Communications Research Centre Canada
An Agency of Industry Canada

Centre de recherches sur les communications Canada
Un organisme d'Industrie Canada

How Different Messaging Semantics Can Affect Applications Performances

Presented by: Steve Bernier, M.Sc.,

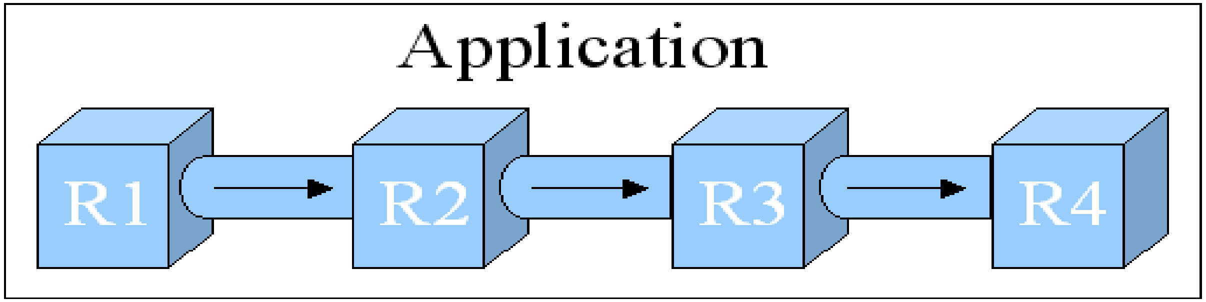
Research Manager

Advanced Radio Systems

Communications Research Centre Canada

How Different Messaging Semantics Can Affect Applications Performances

- SCA applications are made of several software components typically connected in a pipeline configuration



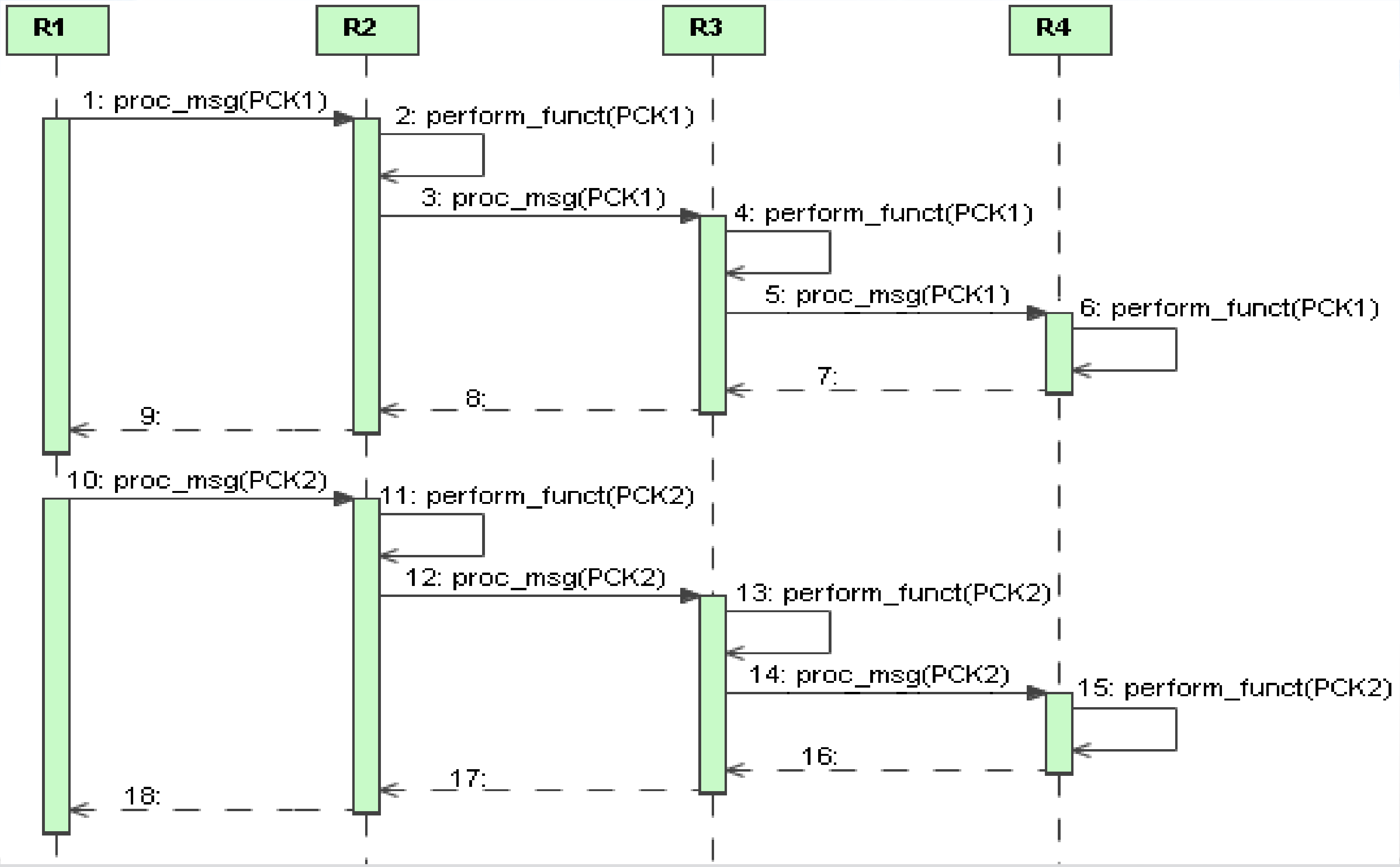
- Using the SCA, software components can be implemented by different organizations
  - Interactions between components requires a middleware
  - The middleware for SCA is CORBA

How Different Messaging Semantics Can Affect Applications Performances

- **This paper provides metrics comparing two types of CORBA interactions: One-way and Two-way**
  - Using CORBA, every interaction is transformed into a message sent from a source component to a destination component
- **Two-way interactions**
  - Source is blocked until a response is received from the destination
  - Synchronized with the target
- **One-way interactions**
  - Source is <u>not</u> blocked until a response is received from the destination
  - 3 levels of synchronization: with the middleware, with the transport, or with the server

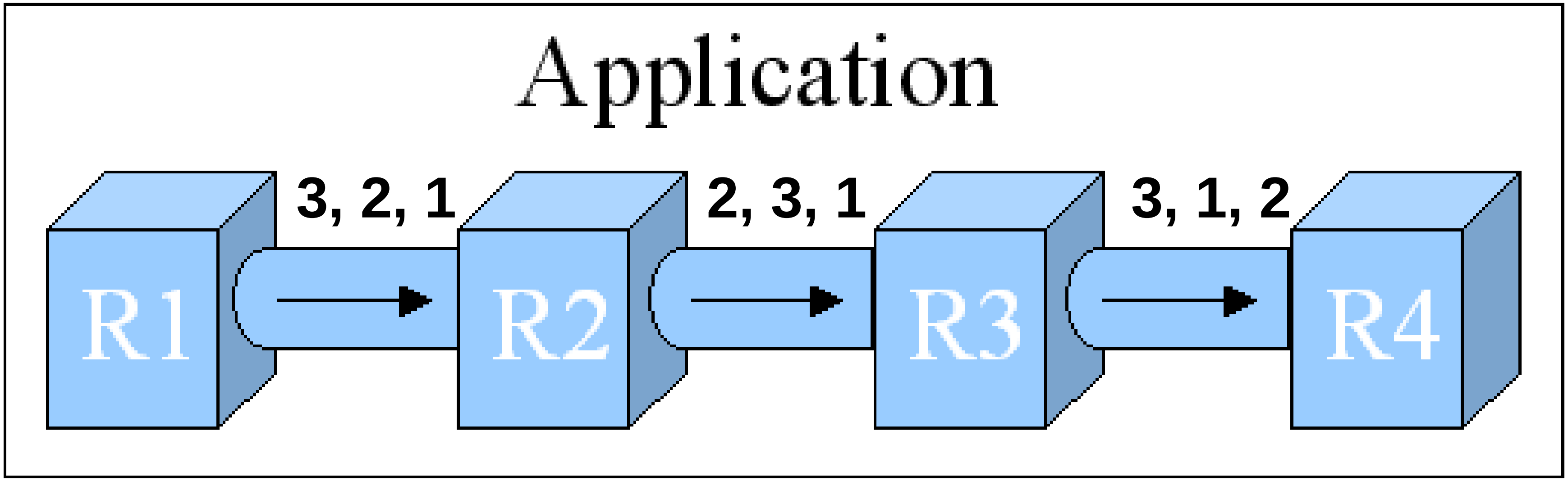
How Different Messaging Semantics Can Affect Applications Performances

- Two-way messaging can lead to the empty pipeline problem



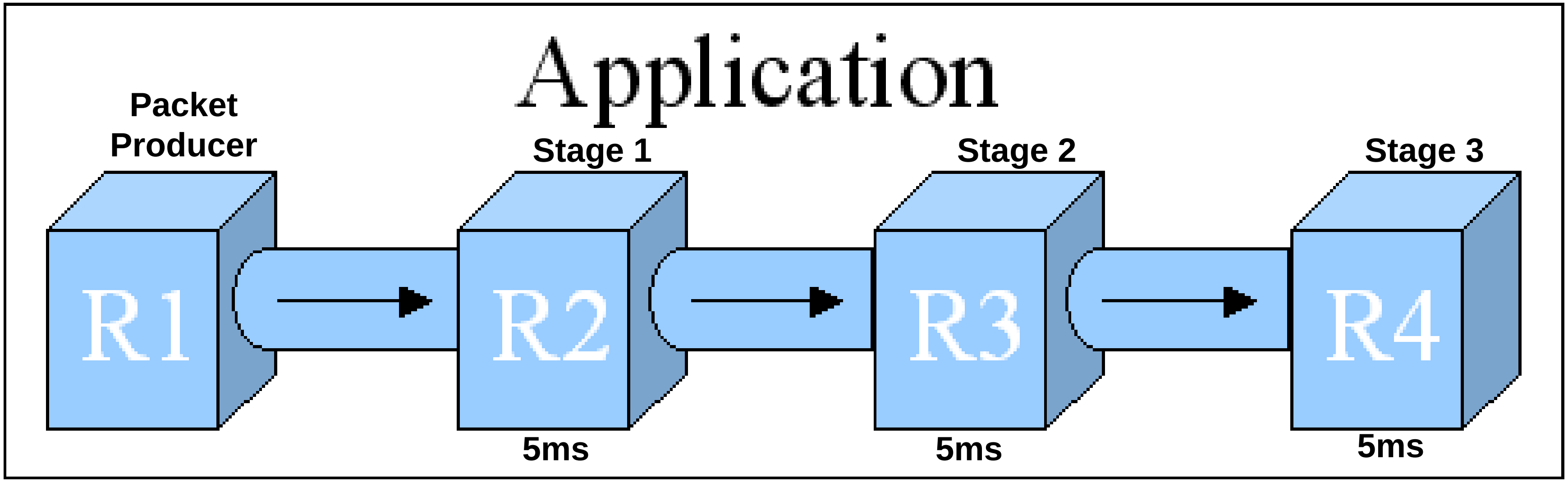
How Different Messaging Semantics Can Affect Applications Performances

- One-way messaging can lead to the packet reordering problem



How Different Messaging Semantics Can Affect Applications Performances

- **This paper provides metrics for 4 tests. All tests work as follows:**
  - Pipeline configuration of 4 components
  - The first component produces 1000 packets and sends them through a pipeline of 3 stages
  - Each pipeline stage performs 5ms of work



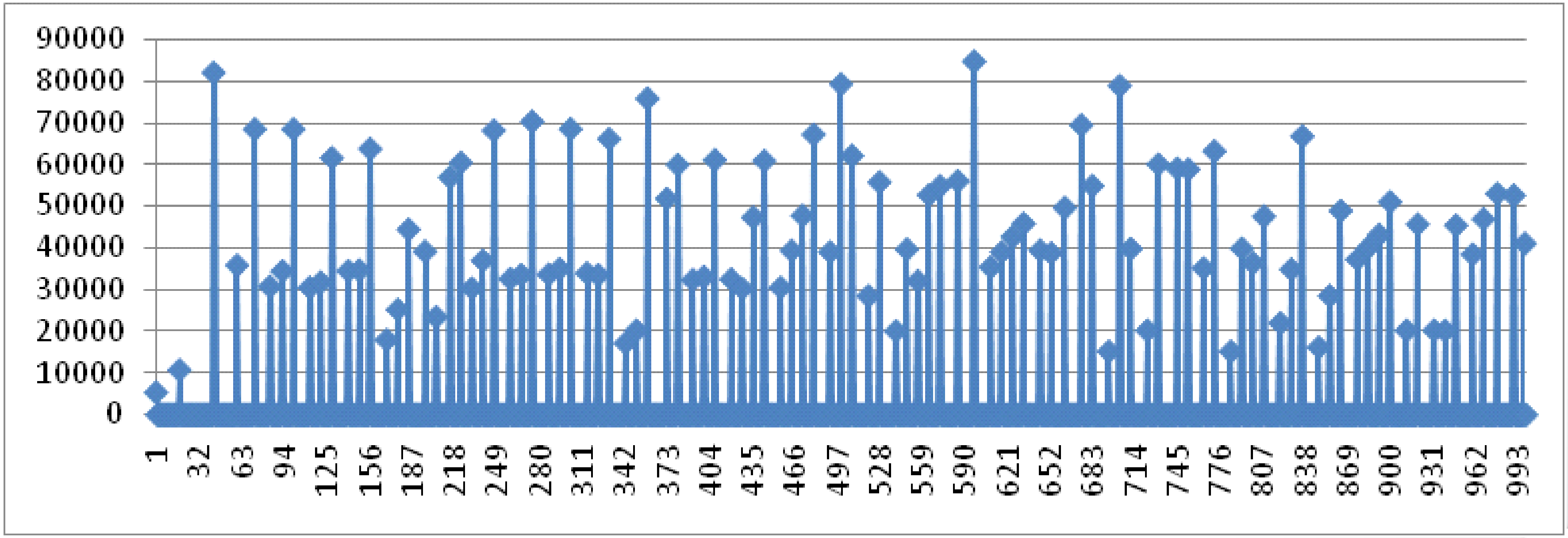
How Different Messaging Semantics Can Affect Applications Performances

- **Test #1**
  - One-way messaging, packet producer does not wait between each packet, synchronized with TCP/IP transport
  - Uses several threads in each pipeline stage
  - Causes lots of packet reordering
  - Should take less than 1000*5ms for all packets to go through the pipeline

	Stage 1	Stage 2	Stage 3
Time of last Pkt arrival	4463.20ms	4508.41ms	4513.61ms
# of Pkt reordered	315	520	612

How Different Messaging Semantics Can Affect Applications Performances

- **Test #1**
  - Time it took for the producer to send each packet to the transport
    - 10% of the packets in 44ms
    - 90% of the packets in 9usec
  - Producer was paced by the transport



How Different Messaging Semantics Can Affect Applications Performances

- **Test #2**
  - One-way messaging, packet producer waits 5ms between each packet, synchronized with TCP/IP transport
  - Uses less threads in each pipeline stage
  - Still causes some packet reordering
  - Should take around 1000*5ms for all packets to go through the pipeline

	Stage 1	Stage 2	Stage 3
Time of last Pkt arrival	5416.57ms	5421.74ms	5426.90ms
# of Pkt reordered	95	216	349

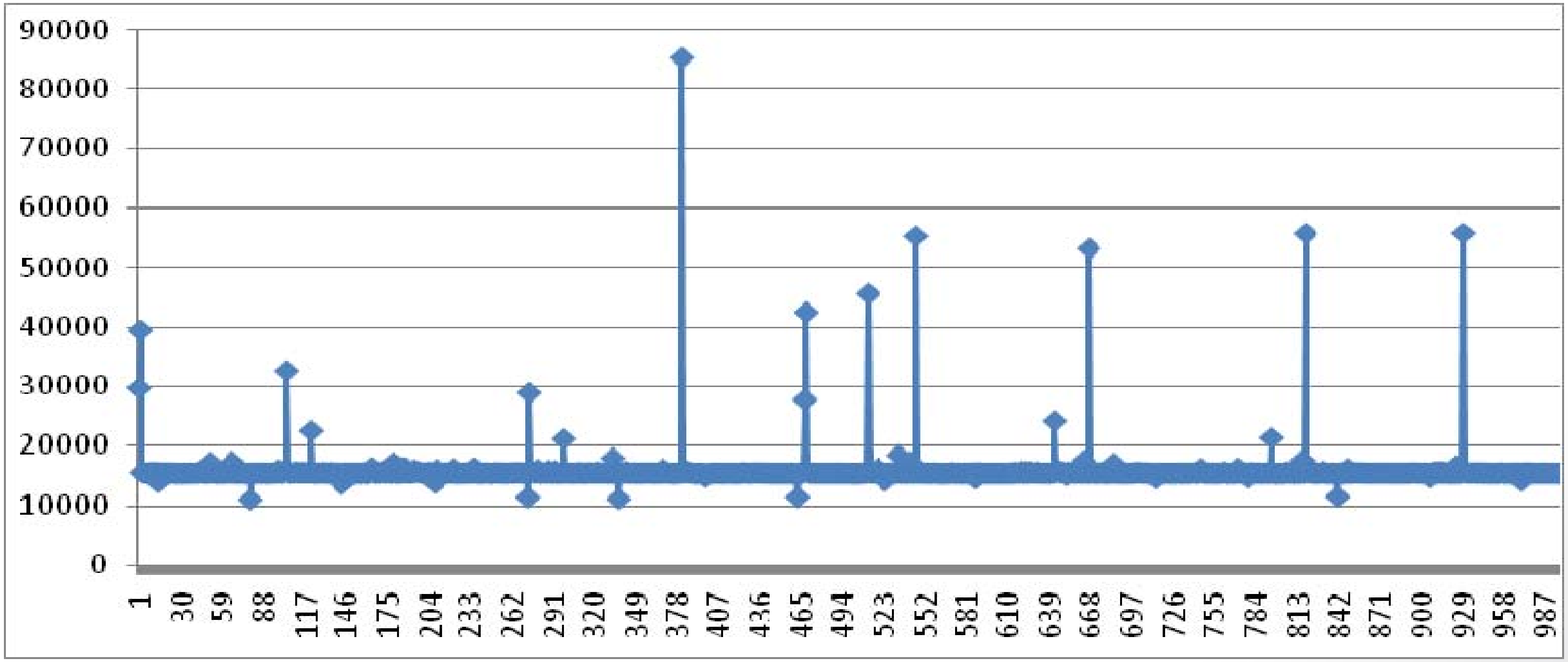
How Different Messaging Semantics Can Affect Applications Performances

- **Test #3**
  - Two-way messaging, packet producer does not wait between each packet, synchronized with TCP/IP transport
  - Causes the empty pipeline problem
  - Should take at least 1000*5ms for each packet to go through each stage of the pipeline

	Stage 1	Stage 2	Stage 3
Time of last Pkt arrival	15,684.19ms	15,684.06ms	15,683.93ms
# of Pkt reordered	0	0	0

How Different Messaging Semantics Can Affect Applications Performances

- **Test #3**
  - Time it took for the producer to send each packet to the transport
    - Average around 15ms with very few peeks
    - Producer was almost never paced by the transport



How Different Messaging Semantics Can Affect Applications Performances

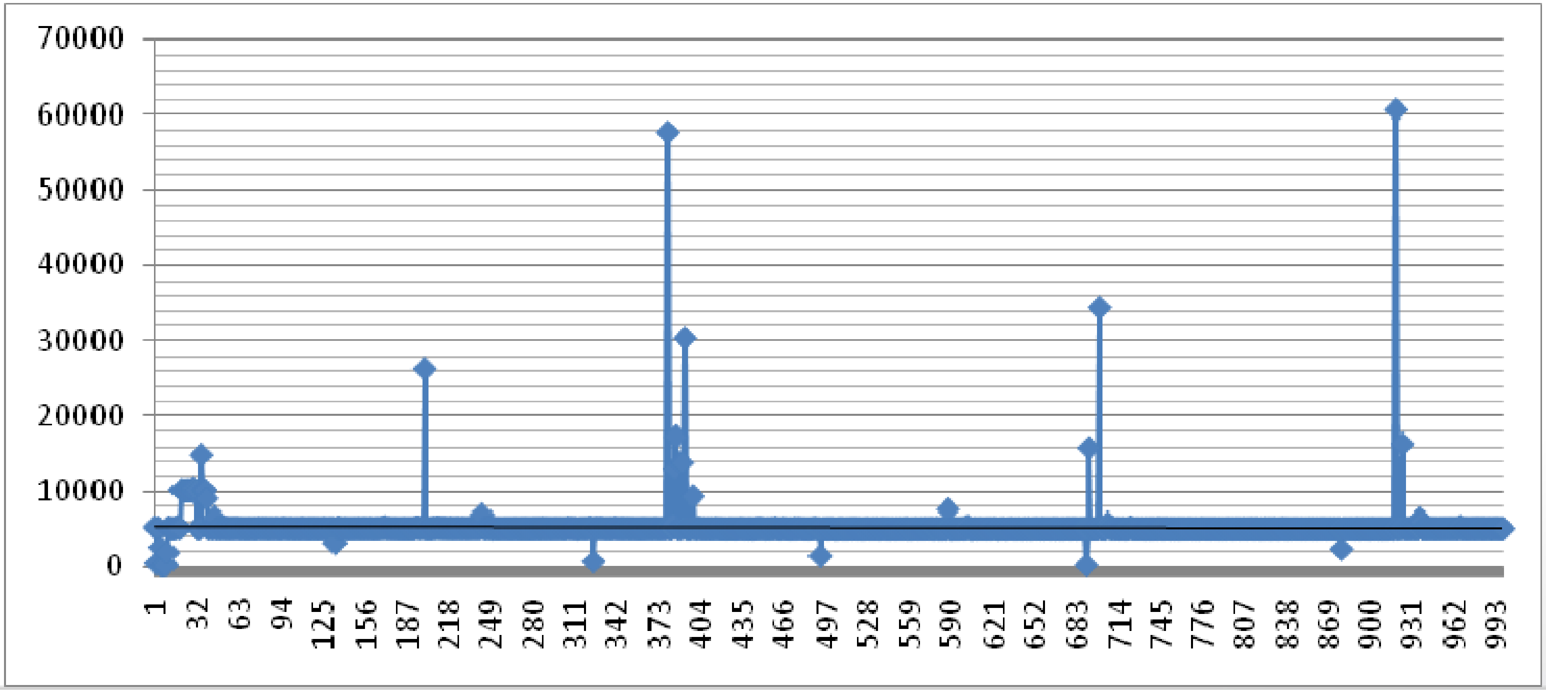
Test #4

- Two-way messaging, packet producer does not wait between each packet, synchronized with TCP/IP transport
- Each stage uses one extra thread to decouple packet reception from packet transmission
- Does not cause the empty pipeline problem
- Does not cause any packet reordering
- Performance is better than using one-way messaging with a paced producer

	Stage 1	Stage 2	Stage 3
Time of last Pkt arrival	5286.22ms	5267.73ms	5297.16ms
# of Pkt reordered	0	0	0

How Different Messaging Semantics Can Affect Applications Performances

- **Test #4**
  - Time it took for the producer to send each packet to the transport
    - Average around 5ms with very few peeks
    - Producer was not paced by the transport as often



How Different Messaging Semantics Can Affect Applications Performances

Conclusions

- One-way messaging does not necessarily offer better performances than two-way messaging
- One-way messaging causes a large amount of packet reordering
  - not be suitable for most waveform applications
- Two-way messaging naturally leads to the empty pipeline problem
- Two-way messaging with an extra thread can yield interesting performances without packet reordering
  - Simple to use since flow control does not require explicit APIs

Questions?

CRC's Achievements

- 1998 – Creates proprietary SDR architecture
- 2000 – Implements FM radio for DnD using SCAv0.3
- 2001 – Introduces the concept of Ports and Connections for SCAv1.0
- 2002 – Releases Java™ open-source Reference Implementation (SCARI)
- 2002 – First demonstration of a commercial SCA waveform (DAB™)
- 2003 – Introduces 1st commercial SCA development kit with modeling tools
- 2004 – ReleasesSCARI2 open source, JTeL Certified (97.39%) SCAv2.2 CF
- 2004 – Adds support for ORBexpress, INTEGRITY, and YellowDog Linux
- 2005 – Introduces 1st SCA Xml validator and code generator
- 2006 – Adds support for VxWorks 6.x
- 2006 – Releases new modeling tool based on Eclipse™
- 2007 – Adds support for LynxOS
- 2007 – Creates the world's smallest SCA FM radio
- 2008 – Releases new generation Core Framework : SCARI-GT
- 2009 – Adds support for TimeSys Linux
- 2010 – Creates the first SCA virtual front panel

CRC's Achievements

- **1998 - Designed a proprietary SDR architecture**
- **2000 - Implemented a proof of concept SCA SDR for the Canadian Department of National Defence**
  - FM Line of sight application running on DSPs (TI C6201)
  - Implemented a SCAv0.3 Core Framework
- **2002 - Released a Java™ open-source SCAv2.1 Reference Implementation (SCARI)**
  - Sponsored by the Software Defined Radio Forum
  - Peer reviewed by a SDR Forum oversight committee:
    - MITRE JPO staff, US AFRL, L3-Communications, Mercury Computer Systems, Sun Microsystems, Space Coast Systems

CRC's Achievements

- **2002 - Demonstrated a Digital Audio Broadcast (DAB™) application**
  - First demonstration of a commercial SCA SDR application
  - Implemented in C++ and runs with SCARI

- **2003 – CRC releases its first commercial product called SCARI-Hybrid**
  - Java™/C++ SCA Core Framework with GUI tools

CRC's Achievements

- **2004 - CRC selected by SDR Forum to develop a JTeL certified Core Framework**
  - **Done in partnership with JTRS/JPO, JTRS/JTEL, NASA, Mercury Computers, Rohde and Schwarz, ISR Technology 19**
  - **Open source Java™ implementation of SCAv2.2**
  - **Includes a one-channel push-to-talk FM application**
  - **Demonstration performed at SDR'04 meeting**
  - **Status: On-site certification process completed in only 5.5 days (2005, June 7-8-9-10, 14-15)**
    - **Meets 635 of the 652 SCA requirements for an unprecedented result of 97.39%**

CRC's Achievements

- **2004 – CRC's first fully embeddable Core Framework – SCARI++**
  - Implementation of the SCAv2.2 specification
  - Support for Linux, Yellow Dog, and INTEGRITY
  - Support for x86 and PPCs
  - Support for CORBA: TAO and ORBexpress

CRC's Achievements

- **2004 – First SDR platform using dynamic partial reconfiguration of an FPGA**
  - Allow more than one application to "share" the FPGA
  - Can switch applications without stopping the FPGA
  - Platform developed by ISR Technologies in collaboration with Xilinx and CRC

CRC's Achievements

- **2005 – Code Generation and XML validation**
  - CRC was 1st to provide modeling tools in 2003
  - CRC was also 1st to offer automated source code and XML generation from graphical models
  - CRC also became 1st to offer reverse engineering and validation of SCA XML domain profiles
  - Latest version of the modeling tools is provided as an Eclipse™ plug-in

CRC's Achievements

PENTEK
Setting the Standard for Digital Signal Processing

- **2005 – Added support for more embedded SDR development kits**
  - Added support for the Pentek 2510 SDR Kit
    - Complete software radio transceiver solution

- **2006 – Added support for more embedded operating systems and processors**
  - Added support for VxWorks and ARM processors

WIND RIVER

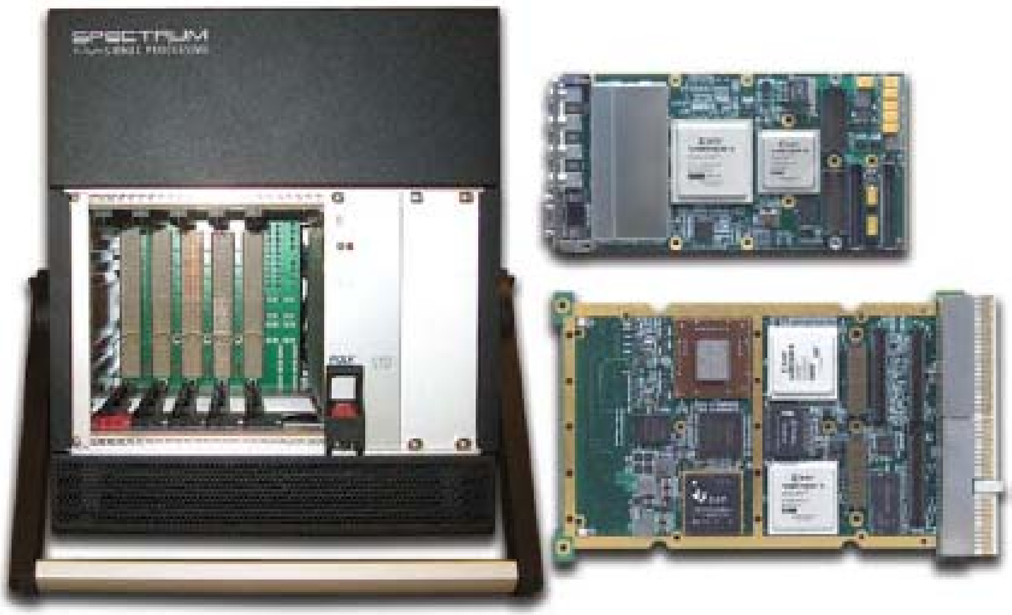
ARM

CRC's Achievements

- **2006 – Added support for the Lyrtech SFF SDR development kit**
  - Partnered with Lyrtech Signal Processing to offer support for the Small Form Factor (SFF) development kit
  - 1st platform to offer SCA integration ORB with DSP/FPGA
    - ORBexpress on DSP and on FPGA

CRC's Achievements

- **2006 – Added support for the SDR4000 development kit**
  - **Partnered with Spectrum Signal Processing to offer support for the SDR4000 SCA SDR development kit**



CRC's Achievements

- **2007 – Added support for more embedded operating systems and processors**
  - Added support for LynxOS and Marvell's PXA270 processor

- **2007 – Demonstration of the 1st SCA Radio using world's smallest computer**
  - FM SCA Radio demonstration using a Gumstix

CRC's Achievements

- **2007 – World's First High-Capacity Tactical Radio based on the SCA**
  - AN/GRC-245A radio deployed by the US Army as part of the Increment-1 of WIN-T
  - Since deployed by the Canadian Forces
  - Ultra has shipped close to 2000 units
  - Uses CRC's SCARI++ Core Framework

CRC's Achievements

- **2008 – New Generation Core Framework SCARI-GT**
  - Results of 18 months or R&D
  - Implements 6 optimization features for fast boots using small memory footprints

- **2009 – Core Framework for smaller form factors**
  - Adds support for TimeSys Linux on PPC

CRC's Achievements

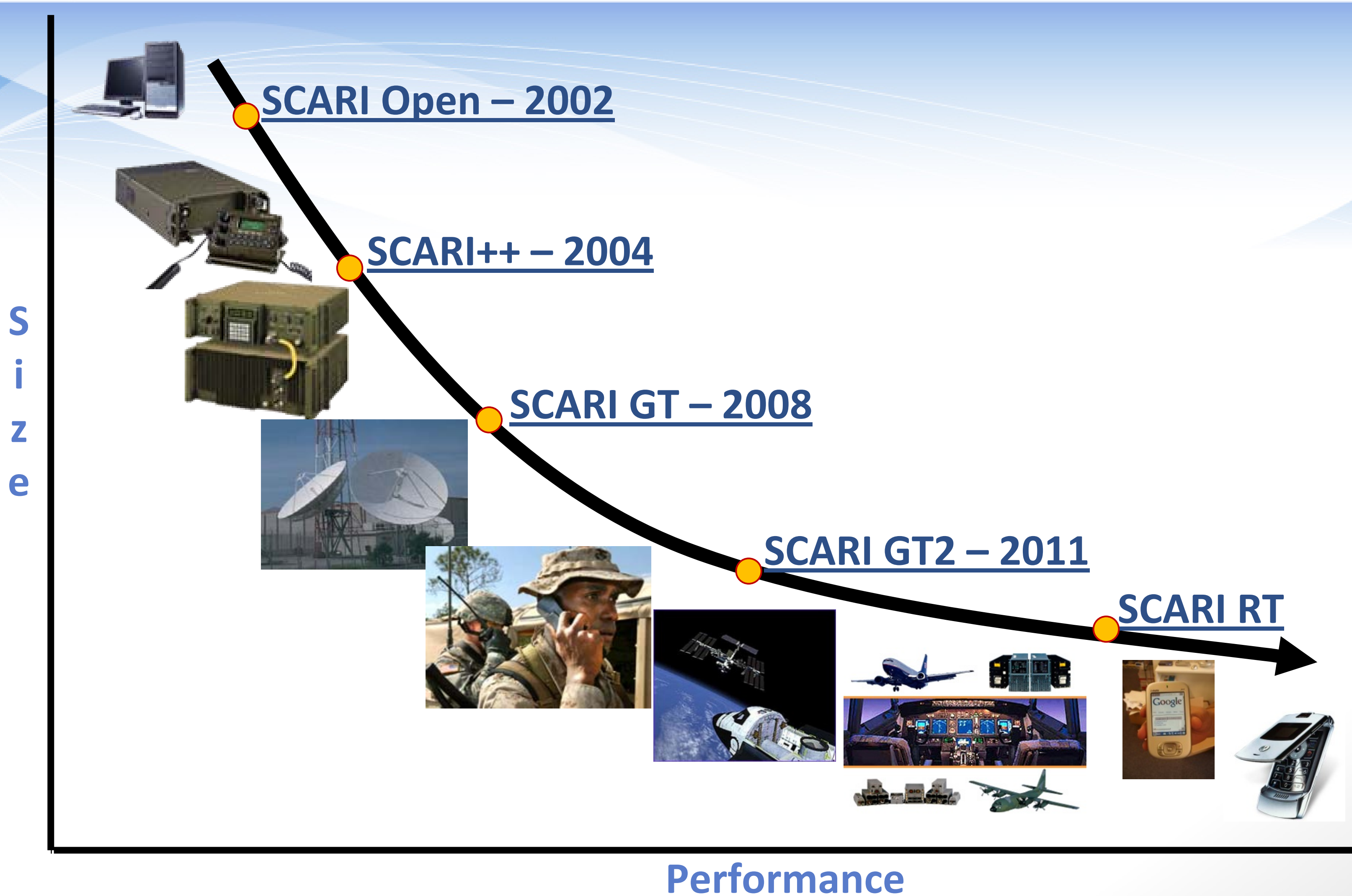
- **2010 – Adding support for new operating systems**
  - Added support for Monta Vista Linux
  - Adding support for Microsoft™ Windows™
  - Adding support for QNX Neutrino

CRC's Achievements

- **2010 – Created the first SCA Virtual Front Panel**
  - Virtual Front Panel all controlled via SCA event channel and SCA PropertySet
  - Everything functional, LCD, Key Pad, and LEDs
  - Remote control HCLoS AN/GRC-245 radio from Ultra Electronics TCS

Ultra ELECTRONICS

Communications Research Centre Overview

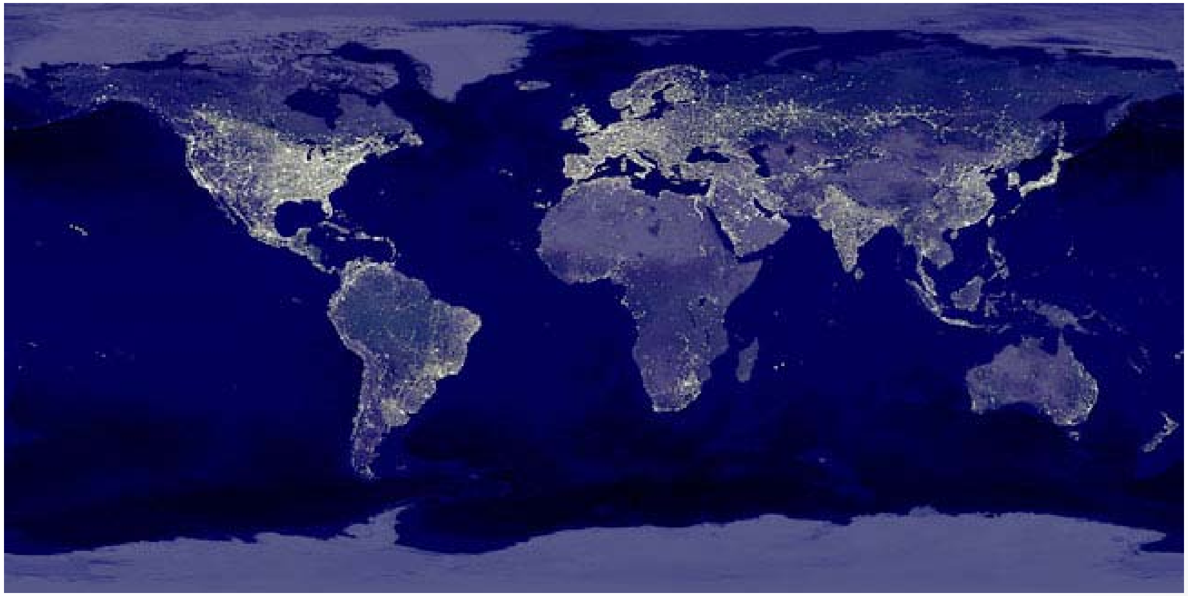


In Summary

- **CRC’s recognized as a leader in the SCA community**
  - Has been leading for more than 10 years
  - Has a long list of industry firsts
  - Influenced every version of the specification since SCAv0.3
  - Is chairing the SDR Forum SCA Working Group
    - Working on an SCA interpretation guide
    - Working on APIs
  - CRC has the largest team of engineers dedicated to the SCA
    - CRC does not sale radios

In Summary

- **CRC's SCA technologies have been licenced to more than 40 organizations in 15 countries**
  - SCARI++ is the only COTS Core Framework to have been deployed in the battlefield
  - Customers in North-America, Europe, Middle-East, and Asia



- THE END -