PC-IRAF V2.12 Installation Guide

Doug Tody
Mike Fitzpatrick

IRAF Group
National Optical Astronomy Observatory†
May 2002

ABSTRACT

This document describes how to install or update IRAF on an Intel-based PC running either Linux, Freebsd, or Solaris, or a PowerPC based Macintosh running Mac OS X. Both standalone and networked, multiple architecture configurations are described. Only those issues which one must understand to install IRAF are discussed here; a companion document, the *SUnix IRAF V2.12 Site Manager's Guide*, deals with other issues such as interfacing new devices, configuring the IRAF networking system, adding layered software, and so on.

July 16, 2002

†Operated by the Association of Universities for Research in Astronomy, Inc. under cooperative agreement with the National Science Foundation.

Contents

1. **Introduction** 1

2. **Installing PC-IRAF** 1
   - 2.1. Prepare the root IRAF directory 2
     - 2.1.1. If updating an existing IRAF installation.... 2
     - 2.1.2. If installing IRAF for the first time... 3
       - 2.1.2.1 Create the ’iraf’ User Account... 3
       - 2.1.2.1 The ’iraf’ User Account for OS X Systems... 3
       - 2.1.2.2 Create the IRAF Directory Tree... 4
   - 2.2. Install the files 4
     - 2.2.1. Installing from a network distribution.... 4
     - 2.2.2. Installing from CD-ROM 5
     - 2.2.3. Configuring the BIN directories 5
   - 2.3. Merge local revisions back into the new system 6
   - 2.4. Run the INSTALL Script.... 7
     - 2.4.1. System Settings.... 8
     - 2.4.2. System Verification.... 8
     - 2.4.3. System Installation.... 9
     - 2.4.4. Post-Install Configuration... 10
       - 2.4.4.1 IRAF Networking Setup... 10
       - 2.4.4.2 Tapecap Setup... 11
       - 2.4.4.3 Deleting Unused HSI Binaries... 12
       - 2.4.4.4 Strip System Sources.... 12
     - 2.4.5. Post-Install Verification... 13

3. **System Checkout** 14
   - 3.1. Frequently Asked Questions 14

Appendix A. A Complete Example Using Single-File Distributions 15

Appendix B. A Complete Example Using Split Distributions 16

PC-IRAF V2.12 Installation Guide

Doug Tody
Mike Fitzpatrick

IRAF Group
National Optical Astronomy Observatory†
May 2002

1. Introduction

This document covers the general procedure for installing IRAF on a UNIX system. It has been tailored for PC-IRAF and is accurate so far as the basic installation is concerned, but we do not attempt to cover the new features of PC-IRAF here. Please see the README file in the PC-IRAF distribution directory (ftp://iraf.noao.edu/iraf/v212/PCIX/README) for information on the platform specific aspects of the release or other timely information. The PC-IRAF README also includes a summary of the installation process for those who are installing from disk and who have already installed IRAF on other platforms.

The PC-IRAF release should install and run on any recent Linux, FreeBSD, or Solaris x86 system on Intel-based hardware, and under Mac OS X for PowerPC-based systems from Apple. Linux contiues to evolve rapidly however, and there are a number of different Linux distributions out there - RedHat, SusE, Slackware, Debian, and others. These are all basically the same thing and they all share the same Linux kernel, but they differ in many details regarding what non-Kernel software is included and how it is configured. FreeBSD on the other hand is controlled by a core group of programmers and there are no variant distributions which may conflict. Solaris is even more tightly controlled by a proprietary vendor and is the same system available on Sun workstation hardware. Mac OS X is the newest of the systems and is changing *very* rapidly as it matures to a full-fledged system. Users should consult the archive distribution README file and the IRAF Project web pages for new information as we learn more about the interaction of OS X and IRAF and the system evolves.

The V2.12 PC-IRAF distribution contains a single IRAF system (directory tree) which will run on a number of different operating systems, where each host system is represented as a different binary architecture. As of this writing the PC-IRAF V2.12 system contains binaries for Slackware Linux, Red Hat Linux, SuSE Linux, Solaris x86, Mac OS X, and FreeBSD. Other architectures will be added in a future release, e.g. LinuxPPC for the PowerPC Macintosh running Linux. Our expectation is that the statically linked linux binaries will *run* on any recent Linux system, however there can be problems when compiling programs on a Linux system unless the IRAF libraries have been tailored for and compiled on the target system, due to numerous minor differences in the different flavors of Linux.

Before installing PC-IRAF, one must 1) obtain an appropriate PC-IRAF distribution, e.g., from the IRAF network archive on `iraf.noao.edu` (or by ordering a CD-ROM distribution from NOAO), 2) select the server or node on which the system is to be installed and arrange for sufficient disk space to hold the system, and 3) set aside sufficient time to do the installation. If these directions are followed carefully and mistakes are avoided the basic installation should only take a few minutes on a fast PC. Additional time may be required to customize the system, to configure the local tape drives and other devices, set up IRAF networking, and so on.

†Operated by the Association of Universities for Research in Astronomy, Inc. under cooperative agreement with the National Science Foundation.

The amount of disk space required to install a full IRAF system depends upon the system configuration, including the number of layered packages installed on top of the core IRAF system. The main system, including both the core IRAF system and NOAO package sources, requires from 32 to over 200 Mb depending on what architectures or other options are installed, whether the system has been stripped of source, etc. See the PC-IRAF README for more information on installation options.

2. Installing PC-IRAF

Although the details of how PC-IRAF is installed or updated depend upon the type of distribution and the desired local system configuration, the basic procedure is always the same. First one obtains the distribution files, then one follows the procedure outlined below to install the system. Most of these steps should be performed while logged in as the 'iraf' user; superuser permission is required to initially create the 'iraf' user account and in the final stages of the installation to run the *install* script.

System managers familiar with the installation of typical UNIX programs should beware that IRAF, being a large system in its own right and not a UNIX program, does not always follow to the usual conventions. It is wise to read and adhere to the installation instructions to avoid problems.

```
# Prepare the root IRAF directory.
if new installation
    create 'iraf' user account
else if updating an old installation
    save locally modified files; delete old system

# Install the files.
login as 'iraf'
unpack the core system distribution
configure the BIN directories

# Merge local revisions into new system.
if updating an old installation
    merge locally modified files back into new system

run the iraf install script to complete the installation
checkout the new system
```

If problems should arise during the installation help is available via the IRAF HOTLINE (520-318-8160), or by sending email to `iraf@noao.edu`.

2.1. Prepare the root IRAF directory

2.1.1. If updating an existing IRAF installation...

If you are updating an existing IRAF installation then you will be replacing IRAF by the new version, and IRAF should already have an account and root directory on the desired host system. You should save any locally modified files and delete the old system, e.g., login as IRAF and enter something like the following.

```
% cd $iraf†
% tar -cf /scr0/oiraf.tar local dev unix/hlib
% /bin/rm -rf *
```

† $iraf symbolizes the UNIX pathname of the root IRAF directory. If no "iraf" environment variable is defined just supply the actual pathname.

There are many possible variations on this, e.g., you could use *mv* to move the above directories to someplace outside the main IRAF directory tree. Although it is probably simplest and safest to save entire directories as in the example, in practice only a few files are likely to have been modified. These are the following.

```
dev/graphcap
dev/hosts
dev/imtoolrc
dev/tapecap.*
dev/termcap
hlib/extern.pkg
hlib/login.cl
hlib/zzsetenv.def
local/.login
local/.cshrc
```

Note: If updating an existing V2.11 installation remember to save *all* the tapecap files in the iraf$dev directory. Some of these files may have been created as symbolic links to the more generalized `tapecap.solaris` and/or `tapecap.sunos` files; to preserve the links it's preferrable to use `tar` over `cp` or `mv` when creating the old system backup files.

Once the old system (or systems) has been deleted you are ready to install the new one, as described in §2.2. It is important to delete the old system first to avoid creating junk files or directories when the new system is installed (due to file or directory name changes or deletions). Once the new system has been restored to disk, do *not* merely restore the files saved above, as you will need to carefully merge local changes into the versions of the files supplied with the new IRAF release (more on this later).

2.1.2. If installing IRAF for the first time...

2.1.2.1. Create the 'iraf' User Account

If you are installing IRAF for the first time then the first step is to set up a new account for the user '`iraf`' (see below for special instructions regarding the iraf user account on Mac OS X systems). This is necessary for IRAF system management, which should always be done from the IRAF account. The IRAF account has custom login files which set up a standard UNIX environment for IRAF system management. For that reason it's **critical** that 1) the iraf account be created to use a C-shell as it's login shell, 2) the account be created *before* the IRAF core system files are unpacked so that the preconfigured IRAF login files overwrite any system skeleton files that are created when the 'iraf' account is first setup, and 3) the iraf login directory be defined as $iraf/local (and not just $iraf as one might expect). Missing any one of these points could cause problems in properly setting up the IRAF environment needed for installation and maintainence.

Assuming the default IRAF directory structure (see below) the `/etc/passwd` file entry for the IRAF account would be something like the following:

```
iraf:*:312:12:IRAF system login:/iraf/iraf/local:/bin/csh
```

Having an IRAF account provides a convenient place (the IRAF system manager's login directory) to keep scratch files created during system configuration, and since it has a standard environment it not only simplifies routine maintanence but provides a baseline for performance when problems are encountered.

2.1.2.2. The 'iraf' User Account for OS X Systems

The need for an 'iraf' user still exists under OS X however the system is not a generally configurable in this regard. Specifically, one would create the iraf account from the control panel but the login directory for the account will be /Users/iraf and changing it is not trivial. For this reason the iraf install script will create links to the environment setup files in the IRAF distribution so the iraf account under OS X will work in the same manner. As with other systems, the iraf user should be created prior to unpacking and installing the system.

2.1.2.3. Create the IRAF Directory Tree

The location of the IRAF root directory is arbitrary. Our practice here is to locate the software in a system file storage area separate from the rest of the operating system files (so they are not lost entirely during OS upgrades), and then use a symbolic link such as `/iraf` to point to the actual root directory. This makes life simpler if IRAF is NFS mounted on several machines and it is later necessary to move the IRAF files, or when NFS mount points may prevent a common path from being established. Try to keep the path to the physical IRAF root directory short to simplify the filename expansion when IRAF is run.

A typical IRAF installation consists of the main IRAF release, a number of BIN directories (the IRAF and NOAO package binaries), and additional directories for layered software such as STSDAS, PROS, display tools like X11IRAF, and so on. If sufficient disk space is available to keep everything in one area the following directory structure is recommended.

```
/iraf/iraf                        # iraf root directory ($iraf)
/iraf/iraf/local                  # iraf login directory (~iraf)
/iraf/irafbin                     # iraf BIN directories
/iraf/irafbin/bin.redhat          # RedHat binaries iraf core system
/iraf/irafbin/noao.bin.redhat     # RedHat binaries NOAO package
/iraf/irafbin/bin.macosx          # OS X binaries iraf core system
/iraf/irafbin/noao.bin.macosx     # OS X binaries NOAO package
       :                         :      :      :
/iraf/x11iraf                     # X11IRAF (i.e. XGterm/XImtool) directory
/iraf/extern                      # external package directory
/iraf/extern/stsdas               # external layered package
/iraf/extern/mscred               # external layered package
       :                         :      :      :
    (etc.)
```

For the purposes of this example we assume that the IRAF files are stored in `/iraf`; as we say this might be a link and the actual directory is arbitrary. Given this directory the IRAF root `$iraf` would be "/iraf/iraf/" and the login directory for the IRAF account would be /iraf/iraf/local. The e.g. RedHat Linux binaries for the core IRAF system would be in /iraf/irafbin/bin.redhat, with a link $iraf/bin.redhat pointing to this directory (more on this later). Keeping external packages and add-on software out of the main IRAF tree simplifies later updates to either the core system or the add-on, mixing them in the IRAF tree is discouraged.

2.2. Install the files

2.2.1. Installing from a network distribution

PC-IRAF is available over the network via anonymous ftp from the node `iraf.noao.edu` (140.252.1.1), in the subdirectory `iraf/v`*nnn*`/PCIX`, where "`v`*nnn*" is the IRAF version number, e.g., subdirectory `iraf/v212/PCIX` for V2.12 PC-IRAF.

If IRAF is being installed from a network distribution or CD-ROM all the architecture independent IRAF files for both the core IRAF system and the NOAO packages will be in the distribution file `as.pcix.gen` for all supported platforms. This is the *only* file in the distribution which is common to all systems. The distributions files are stored in the archive as a large compressed tar archive. For sites with slow network connections it may be easier to download the distribution as a split-set of files and reconstruct them locally later. In this case the distribution "file" is stored in the archive *"splits"* directory as a subdirectory wherein the large distribution file has been split into a number of smaller pieces, e.g.,

```
% ls as.pcix.gen
CHECKSUMS            as.pcix.gen.gz.12    as.pcix.gen.gz.26
FILES.Z              as.pcix.gen.gz.13    as.pcix.gen.gz.27
as.pcix.gen.gz.00    as.pcix.gen.gz.14    as.pcix.gen.gz.28
as.pcix.gen.gz.01    as.pcix.gen.gz.15    as.pcix.gen.gz.29
as.pcix.gen.gz.02    as.pcix.gen.gz.16    as.pcix.gen.gz.30
       (etc.)
```

Assuming that a single-file installation was chosen one would only need to download the `as.pcix.gen.gz` file from the archive. The IRAF source tree would then be restored with the command (remember to login as the 'iraf' user first!):

```
% whoami
iraf
% cd $iraf
% cat /path/as.pcix.gen.gz | zcat | tar -xpf -
```

On the other hand, if a split-file distribution was downloaded, we assume that the directory `as.pcix.gen` as shown above has been recreated somewhere on the local machine onto which IRAF is to be installed. We can then restore the main IRAF source tree as follows (remember to login as the 'iraf' user first!):

```
% whoami
iraf
% cd $iraf
% cat /path/as.pcix.gen/as.* | zcat | tar -xpf -
```

After the above finishes the root IRAF directory should appears as follows (this is for V2.12).

```
HS.PCIX.GEN  bin.generic   bin.macosx  bin.suse  lib    mkpkg  sys
bin          bin.linux     bin.redhat  dev       local  noao   tags
bin.freebsd  bin.linuxppc  bin.sunos   doc       math   pkg    unix
```

The files `bin.freebsd`, `bin.linux`, `bin.linuxppc`, `bin.macosx`, `bin.redhat`, `bin.sunos`, and `bin.suse` are links to the supported IRAF BIN directories (for binary executables), which probably do not exist yet. The `bin` file is a symbolic link to the empty `bin.generic` directory indicating that the system is configured "generically" (i.e. not for development on a particular architecture), this link should **NOT** be changed by hand as it is needed by the system. Binaries for the various architectures will be found automatically by the system at runtime. Configuring the BIN directories is discussed in section §2.2.3.

2.2.2. Installing from CD-ROM

At the time of this writing the CD-ROM distributions of IRAF are snapshots of selected areas of the network ftp archive. Installing from CD is very much like a network distribution except that the CD may be mounted as a local filesystem and accessed directly when unpacking the files. There is no real reason to copy the files to a local disk before installation, simply use */mount/*`iraf/v`*nnn*`/PCIX` as the path prefix when accessing the `AS` and BIN distribution files. Details on how to unpack the distribution file are covered in §2.2.1 above.

2.2.3. Configuring the BIN directories

In IRAF all the files specific to any particular architecture are contained in a single directory called the BIN, or "binary", directory. To run IRAF you must install not only the `AS` (all-sources) directory tree, but the BIN directory for each architecture. The IRAF core system and the NOAO packages have separate BIN directories, commonly referred to as the `IB` (IRAF binary) and `NB` (NOAO binary) distributions respectively.

The BIN directories for the IRAF core system or a layered package (such as NOAO) are located, logically or physically, in the root directory of the IRAF core system or layered package. Every layered package has its own set of BIN directories. In the distributed V2.12 system you will find the following BIN files (directories or symbolic links) at the IRAF root.

```
link         bin -> bin.generic
directory    bin.generic
link         bin.freebsd -> ../irafbin/bin.freebsd
link         bin.linux -> ../irafbin/bin.linux
link         bin.linuxppc -> ../irafbin/bin.linuxppc
link         bin.macosx -> ../irafbin/bin.macosx
link         bin.redhat -> ../irafbin/bin.redhat
link         bin.sunos -> ../irafbin/bin.sunos
link         bin.suse -> ../irafbin/bin.suse
```

If the IRAF directory structure is set up as described in §2.1.2, with $iraf located at /iraf/iraf and the BIN directories stored in /iraf/irafbin, then these links will not have to be modified. If a different directory structure is used you will have to modify the links accordingly.

The *bin* link and the *bin.generic* directory are required for the correct operation of the IRAF system software (*mkpkg*) and are maintained automatically by the IRAF software management utilities. *Under no circumstances should "bin" or "bin.generic" be modified or deleted.* It is a very common error to manually delete the bin link and manually set it to e.g. bin.redhat or some other architecture. The bin.redhat link can be modified as desired but bin and bin.generic should be left alone.

Assume that the bin.redhat directory has been created somewhere (e.g. in the /iraf/irafbin directory) and that the `ib.rhux.sun` distribution files for the core IRAF system solaris binaries have been downloaded from the network archive or will be taken from the CD-ROM. We can then restore the RedHat Linux binaries with the following commands:

```
% cd $iraf/bin.redhat
% cat /path/ib.rhux.x86.gz | zcat | tar -xpf -
or if you use the split distribution
% cat /path/ib.rhux.x86/ib.* | zcat | tar -xpf -
```

Similarly, to restore the NOAO package solaris binaries:

```
% cd $iraf/noao/bin.redhat
% cat /path/nb.rhux.x86.gz | zcat | tar -xpf -
or if you use the split distribution
% cat /path/nb.rhux.x86/nb.* | zcat | tar -xpf -
```

A similar sequence can be used to restore the SuSE or other binaries for both the core system and the NOAO package. Note that you must first `cd` to the proper BIN directory for *each* distribution before unpacking the files. Verify that this command worked as expected and you did indeed end up in the proper directory before attempting to unpack the files. Another common error you should avoid is to unpack both the IB and NB distributions in a single directory, this will lead to a `"cannot open connected subprocess"` error inside IRAF when trying to access core iraf and/or NOAO package tasks.

2.3. Merge local revisions back into the new system

If this is a new IRAF installation this step can be skipped, but you will still need to do the final configuration as directed in the Site Manager's Guide. Otherwise, once the new system has been restored to disk any local revisions made to the previous IRAF installation should be merged back into the new system. See §2.1.1 for a list of the files most likely to be affected and which should have been backed-up prior to the upgrade. When propagating revisions made to these files, be sure to not just simply replace the entire file with your saved version, as the version of the file in the new release of IRAF will often contain important additions or changes which must be preserved. It is best to merge your revisions into the version of the file which comes with the new IRAF.† This task will be easier if the revisions have been localized as far as possible, e.g., keep all `graphcap` additions together at the head of the file, so that they may merely be transferred to the new file with the editor. The task of propagating revisions will also be much easier if detailed notes have been kept of all revisions made since the the last release was installed.

2.4. Run the INSTALL Script

Once all of the IRAF files have been restored to disk the PC-IRAF installation script must be run to complete the system installation. The install script modifies the system as necessary to reflect the new root directory and new default image storage and local bin directories, checks the mode and ownership of a number of files, installs a small set of IRAF commands in UNIX, and so on.

To make a trial run of the install script, enter the following commands:

```
% setenv iraf /path/iraf/
% cd $iraf/unix/hlib
% source irafuser.csh
% ./install -n
% sudo ./install -n          (may be needed for Mac OS X systems)
```

and answer the questions (don't forget the trailing '/' in the "setenv iraf"). The "-n" argument tells install to go through the motions without actually doing anything, so that one can see what will be done before committing to it. ††

Installing IRAF requires a few changes to be made to system directories outside the IRAF directory tree: two fifo device entries are made in /dev, a symbolic link "iraf.h" is created in /usr/include, a number of links (cl, mkiraf, etc.) are made in /usr/local/bin or some similar directory which most users can be expected to have in their search path, the tape allocation task alloc.e is made suid root (there are no known security loopholes, although we cannot make any guarantees), a symbolic link imtoolrc is created in /usr/local/lib, and lastly the iraf path is edited in to several iraf system files (so it is vital this path be correct for your system).

Following one or more trial "no execute" ("-n") runs to see what the install script will do, the install script should be run without the "-n" to complete the installation. This must be done by the superuser as root permission is required to carry out the necessary additions to UNIX. It is essential for the superuser account to also have the proper iraf environment defined before the install script is run, both the $iraf root path and the variables defined by sourcing the irafuser.csh file.

It is necessary to run the install script separately on each node from which IRAF will be used. If a single version of IRAF is installed on a server and NFS mounted on one or more clients, the install script must be run first on the server and then on *each* client (when installing on a client there will be warnings about insufficient permission to make changes to files on the NFS mounted partitions, which can be ignored). To install IRAF on a diskless client it may be necessary to run the install script *on the server* to do the install for the client, since the client's /usr/include and /dev directories may only be writable by root on the server. On some systems /usr is mounted read-only, and must be unmounted and remounted read-write before doing the installation to allow an entry to be made in /usr/include. Once the installation is complete the default mount access mode may be restored.

†The UNIX utility *diff* is useful for comparing files to see what has changed.

†† These commands assume you are using a C-shell (or equivalent such as 'tcsh'), if not then you can start one with the "csh" or "tcsh" commands before beginning. Commands such as 'source' and 'setenv' may not be understood by other shells and the IRAF environment will not be properly defined.

In "no-execute" mode (i.e. with the "-n" flag) the output will look identical to the standard execution however no files are actually changed in the system.

2.4.1. System Settings

The installation begins by first querying the user for a number of system settings such as the iraf root directory, an image pixel storage directory, and where to put the commands in the system. The meaning of these fields is described below but the exchange with the install script will be something like the following (this example is for a RedHat Linux server):

```
% ./install -n    ††
                            IRAF V2.12 System Installation
                            ==============================

     Welcome to the IRAF  installation script.  This  script  will
first prompt you for several needed path  names.  The system  will
be verified for proper structure before the actual install begins,
all error must must be  corrected  before you  will be  allowed to
continue. Recommendations for fixing problems will be  made but no
corrective action will be taken directly. Once properly installed,
you will be allowed to do some minimal configuration.

For each prompt: hit <CR> to accept the default value, 'q' to quit
or 'help' or '?' to print an explanation of the prompt.

==================================================================
==================== Query for System Settings ===================
==================================================================

New iraf root directory (/iraf/iraf):
Default root image storage directory (/iraf/imdirs):
Local unix commands directory (/usr/local/bin):
```

The "*iraf root directory*" is the value of **$iraf** (minus the trailing '/'in this case). The "*root image storage directory*"† is the default place to put image pixel data for users using the default *OIF* IRAF image format, FITS image users do not make use of this directory but it should still be created. The value entered should be the path to a publicly-writeable directory of a designated data or scratch disk on your system; this directory and any user subdirectories will be created as needed by the install script. Lastly, the "*local unix commands directory*" is where the UNIX callable IRAF startup commands (e.g. '*cl*', '*mkiraf*, etc) will be defined. This should be a UNIX directory which is in the default path of anyone who might want to use IRAF; /usr/local/bin or /opt/local/bin are the most common values.

2.4.2. System Verification

Once all the needed information has been entered, the script then echoes the system settings it's been given (or determined automatically about the machine), and performs a number of integrity checks to be sure the path definitions are appropriate, the IRAF directory tree structure is correct, and the BIN directories are populated correctly, etc. Should any of these tests fail, instructions for correcting the error or determining more about the problem will be printed, however the script will take no corrective action itself. All tests must be passed before the

† The program may prompt with /tmp if it cannot find any likely looking data storage areas on your system, but /tmp is not a good place to put image data as the contents are normally deleted whenever the system reboots.

†† **NOTE FOR MAC OS X USERS: It may be necessary to execute this command, and the final no-op install, using the** *sudo* command. The root account is usually not active under OS X so the administrative user should run the install script with *sudo* for a proper installation.

installation will be allowed to continue past this stage. For example, the above install session might continue as follows:

```
================================================================
====================== Verifying System Settings ======================
================================================================

Hostname     = vmware                OS version   = Linux 2.4.7-10
Architecture = redhat                HSI arch     = redhat
New iraf root = /iraf/iraf           Old iraf root = /iraf/iraf
New imdir    = /iraf/imdirs          Old imdir    = /iraf/imdirs
Local bin dir = /usr/local/bin

Checking definition of iraf root directory ...          [  OK  ]
Checking iraf root and imdir directory ...              [  OK  ]
Checking iraf directory write permissions ...           [  OK  ]
Checking for iraf user account ...                      [  OK  ]
    Checking iraf user login directory ...              [  OK  ]
    Checking iraf user account shell ...                [  OK  ]
Checking file ownerships ...                            [  OK  ]
Checking file permissions ...                           [  OK  ]
Checking proper iraf tree structure in /iraf ...
    Checking for 'iraf' subdir ...                      [  OK  ]
    Checking for 'irafbin' subdir ...
        Checking for fallback tree structure ...        [  OK  ]
    Checking for 'irafbin/bin.ssun' subdir ...          [  OK  ]
    Checking for 'irafbin/noao.bin.ssun' subdir ...     [  OK  ]
Checking Core system binary directory ...               [  OK  ]
Checking NOAO package binary directory ...              [  OK  ]
Checking that local bin directory exists ...            [  OK  ]
Checking for existing commands directory...             [ WARN]

   ***  IRAF commands were found in the directory:
   ***
   ***          /opt/local/bin
   ***
   ***      These commands may conflict with the commands now
   ***  being installed in: '/usr/local/bin'
   ***

Do you want to delete commands in the old directory?  (yes): no

Proceed with installation?  (yes):
```

In this example the final test fails when commands from a previous installation are found which may conflict with the definitions for the current install. Here, we replied '*no*' to simplify the output, however it's recommended that you reply '*yes*' to such a question so as to minimize confusion amongst users as to which iraf command is really being used, to clean out invalid command links, etc.

2.4.3. System Installation

At this point we can reply '*yes*' to the prompt to proceed with the actual installation. Unlike earlier versions of the install script, this new version simplifies the output to print the steps being taken and a singular report of success or failure. Error messages from any stage will output any specifics about what failed along with a recommended fix. Again no corrective action will be taken by the install script directly. For example, the output produced from a successful installation will look something like:

```
================================================================
======================= Begin Installation =======================
================================================================
```

Editing Paths

```
Editing iraf user .login/.cshrc paths ...               [ OK ]
Editing iraf/imdir paths into system files ...          [ OK ]
```

Checking File Permissions

```
Checking iraf file permissions ...                      [ OK ]
Creating root imdir at /iraf/imdirs ...                 [ OK ]
Reset /tmp sticky bit setting ...                       [ OK ]
Setting tape device permissions ...                     [ OK ]
Checking alloc.e permissions ...                        [ OK ]
```

Creating File Links

```
Checking for /iraf symlink ...                          [ OK ]
Creating <iraf.h> symlink ...                           [ OK ]
Creating iraf command links in local bin dir ...        [ OK ]
Marking system update time hlib$utime ...               [ OK ]
```

Creating Graphics Device Files

```
Creating /dev/imt1 fifo pipes for image display ...     [ OK ]
Creating /dev/imt fifo pipes link ...                   [ OK ]
Creating /usr/local/lib/imtoolrc link ...               [ OK ]
Checking if termcap file contains an XGterm entry ...   [ OK ]
Checking graphcap file for XGterm/imtool entries ...    [ OK ]
```

Not all errors found at this stage are considered fatal, however the installer should examine each message and take whatever action is suggested or required. The install script may be re-run later with the same inputs to fix any missed steps.

2.4.4. System Post-Install Configuration

A detailed explanation of the various post-install configuration setup can be found in the **Unix IRAF Site Manager's Guide** however the install script is able to do some of the more common setup as part of the installation on each node. This includes things which may be needed for each iraf client node such as creating a local tapecap file link or entry for the machine in the local IRAF network, or things which may only be needed on the server such as stripping the system of sources.

Once the installation is complete you will be prompted with whether to continue to the post-install configuration stage. If you say *no* at this stage the system is fully functional however some features may still need to be configured by hand. Replying *yes* at this stage will allow you to continue where you may choose to configure one or all of the available features.

2.4.4.1. IRAF Networking Setup

IRAF networking allows you to access remote devices (image displays, tape drives, etc), images, or files from within an IRAF session. All nodes in the *"local IRAF network"*, including the local node, must be defined in the **dev$hosts** file for this to be enabled. A *dev$hosts* entry consists simply of the node name and the local path to an *irafks.e* kernel server binary for that system. The install script is able to determine this information automatically and can add the local machine to the network when the install script is run (e.g on each of the client machines getting their IRAF from a central server).

The networking setup looks something like the following:

IRAF Networking Config

```
    IRAF Networking can be used to access a remote image, tape
device, display server, or other network service. It's config-
uration is not a requirement for normal IRAF operations and it
can be updated at any time by editing the IRAF dev$hosts file
with new entries.

    In this stage we will add an entry for the current platform
to the hosts file.  In a local network installation this script
should be run on each system to add a networking entry as well
as to install other system files needed by IRAF.
```

Configure IRAF Networking on this machine? `(yes):` **(1)**

```
Recommended dev$hosts file entry used for this machine:

    vmware        : ozzie!/iraf/iraf/bin.redhat/irafks.e
```

Proceed with this entry? `(yes):` **(2)**
Do you want to sort the hosts file by node name? `(yes):` **(3)**

Checking that iraf networking is properly enabled... [**OK**]

In prompt **(1)** you can reply *no* to skip the setup, hitting the <*CR*> or *yes* will continue with the rest of the output. The script next tries to determine the appropriate entry for this machine and asks in **(2)** whether you would like to accept this entry. Here we accept the default, but replying *no* at this stage will allow you to manually edit the *dev$hosts* file and insert your entry, or you will be asked if you want to skip the setup entirely. If an entry for the machine already exists in the file you will be shown the recommended entry and the existing one and asked if you want to overwrite the entry in the file. Once the entry has been added, you are asked in **(3)** if you would like to sort the hosts file alphabetically by node name (the default is to add the local node at the end of the file). For networks with many nodes this simplifies management of the hosts file since it's easier to locate a particular machine and spot duplicates. Lastly, a check is performed to see whether IRAF networking has been properly configured for this machine. If this last test fails you will be asked whether you want to try again with a different entry or move on.

2.4.4.2. Tapecap Setup

Beginning with IRAF V2.11 the tape configuration file first used will be a "`dev$tapecap.`<*node*>" file on the local machine, where <*node*> is the value returned by the system '*hostname*' command and the nodename used is enabling IRAF networking above. If that file is not found, the standard `dev$tapecap` file will be used. On heterogeneous systems such as Sun and PC-IRAF it will be necessary to have a platform-dependent tapecap file for each supported OS which can be used by machines in the local configuration as a template. At this stage in the post-install configuration the script is able to create a node-specific symlink pointing to the appropriate tapecap template, it can also move one of the templates to be the default `dev$tapecap` filename.

The exchange with the install script will look something like:

Tapecap Device File Config

```
    By default IRAF will search for a dev$tapecap.<node> file
(where <node> is the system name) when looking for a tape config-
uration file.  Platforms such as PC-IRAF and Sun/IRAF support
multiple OS versions and so the proper template file must be used.
This configuration will allow you to setup a default tapecap for
this system, it may be skipped if this machine has no tape drive
attached.
```

Create a default tapecap file? `(yes):` **(1)**

```
Creating default file 'tapecap.vmware' from tapecap.linux...

    In the event a dev$tapecap.<node> file is not found on this
system IRAF will fallback to use just dev$tapecap.  In cases where
the node name changes, this installation is shared with another
machine or in a local network, or any case where a tapecap.<node>
is not found, the dev$tapecap file will be the default tapecap
used for all IRAF systems.
```

Do you wish to create a default dev$tapecap link? `(yes):` **(2)**

```
Tapecap symlink 'tapecap' exists but is invalid....
Deleting invalid link....
Creating default dev$tapecap link to dev$tapecap.linux...
```

In prompt **(1)** you are asked if you wish to create a default tapecap file appropriate for this machine. Replying *yes* (or accepting the default with a <cr>) will create a symlink called `tapecap.`<*node*> pointing to the distributed template file for the current platform. Otherwise, no `tapecap.`<*node*> link will be created and the default tapecap file must be properly configured for tape access on this machine. Prompt **(2)** asks if you wish to create a default `dev$tapecap` link which is correct for the current machine (i.e. a link to the platform-dependent version of the file). If you reply *yes* any existing link or file will be removed and the correct link created, otherwise the default tapecap is left along. Note that you will only want to create the default tapecap when running the install script on the server node, or at least only on one node in your system.

See the **UNIX IRAF Site Manager's Guide** for more details on tapecap configuration. This stage of the configuration is skipped on platforms which require only a single tapecap file.

2.4.4.3. Deleting Unused HSI Binaries

IRAF systems which support multiple operating systems (e.g. Sun and PC-IRAF) will contain a full complement of HSI binaries (i.e. the BIN directories in the *iraf$unix* subdirectory) even if only one set of architecture binaries is installed. On some systems these unused HSI binaries can add up to a major portion of the disk space required for the core IRAF system files. This stage of the post-install configuration allows you to remove these binaries to recover disk space.

It is assumed that all IRAF distribution files are unpacked and in the proper directory. The script first determines which IRAF binaries are installed and compares those to the HSI binaries, any HSI architecture without a corresponding set of IRAF binaries is said to be unneeded and the list of these binaries (and their size) will be presented to the user for deletion. It is **strongly recommended** that you skip this step if you anticipate that at a later time you may install another set of IRAF binaries for a different architecture.

The exchange with the install script will look something like the following (assuming only the RedHat Linux binaries are installed):

Delete Unneeded HSI Binaries
--

```
    The following bin directories in the iraf$unix directories were
found to be unused on this machine:

            ( 5040 Kb)  /iraf/iraf/unix/bin.freebsd
            ( 9555 Kb)  /iraf/iraf/unix/bin.linux
            (  314 Kb)  /iraf/iraf/unix/bin.linuxppc
            ( 3258 Kb)  /iraf/iraf/unix/bin.macosx
            ( 2403 Kb)  /iraf/iraf/unix/bin.sunos
            ( 7719 Kb)  /iraf/iraf/unix/bin.suse
```

```
The contents of these directories may be safely deleted to reclaim
about 28 Mb of disk space without affecting the IRAF runtime system.

Do you wish to delete these unused HSI binaries?  (yes):

Delete HSI binaries in bin.freebsd ...                      [  OK  ]
Delete HSI binaries in bin.linuxppc ...                     [  OK  ]
                  :
                  :
```

The amount of disk space recovered depends on the system installed, the 28Mb saved here is typical.

2.4.4.4. Strip System Sources

The final stage in the post-install configuration is another space-saving step which was often overlooked when left as an exercise in the Site Manager's Guide but which can be important on some systems. The AS distribution file contains the source files for each task in IRAF as well as the various runtime files needed. It is safe to delete the program sources provided:

```
  1)  You do not plan to do any development of the IRAF code itself.
  2)  You do not wish to compile IRAF patches locally and will only
      be installing prebuilt binaries (external packages and local
      tasks may still be compiled on a stripped system).
```

The default is to leave the sources in the system since ar times it will be necessary for site support to supply a code change to fix a particular bug, or recommend a piece of code for inspection to clarify a problem. Workarounds exist for these cases and if disk usage is a concern on a particular system stripping the sources will not hurt IRAF in any way.

The exchange with the install script will look something like:

```
                         Strip IRAF System Sources
                         -------------------------

    Source code for all IRAF tasks and interfaces is included with
this installation, but is strictly only required if you plan to
develop this code.  The sources may be deleted from the system with-
out affecting the runtime environment (including help pages, compil-
ation of external packages or local task development) allowing you
to reclaim 50-60Mb of disk space for the system.  Stripping sources
is recommended for systems very short on space, leaving it on the sys-
tem will allow IRAF site support to send code fixes and compilation
instructions as needed to fix problems which have no other workaround.

Do you wish to strip the system of sources?  (no): yes

Stripping core system sources ...                           [  OK  ]
Stripping NOAO package sources ...                          [  OK  ]
```

Note that all external packages (STSDAS, TABLES, MSCRED, etc) can similarly be stripped of sources using something like "mkpkg -p <*pkg*> strip".

2.4.5. Post-Install Verification

The post-install verification stage of the install script currently checks two things: the first is that there is a suitable image display server (e.g. XImtool, DS9, SAOtng/SAOimg) available on the machine, and the second is that there is a suitable graphics terminal (e.g. XGterm or Xterm) on the system. It's important to note that aside from an Xterm window which is standard on most systems, display servers and XGterm must be installed separately from the rest of IRAF.

XImtool and XGterm are distributed as part of the X11IRAF package from the IRAF archive (iraf.noao.edu or it's mirrors) in the /iraf/x11iraf directory. There are separate README and installation instructions for these tools which should be consulted. The choice of display server is sometimes a matter of personal preference, alternate servers are available from the IRAF archive "/contrib" directory.

3. System Checkout

The basic IRAF system should be usable once the files have been restored to disk, the binaries have been configured or generated, and the install script has been run. To verify that the basic system comes up and runs successfully, login as iraf and startup the CL (IRAF command language) from the iraf account. You should be able to login as IRAF and type "cl" to start IRAF, using the login files which come with the distributed system.

```
% login iraf
% cl
```

To more thoroughly test the installation it is a good idea to test IRAF from a user account. To do this you login to a user account and run the *mkiraf* task to set up the IRAF login files. This will create or initialize the user's `uparm` (user parameter) directory, and create a new `login.cl` file. It may also be desirable to edit the user's `.login` file to modify the way the environment variable `IRAFARCH` is defined. This variable, required for software development but optional for merely using IRAF, must be set to the name of the desired machine architecture, e.g., 'redhat' for RedHat Linux or

```
% mkiraf
Initialize uparm? (y|n): y
Terminal types: xgterm,xterm,gterm,vt640,vt100,etc."
Enter terminal type: xgtermc
A new LOGIN.CL file has been created in the current directory.
You may wish to review and edit this file to change the defaults.
```

The *cl* command should now start up the CL, which will clear the screen and print out a startup message. The standard test procedure included in Volume 1A of the *IRAF User Handbook* should be run to verify the installation, this document is also available as

```
ftp://iraf.noao.edu/iraf/docs/testproc2.ps.Z
```

3.1. Frequently Asked Questions

Many common installation problems are answered in our Frequently Asked Questions file available from either

```
ftp://iraf.noao.edu/iraf/FAQ/FAQ
http://iraf.noao.edu/iraf/faq
```

or from the various iraf archive mirror sites. The WWW version of the FAQ is searchable by keyword.

Appendix A. A Complete Example Using Single-File Distributions

Assume we are installing IRAF for the first time on our spiffy new PC. The IRAF directories will be located at /u1/iraf using a symbolic link /iraf to point to this location. We will configure only the linux binaries, locating the BIN directories in the directory /iraf/irafbin (installing other architectures is just a matter of unpacking the binaries and creating the links). The local user commands will be placed in /usr/local/bin. We will be installing from a network distribution with the distribution files located in /d0.

The first step is for the superuser to create an account for the fictitious user '`iraf`', with home directory /iraf/iraf/local and shell /bin/csh. The /u1/iraf directory is created owned by IRAF, and pointed to by the link /iraf. We then login as IRAF (a warning message will be printed since there is no login directory) and proceed as follows. **Note:** The example here assumes we are unpacking the general *LNUX* distribution for Linux systems such as Slackware or Debian, you will need to adjust the filenames depending on the platform you are installing.

```
% whoami
iraf
%
% setenv iraf /iraf/iraf/                  # set root directory
% mkdir /iraf/iraf
%
% cd $iraf                                 # unpack main IRAF distribution
% cat /d0/as.pcix.gen.gz | zcat | tar -xpf -
%
% cd /iraf                                 # create BIN directories
% mkdir irafbin
% mkdir irafbin/bin.linux
% mkdir irafbin/noao.bin.linux
%
% cd $iraf/bin.linux                       # unpack core bin.linux
% cat /d0/ib.lnux.x86.gz | zcat | tar -xpf -
%
% cd $iraf/noao/bin.linux                  # unpack NOAO bin.linux
% cat /d0/nb.lnux.x86.gz | zcat | tar -xpf -
%
% cd $iraf/unix/hlib                       # run the INSTALL script
% source irafuser.csh
% ./install -n
% su
# ./install
# exit
%
% cd
% source .login                            # read new .login
% rehash                                   # pick up new iraf commands
% cl                                       # verify that the CL runs
```

This will fully install IRAF on a server or a standalone system. If this version of IRAF will be accessed via NFS by client nodes then the IRAF install script must be run on each client node as well. Installing IRAF does not allow one to access local tape drives, printers, and so on. Refer to the *PC-IRAF Site Manager's Guide* for information on how to configure IRAF for the local site.

Appendix B. A Complete Example Using Split Distributions

Assume we are installing IRAF for the first time on our spiffy new PC. The IRAF directories will be located at /u1/iraf using a symbolic link /iraf to point to this location. We will configure only the linux binaries, locating the BIN directories in the directory /iraf/irafbin (installing other architectures is just a matter of unpacking the binaries and creating the links). The local user commands will be placed in /usr/local/bin. We will be installing from a network distribution with the distribution files located in /d0.

The first step is for the superuser to create an account for the fictitious user '`iraf`', with home directory /iraf/iraf/local and shell /bin/csh. The /u1/iraf directory is created owned by IRAF, and pointed to by the link /iraf. We then login as IRAF (a warning message will be printed since there is no login directory) and proceed as follows. **Note:** The example here assumes we are unpacking the general *LNUX* distribution for Linux systems such as Slackware or Debian, you will need to adjust the filenames depending on the platform you are installing.

```
% whoami
iraf
%
% setenv iraf /iraf/iraf/                   # set root directory
% mkdir /iraf/iraf
%
% cd $iraf                                  # unpack main IRAF distribution
% cat /d0/as.pcix.gen/as.* | zcat | tar -xpf -
%
% cd /iraf                                  # create BIN directories
% mkdir irafbin
% mkdir irafbin/bin.linux
% mkdir irafbin/noao.bin.linux
%
% cd $iraf/bin.linux                        # unpack core bin.linux
% cat /d0/ib.lnux.x86/ib.* | zcat | tar -xpf -
%
% cd $iraf/noao/bin.linux                   # unpack NOAO bin.linux
% cat /d0/nb.lnux.x86/nb.* | zcat | tar -xpf -
%
% cd $iraf/unix/hlib                        # run the INSTALL script
% source irafuser.csh
% ./install -n
% su
# ./install
# exit
%
% cd
% source .login                             # read new .login
% rehash                                    # pick up new iraf commands
% cl                                        # verify that the CL runs
```

This will fully install IRAF on a server or a standalone system. If this version of IRAF will be accessed via NFS by client nodes then the IRAF install script must be run on each client node as well. Installing IRAF does not allow one to access local tape drives, printers, and so on. Refer to the *PC-IRAF Site Manager's Guide* for information on how to configure IRAF for the local site.

```
             README FOR PC-IRAF VERSION 2.12
              Updated Fri February 6, 2004
             Current Patch Level V2.12.2-EXPORT


##############################################################################

12-Feb-2004         *** V2.12.2 Patch-2 -- Slackware 4 Compatability Release
06-Feb-2004         *** V2.12.2 Patch-2 Release
13-Aug-2003         *** V2.12.1 RedHat 9 Compatability Release
15-Jul-2002         *** V2.12.1 Patch-1 Release
08-May-2002     *** Initial V2.12 EXPORT Release ***
            (Linux, MacOSX, FreeBSD, and Solaris x86)


IRAF V2.12 is a major update to the system and requires a FULL INSTALLATION
of the package, even if you have an existing V2.11 system.  The Installation
Guide (pciraf.ps.Z) contains a detailed description of the installation
process for both first-time installations and updates to existing systems.
Installation will require downloading the AS, IB, and NB distribution files
from this directory.  The AS distribution is common to all PC-IRAF platforms
and is required for any installation.  There are separate sets of binaries
(the IB and NB distributions) for each PC-IRAF platform and only those
binaries required for a particular platform will need to be downloaded.

PC-IRAF V2.12 supports the following platforms:

    System             Distribution    Add'l Systems
    ------             ------------    -------------
    FreeBSD 4.2 and higher      FBSD
    MacOSX 10.1 and higher      MACX
    RedHat Linux V6.x thru V8.x     RHUX        Mandrake 7.x and 8.x
    Slackware V8.x and V9.x         LNUX/x86        Debian 2.x, all others
    Solaris 7 for Intel         SSOL
    SuSE Linux V6.x thru V7.x       SUSE
    Yellow Dog Linux V3.0       LNUX/ppc


    Compatability Releases
    ----------------------
    Slackware V4.x          LNUX/x86        'slack40' subdir


All versions are supported with a single IRAF distribution, although you need
to install separate binaries for for each platform.

The IRAF distribution does not itself include graphics and image visualization
tools; these are distributed separately.  XGterm is required for any IRAF task
which does vector graphics or which has a GUI.  Any of the XImtool, SAOtng,
SAOimage, and DS9 image display servers may be used for image interaction.
See ftp://iraf.noao.edu/iraf/x11iraf/ for the latest X11IRAF tools
(XGterm/XImtool) and /contrib for additional display servers
(DS9, SAOtng, and SAOimage).

See also the post-distribution notes at the end of this file.  These are
continually updated after the release as any problems are encountered.

##############################################################################
```

```
Contents
--------
```

```
    1. Introduction
    1.1 The PC-IRAF V2.12 Release
    1.2 Who Should Upgrade?
    2. Highlights of this Release
    3. Installing PC-IRAF
    3.1 MacOSX Technical Notes (updated: 5/6/02)
        3.1.1 External Package Support
        3.1.2 The Mac OS X 'iraf' user account
    3.2 Things to Watch Out For (IRAF)
        3.2.1 Special Note to Fedora and RHEL Users
        3.2.2 Special Note to Linux IMFORT Users
        3.2.3 Linux PPC Support (*** NEW IN V2.12.2 ***)
        3.2.4 Optimization Changes For MAC OS X
    4. X11/GUI Support
    4.1 The GUIAPPS External Package
    4.2 The X11 Desktop
    4.3 Things to Watch Out For (X11IRAF)
        4.3.1 RedHat 8 Special Note
        4.3.2 RedHat 9 Special Note
    5. Magtape Interface
    6. Printer Interface
    7. Patch Instructions
    8. Notes Added Since the Initial Release (updated: 8/13/03)
```

```
-----------------------------------------------------------------------------
1. INTRODUCTION
-----------------------------------------------------------------------------

1.1. THE PC-IRAF V2.12 RELEASE

PC-IRAF V2.12 is a port of IRAF to PC platforms running any of a number of
Linux distributions, FreeBSD, or Solaris on Intel-based systems, or MacOSX
on PowerPC-based systems.  Beginning with IRAF V2.11, a single version of
PC-IRAF installed on a central server can simultaneously support IRAF sessions
running on any combination of PC systems or servers.

    README      This file                       (25 Kb)

    as.pcix.gen.gz  All-Sources (main IRAF distribution)    (31 Mb)
    db.fbsd.x86.gz  Core system debuging libraries for FreeBSD  ( 3 Mb)
    ib.fbsd.x86.gz  Core system binaries for FreeBSD        (18 Mb)
    ib.lnux.x86.gz  Core system binaries for Linux (generic)    (23 Mb)
    ib.macx.ppc.gz  Core system binaries for MacOSX     (20 Mb)
    ib.rhux.x86.gz  Core system binaries for RedHat Linux   (13 Mb)
    ib.ssol.x86.gz  Core system binaries for Solaris x86    (14 Mb)
    ib.suse.x86.gz  Core system binaries for SuSE Linux     (15 Mb)
    ib.lnux.ppc.gz  Core system binaries for Yellow Dog Linux   (17 Mb)
    nb.fbsd.x86.gz  NOAO package binaries for FreeBSD       (22 Mb)
    nb.lnux.x86.gz  NOAO package binaries for Linux (generic)   (32 Mb)
    nb.macx.ppc.gz  NOAO package binaries for MacOSX        (26 Mb)
    nb.rhux.x86.gz  NOAO package binaries for RedHat Linux      (19 Mb)
    nb.ssol.x86.gz  NOAO package binaries for Solaris x86   (20 Mb)
    nb.suse.x86.gz  NOAO package binaries for SuSE Linux    (19 Mb)
    nb.lnux.ppc.gz  NOAO package binaries for Yellow Dog Linux  (22 Mb)
    splits/*        Split distribution files for the above distributions

    pciraf.ps.gz    PC-IRAF Installation Guide (compressed)
    pciraf.ms.gz    PC-IRAF Installation Guide (troff source)
    unixsmg.ps.gz   IRAF V2.12 Site Manager's Guide (compressed)
    unixsmg.ms.gz   IRAF V2.12 Site Manager's Guide (troff source)

    guiapps.readme  GUIAPPS installation instructions       ( 7 kb)
    guiapps.tar.Z  GUI Applications package         (54 Mb)


Beginning with V2.12 each of the AS, IB, and NB distribution files will be
available as a single file for more convenient downloading over
high-bandwidth Internet connections.  For sites with slow or unreliable
network connectivity where the transfer may not finish successfully, the
"splits" directory contains copies of the distribution files which have
been split into a number of modest size (e.g. 512KB) chunks.  The
Installation Guide provides detailed instructions on how to use these files
to install IRAF.  A complete installation requires that the user obtain the
AS source distribution (in this case the 'as.pcix.gen.gz' file or the
contents of 'splits/as.pcix.gen') and at least one set of IB and NB binary
distribution (e.g. 'ib.rhux.x86.gz' and 'nb.rhux.x86.gz' for a RedHat Linux
system, or 'ib.macx.ppc.gz' and 'nb.macx.ppc.gz' for a MacOSX system).

GUIAPPS.  The guiapps.tar.gz file is an OPTIONAL external package
containing prototype IRAF applications with GUIs.  Use of these tasks
requires that XGterm be used for the IRAF session.  Unlike other external
packages, guiapps is being distributed with prebuilt binaries to simplify
the installation - all you have to do is unpack the distribution and edit
the hlib$extern.pkg file.  Please see below for detailed information on
this package.


1.2 WHO SHOULD UPGRADE?

IRAF V2.12 is a major IRAF release for all supported IRAF platforms.
Support for older versions of IRAF is limited so sites running earlier
versions of IRAF should update to the new version of IRAF.

There may be incompatibilities between V2.12 and earlier versions of IRAF,
so updating in the middle of an analysis program might not be advisable.
Mixing different versions of IRAF works in most cases, but there can be
complications.  Large installations may want to keep and older version of
IRAF around to give users time to complete their programs before switching
to the new version of IRAF (e.g. here at NOAO, "cl" or "iraf" runs V2.12,
and "irafo" will bring up the older V2.11.3 release).


-----------------------------------------------------------------------------
2. HIGHLIGHTS OF THIS RELEASE
-----------------------------------------------------------------------------

  ************************************************************************
  #     See also the 'v2122revs.txt' file in the PC-IRAF distribution     #
  #  directory for additional changes in the V2.12.2 patch release.   #
  ************************************************************************


    o Pixel Mask Support Added to FITS Kernel
        The FITS kernel was modified to add support for storing pixel masks
        in FITS extensions as compressed images using the binary table
        extension.  These masks appear as images to the system and may be
        accessed like any other image.  Storing the pixel masks directly
        in a FITS MEF file makes it possible to store all the data for
        one observation in a single file, simplifying data management.

    o New Pixel Mask Tasks
    Several new tasks have been added to the system PROTO package for
    manipulating pixel masks:

    o MIMSTATISTICS allows image statistics to be computed while
        rejecting pixels specified by an input mask.
    o MSKEXPR task is a general-purpose mask expression evaluator
        similar to IMEXPR for images, but has builtin boolean region
        functions which can be used to set values inside or outside
        of certain geometric shapes.
    o MSKREGIONS creates an output mask based on an input text de-
        scription.  Region descriptions can be composed of geometric
        shapes and logical operation on mask regions.
    o OBJMASKS in the NPROTO package is a new task for detecting objects
        in an image and creating an output catalog or pixel mask of
        found objects.

    o Shared Memory Limitations Eased
    The IMAGES and DAOPHOT packages executables are now linked statically
    to remove per-process memory limitations imposed by the IRAF shared
    library on Sun and Dec Alpha systems.  Previously tasks such as
    DAOFIND and IMCOMBINE were limited to 268Mb on Sun systems,  these
    tasks can now use up to the machine memory limits.

    o Image I/O Buffer Sizes Increased
        Image I/O performance for very large images was improved by changes
        to the internal buffer sizes.  By default the system will now
        automatically adjust the size of the image buffers to optimize i/o
        for the size of the image being accessed.  Additional control over
        i/o is possibly by tuning environment variables.  In addition, some
        applications will now (memory permitting) adjust the IMIO buffers
        to hold the entire image in memory, where usage warrants it.

    o Simplified Installation Script
        The IRAF install script was rewritten to clarify the output,
        provide more extensive checking of the IRAF system setup prior to
        installation, and to do some of the most common post-install
        configuration.  The script will print an explanation of any errors
        it finds and suggest corrective action.  The hope is this will lead
        the user past some of the more common installation errors.

        In addition, the SYSINFO diagnostic script which does more
        extensive checking of the system is also now part of the
        distribution.  This script can be used to verify the system once
        the install is complete, or to generate a report of the system
        configuration if needed by site support.  An UNINSTALL script to
        remove iraf command links and files created by the INSTALL script
        is also available to remove IRAF from a machine.  All scripts are
        now installed in the hlib$ directory.

    o New HELP GUI and Output Options
        The HELP task was enhanced to provide a new GUI option (XGterm is
        required).  This is essentially the XHELP task which has been
        available in the GUIAPPS external package for some time, however
        this version of the task is fully backwards compatible with old
        HELP task, and text-mode output is still the default.  In addition,
        help pages may now be output in either HTML or Postscript format
        for Web presentation or pretty-printing to a hardcopy device.  The
        LROFF task was similarly modified to provide direct conversion of
        Lroff format text.

    o DISPLAY Task Changes
        As part of the recent X11IRAF enhancements, the DISPLAY task and
        others such as IMEXAMINE which interact with the display server
        were modified to take advantage of the new features in XImtool
        V1.3.  These include support for up to 16 frame buffers (increased
        from 4 in previous versions), and enhanced real-time WCS readout
        capabilities.  The changes are fully backwards compatible for use
        with older XImtool versions or display servers such as SAOimage,
        SAOtng, or DS9 which have not yet been updated, however the new
    features will not be available on these servers.

        X11IRAF V1.3 is being released simultaneously with IRAF V2.12.
        While IRAF V2.12 is fully compatible with older versions of
        X11IRAF, users will need to upgrade both systems to the latest
        versions to take full advantage of all the new features.  Please
        see the X11IRAF Release Notes for details on what has changed.

    o New Packages
    Several new packages are available in this release (see the NOAO
    package change notes below for details):

        - A new ASTCAT package for extracting astrometric and photometric
      calibration data from remote or local catalogs was added to NOAO.
        - A new CRUTIL package for doing cosmic ray detection and removal
      package was installed in the IMRED package.
        - A new QUADRED reduction package for QUAD format data was installed
      in the IMRED package.  This is a generalized replacement for the
      ARED.QUAD and XCCDRED external packages for processing CTIO and
      ESO FORS1 multi-amplifier data.
        - A new OBSUTIL package was installed in NOAO.  This is a collection
      of tasks from various external packages which are useful to plan or
      carry out observations.

    o New Developer Libraries.
    Several new libraries are available for SPP developers:

        - PSIO is a new Postscript text generation library installed in
      sys$psio.
        - CATQUERY is a remote astrometric/photometric catalog access lib
      installed in the XTOOLS utility library.
        - SKYWCS is a sky coordinate transformation library installed in
      the XTOOLS utility library.


-----------------------------------------------------------------------------
3. INSTALLING PC-IRAF
-----------------------------------------------------------------------------

The procedure for installing PC-IRAF is unchanged from earlier versions of
PC-IRAF.  Refer to the PC-IRAF Installation Guide (pciraf.ps.Z) for detailed
installation instructions.  A full installation, as for any major release,
will be required.
```

```
The installation guide contains the full installation instructions (see also
the notes below) but one thing is worth emphasizing here: the installation
will be simplified if you set up the iraf directories as follows:

    <path>/iraf            root of iraf related files
    <path>/iraf/iraf           root iraf directory (AS dist)
    <path>/iraf/irafbin        iraf bin dirs go here
    <path>/iraf/irafbin/bin.redhat  RedHat binaries for core system
    <path>/iraf/irafbin/noao.bin.redhat RedHat binaries for noao packages
    <path>/iraf/irafbin/bin.linux   Linux binaries for core system
    <path>/iraf/irafbin/noao.bin.linux  Linux binaries for noao packages
        :               :
    <path>/iraf/extern          external packages (tables etc.)

Here "<path>" is the path where all this is located, e.g., "/u3/iraf" on the
IRAF development system here at NOAO.  The path can be anything, although it
is best to keep it short.  You might want to also set up a symbolic link
"/iraf" pointing to the "<path>/iraf" directory.  This allows all iraf files
to be referred to relative to /iraf, regardless of where the files actually
are located, and agrees with the default configuration used in the
distribution files.  Having such a link when IRAF is served from a central
NFS system also allows all clients in the network to use a common "/iraf/iraf"
root directory regardless of the NFS mount points.  Graphically such a tree
would look something like (for a RedHat-only system)

                            /iraf
                            /  \
                   (AS) /iraf /irafbin
                                /   \
                    (IB) bin.redhat  noao.bin.redhat (NB)

The graph indicates where each of the three distribution sets should be
unpacked.  The "iraf root directory" in this case is /iraf/iraf, this
is the value you enter to the install script.  See the installation
guide for details, specifically the Appendices which give a complete
examples.

The V2.12 PC-IRAF release supports several architectures;

    System             Distribution       Architecture
    ------             ------------       ------------
    FreeBSD 4.2 and up         FBSD               freebsd
    MacOSX 10.2 and up     MACX           macosx
    RedHat Linux 6/7/8/9       RHUX               redhat
    RedHat Fedora/Enterprise   RHUX               redhat
    Slackware 8/9          LNUX/x86       linux
    Solaris 7 for Intel        SSOL               sunos
    SuSE Linux 6 thru 9        SUSE               suse
    Yellow Dog Linux 3.0       LNUX/ppc           linuxppc

A separate set of CORE and NOAO binaries is provided for each architecture.
The all-sources (AS) distribution supports both architectures.  The binaries
which are correct for your system are identified by the "Distribution" type,
however when creating the IRAF tree for unpacking the system you should
create directories according to the "Architecture" name or they will not
be recognized.  Note that systems such as RHUX for RedHat Linux are appropriate
for Mandrake Linux systems as well, almost all other linux distributions note
handled by specially built IRAF binaries should probably use the LNUX system.
```

```
3.1 MACOSX TECHINICAL NOTES

MacOSX is still a young system and is in many ways quite different from
other systems supported by PC-IRAF.  While we don't yet have a lot of
experience on this system, major items will be documented here and at the
end of the README file as they are discovered.  Problems specific to the
X11IRAF tools will be documented in it's README file and on the IRAF web
pages.  Users should contact IRAF site support with any questions.

In addition, Marcos Huerta at Rice University (marcosh@rice.edu) has created
an excellent support page for IRAF/X11IRAF on OS X systems.  This page
can be accessed at

    http://www.owlnet.rice.edu/~marcosh/iraf/

Marcos has also created package installers for both IRAF and X11IRAF to
simplify the entire process.  These installers are available from our
archive /contrib directory or from the web page above.


3.1.1 EXTERNAL PACKAGE SUPPORT

Since MacOSX and LinuxPPC are new ports, many older external packages may not
yet support, or have been tested with, these platforms.  In most cases
however, all that will be required to use such a package with IRAF is to
modify the root mkpkg file to add an entry for the "macosx" and/or "linuxppc"
architecture, then building the package.  Prebuilt binaries for some of the
most popular external packages are already available in the extern and contrib
directories in the IRAF network archive.

Problems and workarounds for packages will be logged at the end of this
README file as they are discovered, and in the days and weeks following the
release the iraf.noao.edu archives will be updated with new versions of the
packages to fix any problems.  Users should contact IRAF site support with
questions about support for a particular package on this platform.

For the impatient, the mkpkg at the root of each external package will
usually require a branch to set the architecture prior to compilation.  The
statements to be added (at the end of the file) will typically look
something like:

    macosx:                                 # install MacOSX binaries
            $ifnfile (bin.macosx)
                !mkdir bin.macosx
            $endif
            $verbose off
            $set DIRS = "lib src"
            !$(hlib)/mkfloat.csh macosx -d $(DIRS)
            ;
    linuxppc:                               # install Linux PPC binaries
            $ifnfile (bin.linuxppc)
                !mkdir bin.linuxppc
            $endif
            $verbose off
            $set DIRS = "lib src"
            !$(hlib)/mkfloat.csh linuxppc -d $(DIRS)
            ;

Existing entries for the package should be used as a guide; in particular
the directories listed in the "$set DIRS = ...." line must be consistent
```

for the package to be modified correctly. With this change most external packages can be built as they would on any other platform. Packages for which NOAO is responsible are available on iraf.noao.edu in the directory ftp://iraf.noao.edu/iraf/extern. Software from other sources may require you to make this change yourself prior to building the package. Where possible we will try to include binaries and updated code for packages distributed from our /contrib archive directory.

3.1.2 THE MAC OS X 'IRAF' USER ACCOUNT

Under MacOSX it turns out to be difficult to change the iraf account login directory once it is created. The system wants to keep the login directory in /Users/iraf where IRAF wants it to be iraf$local (this can be fixed as root using nidump/niload, but it is not easy and could be dangerous). For the time being we are recommending that users simply create the iraf account but leave it to the install script to handle setting up links as needed so the proper environ- ment is maintained as on other systems. See the PC-IRAF Installation Guide for more information on how this part of the installation differs from other platforms.

3.2 THINGS TO WATCH OUT FOR (IRAF)

Please see the release notes (iraf$doc/v212revs.hlp or just type 'news' when you first log into the system) for information on what has changed in V2.12, and things to watch out for.

As with any major release we STRONGLY recommend that each user reinitialize their 'uparm' directory to pick up the numerous task and package parameter changes in this release. IRAF V2.12 does not require this if you are upgrading from V2.11, but it is still strongly recommended. The uparm directory may be reinitialized by issuing a new MKIRAF command prior to logging into the system.

3.2.1 SPECIAL NOTE TO FEDORA AND RHEL USERS

Preliminary testing under Redhat Fedora and Enterprise Linux systems has revealed a potential problem with the interaction between the MEMIO interface and user resource settings. We do not yet know whether this will affect other distributions using the same newer glibc and kernel versions, or if this is a problem peculiar to Redhat. In either case the workaround will be similar and the problem will be addressed more fully in the next release.

Specifically, pointers allocated in the normal course of a task may occassionally be at an address outside the user's per-process stack space, resulting in a "memory has been corrupted" or "segmentation violation" error. This problem was seen during the beta test period with IMFORT tasks and was originally thought to be a problem with the "exec-shield" feature of Fedora, but has also appeared on RHEL systems without exec-shield.

The workaround is to remove the stacksize limit in the user's shell with a command such as

```
        limit stacksize unlimited              # for tcsh users
        ulimit -s unlimited                    # for bash users
```

As a preventive measure against this problem, the CL startup script was modified to implement this change and so most users who only use IRAF from the CL will not need to take any special action. This remains an issue for IMFORT tasks however, and users may need to use one of the above commands to get the tasks to run correctly.

3.2.2 SPECIAL NOTE TO LINUX IMFORT USERS

In addition to the stack size problems above, platform support for this release was further complicated by changes to glibc and the 'ld' loader on some newer linux distributions, which resulted either in unresolved symbols or a segfault from the loader itself. Users would see various combinations of those problems depending on the distribution being used.

To fix the unresolved symbol problem the compatability library 'libcompat.a' (found in the iraf$unix/bin.<arch>) was updated to include the missing symbols from older glibc versions and the XC compiler modified to use this library on more platforms. The loader segfault is caused by the definition of the "Mem common" symbol "mem_" at address zero in the one and only assembler routine in IRAF (all iraf pointers are relative to this address). To fix this problem it was necessary to define the symbol on the GCC command-line instead, again by modifying the XC compiler to do this automatically.

These changes will be transparent to people compiling external packages, SPP sources using the XC compiler directly, or IMFORT programs using the FC command under the CL and only affect Linux platforms. However, users who build their IMFORT programs by calling the Fortran/C compiler directly with absolute paths to the needed IRAF libraries, say from a Makefile, will need to adjust their link line to include the compatability library and the extra linker argument.

To summarize, an imfort program built with F77/G77, and *not* the iraf XC/FC compilers, must now be linked as something like

```
  g77 myprog.f \
      -L/iraf/iraf/bin.<arch> \          <-- define iraf paths
      -L/iraf/iraf/unix/bin.<arch> \     <-- define iraf paths
      -Wl,--defsym,mem_=0 \              <-- NEW flag, fixes 'ld' segfault
      -lcompat \                         <-- NEW flag, fixes unresolved syms
      -limfort -lsys -lvops -los         <-- link imfort libs
```

Users with questions or problems should contact site support.

3.2.3 LINUX PPC SUPPORT (*** NEW IN V2.12.2 ***)

V2.12.2 is the first release to fully support Linux for the PPC platform as part of the PC-IRAF system. The port was done a while ago but never fully integrated into the V2.12 release until a disk failure prompted us to configure a dual-boot machine. This upgrade contains the completed port which was built using Yellow Dog Linux V3.0.1 on a PowerMac G4 733Mhz system.

The port appears to be stable in our limited testing but we would be interested in working with users on this platform in finding any problems. Initial benchmarking of IRAF V2.12.2 shows that on average IRAF is roughly 30% *faster* under YDL than on the same machine running OS X 10.3.2. The exact cause of this difference is not yet fully understood, see below for benchmark results offering some interesting results.

It is also not known at this time whether the binaries will work on other PPC Linux systems such as Redhat, Debian or SuSE (and support for G5 systems is even more in doubt). The system should build cleanly from source on other distributions however some tweaking may be required. If you are interested in building IRAF for a distribution other that YDL, please see the "IRAF Site Manager's Guide" for details on the complete build steps, or contact IRAF site support (iraf@noao.edu).

We are grateful to Terra Soft Solutions (www.terrasoftsolution.com) and Yellow Dog Linux (www.yellowdoglinux.com) for their continued interest and support in getting IRAF ported to this platform.

3.2.4 OPTIMIZATION CHANGES FOR MAC OS X

The default compiler flags for PC-IRAF systems were re-examined as part of this release with an eye towards improving performance. As part of this, the default optimization level under Mac OS X was raised to "-O3" despite providing only a relatively slight improvement in speed on the same hardware. Similar changes were investigated for Linux systems but had to be abandoned due to a (as yet untraced) compiler optimizer bug present in GCC 2.95 thru GCC 3.2 which produce incorrect results. This bug is present in both x86 and ppc linux systems, however the GCC 3.3 compiler under OS X 10.3 seems have fixed it.

Sample benchmarks (including LinuxPPC for comparison) are shown below. It's important to understand that these tests provide only a very crude benchmark, but show systematic differences between the platforms tested. All tests were conducted on the same physical machine.

```
  XREGISTER Tests:  A CPU intensive test using the following commands:

      cl> blkrep dev$pix test100.fits 4 4
      cl> $xregister test100.fits test100.fits "[*,*]" test1

   Results:
      Version  Opt Flag  Build OS       Time
      -------  --------  --------       ----        (Times were about equal
      V2.12.1    -O      OSX 10.1       1:19         for both FITS and IMH
      V2.12.2    -O      OSX 10.3.2     1:02         image formats)
      V2.12.2    -O3     OSX 10.3.2     0:52
      V2.12.2    -O      YDL 3.0.1      0:52

  BENCH script: A crude script which approximates a CCD reduction and analysis.
        Uses both CPU and Disk-intensive tasks to generate and operate on the
       images.  See script at ftp://iraf.noao.edu/pub/bench.cl

   Results:
               V2.12.1/-O     V2.12.2/-O      V2.12.2/-O3     V2.12.2/YDL 3.0
                imh fits       imh fits        imh fits         imh fits
   Make 7 imgs    41  48         38  50          40  52           26  33
   Proc 5 imgs    28  36         25  37          25  36            6  15
   Combine 5 imgs  16  16        14  17          12  16           12  11
   Median 1 img   32  32         29  33          28  32           27  17
   Total time    117 133        106 137         106 137           71  77
```

Following the V2.12.2 release we will examine the potential gains for further

tuning the OS X binaries for G5 systems. If these are found to be significant
a separate set of binaries for G5 will be released.

```
-----------------------------------------------------------------------------
4. X11/GUI SUPPORT
-----------------------------------------------------------------------------
```

IRAF V2.12 includes full support for the X11IRAF utilities - these include
xgterm for xterm-compatible terminal emulation and graphics, and ximtool
for image display under X. The X11IRAF package is not included in IRAF;
you need to get it and install it separately, as you would any other X
software. X11IRAF is available in /iraf/x11iraf on the main IRAF network
server (iraf.noao.edu). This software is continually under development
and new versions appear on a timetable independent of that for the main
IRAF distribution.

Note that a major new X11IRAF V1.3 release is timed to coincide with the
IRAF V2.12 system. XImtool users especially should upgrade to take
advantage of the many new features and bug fixes. See the README file in
the /iraf/x11iraf archive directory and the X11IRAF V1.3 release notes for
details on what has changed.

Other IRAF-compatible GUIs can also be used with IRAF, e.g. DS9, SAOtng,
and SAOimage. This software is available from the archive /contrib directory
as a convenience but may not always be the most recent release.

4.1 THE GUIAPPS EXTERNAL PACKAGE

Some time ago we started a project to develop prototype IRAF applications
with integral GUIs (graphical user interfaces). These applications have
since served their purpose in prototyping new technology for science GUIs
and component-based applications in IRAF. The GUI applications are much more
than just technology prototypes however: these are very useful science
applications with many new features, with a nice graphical user interface
to boot. With the release of IRAF V2.11.3 and X11IRAF V1.2 several years
ago, it became possible to wrap up work on the GUI prototypes and get them
out for people to use.

The prototype GUI applications are contained in the package GUIAPPS
available from the /iraf/extern directory of the IRAF archive, and contain
the following tasks:

```
              demo - GUI Demo Task
               spt - SPECTOOL Package
             xhelp - GUI Help Browser Task
               xrv - GUI Version of the RV Package
           xapphot - GUI Aperture Photometry Package
```

Note that with V2.12 the XHELP task is included as part of the core system
as a special device type (e.g. "help implot dev=gui" will run HELP as a GUI
task). The GUI tasks require that the user be running an XGterm graphics
window (from X11IRAF) in order to display the GUI: Xterm users WILL NOT be
able to use the GUIs presently.

To install the package users should download the 'guiapps.tar.Z' file from
the IRAF archive directory and uncompress/untar it in the local external
package directory (typically /iraf/extern). Detailed installation inst-
ructions are 'guiapps.readme' file in the same directory.

More information of the GUIAPPS package is available from the project page at
http://iraf.noao.edu/iraf/web/projects/guiapps/.

4.2 THE X11 DESKTOP

The X11IRAF utilities will run under any X desktop. Most people will
probably use Sun's CDE or OpenWindows desktops, but Motif/mwm, twm, fvwm,
and so on may be used as well. As of the time of this writing, the
X11IRAF utilities do not run on 24 bit truecolor screens, only 8 bit
pseudocolor. Support for 24 bit screens is planned.

New users of X11 who are not sure how to configure the window system might
want to look at the .Xdefaults, .xinitrc, .openwin-menu, etc. files in the
IRAF system manager login directory ($iraf/local, or ~iraf). These files
will set up a working IRAF desktop under OpenWindows and should serve as
an example or starting point for an IRAF desktop under OpenWindows.
Logging in as "iraf" and then typing "win" or "openwin" will start up the
windows.

4.3 THINGS TO WATCH OUT FOR (X11IRAF)

Please see the documentation included with X11IRAF V1.2 or later releases
for information on running these utilities, including common problems and
how to deal with them.

One thing we would like to emphasize here is that we STRONGLY suggest that
you use "xgterm", not the standard "xterm" to run IRAF. XGterm, which is
included in X11IRAF, has advanced graphics capabilities developed for use
with IRAF. Xterm graphics work, but not terribly well. When you start up
the cl in an xgterm window, type "stty xgterm" after the CL starts up to
tell it you are using an xgterm window. The default terminal type may be
set with the MKIRAF command to make the change more permanent.

4.3.1 REDHAT 8/9 SPECIAL NOTE

X11IRAF users on RedHat 8 system are likely to encounter two known problems
when using the current binary distribution: The first is an error when start-
ing xgterm such as

```
    % xgterm
    xgterm: error while loading shared libraries: libncurses.so.4: cannot open
    shared object file: No such file or directory
```

This is a known problem with how the binary was built and RH7/8 systems.
The simplest solution is to just create a symlink to satisfy the
dependency, i.e. as 'root' do

```
      # cd /usr/lib
      # ln -s libncurses.so.5 libncurses.so.4
```

Alternatively you could recompile the system from the sources and you should
then be able to run xgterm normally.

Additionally, you may encounter another problem starting xgterm under RH8
caused by the xgterm trying to write a /var/run/utmp entry without the

```
proper permissions, this causes the window to appear briefly and then shut
down.  The workaround is to either start it as "xgterm -ut" to disable
this utmp entry, or else change the permissions on the binary as root
using

      # chown root:root xgterm
      # chmod g+s  xgterm

This will make the binary setgid root and should give it the proper
permissions to write a utmp entry.  The reason 'xterm' doesn't require
this is that it's linked against a special library which calls a different
setgid root task to create the same entry.




-----------------------------------------------------------------------------
5. MAGTAPE INTERFACE
-----------------------------------------------------------------------------

The DEV directory contains default template tapecap files for Linux (i.e.
dev$tapecap.linux), Freebsd (dev$tapecap.freebsd), and Solaris x86
(dev$tapecap.sunos).  There is presently no tape support for MacOSX, as we
don't have a tape drive on any of our test machines.  If a server is
configured to run IRAF for multiple network clients, you can now have
multiple tapecap files, one for each host, e.g. "tapecap.foobar" for host
"foobar".  If a node-specific tapecap file is not found the fallback will
be to use the 'dev$tapecap' file.  Users should copy one of the template
tapecap files provided to the fallback dev$tapecap file to take advantage
of several common predefined tape device entries, however the new install
script in V2.12 will do this automatically if you proceed with the
post-install configuration.

See the release notes for further information on this capability, and the
IRAF Site Manager's Guide for more information on configuring tape devices.


-----------------------------------------------------------------------------
6. PRINTER INTERFACE
-----------------------------------------------------------------------------

Local printers are interfaced to IRAF using the dev$graphcap (for hardcopy
graphics) and dev$termcap (for text output) files.  The default printer,
defined by the "printer" environment variable for text and 'stdplot' for
graphics in hlib$zzsetenv.def, is "lp".  This uses the host system lpr with
no explicit printer name given, allowing the host environment variable PRINTER
to be defined in ther user's environment to direct out to the actual printer.

These files may need to be customized to support the printers in use at your
site.  See the IRAF Site Manager's Guide for detailed instructions on
configuration the printer interface.


-----------------------------------------------------------------------------
7. PATCH INSTRUCTIONS
-----------------------------------------------------------------------------

7.1 PC-IRAF V2.12.2 - Fri February 6, 2004

The PC-IRAF distribution files have been fully regenerated for V2.12.2, so
```

```
one way to upgrade IRAF to V2.12.2 is to do a full install and diff/merge
selected locally customized files from DEV (tapecap, graphcap, imtoolrc,
devices.hlp, etc.) and HLIB (extern.pkg, mkiraf.csh, login.cl, zzsetenv.def).

If you have an existing PC-IRAF V2.12 installation you have the option
of installing the patch file.  Note that THE V2.12.1 PATCH WILL OVERWRITE
CERTAIN SITE SPECIFIC FILES.  These are the following:

    unix/hlib/extern.pkg
    unix/hlib/zzsetenv.def
    dev/graphcap
    dev/termcap
    dev/imtoolrc
    dev/hosts

You may want to save copies of these files and diff/merge them with the
new versions after installing the patch.

To install the patch, login as the iraf system manager, go to the $iraf
directory, and untar the patch as follows:

        % cd $iraf
        % cat patch2.tar.Z | uncompress | tar -xpvf -
or      % tar -zxpvf patch2.tar.gz

Then diff/merge the files listed above.  Next, go to the archives and
download and install the IB (IRAF core system) and NB (NOAO package)
binaries for the architectures your installation will support.  For example,
to install IB.RHUX.X86 (the RedHat Linux binaries for the core system, with
the IB.RHUX.X86 distribution downloaded to the directory
"<path>/ib.rhux.x86" (where <path> is where you created this directory):

    % cd $iraf/bin.redhat
    % pwd
    /iraf/irafbin/bin.redhat       # (you should be here, or somewhere similar)
    % cat <path>/ib.rhux.x86.gz | zcat | tar -xpvf -

A similar procedure is followed to install each of the other binaries.

See the URL http://iraf.noao.edu/v212/v212-patch2.html for a detailed desc-
ription of what is included in the patch.



-----------------------------------------------------------------------------
8. NOTES ADDED SINCE THE LAST RELEASE.
-----------------------------------------------------------------------------

-----------------------------------------------------------------------------
Install Script Bug -- Sun/IRAF tapecap configuration
(10 May 2002)


    Two small typos were found in the new install script during the
post-install configuration of the tapecap files:

     o  The first is simply a typo in the output message where it refers
    to 'tapecap.sunos' rather than 'tapecap.solaris' when the script
    is run on Solaris systems. This is harmless since the links are
        in fact created with respect to the proper file when executed.
     o  The second problem is another small typo in a variable name that
    would prevent an existing tapecap file from being properly deleted,
    the script next tries to create a symlink of the same name resulting
    in an error message.

     Both problems have been fixed and a new version of the install script
is available from:

                ftp://iraf.noao.edu/pub/install


-----------------------------------------------------------------------------
IRAF V2.12 Install Script Bug -- "Badly formed number" Error on Sun/IRAF
(15 May 2002)


The post-install configuration of the tapecap files can sometimes fail
on Sun/IRAF systems with a "Badly formed number" error similar to:

        :
    Create a default tapecap file?  (yes):
    Creating default file 'tapecap.foobar' from tapecap.sunos...
    if: Badly formed number

The cause is a check for a tapecap being a symbolic link using the "-l"
csh directive which isn't portable,  the solution is to simply use a "-e"
to check for the existance of the file/link.

     Both problems have been fixed and a new version of the install script
is available from:

                ftp://iraf.noao.edu/pub/install


-----------------------------------------------------------------------------
IRAF V2.12 Install Script Bug -- More syntax errors
(17 May 2002)


Several more problems have been reported with the new install script on
Mac OS X and Sun systems:

    o   On some Mac OS X systems there is no /usr/include directory where
    the Developer's Tools haven't been installed.  The script was
    modified to check for the existance of this directory and create
    it if needed.

    o   The install script uses the C-shell builtin 'which' command
    in some cases to determine when commands might exist on the system.
    The script however was assuming that this command would fail with
    a "command not found" message but on some systems the error
    would be "no cmd found in ....".  The script was modified
    to handle either case.

     Both problems have been fixed and a new version of the install script
is available from:

                ftp://iraf.noao.edu/pub/install
```

```
-----------------------------------------------------------------------------
IRAF V2.12 Install Script Bug -- More changes
(24 May 2002)


On some systems the detection of an <i>'iraf'</i> user account may fail
since it relies either on the 'id' command or an entry in the local
/etc/passwd file.  An iraf user account may exist under DNS but still not
show up in the install script.  The verification was changed from a
fatal error to a warning to allow the installation to proceed.

     This problem has been fixed and a new version of the install script
is available from:

                ftp://iraf.noao.edu/pub/install


-----------------------------------------------------------------------------
IRAF V2.12 Install Script Bug -- More changes
(28 May 2002)


In some user environments the "which" command may not be the shell-builtin
command but and Open Source application.  When this application fails to
find the command the message is written to the stderr stream which was not
properly being checked in all cases, usually resulting in a "syntax error".

     This problem has been fixed and a new version of the install script
is available from:

                    ftp://iraf.noao.edu/pub/install


-----------------------------------------------------------------------------
IRAF V2.12.1 PHOTCAL -- FITPARAMS task not finding catalogs
(20 July 2002)


The FITPARAMS task may fail to find catalogs due to a invalid 'catdir'
package parameter introduced in the V2.12.1 update. This problems affects
all systems.

This can be fixed at the user level by resetting the PHOTCAL package
parameters to the correct value (before loading the PHOTCAL package), i.e.

             cl> photcal.catdir = "photcal$catalogs/"

Site managers can make the fix system-wide by editing the file
iraf$noao/digiphot/photcal/photcal.par to change the default value for the
'catdir' parameter to remove the 'x' from 'photcalx'.


-----------------------------------------------------------------------------
IRAF V2.12.1  RedHat 9 -- Binary incompatability
(13 August 2003)


RedHat 9.0 introduced the GLIBC 2.3 library which is not bacwards binary
compatible with earlier releases using the system <ctype.h> functions such
```

```
as toupper(), tolower(), etc.  These routines are used in the IRAF HSI (i.e.
libos.a and libboot.a) and so users building software against V2.12.1 will
see linking failures such as

    /iraf/iraf/unix/bin.redhat/libos.a(zfioks.o)(.text+0x19): In function
        `zopnks_': undefined reference to `__ctype_b'

when compiling external packages or local tasks.

        The only workaround is to use a version of the IRAF system compiled
entirely under RH9 (however we are investigating other solutions for the next
patch to avoid having two binary systems).   The "redhat9" subdirectory here
contains such a system and should be used when installing from scratch on a
RH9 platform.  Note this is exactly the same V2.12.1 system as before so
there is no need to upgrade your IRAF unless you are 1) on a RH9 platform and
2) need to build IRAF software.  The directory contains only the AS, IB and
NB distribution files, installation of these files is the same as before.

     In addition, HSI binaries (e.g. RTAR/WTAR, hardcopy SGI graphics
translators, etc) have been known to fail on RH9 systems due to a similar
incompatability. If you need to update your system only to fix this
problem, use the file hs.rhux.x86.gz to replace your existing
$iraf/unix/bin.redhat directory contents.

       It is not necessary to upgrade your entire IRAF system with this
release, simply installing the 'hs.rhux.x86.gz' system by uncompressing
and untarring in your $iraf/unix/bin.redhat directory is sufficient to
fix both compatability problems.

       Please contact iraf@noao.edu if you have questions or problems.


-----------------------------------------------------------------------------
IRAF V2.12.2  Slackware 4 -- Binary incompatability
(12 February 2004)

    The 'slack40' subdirectory contains the IRAF V2.12.2 system built
for Slackware 4.0 and similar systems.  Due to numerous system differences
it was not possible to build a single binary for Slackware that would
run on V4.0 thru the latest (V9.1 at this writing) release, and so
official support for Slackware 4 was dropped.
    If you still have one of these systems and require IRAF V2.12.2,
these binaries should be used instead.  To use this release:

    1) Install the 'as.pcix.gen.gz' file (AS distribution) as
       usual.  This file is found in the parent directory.
    2) Overlay the hs.lnux.x86.gz file from this directory on
       the source distribution by unpacking from the $iraf
       root directory.
    3) Install the IB and NB distributions from this directory.

Then proceed with the rest of the installation as usual.  Please contact
iraf@noao.edu with questions.
```