVM dengan pendekatan yang berbeda, tetapi dengan satu tujuan membuat *real time* JVM. Berlandaskan atas beragamnya pendekatan yang ditempuh untuk membuat *real time* JVM, diperlukan studi perbandingan yang bertujuan memberikan gambaran kelebihan dan kekurangan tiap – tiap implementasi RTSJ. Dengan harapan dapat digunakan sebagai acuan pemilihan RTSJ.

1.2 Perumusan Masalah

Permasalahan yang akan diuraikan pada Tugas Akhir ini meliputi :

a. Perbandingan *real time* JVM antara OVM JVM dengan jRate JVM dari sudut Kecepatan, Keterprediksian, Keakuratan dan Kesesuaian dengan *Real time specification for java* (RTSJ).
b. Perbandingan pada point (a) dilihat dari :
   - Waktu pengalokasian m*emory* pada turunan dari kelas *ScopedMemory*.
   - Waktu melakukan perpindahan eksekusi *thread*, baik saat melakukan *yeilding* satu dengan lainnya ataupun saat dua thread mencoba memasuki suatu *synchronizing code* .
   - Keakuratan waktu tidur sampai dibangunkan kembali pada thread yang ditidurkan dengan fungsi *waitForNextPeriod()*.
   - Keakuratan pengeksekusian triger dari kelas *OneShotTimer*.
   - Keakuratan pengeksekusian triger yang dilakukan secara periodik dari kelas *PeriodicTimer*.
c. Perbandingan antara *real time* JVM (OVMJ, jRate) dengan *non real time* JVM (Sun Microsystem 1.4.2 JVM) .
d. Perbandingan pada point (c) dilakukan pada simulasi :
   - *Automobile cruise control system (ACCS).*

1.3 Tujuan Tugas Akhir

a. Membandingkan antara OVM JVM dengan jRate JVM untuk mengetahui yang terbaik diantara keduanya dilihat dari sudut kecepatan, keterprediksian, keakuratan dan Kesesuaian dengan *Real time specification for java* (RTSJ).
b. Membandingkan antara *real time* JVM (OVM, jRate) dengan *standar* JVM (Sun Microsystem 1.4.2 JVM) untuk mengetahui sejauh mana perbedaan kecepatan, keakuratan dan keterprediksian antara *real time* JVM dengan *non real time* JVM.

1.4 Batasan Masalah

Dalam pembuatan Tugas Akhir, penulis membatasi beberapa hal yaitu :

- *Real time* JVM yang dibandingkan adalah OVM dan jRate.
- *Standar* JVM yang digunakan sebagai pembanding adalah Sun Microsystem 1.4.2 JVM.
- *Operating system* yang digunakan sebagai media pengujian adalah Linux dengan *kernel* 2.6 .
- Pengujian dilakukan pada *perconal computer (*PC*)* dengan processor *x86*.
- Kendaraan roda empat yang disimulasikan mengunakan system *transmisi automatic*.
- Kecepatan maximal *Automobile cruise control system (ACCS)* adalah 300 km/jam.
- Pengujian *Automobile cruise control system (ACCS)* dilakukan pada jalan lurus dan mendatar.
- Tidak melakukan perbandingan compiller.

1.5 Metodologi Penyelesaian Masalah

Metodologi yang digunakan untuk menyelesaikan masalah dalam Tugas Akhir ini adalah :

1. Studi Literatur
   Studi literatur dari beberapa buku, jurnal, artikel yang membahas *real time system*, bahasa pemrograman java, *architecture java virtual machine*, *real time specification for java* (RTSJ), *architecture real time java virtual machine*, *automobile cruise control system* (ACCS), Operating system Linux, Linux Kernel 2.6.
2. Analisa Kebutuhan dan Perancangan Sistem
   Analisa masalah dan kebutuhan perangkat lunak yaitu analisa metode perbandingan antara OVM dengan jRate dan metode perbandingan antara *real time* JVM dan *standar* JVM. Berisi juga *Active diagram* dari analisa kebutuhan.
3. Implementasi
   Pembuatan program *bechmark* mengunakan RTSJ untuk membandingkan OVM dan jRate, pembuatan program *benchmark* mengunakan RTSJ untuk dijalankan pada *real time* JVM dan *standar* java untuk dijalankan pada standar JVM.
4. Pengujian
   Pengujian real time JVM dengan menguji feature – feature RTSJ seperti *scoped memory*, *real time thread*, *timer*, *asynchronous event* untuk melihat kecepatan, keterprediksian, keakuratan dan Kesesuaian dengan *Real time specification for java* (RTSJ). Pengujian *real time* JVM dibandingkan dengan *standar* JVM melalui pengujian *real time system automobile cruise control system* (ACCS), untuk melihat perbedaan antara real time JVM dan standar JVM.

5. Analisis
   Melakukan analisis dari hasil pengujian sehingga didapatkan kesimpulan implementasi RTSJ yang terbaik dan perbedaan antara *real time* JVM dan *standar* JVM.
6. Kesimpulan dan Saran
   Mendapatkan kesimpulan dan saran untuk pengembangan selanjutnya.

1.6 Sistematika Penulisan

Sistematika dari Penulisan Tugas Akhir ini adalah sebagai berikut :

BAB I PENDAHULUAN

Menguraikan mengenai latar belakang dari sistem yang akan dibangun, perumusan masalah yang akan dianalisa, tujuan dari pembuatan sistem ini, pembatasan dari masalah yang terjadi, menentukan metodologi pemecahan serta sistematika penulisan.

BAB II LANDASAN TEORI

Mengemukakan mengenai teori–teori yang mendukung dan mendasari penulisan tugas akhir ini seperti real time system, bahasa pemrograman java, real time specification for java dan lainnya.

BAB III ANALISA KEBUTUHAN DAN PERANCANGAN SIMULASI

Perancangan sistem pengujian dan perencanaan menggunakan Sequence diagram, use case, data flow diagram.

BAB IV SIMULASI DAN ANALISA

Analisa terhadap data yang diperoleh dari hasil percobaan yang menunjukan kecepatan, keterprediksian, keakuratan dan Kesesuaian dengan *Real time specification for java* (RTSJ).

BAB V KESIMPULAN DAN SARAN

Kesimpulan yang dapat diambil dari seluruh rangkaian penelitian yang dilakukan dan saran untuk pengembangan lebih lanjut.

BAB IV
KESIMPULAN DAN SARAN

5.1 Kesimpulan

a. Performansi jRate dua kali lebih cepat dibandingkan OVM untuk simulasi *allocation time test, thread creation test* dan *thread switching test.* Untuk ketiga simulasi di atas OVM dan jRate menghasilkan variasi perolehan nilai yang cukup kecil sesuai dengan yang dispecifikasikan dalam *Real time specification for java* (RTSJ).
b. Untuk simulasi *thread periodic test* dan *timer test* jRate menghasilkan delay yang relatif konstan sebesar 0,9 *millisecond* dalam setiap simulasi, sedangkan ovm menghasilkan delay negatif (*timer* dijalankan sebelum waktunya) yang besarnya berkisar antara 0,000446 sampai dengan 0,519916 *millisecond* untuk simulasi *thread periodic test* dan *periodic timer test*. OVM menghasilkan variasi perolehan nilai yang cukup tinggi dibandingkan dengan JRATE untuk kedua simulasi diatas.
c. JRATE belum sepenuhnya mengimplementasikan *Real time specification for java* (RTSJ) terbukti dengan belum diimplementasikannya kelas *javax.realtime.OneShootTimer*. OVM telah sepenuhnya mengimplementasikan *Real time specification for java* (RTSJ).
d. JRATE dan OVM menghasilkan keterprediksian yang lebih baik di bandingkan dengan Sun JVM 1.4.2 pada simulasi *Automatic cruise control system* (ACCS) sesuai dengan yang dispecifikasikan dalam *Real time specification for java* (RTSJ).
e. JRATE dan Sun JVM 1.4.2 memiliki waktu penyelesean simulasi *Automatic cruise control system* (ACCS) yang relatif sama, sedangkan OVM dua kali lebih lama dibandingkan JRATE dan Sun JVM 1.4.2.

5.2 Saran

a. Penelitian ini dapat dilanjutkan dengan melibatkan *commercial real time java virtual machine.*
b. Pembuatan *spesifikasi* yang mengatur fungtionalitas *networking* pada *real time JVM* .
c. Peninjauan terhadap pengunaan *Real time specification for java* (RTSJ) dan *java 2 enterprise edition* (J2EE) dengan tujuan menciptakan applikasi *real time system* yang terdistribusi.
d. Peninjauan terhadap pengunaan *Real time specification for java* (RTSJ) dalam pembuatan *real time system* pada komputer yang memiliki resource besar (*main frame*).
e. Peninjauan terhadap pengunaan *Real time specification for java* (RTSJ) dalam pembuatan *real time system* yang dijalankan bersama – sama secara *cluster.*

DAFTAR PUSTAKA

[1] http://www.rtj.org/rtsj.html, September 2006.

[2] http://java.sun.com/j2se, September 2006.

[3] http://www.ovmj.org/ovmj.html, September 2006.

[4] http://jrate.sourceforge.net/overview.html, September 2006

[5] Bruce powelDouglass, "Doing Hard Time: Developing Real-Time Systems with UML, Objects, Frameworks, and Patterns", Addison Wesley.

[6] Kevin L. Mills, "An Experimental Evaluation of Specification Techniques for Improving Functional Testing",National Institute of Standards and Technology Gaithersburg, Maryland 20899.

[7] M.Teresa Higuare Taledano, Valeria Issarny, " Java Embedded Real-Time Systems an Overview of Existing Solutions", Campus de Beaulie 35042 Rannes cedex, France.

[8] Juan A. Colmenares, Shruti Gorappa, Mark Panahi, Raymond Klefstad, " A Component Framework for Real-time Java", Department of Electrical Engineering and Computer Science University of California, Irvine, CA 92697, USA.

[9] HSIN-TA CHIAO, SCOTT HSU-JING KAO, YUE-SHAN CHANG,SHEN-TZAY HUANG,SHYAN-MING YUAN, " Experience in Building a Real-Time Extension Library for JavaDepartment of Computer and Information Science National Chiao Tung University Hsinchu, 300 Taiwan.

[10] Krzysztof Palacz, Jan Vitek, " Java Subtype Tests in Real-Time", Universities of Purdue, USA.

[11] Jeremy Manson, Jason Baker, Antonio Cunei, Suresh Jagannathan, Marek Prochazka, Bin Xin, Jan Vitek,"Preemptible Atomic Regions for Real-time Java",Universities of purdue.

[12] Almut Herzog, Nahid Shahmehri, "Problems Running Untrusted Services as Java Threads", Dept. of Computer and Information Science Link¨oping University, Sweden.

[13] OVM Team, “timer.c”, OVM source code.

[14] JRATE Team, “HighResolutionClock.cc”, JRATE source code.