IRAF NEWSLETTER

June 1986 Number 1

Central Computer Services National Optical Astronomy Observatories* P. O. Box 26732 Tucson, AZ 85726

INTRODUCTION

The design, coding, distribution and support of a major software system such as IRAF are only possible if numerous conditions are met. The concepts and plans, established years before the system actually will go into use, must be sufficiently foresighted to accommodate changing scientific needs, evolving technical possibilities, and even development in computer science. The estimates of required effort must be reasonably accurate, and within the scope of the organization. Consistent "political" support is needed to see the project through the developmental stages to a demonstration of success. Lacking any of these, failure - sooner or later - is probable.

But even with the soundest preparation and backing, a successful project will still be the result of the collective efforts of a group of individuals working in close collaboration for a common goal. As software professionals know, the complexity of a project and the demands on group cooperation go up faster than linearly with project size. And IRAF is by an order of magnitude the largest software project ever undertaken at NOAO.

It seems appropriate, in the first issue of the IRAF Newsletter, to introduce the people currently involved in IRAF programming and distribution at NOAO.

Doug Tody is the designer of the IRAF system and has been chief programmer of the IRAF project since the inception of the project at KPNO in the fall of 1981. As chief programmer, he has written nearly all of the IRAF system software. Doug continues to work on the development of the system software while supervising the IRAF science software development group at NOAO.

Frank Valdes came to NOAO following a post-doc at Bell Laboratories, where he helped develop FOCAS, a faint object detection and classification system. After writing a new generation of FOCAS for NOAO he began early work on IRAF applications software. He has contributed to many of the IRAF applications. Currently he is responsible for all the spectroscopic packages: echelle, onedspec, longslit and multispec. In the near future he will develop tasks for performing radial velocity analysis and in the longer term he will incorporate FOCAS into IRAF.

Lindsey Davis joined the IRAF group following a post-doc at KPNO, during which time she worked on multicolor surface photometry of galaxies. In IRAF she wrote many of the image processing tools including surface fitting functions, geometric operators, and 2-d filtering operators as well as many of the IO routines. She is currently working on the aperture photometry reduction package.

---

* Operated by the Association of Universities for Research in Astronomy, Inc. (AURA) under contract with the National Science Foundation

Suzanne Hammond worked with KPNO computer support during the CYBER+IPPS years before joining the IRAF group. Suzanne wrote many of the dataio and mtlocal tasks, as well as many tasks in the plot package. She is currently working on the density to intensity conversion package.

Dyer Lytle has worked primarily in the solar area, first at Sacramento Peak and now in the IRAF group. He recently completed the reduction package for the vacuum telescope synoptic program.

Steve Rooke recently arrived from Duval Corporation where he developed software for geological surveys. He has been working in the graphics subsystem and on an adaptation to the SUN workstation. He is known to many of you already as the 'fix-it' man when there are site problems.

Jeannette Barnes is known to virtually all NOAO-Tucson visitors as the source of up-to-date assistance with almost all kinds of data reduction. She is now organizing the distribution of IRAF to the community, and will likely be the first, and most frequent, contact most of you will have with the distribution process.

Major contributions have also been made by George Jacoby (the originator of the onedspec package), Elwood Downey (who wrote the original CL that was designed by Doug), Richard Wolff (who wrote the current cv display package) and Ed Anderson (for many hours of testing and evaluating).

In December of 1983, Space Telescope Science Institute selected IRAF as the command language and operating environment for their data analysis system known as SDAS. Since that time the VMS/IRAF group at STScI (Tom McGlynn, Jim Rose, Fred Romelfanger, Cliff Stoll, and Jay Travisano), supervised by Peter Shames, have been responsible for the implementation of the VMS/IRAF kernel, the initial port of IRAF to VMS, and much of the development of the IRAF command language since 1984.

We are finding that supporting IRAF for use outside NOAO is challenging and time consuming. But the leverage of providing data reduction software on additional machines in places convenient to our observers amply justifies the effort for NOAO interests alone. And adopting an integrated software system as the vehicle for both NOAO and STScI data reduction promises long term benefits to a generation of astronomers. So we are fully committed to supporting IRAF in the long term to the best of our ability and to the limit of our resources.

Naturally, your advice and assistance are always welcome through any channels, formal or informal. Please let us know how IRAF works for you.

Steve Ridgway
Manager, CCS

IRAF PROJECT STATUS AND ACTIVITIES FOR 1986

While the IRAF system has been in heavy use within NOAO for nearly a year and is now in regular use at a number of sites in the community as well, the system still lacks fundamental system facilities and science packages. Our purpose here is to review the status of the project as of June 1986, presenting what we view as the major limitations of the system and outlining the work which is underway to provide the needed facilities.

1. System Software Priorities

The highest priority system software projects, now that version 2.3 of the system is complete, are the image display and display control subsystem and the SUN workstation project, both of which are closely related. The current IRAF display interface, i.e., the TV package (*display*, *cv*, etc.) is a prototype which has served its purpose and now badly needs to be replaced. This old software lacks many important capabilities, e.g., cursor readback, a device independent low level interface, and a program level interface for graphics and pixel i/o.

We plan to replace all of the old image display software this summer by a fully interactive, device independent display interface to be implemented as part of the GIO interface. Device interfaces are planned initially for the SUN color display, IIS models 70 and 75, the Peritek, a Grinnell, and the Gould DeAnza (IP8400). The image device interface will be designed to make it as easy as possible for user sites to interface their own devices; this is essential due to the large number of different types of displays in use in the community.

A keystroke driven "cursor mode" interface will be provided for interactive control of the image display as well as cursor readback, much as is currently done for graphics terminals. The *implot* program (used to interactively plot cuts through images) will be rewritten to permit use of the image cursor to locate the region to be plotted, and to make other improvements in the *implot* user interface. Modifications to GIO are planned to support device capabilities which change at runtime (e.g., the size of a window), and to support the use of "pop-down" or other device dependent types of menus for interactive option selection.

The reliability of the IRAF system software needs to be improved to make the system more immune to crashes following interrupts or failures of host system services caused by insufficient quota or limited system resources. This work has been put off in favor of other projects since the problem is not a serious one for the experienced user, but we appreciate that it can be quite exasperating, particularly for the user who is first learning to use the system.

All of the IRAF tasks which operate upon images are currently limited by the lack of system support for world coordinates, region masks, bad pixel masks, and history records. The current image data structures are also inefficient for operations upon databases containing hundreds of small images, e.g., one dimensional spectra. These facilities, along with the GIO imaging extensions discussed above, must be provided before IRAF can reach its full potential as an image processing system.

It is not yet clear whether these facilities can be added to the current IMIO interface and later layered upon the more general database facilities, or whether we must wait until the IRAF database interface (DBIO) is implemented. Probably we can provide support for at least image world coordinates before DBIO is available, but DBIO will be required before we can seriously address all of the limitations of the current IMIO interface. Since DBIO is needed to support a full IMIO interface, to provide catalog storage and query facilities for analysis programs, more efficient access to small images, and so on, implementation of the DBIO interface is a high priority, but it is a major project which we cannot hope to complete this year. Realistically, it is unlikely that the first version of DBIO will be available before at least mid-1987.

We badly need documentation for the IRAF programming environment before outside sites can hope to effectively make use of the environment to develop their own software. We can do without such documentation within the IRAF group, and instruction of the occasional outside programmer can be handled by having them come to Tucson for first hand instruction, so this project has yet to achieve a sufficiently high priority to be scheduled. Probably this will continue to be the case until the other projects discussed above have been completed.

Lastly, we are currently involved or soon will be involved in at least half a dozen ports of IRAF to new machines or operating systems, and we are heavily involved in supporting IRAF

on the machines to which IRAF has already been ported. Since this is a potentially infinite time sink, we are constantly working to make IRAF just a little bit more portable, more device independent, and more immune to installation errors, and therefore easier to maintain on all these machines. This is an ongoing activity of high priority which is necessarily competing for resources with the above projects, and which makes it difficult to schedule release dates, etc., with any accuracy.

2. Science Software Priorities

The highest priority science software projects at present are the new, soon to be released *digiphot* package, an improved CCD reductions package for use with the SUN systems at the telescope, a radial velocities package for use primarily for analysis following *longslit* reductions, and further additions to the *images* package including more facilities for image registration and modifications to make use of the planned image cursor input facilities. The initial release of the *digiphot* package will concentrate on fractional pixel aperture photometry in both circular and irregular apertures. Image centering, local background fitting, radial profile fitting, and (prototype) artificial starfield generation facilities will also be provided. This software will make use of the image cursor input facilities as soon as they become available.

All of the above science software projects are expected to be completed by the end of the year. In the longer term it appears that galaxy isophotal analysis should have the highest priority, followed by further development of the *digiphot* package and a port of the faint object detection and analysis package (*focas*) to IRAF. A project to perform image reconstruction from speckle data has been discussed, and the use of IRAF to host some portion of the GONG (global stellar oscillations) data analysis software is being investigated. The relative priorities of these projects are likely to be heavily influenced by the feedback we get from the astronomical community, so don't hesitate to let us know what types of facilities you would like to see added to the system.

3. Scheduled IRAF Releases for 1986

The first limited public release of IRAF occurred last February, when version 2.2 of UNIX/IRAF and VMS/IRAF were distributed to fifty or so mostly VAX sites. The second major release (version 2.3) is currently underway. This release includes the first release of SUN/IRAF and support for local plotter devices, many bug fixes, the new *apextract* package (part of *twodspec*), plus a number of features added to support the first release of the SDAS data analysis software by STScI. A detailed summary of the revisions in IRAF V2.3 is given later in this newsletter.

It seems likely that there will be further releases this summer or fall for SUN sites, as well as the first AOS/IRAF release, courtesy of the Univ. of Arizona working in collaboration with the IRAF group. The next major release of IRAF, however, is not expected until around the end of the year, and will include at a minimum the new image display software, the first version of the *digiphot* package, a density to intensity calibrations package, and the improved CCD image reductions package. No date has yet been set for this release, however it is certain that it will not occur until sometime after the first version of the SUN/IRAF system has been completed and installed on the mountain at CTIO and KPNO. The target date for this installation is October of this year, but with all the things we have going on it is likely that some schedule slippage will occur.

4. Ports in Progress

The VAX/UNIX (4.2BSD Berkeley UNIX) and VAX/VMS versions of IRAF have been in production use for some time now. SUN/IRAF has been functional since last fall, but development of this system has gone slowly due to the diversion of manpower to develop and distribute IRAF versions 2.2 and 2.3. We have recently completed the first version of SUN/IRAF, after a long period of testing which uncovered many bugs in both IRAF and the SUN Fortran compilers. We currently have IRAF V2.3 running on both the SUN-2 and the SUN-3, but our last SUN-2 will soon be converted to a SUN-3 and hence we will no longer be able to directly support the SUN-2. The first version of SUN/IRAF is little more than the VAX version of IRAF ported to the SUN, hence there is no support for windows, pixrects, pop down menus, the Sky Warrior array processor, and so on. This will be remedied by the time SUN/IRAF is installed on the mountain (except possibly for support for the array processor).

A MicroVax II with bit-mapped display has recently been installed, courtesy of the Galileo project. This machine is running VMS and communicates with the 8600 via Decnet, allowing us to test both the MicroVax version of IRAF and the use of Decnet for IRAF network communications.

The Steward Observatory port of IRAF to AOS/VS was set aside for several months in favor of another project, but work on this port resumed two weeks ago with the goal of completing the port sometime this summer. We are providing consulting services and will help test and certify the system when the port is far enough along. The Jupiter/ISI port at IPAC3 (the IRAS data center) has produced an almost functional system, but there continue to be problems with the Fortran compiler.

Other ports not yet started but about which we have been contacted are for the Alliant and Convex vector machines (in which we are very interested), and for a couple Charles River Data Systems supermicros (System V UNIX). Other groups have expressed interest in porting IRAF to an IBM machine (TSO) and a Cyber (NOS/VS), but given the unfriendly nature of the operating systems on these machines these ports may not be worth the substantial effort likely to be required. We are currently obligated to at least a feasibility study for a port to the San Diego consortium Cray, but since this machine runs CTSS we cannot say if the port is practical until we have had a chance to carry out the feasibility study.

Doug Tody

HOW TO GET THE NEW RELEASE

Sites that have requested the SDAS release through the SDAS/IRAF request form in the April Space Telescope Science Institute Newsletter will receive the new release without further communication with us here at NOAO. Bob Hanisch at the Space Telescope Science Institute has sent and will continue to send copies of all SDAS/IRAF request forms to us so that we can ship the IRAF system directly to the requesting sites. (SDAS itself will be provided by STScI.)

All other sites requesting this new release are asked to complete and send to us the attached order form. Sites that have an order form on record pending this new release, of course, do not need to send in another form.

We are also asking all current sites to send, along with the order form, the magnetic tapes previously sent to them containing the IRAF software. We will then copy the new system onto these tapes. Current sites that have already sent their reorder through the SDAS form are requested, as well, to send the old IRAF tapes back to us. Your cooperation with this request will be greatly appreciated.

There will be no charge for this current release or any of the documentation. However, charges are being considered for forthcoming releases as well as for the documentation - we will keep you informed.

Jeannette Barnes

IRAF PERFORMANCE EVALUATION REPORTS

Reports concerning the performance of IRAF are vital for the forward motion of this project. We are very interested in any and all problems and recommendations - system related as well as science software related. Phone calls are always welcome - if we can't answer your questions immediately we'll give you a call back. If reports come in via regular mail or electronic mail be assured that all messages are looked at even though you may not hear from us immediately.

Our electronic mailing addresses, for communicating with the IRAF group, in general, are listed below. (Please note that at the time this Newsletter goes to press the only operable network is the UUCP network. It is hoped that the others will come back on line in a few weeks.)

BITNET	IRAF%DRACO@CITPHOBO
INTERNET(ARPAnet,MILnet,etc)	IRAF%DRACO@Hamlet.Caltech.Edu
Caltech/NRAO network	IRAF@DRACO (softools) or DRACO::IRAF (DECnet)
UUCP/Usenet network	ihnp4!noao!iraf or seismo!noao!iraf

For more information on communicating via electronic mail with the IRAF project see the article "Electronic Communication with the IRAF Project" in the installation guide booklet sent out with the IRAF release.

Phone calls are always welcome. For system questions and/or problems please contact Steve Rooke at 602-325-9399. For distribution questions or applications questions and/or problems please call Jeannette Barnes at 602-325-9381.

If you are requesting the new release of IRAF please be sure to fill in the section on bug reports with your re-order, especially if your bugs and/or recommendations have not been reported as fixed in the attached revisions notices.

Jeannette Barnes

NOTES FROM THE VMS SYSTEM MANAGER'S PORTFOLIO

As you might imagine, we have had plenty of experience with the vagaries of the IRAF/VMS mixture, and the following are a few thoughts, with no attempt at blame.

Quotas

For the users of IRAF, the most important quotas which should be set in SYSUAF are 1) PRCLM = 10, 2) JTQUOTA = 2048 (the DEC rumor mill suggests strongly that a lot of software needs double the default of 1024), 3) PGFLQUOTA (large, over 24,000; but dependent on the user's other needs), 4) BYTLM (over 24000, probably larger), 5) WSEXTENT > 3500 (for best use of an empty machine; we use 4096, and, of course, there's no penalty for making it larger).

System memory

Global space. You will have noticed the comments about the global memory parameters GBLSECTIONS and GBLPAGES, both in the installation guide and elsewhere in this newsletter. In addition to the space needed if you INSTALL the most commonly used IRAF executables, the subprocess structure of IRAF requires global pages for activation: about 8 pages each, which will show up in the 'Group Global Sections' part of a 'LIST/GLOBALS' command from the INSTALL utility. Therefore, if you run close to the edge on global space, you may find that some IRAF users will be unable to create additional subprocesses (this happened to us!).

Working sets and total memory. On a system with not much memory, several IRAF users will rapidly remove all of your free pages (as seen by a 'SHOW MEMORY/PHYSICAL' command). The system will not reclaim this space until it is needed, because DEC recommends that the system parameter PFRATL be set to zero. The possible problem with IRAF is that, in general, only one of the IRAF subprocesses is using much CPU, and the others are 'waking up' every so often just to check flags and look for messages. Because of the QUANTUM parameter, the system is less likely to notice these 'dormant' processes, and so there is generally a slightly longer delay than usual before memory is freed. **Translation**: everyone else's response gets worse, even for trivial tasks - after all, you need memory even to buffer IO to a terminal. Your choice is to set PFRATL to one: then, the system reclaims memory from everyone who uses CPU even if it doesn't need it. Of course, if the dormant IRAF task needs to reactivate, it then has to page its working set back in from memory. On a lightly loaded machine, this is not a hard page fault, because those pages will still be present on the modified list. **Translation**: everyone else gets decent response, but at heavy times the IRAF people will notice a longer delay.

Process slots. Finally, again because of the subprocess structure of IRAF, each user will take from four to as many as 11 (actually, PRCLM+1) process slots. If you increase MAXPROCESSCNT, then AUTOGEN will increase other things, most significantly, the size of your page and swap files, which you don't really want to be too big. I'm still experimenting (they don't let me take the system up and down at will to test these effects!), but it seems that keeping BALSETCNT low may help. You can choose just to let process creation fail, but it's frustrating when you can't even log on at the system console.

Useful utilities

We also have available various DCL files and programs to cope with some of the hiccups, most of which we are rapidly eliminating. For example, as mentioned in the 'experiences installing IRAF' section of this newsletter, if you reset file ownerships you mess up the date checking part of IRAF's parameter file structure. I therefore have a command procedure/program which alters

ownership by reading the date, changing ownership, and then resetting the date. (The program part is available in both C and FORTRAN.) If you notice a need for other programs, we may have them already, so ask!

Nigel Sharp

NOTES AND EXPERIENCES INSTALLING IRAF V2.2

1. VMS/IRAF

1.1. Introduction

Experiences with the Winter 1986 limited release have shown that factors in the local VMS environment can cause problems for the installer and sometimes for general IRAF users. Most of these problems derive from either inadequate process quotas or tight system resources. A solution was found in all cases for each individual system, usually via consulting us over the telephone, and sometimes by having us login remotely.

The *VMS Installation Guide* contains recommended process quotas and mentions certain VMS Sysgen Parameters as they are set on our systems at NOAO. In practice some of these recommendations were insufficient at certain sites. The actual quotas and system resources needed at a site depend on many local factors and are too involved to go into here, even if we understood them all. We don't want to simply recommend higher quotas because that could hinder performance at some sites, e.g., sites that have to support lots of users.

The new release contains a benchmarking package which should help us gauge more accurately the performance of IRAF on the various VAXes (see *A Set of Benchmarks for Measuring IRAF System Performance*, Doug Tody, March 1986, included with this IRAF distribution). We should now be able to compare the observed timings with those of other VAXes to determine whether a problem exists and to help pinpoint its source.

We mailed a letter on April 16 regarding a VMS/IRAF kernel bug that was causing performance problems. Some of the sites without a DEC C Compiler experienced a worsening of performance after making the binary patch we recommended. We suspect the problem had to do with the different version of VMS (V4.1) on the machine we used to test the patch procedure. We apologize for this and we should have investigated the solution more carefully.

These notes are derived from our records of actual experiences at many VMS sites, and are grouped loosely by symptom or the source of the problem. We do not yet have sufficient experience to provide a bulletproof troubleshooting guide, so it is best to just read through this material for suggestions if you encounter problems that you think might relate to the VMS environment in which IRAF is running.

1.2. Installation Problems

A few sites had trouble relocating the IRAF root directory somewhere other than at [IRAF]. More than a few sites had trouble with the VMS Linker failing when trying to rebuild all or part of the system. There were several other problems related to recompiling or relinking the system. These difficulties are discussed below.

1.2.1. Problems Related to Root Directory

The distribution tape comes with the IRAF directory system located at the root directory [IRAF]. There are instructions in the Installation Guide for editing certain files if it is desired to locate the root elsewhere. A few sites left out one or more steps, causing the system to fail at runtime or during relinking.

In the following discussion, as in this entire document, all directory names will be given as "[IRAF...]". If your site wishes to locate the IRAF system elsewhere, e.g. "[USER.IRAF...]" simply make the mental substitution in what follows. The files that need to be edited to relocate the IRAF root are:

`[IRAF.VMS.HLIB]IRAFUSER.COM`

Just follow the directions in the Installation Guide. Remember to reexecute this file ("@[IRAF.VMS.HLIB]IRAFUSER.COM"), or login again if you had done so earlier, before proceeding.

`[IRAF.VMS.HLIB.LIBC]IRAF.H`

Follow the Instructions carefully. In particular, make certain there are definitions for HOST, IRAF, and TMP. One site had commented out the definition of TMP, causing the system to fail at runtime.

Later in the Guide, under "Complete the System Configuration" are directions for copying this file to SYS$LIBRARY. One or two sites changed the directory system after the initial installation (editing this file) but forgot to replace the version in SYS$LIBRARY. The copy in SYS$LIBRARY is the file actually read at runtime and is required to resolve the root directory pathname if the logical name translations fail for some reason (more later). If a site wishes to maintain two installations of IRAF (on different disks for example) on the same computer, please contact us for advice.

`[IRAF.LOCAL]LOGIN.COM`

Just follow the directions.

1.2.2. Relinking Problems Related to Quotas and Resources

Several sites experienced difficulties when relinking the system. In one case the job logical name table had filled up causing one of the subprocesses involved in linking to die when it was unable to resolve the IRAF root directory name. Ordinarily this is not a problem, but in this case an error had been made in the copy of [IRAF.VMS.HLIB.LIBC]IRAF.H in SYS$LIBRARY, which is the backup mechanism for pathname resolution.

At about four sites the VMS Linker died with an access violation immediately after it started up. We reasoned that this had to be due to a scarce global system resource, and in fact at all these sites the problem went away after the system manager increased the two Sysgen Parameters GBLSECTIONS and GBLPAGES. Since this involves rebooting the VAX, it should not be done lightly, and determining the best values of these parameters is a job for the system manager; we can try to assist if necessary.

At one site the Linker apparently died because the process quota PGFLQUOTA was too low (at least the problem went away after increasing it). This is a parameter whose ideal value

seems to depend on the local system environment.

MKPKG died at another site while trying to "copy dua2:[iraf.lib]libsys.olb libsys.olb". It turned out that this site had missed the recommended process quota PRCLM of 10 in the "VMS Quotas..." section of the Installation Guide. (The current value of PRCLM can be determined by the VMS command "show process/quota"; it is the "Subprocess quota" value; note that each additional process decrements one from the PRCLM setting).

MKIRAF failed at one site due to a name collision. At this site, there was a system logical name "SYS$USER1". MKIRAF performs a number of edit substitutions, one of which was for the string "USER". An attempt has been made to make MKIRAF.COM bulletproof in the new release (V2.3).

At another site MKIRAF brought up EDT interactively (it is supposed to run invisibly) during installation. It turned out that the string "EDIT" had been redefined on that system to "@somefile.com". With that command file EDT was unable to take the proper command qualifiers from MKIRAF.COM, and thus came up interactively. This particular problem has been cleared up in the new release, but it is possible that somewhere else in the system problems might occur if the usual name of a VMS command has been redefined.

There was a VMS Fortran compiler problem in VMS V4.1 which prevented IRAF from opening more than one connected subprocess. One site attempted to build IRAF under V4.1 (a full recompile-relink sysgen), and this problem disappeared when VMS was upgraded. We try to keep IRAF free of dependencies on particular versions of VMS, but this is not always possible, and sites running versions much older than the one currently supported may run into similar problems.

One site acquired a DEC C Compiler after the initial IRAF installation, and got the following messages when MKPKG was run afterwards:

```
link:  PANIC:  Illegal file descriptor, bad fd, or file not open
%SYSTEM-F-ILLSEQOP Illegal sequential operation
```

Bootstrapping the system using the new compiler caused the problem to go away (see the paragraphs on "bootstrapping" in the "Relink the System" section of the "Installation Guide").

1.3. Runtime Problems

Most runtime problems resulted from inadequate process quotas. Several sites neglected to establish the minimum process quotas recommended in the relevant section of the Installation Guide. At a few sites the value of PGFLQUOTA we recommended may have been too low. Problems with the VMS Sysgen parameters GBLSECTIONS and GBLPAGES were discussed above, although they may have been noticed only when relinking an application package long after the installation.

1.3.1. Process Quotas and Privileges

The IRAF architecture makes extensive use of communicating processes. Under VMS, each additional subprocess may decrement from the original (startup) process's quotas, causing problems at runtime. The minimum quotas recommended in the Installation Guide are adequate for most sites, but not always. Actions taken in the system and user VMS logins at some sites may use up some of these quotas (particularly logical name table slots), while at other sites this is not a problem.

One way to determine if you are running into resource problems is to attempt to spawn several nested subprocesses:

```
cl> !spawn
$ spawn
$ spawn
$ spawn
$ spawn
$ logout
$ logout
$ logout
$ logout
cl>
```

If at any stage you get a fatal VMS error and are unable to go further, you know you are short on some resource, and if you are lucky the diagnostic message will pinpoint the problem.

We did not publish a recommended value of the process quota "FILLM" ("Open file quota in ‘show process/quota’"). The number of open files required at a given time varies widely with the complexity of the application and the number of simultaneous processes running. Certain image processing applications, if given a long list of images upon which to operate, may require a large number of open files. At NOAO, our normal setting is for 56 open files (don’t ask us why), but typical values range from 20 to 40. UNIX/IRAF is currently limited to 20 physical file descriptors, so any value of 20 or larger should work fine for VMS too.

One site changed all the file ownerships of IRAF users and got "parameter file out of date" messages for all the parameter files in their "uparm" directories. Changing a file ownership or protection also changes the file-modified date in VMS. The only problem this causes with the parameter files is that the original default values replace the most recent user values.

1.3.2. Miscellaneous

One site was able to read only one FITS file at a time from tape. It turned out the user had allocated and mounted the tape from VMS before getting into the CL. When reading tapes within IRAF, simply load the tape onto the drive, placing it "on-line", and use the IRAF "allocate" command from the CL. Mounting is taken care of by the IRAF DATAIO tasks.

2. UNIX/IRAF

Significantly fewer difficulties were experienced installing UNIX/IRAF than with VMS/IRAF (as discussed above, most VMS problems derived from process spawns and resources, which are handled quite differently in UNIX).

2.1. Installation Problems

One site got messages from MKPKG indicating that certain include files were not found at compile-time. It turned out that when they had edited the file $iraf/unix/hlib/libc/iraf.h, they had only edited the root directory pathname in three lines near the top of the file. The instructions in the "Configure the IRAF Environment" section of the Installation Guide call for editing all pathnames in this file. The XC compiler gets some of its pathnames from iraf.h, and consequently all pathnames in this file must reflect the IRAF root directory in use.

Another site had many mysterious symptoms including numerous unresolved symbols when MKPKG was trying to link executables. It turned out they had an old version of IRAF located at another root directory. Subprocesses were picking up the old version through their normal executables path because there was no .cshrc file in their home directory defining the current $iraf root. Beware of trying to run multiple versions of IRAF on the same system. This can be done, but please contact us in order to avoid problems of this type.

3. Problems Not Specific to Either VMS or UNIX

It is important to follow the instructions in the Installation Guide. One site had neglected to issue a "MKIRAF" command before entering the CL (this should be done before entering the CL for the very first time or after a major update). When a user tried to use IMCOPY, he got a message "ERROR: Cannot open imdir$...", since "imdir" was undefined.

One user was getting errors in image processing tasks "image dimensions greater than 2 not supported" when reading IRAF images produced from FITS tapes with the RFITS task. It turned out that although the image was supposed to be two-dimensional ([320,28]), the FITS parameter NAXIS was set to 3, rather than 2, in the data tape. WFITS faithfully created a three-dimensional image ([320,28,1]).

A number of minor difficulties were related to configuring interactive graphics terminals or hardcopy plotter devices (see the section "Interfacing to New Graphics Devices" in the Installation Guides). Most of the problems were highly specific to individual sites and will not be covered here. One or two things should be mentioned, however.

When configuring the file dev$termcap for LPRINT output, attention must be paid to the "dispose" field of the DD instruction (the third and last comma-delimited field). The distribution tape comes with "/local/bin/plot" as the pathname to the Imprint software, and one site had this program in "/usr/local/bin/plot".

Users with Selanar terminals may experience constant clearing of the graphics screen every time a cursor mode command is issued. The problem can be overcome by entering Setup Mode C, and setting the flag "c-0:3" to "false". We do not have a Selanar at NOAO, and we have not heard of a way to download this flag from IRAF.

It is necessary also to have entries for "li" (lines) and "co" (columns) in the dev$termcap entry for a new printer, as in all the existing examples. One site made a printer entry lacking the "li" entry, and their output file was being filled with an infinite number of line-feeds (that particular symptom will be cleared up in a new release).

Sites lacking NCAR metacode translators that have the Versaplot software library can contact us for instructions on configuring the IRAF Calcomp graphics kernel for use with Versatec plotters. Additional documentation will be available in the future to assist in the problem of interfacing new graphics devices.

4. How to Get Assistance

Please do not hesitate to contact us when you have a problem. We cannot fix problems we do not know about. Most difficulties can be cleared up readily with a telephone call. Other times we ask to be able to login to your system remotely to diagnose the situation. Ordinarily we have been able to respond within a day or two, and if a problem is particularly critical we may be able to attack it immediately.

It helps if you can tell us the dialup numbers for your system, and whether we are logging in directly or through a Micom, etc. We prefer that you establish an account for us to use called "IRAF" as recommended in the Installation Guide. We will either leave electronic mail for you or follow up by telephone to explain our findings.

If you experience problems during a complex CL session, please dump a sufficient number of lines from your CL history into a file to help us diagnose the difficulty later:

```
cl> history 200 > temphis.txt
```

Also, please write down or otherwise trap any IRAF or host system error messages. Most application problems can be solved over the telephone by referring to the history and the error messages.

Note the information on the "Gripes" utility and routing electronic mail to us, along with contact personnel and telephone numbers elsewhere in this Newsletter. We welcome written reports on your experiences and suggestions for improvements to the system.

Steve Rooke

SUMMARY OF REVISIONS IN IRAF VERSION 2.3

1. System Software Revisions

In this section we summarize those new features or revisions to the IRAF system software likely to be of interest to the user or programmer. A detailed record of the revisions to the system software, including all bug fixes, is presented in the document *System Software Revisions - IRAF V2.3*. This was too large to include in the newsletter and in any case would probably be of interest only to a few systems programmers.

Many of the system software revisions made in this version of IRAF were made directly or indirectly to support the first release of the SDAS science data analysis software by STScI. Sites which already have IRAF and who do not plan to run SDAS immediately may notice that some things have changed a little without necessarily becoming better. In particular, the bottom end of IMIO (the image i/o interface) has been completely restructured to permit use of IRAF with either IRAF or SDAS format images, and the package structure of IRAF has been changed to isolate the NOAO optical astronomy software from the IRAF system software. We apologize for any confusion these changes may cause to sites which already have IRAF.

1.1. Command Language Revisions

The major changes to the command language made in IRAF version 2.3 are summarized below, in no particular order.

1.1.1. syntax errors

When a syntax error occurs, the illegal statement will be printed followed by a carat pointing to the part of the statement which is illegal. Due to the limitations of the parser technology used, the carat will often point to the token *following* the illegal token.

1.1.2. process cache changes

The size of the process cache is now controlled by the new CL parameter **szprcache**, set at login time by a parameter assignment statement in `hlib$clpackage.cl`. This makes it easier for the system manager to configure the size of the process cache differently for each IRAF host machine. The users may also change the size of the process cache at runtime if they wish, although of course they may not have sufficient quota to make use of all the new process cache slots. Setting the value of *szprcache* does a `flprcache` *and* restarts the system

subprocess `x_system.e`, hence a command such as "szpr=3" is useful for completely initializing the process cache without having to log out.

When a package is loaded and later exited with "bye", the CL will now remove processes from the process cache connected since the package was loaded. For example, if one loads the `proto` package, runs the `proto.fixpix` task, and then "bye's" out of the package, the executable `x_proto.e` will be disconnected. This is desirable for VMS/IRAF since under VMS, a process will consume a significant fraction of the available system resources even when idle.

1.1.3. minimum match abbreviations

The minimum match abbreviation algorithm has been modified to permit a minimum match to be specified uniquely within the context of the *current* package, rather than for all loaded packages, provided the abbreviation entered does not match the *full* name of a task in some other package. In practice this means that the minimum acceptable abbreviation will be a function only of the current package, and will rarely be affected by any other packages that may also be loaded.

Abbreviations are now permitted regardless of context, i.e., task and parameter names may now be abbreviated in scripts if desired. This is not recommended for procedure scripts, but is often desirable when entering a sequence of commands into a file.

1.1.4. string operations

The expression *param_name*.`p_length` will now return the maximum length of a string valued parameter. A new intrinsic function **strlen** has been added to return the current length of a string valued parameter.

1.1.5. dictionary and stack sizes increased

The size of the CL dictionary and stack areas have been increased significantly to accommodate larger CL procedure scripts, deeper procedure/task nesting, etc. Note that although all storage is dynamically allocated in the CL, there is nonetheless a compile time limit on the size of the stack areas.

1.1.6. loading of packages from within procedures

It is now possible to load a package from within a CL procedure (or block) to reference the tasks therein, without actually executing the package, i.e., interpreting commands from the terminal.

1.1.7. eparam enhancements

The *eparam* task has been modified to display hidden parameters in parenthesis like *lparam*. Other changes were made to fix bugs, add support for menu mode and the new VOS terminal driver, and make other minor enhancements.

1.1.8. menu mode

The CL now supports a new mode at the task and package level. Recall that modes are used to control whether or not the CL queries for the value of a parameter at runtime. Hidden mode parameters are used without a query, query mode parameters are always queried for unless set on the command line, and auto mode is set to defer the selection of the actual mode to a higher level. If menu mode is in effect when a task is run, *eparam* will be automatically called to allow the user to edit or review the parameter set before the task is executed, and all queries for auto mode parameters will be disabled when the task begins executing. Exit *eparam* with

<ctrl/z> to run the task, or with <ctrl/c> to abort the task. Menu mode is permitted only for tasks run interactively; menu mode is not currently permitted for tasks called within a script or procedure.

To set menu mode for a task set the value of the "mode" parameter. All tasks have such a parameter.

```
task.mode = menu + learn
```

To set menu mode for some subset of the tasks in a package, one can either set menu mode for each individual task, or set auto mode for each task, and menu mode at the package level, e.g.:

```
task.mode = auto
package.mode = menu
```

The advantage of setting the mode at the package level is that the mode of all "auto" mode tasks in the package may be changed by a single assignment.

Menu mode is undesirable for tasks with only a few parameters normally entered on the command line, but may be useful for tasks with such large parameter sets that *eparam* is often called before running the task.

1.1.9. learning of parameters

Learn mode is now enabled only for tasks run interactively. The parameter set for a task called from within a script or CL procedure will not be learned unless an explicit parameter assignment takes place.

1.1.10. new logging facilities

The logfile mechanism has been extensively revised to increase the amount and type of information which can be logged. The new CL parameter *logmode* controls the type of information to be logged. The old CL boolean parameter *keeplog* is still used to turn logging on and off, and *logfile* is still used to store the name of the logfile. The default name of the logfile has been changed to `home$logfile.cl`, hence the logfile will appear in the users login directory just before the *login.cl* file.

The new logging mechanism, like the original one, will record commands entered by the user in the logfile. In addition, CL error messages may now be logged, and a "trace" feature will cause messages to be logged whenever a CL task is run (this feature is useful for debugging CL scripts). A new builtin task **putlog** has been added so that the user or a script may easily add lines to the logfile. Type "help logging" for additional information on the new logging facilities.

1.1.11. ehistory, eparam initialization

The **ehinit** and **epinit** variables, used to set options in the history and parameter editors, have been changed from environment variables to CL parameters. Since the scope of the variables is limited to the CL there is no need to use the more general system wide environment facilities.

1.1.12. graphics stream redirection

The CL i/o redirection syntax has been extended to permit redirection of the standard graphics streams on the command line. The syntax is as follows:

```
>(G|I|P) + file        # (create file)
```

or

```
>>(G|I|P) + file       # (append to file)
```

e.g., `>G` *file* to redirect STDGRAPH to *file*, or `>GI` *file* to redirect both STDGRAPH and STDIMAGE to *file*. Note that multiple redirection arguments may be given on the command line. The `>GI`, etc., is lexically a token, hence there must be no space after the >, and the `GIP` must be upper case. A special case is

```
>G dev$null
```

which may be used to redirect the standard graphics output of a task to the null file, i.e., discard any graphics output.

1.1.13. logout.cl support added

If a file named `logout.cl` is present in the users HOME directory when logging out of the CL, the CL will now interpret any commands in the file before shutting down.

1.2. VOS Revisions

The major VOS revisions in this release are the modifications to IMIO (the image data structures interface) to support multiple disk image formats, and the addition of a new graphics kernel to make it easier for user sites to interface their own plotter devices. Other revisions include a new VOS terminal driver and the debugging of the backlash escape mechanism to permit use of FIO metacharacters (`$[]`) in host filenames.

1.2.1. multiple image formats

The IMIO (image i/o) interface has been restructured to isolate all knowledge of the physical image data structures into a new interface called the **image kernel interface** (IKI). The IKI can support multiple image kernels, each of which implements a standard set of operations for a particular image disk format. Currently there are two such image kernels, the OIF kernel for the old IRAF image format (still the current format but not for long), and the STF kernel for the STScI SDAS/GEIS VAX/VMS group data format. IRAF programs can read, write, create, and update either format, for the most part transparently to the user and to the applications programs.

The type of image format used for a particular image is indicated by the filename extension of the image header file. OIF format images will now be created with the filename extension ".imh", whereas originally they did not have any extension. The filename of OIF format pixel files is now formed by taking the root name of the image and adding the extension ".pix", whereas originally the pixel files had a meaningless computer generated unique filename. The new IMIO allows OIF format pixel files to be created in the same directory as the header files if desired, or in a subdirectory of the directory in which the header files reside. Old OIF format files can still be read by the new IMIO even though the filenames are different.

The new IMIO provides full support for the STF group structured images. A special notation permits the creation of new STF group format images, and an index notation may be used to indicate the individual group to be used in a particular operation. For example, in an image create operation, `pix[1/10]` would create a new group format image with GCOUNT=10 and write into group 1. A subsequent reference to `pix[3]` would access group 3. Image sections may be combined with the new group notation if desired.

1.2.2. new SGI graphics kernel added to GIO

A new graphics kernel called the SGI kernel (simple graphics interface) has been added to GIO to make it as easy as possible to interface new plotter devices to IRAF. The SGI kernel will write a simple format metacode or bitmap raster file and then call a user supplied foreign task to process the plot file to the host system plotter. For example, it is relatively easy to write a host Fortran or C program to translate an SGI metacode file for a QMS or Imagen laser printer, since the SGI metacode format is very simple and only four metacode opcodes need be dealt with (all character generation, etc. is done in the higher level IRAF code). The interface for a bitmap raster device such as a Versatec or Printronix is even easier: on UNIX one can dispose of the bitmap file directly with *lpr*, and on VMS one need only write a small VMS Fortran program to reformat the raster file with the correct VMS record structure, and then copy it to the device.

One might well ask why we chose to invent yet another graphics interface when NCAR, Calcomp, GKS, CGI, etc. are already available. The reason is that it is quite a difficult task to add support for a new device to any of these other, more "fully featured"(vastly more complex) standard graphics interfaces. If a site already has one of the standard graphics libraries and it supports their local plotter device, we can probably interface to it with one of our other kernels (a VMS/McVAX interface has been added in this release, for example). If that is not the case, however, the SGI interface simplifies the task of building a new interface. The point of the SGI interface is that the data structures have been reduced to the simplest possible form, and no knowledge of the IRAF internals is required to provide the last little bit of software needed to connect to the physical device. To make the task even easier, we will supply translators for the Imagen, Versatec, etc., which can be used as templates to interface new devices.

1.3. Host System Interface Revisions

The detailed system software revisions notes should be read for a comprehensive review of the changes at that level in the system. The major revisions were the addition of NCAR/MCVAX support to VMS/IRAF, so that VMS sites which have or which can obtain the MCVAX metacode translator can use it to get IRAF plotter output, changes to the VMS *mkiraf* to provide conditional deletion of the `UPARM` parameter files, and a query for the default terminal name. There were a number of changes and bug fixes to the *mkpkg* bootstrap utility. Other changes were made to the `HLIB` system configuration files to make it easier to interface outside packages (e.g., SDAS), and to give the system manager more control over the important IRAF system parameters.

2. Applications Packages

The basic package structure of IRAF has been modified to make a distinction between the system packages and the NOAO optical astronomy packages. The basic directory structure of the system was also changed to reflect the new package organization. These changes were necessary to better isolate science software such as the NOAO and STScI/SDAS packages from the system software, producing a more logical package structure, and to make it easier to install and maintain the science packages.

The new root menu of IRAF is as follows:

```
dataio      images      lists       noao        sdas        system
dbms        language    local       plot        softools    utilities
```

The NOAO menu is as follows:

```
artdata     astutil     focas       mtlocal     proto       twodspec
astrometry  digiphot    imred       onedspec    surfphot
```

Three new packages were added and three old packages were extensively revised. The NOAO local tape readers were moved from the DATAIO package into the new MTLOCAL package. The astronomically oriented utility tasks were moved from the UTILITIES package to the new ASTUTIL package. The old LOCAL package was renamed PROTO, and a new LOCAL package was added in the directory "iraf$local/tasks".

The concept of the new PROTO package is appropriate for what the old LOCAL package was used for, i.e., prototype, temporary, or contributed tasks which are part of the NOAO package and are exported with the system, but which are expected to eventually disappear or be replaced by planned system facilities. The new LOCAL package is a place to put tasks of strictly local interest, or tasks which are not portable, e.g., foreign tasks and the Peritek package. The LOCAL package should be particularly useful for outside sites as it gives them a place to put locally added tasks which will not be affected by future updates of the system. Also, the framework (mkpkg, local.cl, etc.) is all set up, making it easier for outside sites to add their own software without having to figure out how to set up an IRAF package.

2.1. System Packages

In the SYSTEM package, a completely new **directory** task has been added to replace the old program which never worked very well. An almost completely new **page** program has also been added; the new program provides a nice keystroke driven interface, forward and backward motions through a file list, and many other new features. A similar keystroke driven interface has been added to the **help** task.

A new terminal driver was added to the VOS with support in the **stty** program in the system package. A *ucasein* terminal driver option permits IRAF to be used with monocase terminals, or ordinary terminals with the shiftlock on. A *ucaseout* option causes all textual output to the terminal to be converted to upper case. Finally, a *logio* option causes all physical i/o to the terminal to be logged in a file in the user's home directory; this is especially useful for debugging new *graphcap* or *termcap* device interfaces. Read the manual page for *stty* and the detailed systems revisions notes for additional information.

A new task **imrename** has been added to the IMAGES package. Unlike the ordinary file renaming task, *imrename* will rename the pixel file as well, or even move it to a different directory depending upon the value of the IMDIR environment variable when the rename operation takes place. In the PLOT package a new task **gkimosaic** has been added, along with a couple of new tasks to support the new SGI graphics kernel.

2.2. NOAO Packages

A new version of the **apextract** package has been installed in the **twodspec** package. A task called **r2df** has been installed in the **mtlocal** package to support the CTIO 2D-Frutti tape format.

A detailed summary of the revisions to the NOAO science packages as well as additional information on revisions to the system packages is included later in this newsletter.

Doug Tody

The IRAF NEWSLETTER is published prior to the release of a new version of the IRAF system. The newsletter's primary functions will be 1) to announce a new IRAF release, 2) to inform sites of the revisions and updates contained in the new release, and 3) to present an overall status report of IRAF. This newsletter announces the release of IRAF Version 2.3, the first and previous release being IRAF Version 2.2 in February 1986.