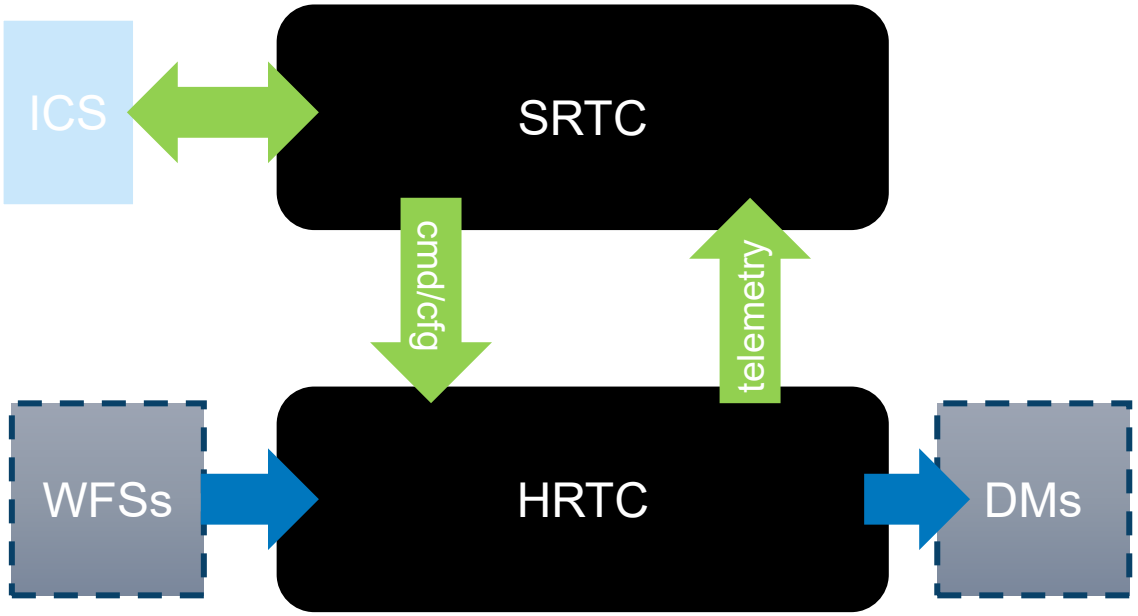
# RTC Toolkit
# Overview and Status

*Bogdan Jeram (bjeram@eso.org)*
*on behalf of*
*RTC toolkit development team*

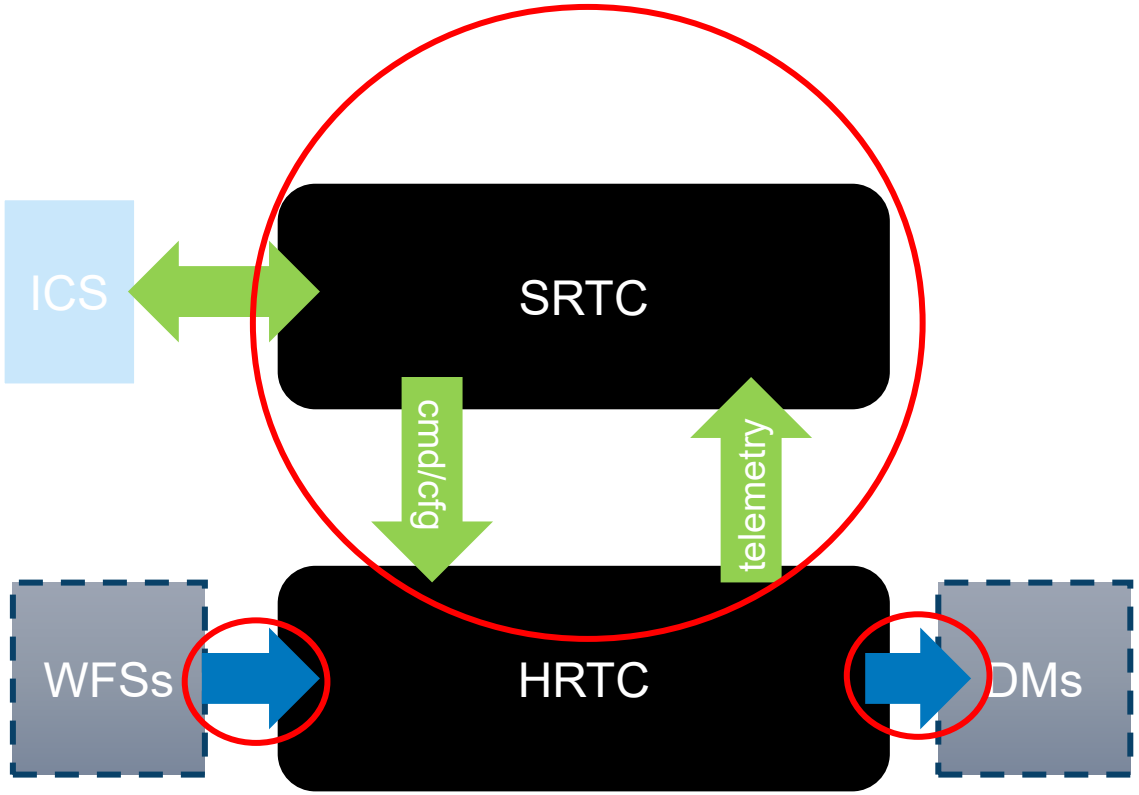
# AO RTC Standard Architecture

- HRTC:
  - AO correction
  - performance-critical:  several 100 Hz / kHz
- SRTC:
  - Receiving telemetry data
  - Optimization
  - (re)configuring HRTC by sending cfg to
  - Commanding and Monitoring HRTC
  - Interacts with the rest of the instrument (ICS)



# RTC Toolkit

- SW building blocks to build SRTC and interface with HRTC and with the rest of the instrument:
  - Libraries
  - Executables
  - Tools
  - SW interfaces
- Guidelines: SW, HW, network, and set-up/tunning
- Simulators: WFS, DM-echo and CCS deterministic



# Highlights

- Performance oriented
- Scalable:
  - # of servers in SRTC
  - amount of data transfer from/to HRTC
  - Calculation needs
- Customizable/extendable
- Flexible:
  - CPU computation: C++ / Python
  - GPU computation using CUDA
- Platform: Linux
- Programming language: C++ / Python (limited)

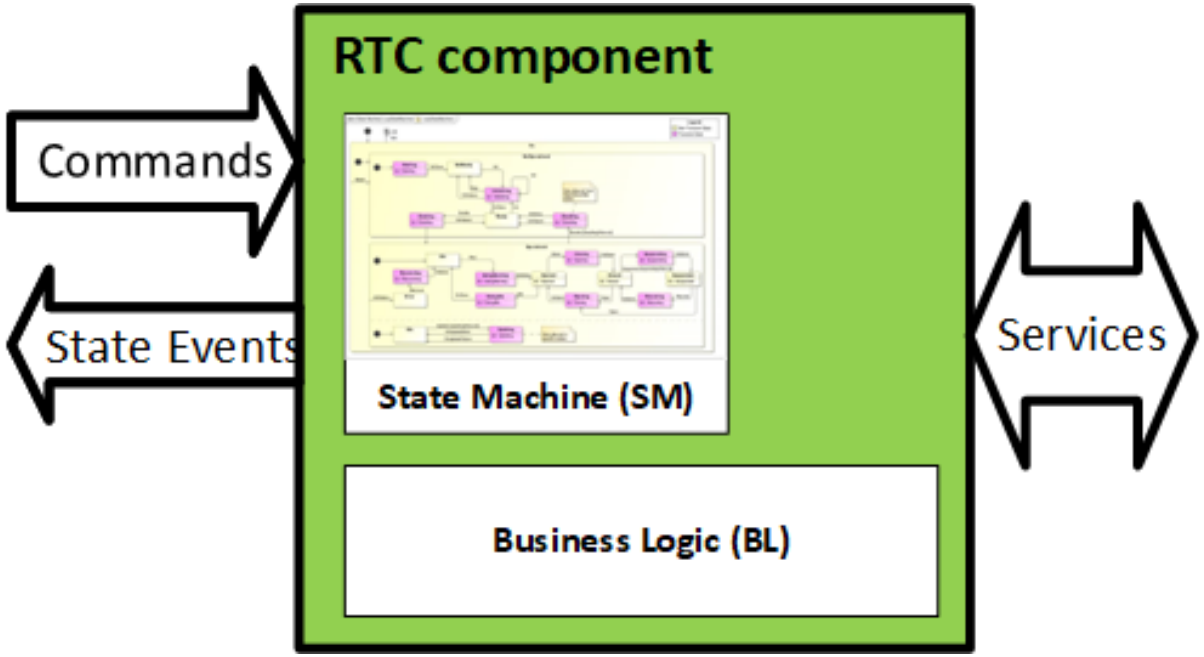
# Main Functionality

- Component framework – processes/service
- Configuration: persistent and run-time
- Telemetry propagation
- Data Task infrastructure for number-crunching (processing) data
- Recording: raw and preprocessed telemetry
- Visualization
- Simulators

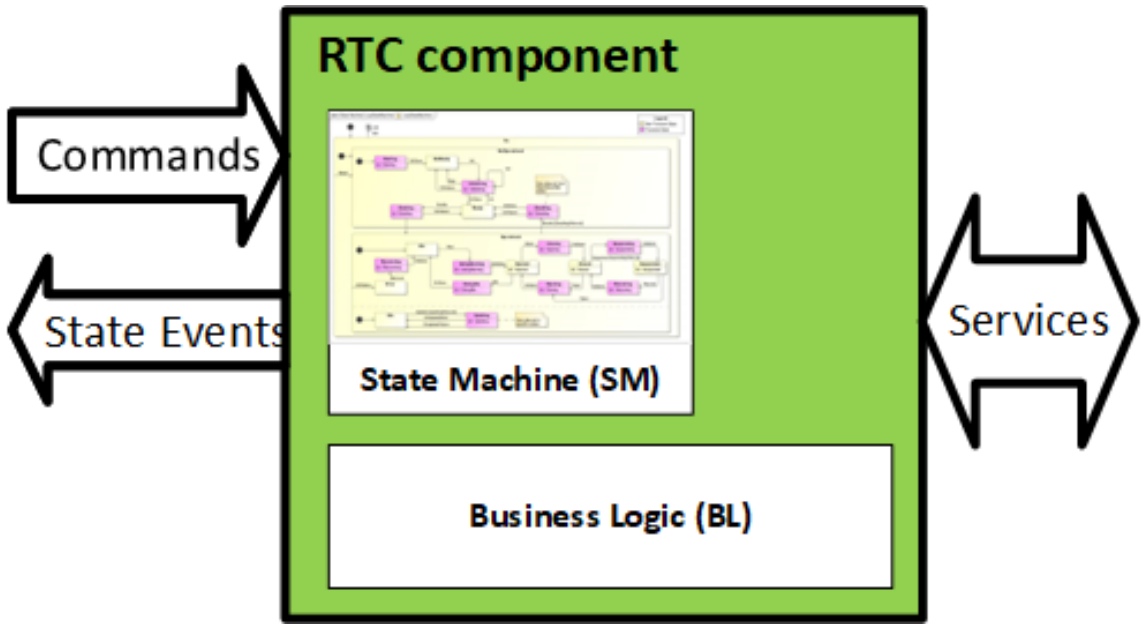
# Component Framework

# Commands & Events

- Commands: Request/reply based on ELT CII MAL using 0mq + Protobuf
- State Machine (SM) engine based on RAD
- Business Logic (BL) – actual implementation
- Standardized Command interfaces. E.g:
  - *Init->Enable->Run*
  - *Init->Enable->Run->CloseLoop*
- Customizable Interface + SM
- State events: CII MAL Pub/Sub using DDS



# Services




- Service Discovery
- Logging
- Configuration: RTR/PSR
- Event Channel: application-specific events
- Online Database (OLDB): monitoring info
- Alerts
- Metrics/Performance counters

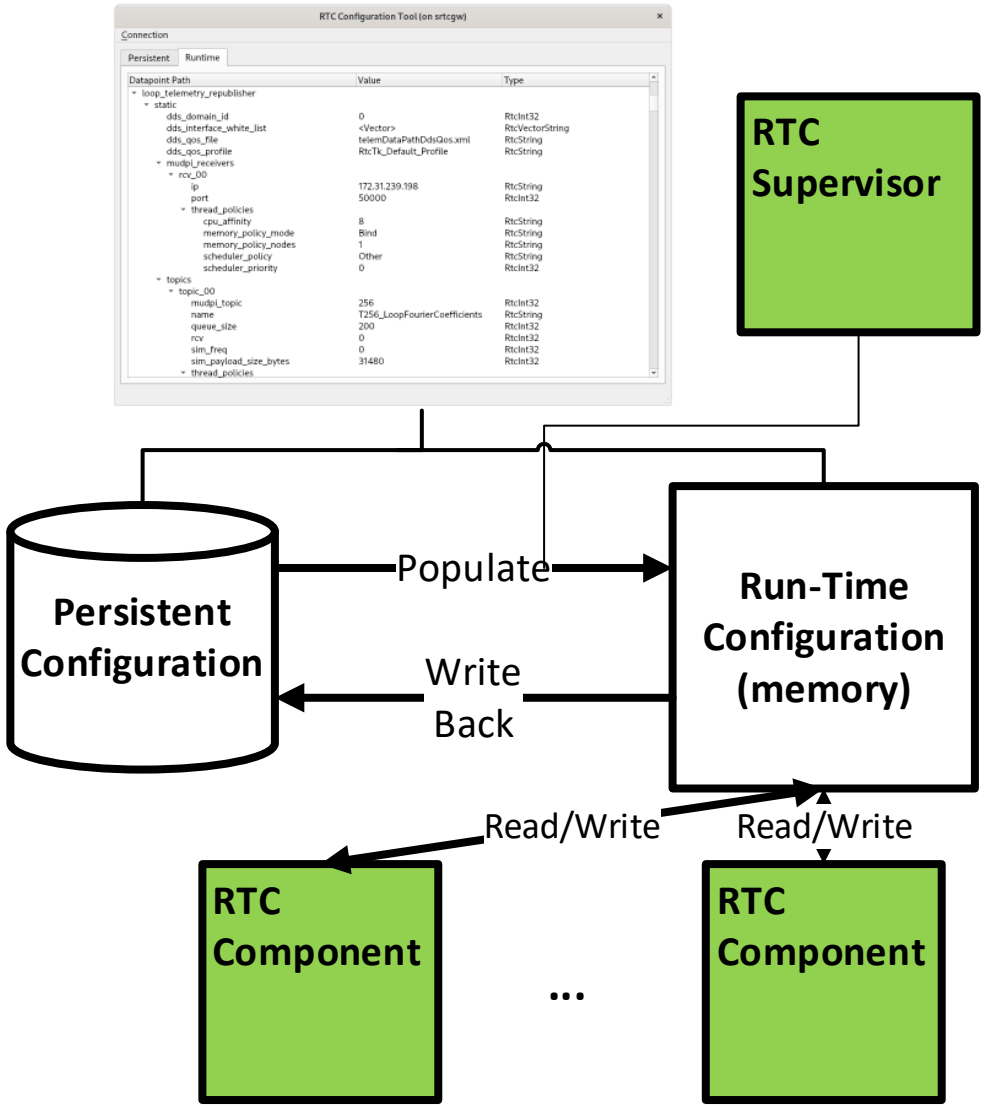
# Component customization

- Configuration:
  - just configuration needed.
  - E.g. Telemetry Republisher
- Extension (optional):
  - provide implementation class.
  - E.g. Telemetry Subscriber
- Composition:
  - put provided parts together.
  - E.g. Data Task, Telemetry Recorder

# Configuration

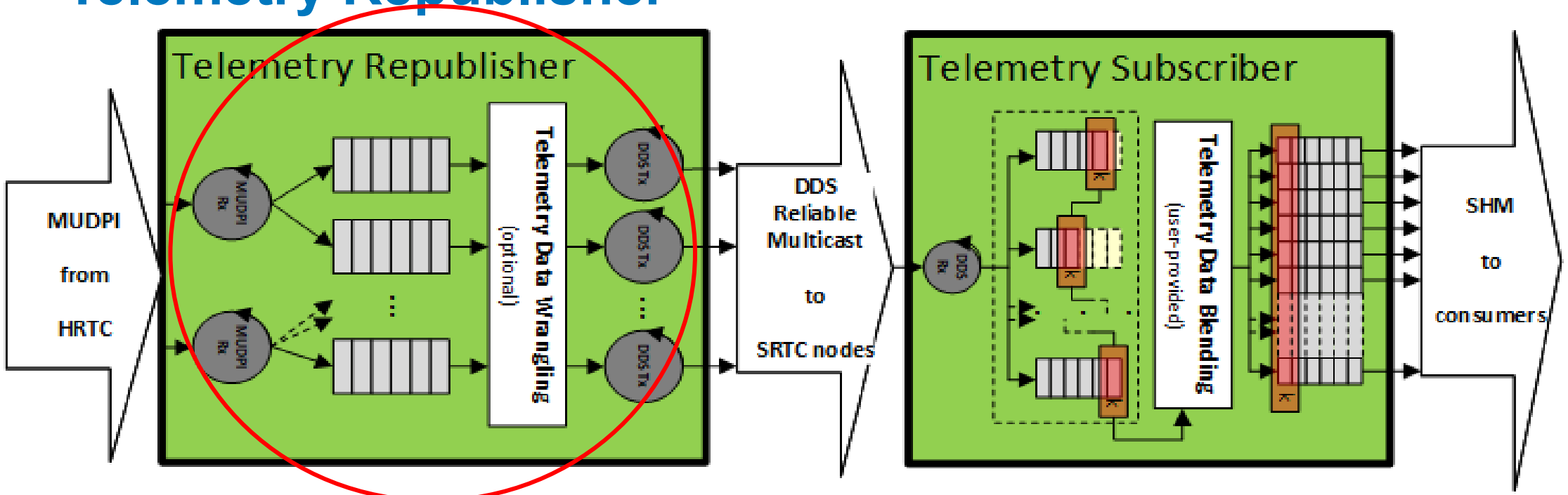
# Configuration architecture

- Persistent Configuration Repository (PSR):
  - on disk or central
  - support for deployment-sets and modes
- Runtime Configuration Repository (RTR):
  - in memory => fast access
  - Distributed
  - populated from PSR
  - write back
- Supported types:
  - types: bool/(u)int(32/64)/float/double/string/…
  - scalars/vectors/matrices



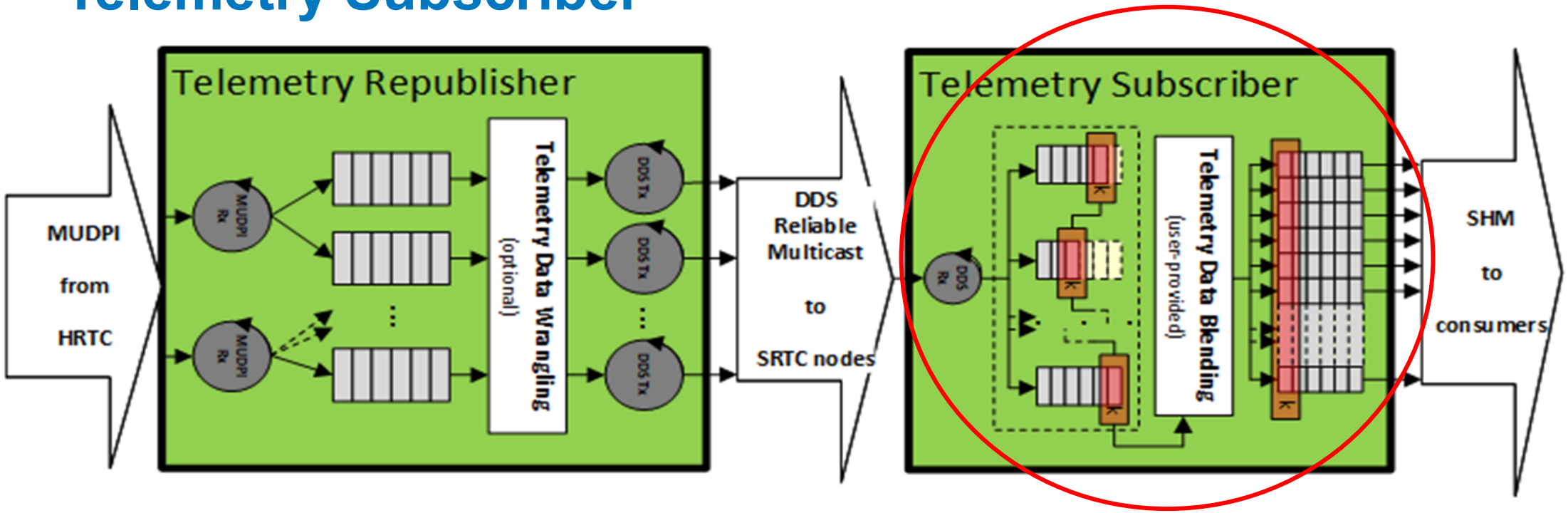
# Telemetry

# Telemetry Republisher




- MUDPI to DDS (FastDDS)
- Agnostic topic payload => no recompiling needed
- Data Wrangling (endianness, padding, …) optional
- Flexible: 1..* instances, each handling 1..* topics, receiving at 1..* NICs

# Telemetry Subscriber

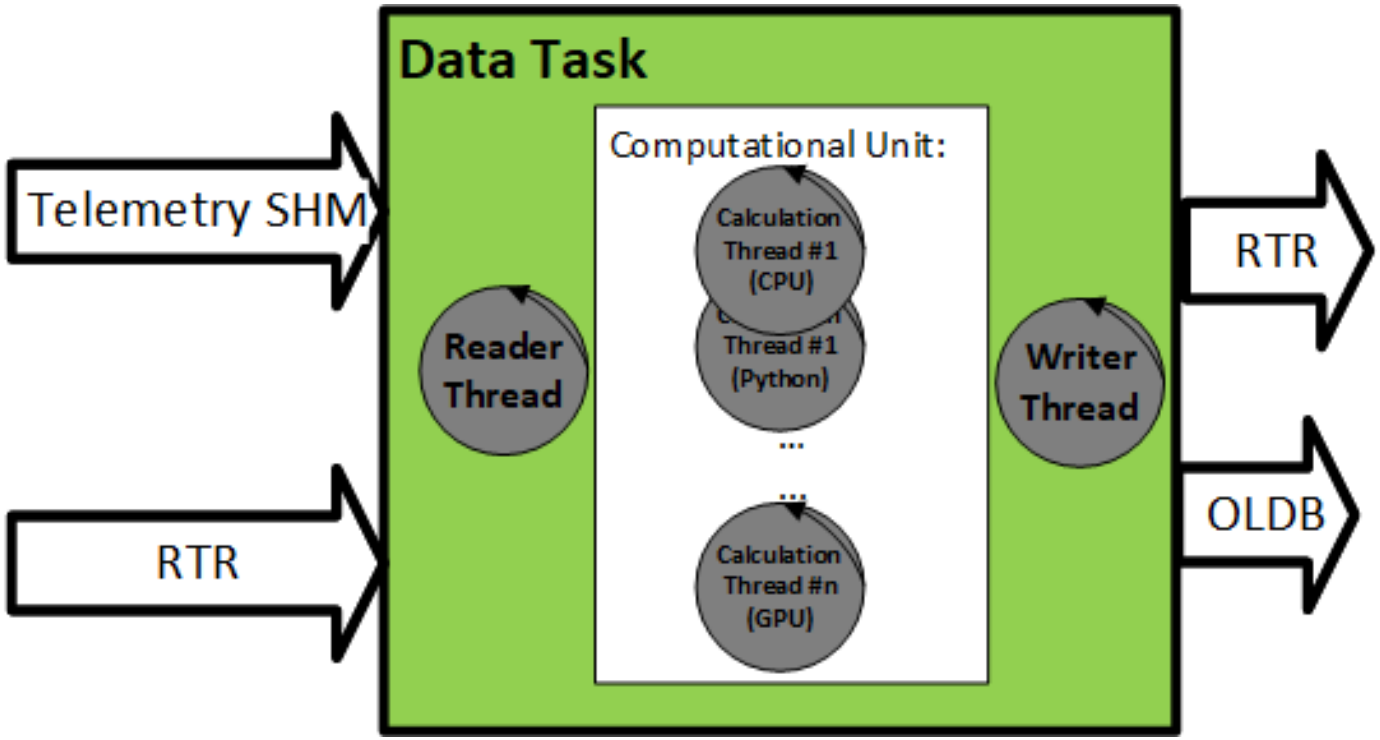



- DDS to SHM ring buffer
- Data Blending (combine several topics into one big super-topic)
- One instance per SHM super-topic and server
- SHM ring buffer consumers: Data Task, Recorders, DDT, …

# Telemetry Data consumers

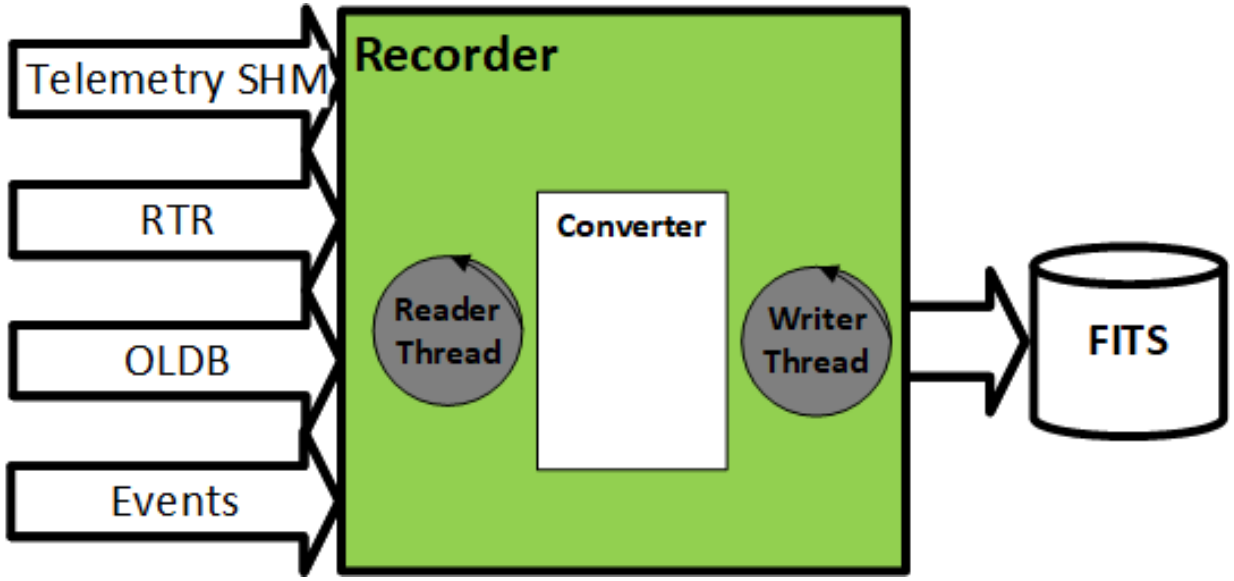
# Data Task

- Input:
  - Telemetry data from SHM
  - Datapoints from Run-Time Repository (RTR)
- **Computing** (Processing):
  - CPU/GPU
  - C++/Python
- Output:
  - Run-Time Repository (RTR)
  - OLDB
  - Events



# Recorders

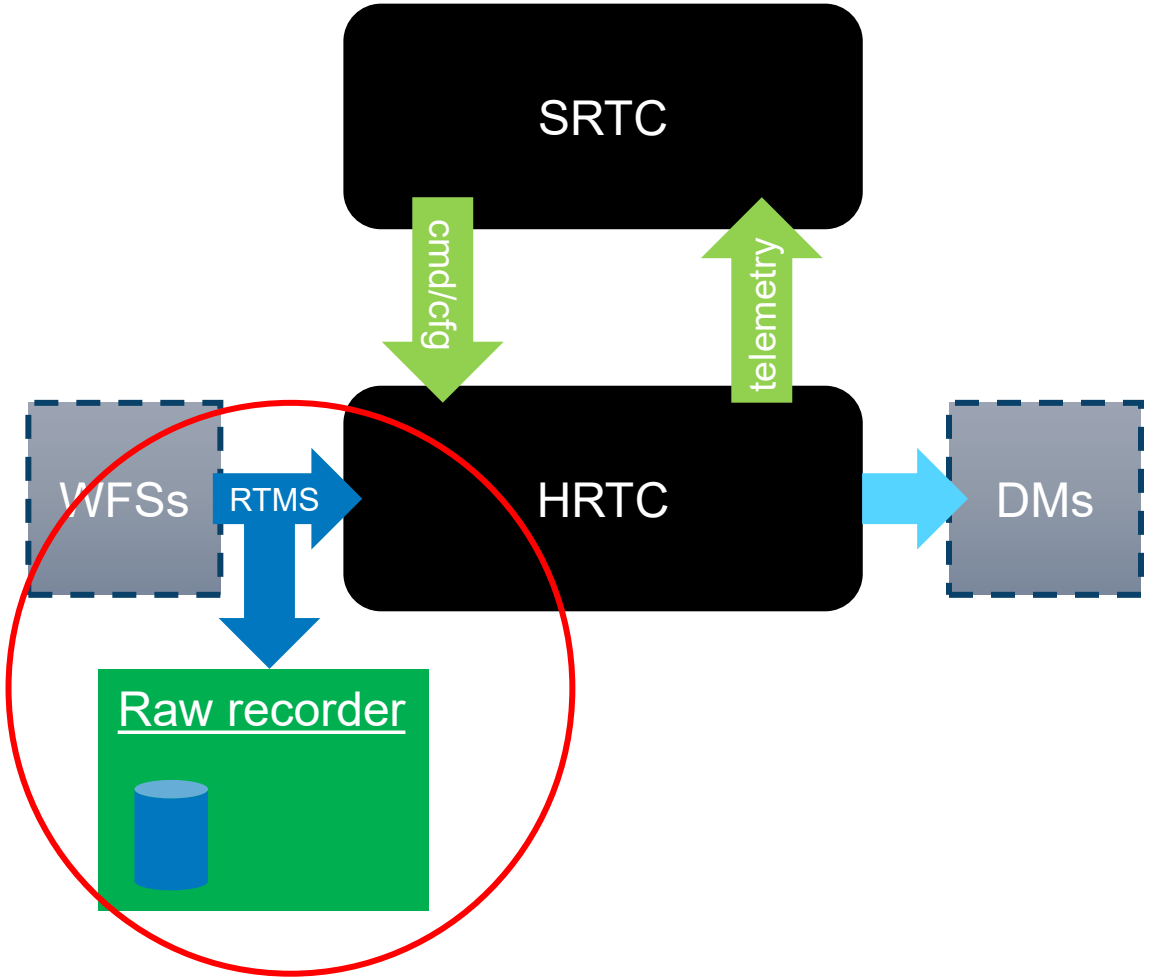
- Telemetry Recorder:
  - Input: Telemetry/Events/OLDB/RTR
  - Output: FITS files
- Meta-data collector:
  - Input: Events/OLDB/RTR
  - Output: FITS files with Exposure Metadata



# Raw Recorder and Simulators

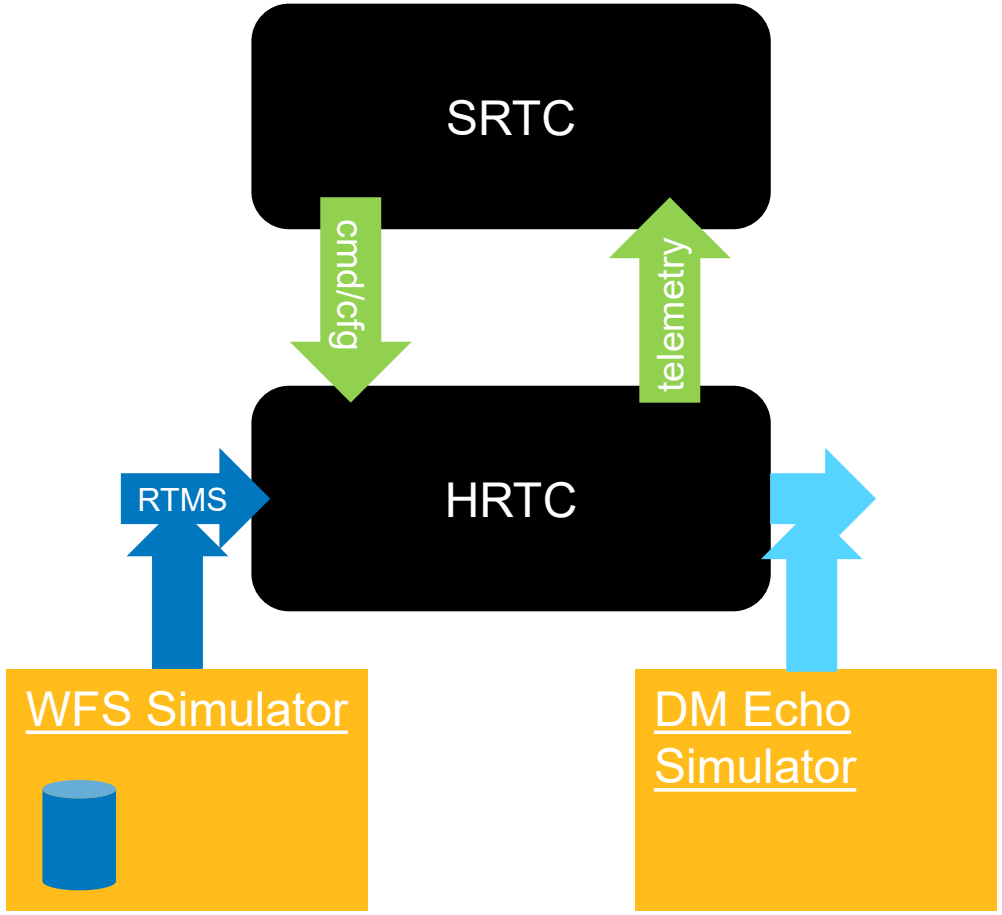
# Raw recorder

- Captures Pixel data in RTMS/MUDPI format
- Extracts and writes to FITS



# Simulators

- WFS: generate or read from FITS file RTMS/MUDPI
- DM Echo: commands' echo from DM



# Deployment and supervision

# Deployment

Deployment Daemon:

- Uses Nomad/Consul to deploy components and services
- Provides command interface to
  - start/stop all components of a selected deployment set
  - start/stop individual components

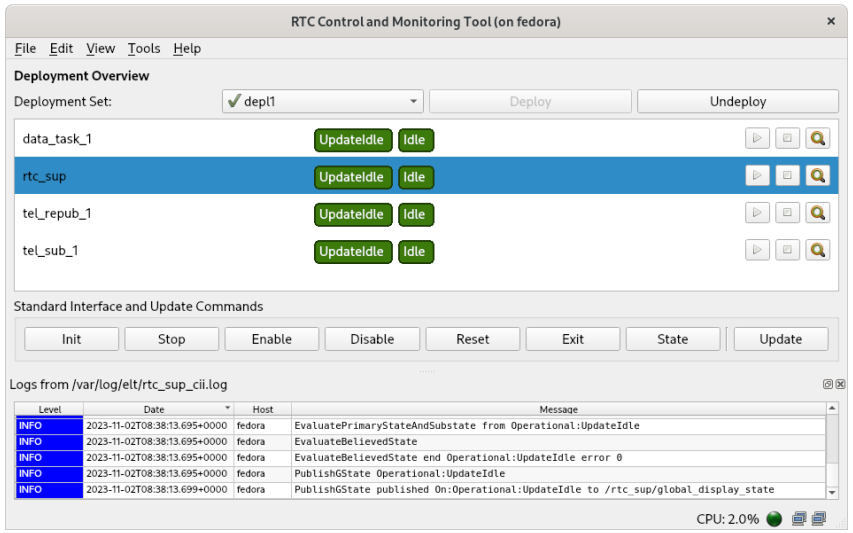
# Supervision

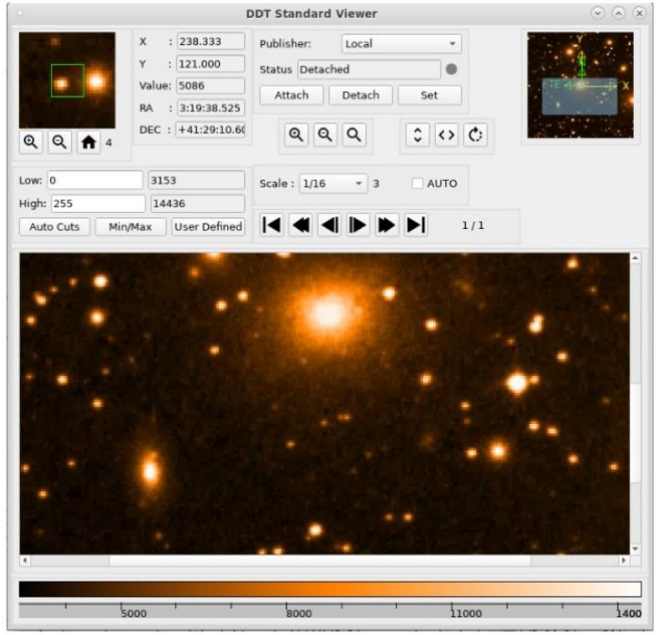
- RTC Supervisor: supervision/orchestration of whole SRTC:
  - populate and update run-time configuration
  - walk components through basic life-cycle
  - monitor individual components
  - estimate overall  status of entire RTC
- HRTC Supervisor:
  - command HRTC functions & loops
  - configure HRTC functions & loops
  - monitor HRTC functions & loops

# Visualization

# GUIs

- Control and Monitoring Tool
  - Un/Deploy the system
  - Steer RTC components and observe states and other monitoring information
- Configuration Tool:
  - Inspect and modify PSR and RTR configuration
- Widgets:
  - to create application-specific GUI
- Data Display Tool (DDT):
  - pixel data
  - telemetry data



# Project Status

# Status

- Yearly releases

- Released:
  - Alpha (beginning 2021)
  - v1.0 (end 2021)
  - v2.0 (mid 2022)
  - v3.0 (mid 2023)

- Working on v4.0

# Users (so far)

- ELT WFRTC
- ELT instruments:
  - 1st generation: MICADO, METIS, HARMONI, MORFEO
  - 2nd generation: ANDES
- VLT instrument: MAVIS

# Thank you!

**RTC toolkit development team**