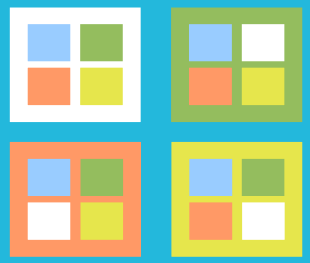
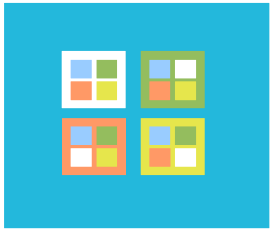


KydoSoft
High Integrity RT-Java DDS

Vincenzo Caruso, **Angelo Corsaro, Ph.D.**
{vincenzo,caruso, angelo.corsaro}@kydosoft.com



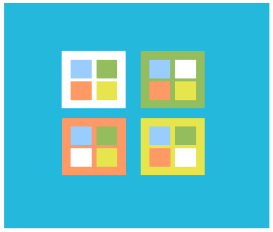
Agenda

- **Overview**
- **Architectural Highlight**
- Performance
- Concluding Remarks

Trends in High Integrity Systems

- A large class of embedded real-time systems, such as Avionics Systems, Weapon Systems, Nuclear Reactors, etc., are classified as High-Integrity

  - These Systems are rather expensive to develop due to:
    - strict certification requirements (DO-178B, SWAL, ...)
    - limited number of HI-enabled technologies
  - In recent years, R&D and standardization efforts such as DO-178C, RTSJ, HI-RTSJ, etc., are being directed toward improving the productivity of these systems
  - However, one of the consistently missing piece of the puzzle has been the support for publish/subscribe



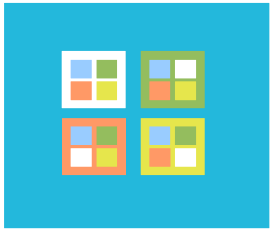
DDS & High Integrity Systems

- The current efforts in improving the technology base for HI Systems has been somewhat lacking with respect to distributed middleware, and more specifically data distribution
- The full DDS has a series of complex, *e.g.*, rich set of QoS, and dynamic features, *e.g.*, dynamic discovery, which are not necessary for small static high integrity real-time systems
- Too complex for High Integrity -- **Hard to Certify**
- Lacks of Hard Real-Time Support:
  - No way to set the priority of a publication
  - No way to support publication scheduling
  - Discovery is not time-predictable

KydoSoft DDS

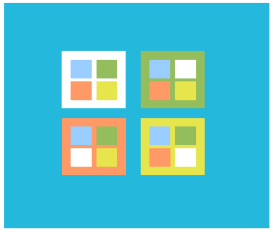
- Real-Time Java, and the upcoming HI-RT-Java are becoming more and more relevant in the avionics market, and especially UAVs

- Thus, a natural choice as a platform for a HI-DDS is HI-RT-Java!
- KydoSoft DDS address these issues by:
  - Excluding the most complex and dynamic features
  - Defining a *Reference Architecture*
  - Defining a priority and concurrency model



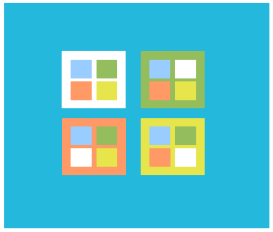
Why RTSJ?

- Use a Safe, Efficient and Certifiable Platform
- RTSJ (JSR-1), the Real Time Specification for Java provides two High Integrity Profiles:
  - Ravenscar Java (from Ravenscar ADA and SPARK)
  - Safety Critical Java, defined by JSR-302 leaded by Doug Locke
  - Both aims to be DO-178B Level-A or SWAL-1 compatible. They can fly!
- RTSJ adoption: Boeing, IBM, Siemens AG, Rockwell Collins, …



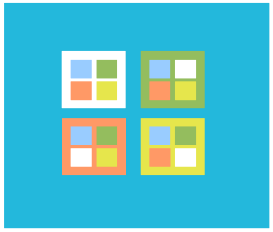
Agenda

- **Overview**
- **Architectural Highlights**
- Performance
- Concluding Remarks



Key Architectural Features

- KydoSoft DDS has been **architected** so **to exploit** the full features of **DDS and Real-Time Java**
- It fully **exploits** the **RTSJ memory model** to ensure predicability and avoid tracing garbage collection
- **Removes dynamic features** which could introduce timing **unpredictable timing behaviour**
- Introduces **Hierarchical End-to-End Priority Model**
- Concurrency model **bounds priority inversion** and minimizes context switching

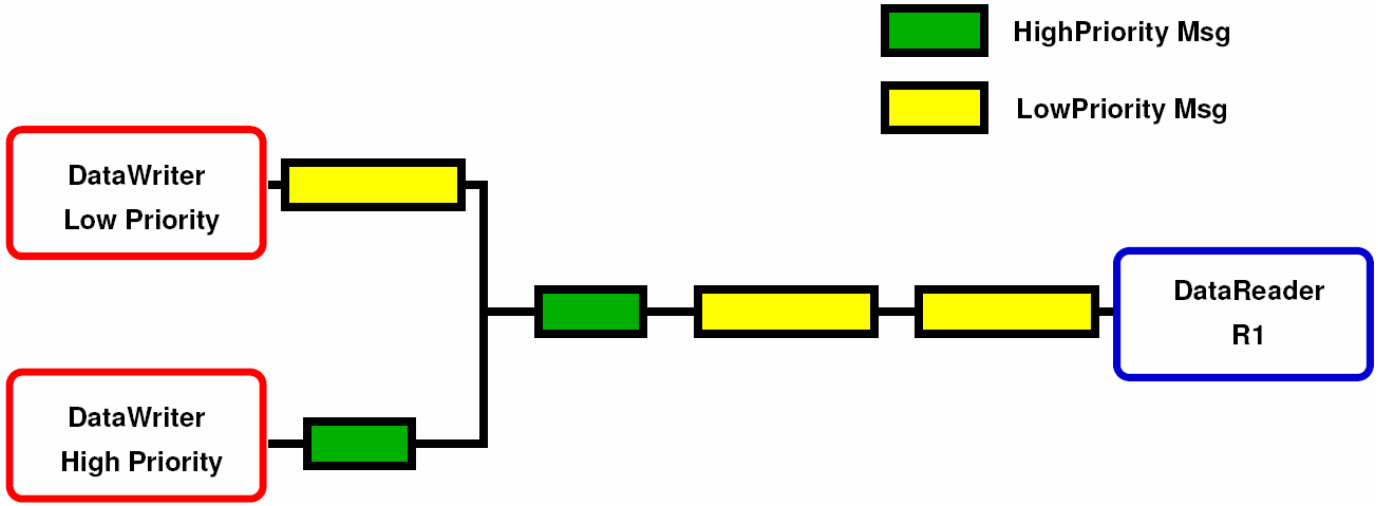


Agenda

- **Overview**
- **Architectural Highlights**
  - **Priority**
  - **Memory Model**
- Performance
- Concluding Remarks

The Need for Priority

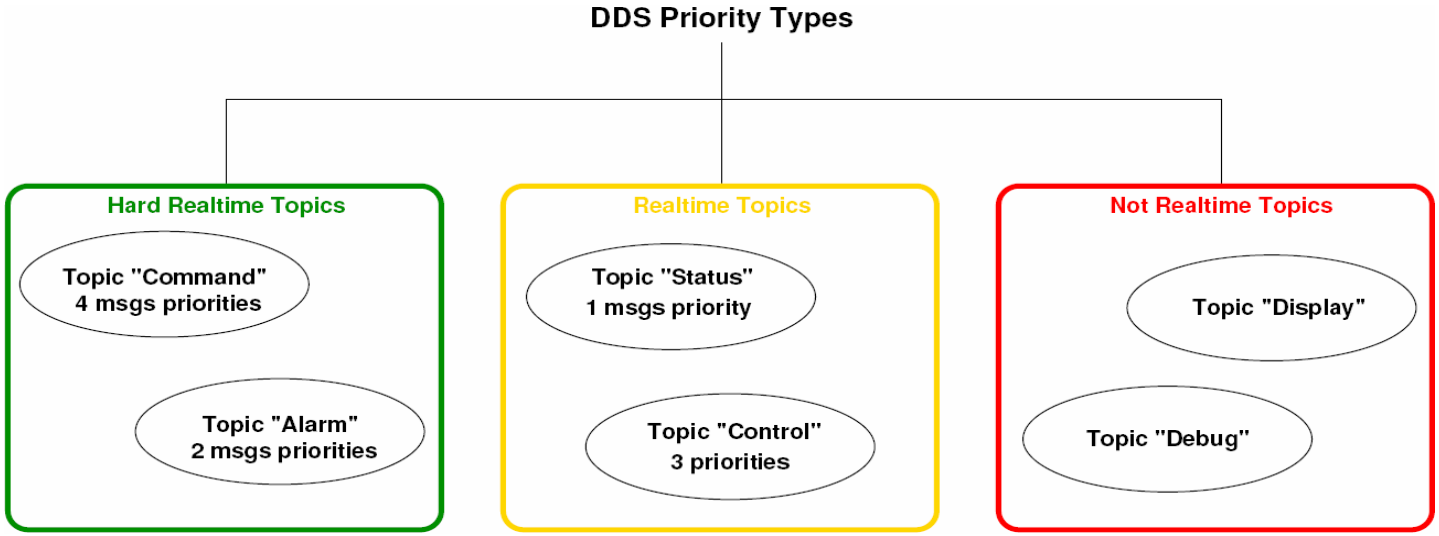
- **Problem**: In the current DDS there is no way to specify the priority of a publication
  - With DDS you can only set the priority at the transport level
  - This, on non priority preserving transports, can lead to Head Of Line Blocking (HOL) phenomenon at reception-time!



- **Solution:**
  - **Definition of a Hierarchical Priority model that takes care of both topic and publication priority**
  - **Definition of a concurrency model that supports the priority model**

A Hierarchical Priority Model

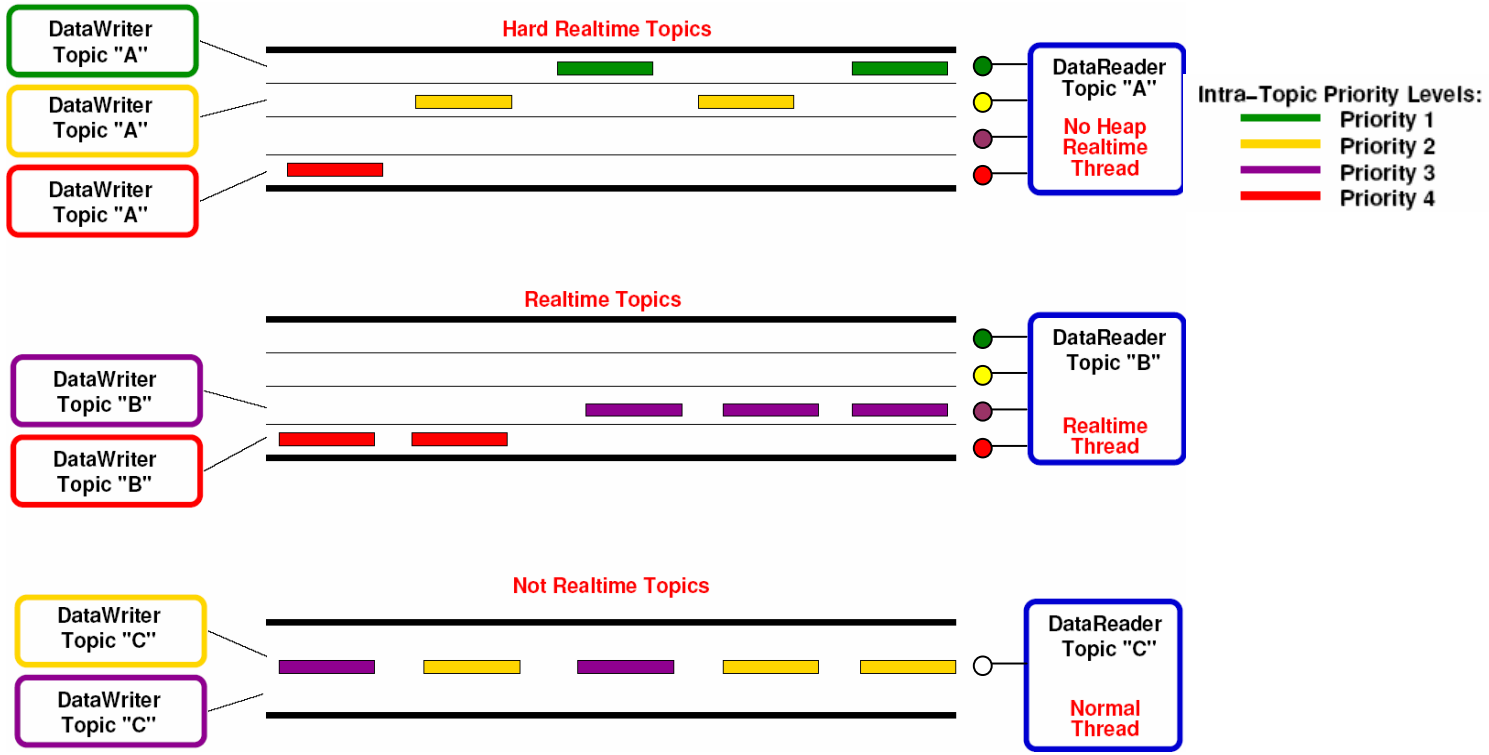
- **Three kinds of Topic**: Hard Real-Time, Real-Time, Not Real-Time
- Real-Time Topics has two levels of priority
  - Topic level (by a Topic QoS)
  - Publication level (by the new `write(data, priority)` API)
- No Priority Inversion inter-topic and intra-topic



- No Priority Inversion inter and intra-topic
- Virtual channels enforce the priority end-to-end, even with non-real-time transports

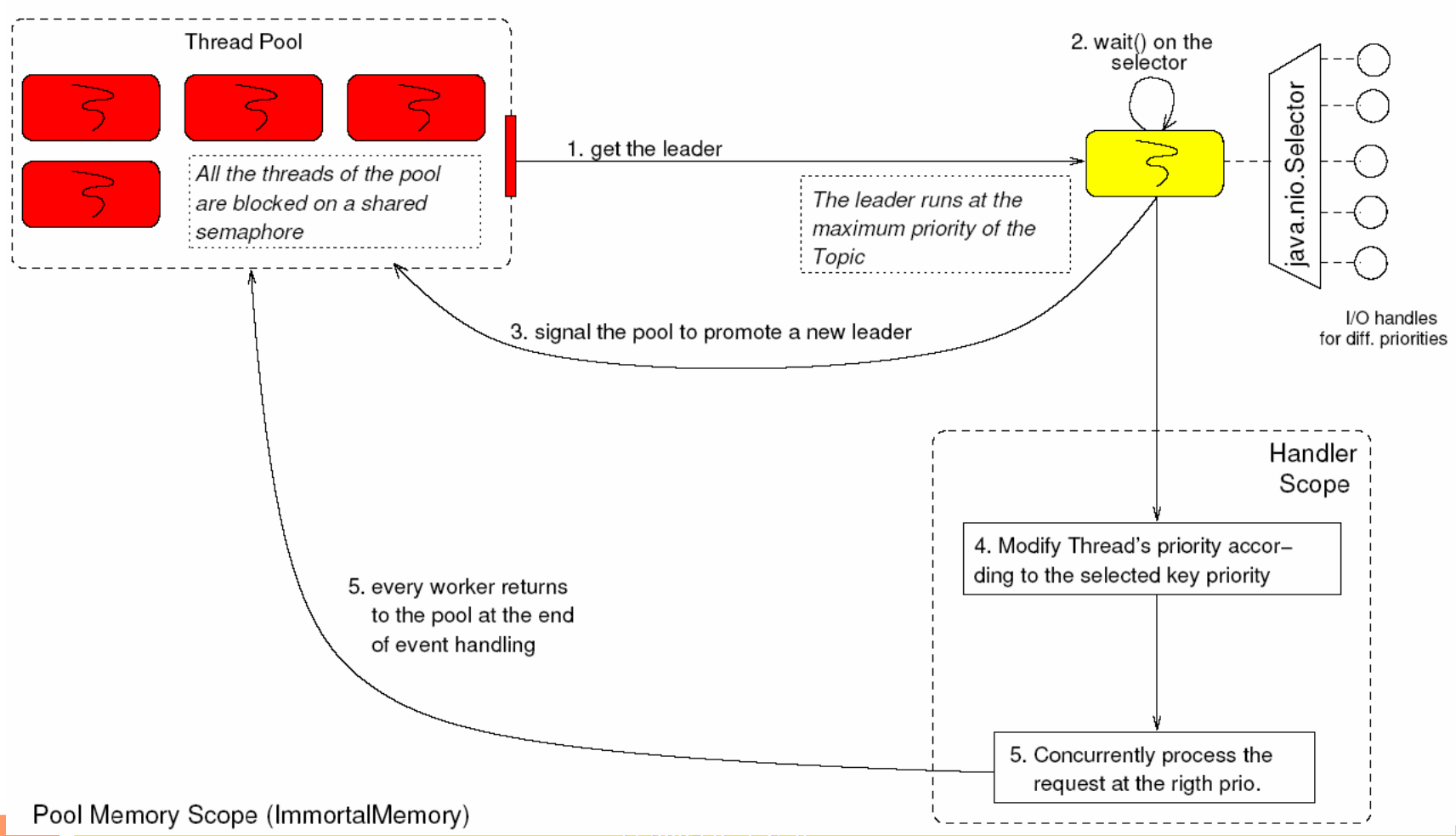
Realize the Priority Model

- Publications are addressed by the middleware to the correct Subscriber Endpoint
- An endpoint maintains a queue of requests of the same priority



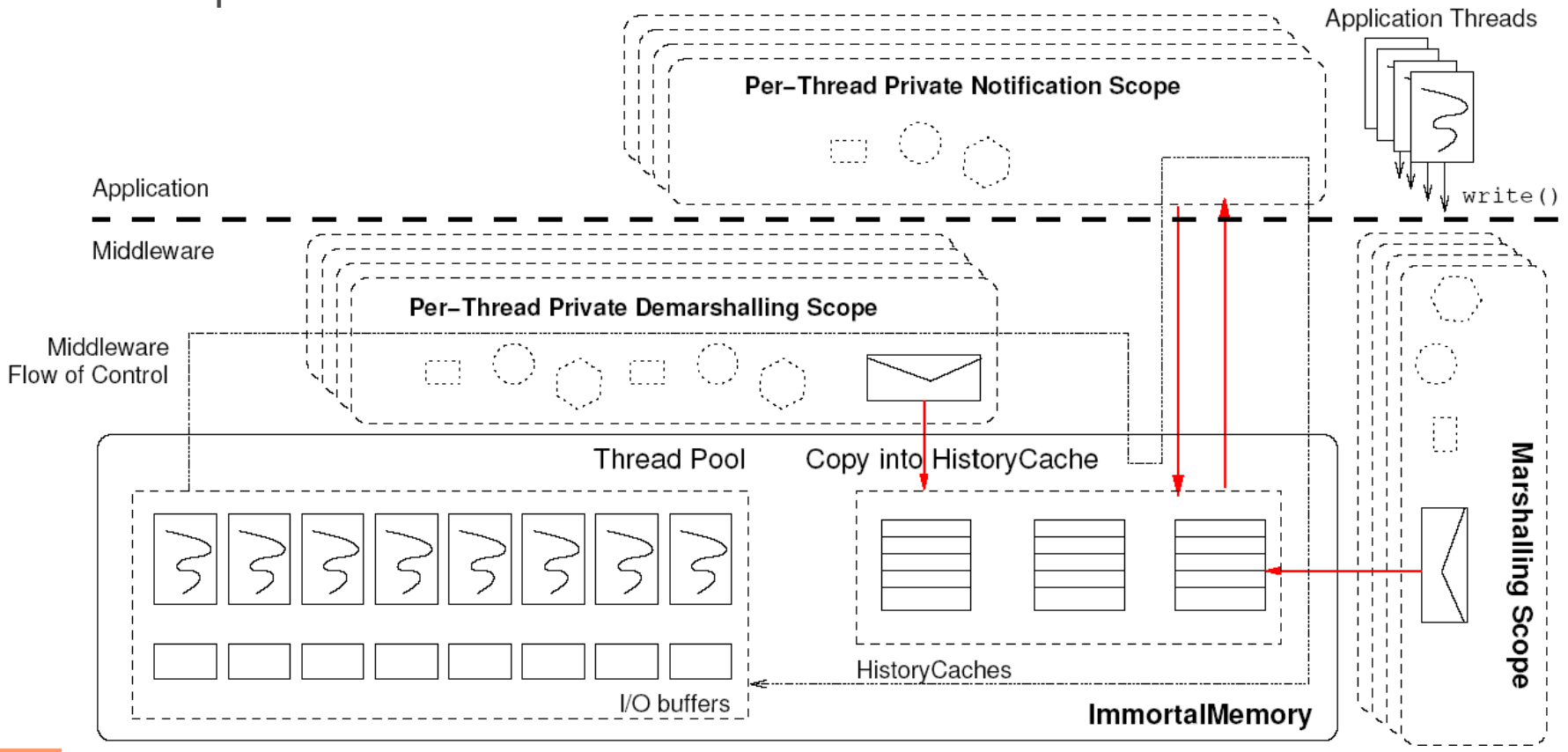
Implement the Priority Model

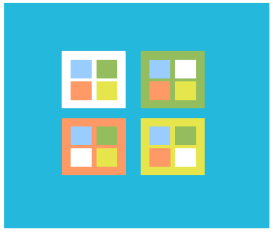
- **Problem**: It is highly inefficient to have one or more threads bound to a single socket, some of them can be inactive while some can be stressed!
- **Solution**: Use a modified version of the Leader/Follower Pattern:



Memory Model

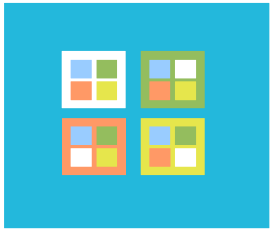
- All DDS entities are allocated in the Immortal area (a primordial area where object are **never** collected)
- Each accepting thread has a private scope to un-marshal the request and another to notify the application
- Each sending thread has its own private scope to marshal the request





Agenda

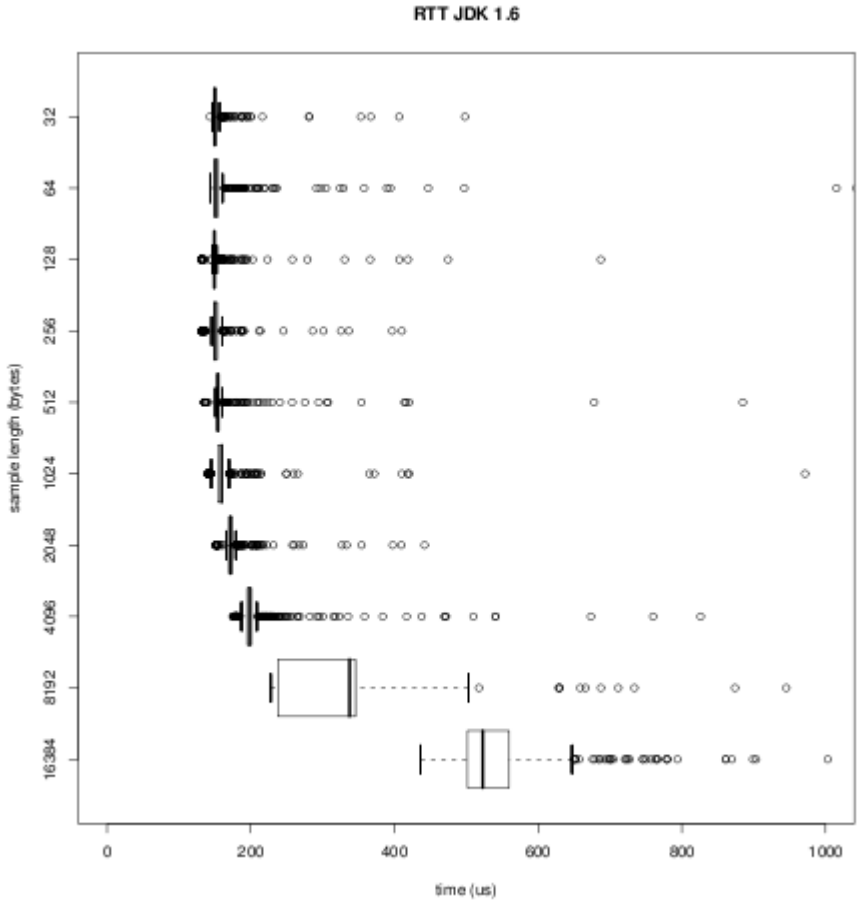
- Overview
- Architectural Highlights
- Performance
- Concluding Remarks



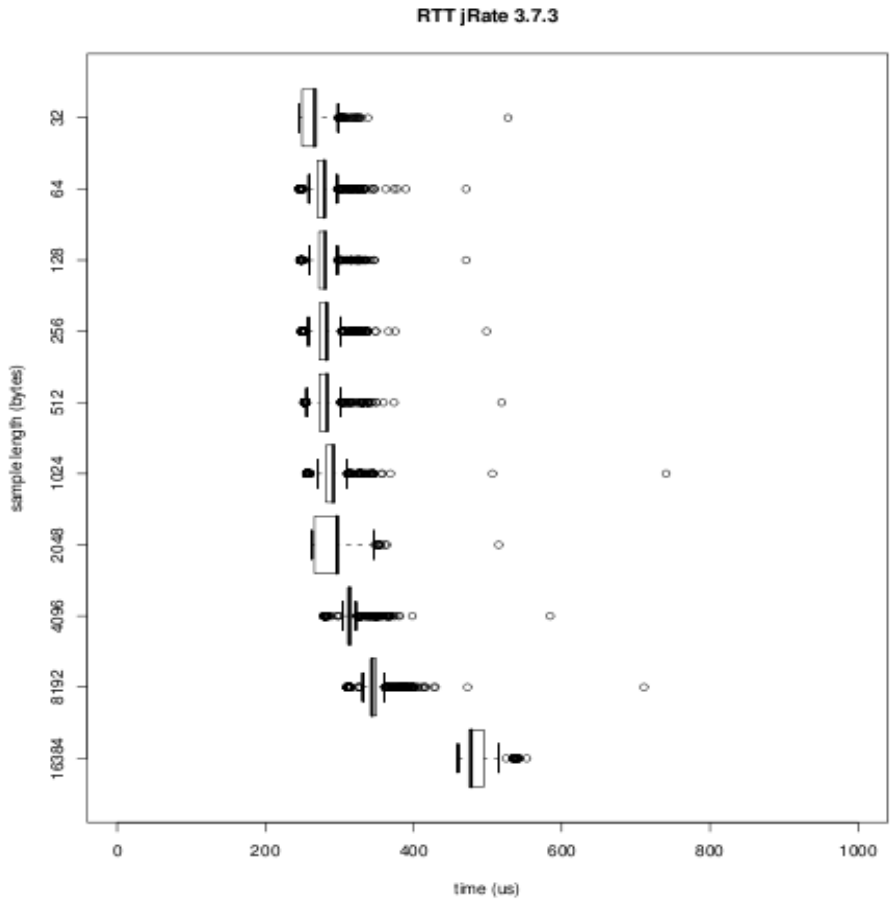
Experimental Results

- We have evaluated initial performance of KydoSoft RTJ-DDS
- The experimental testbed was based on:
  - HW – Pentium Duo @ 1.8GHz 1GB RAM
  - Network – loopback interface (127.0.0.1)
  - OS – **Linux 2.16.17** + **Debian** Sid, RTAI Patches
  - Real-Time Java: **jRate** 0.3.7.2 based on GCJ 3.3.3
  - DDS – **KydoSoft RT, configured with unreliable delivery QoS**
- **Initial performance of KydoSoft DDS is within a factor 2 of the best performing C-based implementation**
- This is an **excellent results** considering that our prototype **(1)** executes on a Real-Time Java Platform, thus incurring in extra overhead due to the safety checks, and **(2)** has not been optimized

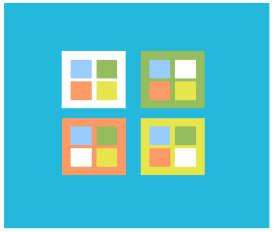
Roundtrip Latency



JDK 1.6

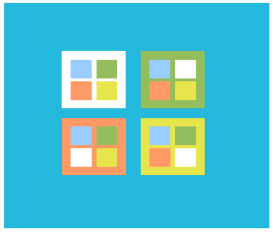


jRate 3.7.3



Agenda

- Overview
- Architectural Highlights
- Performance
- Concluding Remarks

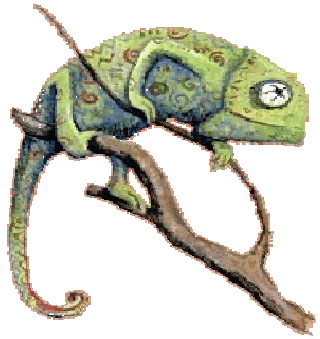


Concluding Remarks

- DDS has a great potential for penetrating the High Integrity Real-Time Market
- Real-Time Java and High-Integrity Real-Time Java provide a perfect platform for this endeavor
- KydoSoft RTJ-DDS has shown that good performance and predictability are achievable by disciplined design and architecture

References

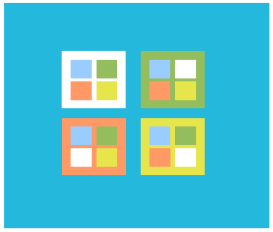
- **jRate**
  [http://jrate.sourceforge.net/]



- **KydoSoft RTJ-DDS**

  [contact: www.kydosoft.com]

- **Additional Information**
  [email: info@kydosoft.com]



Thank You!