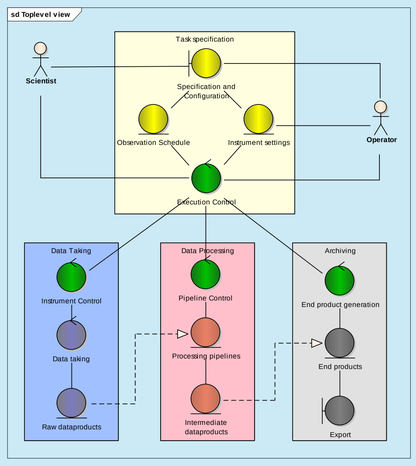
Monitoring & Control Software for the new Westerbork Phased-Array Feed System

Author: Marcel Loose (loose@astron.nl)

APERTIF
APERture Tile In Focus

System Overview

From a software point-of-view



- Task Specification
  - Schedule observations
- Data Taking
  - Write raw UV-data
- Data Processing
  - Calibration & Imaging
- Archiving
  - Data Quality & Ingest
- Monitoring and Control

Monitoring & Control

Layered Design

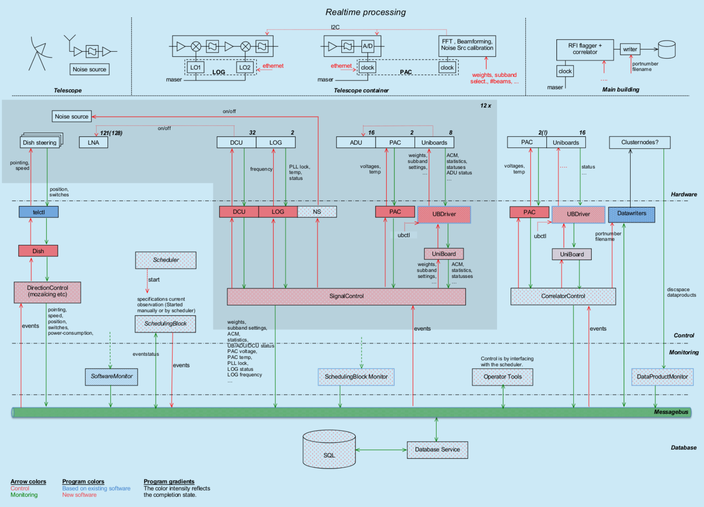
Hardware

Firmware

Driver

Control

Monitoring



Controllers

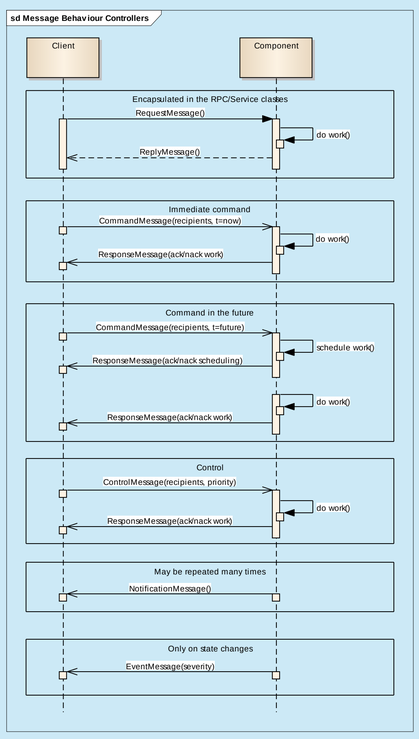
- Handle commands that arrive via the Message Bus
- Serialize access to the Drivers
- High-Level commands
  - Are user- or task-oriented
  - Are typically sent to the *whole system*
  - Examples: “Start Observation”, or “Calibrate PAF”
- Low-Level commands
  - Are subsystem- or driver-oriented
  - Are typically sent to *one sub-system*
  - Are often sub-system specific
  - Examples: “set_vamp”, “set_vcoax”, “get_status”

Drivers

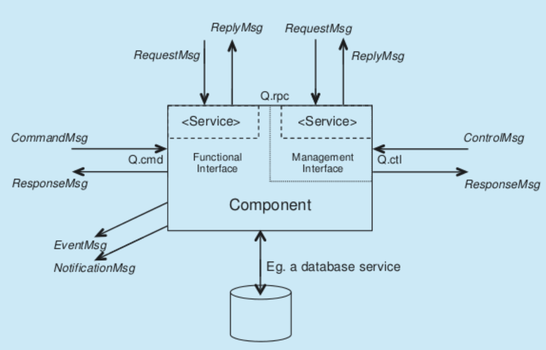
- Drivers interface with (custom-made) hardware
- Interface definition: wire protocol
- Written in Python
  - UniBoard has low-level C++ driver to meet performance requirements

Software Components

Behavior of a component



Interfaces of a component

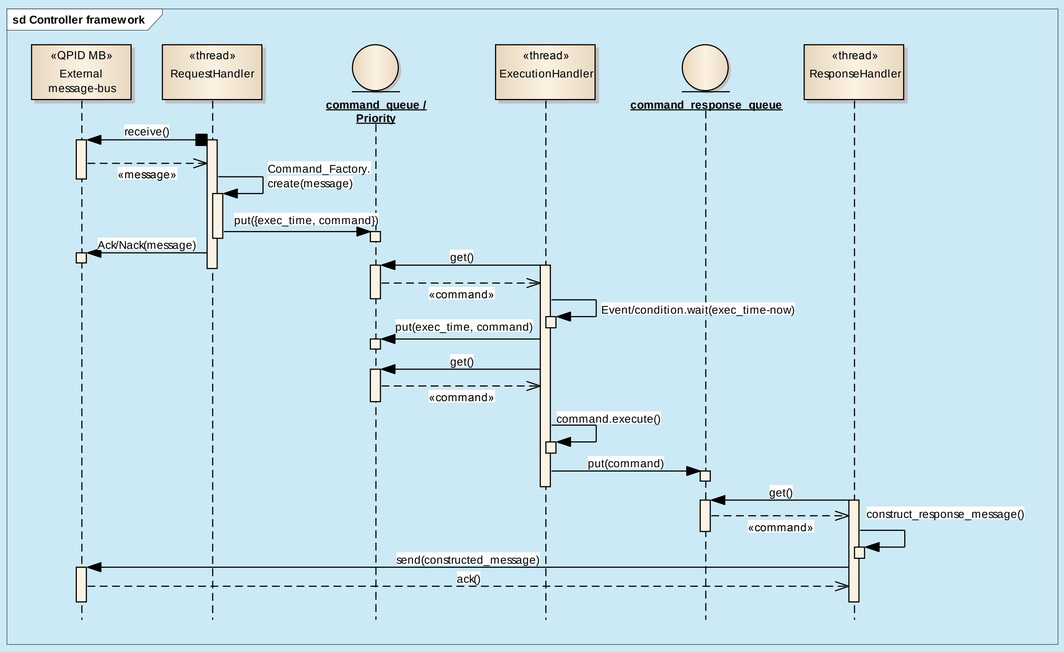


Communication with components

	Synchronous	Asynchronous
Functional	*Communication*: RPC calls (p2p) *Messages*: Request/Reply *Queue name*: .rpc *Purpose*: User interface to RPCs *E.g.* low level commands to the controllers, database access, etc.	*Communication*: Commands (scatter/gather) *Messages*: Command/Response *Queue name*: .cmd *Purpose*: User interface to multiple components with asynchronous answer *E.g.* high level commands
Management	*Communication*: RPC calls (p2p) *Messages*: Request/Reply *Queue name*: .rpc *Purpose*: Ask about health or behaviour of the component *E.g.* queries about counters, performance, etc.	*Communication*: Control (p2p) *Messages*: Control/Response *Queue name*: .ctl *Purpose*: Change behaviour of component *E.g.* shutdown, start/stop/set Notification sending, suspend/resume RPC queue, etc.

Monitoring & Control

Controller Framework

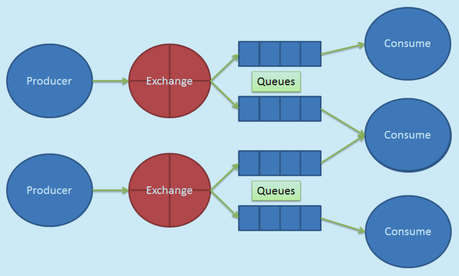


Controller Framework handles three main tasks any controller must do:

- Reception and execution (at the right time) of commands received over the message bus
- Guarantee that commands are not mixed (executed in sequence)
- Publish a configurable set of status information at regular intervals

- RequestHandler
  - Listen for new message on the message bus
  - Turn messages into command objects
  - Put commands in command queue
- ExecutionHandler
  - Wait for new commands in command queue
  - Handle scheduling and execution of commands
  - Put command responses into response queue
- ResponseHandler
  - Listen for responses in response queue
  - Construct response message
  - Send response message to the message bus

Message Queues

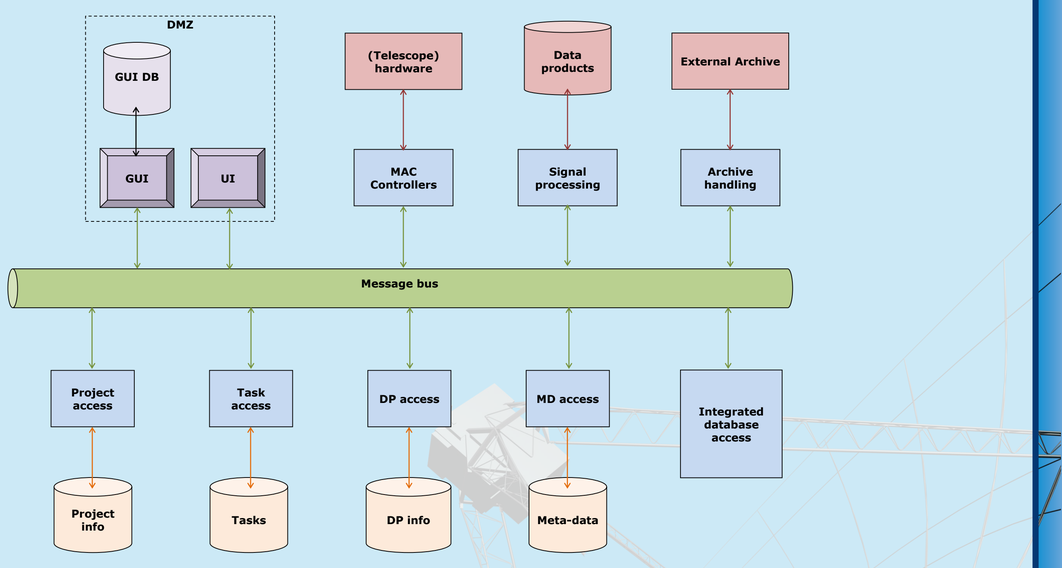


Advantages of Message Queuing

- Reliable communication, guaranteed delivery
- Ease of use: simply post a message to an exchange
- Routing is run-time configurable
- Eases loose coupling of system components

For APERTIF we chose to use AMQP

- Advanced Message Queuing Protocol (AMQP)
  - Based on IEEE standard (ISO/IEC 19464)
- Apache Qpid
  - Messaging built on AMQP
  - C++ and Python bindings
  - Also used by LOFAR
  - Ubuntu PPA (deb http://ppa.launchpad.net/qpid/released/ubuntu trusty main)
- Wrapped Qpid in C++ library & Python module
  - Eases send & receive of messages (“ToBus” and “FromBus”)
  - Classes like CommandMessage, EventMessage, NotificationMessage, etc.



Message Routing

- Broker daemon on every host
- Routing configuration tool
- Single exchange: “APERTIF”
- Routing based on subject
- Destination queues for each component

```
# Routing configuration for APERTIF
#
# Use the tool 'config_routing' to apply this table
[system]
name=APERTIF

[routes]
# APERTIF Routing tables
#
# routingkey  destinations
control:      SignalControl.ctl, DirectionControl.ctl
command:      SignalControl.cmd, DirectionControl.cmd
response:     SignalControl.notification, DirectionControl.notification
```

ASTRON
Netherlands Institute for Radio Astronomy

ASTRON is part of the Netherlands Organisation for Scientific Research

NWO
Netherlands Organisation for Scientific Research