QoS Issues in Java Platforms

Antonio Kung

Trialog

Presentation Outline

- HIJA project
- QoS is a requirement for Business Critical Systems
- Security and Dependability Issues in Java Platforms
- HIJA approach
- Conclusions and recommendations

Context: HIJA Project

- HIJA (High Integrity Java Applications)
  - STREP (embedded system)
  - Mission: create the technology conditions that will allow architecture neutrality for *high-integrity* applications

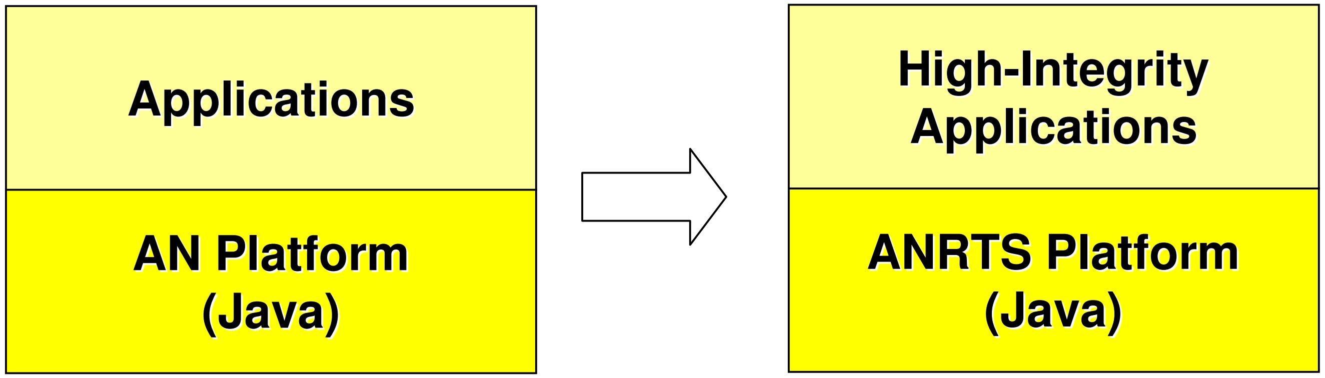
- High Integrity Applications:
  - Applications which need assurance level
    - Safety critical systems e.g. avionics
    - Business critical systems e.g. telematics

HIJA Partners

- Coordination
  - The Open Group
- Technology partners
  - Trialog
  - Aonix
  - Aicas
- Academic partners
  - FZI Karlsruhe
  - U. Karlsruhe
  - U. York
  - U. P. Madrid
- User partners
  - Thales
  - Centro Riserche Fiat
  - Telecom Italia
  - Bellstream

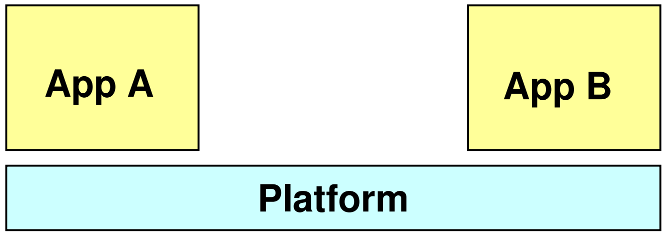
Approach

- Add real-time system (RTS) features to Java

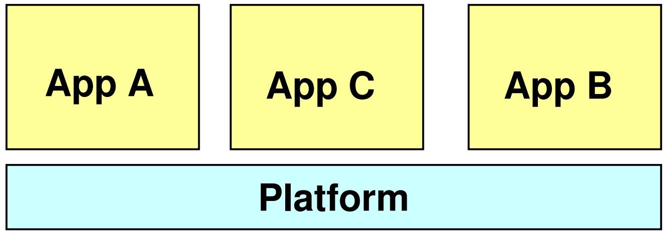


Business Critical Systems

- Include multiple independent applications
  - e.g. telematics platform

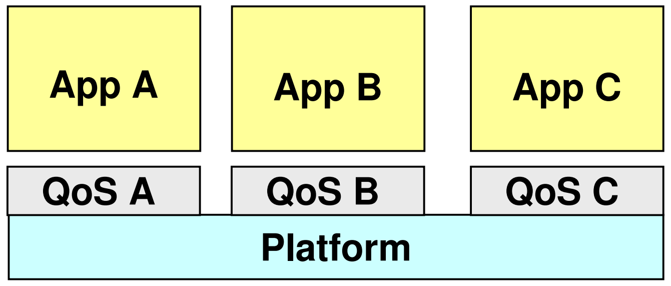


- Required level of assurance :
  - application system resource (CPU, Memory…) protection
  - allocation of resources for new applications



QoS Requirement

- QoS
  - collective effect of service and performances that determine the degree of satisfaction of the service
- Platform QoS requirement
  - ensure system resource protection to each individual application



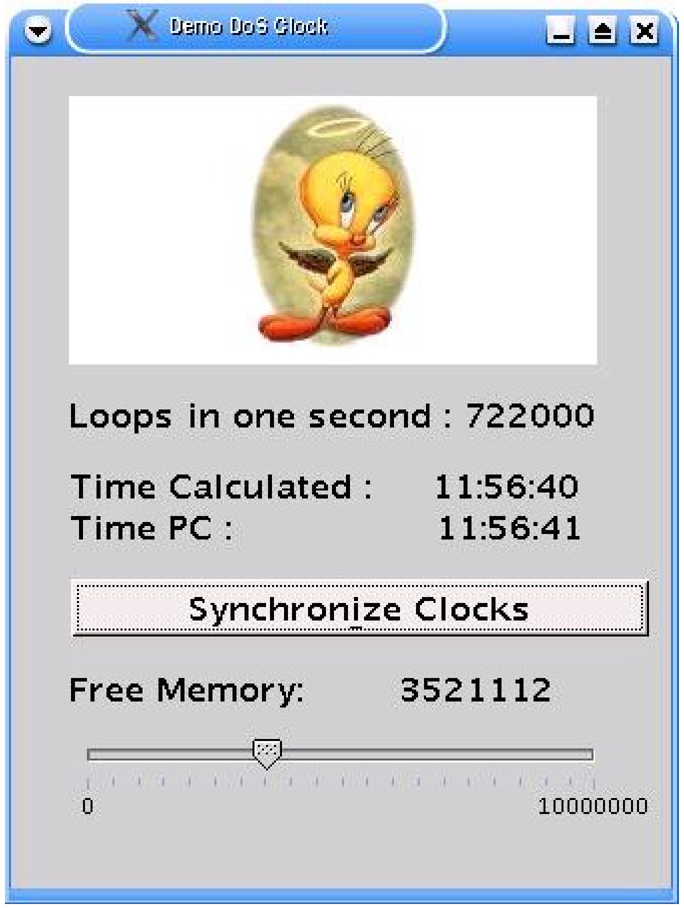
Issues with OSGi

- Most OSGi-based platforms do not provide QoS
- raises security issue
  - Straightforward to provoke a denial of service attack
- raises dependability Issue
  - Denial of service is not necessarily provoked by an attack
  - Most likely provoked by bugs

- Platforms which do not provide QoS are not suitable for business critical applications

Example : Application A

- OSGi bundle
- Required resource
  - periodic CPU
  - some memory
- Task
  - Executes a payload in a loop
  - Counts number of loops N in 1 second
  - Implement a software clock, time incremented every N loops
- If QoS ensured then
  - Software clock is « accurate »



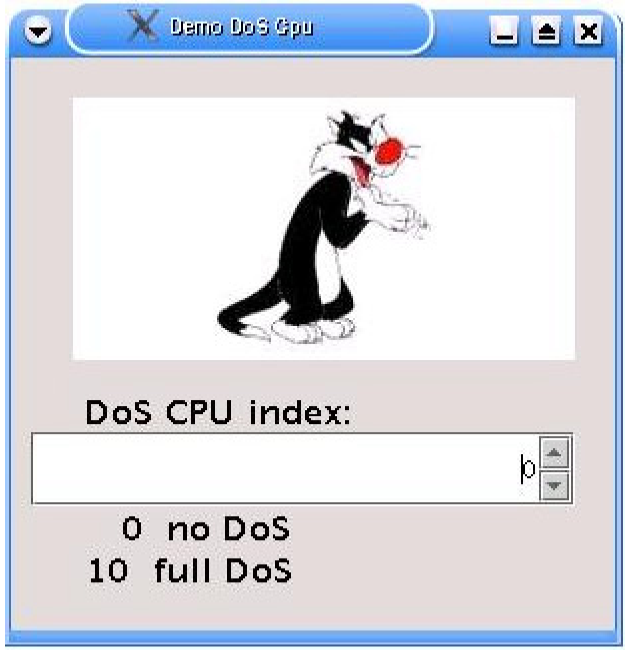
Loops in one second : 722000

Time Calculated : 11:56:40
Time PC : 11:56:41

Free Memory: 3521112

Application A and B

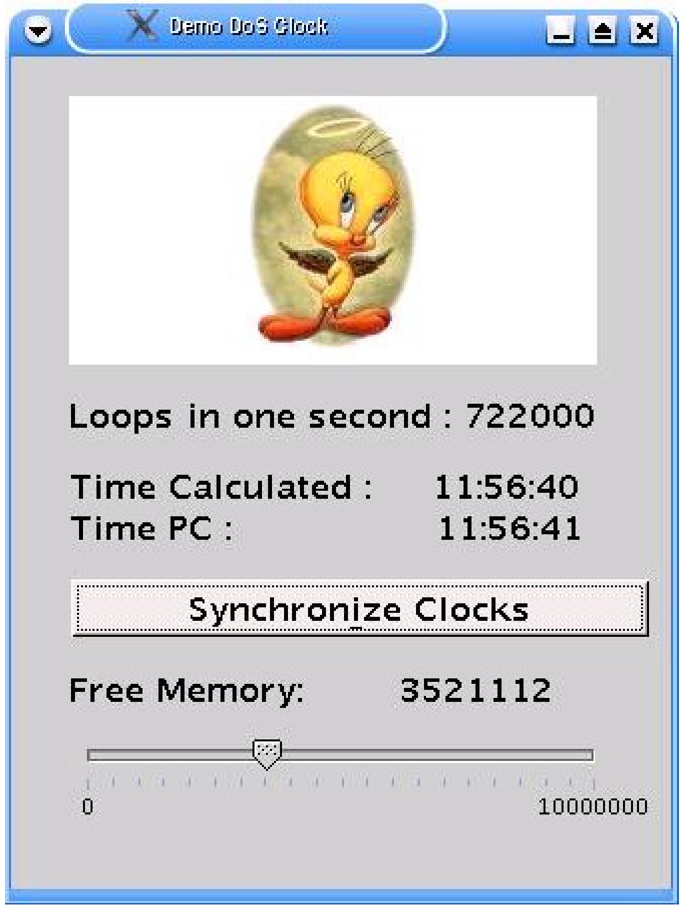
- OSGi bundle B (malicious)
- OSGi bundle A (clock)



Demo DoS Cpu

DoS CPU index:

0 no DoS
10 full DoS



Demo DoS Clock

Loops in one second : 722000

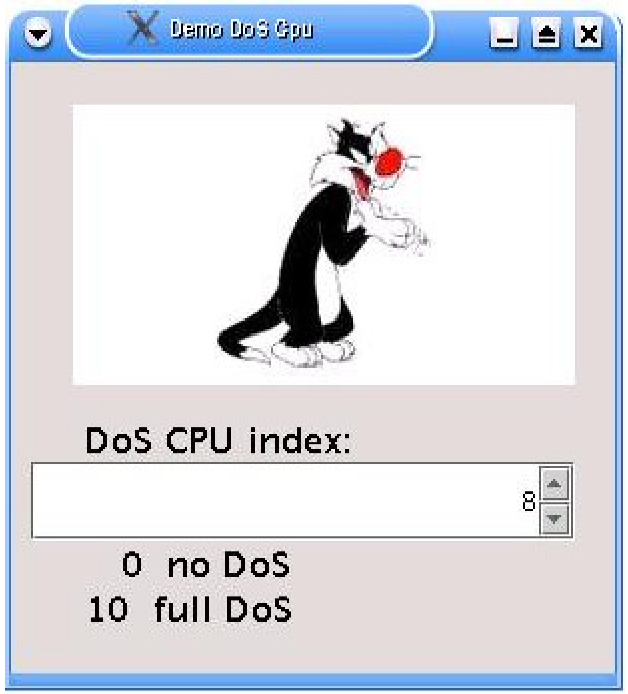
Time Calculated : 11:56:40
Time PC : 11:56:41

Synchronize Clocks

Free Memory: 3521112

CPU Denial of Service

- OSGi bundle B (malicious)

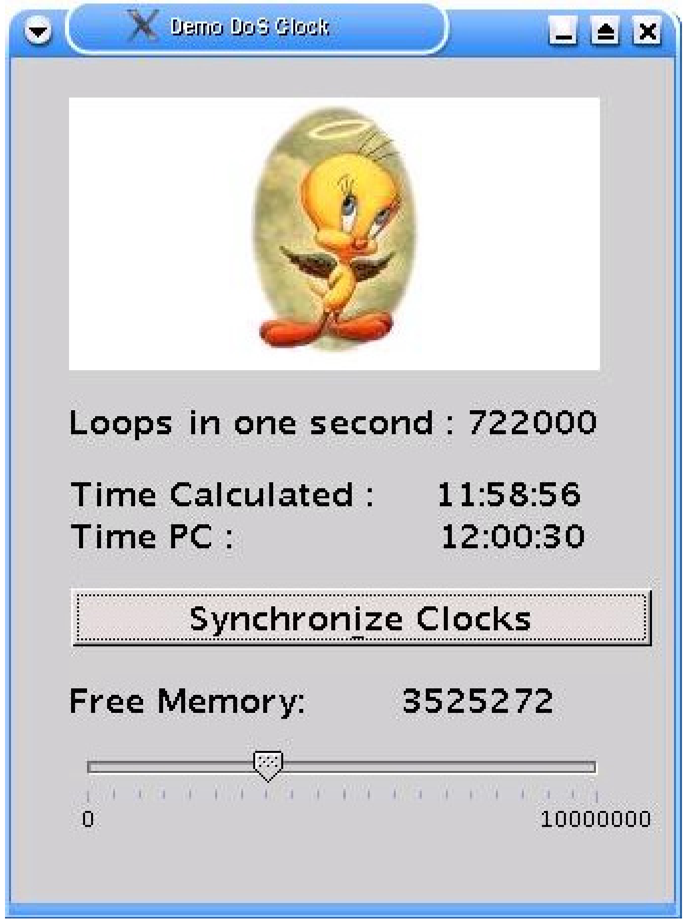


DoS CPU index:

8

0 no DoS
10 full DoS

- OSGi bundle A (clock)



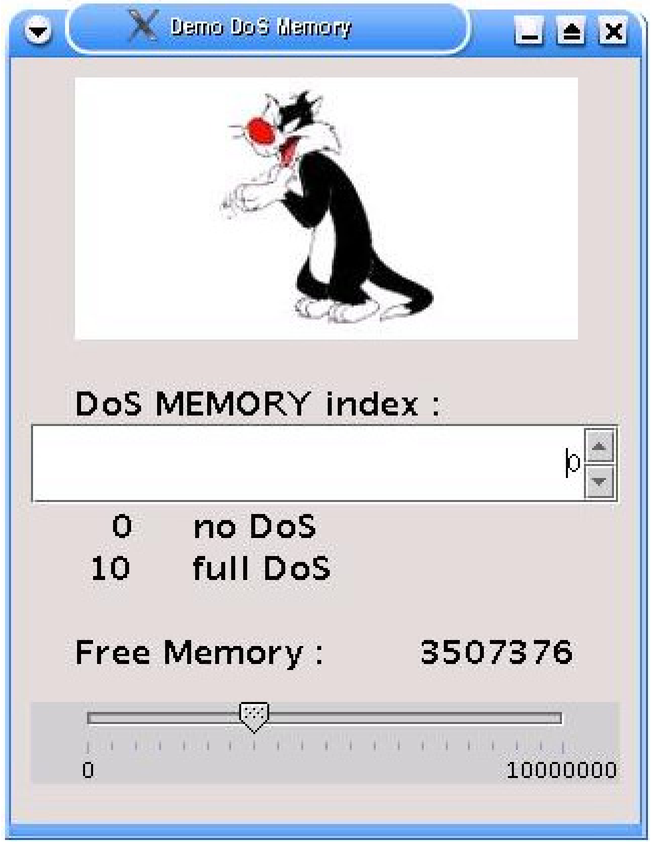
Loops in one second : 722000

Time Calculated : 11:58:56
Time PC : 12:00:30

Free Memory: 3525272

Other case

- OSGi bundle C (malicious)

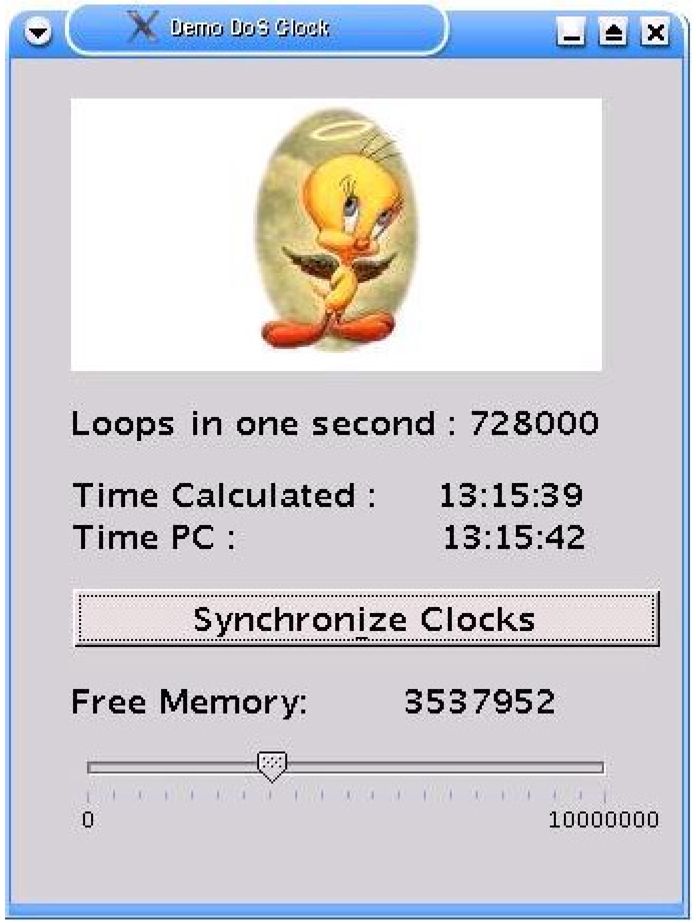


DoS MEMORY index :

0 no DoS
10 full DoS

Free Memory : 3507376

- OSGi bundle A (clock)

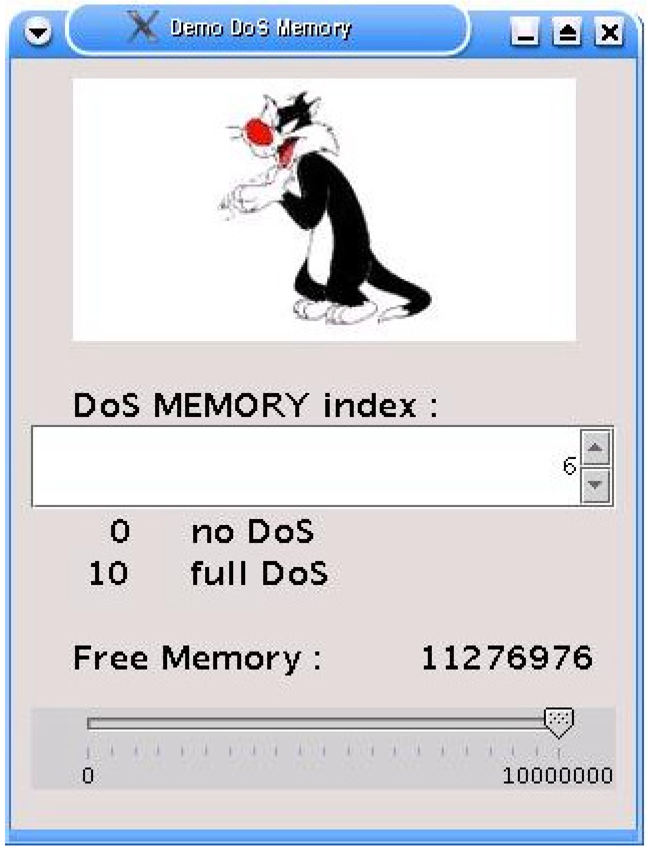


Loops in one second : 728000

Time Calculated : 13:15:39
Time PC : 13:15:42

Free Memory: 3537952

Memory Danger



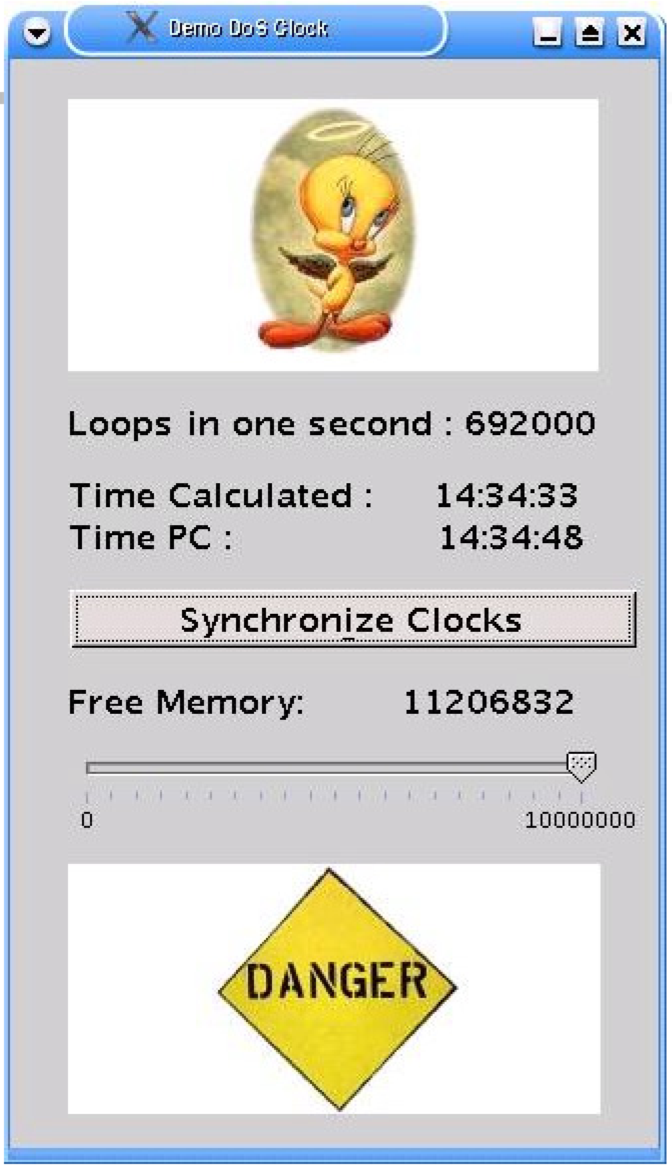
DoS MEMORY index :

6

0 no DoS
10 full DoS

Free Memory : 11276976

0 10000000



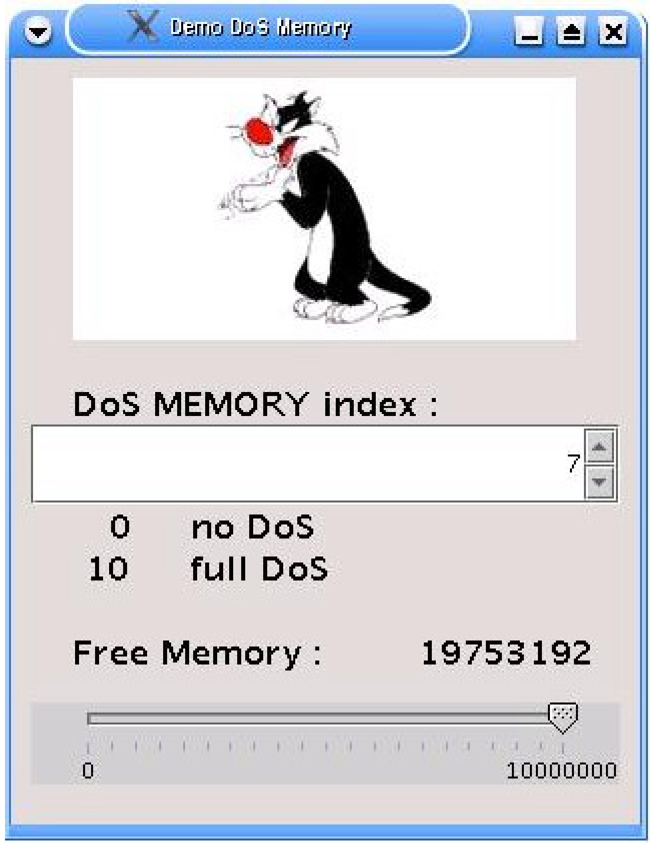
Loops in one second : 692000

Time Calculated : 14:34:33
Time PC : 14:34:48

Free Memory: 11206832

0 10000000

Denial of Service



Demo DoS Memory

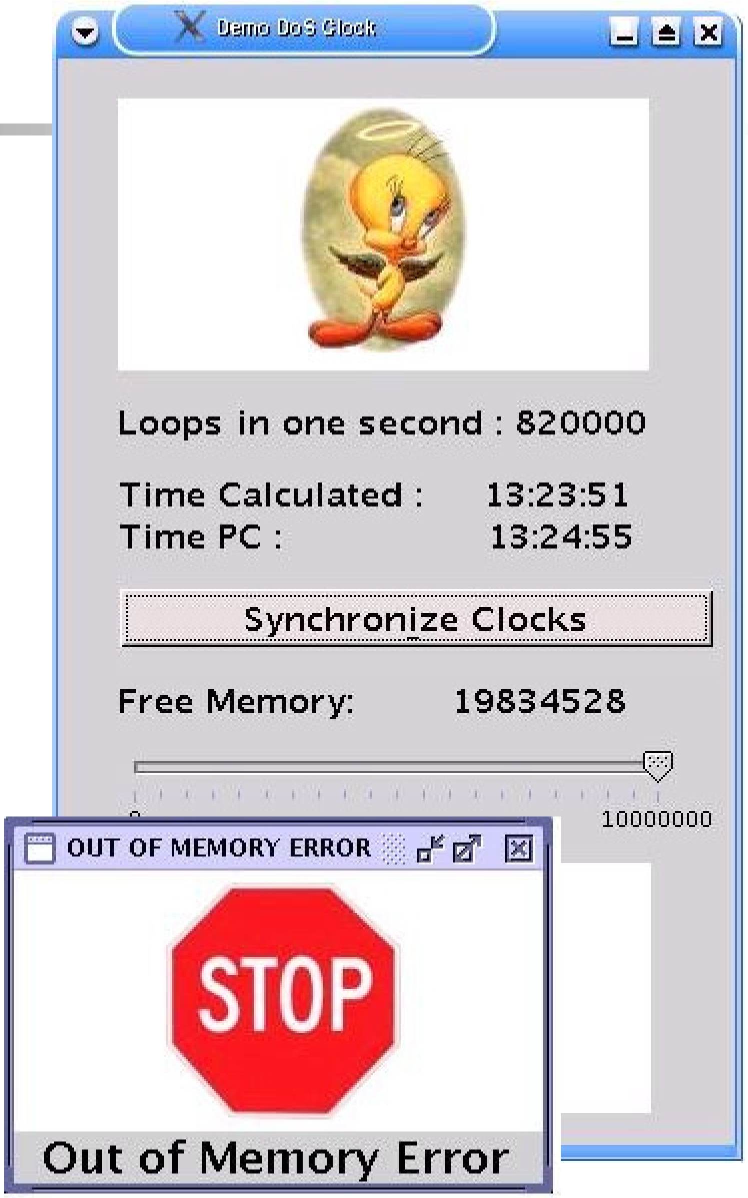
DoS MEMORY index :

7

0 no DoS
10 full DoS

Free Memory : 19753192

0 10000000



Demo DoS Clock

Loops in one second : 820000

Time Calculated : 13:23:51
Time PC : 13:24:55

Synchronize Clocks

Free Memory: 19834528

0 10000000

OUT OF MEMORY ERROR

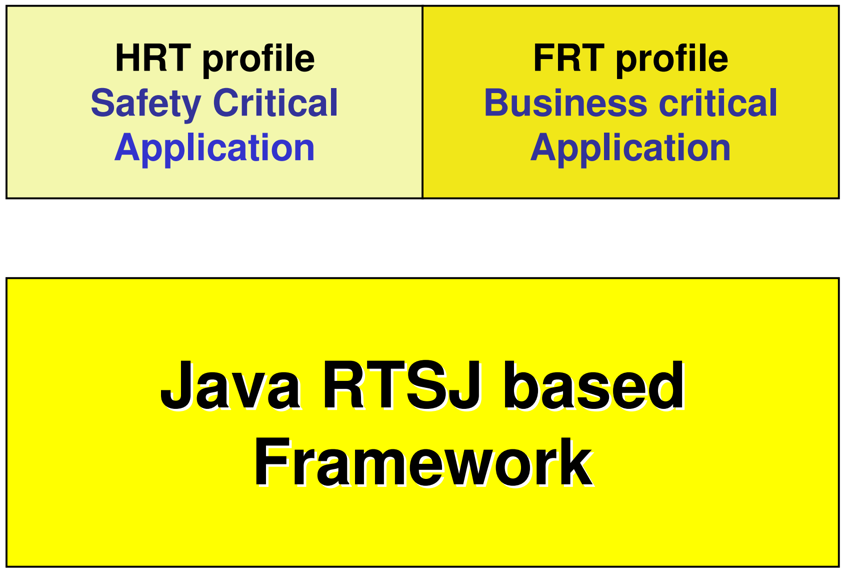
Out of Memory Error

Platforms Integration Issues

- OSGi implementation rely on « typical JVM »
  - no CPU protection
  - no Memory protection
- JVM with multiple memory spaces are not mainstream
  - e.g. JSR 121 Isolate
  - OSGi framework would have to be fully reimplemented
- JVM with CPU enforcement are not maintream
  - depends on underlying OS e.g. RTOS
  - JVM would have to be fully reimplemented

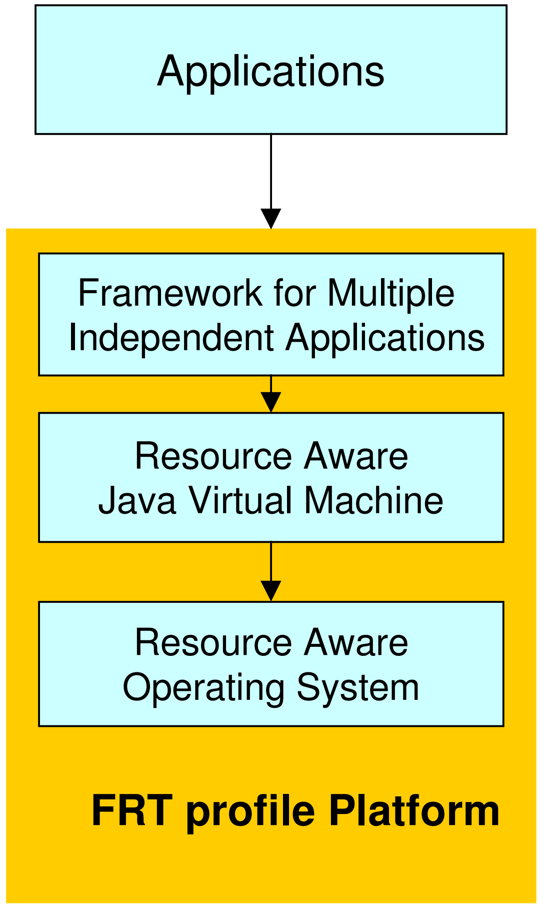
HIJA Approach

- Define profiles on top of RTSJ
  - JSR 1 + Modification specified in WG (Open group forum)


HRT profile
Safety Critical Application
FRT profile
Business critical Application
Java RTSJ based Framework

Resulting Platform

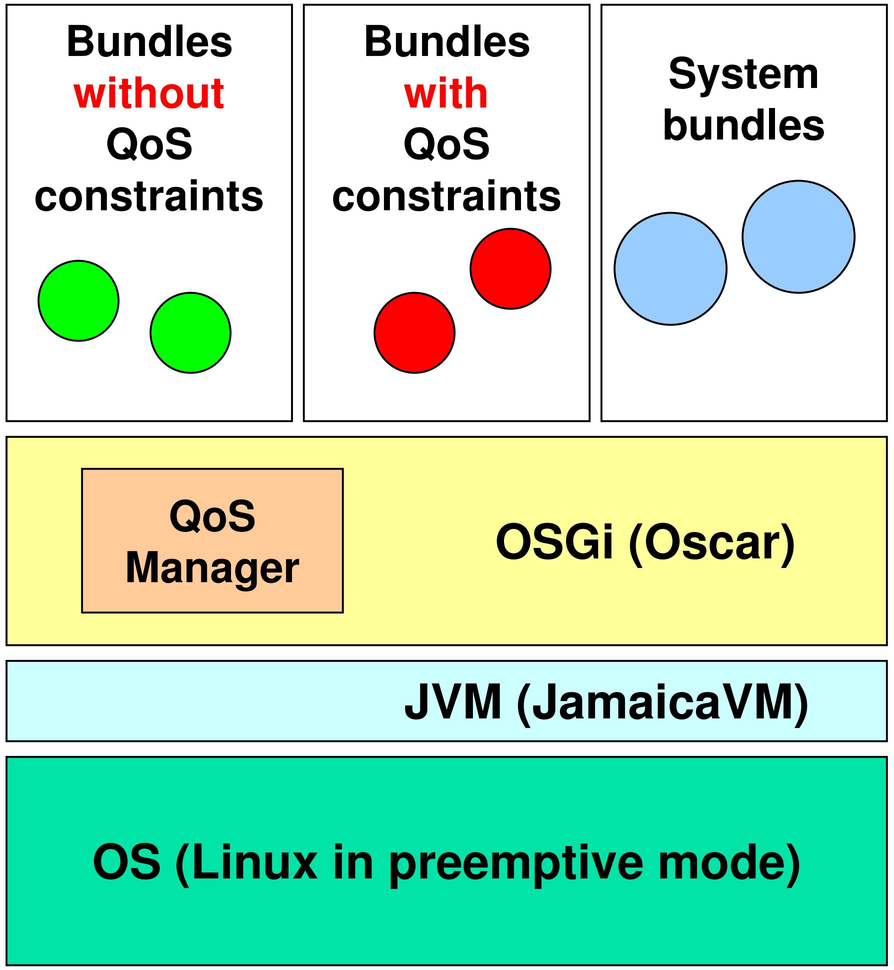
- A resource aware JVM+OS implementing the following mechanisms
  - Admission control
  - Resource accounting
  - QoS enforcement
- Assumption
  - Application declares resource needs


Applications
Framework for Multiple Independent Applications
Resource Aware Java Virtual Machine
Resource Aware Operating System
FRT profile Platform

HIJA Proof of concept application

- Three types of applications:
  - No QoS constraints
  - QoS constraints
  - Communication middleware

- QoS resources are partitioned accordingly


Bundles without QoS constraints
Bundles with QoS constraints
System bundles
QoS Manager
OSGi (Oscar)
JVM (JamaicaVM)
OS (Linux in preemptive mode)

Downloading a new application/bundle

- retrieve QoS contraints from bundle manifest files
  - new attributes
    - QoS-Memory: memory needs
    - QoS-Period & QoS-Budget: CPU needs
- modifies bundle
  - references to java.lang.Thread changed to javax.realtime.RealtimeThread
- stores modified bundle in run-time cache

Enforcing Policy

- QoS manager
  - Relies on RTSJ-compliant VM to enforce QoS policy
    - MemoryParameters
    - ProcessingGroupParameters

More information on HIJA

- ACM proceedings of JTRES 2006
  - 4th Workshop on Java Technologies for Real-time and Embedded Systems
    - 11th to 13rd October 2006, Paris
  - 6 Papers including
    - *ANRTS Platforms*, A.Kung, S.Hansen
    - *Issues in Building ANRTS platform,* A.Kung. J.Hunt, L.G, M.Richard-Foy
    - *Flexible Java Real-Time Profile for Business-Critical Systems,* A.Alejandro, N.François, E.Yu, M.Bianconi, G.Cortese
    - *Safety Critical Applications and Hard Real-Time Profile for Java: A Case Study in Avionics,* E.Hu, E.Jenn, N.Valot, A.Alonso.

Conclusion

- OSGi based platforms are subject to DOS problems
- Approach through static assurance alone is flawed
  - Static assurance:
    - only « trusted » possibly certified bundles
  - Level of assurance is probably costly, and not guaranteed.
- Approach including QoS at platform level is needed
  - Level of assurance is guaranteed

HIJA High-Integrity Java Application

Questions?

www.hija.info

Information Society Technologies