Overview

• RAPID vs. DDS
  • What is RAPID
  • What is DDS

• DDS Components
  • DDS Comm Model
  • DDS Infrastructure

• RAPID Design
  • RAPID Messages
  • RAPID Services

• Impact on RoverSw
  • DDS and RoverSw
  • RAPID and RoverSw
What is RAPID

Rover API Delegate
- Interoperability layer for ETDD robots
- Originally implemented as robot-bridge
- Open-source

Inter-center standard
- Supported by (Tri-)Athlete, K10, LER
- Interest by ESA
- Targeted as NASA standard

Addressed functionality
- Robot telemetry
- Robot geometry
- High-level commanding and sequencing
# What is DDS

**Distributed System Middleware**
- Publish/subscriber architecture
- Extended by relational data model
- Extensive QoS
- Orthogonal to CORBA
- Competitor of CORBA Notification Service

**Open Standard (OMG hosted)**
- Multiple vendors (RTI, Prismtech, Open-source)
- Push for interoperability (young standard)

**Target Domains**
- Distributed, embedded, real-time systems
- High throughput, low jitter
- Complex (non-uniform) network characteristics
**DDS Communications Model**

**Multi-cast messaging core**
- Data bus
- One-to-many communication
- Unreliable core protocol

**Shared information space concept**
- Data-type instances are entities of shared information
- DDS manages, how this information is shared within a domain
  - Distribution
  - Updates
  - Ownership
  - …
**DDS Communication Entities**

**Domain**
- Defines a shared information space
- Participant manages membership (protocol, ports, etc)

**Partition**
- Defines communication sub-groups within domain
- Publisher/Subscriber manages membership (among other resources such as threads/msg queues etc)

**Topic**
- Defines name and type of a set of data instance(s)
  - Unkeyed data: topic names a single datatype instance
  - Keyed data: topic manages set of instances, differentiated by key
- Communicates samples of instance(s) of one specified data-type
- DataWriter/-Reader reads/writes & manages samples of a topic

Fig © RTI
DDS Quality of Service (QoS)

QoS at every level
- Ports, Multi-cast groups, discovery, etc
- Sometimes more than you’d want to care about

Topic level QoS
- Reliability
  - Best effort
  - Reliable (TCP-like resending)
- Durability
  - Transient
  - While writer instance is available
- History
  - How many samples to store per instance
  - On writer-side required for reliability
  - Can be used on reader-side as histogram queue
- Filtering – reader triggered
  - Message-timeout filter for rate-filtering
  - Partitions for per-robot filtering
- Flow Control – writer triggered
  - Required to protect small links
  - Token-bucket filter
 DDS Infrastructure (RTI)

DDS Core
- (CORBA) IDL defined data types
- Communication of data samples between publishers/subscribers of a domain

Routing Service
- Manage message replication between DDS domains
- Used to manage/restrict traffic on thin links (satellite etc)

RT-Connect & Recorder
- Linking of DDS topics/instances to SQL database tables and entries
- Can be used for logging
- Possible interface to GDS services

Analyzer and Monitor
- Run-time analysis: Connectivity, QoS mismatches, bandwidth etc
- Very powerful, very necessary
- Too little static analysis tools so far

Fig © RTI
RAPID Design

<table>
<thead>
<tr>
<th>Config messages</th>
<th>State messages</th>
<th>Sample messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Subsystem configuration (typically static)</td>
<td>• Subsystem state changes</td>
<td>• Fixed (high) rate telemetry</td>
</tr>
<tr>
<td>• Potentially verbose specification</td>
<td>• Concise</td>
<td>• Mostly continuously changing parameters</td>
</tr>
<tr>
<td>• Reliable &amp; durable</td>
<td>• Reliable &amp; durable</td>
<td>• Best effort, transient</td>
</tr>
<tr>
<td>➢ Send once, subscribe and analyze on client startup</td>
<td>➢ Send on change, might be bursty</td>
<td>➢ Classic telemetry</td>
</tr>
</tbody>
</table>
RAPID Services

Telemetry
- Robot pose
- Robot joints
- Images & Point-clouds

FrameStore
- Classic tree of coordinate frames
- Tree-walking for coordinate transformations
- Local instance, updated from robot telemetry
- Provides location awareness between robots

Sequencer
- Synchronous command queue
- Designed for time-delayed teleop

Access Control
- Cooperative management of teleop access
RAPID Message Examples

**PositionConfig, PositionSample**
- Config: Coordinate frame of robot pose estimate
- Sample: Pose estimate

**JointConfig, -Sample**
- Config: Joint frames
- Sample: Joint positions

**PointCloudConfig, -Sample**
- Config: Frame, data-format details
- Sample: Point cloud

**ImageSensorState, -Sample**
- State: Imager parameters
- Sample: Image

**FrameStoreConfig**
- Coordinate frame tree with initial values

**AccessControlState, QueueState**
- Current controller and control requests
- Queued and executed commands
DDS and RoverSw

Replacement for EC
- Notify Pipe Svc
- LogSvc and LogPlayer
- Telemetry-types mostly stay unchanged
  (some changes to header information)

Configuration and state related parts of GRI CORBA interfaces
- Readonly attributes become Config message
- Periodic SSubsystemState become reliable/durable

Not a good replacement for commanding
- No request/reply pattern
- No object oriented method-call
- Mix of CORBA & DDS is part of the COBA Component Model (CCM)
- It’s two big libraries to carry around
RAPID and RoverSw

KN rovers support it
  • knRapidSvc RAPID bridge
  • Resending GRI telemetry as RAPID telemetry
  • Redirecting RAPID commands to GRI interfaces
  • RAPID sequencing and access control protocols honored

Core RAPID services are part of IRG core technologies
  • FrameStore (C++) implementation part of VisionWorkbench/knSvcs
  • FrameStore (Java) implementation part of Verve/Eclipse
  • Updates of frames directly from GRI telemetry

Might replace telemetry messages where redundant
  • SJoints vs JointSample
  • SPose vs PositionSample