João Américo and Walter Rudametkin
Bull S.A.S./LIG Grenoble

Predictability vs. Dynamism: managing dynamic real-time applications
Outline

• Context
• State-of-the-art
• Problem Identification
• Suggested Approach
• Limitations
• Conclusions and perspectives
About

• Walter RUDAMETKIN
  - PhD student at Université de Grenoble

• João AMÉRICO
  - PhD student at Université de Grenoble
  - BSc at UFRGS (2010), MSc at Université Joseph Fourier (2010), Engineer Degree at ENSIMAG (2009)
Context

Dynamic Adaptive Applications
- Architecture evolution
- Software maintenance

Real-time Applications
- Deterministic execution
- Low jitter

September 2010 AMÉRICO, RUDAMETKIN – Predictability vs. Dynamism: Managing dynamic real-time applications
State-of-the-art

RTSJ: Real-time Specification for Java
• Issues: garbage collection, dynamic class loading, thread scheduling, etc.

Dynamic Evolution/Adaptation
• Architecture modification at runtime

Real-time dynamic adaptive software
• Based on QoS objects (QoSkets), modes (SOFA-HI/Blue-ArX), and real-time adaptations for CCM (CIAO, Cardamom).
State-of-the-art

Real-time OSGi

• Works focused mainly on isolation issues: ARFLEX Project, [Richardson, 2009], AONIX’s Real-time OSGi model

• Industry initiatives: Oracle/BEA’s WebLogic Real-time, Integration between Perc and mBS
Problem Identification

• OSGi platform is inappropriate for real-time applications
  - Memory issues
  - Scheduling issues
  - Isolation issues
  - Runtime software evolution
Simple Use Case

Motion Detection System

Security Camera
\[ T_{Frame} = 4 \text{ ms} \]

Security Camera
\[ T_{Frame} = 6 \text{ ms} \]

Security Camera
\[ T_{Frame} = 3 \text{ ms} \]

Display Application
Non real-time

Motion Detection System
Real-time
\[ \Sigma T_{Frame} \leq 10 \text{ ms} \]

Update/Reconfiguration

Notation
- Required Service
- Provided Service

September 2010 AMÉRICO, RUDAMETKIN – Predictability vs. Dynamism: Managing dynamic real-time applications
Suggested Approach

• Distinction between critical and non–critical code
  – Architecture freezing policy
  – Dynamic Real–time SLA
Architecture Freezing

- Application = set of states
  - Each state corresponds to an architecture (service bindings)
Architecture Freezing

- Real-time processing states
  - Architecture modifications forbidden
Service Level Agreement

![Diagram of Service Level Agreement]

- **Service Registry**
- **SLA**
- **Needs**
- **Contract**
- **Monitor**

**Notation**
- Red: Required Service
- Pink: Provided Service
Real-Time Dynamic SLA

- Extension to the D–SLA model [Touseau, 2010]
  - Task type
  - Period
  - Worst case execution time (WCET)
  - Resource Utilization
  - Priority
Implementation

• iP0JO component model extension
Validation

- Architectures frozen during real-time processing states
- SLM not implemented
Limitations

• One real-time application at a time
• Unknown update times
• Component characterization
  – Resource utilization measures
• Overhead
Results

• Architectural Freezing solves:
  – Dynamic update
  – Service interruptions
    • but not disappearance of physical devices

• Dynamic RT–SLA solves:
  – Service admission
    • based on resource consumption, deadlines, ...

• Both require modifying apps (explicit notifications)
THANK YOU FOR YOUR ATTENTION!