United Kingdom Infra Red Telescope

Joint Astronomy Centre

Royal Observatory Edinburgh

UKIRT Data Reduction: Requirements

Alan Bridger, Frossie Economou and Gillian Wright

orac003-udrr Version: 01 **Original: 11 July 1997 Modified: 17 November 1997**

This document sets out a specification and list of requirements for the UKIRT Data Reduction System, part of the UKIRT ORAC (Observatory Reduction and Acquisition Control) project.

1.0 Introduction

1.1 Purpose

This document is intended to outline the specification of the UKIRT Data Reduction system and provide a list of requirements. The requirements list will consist of both user requirements and software requirements. It is intended to be read by anyone with an interest in the UKIRT Data Reduction system.

1.2 Scope

The UKIRT Data Reduction system is intended to run at the telescope, reducing data as they are acquired, by all current and planned common-user instruments[1], in near real-time. The main aim is to provide feedback to the observer(s) that is sufficient to allow sensible decisions to be made about the quality of the acquired data. However, where possible this should also be the only time the data needs to pass through the relevant reduction stages.

It is not intended that this system should provide completely reduced, publishable, data, however it should be flexible enough to allow extension of the reduction pipeline, and to accomodate the (possibly new) reduction needs of all planned future instrumentation.

1.3 Overview

The rest of this document will briefly describe the UKIRT Data Reduction system and its relationships to other systems. A comparision with previous similar systems will also be made. The system context will be described and a brief design presented.

Finally a list of specific user and software requirements will be given.

1. Except for CGS3.

2.0 General Description

2.1 Relationship to Other Projects

The UKIRT Data Reduction system is one part of the UKIRT Observatory Reduction and Acquisition Control (ORAC) project. Reference [O-1] gives an overview of this project and summarises the relationships.

2.2 Predecessors

This project will build on the experience gained with the CGS4DR [X-10] project, which it will replace at the telescope. It will also replace the IRCAMDR system. CGS4DR was extremely successful, but it was the first of its kind and as a consequence many years of use have revealed a number of flaws. The experience gained with the system will serve as input to the new system. In addition the move of astronomical computer systems from VMS to Unix, and the appearance in recent years of a number of new tools, such as graphical user interface builders and scripting tools (*e.g*. tcl/Tk, perl, python), provide an opportunity to reimplement the aims of CGS4DR in a cleaner, more supportable, and more extensible manner.

2.3 Function and Purpose

In summary the main functions of the UKIRT Data Reduction system are:

1. To reduce acquired data from all common-user UKIRT instruments to a stage where the quality of the data may be assessed, in near real time.
2. To be extensible to future new instruments.
3. To be modifiable in the light of changed reduction requirements.
4. To be easy to use by novice observers at 14000 feet altitude.
5. To concentrate the support requirements of the UKIRT staff on the on-line aspects of the data reduction, not the algorithms of individual steps.

2.4 Environmental Considerations

The system shall be designed to run on Sun workstations running Solaris v2.5 (or later). Where possible the system shall be made portable (a) to other Unix systems and (b) to non-Unix operating systems, but this should not be allowed to interfere with the production of the system.

The system should be designed with a range of users in mind - expert support scientists, instrument hardware engineers, both novice and expert observers.

Where possible it should use standard supported software tools and astronomical data reduction packages. Where a scripting language is required it should be one in common use and with external support. Any modifications to it should be kept small and, if possible, fed back into the standard support sctructure.

Where possible existing algorithms from commonly used data reduction systems should be used. If new or modified algorithms are required then these should, if possible, be fed back into the standard package.

2.5 Relationship to other systems

Figure 1 shows a context diagram for the UKIRT Data Reduction System.

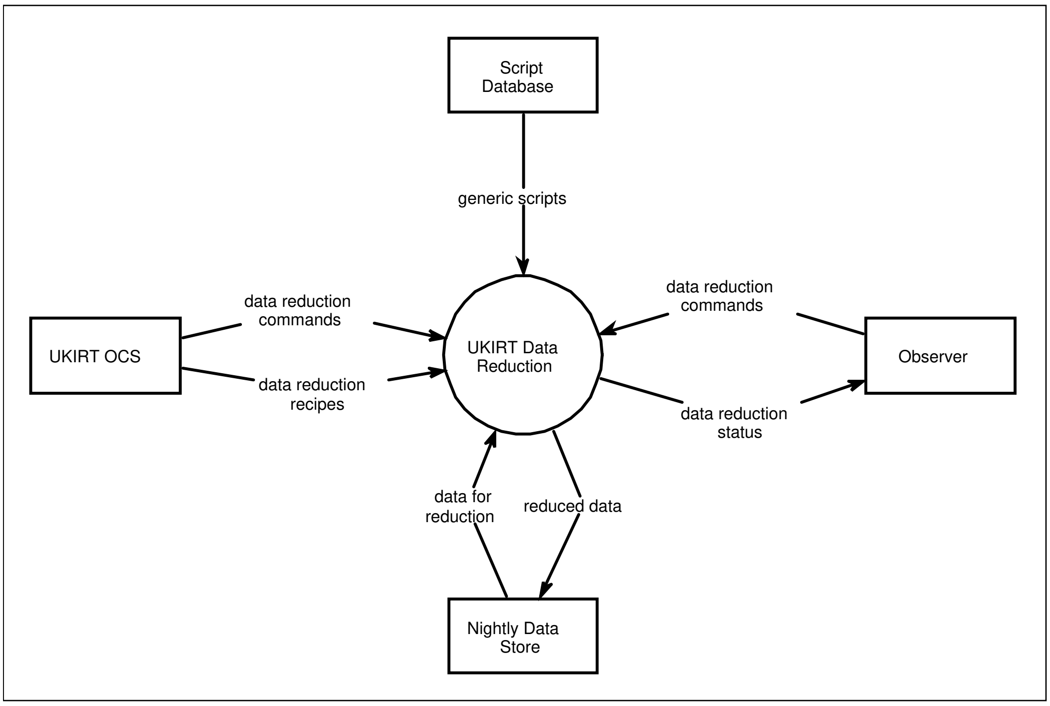

FIGURE 1. Context Diagram for the UKIRT Data Reduction system

In this diagram the external entities are:

UKIRT OCS. The UKIRT Observatory Control System. This is also part of the ORAC project. In the context of the data reduction system it will provide commands to the system to configure it and will also inform the system of new data to be reduced. It may supply **data reduction recipes** to the system, but these may be embedded in the **data for reduction**.

Script Database. This is a store of generic scripts that will be used by the system to create scripts that are specific to the data arriving. It is not clear yet that this store is required (the scripts might be generated "on the fly" by the system) and perhaps it might be regarded as part of the system. If it is required and is not regarded as part of the system then a separate system must be provided to create it.

Observer. This represents the observer who is most probably at the telescope, but might also be remote. The observer may play a passive role - simply monitoring the **data reduction status** and the display of reduced data, or they may also supply commands to the system, *e.g.* to change a configuration, interrupt the reduction or re-reduce data.

Nightly Data Store. This is where the data reduction system will find the data that are to be reduced and also where it will write the reduced data.

The data flows on this diagram are:

data reduction commands. These are commands sent to the system which will control it (e.g. **start, stop, abort, pause, continue**), or configure it, or to change the list of data that the system is to reduce.

data reduction recipes. This flow represents the information that tells the system how to reduce certain data or types of data. It is likely that a recipe will map to a script to be used to perform the actual reduction. The format of this flow is not yet decided - a recipe may simply be a name, or it may be a simple sequence of names. It is possible that the recipe might be embedded in the **data for reduction,** in which case this flow would not appear explicitly at this level.

generic scripts. This flow presents an actual script, written in a scripting langauge, which may be used to reduce certain types of data. It is "generic" in the sense that it does not have specific information about the data in it (*e.g.* names or numbers of files). It is possible that this flow will not exist at this level.

data reduction status. This is the feedback to the observer of how the data reduction is proceeding. It also represents the response to certain **data reduction commands** sent to the system.

data for reduction. This represents the data that are to be reduced by the system. This may be "raw" acquired data or it may be products of previous reduction processes.

reduced data. This represents the data written by the system as a result of a reduction process.

2.6 General Constraints

Some of these are set out in Section 2.4. In addition it is assumed that the developers will develop in a standard programming language (Fortran, C, Java, allowed) and if an "environment" is required then *Drama* is recommended. Reasons for using others should be provided and such use agreed.

2.7 Model Description

An initial attempt at describing how the system will operate is shown in Figure 2. In addition this section attempts to flesh out that diagram.

In this model the idea is that the system maintains an internal queue of datasets to be reduced. Entries to this queue are made in some way as each dataset is stored to disk (see the flow **data reduction commands**). Then as each dataset is recovered from this queue (first-in, first-out) the recipe associated with it is "parsed" by **Parse DR Recipe.** This may be as simple as converting a name into the name of a script to be used, but might be more complex. The information derived from this (**recipe info**) is sent to **Generate Specific Script**. This process also knows about the configuration of the system (the flow **drs setup**) and has access to information about datasets that have been previously reduced (**index information**). The process uses these inputs to generate a **specific script** which pertains to the data that are to be reduced. This script might be generated "on the fly" or may be derived from **generic scripts**. The **specific script** is sent to **Execute Script** which has responsibilty for running it. **Execute Script** might be as simple as an existing scripting language interpreter.

This is very much an initial design and does not yet meet all of the requirements set out in Section 3.0. However, it does give an idea of the current model.

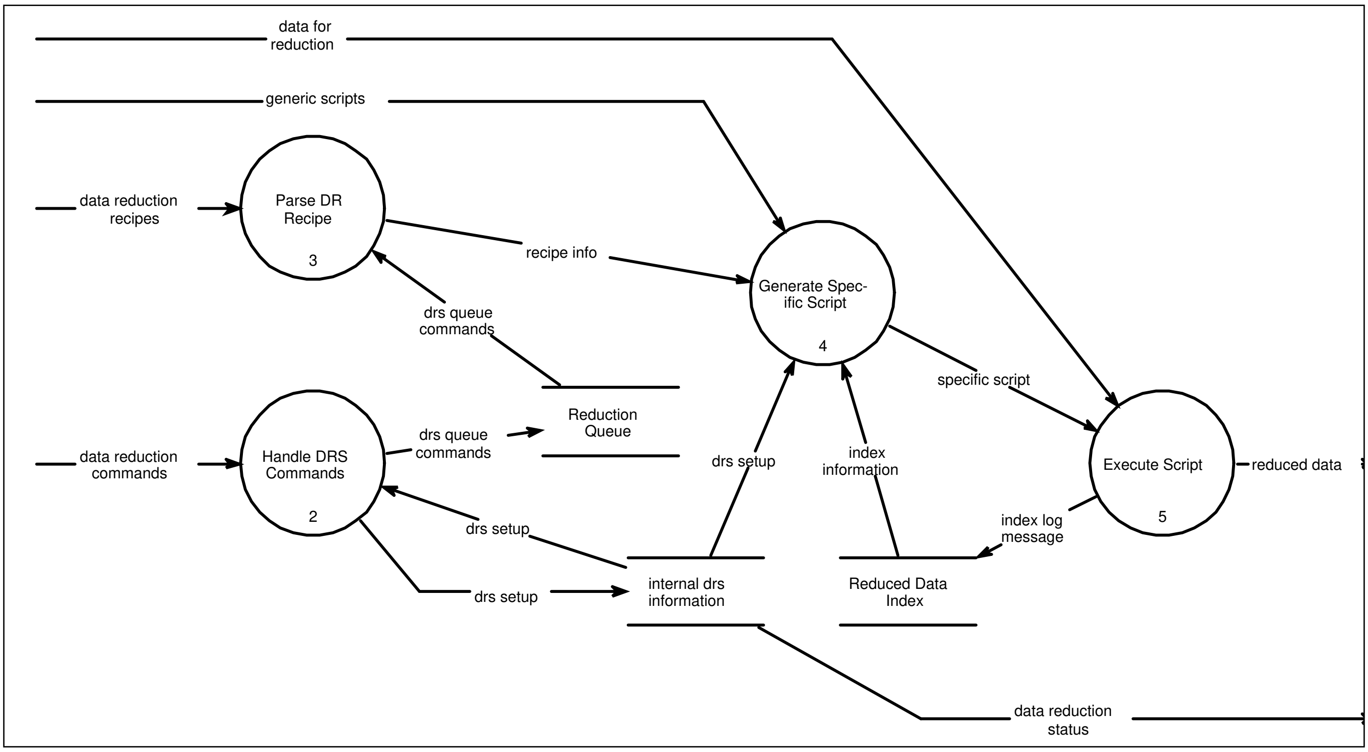

FIGURE 2. Level 0: UKIRT Data Reduction

3.0 Specific Requirements

3.1 Functional Requirements

This is a list of the overall functional requirements that the data reduction system must meet.

FD1. Complete replacement of all CGS4DR and IRCAMDR on-line reduction algorithms which are needed to determine results of observing.

FD2. Make extensive use of existing data reduction software (not CGS4DR for support reasons).

FD3. Extensible and flexible *i.e.* must separate data reduction algorithmic code from data reduction pipeline control.

FD4. Must run on all Starlink supported data reduction platforms: Solaris, Linux, Digital Unix

FD5. Must run stand-alone.

FD6. Must run automatically *i.e.* must recognise and reduce an incoming sequence of observations according to a pre-selected recipe with no/minimal input from the user.

FD7. Must have on-line control - interruptible and configurable.

FD8. Must provide feedback ("see" scripts as they execute?)

FD9. Data format (replaced by ID2.)

FD10. Capable of concurrent operation.

FD11. Must be robust - must handle script crashes in a sensible way.

FD12. Problems with the reduction must not affect acquisition.

FD13. Must handle both error bars and quality arrays.

FD14. Must reduce data taken with different array sizes and sub-array sizes.

FD15. File header of final reduced data should indicate how it was reduced.

FD16. Must allow for both UKIRT observatory reduction scripts and user defined scripts.

FD17. Must auto-plot results of reduction and allow additional user specified plots.

FD18. Must recognise exposure types: stare, nd-stare, chop, and handle data appropriately for bias and dark subtraction.

FD19. Must be extensible for new observing sequences that might be needed.

FD20. Imaging reduction: basic functionality to be delivered with UFTI (need not be fully automated in the first instance).

FD21. Spectroscopy reduction: to be delivered with Michelle.

3.2 Initial Script Requirements

This is a list of the initial recipes or scripts that will be required in order to meet the data reduction requirements of existing instrumentation. The system must be capable of reducing data taken using the following observing sequences:

SD1. Beginning of the night instrument checkout sequence.

SD2. Point source photometry where the flat is derived by median filtering jittered source frames.

SD3. Point source photometry where the flat is derived from other jittered frames, and where the source may or may not have also been jittered.

SD4. Point source photometry where the flat is derived from observation of a calibration source and where the source may or may not have been jittered.

SD5. Images of crowded fields where the flat is derived from a jittered sky frame or from a calibration source, and where the field image may or may not also have been jittered.

SD6. Images of large areas made from a mosaic of smaller frames with flat field derived from median filtered jittered sky frames or from a calibration source.

SD7. Imaging polarimetry data taken using either of the two the standard sequences of waveplate and telescope positions.

SD8. Fabry-Perot images.

SD9. Spectroscopy of point sources taken by nodding the telescope between two positions on the slit, with flat derived from observation of a calibration source or no flat.

SD10. Spectroscopy of point sources with chopping and nodding along the slit, with flat derived from observation of a calibration source or no flat.

SD11. Spectroscopy of point sources with multi-position nodding (up to 5 positions?) with flat derived from observation of a calibration source. (it doesn't make sense to have no flat for this).

SD12. Spectroscopy of small extended sources taken by nodding along the slit with flat derived from observation of a calibration source.

SD13. Spectroscopy of extended sources taken by nodding to blank sky with and without an equal number of sky and object frames and with flat derived by observation of a calibration source.

SD14. Spectroscopy of point and small sources taken in cross-dispersed mode (for UIST), method of flat fielding tbd.

SD15. Spectroscopy of extended sources taken with integral field unit (for UIST and perhaps for CGS4?), method of flat fielding tbd.

SD16. Spectro-polarimetry data taken using any of the three the standard sequences of waveplate and telescope positions.

For images the following general calibrations and corrections must be available:

SD17. Photometric calibration in real time - set and use a zero point to monitor photometric accuracy by deriving a magnitude for each point source observed.

SD18. Photometric calibration of an entire nights data by determining zero point and extinction curves from photometry of standard stars.

SD19. Photometric calibration of point sources assumes that source is centered in "photometric box" on the array at the start of the observation. User defined photometric box location allowed.

SD20. Flux calibration (choice of units) for images of extended sources.

SD21. Correction for rotation of array to provide images aligned accurately N-S.

SD22. Pixel scale or co-ordinates on X and Y axes of image.

SD23. For all jittered or mosaic images offsets between frames to be determined from either the telescope offsets or from a source in an identified area in the overlap region.

SD24. Sky subtraction and interpolation for mosaiced images.

SD25. measurement of FWHM and peak counts for point source images.

For spectroscopy the following general calibrations and corrections must be available:

SD26. Wavelength calibration - based on auto-identification of arc or sky lines, where possible.

SD27. Correct for curvature/tilt of slit in spatial direction.

SD28. Correct distortion of spectra in dispersion direction for extended sources.

SD29. optimal extraction of point source spectra (and hence distortion correction) for all varieties of point source observing described above (assumes star is centred on "spectroscopy row" at start of observation, user defined row to be allowed also).

SD30. ratio by normalised stellar spectrum to remove atmospheric features.

SD31. flux calibration assuming magnitude of ratioing star is known.

SD32. polynomial fitting to and subtraction of residual sky emission lines.

SD33. cross correlation prior to ratioing by atmospheric correction star.

SD34. removal of ripple caused by poor seeing/image quality/guiding and sampling of spectra by stepping the detector.

For engineering purposes the following general analysis tools need to be easy to use:

SD35. two row and two column line fitting on spectroscopic data that has not been wavelength calibrated, with output to an engineering file.

SD36. Histograms of data frames and areas of frames which are meaningful for checking array readout modes: *e.g.* for the 256x256 InSb array it is necessary to be able to compare histograms of data from alternate columns of the array.

SD37. creation of bad pixel masks from both data and error arrays.

3.3 Interface Requirements

These are requirements placed on the system by the need to interface to external systems.

ID1. The command interface to the UKIRT OCS must be determined, preferably in conjunction with the OCS developers.

ID2. Data format for both input and output - TBD, but NDF or FITS seem to be the only reasonable contenders. The requirements of the data reduction system will drive the requirements of the storage system.

ID3. How recipes are obtained needs to be defined - so that the acquisition system can meet it.

3.4 Documentation Requirements

This is a list of the documentation requirements for the system. Similar requirements are placed on other aspects of the ORAC project.

DD1. A programmers guide is required both on paper and html.

DD2. A user guide is required on paper and html. This should include a guide to creating new recipes.

DD3. A guide to writing generic scripts. This may assume some knowledge of the scripting language chosen.

DD4. There should be a guide describing what is required to add a completely new reduction pipeline - *e.g.* for a new instrument or observing mode.

DD5. User guide to be accessed from "help" in GUI.

DD6. Link to instrument cookbook written by support scientist (*e.g.* CGS4 manual).