

# aicas news

Neuigkeiten für Softwareentwickler kritischer Anwendungen

Frühling 2008

## ↓ Editorial

Liebe Leser,

der Frühling ist eine Zeit der Erneuerung und aicas ist dabei keine Ausnahme. Der Frühling bringt eine neue Version von JamaicaVM mit vielen neuen Features für Java in Embedded Systems und neuen Werkzeugen wie dem Thread Monitor sowie JamaicaVM für CoDeSys. Diese neuen Werkzeuge können die Entwicklung künftiger Embedded Projekte deutlich optimieren.

Dieser Newsletter gibt Ihnen eine Übersicht über die neuen JamaicaVM-Features und aicas Produkte. Der Hauptartikel im Innenteil beschreibt, wie der Thread Monitor Sie bei der Entwicklung unterstützen kann. Auf Seite 4 stellen wir Ihnen JamaicaVM für CoDeSys und ARINC 653 vor.

Nicht nur auf der technischen Seite gibt es Neuigkeiten: Diese Ausgabe präsentiert auch unsere neuen Vertriebspartner rund um den Globus. Wo immer Sie sich befinden, erhalten Sie nun noch besseren Support für aicas-Produkte.

Ob Sie Systeme für die Luftfahrt, Industrie Automatisierung oder andere Märkte entwickeln, sicher finden Sie in diesem Newsletter interessante Anregungen für Ihre Projekte.

Mit freundlichen Grüßen,

Dr. James J. Hunt, CEO

## JamaicaVM Version 3.2

### Generics und Enumerations für Embedded Systems

Die neue JamaicaVM 3.2 unterstützt die Neuerungen von Java 5. Nun können auch Programmierer von Embedded Systems und Echtzeit Anwendungen die Vorteile von Generics und Enumerations mit allen Vorzügen der RTSJ und Latenzen im  $\mu$ -Sekundenbereich nutzen. Garbage Collector, Interpreter, Scheduler und die native Schnittstelle JNI wurden optimiert.

Die Unterstützung von Generics und Enumerations aus Java 5 verbessert die Typsicherheit eines Programms. Beide Features sind sehr hilfreich, um die Korrektheit von Programmen nachzuweisen.

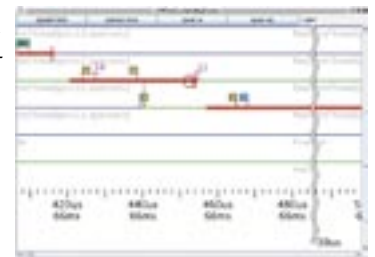
Neben den Java Features lässt sich die neue Version flexibler für bestimmte Anwendungen konfigurieren. Die durchschnittliche Allokationszeit wurde erheblich verbessert, was vor allem Grafikanwendungen zu Gute kommt.

Die Standard-Einstellungen des Builders wurden so geändert, dass bereits „out of the box“ gute Performance erreicht wird. Durch die Verwendung zusätzlicher Optimierungen kann die Performance weiter verbessert werden und ein Footprint von weniger als 800KB erreicht werden.

Auch die VM selbst ist leistungsfähiger geworden. Sie kann Deadlocks zur Laufzeit erkennen und wirft eine Exception, sobald ein Deadlock auftritt. Außerdem können nun auch fatale POSIX-Signale als

RTSJ Happenings behandelt werden. Beide Features sind nützlich für Debugging und zur Fehlerbehebung.

Selbst mit allen neuen Features ist diese Release die schnellste JamaicaVM, die es je gab.



Thread Monitor

Der Scheduler wurde überarbeitet und unterstützt nun Echtzeit Prioritäten für Java Threads auch auf Nicht-Echtzeit-Betriebssystemen, beispielsweise unter Linux. Die Umschaltzeit bei Thread Wechseln wurde deutlich verbessert. Auch der Interpreter der neuen JamaicaVM ist etwa 25% schneller geworden. Wichtige Systembibliotheken wie ZIP und SSL wurden ebenfalls optimiert. Das Aufrufen von in C geschriebenen Methoden über das Java Native Interface (JNI) ist mehr als doppelt so schnell wie in früheren Versionen.

Der neue Thread Monitor, der JamaicaVM ab Version 3.2 unterstützt und den wir auf Seite 2 vorstellen, erleichtert die Entwicklung und Optimierung von Embedded Systems erheblich. Evaluierungsversionen sind erhältlich auf der aicas Homepage: <http://www.aicas.com/download>

## ↓ Neuigkeiten

### SuReal Forschungsprojekt

aicas ist Partner im deutschen Forschungsprojekt SuReal. SuReal verbessert Software-Entwicklungswerkzeuge für echtzeit- und sicherheitskritische Systeme. Weitere Industrielle Partner sind AbsInt für WCETA, Syntavision für Scheduling-Analyse und ScopeSET für UML-Werkzeuge, sowie DF-KI und die technischen Universitäten von Dresden, Braunschweig und München. aicas arbeitet an der Kombination dieser Werkzeuge mit ihrer Datenflussanalyse um genaue Timing- und Scheduling-Analyse

und Java-Code-Generierung aus UML zu ermöglichen. Die Werkzeuge werden am Ende des Projekts veröffentlicht.

### Microsoft Embedded Partner

aicas ist dem Microsoft Windows Embedded Partner Program beigetreten. JamaicaVM bringt die Leistung und Sicherheit von Java auf WindowsCE und ermöglicht Entwicklern dort nun echte „write once, run anywhere“-Programmierung. JamaicaVM ist die einzige JavaVM für Windows CE, die Swing und RTSJ APIs vereint.

### JEOPARD Projekt beginnt

Europa investiert über 3 Mio. Euro in Forschung und Entwicklung von Echtzeit-Java-Technologie für Multicore-Systeme. Unter Leitung von aicas und der Open Group arbeitet ein Konsortium führender europäischer Firmen und Universitäten an Werkzeugen und Ausführungsumgebungen für deterministische Multiprozessor-Systeme. Das Konsortium trägt auch zu relevanten Standards wie RTSJ bei. Weitere Informationen sind unter <http://www.jeopard.org> erhältlich.

# Jamaica Thread Monitor

## Multi-threaded Anwendungen in Nanosekundenaufösung analysieren

Für die JamaicaVM 3.2 hat aicas neben einem verbesserten Scheduler ein Werkzeug entwickelt, welches dabei hilft, das Verhalten von Anwendungen mit mehreren Threads besser zu verstehen und diese zu debuggen. Mit Hilfe des Thread Monitors kann der Entwickler in eine unter JamaicaVM laufende Java-Anwendung hineinschauen und sich wichtige Ereignisse, die in der Java VM auftreten, grafisch anzeigen lassen.

Die verfolgbareren Ereignisse reichen von Kontextwechseln über Monitor-Operationen bis hin zur Garbage-Collector-Akti-



Abbildung 1: Thread Monitor-Kontrollfenster

vität. Zusätzlich kann ein Anwendungsentwickler eigene Ereignisse erzeugen die bestimmte Codestellen seiner Applikation im Thread Monitor sichtbar markieren. Alle Events können bis in den Nanosekundenbereich beobachtet werden.

Das Sammeln von Ereignissen für den Thread Monitor wird auf Seiten der VM mit den beiden System Properties *jamaica.scheduler\_events\_port* und *jamaica.scheduler\_events\_port\_blocking* gesteuert. Um das Sammeln von Ereignissen in der JamaicaVM zu aktivieren, setzt der Benutzer den Wert einer dieser Properties auf die Portnummer, zu der der Thread Monitor später verbinden soll. Wenn die Ereignis-Sammlung aktiviert ist, werden die angeforderten Ereignisse in einen Puffer geschrieben und durch einen periodischen Thread mit hoher Priorität an den Thread Monitor gesendet. Die Größe des Puffers und die Zeitintervalle können im Monitor-Werkzeug festgelegt werden.

Wenn der Thread Monitor gestartet wird öffnet sich ein Kontrollfenster (Abb. 1). In diesem Fenster kann die Verbindung zur JamaicaVM auf dem Zielsystem konfiguriert werden. Der Benutzer kann die Netzwerkadresse und Portnummer des Scheduling Event Servers in der VM an-

geben und auswählen, welche Arten von Events angezeigt werden sollen.

- "Thread State Changes" zeichnet auf, wie sich der Zustand eines Threads im Laufe der Zeit ändert, inklusive der Angabe, welche Threads Zustandsänderungen in anderen Threads hervorrufen.
- "Thread Priority Changes" zeigen alle Arten von Prioritätsveränderungen: Sowohl explizite Aufrufe von *Thread.setPriority* als auch Anpassungen durch die Prioritätsvererbung in Java-Monitoren.
- Thread-Namen zeigen den Java-Namen eines Threads.

- "Monitor Enter/Exit"-Ereignisse geben an, wann ein Thread einen Monitor erfolgreich betritt bzw. verlässt und wann er blockiert wird, weil bereits ein anderer Thread diesen Monitor betreten hat.

- "GC Activity" zeichnet auf, wann die inkrementelle Speicherbereinigung arbeitet.

- "Start Execution" zeigt, wann ein Thread tatsächlich mit der Codeausführung beginnt, nachdem sein Zustand auf "Running" gesetzt wurde.

- "Reschedule" markiert, wann ein Thread durch einen Reschedule Request seinen Zustand ändert.

- Alle Threads innerhalb der JamaicaVM, die den Zustand "Ready" (bereit) haben, sind auch aus Sicht des Betriebssystems bereit, gestartet zu werden. So kann es passieren, dass das Betriebssystem einen Thread zur Ausführung auswählt, der nicht mit dem laufenden Thread innerhalb der VM übereinstimmt. JamaicaVM bemerkt dies und lässt den Thread, der vom Betriebssystem gewählt wurde, einen "Yield" durchführen, um einem anderen Thread die Ausführung zu ermöglichen.

- "User Events" enthalten benutzerdefinierte Nachrichten und können aus dem Java-Code heraus ausgelöst werden. Somit können wichtige Codestellen bequem im Thread Monitor markiert werden.

Zwischen den Klicks auf "Aufzeichnung starten" und "Aufzeichnung beenden" erhält das Werkzeug Ereignisse von der Ziel VM. Nach der Datensammlung öffnet sich ein neues Fenster. Innerhalb dieser Visualisierung kann der Benutzer hinein- und heraus-zoomen und die Zeitachse expandieren oder zusammenziehen. Das Werk-

zeug ermöglicht es auch, die gesammelten Daten in einer Datei zu speichern.

Um die Ausgabe des Thread Monitors besser zu verstehen, ist es hilfreich, einige Kenntnisse über den JamaicaVM Scheduler zu besitzen. JamaicaVM 3.2 verfügt über einen neuen Scheduler wurde für die Version 3.2 implementiert. Er stellt nun die Möglichkeit bereit, Echtzeitprioritäten in Java-Programmen auf Betriebssystemen zu erzwingen, die kein striktes prioritätsbasiertes Scheduling (z.B. Linux für Programme im User Mode) bieten. Der Scheduler reduziert den Overhead für JNI-Aufrufe und hilft dem Betriebssystem, die CPU-Ressourcen für Threads der VM besser einzuteilen. JamaicaVM 3.2 ist somit deutlich besser in das Betriebssystem integriert und ermöglicht Java-Anwendungen mit mehreren Threads einen wesentlich besseren Durchsatz.

Der VM-Scheduler kontrolliert, wie in früheren JamaicaVM Versionen auch schon, zu jeder Zeit, welche Threads innerhalb der VM laufen. Dabei ersetzt der VM-Scheduler nicht den Scheduler des Betriebssystems, sondern unterstützt ihn bei seiner Arbeit. Dies erlaubt beispielsweise eine einfache (und effiziente) Implementierung von Java-Monitoren, die nicht auf aufwendige Semaphoren der Betriebssysteme zurückgreifen müssen.

Wie in Abbildung 2 gezeigt, können Threads attached oder detached sein. Alle Threads, die in der VM erzeugt werden, sind standardmäßig attached, werden also vom VM-Scheduler gesteuert. Threads, die Systemaufrufe ausführen, müssen detached laufen. Dies erlaubt dem VM-Sche-

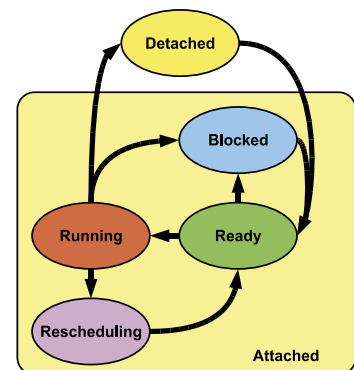


Abbildung 2: Thread-Zustände

duler, einen anderen Java Thread auszuführen, während der erste Thread zum Beispiel in einem I/O Request blockiert ist. Da es entscheidend für Echtzeitanwendungen ist, dass kein Thread jemals in einem Systemaufruf blockiert, während er an die VM gebunden ist, muss jeglicher

JNI Code detached von JamaicaVM ausgeführt werden.

Für die Interpretation der Thread-Monitor-Daten ist die Unterscheidung zwischen dem attached- und detached-Modus wichtig. Threads, die detached von der VM sind, könnten die CPU nutzen, was bedeutet, dass der in der VM als laufend angezeigte Thread tatsächlich gar keinen Code ausführt.

Threads, die attached sind, können die Zustände "Running", "Rescheduling", "Ready" oder "Blocked" annehmen. Als "Running" gilt der Thread, der gegenwärtig innerhalb des Kontexts der VM ausgeführt wird. "Rescheduling" ist ein Unterzustand eines laufenden Threads. Der "Running"-Zustand eines Threads wird dann zu "Rescheduling" geändert, wenn ein anderer Thread eine höhere Ausführungsberechtigung erhält. Dies passiert, wenn ein Thread höherer Priorität nicht länger blockiert wird oder attached ist.



Abbildung 3: Einfache Dead Lock-Situation

Die VM führt den laufenden Thread noch bis zum nächsten Synchronisationspunkt aus und wechselt dann zu dem Thread mit höherer Berechtigung. Threads sind "Ready", wenn sie attached sind und ausgeführt werden können sobald kein anderer Thread mit höherer Berechtigung mehr läuft. Attached Threads können aus verschiedenen Gründen blockieren. Der häufigste Grund sind Aufrufe von *Thread.sleep*, *Object.wait* und das Betreten eines besetzten Monitors.

Das folgende Beispiel demonstriert ein weiteres neues Feature in JamaicaVM 3.2: Im Thread Monitor lässt sich beobachten, wie sich eine Anwendung dank der neuen Deadlock Erkennung von einem Deadlock erholt.

Abbildung 3 zeigt eine einfache Situation mit zwei Threads und zwei Locks. Zunächst läuft *Thread\_1*, erkennbar durch die breite, rote Linie. Dieser Thread betritt den Monitor A. Die weißen Kästchen *enter:A* und *bold:A* repräsentieren benutzerdefinierte Ereignisse kurz vor und nach dem "synchronized" Statement. Die kleine Flagge zwischen den benutzerdefinierten Ereignissen zeigt, dass der Thread erfolgreich den Monitor betreten hat. Farbe und Nummer identifizieren den Monitor. Wenn die Sammlung von Monitor-Daten aktiviert ist, werden sehr viele Monitore aus den Java-Standardklassen erscheinen. Der Benutzer kann bestimmte Monitore durch einen Rechtsklick auf ei-

nes der Ereignisse und den entsprechenden Menüpunkt verstecken. Nachdem *Thread\_1* Monitor A betreten hat, übergibt er an *Thread\_2*. *Thread\_2* betritt dann den Monitor B und übergibt zurück an *Thread\_1*. Wenn *Thread\_1* nun versucht, Monitor B zu betreten, blockiert er, da er einen besetzten Monitor betreten möchte (angezeigt durch das kleine Achteck). Da *Thread\_1* nun blockiert, erhält *Thread\_2* den Zustand "Running" zurück und versucht Monitor A zu betreten, was zu einem Deadlock führt. JamaicaVM erkennt diese Situation und wirft einen *DeadLockError* der von der Anwendung aufgefangen und behandelt werden kann. Unter der Linie von *Thread\_2* ist auch Garbage-Collector-Aktivität zu sehen. Diese wird durch die Zuweisung des Fehlerobjekts und den Stack Trace hervorgerufen. Das Beispiel fängt den *DeadLockError* auf und gibt Monitor B sicher frei, der nun von *Thread\_1* beansprucht werden kann.

In einem neuen Projekt half kürzlich der Thread Monitor dabei, einen Flaschenhals in einer Kundenanwendung zu identifizieren. Diese Applikation besitzt einen Kommunikations-Thread, der eine zyklische Abfrage von Datenobjekten über das Netzwerk ausführt. Jedes Datenobjekt wird darauf untersucht, ob sich der Wert geändert hat oder nicht und ob seine "Change Listeners" aktiviert wurden. Die Priorität dieses Threads war höher gesetzt als die Priorität aller anderen Threads in der VM. Dies sollte dem AWT Event Dispatcher erlauben, für mehrere gesammelte Zeichen-Events lediglich ein einziges Bildschirmupdate durchzuführen. Nichtsdestotrotz zeigte das Benutzer-Interface nicht das gewünschte Aktualisierungsverhalten. Anstatt alle UI-Elemente in einer einzigen Aktualisierung zu erneuern, gab es mehrere Aktualisierungen, die jeweils eine unterschiedliche Anzahl von Elementen erneuerten. Ohne den Jamaica Thread Monitor wäre diese Problem nur sehr schwer zu finden gewesen.

Die Ausgabe im Thread Monitor zeigte, dass der AWT Event Dispatcher Thread lief, bevor alle Änderungen im Netzwerk übertragen wurden, da der Kommunikations-Thread blockierende I/O-Zugriffe durchführte. Bei jeder Netzwerk-Kommunikation blockierte der Thread hoher Priorität für eine kurze Zeitspanne, welche dem Event Dispatcher Thread die Möglichkeit eröffnete, Zeichen-Events, die bereits in der Warteschlange waren, durchzuführen. Das Problem konnte beseitigt werden durch das Sammeln der Zeichen-

Events in einer Queue, die erst nach der vollendeten Netzwerk-Kommunikation abgearbeitet wurde.

Multithreading bringt eine neue Klasse von potentiellen Fehlern in die Programmierung von Anwendungen. Diese Fehler können nicht nur auf Anwendungsfehler wie Deadlocks hinauslaufen, sondern können auch dazu führen, dass die Anwendung langsamer als erwartet läuft. Die Erkennung solcher Probleme ist umständlich, da Fehler, die nur bei einer bestimmten Ausführungsreihenfolge der Anwendungs-Threads auftreten, unter Umständen verschwinden, sobald Codeänderungen zur Fehlersuche vorgenommen werden. Der Jamaica Thread Monitor ermöglicht die Untersuchung einer Anwendung, ohne das Zeitverhalten zu beeinflussen und erleichtert somit die Fehlersuche ungemein.

## ↓ Neue Distributoren

Um ihre Marktposition noch weiter zu verbessern und neue Märkte zu erschließen, hat die aicas GmbH 2007 ihr Vertriebspartnernetz stark ausgebaut. Somit kann sichergestellt werden, dass auch Kunden in Ländern ohne aicas Niederlassungen hervorragenden Service und regionale Ansprechpartner vorfinden. Neu hinzugekommen sind dabei die Partnerschaften mit Sysgo, Realtimewave und Masterpiece.

Die Sysgo AG ist weltweiter Anbieter von hochzuverlässiger Software für kritische Systeme und bietet gemeinsam mit aicas JamaicaVM für PikeOS an. Die Zusammenarbeit fokussiert sich auf die Bereiche Luft- und Raumfahrt, Verteidigung, Industrie-Automatisierung und medizinische Systeme.

Realtimewave Co., Ltd., ist ein bedeutender Dienstleister und Systemintegrator in Südkorea. Zu den Geschäftsfeldern gehören Embedded Systems in Hightech-Anwendungen in den Bereichen Avionik, Robotik, Simulationen und Steuerungen.

Masterpiece Technology Co., Ltd., ist ein Vorreiter der Systemintegration und Anbieter von Embedded-Lösungen in China. Masterpiece erweitert durch die Partnerschaft sein Portfolio um ein angesehenes europäisches Produkt. Gleichzeitig erhalten die chinesischen Kunden von aicas einen kompetenten Ansprechpartner vor Ort.

Durch diese Partnerschaften baut aicas seine globale Position als Anbieter von Entwicklungsumgebungen für Embedded Systems weiter aus und bietet seinen Kunden bessere Unterstützung im Vertrieb und im Service.

# ARINC 653

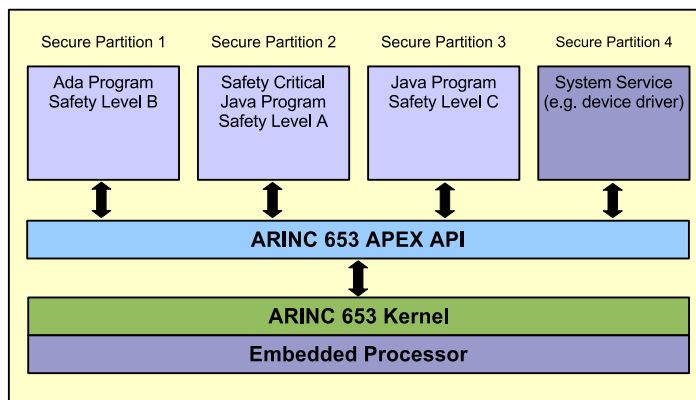
## Ein Betriebssystemstandard in der Luftfahrt für Ressourcen- und Zeit-Partitionierung

Die ARINC-Spezifikation 653 definiert einen Betriebssystemstandard, der hauptsächlich in sicherheits- und missionskritischen Systemen in der Luftfahrtindustrie zum Einsatz kommt.

Dabei ermöglicht ein Application EXecutive (APEX), mit einer Ressourcen- und Zeit-Partitionierung das sichere Ausführen verschiedener Anwendungen auf einer Hardware zur gleichen Zeit. Jede Anwendung läuft in ihrer eigenen Partition, wobei Multitasking innerhalb einer Partition erlaubt ist. ARINC 653 Partitionen können lediglich auf ihre eigenen Ressourcen zugreifen. Somit sind beispielsweise Speicherbereiche anderer Partitionen für sie nicht sichtbar. Darüberhinaus teilt

die APEX jeder Partition feste Zeitscheiben zu. Die zyklus-basierte Zeiteinteilung garantiert eine deterministische Ausführung und verhindert so Race Conditions zwischen den Anwendungen.

Für die Kommunikation sind vereinheitlichte Schnittstellen definiert. Diese können sowohl zur Kommunikation der Partitionen untereinander als auch zum Datenaustausch mit der Aussenwelt über TCP/IP verwendet werden.



ARINC-Beispiel mit 4 Partitionen

Die Erfüllung der ARINC 653 Spezifikation ist ein erster Schritt hin zu einer DO-178B Zertifizierung. DO-178B ist ein Standard für die Software-Entwicklung in Luftfahrtsystemen. Ohne eine Zertifizierung gemäß dieses Standards darf keine Software in sicherheitskritischen Systemen im Luftfahrtumfeld zum Einsatz kommen. Abhängig von der Auswirkung eines Fehlers werden die Anwendungen in verschiedene Gefahrenklassen von E (ohne Wirkung) bis A (Katastrophal) eingeteilt. Die Isolation der Anwendungen durch ARINC 653-Partitionierung ermöglicht es, DO-178B zertifizierte Anwendungen verschiedener Klassen auf der selben Hardware laufen zu lassen. Systeme, die konform zum ARINC 653 Standard sind, lassen sich bis hin zu Klasse A zertifizieren.

aicas unterstützt derzeit die ARINC 653 konformen Betriebssysteme PikeOS von SYSGO und VxWorks 653 von WindRiver. In Kürze wird außerdem eine Version für LynxOS 178 von LynuxWorks verfügbar sein.

## JamaicaVM für CoDeSys WebVisu

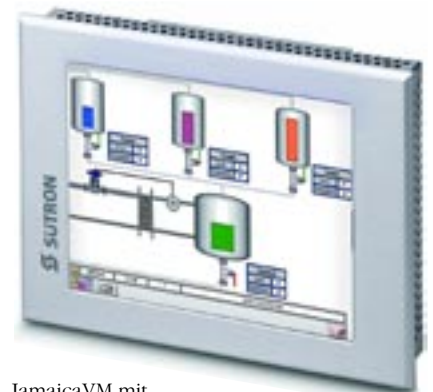
### IEC-61131-3 Visualisierungen für Embedded Systems

In Zusammenarbeit mit 3S - Smart Software Solutions bietet aicas nun JamaicaVM für CoDeSys WebVisu an. Die CoDeSys-Projektierungssoftware ermöglicht es, schnell und mit wenig Aufwand IEC 61131-3 konforme Visualisierungen zu erstellen. Mit JamaicaVM ist es nun erstmals möglich, mit CoDeSys erstellte Visualisierungen auf einer Vielzahl von Embedded Systems anzuzeigen. Das von aicas erhältliche JamaicaVM für CoDeSys WebVisu lässt sich schnell und unkompliziert installieren und integriert sich nahtlos in eine vorhandene CoDeSys-Entwicklungsumgebung. Mögliche Zielsysteme sind alle Embedded Systems mit Display und Steuerungen mit in-

tegriertem Display. Als Betriebssysteme werden zur Zeit WindowsCE, VxWorks, OS-9 und Linux unterstützt. Weitere Betriebssysteme sind auf Anfrage erhältlich.

Die gewünschte Visualisierung auf einem Embedded System anzuzeigen ist dabei einfach und unkompliziert. Der Kunde erstellt zunächst mit seiner CoDeSys-Software eine Visualisierung. Diese wird zusammen mit der ebenfalls projektierten Steuerungsapplikation auf die (Soft-)SPS übertragen. Beim Start auf einem Webpanel holt sich JamaicaVM für CoDeSys WebVisu die projektierten Daten von der Steuerung und zeigt die Visualisierung an. Änderungen am Projekt brauchen also lediglich neu auf die Steuerung geladen zu werden. Durch die JamaicaVM-Technologie wird die Visualisierung dabei selbst auf kleinen Systemen mit ausgezeichneter Performance ausgeführt.

Für den Kunden bietet das neue Lizenzmodell den großen Vorteil, dass er keine Projektlizenz für die komplette JamaicaVM Toolchain erwerben muss, sondern eine fertige Anwendung für sein Zielgerät erhält. Dadurch ist es möglich, JamaicaVM für CoDeSys WebVisu zu sehr geringen Stückkosten zu liefern. Dies macht diese Lösung besonders für Hersteller und



JamaicaVM mit CoDeSys Webvisualisierung

Vertreiber von Panels attraktiv. Dabei ist das flexible Preismodell in seiner Gestaltung an das Preismodell von 3S - Smart Software Solutions angelehnt, und sowohl für kleine als auch für größere Stückzahlen gut geeignet.

### ↓ Veranstaltungen

Wir möchten Sie auf folgenden Veranstaltungen willkommen heißen:

**ERTS 2008,**  
Toulouse, 29. Jan – 1. Feb 2008

**Embedded World 2008,**  
Nürnberg, 26. – 28. Februar 2008

**Avionics Event,**  
Amsterdam, 5. – 6. März 2008

**Embedded Systems Conference**  
San Jose, 14. – 18. April 2008

### ↓ Kontakt

**aicas GmbH**

Haid-und-Neu-Straße 18  
76131 Karlsruhe  
Deutschland

tel. +49 721 663 968-0  
fax +49 721 663 968-99  
email [info@aicas.com](mailto:info@aicas.com)  
web [www.aicas.com](http://www.aicas.com)