Science planning for the Gemini 8m telescopes

Steve Wampler, Kim Gillies, Phil Puxley, and Shane Walker

Gemini 8m Telescopes Project, 950 N. Cherry Ave, Tucson, AZ 85719

ABSTRACT

The new 8-meter class telescopes represent large investments by the development communities. This means that these telescopes must be operated efficiently to provide the best possible return on these investments and a great deal of effort has been made to provide control software that supports effective use of the telescopes. However, efficient use must be more than just keeping the telescopes operating; it is important that observers be provided tools that enable them work effectively.

The Gemini 8m Telescopes have developed a strategy for helping astronomers plan observations through the design of Science Programs. While there are a number of unique aspects to this strategy, this paper focuses on the methods used as the foundation for connecting astronomers to the facilities of the observatories during the design of Science Programs.

The methods under development take advantage of emerging Internet technologies to help reduce the maintenance issues normally associated with supporting remote sites, while freeing users from many of the performance problems associated with web-based solutions.

Keywords: telescope science planning, network-based solutions, Gemini software

1. INTRODUCTION

The new class of eight-meter telescopes are being developed at a time characterized by both expanding software technologies and shrinking astronomical budgets. Furthermore, new telescopes in this class, such as the Gemini telescopes, have been given performance requirements[1] that far exceed those for telescopes from earlier generations. The Gemini 8m Telescopes Project has approached the high-level software design with the goal of maximizing the effective use of the telescopes efficiently[2]. To full utilize the best seeing conditions on Mauna Kea (Hawaii) and Cerro Pachon (Chile), the observatory must encourage early planning of *Science Programs*[3] by astronomers. These Science Programs are critical to successful queue-based observing, where the observatory staff can arrange observations to best match observing conditions.

Successful support of early program development requires tools that are easy to use and readily available to off-site astronomers. Through this software, astronomers should be able to obtain information on the status of observatory facilities (instrument configurations, etc.), provided that the software is always kept-up-to-date. Previously, meeting these conditions meant significant support effort and often resulted in having to maintain multiple versions of the software, often for multiple software and hardware environments. These maintenance efforts consume more and more of an observatory's resources as time goes on.

The expansion of the World Wide Web has helped by allowing an observatory to keep the software on-site, with astronomers accessing and using this software through various forms-based web pages. Unfortunately, there is a price - interactive use of the web is dependent upon a variety of factors, most of which are beyond the direct control of either the astronomer or the observatory. Few things lower opinions of software faster than poor response. Also, the use of HTML documents and CGI scripts imposes limitations on the interface that can be presented to the astronomer.

The Observatory Control System Group of the Gemini 8m Telescopes Project is responsible for developing the high-level software for science planning. This report describes one important tool, called the *Observing Tool*, that is a key component of science planning at Gemini. The focus of this tool is to provide an easy-to-use, efficient environment for off-site development of Science Programs. The implementation of the Observing Tool takes advantage of new Web-based technology to overcome the limitations mentioned above.

The Gemini 8m Telescopes Project is managed by the Association for Research in Astronomy, for the National Science Foundation, under an international partnership agreement.

2. QUEUE OBSERVING WITH THE GEMINI TELESCOPES

A pool of observations requiring different observing conditions allows the on-site observing staff to select observations that best match current conditions. This is only possible if the observations are prepared well in advance. The planning process for the Gemini telescopes includes a number of well-defined steps.

During Phase I proposal preparation, astronomers produce an outline of their intended Science Program as part of the application process[4]. If the proposal is accepted, Gemini constructs a Science Program template from this outline and notifies the astronomer.

The astronomer retrieves, completes, and resubmits the Science Program. This program describes the process for performing the observations to such an extent that it may be accurately conducted by support staff. Submitted programs are stored in the observatory's database and scheduled to execute in the coming semester.

Each of these steps is assisted through the use of software tools. Phase I proposals can be developed using the Gemini Phase I Tool, although the Gemini partner countries are free to develop their own software for proposal submission.[‡] Once the proposal has been accepted, the Observing Tool is used to edit the generated Science Program template.

3. OBSERVING TOOL GOALS

The Gemini Observing Tool is designed to meet the following goals:

- simplify development of sophisticated programs, allowing observers to take full advantage of the features of the Gemini telescopes without undue complexity
- support off-site program development, allowing and encouraging astronomers to do careful planning of observations
- support off-line program development, freeing the use of the tool from the impact of Internet access and performance
- while planning is the prime focus of program development, the Observing Tool should also be used on-site, both for interactive observing and to allow staff observers to develop night plans from a set of Science Programs
- keep support and maintenance costs low.

4. DEVELOPMENT TECHNOLOGIES

One of the early design decisions in the development of the Observing Tool was to adopt modern software technologies. This has to be done carefully to avoid getting trapped in a 'fad-of-the-day' software dead end. However, there are definite advantages to choosing well-designed, modern tools. Language features for modern programming languages improve productivity and maintainability. Improvements in network connectivity are also best exploited through modern software technologies.

Another early design decision was to target the Observing Tool to run on the user's machine. This approach is the opposite of early efforts to use the Internet to support software. Traditionally, providing tools to run off-site requires substantial development efforts to produce software for a variety of hardware and operating system mixes and increased maintenance efforts to keep these multiple versions running. Less obvious problems include keeping remote sites updated on changes at the observatory and making sure users have access to the most recent release of the software. Nevertheless, if these problems can be addressed, remote execution of the software offers a number of advantages: network delays and outages are less of a concern, the load on observatory machines and networks decreases, and astronomers can more easily take advantage of their own resources.

At the Gemini site, Science Programs built by the Observing Tool must be maintained and manipulated. An object-oriented database was determined to be the best way to hold Science Programs. The data model provided by an object-oriented database fits well with the other design decisions, including the choice of language for the Observing Tool. After considering several alternatives, the ObjectStore database was selected for use in Gemini software.

The Java programming language was adopted as the vehicle for implementing the Observing Tool because Java offers the best hope for machine and operating system independence. Programs written in Java are easily transported across the Internet and there is good support for graphical user interface development in Java. Java's well thought out object-oriented programming support is ideally suited to both the design and implementation of graphical user interfaces and

[‡] They must nevertheless follow the Gemini interface requirements for submission content and format.

for the manipulation of the components in a Science Program. Multithreading and modern network programming features are fully incorporated into the Java language, greatly simplifying important tasks such as Guide Star Catalog searches. Also, Java development environments support rapid prototyping. This ability to quickly produce and adapt software during the design phase is especially important in software where user interface issues abound.

To deliver the Observing Tool, the Gemini Project has adopted Marimba's World-Wide Web software broadcast technology (Castanet). Remote sites install Marimba's Tuner software and then subscribe to the Observing Tool channel by attaching to the Gemini transmitter. The *push technology* provided by the transmitter/tuner channel allows astronomers to download and install the Observing Tool. Installation is automatic - there are no tar files to unload, no Makefiles to modify, or other software to obtain and install. In addition, the Observing Tool can be updated automatically through the channel subscription. Alternatively, the astronomer can choose when to update the Observing Tool. Updates are installed as they are received through the channel with no need for intervention by the astronomer. Furthermore, *only the parts of the software that have changed* are transmitted and installed.

Once the Observing Tool is installed, it can be run without access to the Web. This makes it possible for astronomers to develop Science Programs using laptops while traveling, for example. Once completed, the Science Program can be transmitted to Gemini when a network link is re-established.

Remote communication between the Gemini sites and Observing Tools running worldwide is handled via Sun's *Joe* package. Joe provides Java client access to the Gemini Observatory Control System's software using CORBA/IIOP. For example, Joe is used when a Science Program is submitted to the site. A CORBA solution was chosen over other options because it

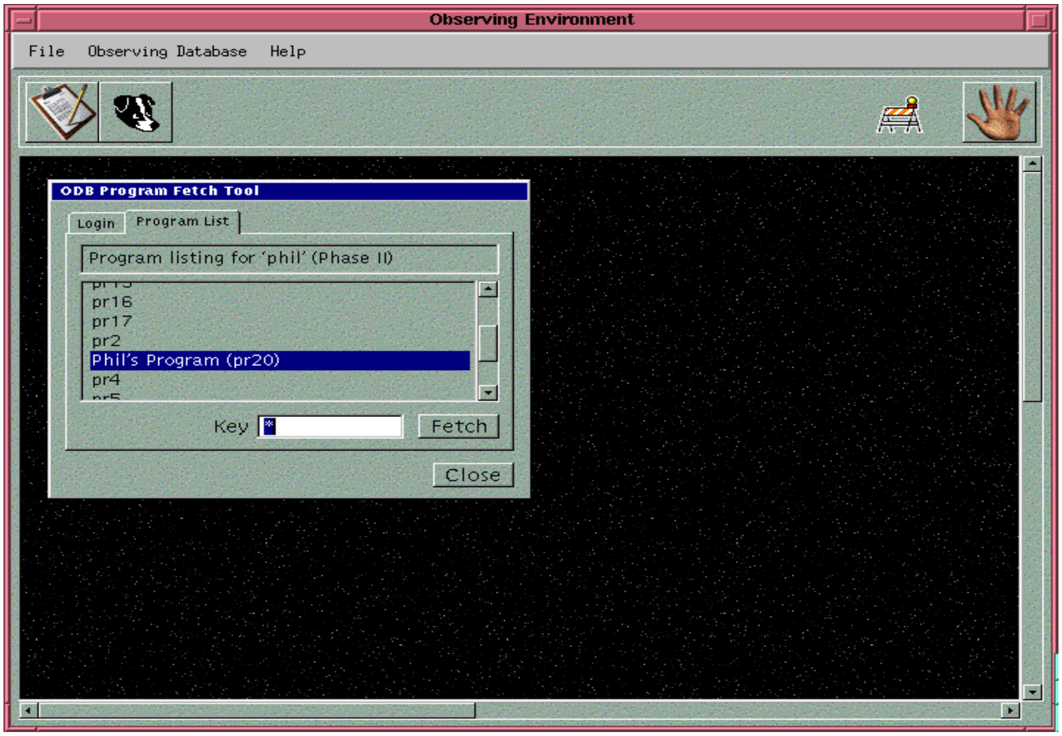
- meshes well with existing CORBA-based Observatory Control software,
- permits a rich communication architecture (including, for example, ‘‘callbacks’’),
- does not restrict us to Java-based servers (unlike RMI), and
- presents a very simple method-call interface (unlike raw sockets).

5. TYPICAL USE OF THE OBSERVING TOOL

Figure 1 shows the Observing Tool after launching it from Marimba's Tuner. A complete environment for working with Science programs is presented to the astronomer. In figure 1, the astronomer has logged into the Gemini system and is browsing the list of programs that are available to that account in the *Observing Database*. Access to the list of programs is protected by a login ID and password. Each program in the database is assigned a unique key. This allows astronomers to grant access to specific programs for collaboration without exposing all of their work to outside access.

Once notified that his ‘‘Phase 1’’ proposal has been accepted, the ‘Fetch Tool’’ depicted in the figure is used by the astronomer to locate and retrieve the initial program template that has been constructed for him. The Science Program is then edited to fill in additional detail. During the editing process, programs can be stored locally on disk and/or resubmitted to Gemini multiple times for safe-keeping.

A distinction is drawn between programs in the ‘‘editing’’ (or Phase 1) database, and those that are actively being used to schedule and run the telescope. During the editing process, observers only have access to the Phase 1 database. At some point before the start of the semester, the deadline for Science Program submission arrives and complete programs that have been submitted are moved into the active Observing Database for scheduling. Once in the active database, further editing requires explicit permission and support of observatory staff. This separation and exclusion is essential because the ‘‘active’’ database is being used to create schedules and run the observatory.



Program listing for 'phil' (Phase II)

pr16
pr17
pr2
Phil's Program (pr20)
pr4

Key *

Figure 1 - Browsing the Gemini Science Program database

After a Science Program has been selected and fetched (or loaded from disk), it is displayed in an Observing Tool window as shown in figure 2.

More than one program may be simultaneously edited in separate Observing Tool windows. The standard drag-and-drop/cut-and-paste mechanisms are available to move items between programs. Because multiple programs can be open simultaneously, the ability to copy and paste between programs makes it possible for astronomers to maintain libraries of common operations, quickly develop new versions of existing programs, etc.

A Science Program is a hierarchical collection of objects. Basic levels in the hierarchy are the Program, groups of Observations called Folders, Observations, Components, and Iterators. The Program level collects a potentially large set of observations all related to a single science goal. Programs should map roughly one-to-one with proposals. Observations collect information for one or more science frames associated with a given base position in the sky. Observations range from a simple stare at a given RA/Dec to complex mosaic patterns, etc. Observations consist of various Components and an (optional) Iterator. The Components describe the static setup and scheduling constraints of an observation, for instance, the telescope slew target, guide star selection, initial instrument configuration, etc. Iterators describe how to operate on the static Components in the Observation to produce Science Data. Example Iterators include offset patterns for dithers and mosaics and cycles of instrument filters. Iterators allow sequences of repetitive tasks to be expressed concisely, even if some properties (e.g., offset position or filter selection) vary from exposure to exposure.

Observation Folders group related Observations conveniently and facilitate sharing of Components. Here the hierarchical nature of Science Programs is utilized. Components may be placed at higher levels of the hierarchy than just the Observation, including both Folders and the Program itself. Components inserted at a higher level (i.e., a broader "scope") apply to all lower levels by default, may be redefined at any point. This approach lets astronomers avoid duplicating information that is needed at several places within the program while allowing specific customizations as needed.

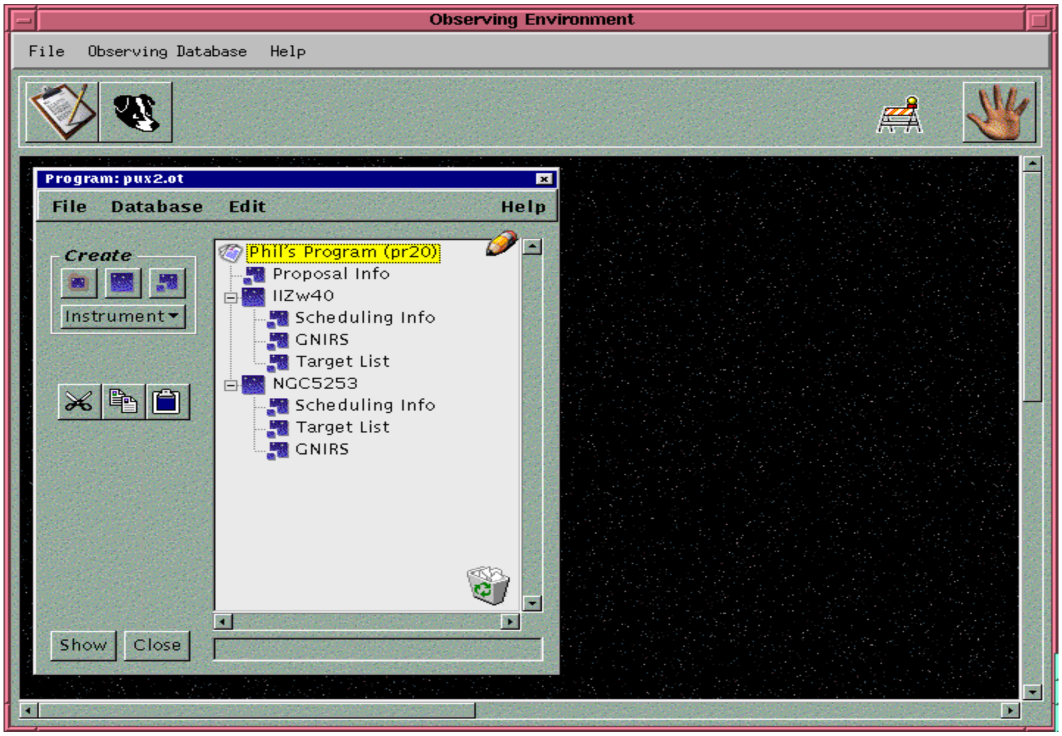


Figure 2 - A simple Science Program

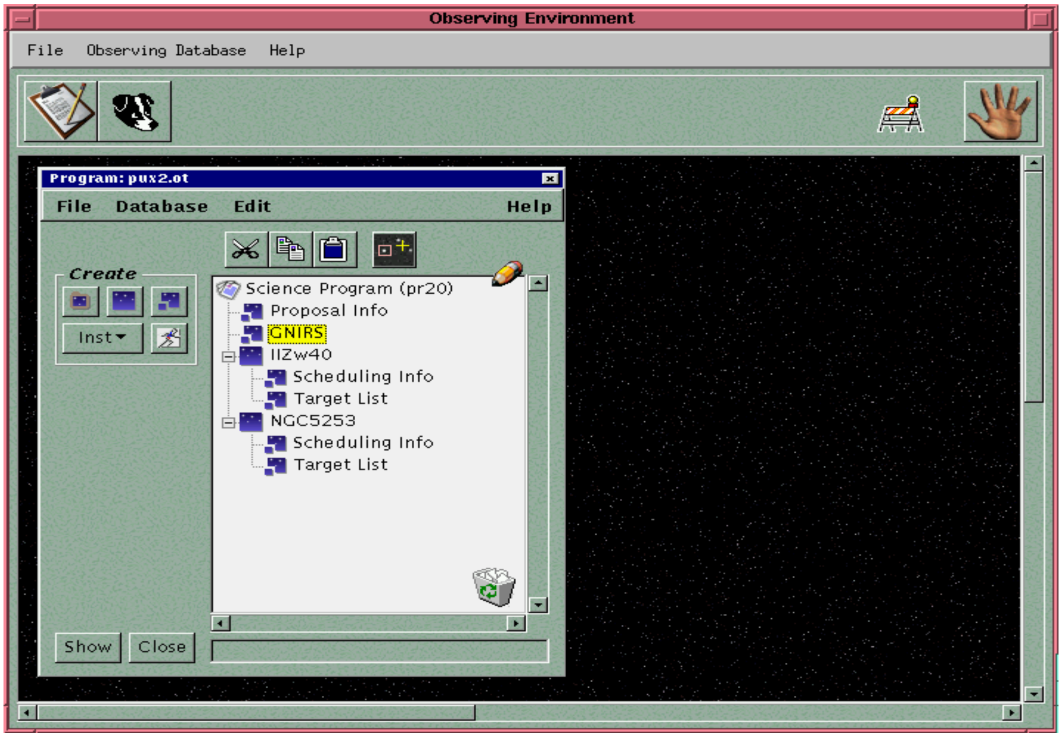
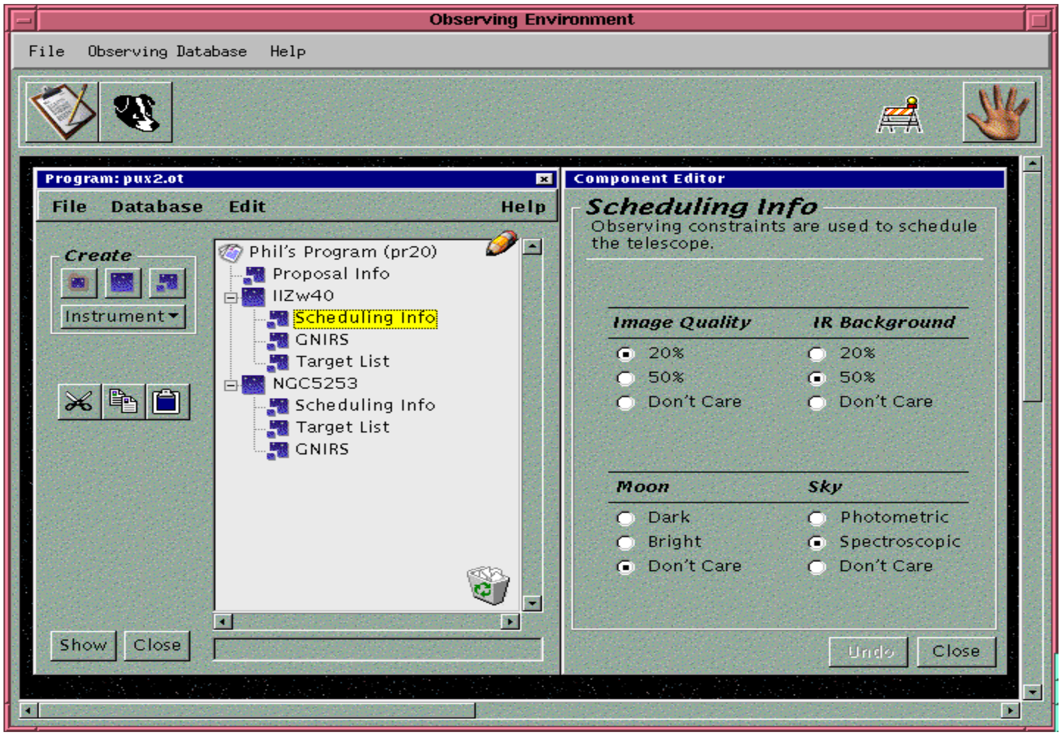


Figure 3 - Sharing a common component between observations

In figure 2, the selected Science Program currently contains two observations (IIZw40 and NGC5253), each composed of a set of components describing details of the observation. No folders are used in this program. Though both observations use the Gemini Near-Infrared Spectrograph (GNIRS), the configuration of the instrument is different in each observation. For this reason, the instrument configuration (labeled GNIRS in the program) has been placed in each observation. If the initial instrument configuration were identical for both observations then the astronomer can place the configuration above both observations, as shown in figure 3. In this case any change to the GNIRS component affects both observations.

Up to this point, we have only concerned ourselves with the program hierarchy. Objects within a program can be opened for editing by double-clicking on them. This opens up an *edit window* for the selected object. Figure 4 shows the Observing Tool with the edit window for the scheduling information component open. This component places constraints on the scheduling software. In this case, the Observation cannot be scheduled unless the Image Quality is in the upper 20th percentile, IR Background is in the upper 50th percentile, and the sky conditions are spectroscopic.



Phil's Program (pr20)
- Proposal Info
- IIZw40
  - Scheduling Info
  - GNIRS
  - Target List
- NGC5253
  - Scheduling Info
  - Target List
  - GNIRS

Scheduling Info

Observing constraints are used to schedule the telescope.

Image Quality	IR Background
● 20%	○ 20%
○ 50%	● 50%
○ Don't Care	○ Don't Care

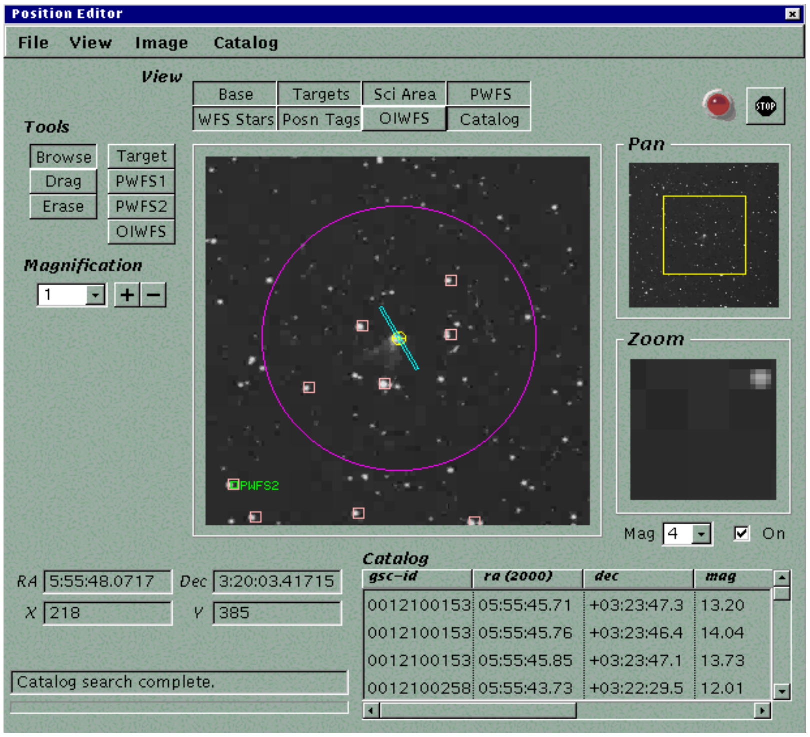
Moon	Sky
○ Dark	○ Photometric
○ Bright	● Spectroscopic
● Don't Care	○ Don't Care

Figure 4 - Component editor for scheduling information

Another editor is the *Telescope Target* editor. The astronomer can use this window to enter any target information that is needed: base position on the sky, position for wavefront sensors probes, chop positions, additional sky positions, etc. Once the base position has been entered using the Telescope Target editor, most astronomers are likely to choose to enter the remaining information through the more graphically oriented *position editor*.

Using the position editor, the astronomer may download an image of the region around the sky position from one of several servers, load a previously taken Gemini image, view guide star catalog positions, examine and move target and probe positions, alter the rotator position angle, etc.

Figure 5 shows the position editor as seen when using the Near-Infrared Spectrograph. The inner edge of the peripheral wavefront sensor annulus is visible, as are possible guide stars and the spectrograph slit. One of the guide stars in the lower left corner has been assigned to wavefront sensor probe 2. The other probe position is not visible at the current image magnification level. (In addition to the zoom and pan windows shown, the image itself can be scaled. For example, magnifying the image allows more precise RA/Dec positions to be obtained from mouse positions.)



Magnification 1

Mag 4 ☑ On

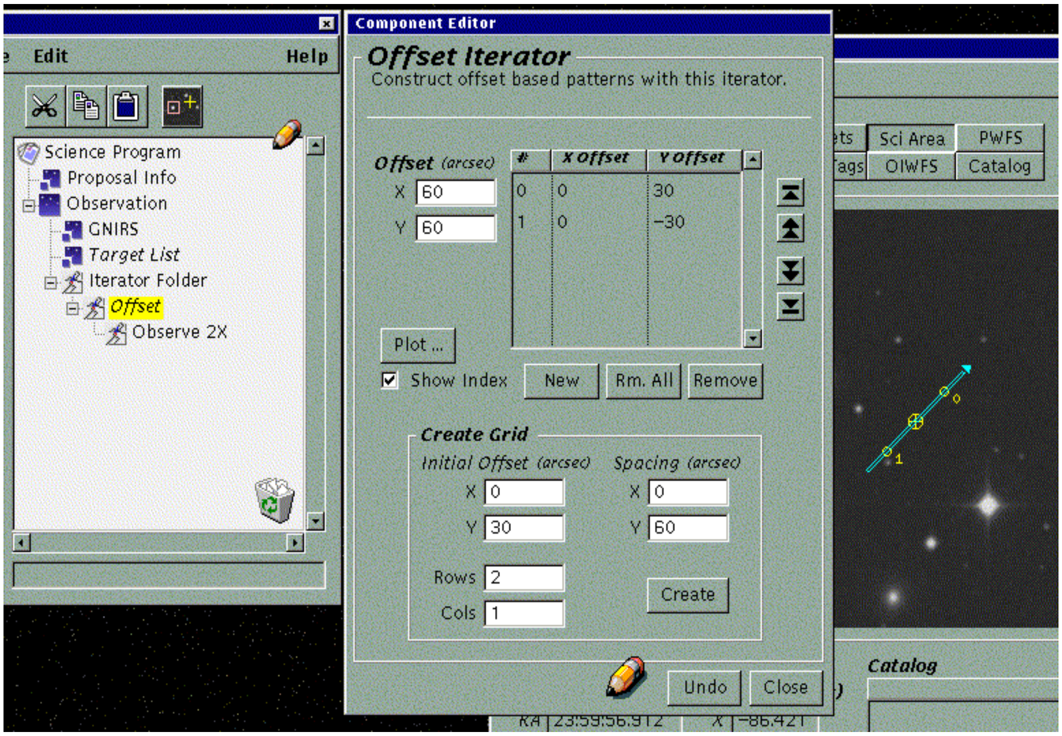
RA 5:55:48.0717 Dec 3:20:03.41715

X 218 Y 385

Catalog search complete.

gsc-id	ra (2000)	dec	mag
0012100153	05:55:45.71	+03:23:47.3	13.20
0012100153	05:55:45.76	+03:23:46.4	14.04
0012100153	05:55:45.85	+03:23:47.1	13.73
0012100258	05:55:43.73	+03:22:29.5	12.01

Figure 5 - The Position Editor



- Science Program
  - Proposal Info
  - Observation
    - GNIRS
    - *Target List*
    - Iterator Folder
      - *Offset*
        - Observe 2X

Offset Iterator

Construct offset based patterns with this iterator.

Offset (arcsec)

X 60

Y 60

#	X Offset	Y Offset
0	0	30
1	0	-30

Plot ...

☑ Show Index New Rm. All Remove

Create Grid

Initial Offset (arcsec)		*Spacing (arcsec)*	
X	0	X	0
Y	30	Y	60

Rows 2

Cols 1

Create

Undo Close

RA 23:59:56.912 X -86.421

Figure 6 - The Offset Iterator

Figure 6 shows the Offset Iterator Component Editor, as well as its graphical view on the Position Editor. Offset positions are specified in arcsec relative to an unrotated position angle. As the position angle is rotated (either on the Position Editor or through the GNIRS instrument component), the display of offset positions rotates accordingly. Note that Grid based patterns are easily constructed through the Component Editor.

In the program hierarchy view, an "Observe" Iterator has been nested inside the Offset Iterator and has been configured to take two exposures. When this observation is executed, the telescope slews to the base position, the instrument and telescope are configured accordingly, and then two exposures are taken at each offset position consecutively.

Finally, after an edited program has been submitted, scheduled, and executed, the observer is notified and may use the Observing Tool to retrieve his data. A Data Folder is added to a completed Observation by Gemini. It contains URLs that may be used to fetch the actual science data.

6. CURRENT STATUS

The Observing Tool is currently running in beta test mode. A number of component editors remain to be written, but are waiting on detailed design information from instrument development groups. Nevertheless, all of the basic functionality of the graphical user interface/observing database interaction is working, and a primitive component editor is available for use with components that still need better component editors written.

A feature of Marimba's Tuner currently prevents the launching of the Observing Tool directly from Tuner. Although it is possible to acquire and update the Observing Tool through the Tuner, it must be launched from the command line at the current time. This restriction is expected go away with a future release of Tuner.

At some point the Observing Tool needs to be integrated with the Gemini system software used to actually execute observations on the system. The Observing Tool has a role to play during the actual acquisition of the science data. This includes both interactive observing use, remote monitoring, and nightly plan creation. These topics have not been detailed in this paper but are important features of the Observing Tool that are under development. Though the Observing Database is in place, other observing support software the Observing Tool must communicate with is still under development as well.

7. ACKNOWLEDGEMENTS

The Observing Tool and Science Program design are primarily the work of Shane Walker and Kim Gillies. Shane Walker is responsible for the implementation while Kim Gillies created the Observing Database and conceived of most of the Observing Tool's features. Phil Puxley guided the Observing Tool requirements and provided much valuable testing and input. Many others have contributed to other parts of the control system design.

The position editor was inspired by the SkyCat program developed by the VLT project at the European Southern Observatory.

The Gemini 8-M Telescopes Project is managed by the Association of Universities for Research in Astronomy, for the National Science Foundation and the Gemini Board, under an international partnership agreement.

8. REFERENCES

1. S. Wampler, et al, ‘‘Software Design Description’’, *Controls Group, Gemini 8m Telescopes Project*, 1994.

2. S. Wampler, ‘‘Goals and Requirements for Software and Controls’’, *Controls Group, Gemini 8m Telescopes Project*, August, 1993.

3. S. Walker and K. Gillies, ‘‘Observing Tool Design Foundations’’, *Controls Group, Gemini 8m Telescopes Project*, 1997.

4. P. Puxley, ‘‘Execution of Queue-scheduled Observations with the Gemini 8m Telescopes’’, these proceedings.