LT Operations Monitoring

A new architecture for telescope systems telemetry

Introduction

- Introduction to systems monitoring
- Telemetry architecture
- Operations UI architecture
- Demonstration

PART 1 - Systems Monitoring

Systems Monitoring

What is it and why do we need it ?

- Finding out what the system is doing.
- Measuring its performance against metrics.
- Detecting problems early.

Solutions

- Logging

  text generated by embedded code

  A human readable account of what the various software components of the system are doing.

- Telemetry

  data generated by embedded code

  structured data which can be used by remote clients to interpret and analyse what the system is doing.

Logging

- May be formatted to show things like:
  - Time
  - Source component
  - Level of detail
- Keywords to highlight specific categories of message.
  - Error
  - StartingOperation
  - DataReceived
  - CalculationResult
- Data logging – logging values of environmental variables
  - (sometimes confusingly called telemetry)

Telemetry

- Event notifications
  - starting-scheduler-sweep (time, swpno, conditions)
  - completed-group (group, duration, status)
- Environmental variables
  - buffer-size
  - instrument-temperature
  - rotator-position
  - wind-speed

Logging versus Telemetry

Telemetry differs from logging in terms of both goals and content.

- Logging attempts to provide information for both operational monitoring and software debugging.
- In doing so, it is required to serve two masters and often neither of them particularly well.
- The software developer is constantly faced with the problem of what to log with the result often being too much and too little is logged, at the same time.
- Telemetry is focused solely on providing information that is useful for understanding the operation of the software from an operational perspective.

Operations Monitoring

- How healthy is the application ?
- What errors is the application encountering that block proper operation ?
- How well is the application performing?
- What external dependencies does the application have that might impact it's operation ?
- Are the business metrics in the appropriate ranges ?

Software versus Operations

- Developers are more focused on the flows of the application internally while operations are more focused on ensuring the business goals are being met by the system.
- Developers need to understand why the application fails in a particular way while operations needs to understand what components can impact the application and how those components are behaving.
- The challenge to date has been that logging frameworks, which are focused on the application developer have been pressed into the secondary duty of providing operational information as well.

For and against

	Logging	Telemetry
For	• Trace events in sequence • Human readable • May be able to extract info using scripts	• Machine readable • Interpret sequences • Disentangle multi-threaded issues • Extract timing information • Instantaneous display
Against	• Difficult to disentangle multi-threaded logs. • Difficult to extract embedded sequences • Scripts reliant on constancy of message format.	• Needs clients written and modified if message format changes • Not immediately human readable

Performance analysis

Measuring system metrics

- Low level logging
  Simple logging.
- Medium level logging
  Add categories keys, operation timings
- High level logging
  Operation statistics
- Embedded monitoring
  Full monitoring, analysis, statistics.
- Telemetry
  External client performs analysis and statistics.

Metrics

Monitoring System	Operator effort	System intrusion
Low level logging	V. High	Low
Medium level logging	High	Medium
High level logging	Medium	High
Embedded monitor	Low	V. High
Telemetry	Low	Low

Scenario

- System has numerous processes running in parallel and interleaved.
- We are only interested in instances of process (A).
- While these (A)'s are running, associated sub-processes B,C, D, E etc will also run but not always the same ones and not in same order.
- Aims are to measure:
  - Average run time of an (A) process.
  - How many sub-processes run by (A).

Scenario solution (1)

Low level logging

- Can we identify in log which processes are (A) ?

  Starting process A-16
- Do we log both start and end of an (A) process ?

  End process A-16
- Can we identify B, C, D etc in the logs ?

  Starting sub-process B-3

  Starting sub-process A-16.B2

Just need to check these logs !

- Might be able to grep for *starting process* and then
- manually go thro identifying which type
- note start and end times
- work out duration
- count number of other processes starting and ending in that period
- then sum up and calculate average etc

Scenario solution (2)

High level (intelligent) logging

- Embedded code works out the start and end of each process and totals up the run-time, running average and standard deviation.
- Sometimes processes B,C,D etc run outside the context of a process (A) so the code needs to be doctored to push various flags around telling the monitoring code whether to record stuff or not.

Scenario solution (3)

Telemetry

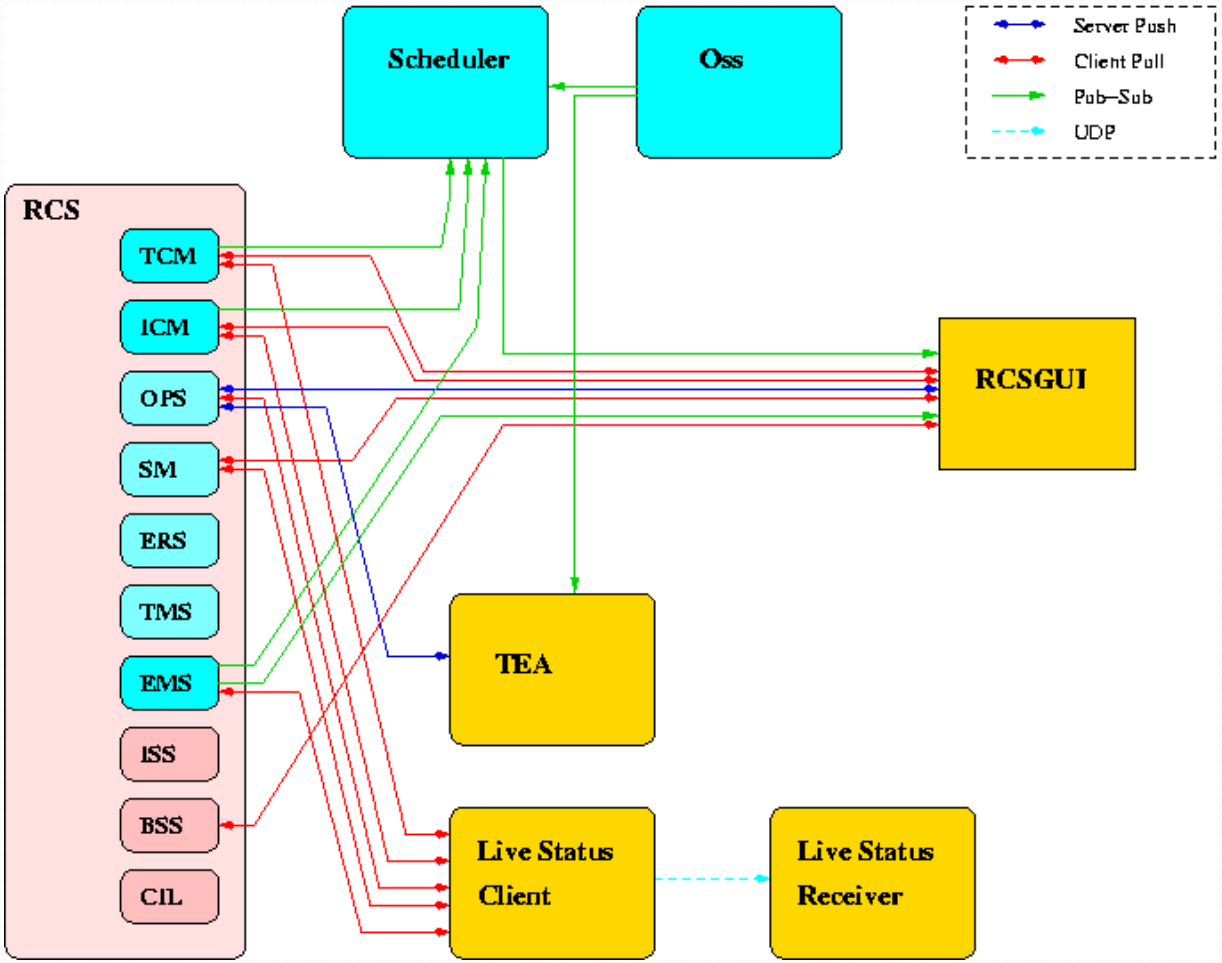
- System sends out messages when each process starts and ends along with time and process class.
- Receiving client extracts sequence information, calculates average, std dev etc.
  - Plots graphs of runtime against time,
  - histogram of range of runtime and number of sub-processes,
  - data fields showing current runtime and number of sub-processes running,
  - indicator showing red if runtime is too high, green if in-range.

Is it working ?

- The green light is on so all is well.
- I see the run-time has settled down over the last hour.
- We seem to be running a lot of sub-processes at the moment

PART 2 – Telemetry Architecture

Old Telemetry Network



Design Requirements

- Different streams of data should all look essentially the same - *consistent architecture*.
- Extendable to new systems.
- Able to obtain historic data in addition to live data.
- Event data available as it happens.
- Detachment of source from receiver.

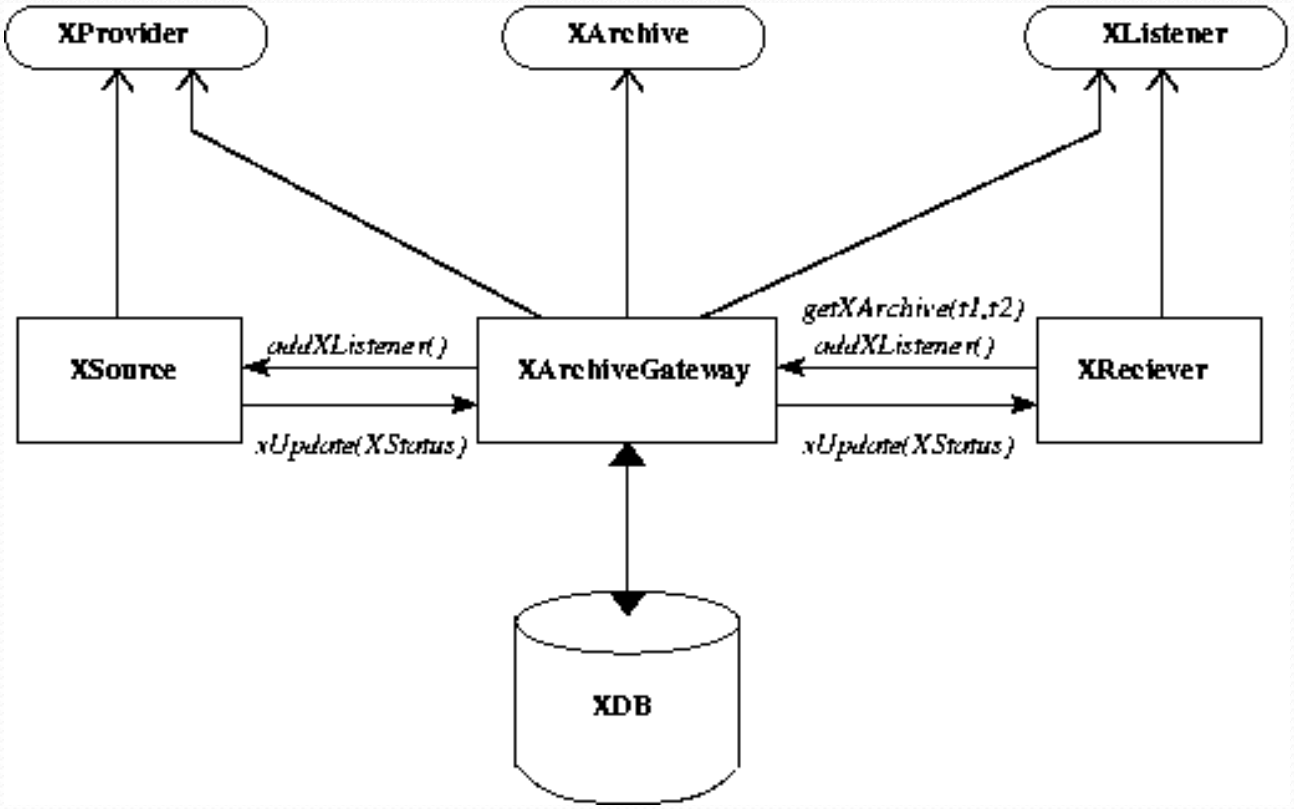
Getting the data

- Polling
  - Client sends out requests to server at regular intervals.
- Publish-Subscribe
  - Client subscribes to feed, system sends out data as available.

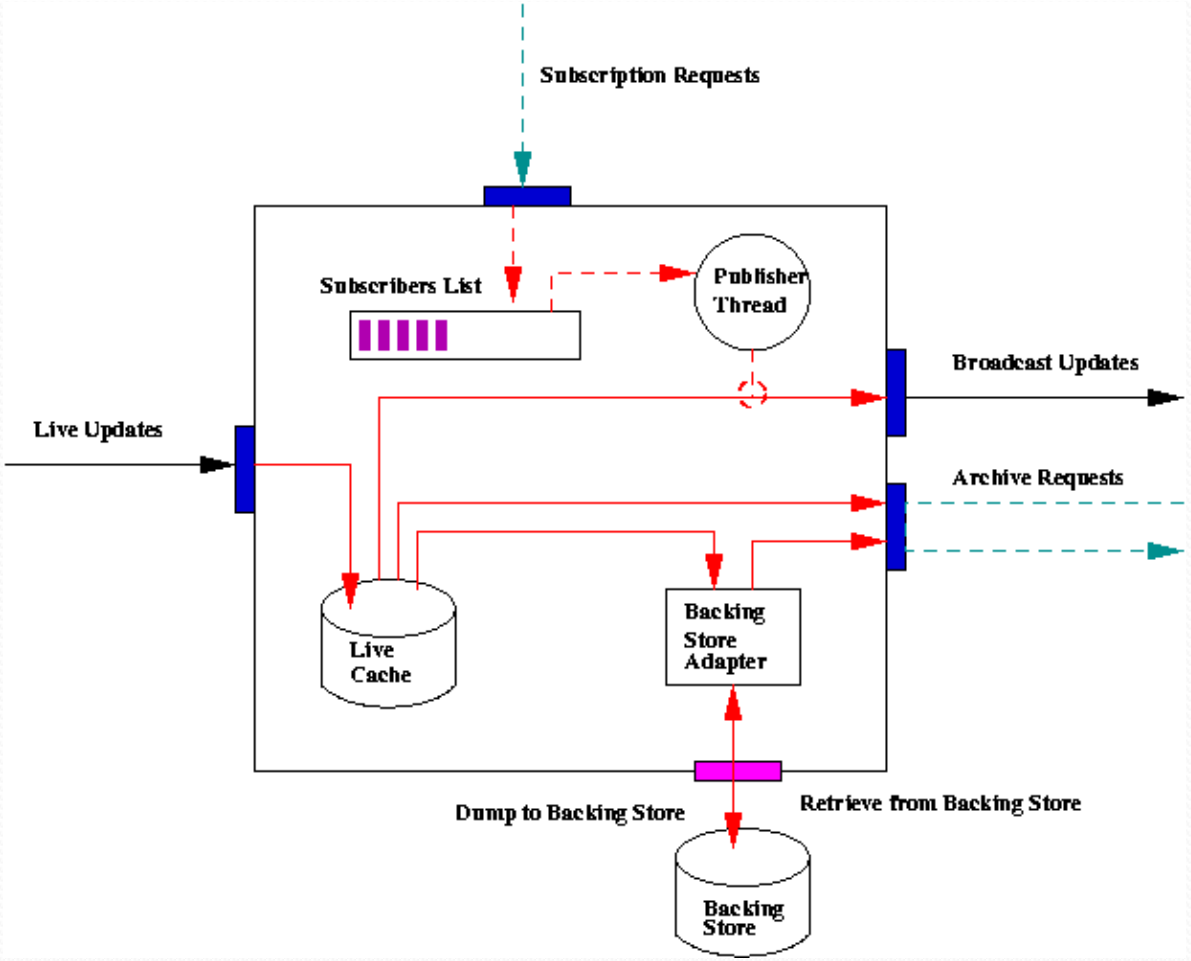
Polling v Pub-Sub

	Polling	Publish-Subscribe
For	• Conceptually simple	• Minimum intrusion to system. • Up to date, speed matching. • Event handling. • Add new data categories easily.
Against	• Infrastructure at system. • Poor for event handling. • Over/under sampling. • Client polling - infrastructure and configuration.	• Re-registration “polling”.

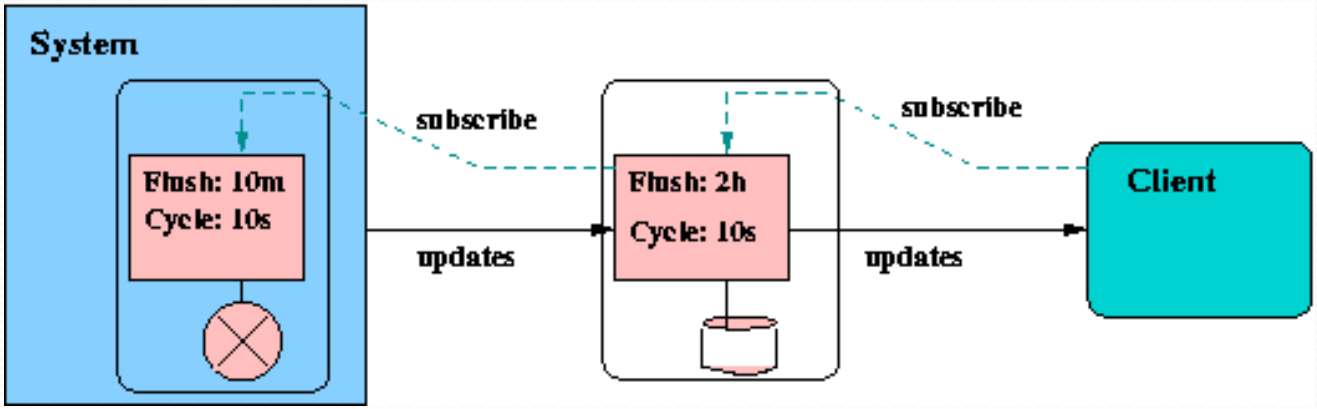
Telemetry Architecture



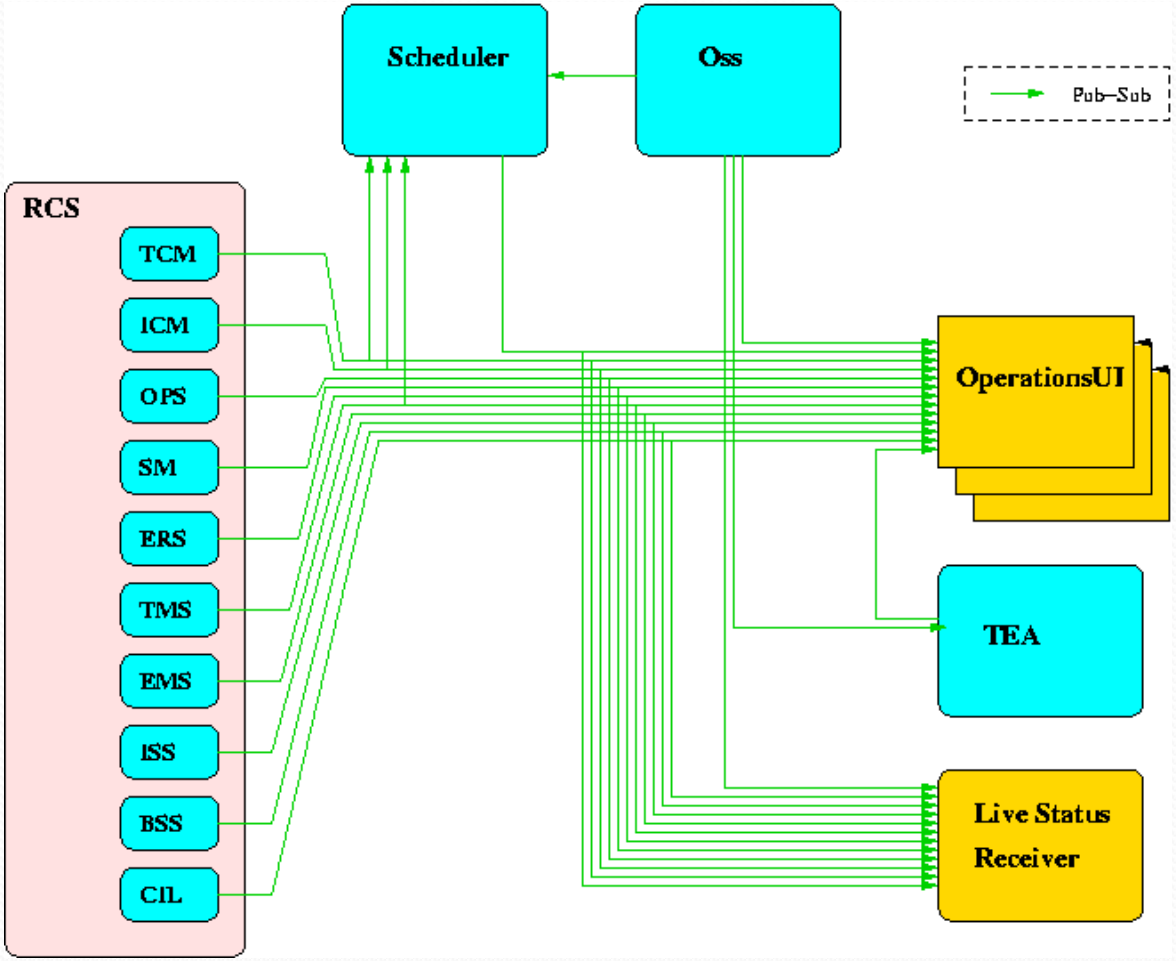
Gateway architecture



Single System

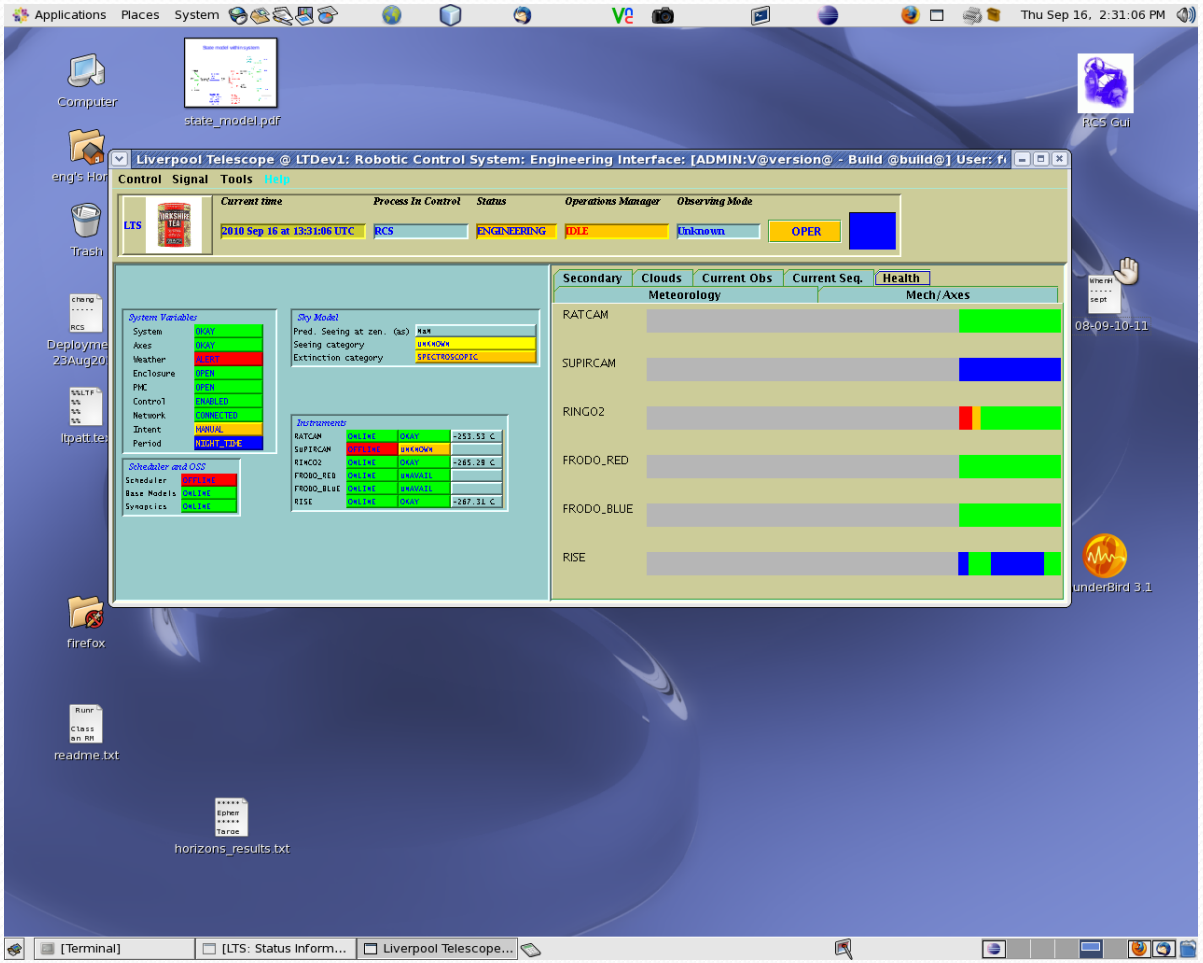


New Telemetry Network



PART 3 – UI Architecture

Old RCS-GUI


Applications Places System
Thu Sep 16, 2:31:06 PM
Computer
state_model.pdf
RCS Gui
Trash
Liverpool Telescope @ LTDev1: Robotic Control System: Engineering Interface: [ADMIN:V@version@ - Build @build@] User: f
Control Signal Tools Help
Current time
Process In Control
Status
Operations Manager
Observing Mode
LTS
2010 Sep 16 at 13:31:06 UTC
RCS
ENGINEERING
IDLE
Unknown
OPER
Secondary
Clouds
Current Obs
Current Seq.
Health
Meteorology
Mech/Axes
RATCAM
SUPIRCAM
RING02
FRODO_RED
FRODO_BLUE
RISE
08-09-10-11
underBird 3.1
firefox
readme.txt
horizons_results.txt
[Terminal]
[LTS: Status Inform...
Liverpool Telescope...

Design Requirements

- Displays from different sources (themes/topics) collected together.
- Easy to personalize – individual user configuration of layouts and content.
- Able to link components to do searches, tracing, auditing.

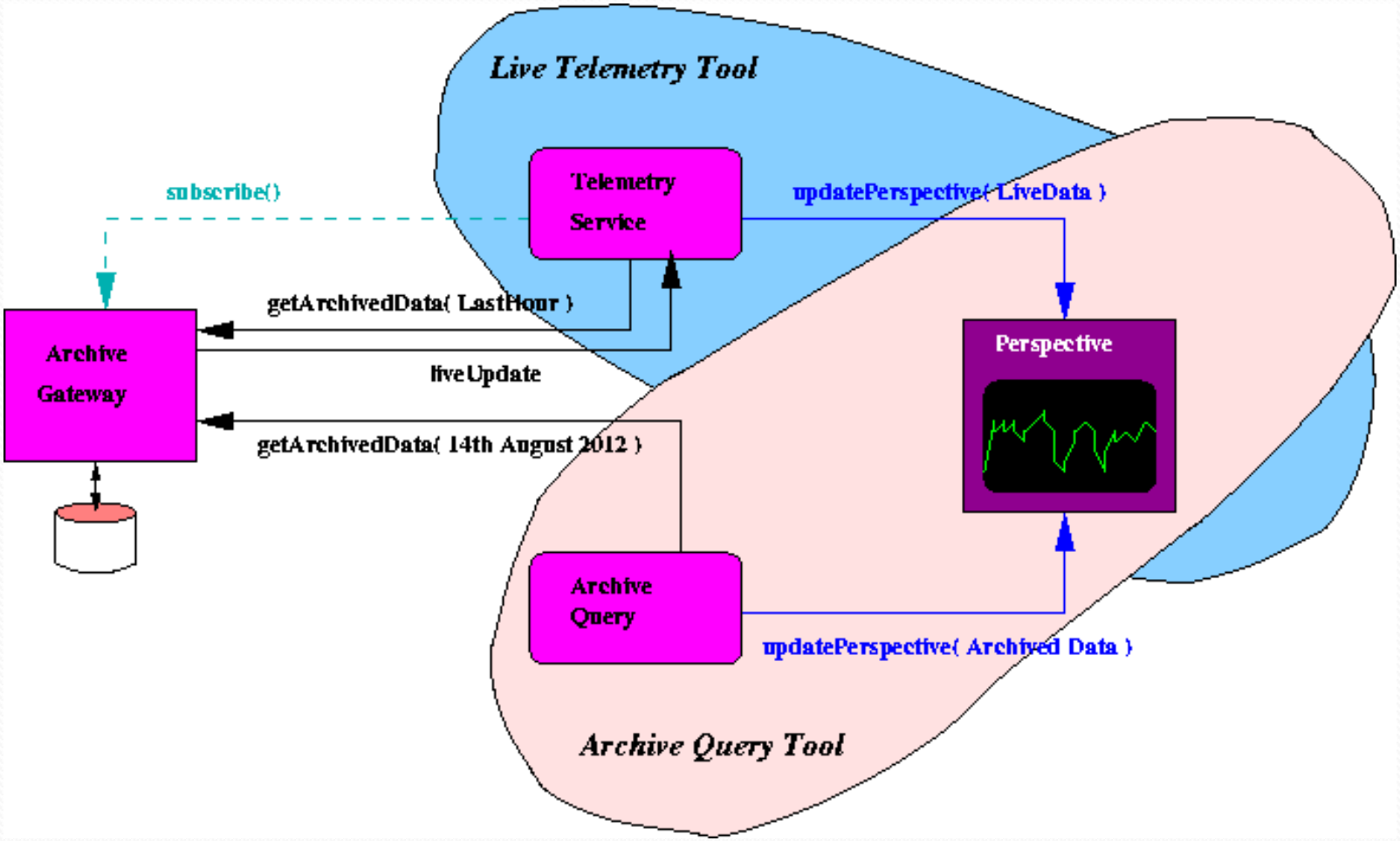
Terminology

- Perspective
  - A component which processes all of the data relating to a particular theme/topic.
  - Typically this originates from a single source/provider.
  - Contains various visual display sub-components.
    - Graphs, Histograms, Data fields, State indicators, Timelines, Tables, Trees, Custom components.
- Display
  - A frame or window containing one or more perspectives.
- Layout
  - A collection of displays.

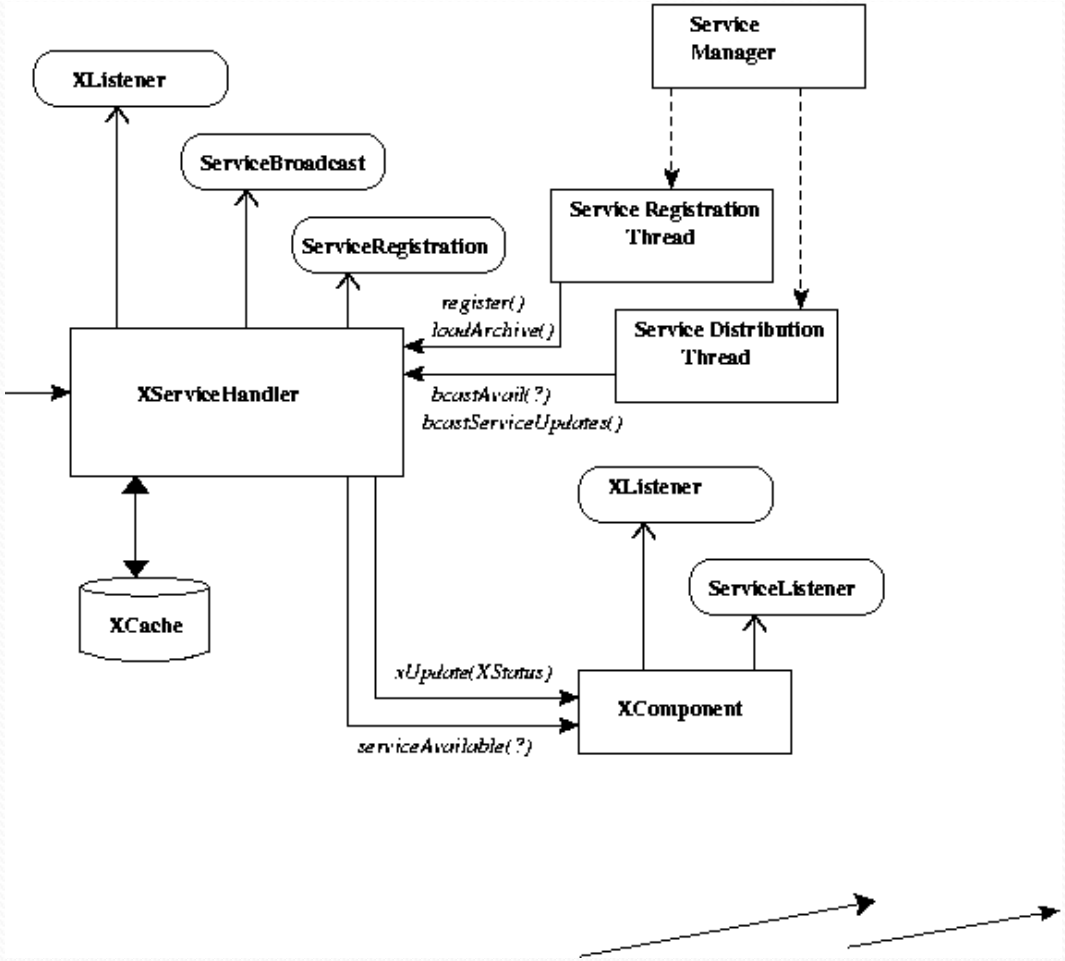
Perspectives

- Scheduling
- Instruments
- Operations
- Telescope
- Meteorology
- Task Management
- Reactive State Model
- PhaseII
- Tracking
- Services

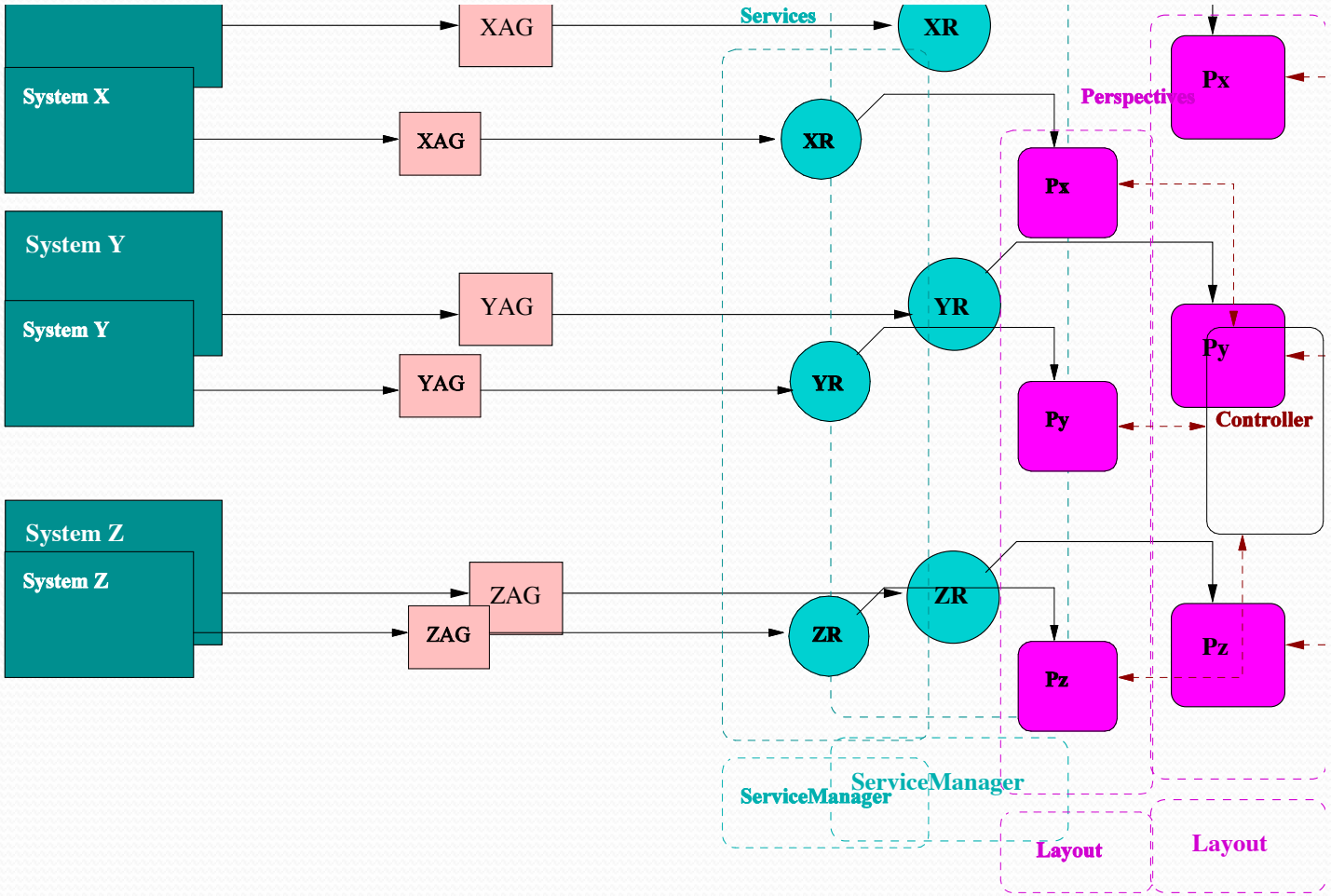
Communications



GUI Architecture

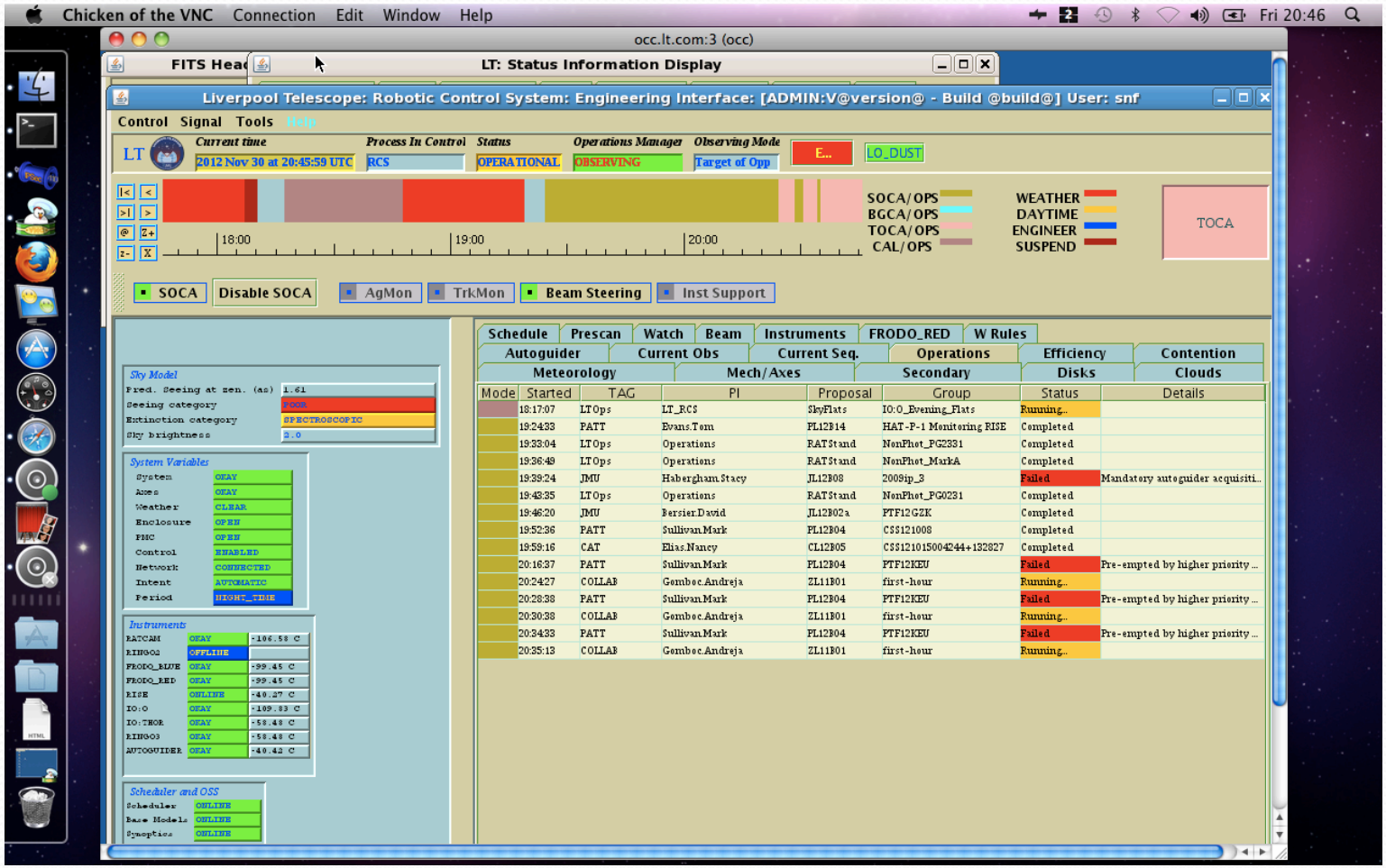


Multiple Systems

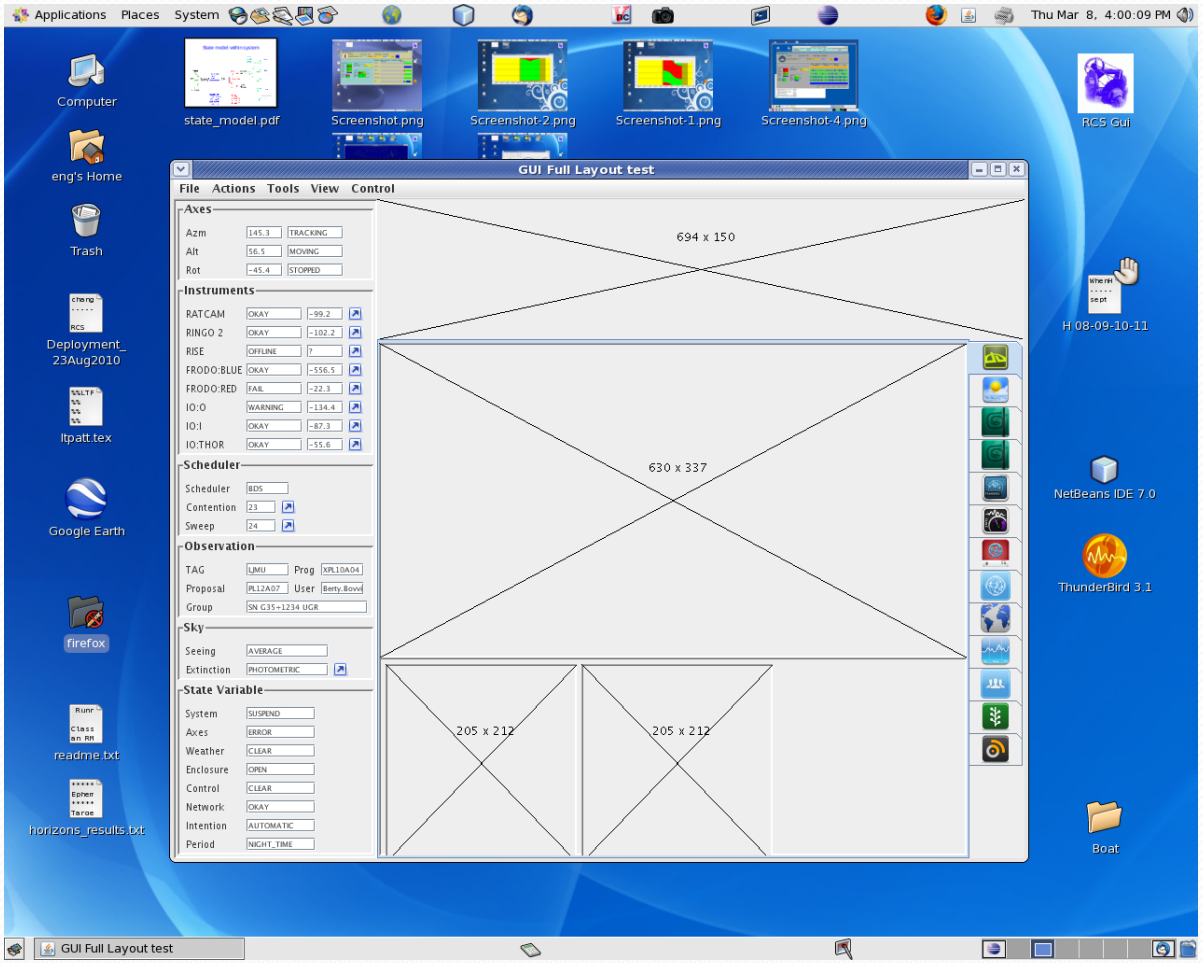


PART 4 – Operations UI demo

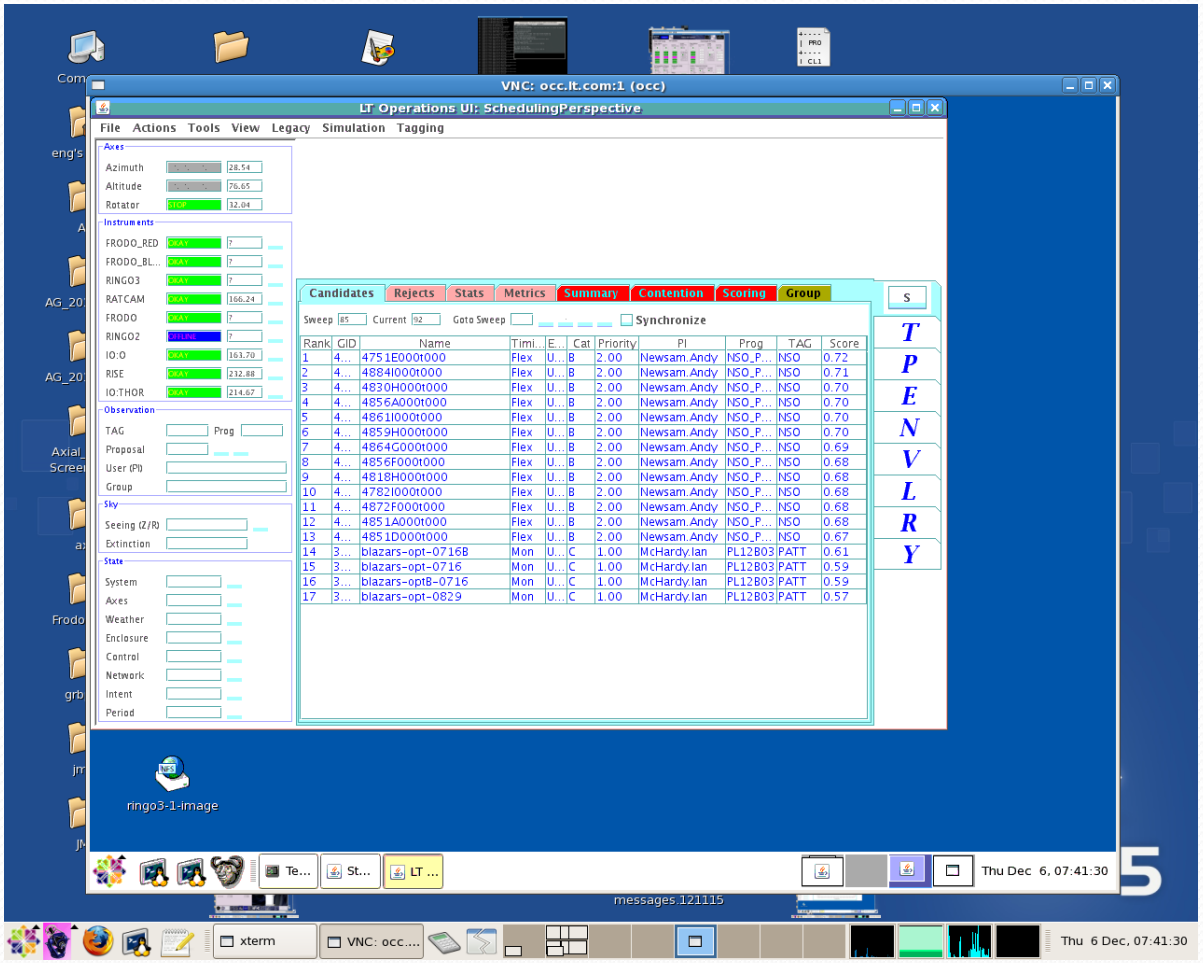
Current RCS-GUI



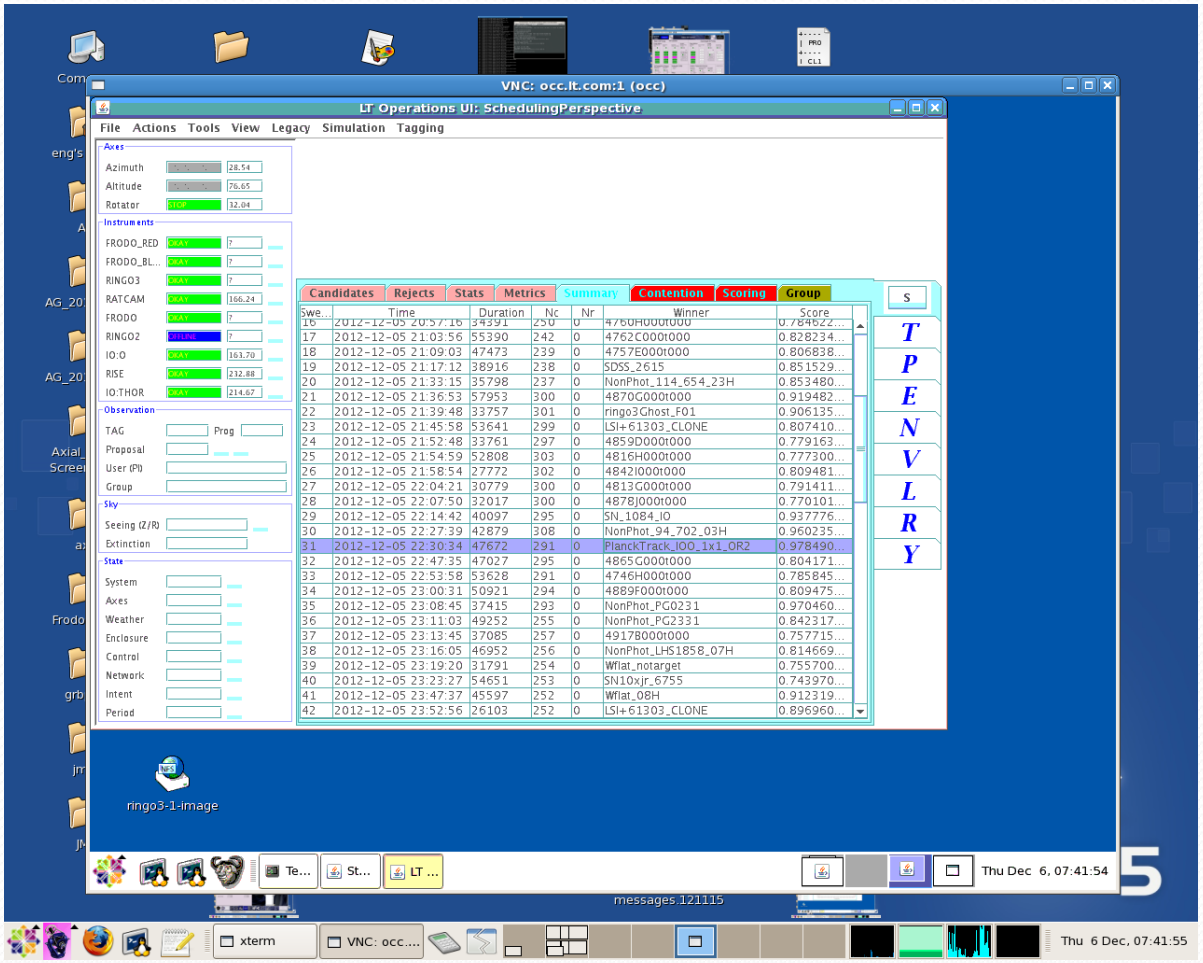
Ops UI: General Layout



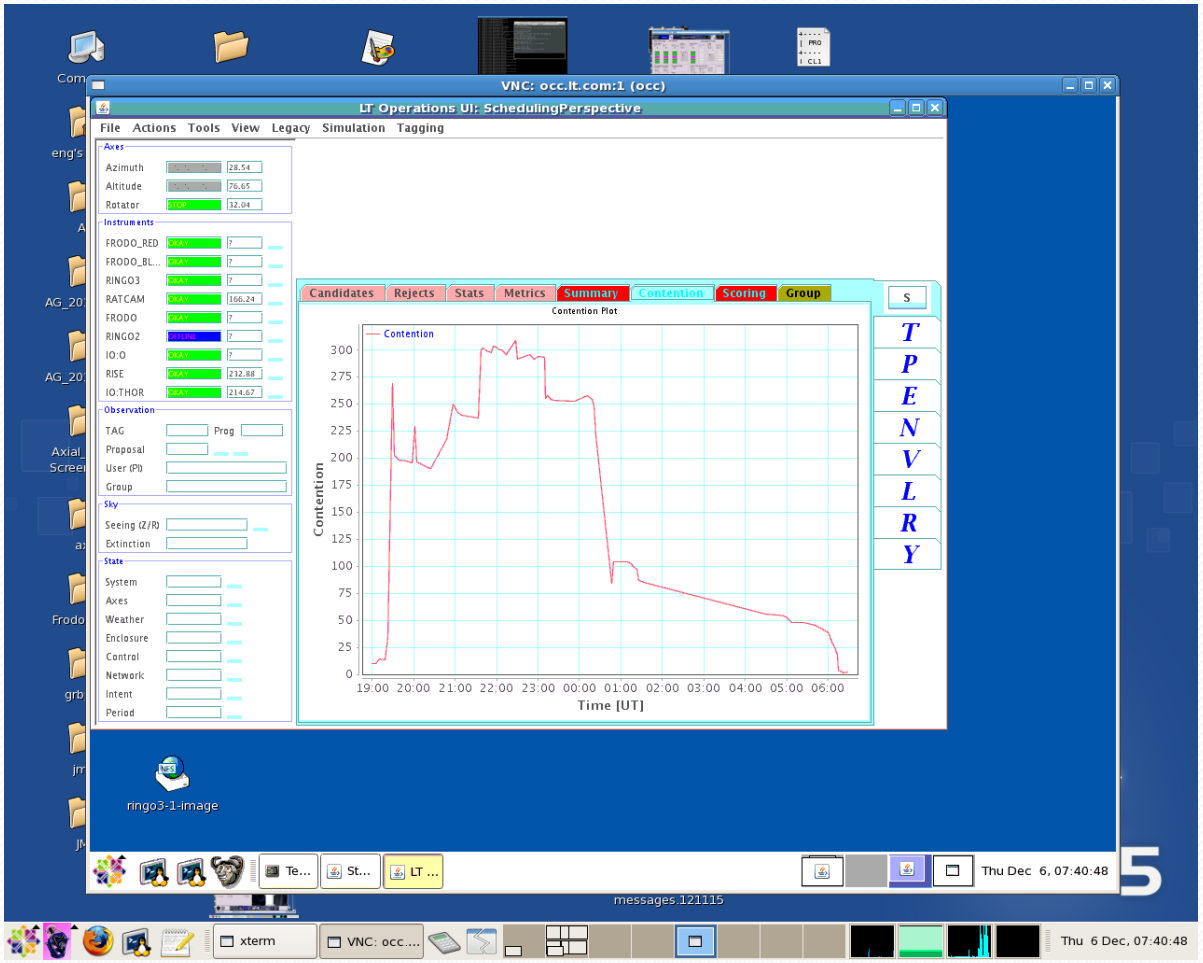
Ops UI: Sched:Candidates



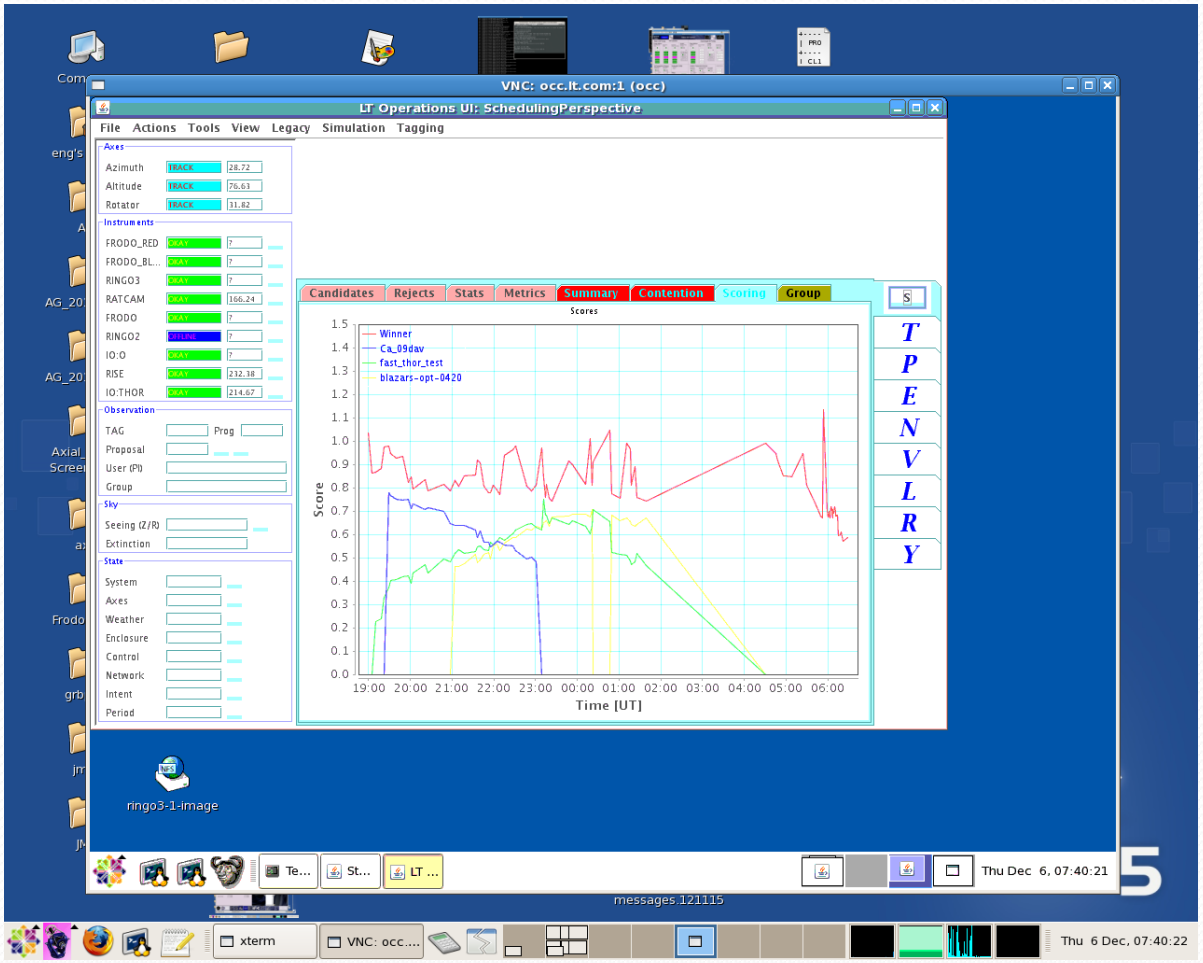
Ops UI: Sched:Summary



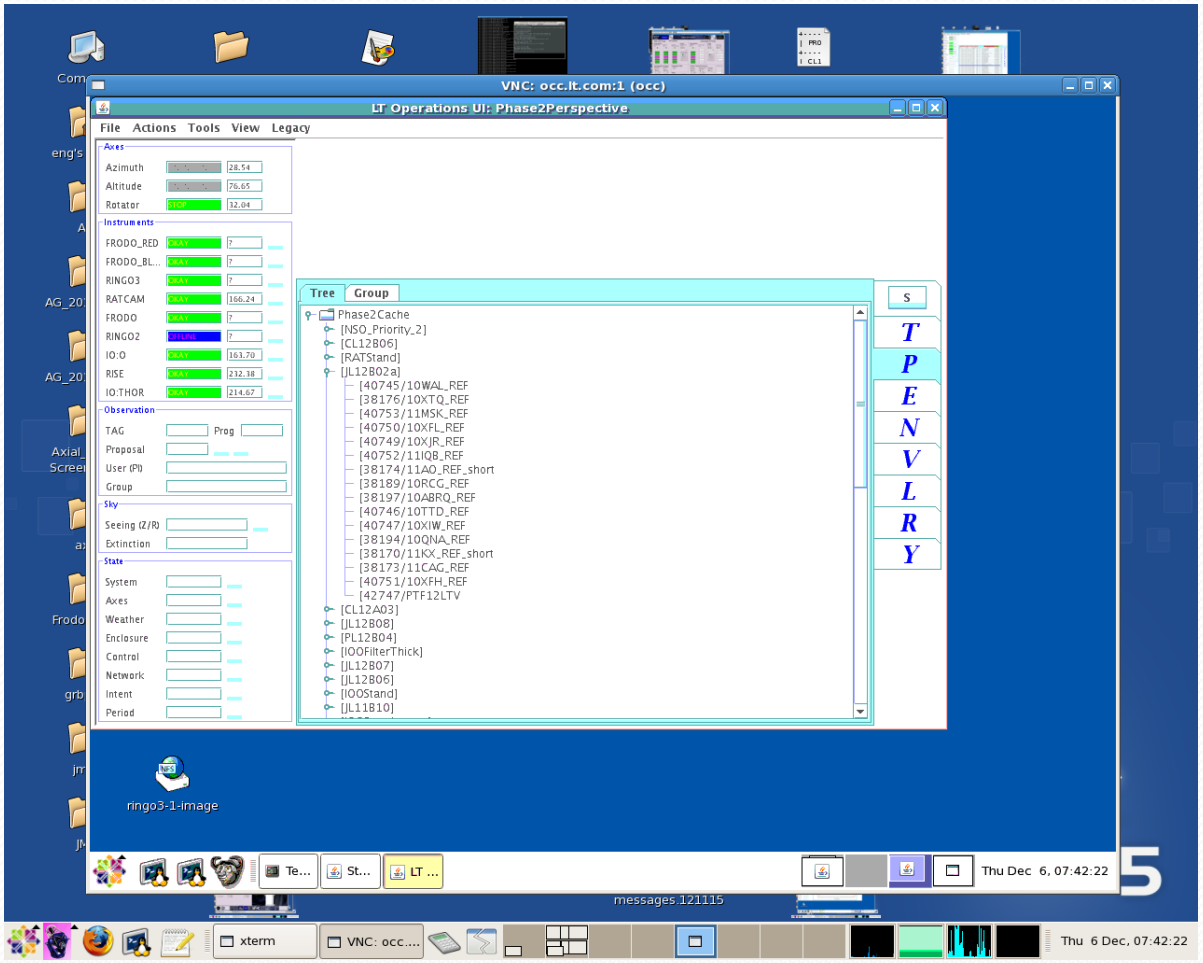
Ops UI: Sched:Contention



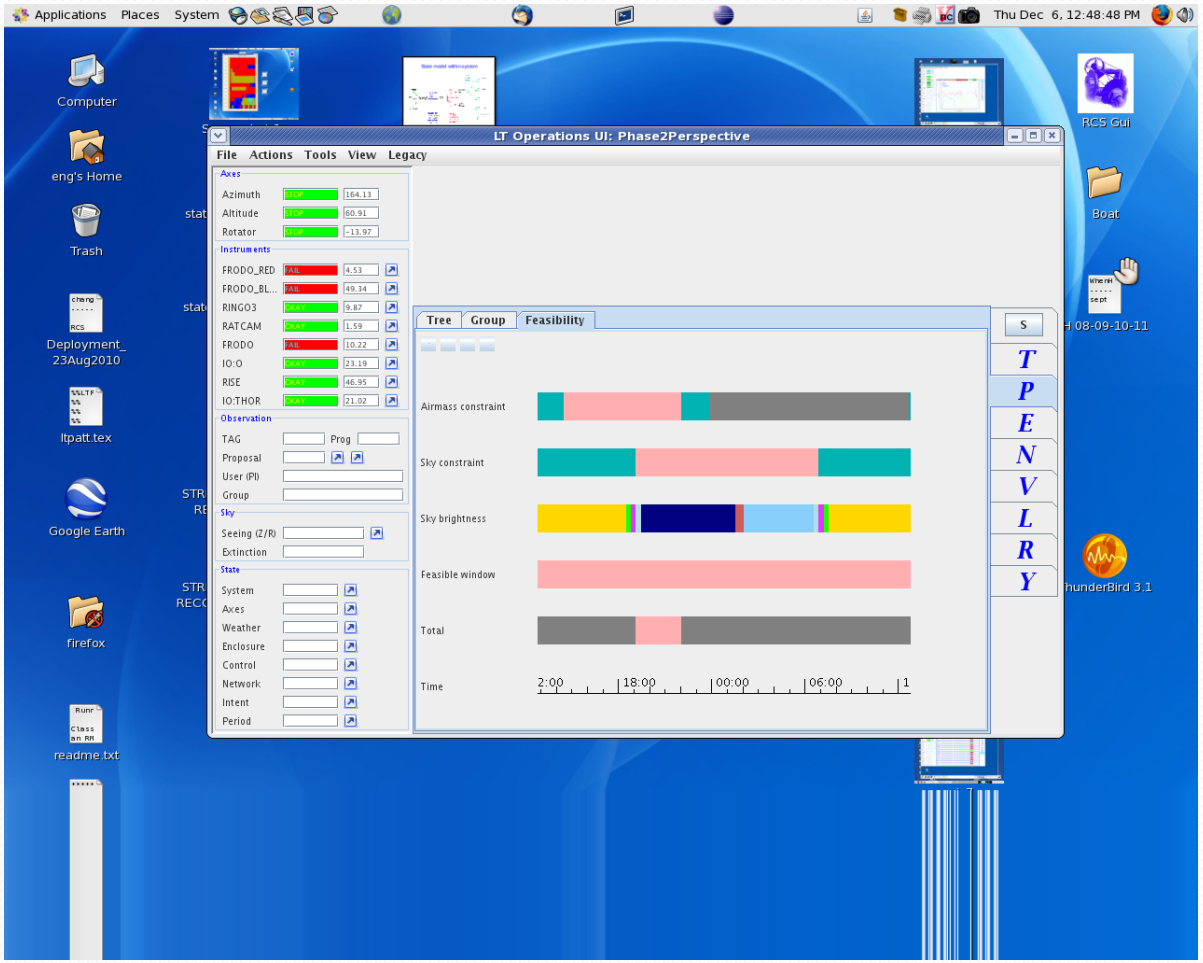
Ops UI: Sched:Scoring



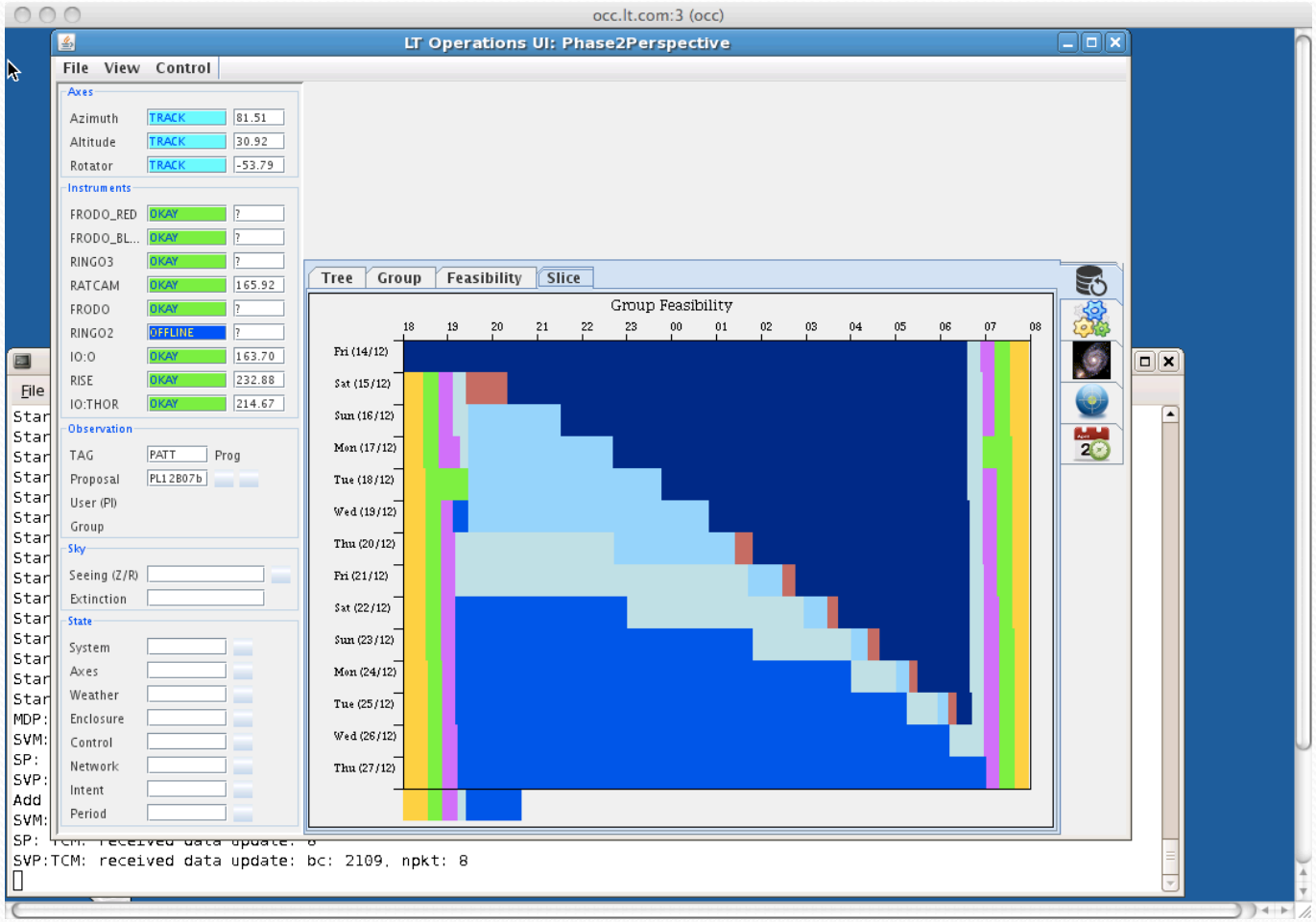
Ops UI: Phase2:Tree



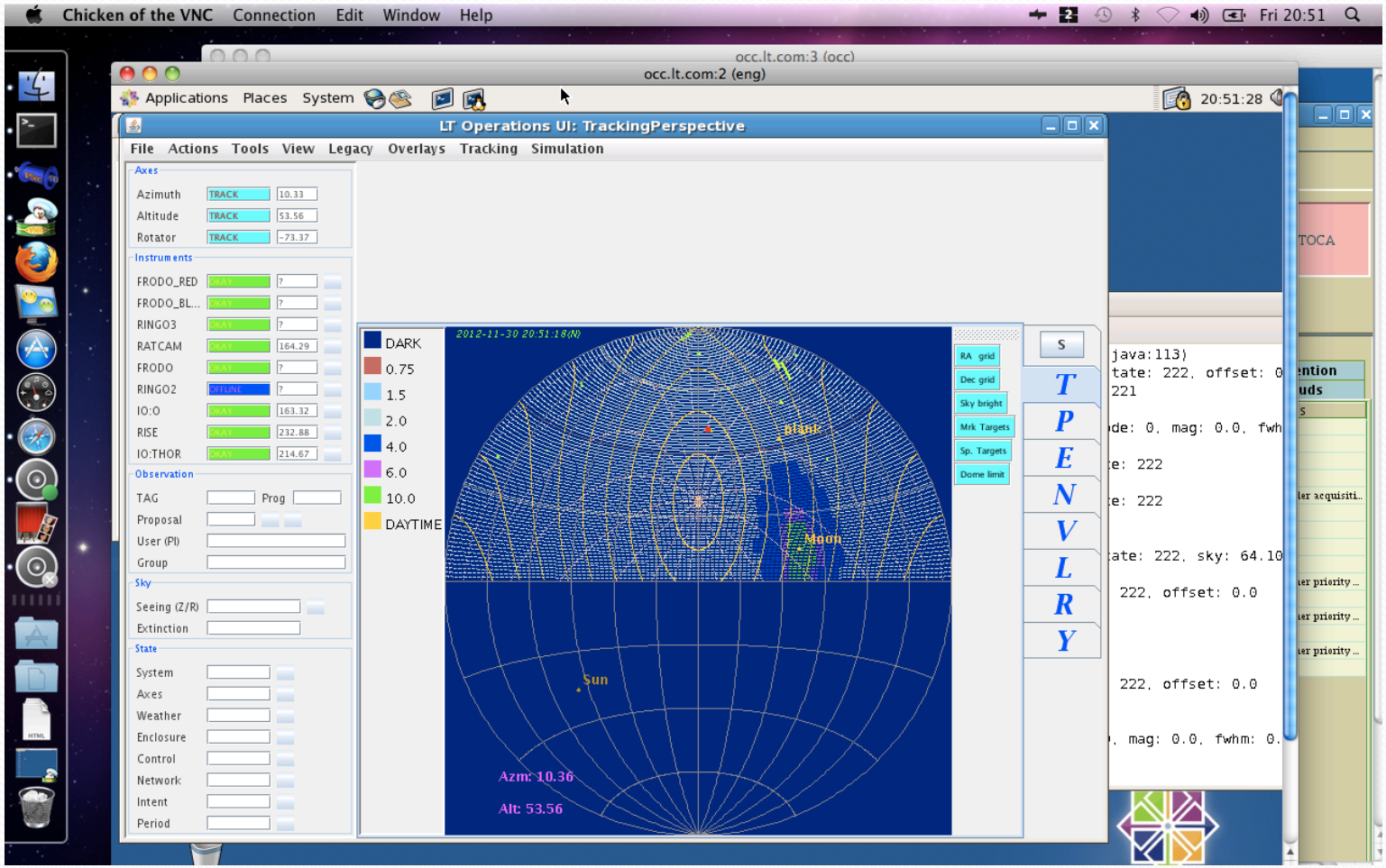
Ops UI: Phase2:Feasibility



Ops UI: Phase2:SkyBSlice



Ops UI: Tracking:Altaz



Ops UI: ERS:MeteoGraphs