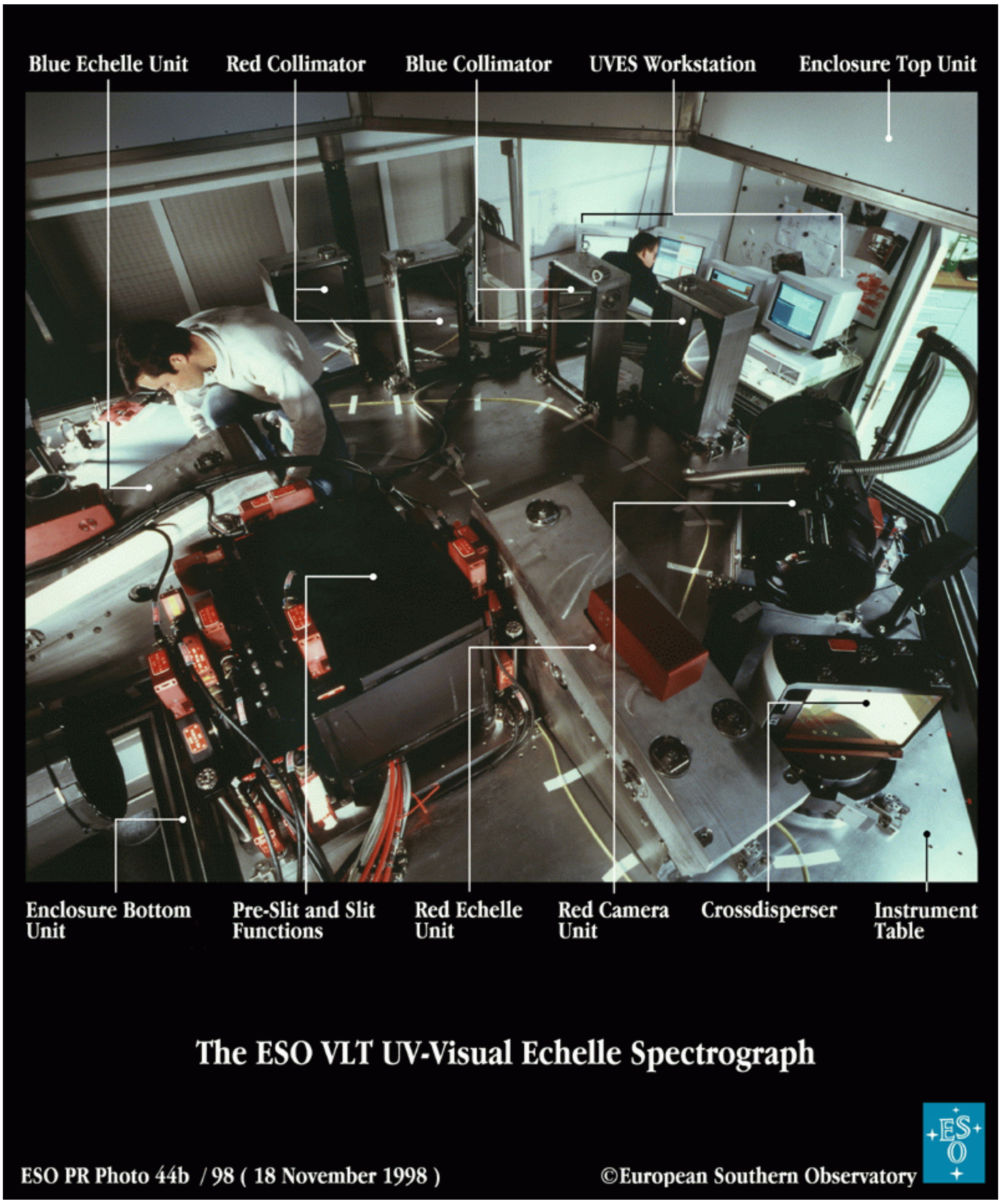
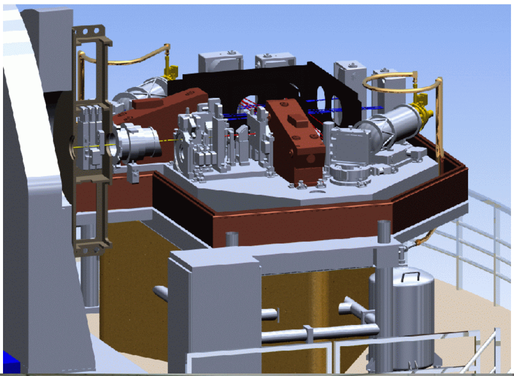
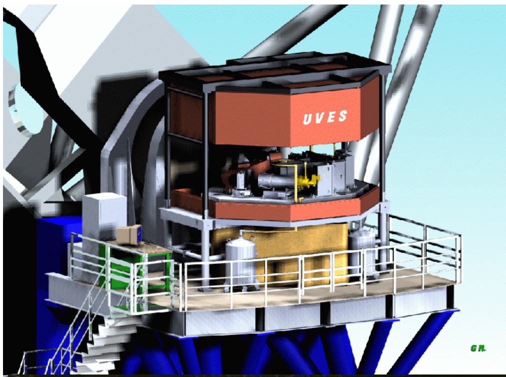
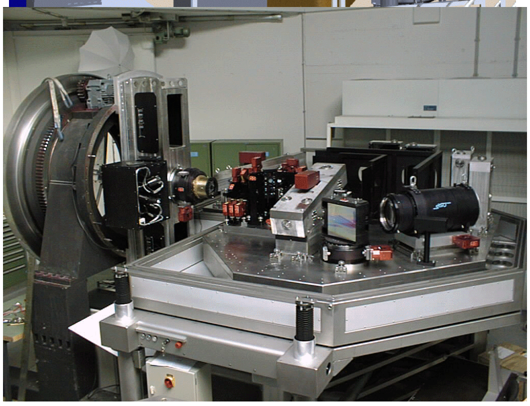
Uv-Visual Echelle Spectrograph



http://www.eso.org/instruments/uves

UVES integration




In total 122 devices to control on 7 computers

- 26 motorised functions
- 20 lamps/shutters
- 72 sensors
- 4 detector systems:
    - 2 scientific CCDs (24 Mpixels in total)
    - 2 technical CCDs (slit viewers)

Project milestones

✓**JAN94** Preliminary Design Review

✓**JUN96** Final Design Review

✓**SEP98** First Light

MAY99 Acceptance Europe

OCT99 Commissioning phase 1

DEC99 Commissioning phase 2

FEB00 Internal release for Science Verification

APR00 External release

(✞ ?)

✌ or ✝ ?

✌

- PDR: We understood ALL Requirements
- FDR: We defined a solution to ALL critical issues and verified with a prototype
- Acceptance Europe: There is NO pending issue
- Commissioning: We managed to keep cool
- All over the project: use standards and keep Sw AND Instrument under configuration control

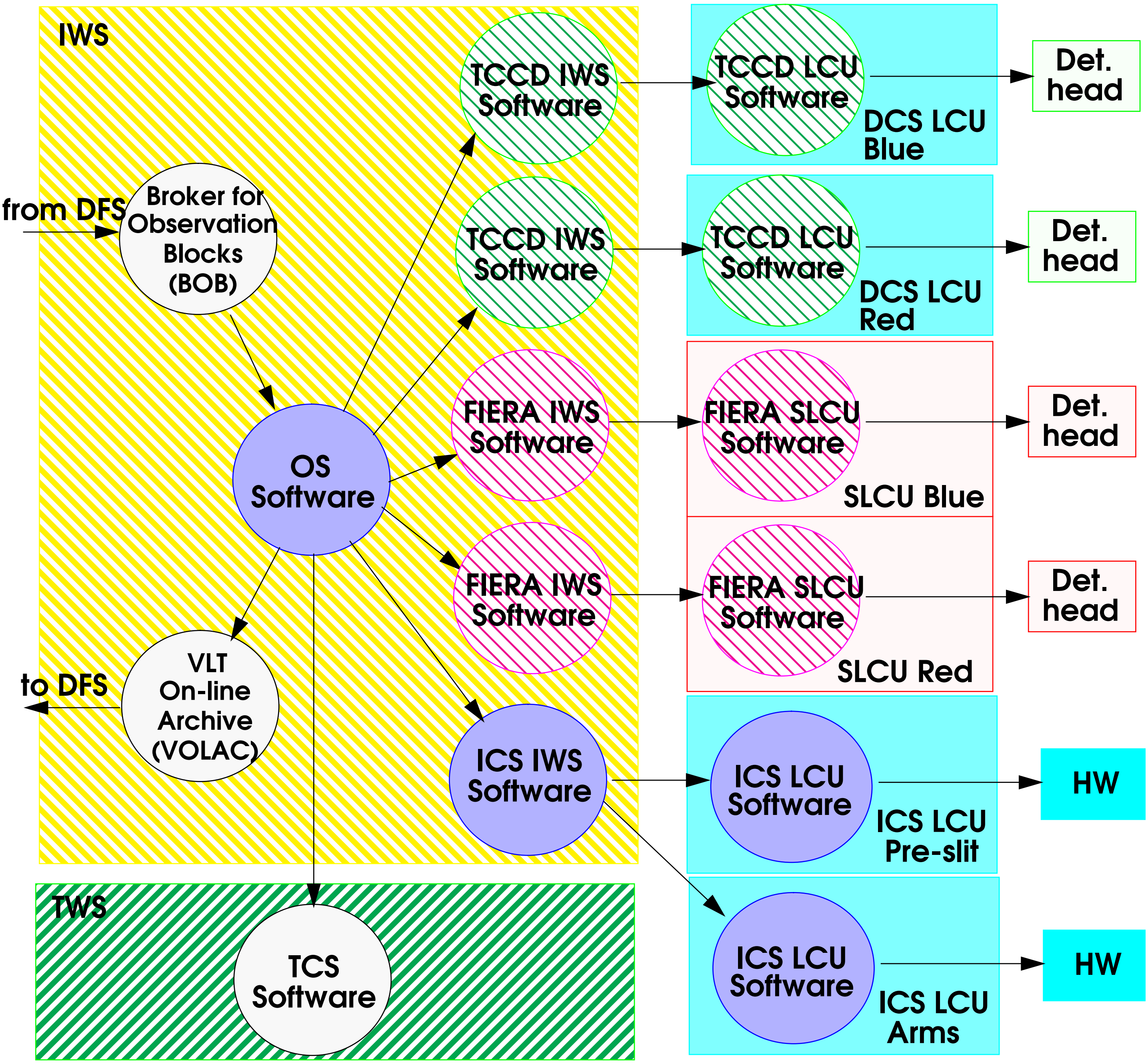
✝

- PDR + FDR: loss of time: we want to code !!
- Acceptance Europe: we will finish to code at Paranal
- All over the project: no standards: we can do it better

Sw developed at several locations ?

- If possible, **do not split responsibilities within the major modules** (DCS, ICS, OS, MS).
  UVES: OS, MS --> Trieste Observatory
  ICS, Integration --> ESO
- If hw available on one side only, **ICS** must provide a **good level of simulation**
- ICS should better follow a top-down approach, in order to **provide sooner the interface layer to the other party**
- Do not underestimate the **time and effort needed for integration** !!
- **Integration** does NOT only mean *put pieces together*, but **consists also of scripts/tools**, e.g. to build the sw and the environments, to install the instrument configuration etc.
  Do NOT forget to assign a **responsible** to it !!
- We **used the VLT SPR system** already for integration as problems/solutions repository.

Sw architecture



General hints

- Keep the **interface between modules as simple as possible**
- When writing WS sw, remember that the **scan system works asynchronously** with respect to the Message System.
- On WS use only the **message system as inter-process communication mechanism**.
- Define the **OLDB contents at an earlier stage** (structure at PDR, detailed contents at FDR)
- **Do not invest too much time in** making beautiful **GUIs**. At Paranal everything will have to fit into 2 screens.
- Having **clean installation procedures** from and earlier stage, speeds up development.
- **Test sw** is part of the Instrumentation Sw !! Allocate time and resources to it in due time.

Installation procedures (*uvins*)

mkdir $UVES_SOURCES; cd $UVES_SOURCES

cmmCopy uvins; cd uvins/src

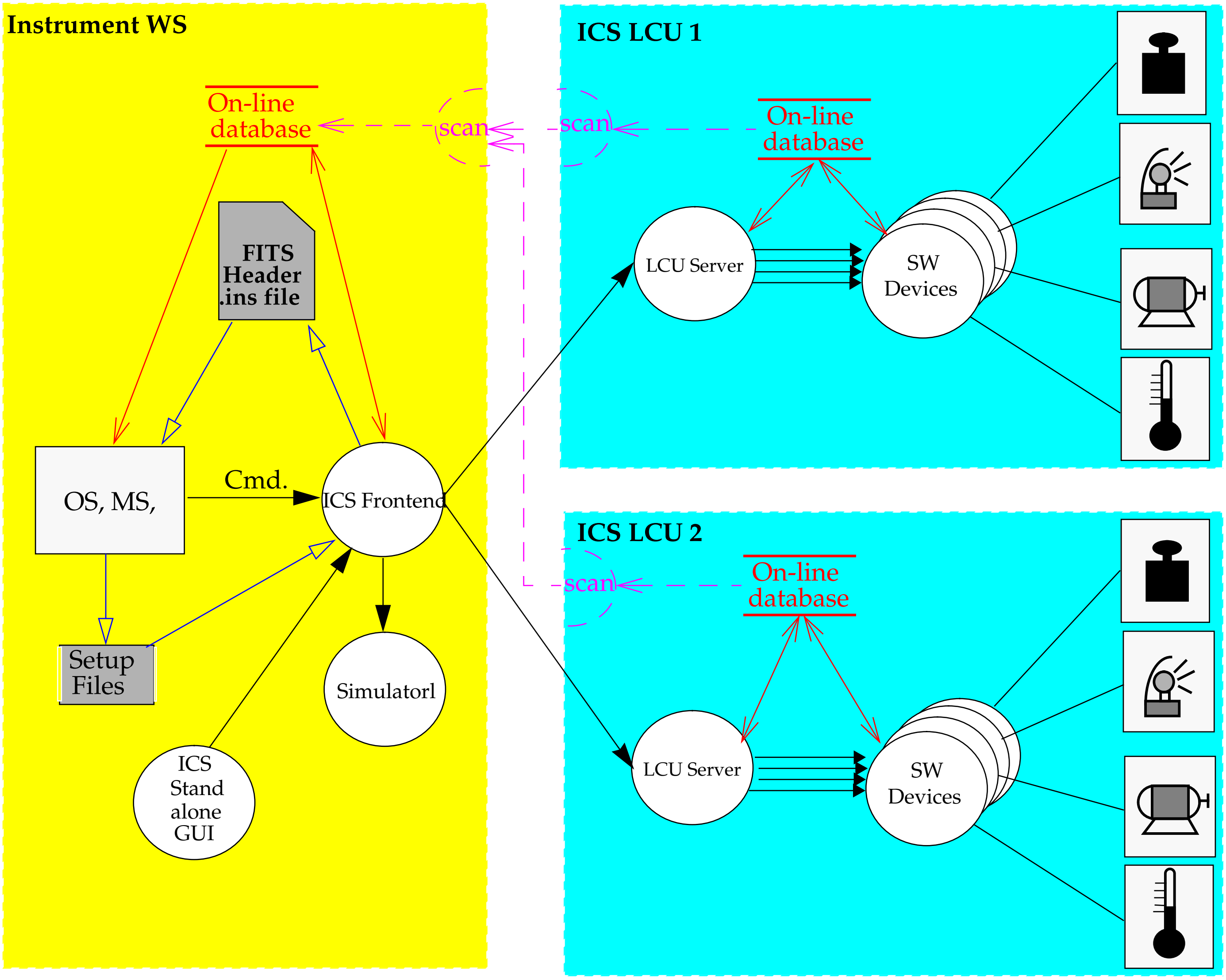
Build (from VLT sw archive to $INTROOT)

- .buildInsFromArchive
  - *cmmCopy* all UVES modules
- .buildInsKit
  - *make clean all man install* all UVES modules

uvinsInstall

- uvinsInstallInsRoot
  - Install files from $INTROOT to $INS_ROOT
- uvinsInstallEnvironments
  - Create UVES WS and LCU environments
  - Start environments
  - Configure scan links
- uvinsInstallCfg
  - install instrument configuration parameters from configuration files in $INS_ROOT into the UVES OLDB

ICS architecture



Same as Test Camera

ICS Remarks

- We **re-used** as much as we could of **existing sw**, including *ic0* (base ICS modules) and standard engineering tools (e.g. *motei*).

- We were two persons working on ICS. We placed the **border of responsibilities between WS and LCU**.

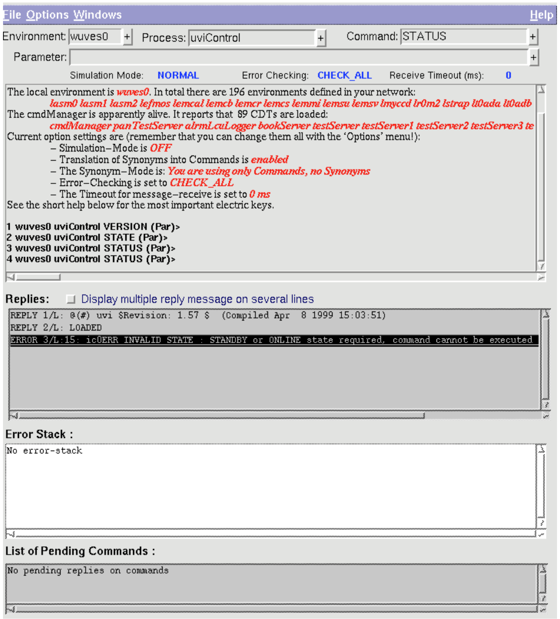
- We made the **WS-LCU interface as simple as possible** in that communication takes place between two processes only.

- To make OS life easier, we **improved significantly the simulation level** provided by *ic0*. To improve the simulation capabilities we used the LCU Server CDT for the Simulator as well.

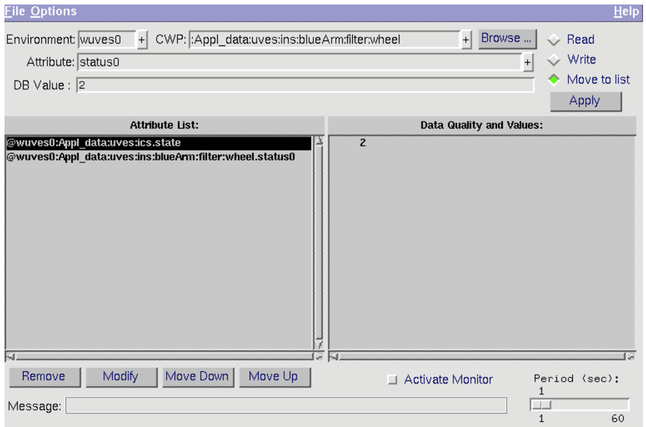
- We had very **little problems with the hw**, thanks to the **very wide standardisation**.

Engineering Tools used by ICS

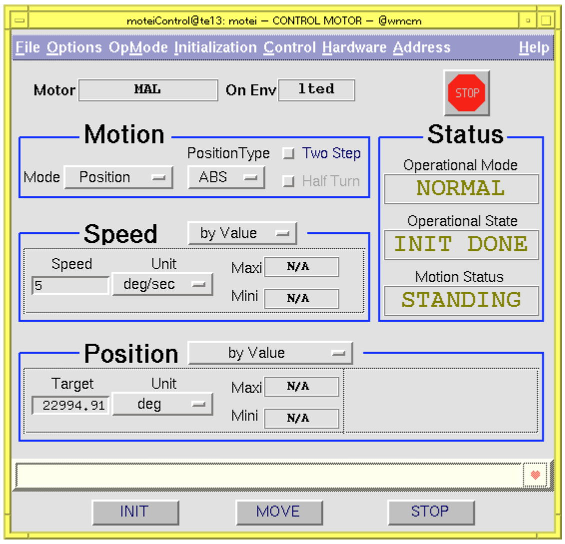
Messages



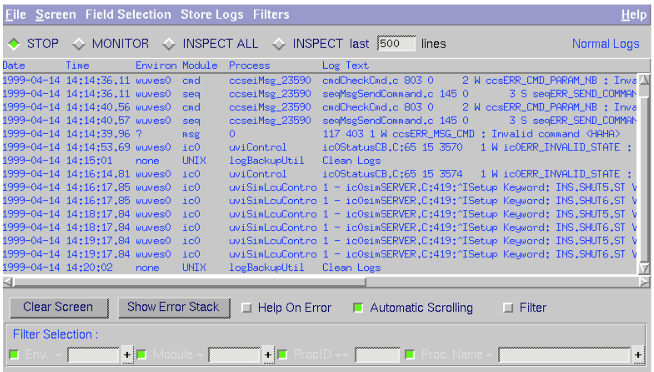
On-line database



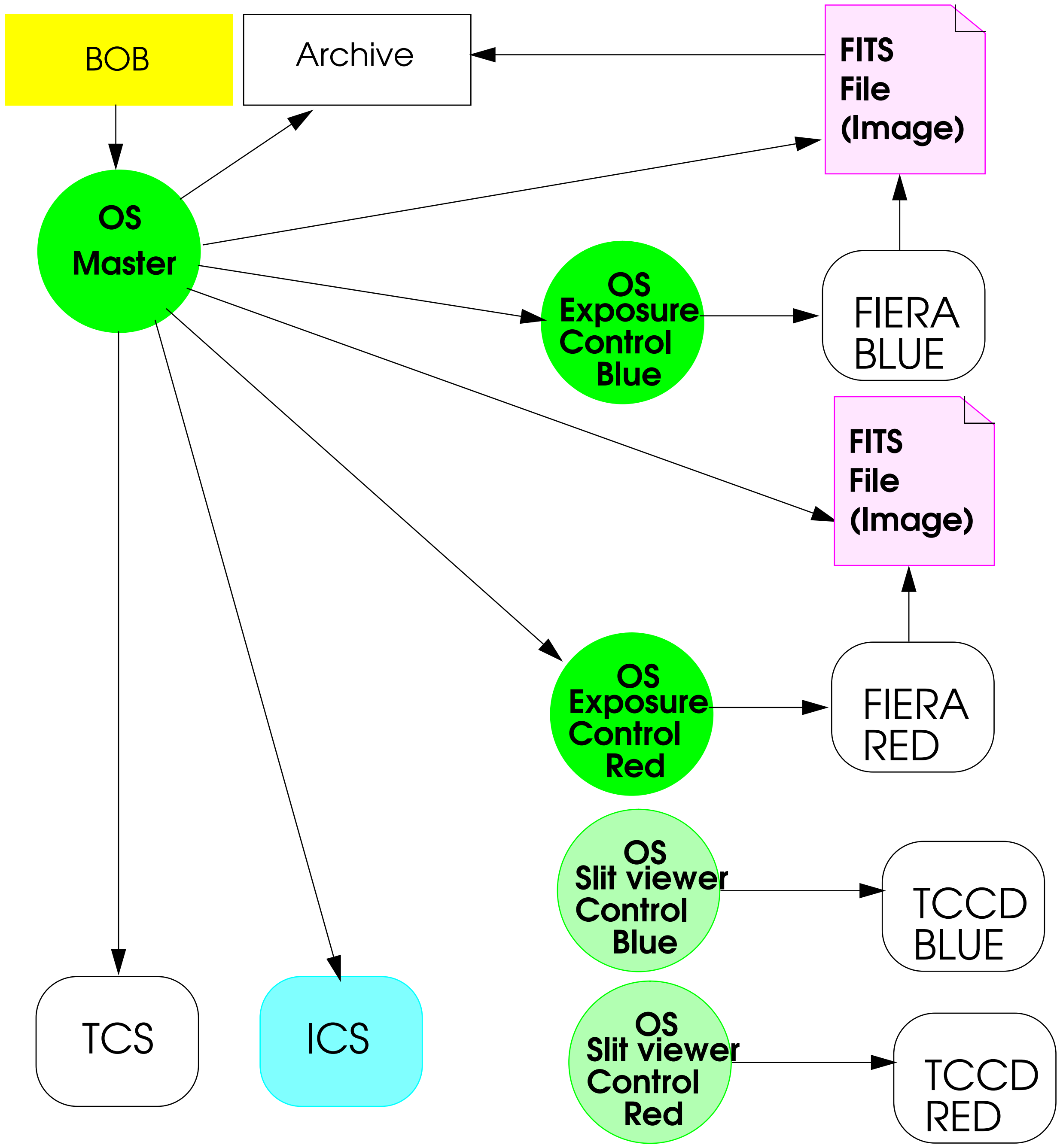
Motors



Errors & logs



OS Architecture

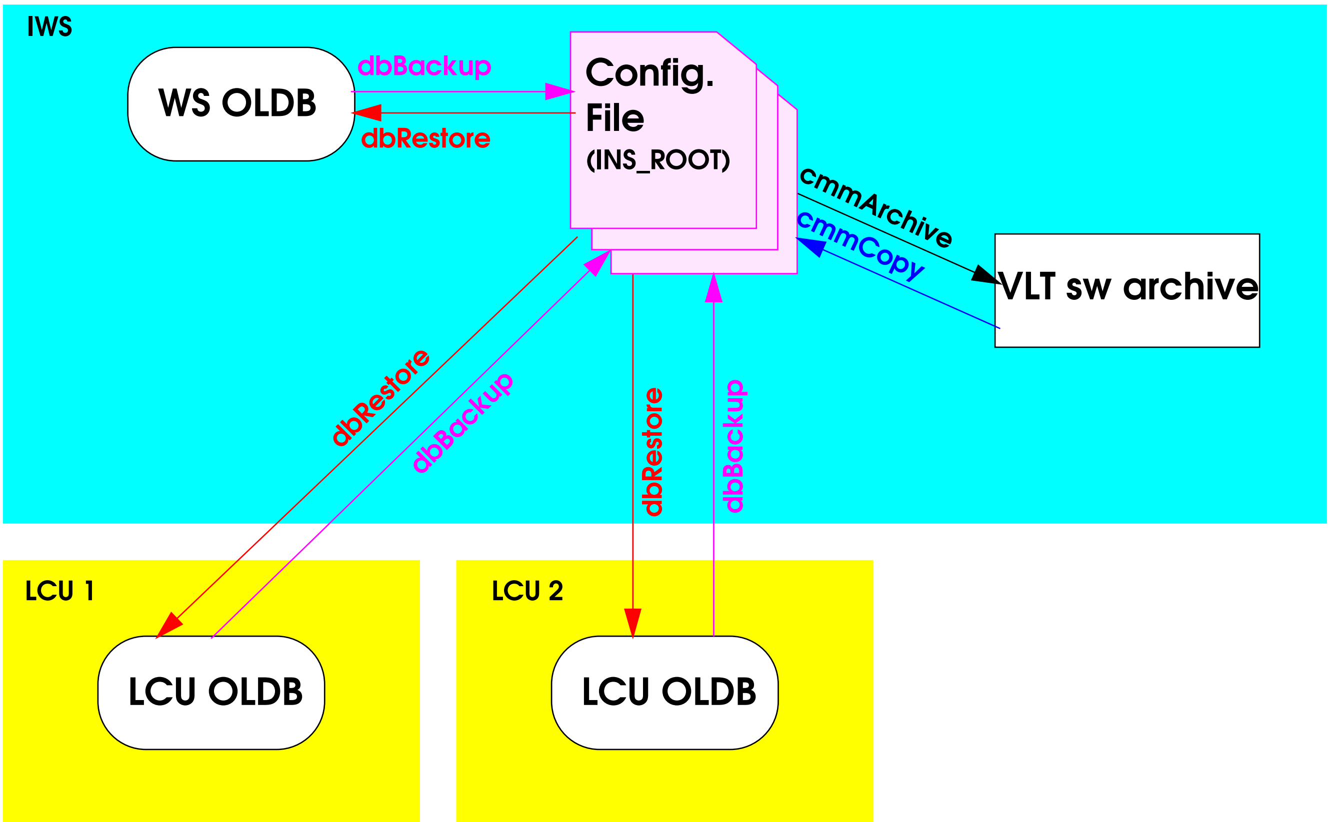


OS Remarks

- One single **front-end process** to BOB and Archive
- **One exposure control process** for each DCS (parallelism required)
- **One** single **process communicating with ICS**
- OS has to be more intelligent than one might think. **Tendency to underestimate development**

Do NOT forget MS !!

- **Essential for Instrument Configuration Control. Absolute need**, not a nice to have (ref. *uvins*).



- Important to **monitor Instrument Health**

- Important to make life of **Maintenance** and Operations Staff **at Paranal easier**

UVES demo

Solutions to different needs

- Single device test
  ---> motei, acromag inducer
- Sub-set of devices + CCD camera
  ---> ICS and DCS stand-alone GUIs
- Maintenance and verification procedures
  ---> MS
- Define setup, start exposure and get results
  ---> OS stand-alone GUIs
- Automatic execution of sequences
  ---> MOBS

  ---> BOB

All modes but BOB are meant to support development and integration only. They are not available to the observing astronomer at the telescope