National Aeronautics and Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Raytheon Java Day

Evaluation the RTSJ Distributed Platform

Kenny Meyer

Jet Propulsion Laboratory

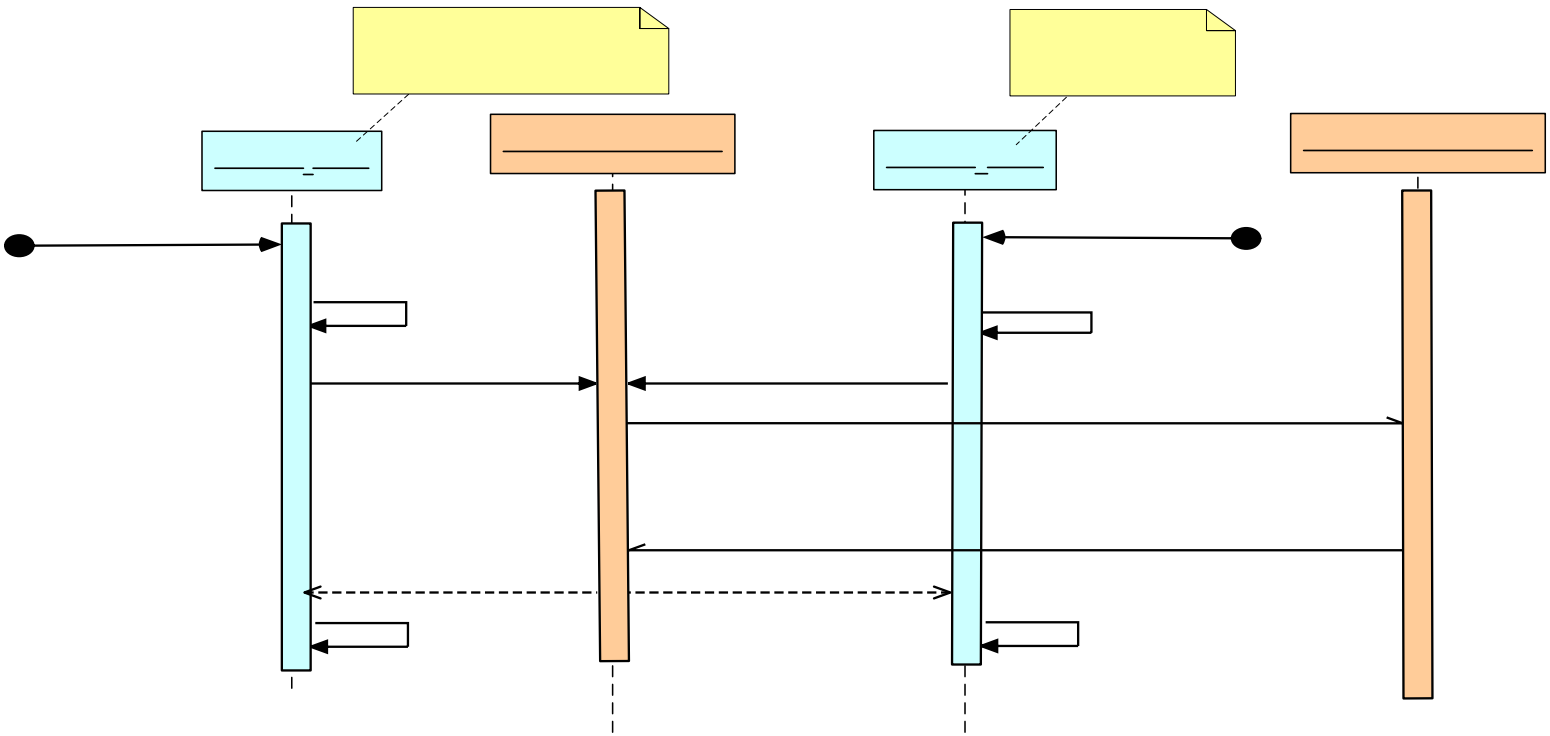
July 19, 2005

Background

- NASA funded several studies to determine if RTSJ suitable for flight systems (1999 - present). Including…
  - Golden Gate: RTSJ version of Rock 7 Rover (2003)
  - MXJ Project: JPL is Co-I to Kestrel lead NASA-sponsored effort to synthesize safety critical code for Project Constellation
- Raytheon funded a 2-Phased evaluation of RTSJ tools
  - Phase 1 (Summer ‘04)
    - Evaluated function and behavior of RTSJ JVMs
    - Identified gap: No RTSJ compatible CORBA solution available
    - Report submitted to Raytheon in August ‘04
  - Phase 2 (Spring ‘05):
    - Evaluated function and behavior of RTSJ compliant RT-CORBA ORBS
    - Evaluated the interaction between the RTSJ-enabled RT-CORBA ORBS and the RTJVM
    - Report submitted to Raytheon in May ‘05
- Objective for today: Provide overview of the Phase 2 evaluation

Experimental approach

- **Experimental Goal**: Observe effect of processor loading on a control task
- **Basic Protocol**
  - Run two publish-subscribe tasks: High-priority, foreground task and a low-priority background task over over RTSJ-enabled-RT CORBA middleware
    - High priority task published small objects @ 100ms
    - Low priority task published was varied to observe effects
  - Both tasks send objects to RTSJ server applications over RTSJ-enabled-RT CORBA ORBs



Test Cases

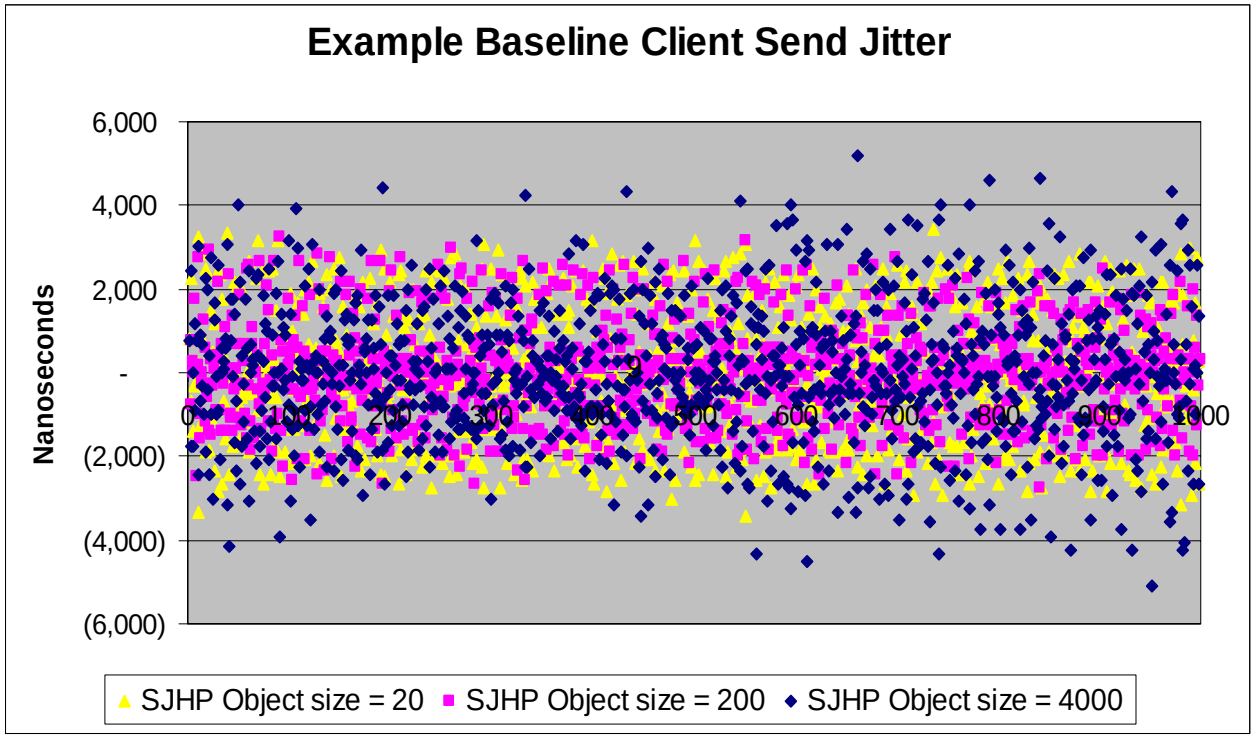
Control Case	NHRT baseline test	Measure jitter of a NHRT high priority thread with no background thread
No Garbage	NHRT Interaction Test	Measure jitter of an NHRT high priority thread publishing small objects, while an **NHRT** low priority thread is concurrently publishing objects of small, medium and large object sizes.
Garbage Producing	NHRT Garbage Collection Test	Measure jitter of an NHRT high priority thread publishing small objects, while an **RT** low priority thread is concurrently publishing objects of small, medium and large object sizes.
	RT Garbage collection Test	Measure jitter of an RT high priority thread publishing small objects, while an **RT** low priority thread is concurrently publishing objects of small, medium and large object sizes.

Measures include:

- Client send jitter
- Client receive jitter
- Round trip time

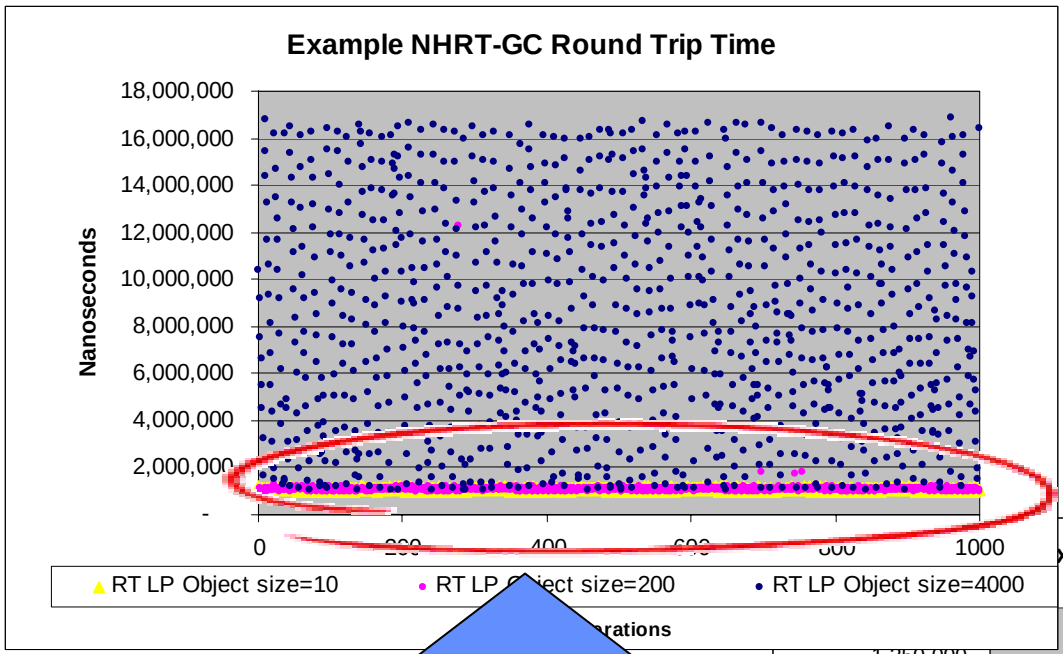
Data Collected for three RTSJ-enabled ORBs

Sample Findings



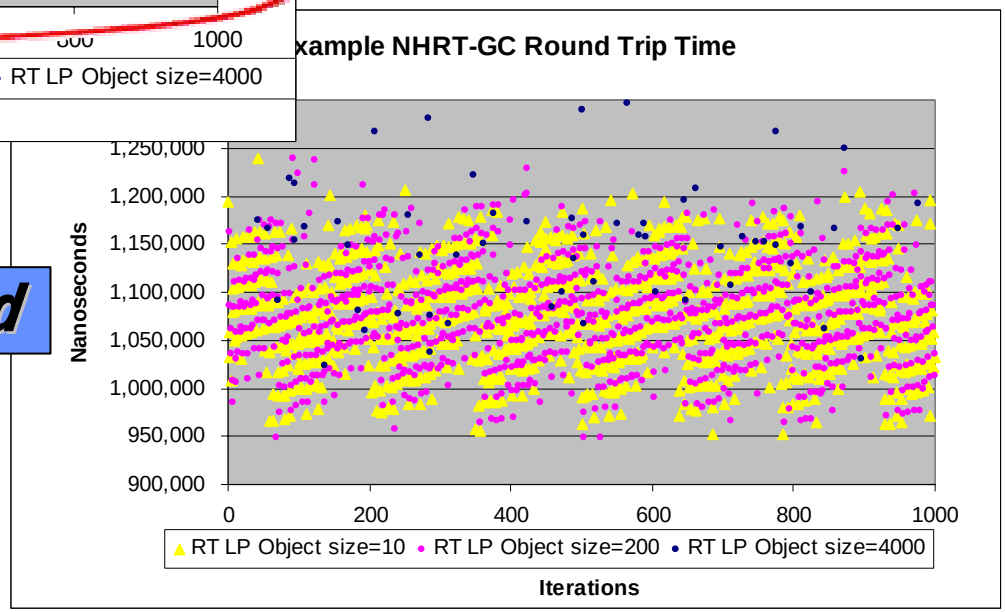
- Client send jitter
- Single thread
- No Garbage
- Very stable for small, medium and large objects (< 5 µs)`

Sample Findings (2)



- Round Trip data
- Low priority background thread producing garbage
- Considerable jitter when large objects on the net (~18 ms)
- Modest jitter with small and medium objects on the net (< 1.2 ms)

Zoomed



- Oscillation probably caused by Orb interaction with network interface

National Aeronautics and Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Key Findings

- Tech Support
  - All vendors demonstrated an extraordinary effort to support our evaluation. Each of the ORBs is backed by dedicated and competent professionals.
- Intra-ORB comparison
  - Small and medium background objects tended cause similar jitter. Large object cause significantly worse jitter.
- Inter-Orb comparison
  - One ORB was the best overall performer. The most significant differences were with large background objects. The differences were much less significant with small a medium background objects
- SMP may adversely affect network performance
  - The RTJM runs all NHRTs on a single processor with disabled interrupts. Network traffic is handled another processor that may handle RT processes. Consequently, all NHRT network traffic is handled by a node that may be running non-real time tasks.
- Maturity
  - The evaluated RTSJ-enabled ORBs are engineering prototypes. Nevertheless, the overall performance was surprisingly good—especially when compared to data collected during Phase 1.
- CORBA support for RTSJ
  - RT CORBA does not address the mapping of the memory or thread types defined by the RTSJ. No RT ORB would be expected to map client memory and thread types to servant memory and thread types.
- White-box RT clients
  - Black box style development of distributed real-time applications is not practical with the current ORBs. Successful integration will require that client-application developers have white box knowledge of the real-time servant. The current CORBA spec does not support black-box implementation of RTSJ clients

Recommendations

We recommend …

- OMG develop standards for a RT CORBA that supports real-time RTSJ semantics across the network.
- Evaluation of techniques for designing RT clients that use black-box RT servants
- The ORBs be re-tested using…
  - Client propagated priorities. This is a key step toward black-box client development
  - Ahead-of-time compilation. This is a necessary step to understand NHRT performance
- Evaluation of alternative network protocols and publish-subscribe middleware services like OMG's Data Distribution Services
- Evaluation of design strategies for the development of fault-monitoring and recovery of critical, distributed RTSJ applications.