Autonomous Observing Systems
The Making of Chimera

Isaac Richter, Paulo Henrique Silva, Antonio Kanaan, Denis Barrett, Timothy DeHass, Ryan Mann, Andrew Missert, Matthew Turner, and Shashi Kanbur

The Premise

What would it be like to have a telescope where all you have to do is send a list of targets and in a month, all of your data will have been captured, reduced, and perhaps even analyzed for you by an autonomous robotic telescope. What if you wouldn't have to spend any more nights sitting on a freezing mountain to get night-by-night observations of transient phenomena. What if you didn't even have to connive graduate students to do this for you? The goal of Chimera is to make this a reality.

The Methodology

Making that premise a reality is really much more a computer science issue than an astronomical one. Astronomers have been instinctively selecting targets from what they want to observe based upon time of night and various other conditions. Getting a computer to do this, and do it reliably, is more of a challenge.

One of the biggest complaints that Kanaan and Silva had about existing computerized telescope control programs were that they tended to be inflexible, and difficult if not impossible to extend with new functionality. They wanted an open-source system that anyone could extend with the specific functionality they might need. To accomplish these, goals, Chimera was written in Python, and is available under the GNU General Public License (GPL). It is built in an extensible, multi-threaded, distributed fashion. To add an additional control mechanism, one only needs to write the controller logic.

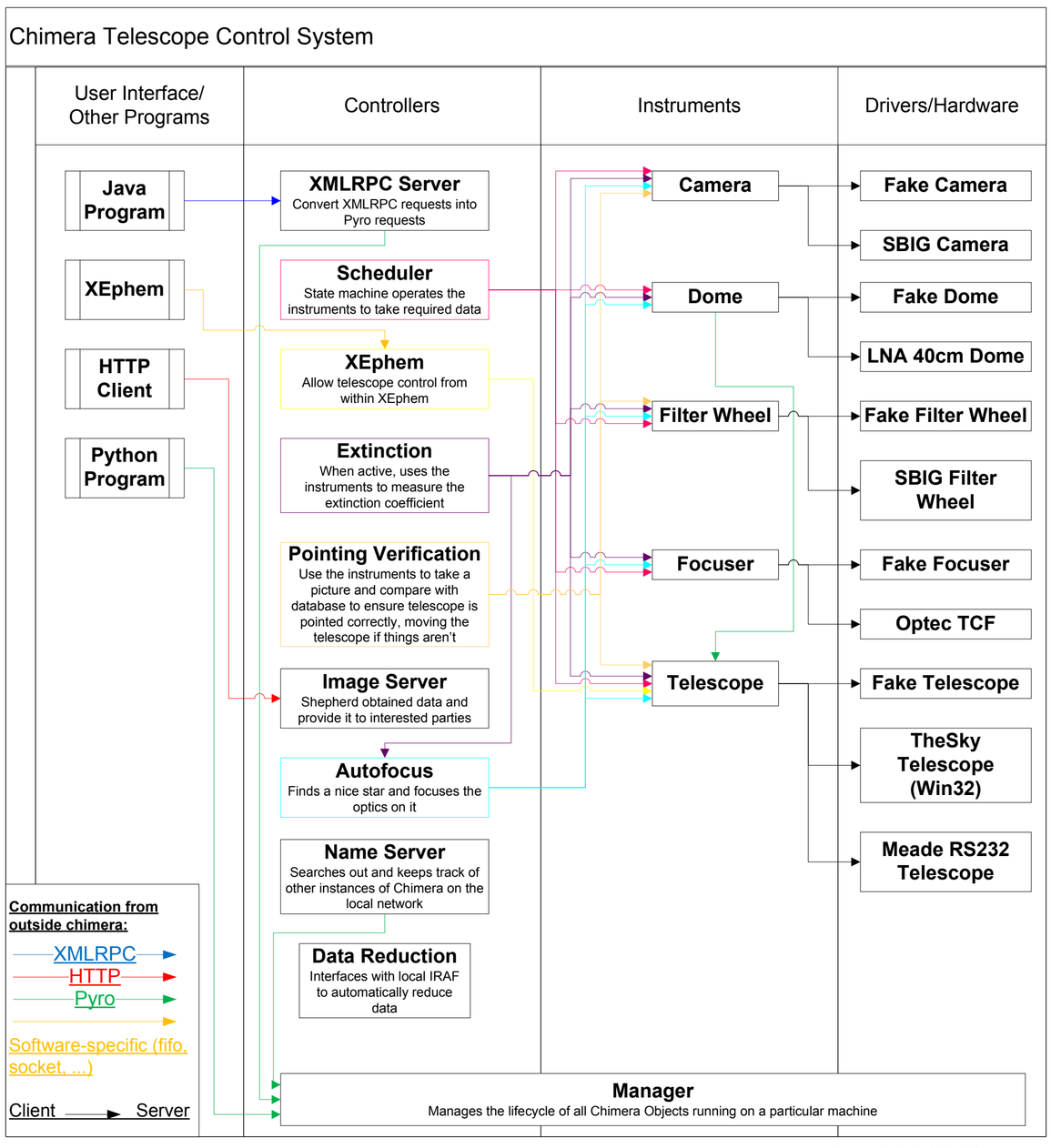
Structure of Chimera

Overall Structure

Every function of Chimera is handled by a "ChimeraObject". A ChimeraObject may be accessed remotely via the Pyro library. Every instance of Chimera is based around a manager. The manager is responsible for starting, stopping, and otherwise handling ChimeraObjects. The manager also helps ChimeraObjects to contact each other, even ChimeraObjects running under a different manager.

Observatory Control

Within the framework provided by Chimera, the ChimeraObjects used to control an observatory can be divided up into controllers, instruments, and drivers. (See graphic in lower-left corner.) Controllers are responsible for automated control of the instruments. They can monitor the instruments and other controllers, and send commands to the instruments as needed. The scheduler is an example of a controller.



Instruments

Each physical "thing" that Chimera interacts with is considered by Chimera to be an instrument. In order to make Chimera extensible, each actual piece of hardware is abstracted as a device-independent instrument. The instrument is then given a device-specific driver that handles the specifics of dealing with the hardware at any particular observatory This allows Chimera to place code that is generic to an entire class of hardware in an instrument file, leaving drivers to worry only about interfacing the driver API with the serial port or vendor specific library. For example, the dome instrument contains the code necessary to talk to a telescope and slew the dome to match the position of the telescope. This code is written only once — in the dome instrument. Each dome driver doesn't have to worry about this sort of thing.

This also allows one to create a virtual observatory for use by students, or testing the software, because each type of hardware supported by Chimera also comes with a fake driver. This driver emulates what average hardware of that type would do, allowing the entire system to be tested without wasting valuable observing time.

Controlling a Chimera

Manual Control

Chimera will be able to operate in either manual or automatic mode. Under manual control, the operator will be able to give instructions to the instruments as in a non-automated system. Chimera provides two "standard" methods to do this, and an interested party could create additional ones. The first interface created involves using command-line python scripts. These use Pryo to directly talk to the controller, instrument, or driver that the user wants to control. These scripts also allow the user to load and unload ChimeraObjects from the manager. A cross-platform Java GUI was also created that sends XMLRPC requests to a listener that is started by Chimera and converts that request into Pyro for sending on to the appropriate ChimeraObject.

Automatic Control

The ultimate objective of this system, of course, is to be able to have autonomous control of the telescope. This is accomplished through a controller known as the scheduler. When finished, the scheduler will work by looking through its database of requested observations, looking for which observation makes the most sense to do at the current time. Observations will have various constraints associated with them, and the scheduler will be responsible for making sure that the constraints are satisfied. The scheduler will also have to be able to make sure that the data it is getting is reasonable and that the equipment is generally working. To that end, other controllers will operate upon request of the scheduler to ensure that the telescope is pointing in the correct location and that the camera is properly focused. Data reduction routines will automatically be launched once sufficient data is collected that it makes sense to run them. While much more coding still needs to be done before Chimera will be completely automated, steps toward this goal are being made, and before long, automated observations may be considered the norm.

Learn More

More information about using chimera may be found on Chimera's Source Forge page: http://chimera.sourceforge.net/. Chimera is licensed under the GNU Public License (GPL), and the source is available at http://chimera.googlecode.com/. Pyro utilizes the remoting library Pyro, which can be found at http://pyro.sourceforge.net/.

The authors would like to acknowledge support of:

- NSF grant OISE 0755646
- HST grant HST-AR-10673.04-A
- The 2006 Chretien Award of the American Astronomical Society to SMK
- Laboratorio Nacional de Astrofisica, Brasil
- The State University of New York at Oswego
- Grupo de Astrofisica, Universidade Federal de Santa Catarina.
- CNPq — The National Council for Scientific and Technological Development [Brazil]