Starlink Project
Starlink System Note 64.5

A J Chipperfield

3 September 2002

ADAM — Organization of Application Packages

Abstract

This document explains the standard way in which ADAM applications packages are set up to run on Starlink machines. It covers the systems permitting use of the applications direct from the shell and from the Interactive Command Language (ICL) on Unix.

This document is relevant to applications programmers, the Starlink Software Librarian and Site Managers.

Contents

1 Introduction

This document explains the standard way in which ADAM applications packages are set up to run on Starlink machines. It covers the systems permitting use of the applications direct from the Unix shell and from the Interactive Command Language (ICL).

Changes in this version of the document are listed in Section 6.

When a new package is being developed, the way in which it will fit into the scheme should be discussed with the Starlink Software Librarian and the Head of Applications. KAPPA provides a good example of how things are done and may be used as a template. Deviations from the scheme will be permitted but compliance with it will ease the task of setting up and maintaining the package.

In this document *package* means 'the name of the package (in upper case for environment variables)'.

2 Packages

2.1 General

The term 'package' in this document may be taken to mean a related group of ADAM applications. Packages are described as *standard* if they are installed at every Starlink site or *option* if the are only installed when requested by users. Allowance is also made for *local* packages which are non-Starlink packages set up for general use at particular sites. It is assumed that packages may be run directly from the shell or from ICL; where this is not the case, the inappropriate files are omitted from the installation.

In most cases applications in the package will all be linked into a single monolith (see SUN/144) for efficient running from ICL and to save disk space. Packages could however be defined to consist of any mixture of task types and procedures – they could, in fact, include elements from other packages. Because a package may well consist of applications written by a number of different authors, the term *Package Administrator* is used for the person with overall responsibility for the package.

Executable files, shell scripts, ICL command files and compiled interface files associated with a package will usually be installed in a directory referred to by environment variable *package*_DIR. For most applications packages this directory is `/star/bin/`*`package`*. If the package can be run direct from the shell, *package*_DIR should also contain links with the name of each application in a monolith pointing to that monolith.

e.g.

```
% ln $KAPPA_DIR/kappa_pm $KAPPA_DIR/add
```

Individual interface files will also be required in addition to the monolithic interface file in this case.

2.2 Package Help Libraries

Each package should provide a *Package Help Library*. This is a Starlink HELP library (see SUN/124). The structure of the HELP library is at the Package Administrator's discretion but at the very least it should contain an overall description of the package with subtopics for each application.

The usual format is described below:

Level 0 A general description of the package.

Level 1 A description of individual applications and other general topics such as 'Getting Started'.

Level 2 As required. Each application will normally have a subtopic 'parameters' which has subtopics describing each parameter. These parameter subtopics may be referred to in parameter help specifications in task interface files (see SUN/115).

Package Help Libraries will usually be installed in directory `/star/help/`*`package`* and named *`package`*`.shl`. Environment variable *package*_HELP should be defined to point to the Package Help Library and this used in the relevant commands.

2.3 Shell Package Startup Scripts

Generally, to start up a package for use direct from the shell, the user will type:

```
% package
```

The Starlink login procedures will have defined command, *package*, which, when issued by the user, will cause commands in the *Package Startup Script* to be obeyed.

The Package Startup Script will define commands for running the individual applications and for obtaining help. It should also issue a 'Welcome' message stating the package name and version. Typically it will contain mainly alias commands of the form:

```
% alias application $package_DIR/application
```

e.g.:

```
% alias add $KAPPA_DIR/add
```

(Note that here $KAPPA_DIR/add is a link to the KAPPA monolith as described in Section 2.1).

The Package Startup Script should be provided by the Package Administrator and be installed in *package*_DIR.

2.4 ICL Package Definition Files

Generally, to start up a package under ICL, the user will type:

```
ICL> package
```

This will cause ICL commands in the *Package Definition Command File* to be obeyed. All but the simplest packages will contain a Package Definition Command File which is a file containing ICL commands to:

(1) Define the commands which the user will use to run the package.

(2) Specify the source(s) of help information for the package.

(3) Display information about the package.

For example, the Package Definition Command File for KAPPA on Unix could be something like:

```
{ KAPPA - Package Definition Command File}

{ Re-define the top-level help topic
DEFHELP KAPPA $KAPPA_HELP 0

{ Define the individual commands
DEFINE ADD $KAPPA_DIR/kappa_pm
DEFHELP ADD $KAPPA_HELP

DEFINE (APER)ADD $KAPPA_DIR/kappa_pm
DEFHELP APERADD $KAPPA_DIR/KAPPA

... etc...

PRINT
PRINT  " --    Initialised for KAPPA    --"
PRINT  " -- Version 0.8-SU, 1993 January --"
PRINT
```

The Package Definition Command File is controlled by the Package Administrator and should be installed in *package*_DIR.

For very simple packages, a Package Definition Command File may not be appropriate – the commands required to run such a package may be set up directly by the overall system startup procedures.

3 The Overall System

3.1 The ADAM_PACKAGES Help Library

A Starlink HELP library giving general descriptions of all Starlink packages will be maintained in `/star/help/adam_package.shl` by the Starlink Software Librarian. The top-level topic will give a list, with a single line description, of all the *standard* and *option* packages and will be displayed if the user types:

```
ICL> HELP PACKAGES
```

The second level will be subtopics for the individual packages. Each subtopic will give a brief description of the package and describe how to start using it. Before any packages have been started up by the user, the command:

```
ICL> HELP package
```

will display the appropriate subtopic.

Environment variable ADAM_PACKAGES points to this help file.

3.2 ICL Startup Command Files

ICL uses environment variables to identify files containing commands which it will obey automatically before taking input from any file specified as a parameter of the ICL command, or prompting for input. The environment variables, in the order they are accessed, are:

ICL_LOGIN_SYS	Intended for 'system' login commands
ICL_LOGIN_LOCAL	Intended for local site login commands
ICL_LOGIN	Intended for user's login commands

For Starlink sites, ICL_LOGIN_SYS points to a file, controlled by the Starlink Software Librarian which contains ICL commands which:

(1) For all Starlink packages, define, for the ICL help system, the relevant entry in the ADAM-_PACKAGES help library.

(2) For each *standard* package, define an *ICL Package Startup Command* which, if issued by the user, will start up the package for use with ICL (usually by LOADing the Package Definition Command File).

(3) For each *option* package, check whether the package is installed at the site by checking for the existence of an appropriate file (usually the Package Definition Command File). If the package is installed, an ICL Package Startup Command is defined as for *standard* packages; if not, an ICL Package Startup Command is defined which, if issued, will inform the user politely that the package is not available at the site[1].

(4) Check if environment variable LADAM_PACKAGES defines a file which exists and, if it does, LOAD the file (see Section 3.3).

(5) Print a brief introductory message.

For example, for KAPPA (a *standard* package), on Unix it will contain:

```
{  Definitions for KAPPA }
DEFHELP KAPPA $ADAM_PACKAGES KAPPA
DEFSTRING KAPPA LOAD $KAPPA_DIR/kappa
```

For CCDPACK (an *option* package), it contains:

[1] As a temporary measure, this is also done for *standard* packages which have not yet been set up to run from ICL.

```
{  CCDPACK  }
DEFHELP CCDPACK $ADAM_PACKAGES CCDPACK
IF FILE_EXISTS("$CCDPACK_DIR/ccdpack.icl")
  DEFSTRING CCDPACK LOAD $CCDPACK_DIR/ccdpack
ELSE
  DEFSTRING CCDPACK NOTINSTALLED CCDPACK
ENDIF
```

Where NOTINSTALLED is a procedure which politely tells the user that the package is not available. Note that, because IF statements can only be included in ICL procedures (and not directly in command files), the command file first defines a hidden procedure containing the appropriate code for all *option* packages and then obeys it to define the required Package Startup Command.

3.3 LADAM_PACKAGES

This is an environment variable pointing to an ICL command file which, if it exists, will be LOADed during the ICL_LOGIN_SYS sequence. The file is intended to define both the Package Startup Command and the source of introductory help for *local* packages.

LADAM_PACKAGES will be controlled by the local Site Manager and may also contain any other site-specific login commands.

3.4 Starlink Login Actions

3.4.1 /star/etc/login

The following environment variables are defined in `/star/etc/login` which must be 'source'ed in order to use any Starlink software.

Variable name	Normal setting
ICL_LOGIN_SYS	/star/etc/icl_login_sys.icl
ADAM_PACKAGES	/star/help/adam_packages.shl

and for standard packages

*package*_DIR	/star/bin/*package*
*package*_HELP	/star/help/*package*

For *option* packages, the existence of an appropriate file within the package is checked to determine whether or not the package is installed. If it is, *package*_DIR and *package*_HELP for the package will be defined.

3.4.2 /star/etc/cshrc

Shell Package Startup Commands are defined in `/star/etc/cshrc` which must be 'source'ed in order to use any Starlink software. For *standard* packages a command is defined which will ‘source’ the Package Startup Script to start up the package for use from the shell. For *option* packages, the existence of an appropriate file within the package is checked to determine whether or not the package is installed. If it is, a Startup Command is defined as for *standard* packages; if not, a command is defined which will politely tell the user that the package has not been installed.

4 The Effect

The following example session shows the sort of effect it is hoped to achieve by means of this scheme. It does not pretend to be an exact copy of what you would see – in particular the help text is probably out of date.

```
% icl

ICL (UNIX) Version 3.0  04/08/94

  - Type HELP package_name for help on specific Starlink packages
  -   or HELP PACKAGES for a list of all Starlink packages
  - Type HELP [command] for help on ICL and its commands

ICL> HELP PACKAGES

PACKAGES

   The following ADAM applications packages are available from Starlink:

   Standard Packages:
    CATAPP    -  Catalogue applications
    CONVERT   -  Data format conversion.
    FIGARO    -  General data-reduction.
    KAPPA     -  Image processing.
    SST       -  Simple Software Tools
    UTILITIES -  Miscellaneous useful tools

   Option Packages:
    ASTERIX   -  X-ray data analysis.
    CCDPACK   -  CCD data reduction
    DAOPHOT   -  Stellar photometry.

    ... etc...

  Additional information available:

  ASTERIX    CATAPP     CCDPACK    CONVERT    DAOPHOT    FIGARO     KAPPA

    ... etc...

Topic? KAPPA

KAPPA
```

```
  KAPPA---the Kernel APplication PAckage---currently comprises applications for
  general image processing, many of which will function with data of
  dimensionality other than two; data visualisation, with flexible control of
  the location and size of pictures; and the manipulation of NDF components.

     ... etc...

  To make the commands of KAPPA available, type:

      ICL> KAPPA

Topic?
ICL> HELP KAPPA

KAPPA

  KAPPA---the Kernel APplication PAckage---currently comprises applications for
  general image processing, many of which will function with data of
  dimensionality other than two; data visualisation, with flexible control of
  the location and size of pictures; and the manipulation of NDF components.

     ... etc...

  To make the commands of KAPPA available, type:

      ICL> KAPPA

Topic?
ICL> KAPPA

 --      Initialised for KAPPA      --
 --   Version 0.8-SU, 1993 January  --

 Type HELP KAPPA or KAPHELP for KAPPA help

ICL> HELP KAPPA

Help

   Welcome to the KAPPA online help system.  Here you can display
   details about a specific KAPPA command or more-general information
   such as what KAPPA can do and how to use it.

   ... etc...

  Additional information available:

  ADD        APERADD    BLINK      BLOCK      CADD       CDIV       CENTROID
  Changes_to_KAPPA      CHPIX      Classified_commands   CLEANER    CMULT

   ... etc...

Topic? ADD
```

```
ADD

  Adds two NDF data structures.

  Usage:

     ADD IN1 IN2 OUT

  Description:

     The routine adds two NDF data structures pixel-by-pixel to produce
     a new NDF.

  Additional information available:

  Parameters Examples    Notes      Authors    History

ADD Subtopic? PARAMETERS

ADD

  Parameters

    For information on individual parameters, select from the list below:

    Additional information available:

    IN1        IN2        OUT        TITLE

ADD Parameters Subtopic? IN1

ADD

  Parameters

    IN1

      IN1 = NDF (Read)
         First NDF to be added.

ADD Parameters Subtopic?
ADD Subtopic?
Topic?
ICL> ASTERIX

      ASTERIX  is not installed at this site.
   If you really want it, contact your Site Manager.

ICL> EXIT
%
```

5 Summary

The Starlink Software Librarian maintains:

/star/etc/login which defines environment variables required by *standard* and installed *option* packages.

/star/etc/cshrc which defines Shell Package Startup Commands for *standard* and installed *option* packages to be run from the shell.

ICL_LOGIN_SYS an ICL command file, automatically LOADed at ICL startup, which defines ICL Package Startup Commands and basic help for all Starlink packages. (In the case of non-installed *option* packages, the Startup Command will inform the user than the package is not available.)

ADAM_PACKAGES which is a Starlink HELP file giving basic information about all available packages.

If required, Site Managers provide and maintain:

LADAM_PACKAGES analogous to ADAM_PACKAGES but for *local* packages.

Package Administrators provide and maintain:

Package Startup Script a shell script defining commands to be used when running the package directly from the shell.

Package Definition Command File an ICL command file defining commands and help topics to be used when running the package from ICL.

Package Help File a Starlink HELP file providing detailed help information about the package.

References

[1] J A Bailey, *ICL — The New ADAM Command Language - Users Guide*, Joint Astronomy Centre.

[2] M J Currie, *KAPPA — Kernel Application Package*, Starlink User Note 95.

[3] M D Lawden, *Starlink Software Management*, Starlink General Paper 20.

[4] A J Chipperfield, *ADAM — Unix Version*, Starlink User Note 144.

[5] P T Wallace, *HELP — Interactive Help System* Starlink User Note 124.

[6] A J Chipperfield, *ADAM — Interface File Reference Manual*, Starlink User Note 115.

[7] A J Chipperfield, *ADAM — Installation Guide*, Starlink System Note 44.

6 Document Changes

This document has been reformatted and references to the VAX/VMS setup removed. There are no material changes for the Unix systems.