Astronomical Data Analysis Software and Systems VI
ASP Conference Series, Vol. 125, 1997
Gareth Hunt and H. E. Payne, eds.

The SAOtng Programming Interface

E. Mandel

Smithsonian Astrophysical Observatory, Cambridge, MA 02138

Abstract. The architecture of the *SAOtng* image display program is centered on supporting communication with external programs and processes. Easy to use C, FORTRAN, and UNIX interfaces provide direct access to *SAOtng* information, data, and control functions. Through these interfaces, developers can utilize such advanced *SAOtng* features as direct *FITS* transfer, shared memory *FITS*, WCS-enabled positioning, and region of interest demarcation. In this paper, we describe the *SAOtng* programming interface and show, by example, how easy it is to integrate *SAOtng* into astronomical analysis systems.

1. Introduction

SAOtng (SAOimage: The Next Generation) is an updated and improved version of the popular *SAOimage* display program for X Windows. It utilizes the Xt Toolkit and Xt-based widgets, the Gterm-image widget, and the NOAO widget server as the basis for its graphical and imaging functionality (Mandel & Tody 1995). *SAOtng* supports direct display of *IRAF* images and *FITS* images (and easily can support other file formats), multiple frame buffers, region/cursor manipulation, add-on scaling algorithms, many colormaps, and communication with external analysis tasks. It is highly configurable and extensible to meet the evolving needs of the astronomical community.

2. The SAOtng Public Access Interface

SAOtng uses the X Public Access mechanism (*XPA*) to give external processes access to its data and algorithms (Mandel, Swick, & Tody 1995). *XPA* allows an Xt program such as *SAOtng* to define named "public access points" through which data and commands can be exchanged with other programs. Each public access point in *SAOtng* supports the exchange of arbitrary amounts of information through the call-back paradigm already familiar to Xt application programmers.

XPA extends the *SAOtng* graphical user interface beyond the customary boundaries of the program. Usually, actions in a graphical program are initiated by mouse and keyboard events input by the user. But in *SAOtng*, these actions can be initiated by external programs using the public access programming interface. In addition, new actions that are not part of the GUI can be added to this interface.

The public access interface to *SAOtng* supports UNIX, C, and FORTRAN calls. For example, the UNIX public access commands are:

```
    # does SAOtng exist? (returns ''yes'' or ''no'')
    csh) xpaaccess SAOtng

    # send data or commands to SAOtng
    # access points can accept modifying parameters
    csh) <data> | xpaset SAOtng <access_point> [parameters]
or  csh) imset <access_point> [parameters]

    # retrieve data or info from SAOtng
    csh) xpaget SAOtng <access_point> [parameters]
or  csh) imget <access_point> [parameters]
```

These programs can be used interactively on the command-line or in batch scripts. They also can be used directly in application programs by means of the *system()* routine. The latter mechanism has the drawback that the *system()* routine must start a new shell process each time it is called. In cases where efficiency is at a premium, external programs can use the C or FORTRAN public access interface to communicate with *SAOtng*.

The C and FORTRAN interfaces define a set of calls similar to the UNIX commands described above. For example, the C interface contains the following routines:

```
/* open a connection to the xpa server */
void *xpafd = OpenXPA(''SAOtng'');

/* send data or commands to SAOtng */
SetXPAValue(void *xpafd, char *paramlist, char *data, int len);

/* retrieve data or info from SAOtng */
GetXPAValue(void *xpafd, char *paramlist, char *data, int *len);

/* close connection */
CloseXPA(void *xpafd);
```

When *OpenXPA()* is called, an *XPA* server program is started to mediate communication between the calling program and *SAOtng*. This server program remains active until a call is made to *CloseXPA()*. It significantly improves the communication speed between *SAOtng* and external programs over the repeated use of *system()* to call to xpaset and xpaget. Note also that the use of this intermediary server program obviates the need for applications to link the X11 libraries directly.

More than thirty-five public access points in *SAOtng* (a selection of which is shown in Table 1) support communication with external processes. Several of these are designed specially to provide commonly-used information and data to astronomical analysis programs. It is this combination of broad flexibility and specific useful functionality that makes the public access interface a valuable part of *SAOtng*'s image display services.

3. Examples of the SAOtng Public Access Interface

New image files can be loaded using the *file* public access point:

```
csh) imset file ../data/coma.fits
```

Table 1. A Selection of Public Access Points in SAOtng

Name	Description
analysis	run analysis command on current image
blocking	set/get blocking info for extracting image data
colormap	set/get colormap for current frame
coords	get coordinate values
file	display image file in current frame
frame	create or change display frame
gif	create gif from current frame
print	set execute print command in current frame
redisplay	redisplay selected image in current frame
regions	get/set regions markers in current frame
rescale	rescale selected image in current frame
shm	map FITS image in shared memory
tcl	execute Tcl code
zoom	set/get zoom factor for current frame

Sending an image file name to the *file* access point causes *SAOtng* to load and display that image. If the image file is in *FITS* format, the data can be loaded directly. Otherwise, *SAOtng* calls the appropriate external image access program to extract a section of the image and send it back to *SAOtng* for display.

The *file* access point can contacted from the command line or from programs such as archive servers and file browsers to display local files. For example, the *XDir* file browser issues a *file* command to *SAOtng* when the user double-clicks on an image file. UNIX scripts also can use *file* to process a list of images.

For processes requiring the fastest possible display of image data, the *shm* (shared memory) access point can be used to share in-memory *FITS* with *SAOtng*. For example, a data acquisition program can collect *FITS* data into a shared memory segment. The program sends *shm* information to *SAOtng* so that the latter can access the same shared memory:

```
sprintf(paramlist, ``shm %s %d %d'', name, shm_id, shm_segsz);
SetXPAValue(xpafd, paramlist, NULL, 0);
```

The *shm* message instructs *SAOtng* to display the contents of the shared memory as a *FITS* file. Then, as data are collected over time, the *redisplay* public access point can be used to update the evolving image.

SAOtng supports the ability to mark regions of interest on images. Geometric markers (circles, rectangles, ellipses, polygons, points, and lines) and text can be placed interactively on an image and then moved, resized, reshaped, etc. Markers can have attributes of "include" or "exclude" and "source" or "background." They are used in systems such as *IRAF/PROS* and *HEASARC* xselect to specify spatial filtering of image data (Mandel et al. 1993).

The *regions* public access point supports external control of these markers. Using this access point, markers can be loaded from a file:

```
csh) cat regions.file | xpaset SAOtng regions
```

Once the shape, size, position, color, etc., of these markers are finalized, they can be saved for later use:

```
csh) xpaget SAOtng regions degrees > newregions.file
```

As indicated above, the coordinates used in the region descriptors can be loaded or saved in hms, degrees, or pixel format.

Finally, the *coords* public access point allows a user to retrieve the spatial position (and associated data value) pointed to by the mouse:

```
GetXPAValue(xpafd, ``coords'', cbuf, 132);
```

When a request is made to the *coords* access point, *SAOtng* waits for the user to position the mouse on the desired pixel and then press the *Return* key. The coordinates and raw data value at that point are returned to the calling program.

4. Summary

SAOtng and *XPA* are available as part of the SAO R&D software suite. Other programs contained in this package include *ASSIST* (a uniform interface to heterogeneous analysis systems), *XDir* (an X directory and file browser which supports user-defined actions for different file types), and *ncl* (*IRAF* cl with line-editing). This software suite has been ported to Sun, SGI, HP, Dec Alpha, PC/Linux, and PC/FreeBSD systems, and is available via anonymous ftp.[1]

The SAO R&D software suite is an embodiment of an evolving software cooperation philosophy that we hope to bring to astronomy and other disciplines. It reflects our understanding about how software systems (and researchers and developers) can act in concert without sacrificing their independence.

Acknowledgments. This work was performed in large part under a grant from NASA's Applied Information System Research Program (NAGW-3913), with support from the AXAF Science Center (NAS8-39073).

References

Mandel, E., Roll, J., Schmidt, D., VanHilst, M., & Burg, R. 1993, in ASP Conf. Ser., Vol. 52, Astronomical Data Analysis Software and Systems II, ed. R. J. Hanisch, R. J. V. Brissenden, & J. Barnes (San Francisco: ASP), 430

Mandel, E., & Tody, D. 1995, in ASP Conf. Ser., Vol. 77, Astronomical Data Analysis Software and Systems IV, ed. R. A. Shaw, H. E. Payne, & J. J. E. Hayes (San Francisco: ASP), 125

Mandel E., Swick R., & Tody D. 1995, The X Resource, Vol. 13, (Boulder: O'Reilly), 235

[1]ftp://sao-ftp.harvard.edu/pub/rd/saord.tar.Z