

Editorial

Liebe Leserin, lieber Leser,

trotz der diesjährigen Rekordhitze hat das aicas-Team alles gegeben, um Ihnen noch bessere Entwicklungswerkzeuge als bisher anbieten zu können.

Nachdem JamaicaVM seit Jahren im Bereich echtzeitfähiger Java Lösungen die Technologie anführt, wurde bei der neuen Version 2.2 besonderes Augenmerk auf die Bedienbarkeit und Codegröße gelegt.

Insbesondere das Eclipse Plugin für Jamaica hat bei unseren Kunden starken Anklang gefunden und wurde aufgrund Ihres zahlreichen Feedbacks noch besser in die grafische Entwicklungsumgebung integriert.

Wir haben auch mit dieser neuen Ausgabe der aicas news eine Reihe interessanter Artikel für Sie zusammengestellt und wünschen Ihnen eine angenehme Lektüre.

Ihr

Andy Walter

JamaicaVM 2.2

Am 13. Oktober 2003 erscheint JamaicaVM 2.2

JamaicaVM wurde vor allem im Bereich der Java Standard-Bibliothek sowie durch Optimierungen an der Virtual Machine (VM) selbst und an den Tools verbessert.

Neu ist die Unterstützung von Remote Method Invocation (RMI). RMI ermöglicht den entfernten Methodenaufruf und bildet die Grundlage für verteilte Systeme in Java. Die in JamaicaVM benutzte Standardbibliothek wurde auf den aktuellen Stand von GNU Classpath gebracht. Bei den erweiterten Klassenbibliotheken unterstützt JamaicaVM für die Betriebssysteme Linux, NetOS und VxWorks das Package javax.comm. Dieses API erlaubt das Programmieren von seriellen Schnittstellen.

Die JamaicaVM zeichnet sich durch hohe Leistung bei geringem Ressourcenbedarf aus. Dadurch ist sie speziell für eingebettete Systeme geeignet. In der neuen Version wurde der Byte-Code Interpreter verbessert und die Zeiten zum Laden von

Klassen verringert. Durch weitere Optimierungen interner Strukturen konnte der Speicherbedarf der VM um bis zu 50% verringert werden.

Neben der eigentlichen VM sind in der Distribution eine Reihe von Tools enthalten. Diese erlauben es, eine Java-Applikation optimal an das gegebene Zielsystem (vorhandener Speicherplatz und Prozessor) anzupassen. Der Profiler, ein Tool zur Analyse des Laufzeitverhaltens, wurde erweitert und sammelt nun detaillierte Informationen zu Methodenaufrufen, der Anzahl Threads, und Klassenabhängigkeiten, die durch die Verwendung von Reflection entstehen. Über Reflection referenzierte Klassen können daher automatisch in eine Applikation miteingebaut werden. Die bisher nötige manuelle Konfiguration wird dadurch stark erleichtert.

Unter Windows als Entwickler-Plattform steht jetzt auch eine Windows-Version zur Verfügung. Mehr dazu im Artikel *Neue JamaicaVM für Windows* auf Seite 4. (rs)

Neuigkeiten

Update-Site für Eclipse-Plugin

Das JamaicaVM Plugin für die Eclipse Entwicklungsumgebung (www.eclipse.org) wurde um eine Update-Site erweitert. Dies ermöglicht das einfache Herunterladen und Installieren des Plugins mittels Eclipse selbst. Neue Updates werden ebenfalls auf diese Weise verfügbar gemacht. Für weitere Informationen bezüglich nötiger Konfiguration in Eclipse siehe: <http://www.aicas.com/eclipse.html>.

Distributor in Benelux

Mind NV Belgium hat eine Vertriebspartnerschaft mit der aicas GmbH geschlossen. Die JamaicaVM Echtzeit-Java Technologie und Minds Consulting, Embedded Linux und eCOS Dienstleistungen ergänzen sich dabei perfekt.

Partnership für Weltraum-systems

Astrium EADS ist mit der aicas GmbH eine Entwicklungspartnerschaft eingegangen um sich gemeinsam für Java Lösungen in eingebetteten Weltraum-

anwendungen einzusetzen. Beide Partner entwickelten im ESA-Projekt AERO die AERO-VM basierend auf JamaicaVM.

Japanischer Markt adressiert

aicas weitet seinen Tätigkeitsbereich aus und bietet seine Produkte nun auch in Fernost an. Die Website wurde um einen speziellen japanischen Bereich unter www.aicas.com/japan erweitert. Von Japan aus ist aicas über die Free Phone Nummer 0066.3.384.9014 erreichbar.

Echtzeit und die Real-Time Specification for Java

Dieser Artikel beschreibt die Problematik des Einsatzes von Java-Technologie in Echtzeitsystemen, die Möglichkeiten durch die Real-Time Specification for Java und die Implementierung am Beispiel der JamaicaVM.

Der Erfolg von Java-Technologie basiert auf den Vorteilen wie höhere Produktivität und Sicherheit. Auch kritische Anwendungen, etwa in Automobil- und Flugzeugsteuerungen, profitieren hiervon. Allerdings gibt es auch Nachteile bisheriger Java-Implementierungen, die einen Einsatz in kritischen Systemen verhindern. An erster Stelle steht dabei der Mangel an Echtzeitfähigkeit.

Das größte Problem: Garbage Collection

Einer der größten Vorteile von Java-Technologie, die sichere Speicherverwaltung durch einen automatischen Garbage Collector, ist auch das größte Problem für den Einsatz in Echtzeitsystemen. Der Garbage Collector ist jedoch als Basis für die Sicherheitsmechanismen zwingend erforderlich.

Der Garbage Collector in einem klassischen Java-System kann alle Anwenderthreads zu unvorhersehbaren Zeitpunkten und für eine unvorhersehbare Dauer anhalten. Dies führt zu Pausen, wie sie in Bild 1 dargestellt sind. Diese

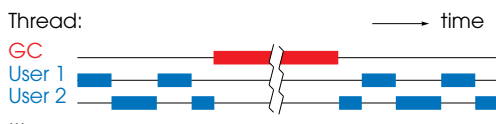


Bild 1: Anwendungsthreads, werden durch Garbage Collector Thread unterbrochen.

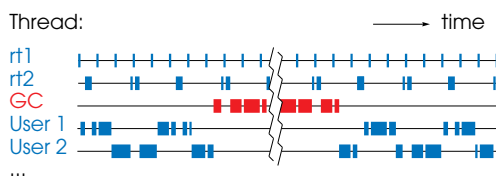


Bild 2: RealTimeThreads in der RTSJ sind nicht von GC Unterbrechungen betroffen.

Pausen machen die Vorhersage des Zeitverhaltens der Anwendung unmöglich. Um Echtzeitprogrammierung zu ermöglichen, muss daher eine Möglichkeit geschaffen werden, diese Pausen zu vermeiden, oder es muss ein deterministischer, inkrementeller Garbage Collector eingesetzt werden, der keine unvorherseh-

baren Unterbrechungen der Anwendung verursacht.

Echtzeitprogrammierung mit der RTSJ

Das Ziel der Real-Time Specification for Java ist die Erweiterung der Java Sprachdefinition und der Standardbibliotheken um Echtzeitthreads, also Threads deren Ausführungszeit bestimmten zeitlichen Kriterien unterliegen muss. Dabei ist diese Spezifikation kompatibel zu unterschiedlichen Java-Umgebungen und rückwärtskompatibel zu bestehenden, nicht zeitkritischen Java-Anwendungen.

Die wichtigsten Verbesserungen durch die RTSJ betreffen die folgenden sieben Bereiche:

- Thread Scheduling
- Speicherverwaltung
- Synchronisation
- Asynchrone Ereignisse
- Asynchroner Kontrollfluss
- Threadterminierung
- Physikalischer Speicher

Die RTSJ umfasst damit auch Funktionen, die keinen direkten Zusammenhang mit zeitkritischen Anwendungen haben, die jedoch in vielen eingebetteten Echtzeitsystemen von großer Bedeutung sind, etwa direkter Speicherzugriff oder asynchrone Mechanismen.

Thread Scheduling: Um die Entwicklung von Echtzeitsoftware zu ermöglichen, selbst wenn ein Garbage Collector Thread die Ausführung der Anwendungsthreads in unvorhersehbarer Weise unterbrechen kann, werden neue Threadklassen RealtimeThread und NoHeap-RealtimeThread definiert, die von den Garbage Collector Unterbrechungen nicht oder weniger stark betroffen sind. Diese Threads können mit 28 neuen Prioritäten versehen werden, die logisch "höher" sind als die des Garbage Collector Threads. Bild 2 illustriert dies.

Speicherverwaltung: Damit Echt-

zeitthreads nicht vom Garbage Collector unterbrochen werden, müssen sie Speicherbereiche verwenden, die nicht unter der Kontrolle des Garbage Collectors liegen. Mit ImmortalMemory und ScopedMemory gibt es solche Bereiche. Dies führt natürlich dazu, dass die Vorteile der dynamischen Speicherverwaltung nicht voll genutzt werden können.

Synchronisation: In Echtzeitsystemen mit Threads unterschiedlicher Priorität muss Prioritätsinversion vermieden werden. Prioritätsinversion tritt auf, wenn ein Thread hoher Priorität auf einen Monitor wartet, der von einem Thread niedriger Priorität gehalten wird. Die RTSJ bietet hierfür Alternativen wie Prioritätsvererbung und Priority Ceiling.

Die RTSJ bietet somit leistungsfähige Funktionen, die die Entwicklung von zeitkritischen Anwendungen ermöglichen. Bild 3 zeigt, wie die RTSJ in der Praxis verwendet werden kann. Hier wird ein periodischer Thread gestartet, der alle 20ms aktiv wird und eine Ausgabe auf die Konsole schreibt. Dazu wird ein RealtimeThread gestartet. Die Priorität und die Periode des Threads müssen dafür angegeben werden. Ein Aufruf von `waitForNextPeriod()` legt den Thread nach jeder Periode schlafen, bis die jeweils nächste Periode beginnt. Eine Einführung in die RTSJ mit vielen weiteren Beispielen findet sich etwa im Buch von Peter Dibble(4).

Die RTSJ bietet jedoch nicht nur Lösungen für die Echtzeitprogrammierung, sondern bringt auch neue Probleme für den Entwickler. Die Anwendung muss strikt in zwei Hälften aufgeteilt werden: In einen Echtzeit- und einen Nicht-Echtzeit-Teil. Dabei ist die Kommunikation zwischen diesen Teilen stark eingeschränkt: Echtzeit-Threads können keine Speicheranweisungen durchführen, wie etwa die Allokation auf dem normalen Heap, und die Synchronisation zwischen beiden Teilen ist stark eingeschränkt,

```

/* Periodic thread in Java: */
import javax.realtime.*;
public class HelloRT {
    public static void main(String [] args) {
        /* priority for new thread: min+10 */
        int pri = PriorityScheduler
            .instance()
            .getMinPriority() + 10;
        PriorityParameters prp =
            new PriorityParameters(pri);
        /* period: 20ms */
        RelativeTime period =
            new RelativeTime(20 /* ms */,
                0 /* ns */);
        /* release parameters for periodic thread: */
        PeriodicParameters perp =
            new PeriodicParameters
                (null,period,null,null,null,null);
        /* create periodic thread: */
        RealtimeThread rt= new RealtimeThread(prp,
            perp) {
            public void run() {
                int n=1;
                while (waitForNextPeriod() && (n<100)) {
                    System.out.println("Hello "+n);
                    n++;
                }
            }
        };
        /* start periodic thread: */
        rt.start();
    }
}

```

Bild 3: Beispielprogramm zum starten eines periodischen RealtimeThreads.

damit Echtzeitthreads nicht durch den Garbage Collector unterbrochen werden.

Echtzeit Garbage Collection

In einem System, das echtzeit Garbage Collection unterstützt, ist diese strikte Unterscheidung von Echtzeit- und Nicht-Echtzeit-Threads nicht nötig. Die strikte Aufteilung einer Anwendung in zwei Teile ist damit nicht mehr erforderlich. Threads werden allein nach Ihrer Priorität aktiviert. Bild 4 illustriert solch ein System. Die JamaicaVM ist eine solche Java-Implementierung die Echtzeitverhalten für alle Threads bietet.

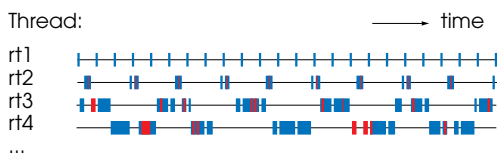


Bild 4: Echtzeit Garbage Collection: Alle Threads sind Echtzeitthreads.

Der echtzeit Garbage Collector erledigt seine Arbeit auf vorher-sagbare Weise innerhalb der Anwendungsthreads immer dann, wenn Speicher alloziert wird. Diese Arbeit muss jederzeit unterbrechbar sein, so dass andere Threads aktiv werden und dringendere Aufgaben erledigen können.

Die Implementierung eines solchen echtzeit Garbage Collectors muss zunächst eine Reihe an technischen Herausforderungen lösen. Insbesondere muss die Garbage Collector Aktivität in sehr kleine, einzelne Arbeitseinheiten aufgeteilt werden. Bei JamaicaVM besteht eine solche Arbeitseinheit aus dem Aufräumen von einem minimalen Speicherbereich der Größe 32 Bytes. Bei jeder Allokation bezahlt der Allokierende Thread für den Speicher durch eine bestimmte Anzahl an Arbeitseinheiten für den Garbage Collector. Die dafür nötige Zeit ist analysierbar, so dass dies sogar in zeitkritischen

Codeteilen möglich wird.

Zusammenfassung

Obwohl es eine Reihe an technischen Herausforderungen gab, die gelöst werden mussten, gibt es mit der Real-Time Specification for Java jetzt einen Standard, der die Entwicklung von zeitkritischer Software auch in der Java Programmiersprache ermöglicht. Dadurch wird diese erfolgreiche Programmiersprache auch für eingebettete und Echtzeitsysteme nutzbar.

Noch größer sind die Vorteile jedoch, wenn die Echtzeitfähigkeit nicht als bloßes "Add-On" einer klassischen Java-Implementierung hinzugefügt wurde, sondern wenn die Implementierung selbst schon echtzeitfähig ist und so die Einschränkungen der RTSJ umgeht.

Damit kann der Entwickler voll von den Vorteilen profitieren, die die Java-Technologie in anderen Anwendungsbereichen so populär gemacht haben.

Ausblick

Mehr Technologiegrundlagen zu folgenden Themen stellen wir Ihnen in der nächsten Ausgabe vor:

Dynamisches Klassenladen

Die dynamischen Fähigkeiten in eingebetteten Anwendungen richtig nutzen. (fs)

- (1) Gregory Bollella (Editor): "The Real-Time Specification for Java", Addison-Wesley, 2000. Die aktuelle Version der Spezifikation ist online unter www.rtfj.org erreichbar.
- (2) The Real-Time for Java Expert Group (RTJEG), www.rtfj.org
- (3) The Java Community Process, www.jcp.org
- (4) Peter C. Dibble: Real-Time Java Platform Programming, Prentice Hall, 2002

Kurse und Vorträge

Realtime Java Ausbildung am Forschungszentrum Informatik

Das Autorisierte Java Center (AJC) am FZI erweiterte sein Kursangebot für Real-time und Embedded Java auf Basis von Kundenanforderungen und Projekterfahrungen. Ein neu konzipierter Kurs (REJ017) umfasst neben der Einführung in RTSJ auf Basis der JamaicaVM auch dem Umgang mit der UML und deren spezifischen Erweiterungen für Echtzeitanwendungen. Der darin enthaltene Workshop festigt die Fähigkeiten mit einem praxisnahen Anwendungsbeispiel, das die gesamte Entwicklungskette von Anforderungsanalyse über Design bis hin zu Implementierung und dem Entwurf von Tests erprobt.

Kontakt: judt@fzi.de, www.swf.fzi.de

Java in Wireless und Embedded Umgebungen

6.-10. Oktober 2003, 9.-13. Februar 2004.

5-tägige Schulung. Aus dem Inhalt: Editionen und Profile; Anforderungen an Embedded Platforms, Knackpunkte: Java-Umgebung und Java-Anwendung; Wireless- und Embedded-Szenarien; Echtzeit-OS, VM und Tools; Java und Echtzeit; Java und Performance; Java und Codegröße. Dieser Kurs wird zusammen mit rpCube und Microconsult angeboten. Kontakt: info@aicas.com

Kompression spart Zeit und Geld

jaccelerator reduziert Flash-Speicherbedarf und Übertragungszeit

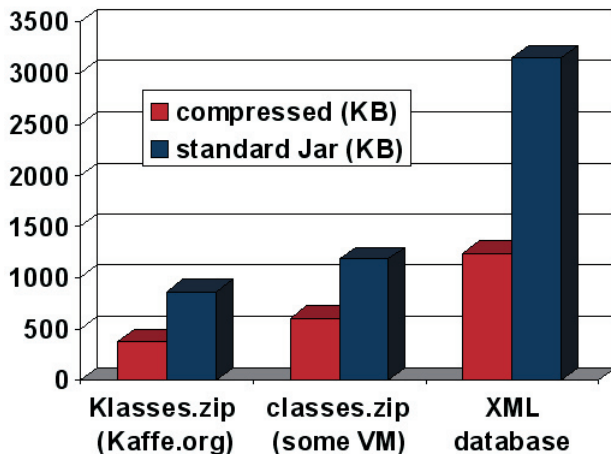
jaccelerator komprimiert Java-Anwendungen 2-3mal kompakter als das JAR-Format. Anwendungsgebiete finden sich überall dort, wo Java-Anwendungen teuren (Flash-) Speicher belegen oder über langsame und kostenintensive Funkverbindungen übertragen werden müssen: Mobiltele-

fone und PDAs, Automotive-Infotainment, Interactive TV-MHP etc.

Je nach Anwendungsszenario geschieht die Kompression als Teil des Entwicklungsprozesses, "on-the-fly" auf einem Application bzw. Provisioning-Server, oder "in-system" direkt im Zielsystem.

Die Dekompression erfolgt zur Ladezeit der Anwendung völlig transparent durch einen erweiterten Classloader der Virtual Machine.

Neben dem reinen Java-Code komprimiert jaccelerator (optional) auch die häufig in Anwendungen enthaltenen Grafikdateien im PNG-Format.



Jaccelerator Kompressionsraten

Neue JamaicaVM für Windows

Keine zusätzlichen Tools nötig.

In JamaicaVM Version 2.2 wurde die Unterstützung für Windows als Entwicklungsplattform verbessert. Im Gegensatz zu früheren Versionen ist es nicht mehr nötig, zusätzliche Tools auf dem Windowsrechner zu installieren.

Diese Version unterstützt nun auch die Code-Erzeugung für Windows-Konsole-Applikationen als Zielplattform sowie die Plattformen VxWorks/PowerPC und VxWorks/x86 von Wind River und NetOS/Net+ARM von NetSilicon. Für diese Zielsysteme werden Native-Thread-, Dateisystem-Operationen-, Netzwerk-Funktionen und die serielle Kommunikation bereits unterstützt.

Das JamaicaVM Plugin für Eclipse für Windows ist auch verfügbar und kann auf der aicas Web-Seite heruntergeladen werden. Die Verwendung von Eclipse erleichtert die Entwicklung von Java-Programmen unter Windows. (tr)

Release 4.0 von RCE und BDE

Die Revision Control Engine und die Byte Differencing Engine.

aicas GmbH folgt den Anforderungen der RCE und BDE Kunden mit einer neuen Release im Oktober.

RCE bietet Revisionskontrolle hoher Qualität, sowohl mit einer grafischen Oberfläche (VRCE), als auch einer API für die Einbindung in industrielle Produkte.

Die neue Version bietet ein schnelleres Streaming Interface. Ein- und Ausgabeströme können nun direkt an RCE übergeben werden an Stelle von Files oder Filedeskriptoren. Dies führt zu deutlich verbesserter Performance in Netzwerkumgebungen.

Eine aufpolierte grafische Oberfläche erhöht die Benutzbarkeit.

BDE, eine Bibliothek für extrem kompakte Deltakomprimierung, unterstützt auch das Streaming Interface und wird dadurch ideal für kleine, tragbare Geräte. (jjh)

jaccelerator ist RTOS-unabhängig, läuft mit J2ME CLC und CLDC, J2SE und natürlich der JamaicaVM. Optional ist die Jamaica Virtual Machine auch mit bereits integrierter jaccelerator-Unterstützung von aicas erhältlich.

Rolf Matzner, RP-Cube / (aw)

Veranstaltungen

GSOFT, Noordwijk

14.-16. Oktober. Der zweite Galileo Software Workshop veranstaltet von der ESA findet am ESTEC Weltraumzentrum in Holland statt.

SPS/IPC/Drives, Nürnberg

25.-27. November. Die Messe für elektrische Automatisierungstechnik. Sie umfasst alle Komponenten bis hin zum System und bietet damit integrierte Automatisierungslösungen.

Die aicas GmbH präsentiert sich zusammen mit Euros Embedded in Halle 7A, Stand 622.

Embedded World, Nürnberg

17.-19. Februar. Die wichtigste Embedded Messe in Europa. Besuchen Sie uns auf unserem Gemeinschaftsstand mit Euros Embedded in Halle 11, Stand 222.

Kontakt

Redakteure:

Dr. James J. Hunt (jjh), Dr. Torsten Rupp (tr), Roman Schnider (rs), Dr. Fridtjof Siebert (fs), Andy Walter (aw)

aicas GmbH

Hoepfner Burg
Haid-und-Neu-Str. 18
76131 Karlsruhe
Germany

tel +49.721.663.968-0
fax +49.721.663.968-99
email info@aicas.com
web www.aicas.com