Swope IMB Requirements and Architecture

Purpose

This document describes a possible data and controls architecture for the Swope instrument mounting base upgrade and its interfaces with the telescope and science instrument. Its purpose is to provide a baseline for discussion and to help identify important software interfaces.

Philosophy & Assumptions

1. The science instrument is the communications nexus. All action commands are issued by the science instrument server. The observer or automated observing program need only talk with the science instrument to observe.

2. The science instrument server accepts commands from a (possibly) remote client. Assumption 1 means that the remote client software can be simple because only one communications channel is needed and all commands can be described in one document, that of the science instrument. The downside is that it's possible for different science instruments to have different commands for shared common items (filter wheels, shutters, e.g.) but a bit of coordination will keep this under control if that turns out to be important.

3. Individual subsystems (e.g., the IMB) should minimize the knowledge required by higher-up systems by having sufficient intelligence to handle bad input, error situations, and hardware failures. For example, the science instrument should not need any knowledge of the telescope controller to command a motion:

   SLEWTO RA DEC [EPOCH [EQUINOX pmRA pmDEC]]

   This command would send the telescope to that current precessed, aberrated, nutated, and refracted position without the science instrument having to know about the telescope motor controller commands, pointing model, torque tube east/west, track rates, etc. A query to the telescope would report back the command status (e.g., below horizon, moving, dome fail, done).

4. A database is used for star catalogs and exposure logs but not (?) for instrument control.

Subsystem descriptions

Science instrument

Besides operating the science instrument, the science instrument server software issues all high-level motion commands, receives all sensor data, and is the interface between the observer (or automated observing program) and everything else. Note that the science instrument server need not have a science instrument. That is, separate servers for the TCS and IMB can be created if non-engineering level access for those is needed.

Here's an example of how an imager might take a picture:

- The science instrument (SI) is asked to observe TargetA for 10 seconds in filter 2U by a human or an automated observing program.
- SI gets TargetA coordinates and proper motions from the database server.
- SI commands the TCS to move the telescope to those coordinates.
- SI commands the TCS to set the telescope focus for filter 2U.
- SI commands the IMB to select filter 2U.
- SI commands the IMB to set guider focus for filter 2U.
- SI gets guide star and S-H star from database (canned mySQL script?)
- SI commands the IMB to position the guider and S-H cameras
- SI sets up the science instrument.
- SI polls the TCS status, waiting for a "move complete" statement.
- SI polls the IMB status, waiting for a "move complete" statement.
- SI sends a command to the IMB to open the shutter for 10 seconds.

During the exposure (and maybe all the time), the SI server polls the TCS and IMB for status every second or two so the observer can be alerted to unusual situations (motor failures, limits reached, etc). After the exposure, the SI updates the database with the exposure information.

The SI server is never interrupted by subsystems. Because the SI server is likely running on a office-style PC operating system (Linux, Mac, Windows), it's easy to miss asynchronous communications (daemons die, ports get blocked, etc.) and you end up polling anyway. Any problem requiring fast response must be handled by the responsible subsystem.

Remote client

The science instrument server talks to the outside world through a remote client interface. Having this interface opens up a lot of possibilities, including remote and automated observing, because instrument commanding can be done with common programming tools. This can take many forms:

- A Python script that runs an automated observing program (pointing models, RR Lyr light curves, supernova monitoring, etc.)
- A telnet (or other service) port from a terminal session. This is the ultimate engineering interface with full access to all functions. Probably not useful for the observer but it's an effective way to troubleshoot.
- A PHP script either on the science instrument computer, or perhaps local to the observer (if the latter, the science instrument needs to be smart enough to catch dangerous commands since the observer could change the prepackaged commands).
- A full-up client observer's GUI independent of the server. This could be on the science instrument (accessed remotely through VNC) or local to the observer.
- iPad client (but the PHP solution could work here anyway)

CCD

The science instrument includes the CCD, but since the detector package is often built as a separate subsystem, it's called out here in its own box.

IMB

The instrument mounting base controls:

- Two guider camera X-Y axes
- Two guider camera focus stages
- Two six-position filter wheels
- One shutter (but current systems run the shutter from the CCD controller)
- S-H lenslet/field viewer/LED selection
- Calibration lamps

This system accepts commands from and sends data to the science instrument.

Using an 8-axis Galil controller with built-in amps (something like DMC-4080-C012(5V,P1422)-I000(SSI)-I000(SSI)-D4040-D4040, $3795 or DMC-4183-BOX8-ABCD(SSI)-EFGH(HSRC,SSI)-D4040-D4040, $2695) and adding a power supply and connectors, the physical package should be small enough to fit in the blister box area (how big is the blister box area???).

TCS

The TCS accepts commands from a science instrument server and returns data to the science instrument. It:

- computes precession, aberration, nutation, and proper motion offsets before commanding the telescope mount
- computes the dome position and commands the dome control system
- computes UT0 from GPS clock data (or do we care?)
- starts and stops the guider CCD
- accepts and corrects guider error data from the guider CCD
- commands focus corrections from S-H analyzer data
- computes and commands track rates, including moving targets
- sets up the dome and telescope for white screen calibrations
- may present a status screen to the telescope operator

The TCS is responsible for telescope safety. That is, it is responsible for not pointing the telescope below the horizon, etc. In a “next object” report it might provide information about time-to-horizon limits, airmass, moon distance, etc.

WX

The TCS is responsible for checking if the weather is good enough to open. The WX box might be a link to the weather station (evaluating wind speed and direction, for example) but should also include a signal from an on-site human telescope operator (Magellan or du Pont).

Mount control

Telescope motion controller. This system receives instructions from the TCS and does the following:

- Moves the telescope (RA, DEC, Focus, Rotator)
- Reports the telescope status to the TCS

Dome control

Dome motion controller. This system handles:

- dome rotation
- dome slit open/closed
- windscreen positions
- status reporting to the TCS

Guider CCD

This system:

- analyzes guide star images and sends guider error data back to the TCS
- delivers S-H images to the S-H analyzer

Copied from Magellan.

S-H analyzer

This computer takes the S-H image from the guider and computes the focus error, sending that data back to the TCS. Copied from Magellan.

Database

The database contains information about reference stars for guiding and wavefront sensing as well as observing target lists and standard stars. The science instrument can use it to store exposure information.