



Leveraging DDS-RTPS Interoperability Protocol towards a Canonical tool-set

Fabrizio Bertocci, July 2009

**The Real-Time
Middleware Experts**

Fabrizio Bertocci
Principal Engineer
Real-Time Innovations
fabrizio.bertocci@rti.com

Agenda



- Introduction to RTPS
- Benefits of RTPS interoperability
- Gaps
- Conclusions

RTPS Wire Interoperability Protocol



- DDS Interoperability Protocol formally adopted:
 - Version 1.2 initially drafted on July 2006
 - Version 2.0 adopted in June 2007
 - Version 2.1 adopted in June 2008
- 3-vendor demonstrated interoperability on March 2009 and July 2009
- Open source dissectors (Wireshark) available
- Already deployed in many systems
 - RTI Implementation available since 2007

What does the protocol specify?

- Discovery mechanism:
 - For participants on the network
 - For topics
 - For endpoints (writers and readers)
 - For quality of service associated with topics
- Data encapsulation:
 - For any given type how it is encoded on the wire
- Transport:
 - How to encapsulate it in UDP/IP
 - Mapping of ports and multicast address
- Message formats
 - How to send data, subscribe, send acknowledgments, announce presence, etc.
- Protocol / Handshakes:
 - Timing of messages
 - How to do reliability on top of UDP
 - How to use multicast
 - How to do reliable multicast

RTPS Features (1/2)



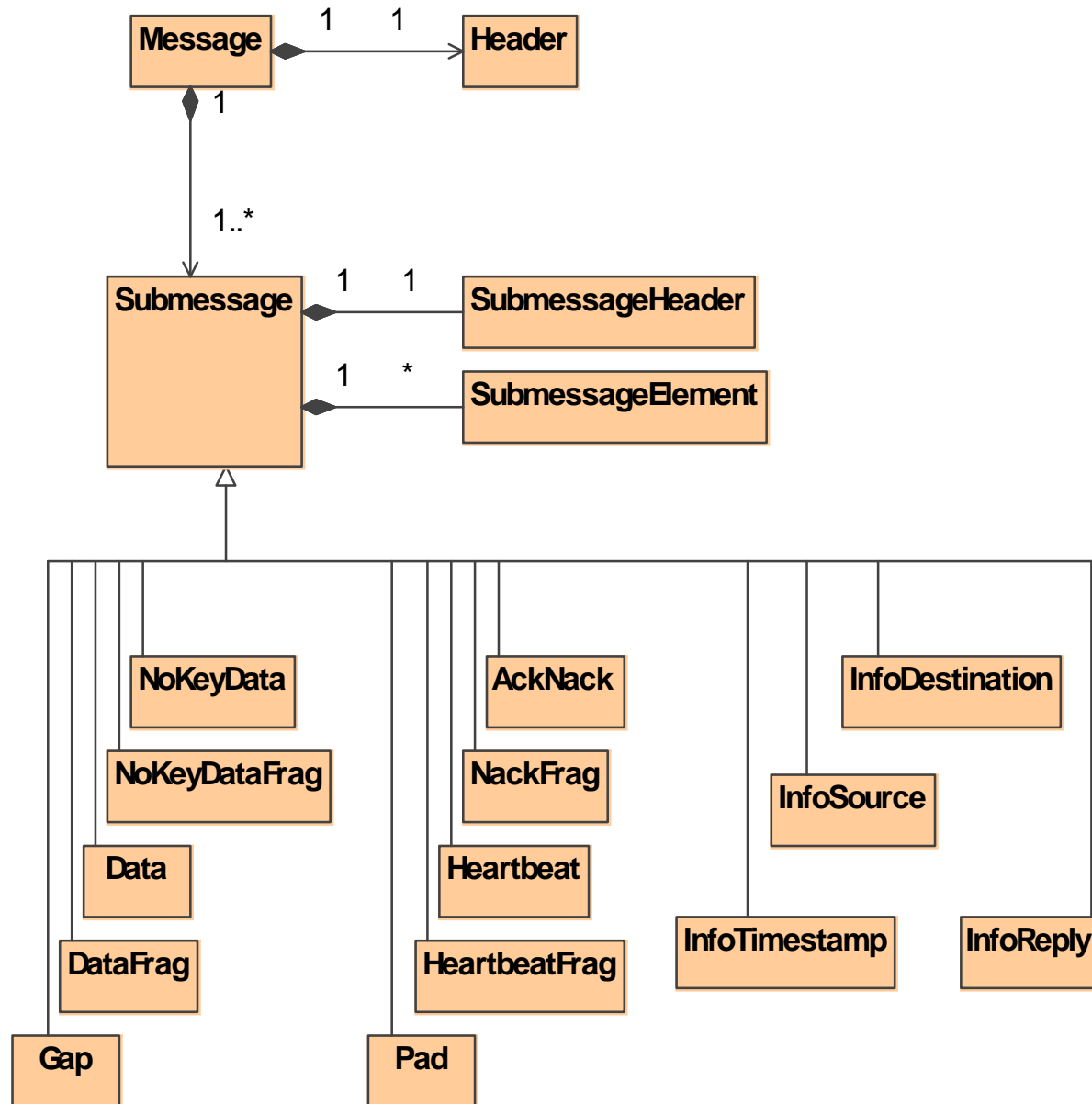
- Best Efforts and reliable protocols:
 - At most once delivery of messages for best-efforts
 - Exactly once delivery for reliable
- Multicast & reliable multicast
- Unique identification of:
 - Data sources: GUID
 - Messages: sequence number
- Support for arbitration over redundant data channels or sources:
 - I.e. multiple network connections
 - I.e. Original writer and persistence service
 - A “Virtual GUID” allows correlation of data streams even if the originate on different sources

RTPS Features (2/2)



- Support for message filtering at the source:
 - Special GAP message can efficiently communicate absence of data for sequence number ranges
 - Used for content-based filtering at the source
 - Used for time-based filtering at the source
 - Used for history-depth filtering at the source
- Efficient message encoding:
 - Support for message aggregation
 - Many data-updates can be bundled in a single RTPS message
 - Timestamps can be shared across multiple messages
- Message fragmentation & re-assembly
 - Unlimited message sizes can be sent over UDP independently of transport-level limits

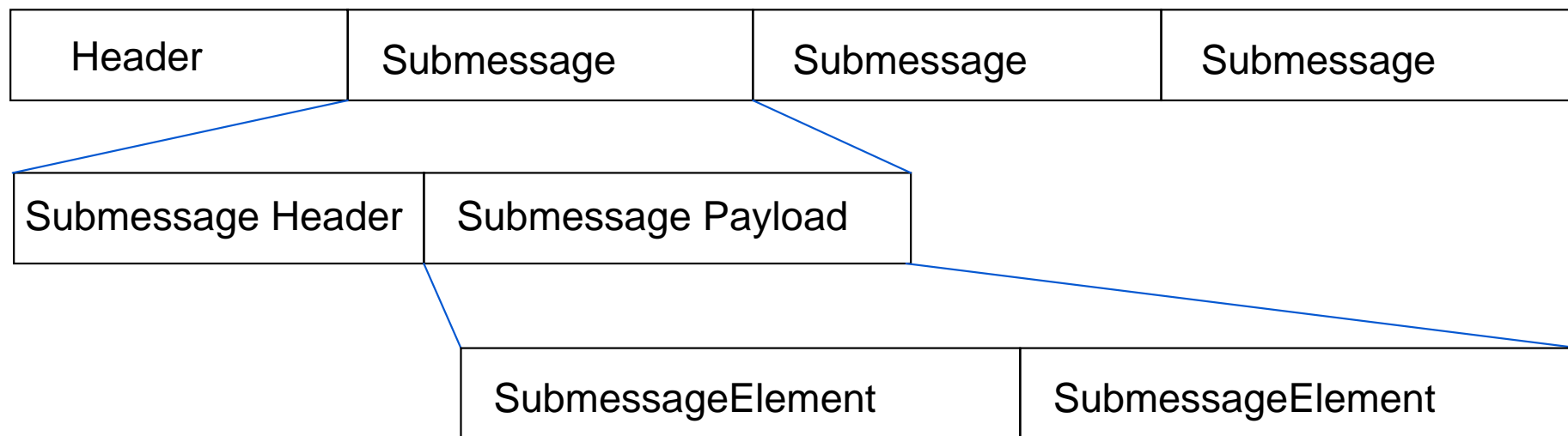
Message Modules



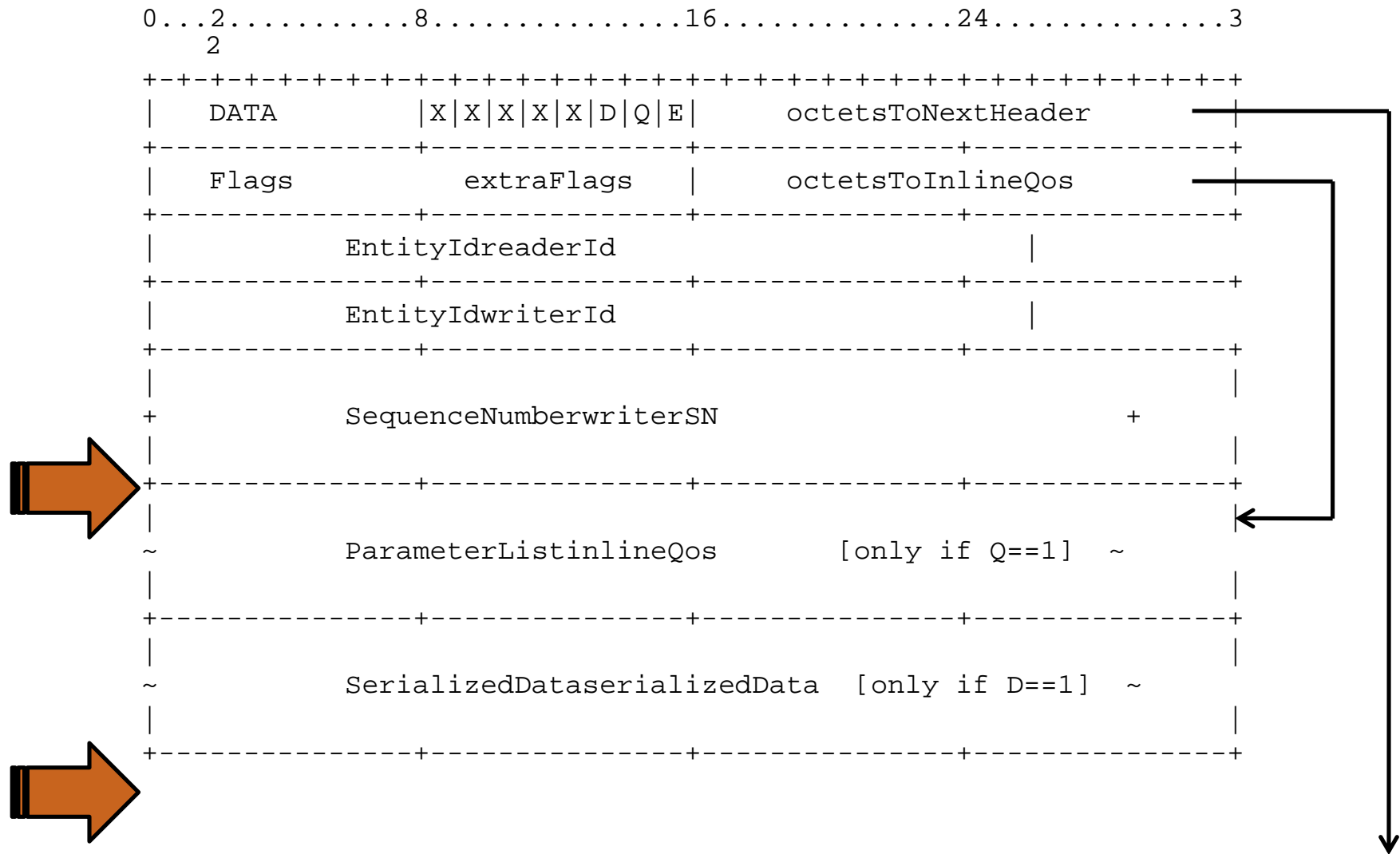
- Multiple extension mechanisms provided:
 - Protocol Extensions
 - Define new submessages (can be vendor specific)
 - Add additional information to an existing submessage
 - QoS Extensions
 - Support additional QoS
- Extension mechanisms are designed to maintain backwards compatibility
 - Well defined rules exist for what can be changed within a major version of the protocol
 - Incompatible changes require a new major version number

Protocol Extensions

- Modular message composition:

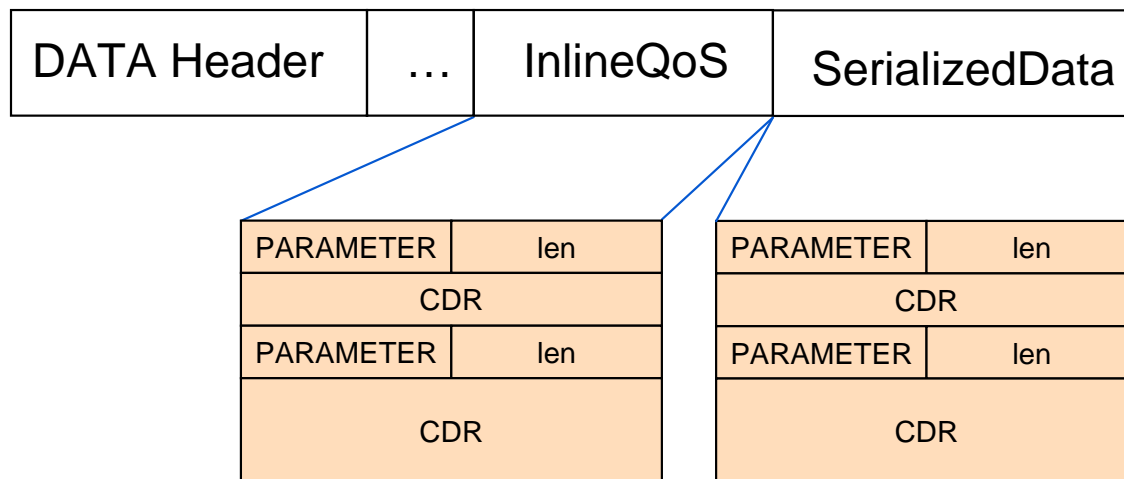
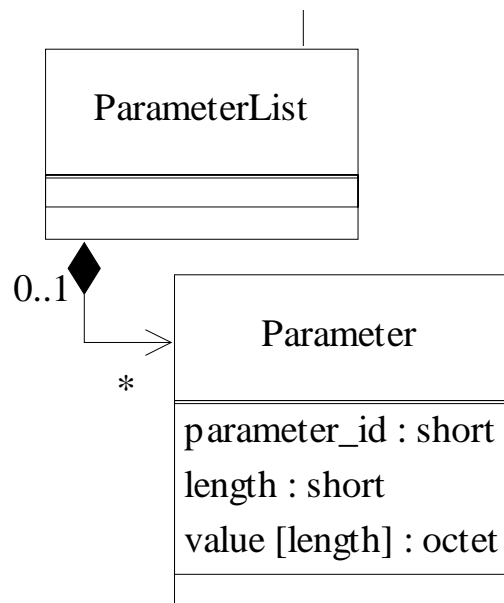


Submessage extensions



QoS Extensions

- Uses the ParameterList SubmessageElement
 - Variable length list of parameters identified as key-value pairs

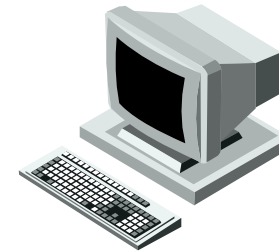
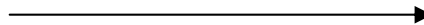


Example: Skip parameter if not recognized



DDS v1.1 (with new QoS)

Discovery traffic



DDS v1.0
(without the new QoS)

PID_RELIABILITY	4
CDR encoded reliability QoS value	
PID_NEW_QoS	length
CDR encoded new QoS value	
PID_USERDATA	8
CDR encoded DDS User Data QoS	
...	
PID_SENTINEL	0

Skip
"length" bytes

PID_RELIABILITY	4
CDR encoded reliability QoS value	
PID_NEW_QoS	length
CDR encoded new QoS value	
PID_USERDATA	8
CDR encoded DDS User Data QoS	
...	
PID_SENTINEL	0

Agenda



- Introduction to RTPS
- Benefits of RTPS interoperability
- Gaps
- Conclusions



What does it all mean?



- An application that understands the protocol can communicate with any other application that also implements the protocol
 - Including applications that don't use a DDS API
- This is independent of:
 - Vendor: RTI, PrismTech, Twin Oaks, homegrown middleware, ...
 - Platform/Technology: Java, C++, .NET
 - OS: Linux, Windows, VxWorks, ...
 - Architecture: Intel, SPARC, PowerPC, ARM, ...

Kinds of tools / services

- Data-Oriented Services
 - Persistence
 - Recording, Replay
 - Graphic front-end (Excel, Scope) ← DEMO
 - Injection (read CSV, read database, bare-bones publishers)
 - Transformation (C8, router)

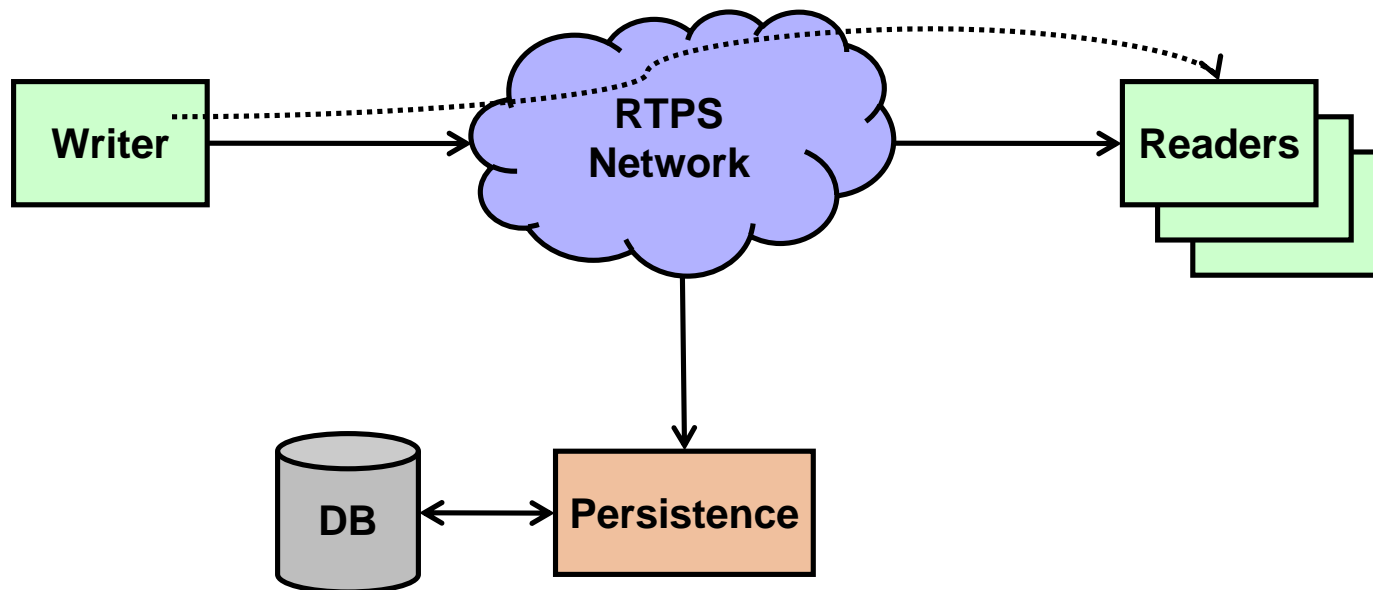
- System Visualization
 - DDS Entities (RTI Analyzer, PT's Splice Tuner)
 - Sessions/Conversations
 - Protocol-level information (Wireshark) ← DEMO

- Integration/Exposure/Access
 - Databases (Real-Time Connect) ← DEMO
 - Files (RTI FS API)
 - Web APIs (Web-Enabled DDS)
 - Other adaptors

Data-oriented Service: Persistence



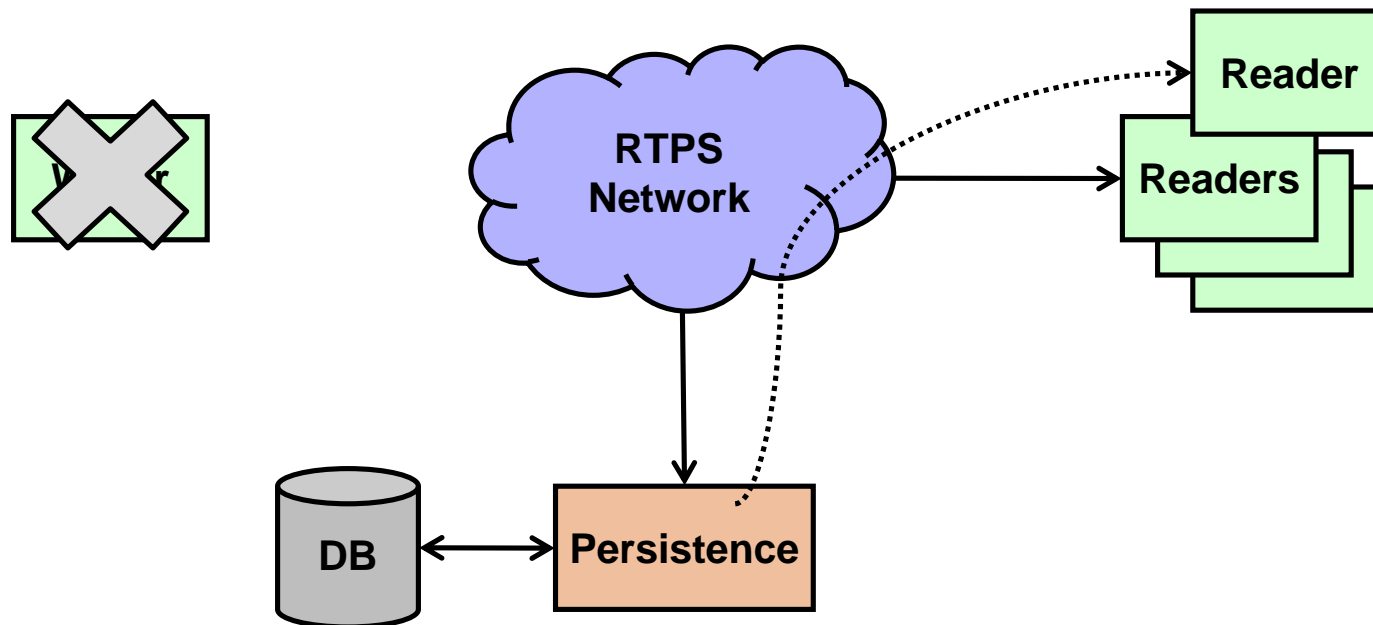
- Guarantee data is available for late joiners even if the original writer disappear from the network
- Ensure data is persisted to a non-volatile memory
- The only service defined by the standard



Data-oriented Service: Persistence



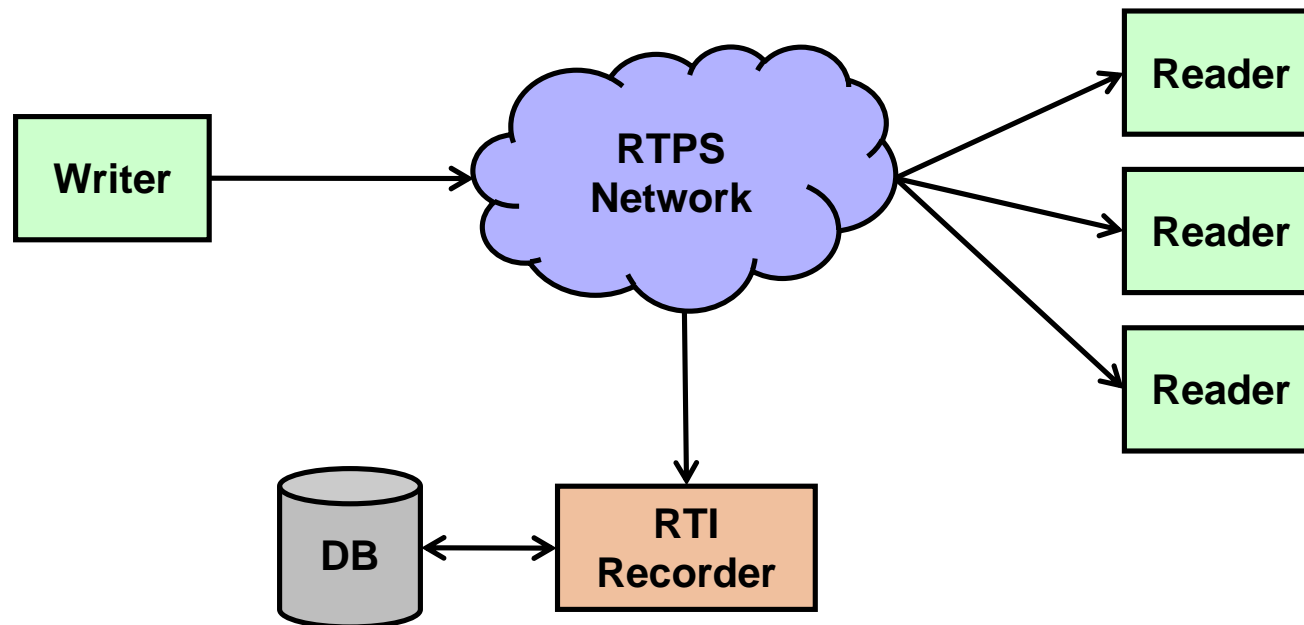
- Guarantee data is available for late joiners even if the original writer disappear from the network
- Ensure data is persisted to a non-volatile memory
- The only service defined by the standard



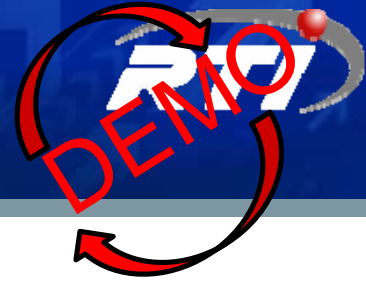
Data-oriented Service: Record / Replay



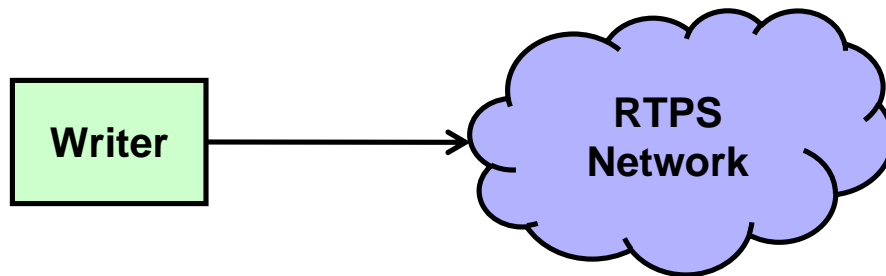
- Record data as it gets published on the network
- Allow off-line analysis of data
- Replay data in the same order as it was originally published



Data-oriented Service: Graphic front-end



- Display published data in real-time
- Easily create graphic front-end to publish data
- Example of applications:
 - Spreadsheet (i.e. Microsoft Excel, OpenOffice)
 - RTI Scope (software scope analyzer)



RTI Scope

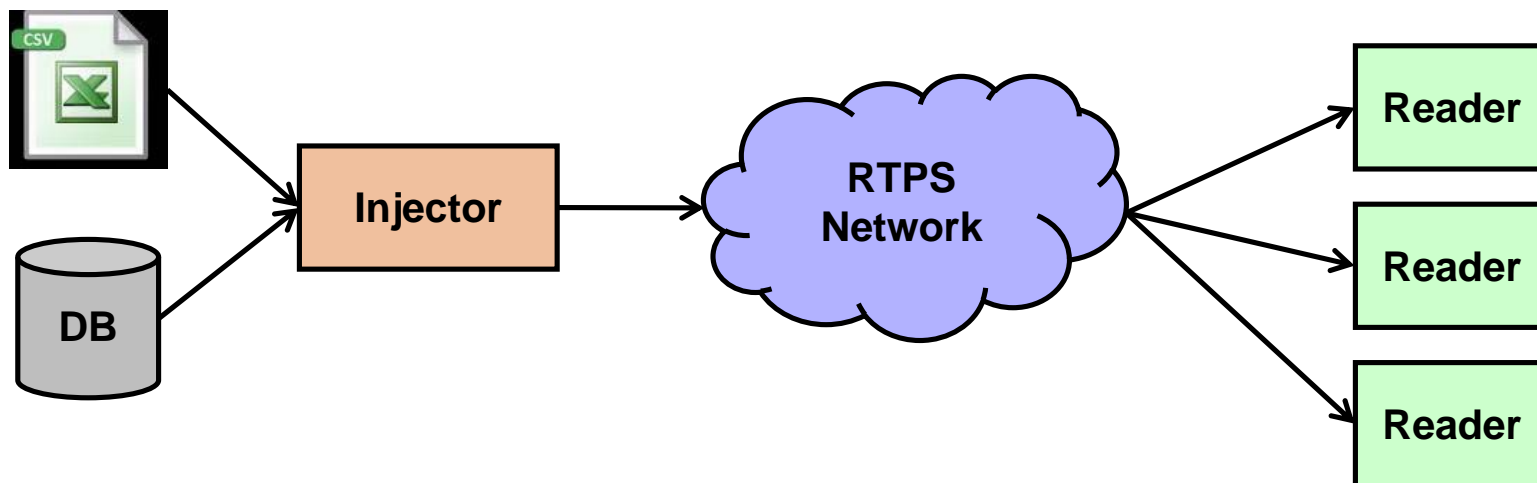


Microsoft Excel

Data-oriented Service: Injection



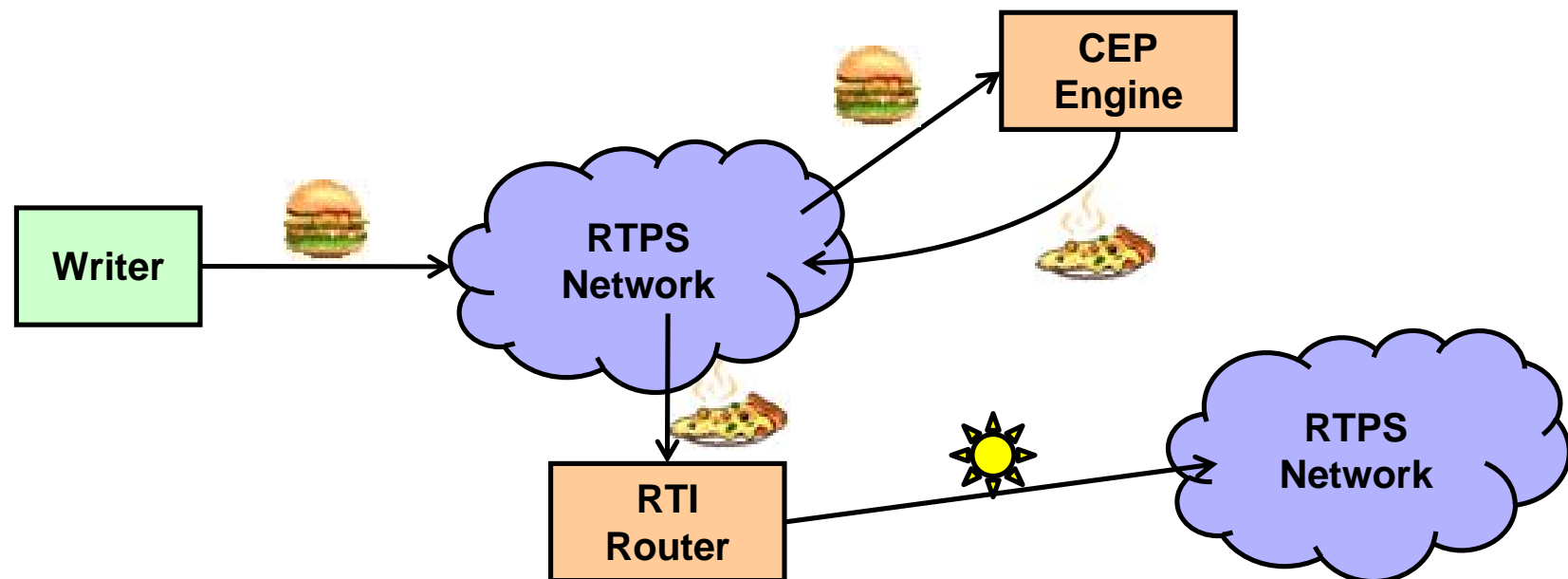
- Publish data read from a CSV file
- Publish data obtained from a database table
- Generic application that publish data from a particular source (data stream



Data-oriented Service: Transformation



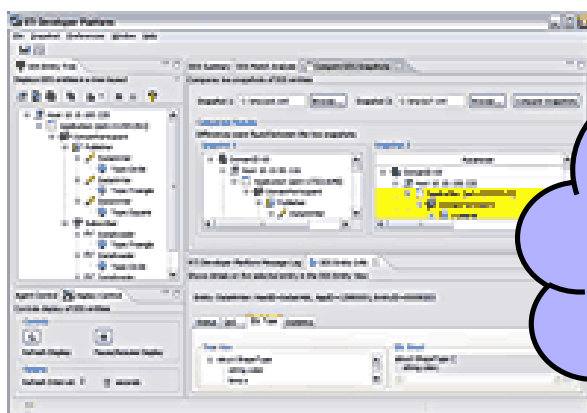
- Subscribes to some data, transform it, then publish it back
- Examples
 - RTI DDS Router
 - Complex Event Processing



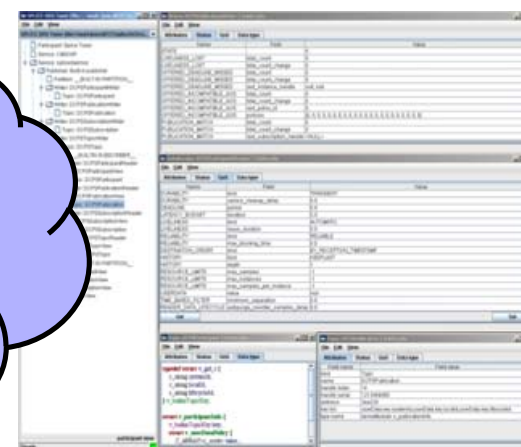
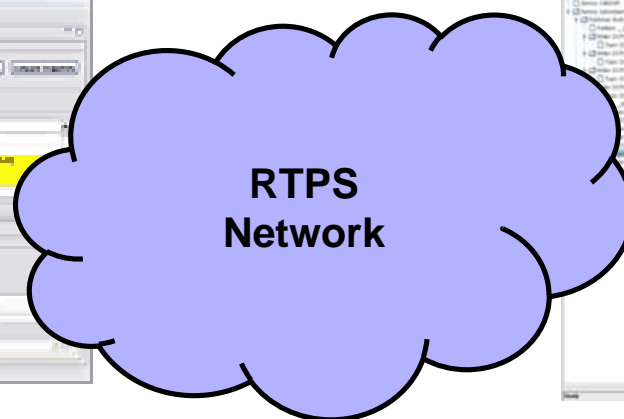
System Visualization: DDS Entities



- Shows all the DDS entities in a domain
- Inspects individual entity
 - Show exposed QoS
 - Show what is publishing



RTI Analyzer

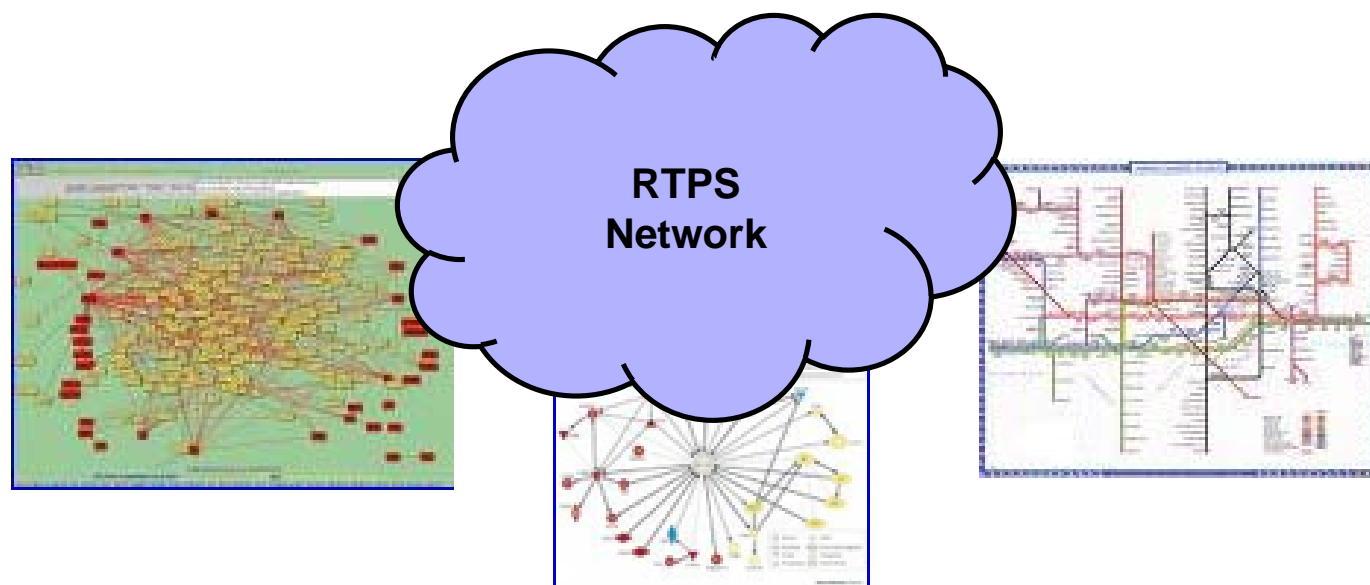


PrismTech Splice Tuner

System Visualization: Session / Conversation



- Visually map conversations between pub and subs
- Shows who's subscribing to a particular topic
- Dynamically groups entities by their role in the system
- Allow dynamic introspection of data



System Visualization: Protocol-level information



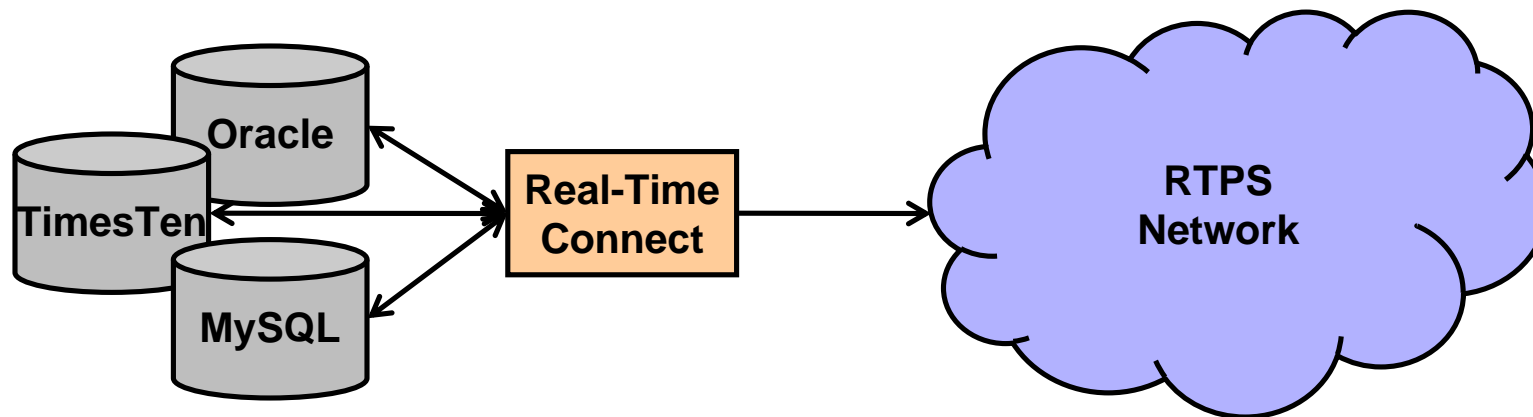
- Low-level packet dissector:
 - Shows individual submessages
 - Shows QoS propagated
 - Separate meta-traffic from user's data
 - Visually shows bandwidth used by discovery process
 - Debug RTPS implementers to comply with the standard
- One tool to rule them all:



Integration / Exposure / Access: Databases



- Automatically map data in the global data space in a database table
 - When a new sample is published, a row is updated on the table
 - When a row is updated, a sample is published on the net
- I.e. : RTI Real-Time Connect



Integration / Exposure / Access: File system



- Maps DDS concepts on familiar file-system concepts
- Tight interaction with the OS
- ANY programming language
- Can access DDS entities through command-line utilities, or existing tools
- Still confused? Pay attention to the demo!

Integration / Exposure / Access: Web-API



- Allow access to DDS data using WS-* technologies:
 - Web services using SOAP
 - RESTFul API
- Using existing standard:
 - WS-Eventing
 - WS-Notification
 - Web-Enabled DDS (?)
- Demonstrated this morning

Integration / Exposure / Access: Other adaptors



- Allow interoperate with other middleware...
- ... or with other data bus
- I.e.:
 - RSS feeds
 - PODCasts
 - Twitter
 - ...

Agenda



- Introduction to RTPS
- Benefits of RTPS interoperability
- **Gaps**
- Conclusions



- Features required in order to enable vendor interoperability of tools/services:
 - Type-Description
 - Standard transformations to other formats: XML, SQL, ...
- No statistics information available through built-in topics.
- Most of these gaps will be solved by Extensible Topics spec. (expected in 2009)

Agenda



- Introduction to RTPS
- Benefits of RTPS interoperability
- Gaps
- Conclusions



Conclusions



- RTPS opens the door to a whole ecosystem of interoperable tools & services
- These can be developed by any vendors and interoperate with any compliant implementation
- The main missing gaps to this vision are being closed with the Extensible Topics spec
- Why is this great?
 - I can mix/match tools from multiple vendors and have a much bigger market for my technology
 - Bigger Audience -> Bigger Market -> Lower cost

- Recommendations for the DDS SIG and the DDS Revision Task Forces:
 - The problem that extensible topic is trying to solve is critical for the tools ecosystem.
 - There can be more services that can be standardized (like the persistence service)
 - Extend RTPS to define how statistics are exposed