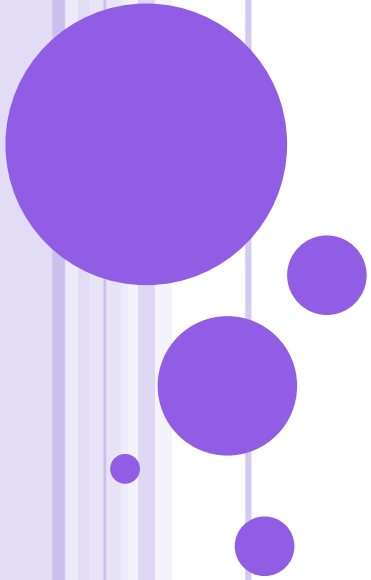


Telemetry

Revision

- Read the LT Telemetry presentation first, it contains useful background info and various tables and diagrams which I may not reproduce here.

Paradigms

- There are basically 2 paradigms used to obtain telemetry:-
  - Polling
    - Client sends a request for some sort of data.
    - Server handles request, obtains data, sends back.
      - Client is responsible for determining frequency of requests.
        - Can miss rapid changes.
        - Can overload system if polling too often.
  - Pub-sub
    - Client registers for specific feed(s).
    - Provider sends data back as soon as available.
      - Data is always most recent and fresh however fast it is changing.
      - Provider may loose client registration – client does not know.

Feed types

- There are basically 2 types of feed available.
  - Regular/continuous feeds.
    - These consist typically of measured values or states which can be sampled at intervals and for which it is not necessary to have every data point.
      - Mechanism positions, system states etc.
  - Event-driven feeds.
    - These feeds consist of events or state-changes for which the full history is needed to make sense of the data i.e. they cannot just be sampled at intervals.
      - Detection of the start and end of a task or scheduling sweep.
      - Seeing updates which occur at irregular intervals.

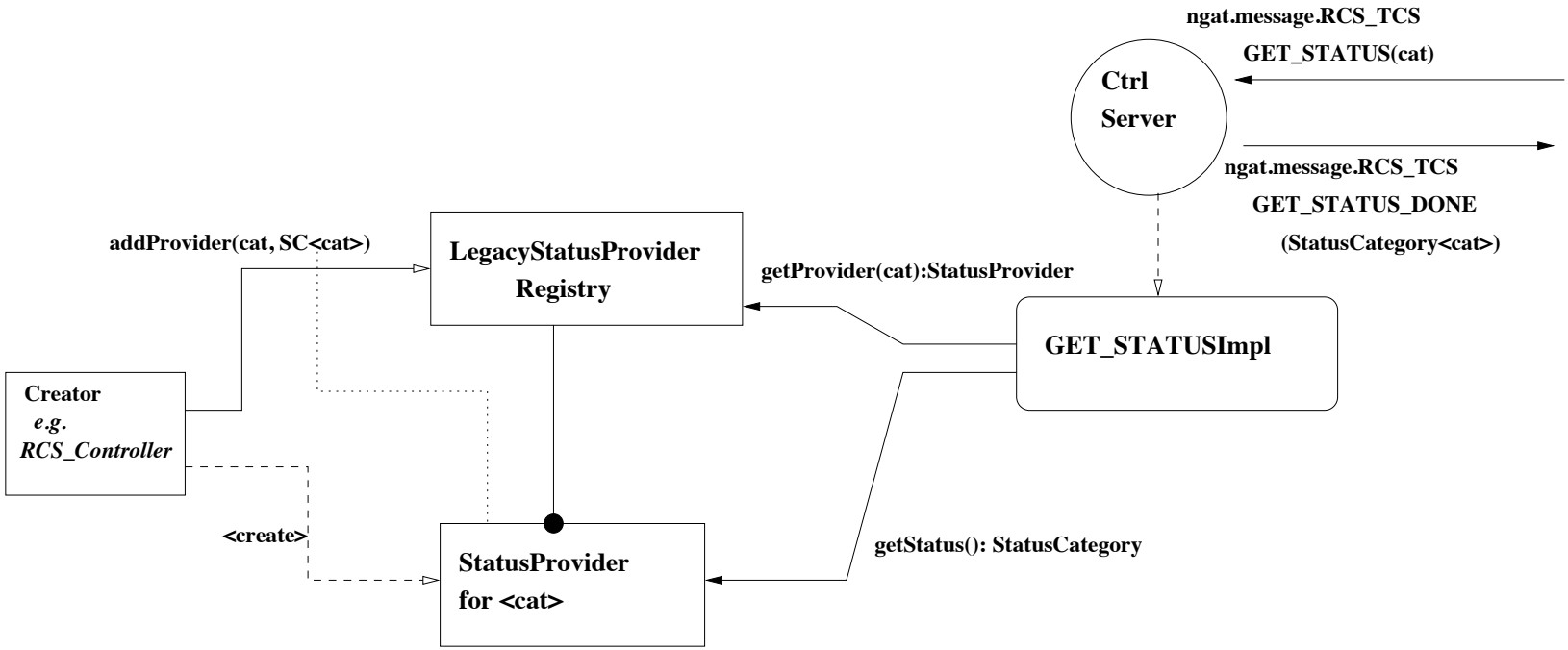
Systems in use

- There are 2 different telemetry systems in use at this time.
- Old system – based on polling was historically used by the RCSGui.
  - Due to network *features*, a modified version of this system is still required to obtain data for the Live Status web pages.
  - It is limited to providing telemetry from the RCS on a single port.
  - It is also limited to providing regularly changing feeds rather than event-driven feeds.
- New system is based on pub-sub.
  - Feeds are available from a number of originating systems:-
    - RCS, Scheduler, EMS, TCM, ICM.
  - Event-driven feeds are available.

OLD SYSTEM

- The RCS provides a JMS based control-server on port 9110.
- When a GET_STATUS request is received, the server creates a handler and processes the request.
- The category parameter is extracted and used to obtain a reference to a StatusProvider (for that category) from the registry.
  - E.g. MECH, AUTOGUIDER, METEO, RATCAM
- The individual providers having been previously registered against their category names.
- The StatusCategory object is packed into a GET_STATUS_DONE and sent back to the client.
- The client is responsible for deciding the polling rate.
- Providers obtain their statuses in various ways:-
  - Various *TcsStatusClients* for each TCS category.
  - *InstrumentStatusClients* for each instrument.
  - Certain internal providers within the RCS.
  - E.g. SM (state model), SEEING (predecessor to SkyModel)
- Most of these providers are now defunct due to the way in which status information is collated within the RCS now using TCM, ICM, EMS.

ARCHITECTURE – OLD SYSTEM



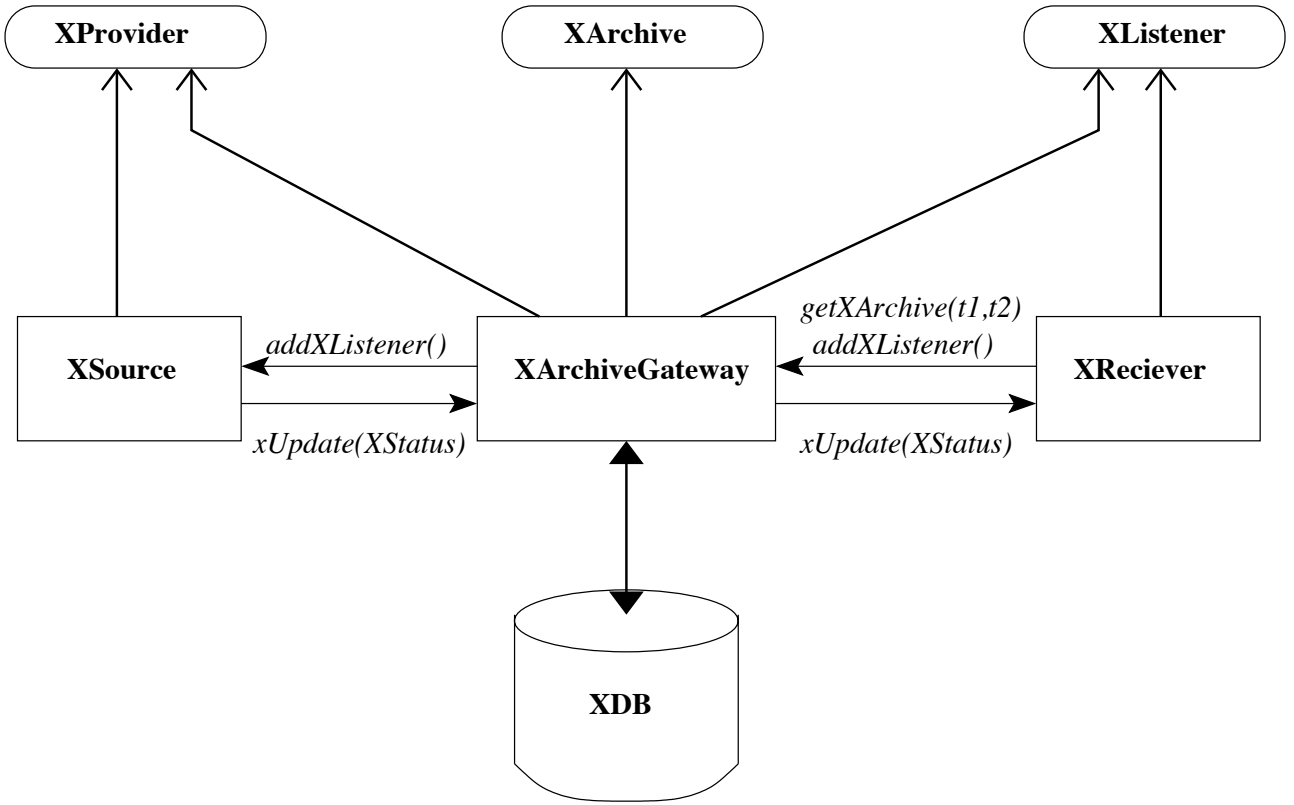
Old Providers

- Mostly in ngat.rcs.scm.collation package.
- Implement StatusMonitorClient interface.
- StatusMonitorThread associated with each
  - Calls on client to obtain status from its source at configured intervals.
  - TcsStatusClient (per TCS SHOW category).
  - InstrumentStatusClient.
  - RCSInternalStatusClient (statemodel etc).
  - URLStatusClient – reading from data files e.g. cloud, dust, robodimm created by cron scripts.
- Now all defunct.

New System

- The RCS subsystems – TCM, ICM, EMS, ERS, TMS, etc and other systems e.g. SCHED, implement specific providers for the various types of status feed.
- These providers act as the publishers which external clients attach to as subscribers.
- Internally a provider is part of the running system, i.e. there could be a line of code at the start of some important operation (X) where a call is made to a method which sends out an event notification to subscribers.
  - startingOperationX(subs); // notify registered subscribers…
- It is unwise to have it make the publishing calls out to the clients directly, a slow receiver could potentially block a time-critical operation.
- A gateway is introduced to handle the direct publishing from the provider.
  - Gateway acts as publisher to client
  - Acts as client to real provider.
  - Disconnects publisher from actual client.

Architecture – New system



Components

- Typical components which go to make up a telemetry feed/stream.
- XProvider – publisher interface
  - Typically has methods to add and remove subscribers
    - addXListener(xl), removeXListener(xl)
- XSource – an implementation of XProvider
  - This is probably a class which is an integral part of the running subsystem.
- XListener – listener inteface
  - Typically has callback methods like
    - xUpdated(Xstatus), xEventA(), xEventB(), etc
- XReceiver – an implementation of XListener
  - Could be a GUI component or logging class.
- XArchive – archive retrieval interface
  - Typically has methods to obtain historic status.
- XArchiveGateway – a class which implements all the interfaces and acts as both a publisher endpoint for remote clients to connect with and keeps the key activity of the source separate from its publishing commitments to avoid blocking.

Example

- Telescope system telemetry feed.
- BasicTelescope is the primary publisher and implements TelescopeStatusProvider.
- It is also the class which collates status from the TCS and Autoguider.
- TelescopeArchiveGateway subscribes to Telescope as a listener. It stores the received statuses and then posts these out to real clients under control of a processing thread.