United Kingdom Infra Red Telescope

Joint Astronomy Centre

Royal Observatory Edinburgh

Observation Preparation System: Requirements

Alan Bridger, Gillian Wright and Frossie Economou

orac004-opsr Version: 02 **Original: 15 July 1997 Modified: 17 November 1997**

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

1.0 Introduction

1.1 Purpose

This document is intended to outline the specification of the UKIRT Observation Preparation 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 Observation Preparation System.

1.2 Scope

The UKIRT Preparation System is intended to aid observers (including engineers and support scientists) in the preparation of *Observation Definitions* to be used in the new UKIRT Observatory Control System. It will allow these definitions to be created at the telescope (online) and also away from the telescope (offline), given a certain (minimal) set of computing requirements. The *Observation Definitions* maybe submitted to and stored in the Observation Database, to be maintained at the telescope. The system will allow verifiable users to submit the definitions directly to the Observation Database. It will also permit some pre-submission verification of the definition, and the automatic derivation of some of the components of the definition.

1.3 Overview

The rest of this document will briefly describe the UKIRT Observation Preparation System and its relations ships to other systems. A comparison 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.

2.0 General Description

2.1 Relationship to Other Projects

The UKIRT Observation Preparation 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 existing **UKIRT_PREP** system. This system runs only on VMS systems running the ADAM system, and uses a non-portable, simple menu system. It can also only create instrument configurations (*Config*s) and observing sequences (*EXEC*s). The new system will be a complete replacement for **UKIRT_PREP**, but will also go much further in the area of observation preparation.

2.3 Function and Purpose

In summary the main functions of the UKIRT Observation Preparation System are:

1. To create Observation Definitions, and the various components of Observation Definitions, given user input.
2. To allow automatic derivation of certain items in Observation Definitions.
3. To enforce some verification of Observation Definitions, or their components.
4. To allow a certain amount of verification of Observation Definitions, or their components, beyond that enforced in (3) above.
5. To allow submission of Observation Definitions to the Observatory Observation Database, by authorised and verified users.
6. To allow all of the above to be done both online - at the telescope - and offline - away from the telescope. In the latter case a certain minimal set of computing requirements will be defined.
7. To allow Observatory Operation staff to modify some of the items requested in the definitions and the rules used to derive the values of some items.

2.4 Environmental Considerations

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

The system shall be designed to be accessible from any reasonable workstation connected to the internet. As a guideline any system that can run a graphical web-browser or an implementation of the Java Virtual Machine should be a suitable minimal requirement.

The Observing Database will be located at the telescope.

2.5 Relationship to other systems

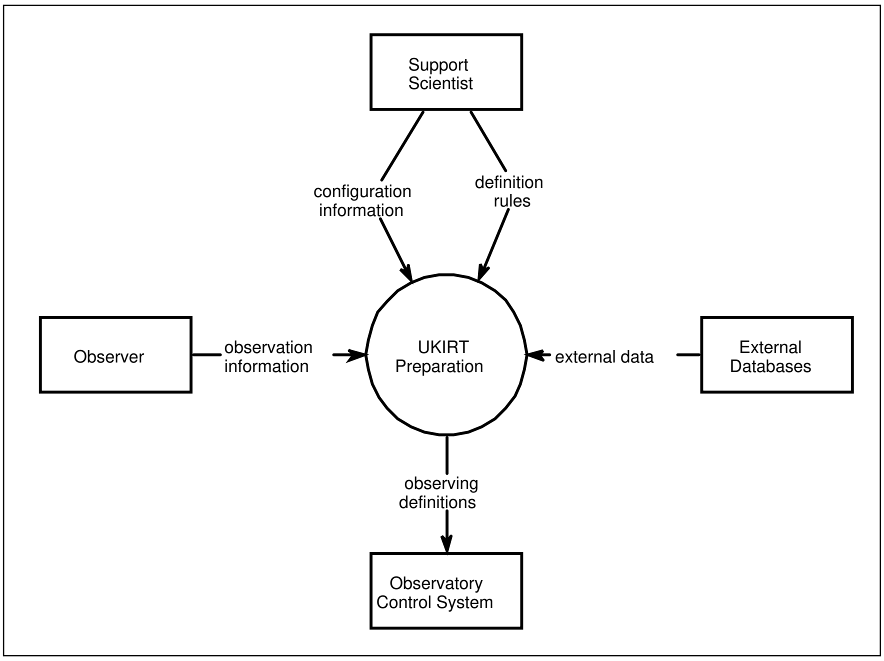

FIGURE 1. Observation Preparation System context diagram

Figure 1 shows a context diagram for the UKIRT Observation Preparation System. In this diagram the external entities are:

Observer. This is the observer who wishes to create *Observation Definitions*. The observer may be local (to UKIRT) or remote. This also covers UKIRT operations staff wishing to use the system to create standard or engineering *Observation Definitions*.

Support Scientist. This represents the UKIRT Operations support scientist who wishes to maintain the system by updating its rules for deriving certain quantities or to modify its appearance to take account of changed instrument abilities.

External Databases. The system may take advantage of access to external data bases (local or remote) in order to present the **Observer** with useful information about their targets or instruments, or other such information. Some of this information may also be fed directly into an *Observation Definition*. This entity represents all possible sources of such information.

Observatory Control System. This is the system that will read *Observation Definitions.* The database of *Observation Definitions* is considered to be part of the Observing Preparation System.

The major dataflows in Figure 1 are:

observation information. This is the information provided by the **Observer** about his or her planned observations. This may be broken down into various components.

configuration information. This is information provided by the **Support Scientist** in order to modify the appearance of the Preparation system to reflect changes in the abilities of instrumentation, e.g. different filters in a camera, different gratings in a spectrometer.

definition rules. This represents maintenance information provided by the **Support Scientist** that is used as the rule base for the derivation of the values of some items in *Observation Definitions*.

external data. This is the flow of data from external databases that is obtained by the system to provide aid in constructing *Observation Definitions*.

observing definitions. This represents the *Observation Definitions* as they are exported from the preparation system to be used by the **Observatory Control System**.

2.6 General Constraints

Some of these are set out in 2.4. In addition it is assumed that the developers will develop in a standard programming language (Fortran, C, Java, allowed) and if a “scripting” language is necessary then if possible it should be the same as the scripting language used in other parts of the ORAC project. If not then reasons should be provided and such use agreed.

2.7 Model Description

The initial model for the UKIRT Preparation System is based around the idea of creating a descriptive file for each major UKIRT system (telescope, instrument, data reduction and the overall observation control) that may be used to control (or configure) that system. These descriptions are referred to as a *Target Description* (for the telescope), an *Instrument Configuration*, a *Data Reduction Recipe* and an *Observing Sequence*. These descriptions would then be linked together to form an overall *Observation Definition* which would thus hold all the information necessary to take an observation. The Observation Definition would be completed by a component containing *Scheduling Information* and finally by some header information Exactly what detailed parameters makes up each of these descriptions, and indeed exactly what an *observation* is by this definition will be set out in the more detailed design document for the preparation system.

The software for creating these definitions will be controlled by a friendly graphical user interface that helps observers that are more or less ignorant of UKIRT and its instrumentation through the process of preparing for their observing. It will do this not only by being easy and relatively obvious in its use but also by prompting with sensible default values for items and standard descriptions for common observing methods.

These observation definitions would then be read in by the control systems at the observatory and acted up on.

The preparation system will be able to be run at the telescope, but, more importantly, also at the observer’s home institution (or indeed on his or her home PC), with the final prepared *Observation Definitions* being submitted to a database at the UKIRT ready for use at the telescope. These requirements for portability and remote use (and in addition the realisation that the version of the system being used by a remote user must be up to date) place strong restrictions on the sort of solution found to the problem.

Finally the remote submission of definitions also implies the setting up of a server and database at UKIRT and putting in place appropriate security and authentication protocols.

Though there are a number of ways to solve the problems it seems that the use of Java would seem to be the best. Java is portable (to any architecture that runs a Java Virtual Machine, which covers most likely to be in use by observers). It is object-oriented which suits the nature of the problem at hand, and it has built in graphical user-interface and networking functions. Finally it supports the idea of "applets" which may be run from the ever-present web-browsers, thus also solving the problem of keeping software up to date). (However, use of stand-alone Java applications and internet "push" technology would seem to be a more powerful and flexible solution to this problem - the user does not have to be online to run the application and the security restrictions placed on applets are not a problem) Java's major drawback is the learning curve involved, though there may also be concerns about use of a relatively young technology and its future success and growth.

Other solutions to these problems would be likely to involve the use of a standard scripting language (perl, tcl or python) in conjunction with the Tk graphical interface toolkit. These languages are also more or less "portable" and also object-oriented to varying degrees. There would be concern here about the future of these languages too (perl has a large following, but tcl and python might both be a little subject to "fashion trends") and in addition there is always the maintainability worry about the use of scripting languages - it is often much easier to write a quick script than to read and modify a large existing one. However, the final reason for rejecting these routes probably rests with Tk. Though Tk helps a lot in the quick and easy production of user-interfaces, it is not easy to get Tk interfaces to do *exactly* what you want, mostly because it has simplified the production of a user-interface and the final touches that make an interface good usually involve subverting this simplicity - often with some difficulty.

Having set out this broad solution to the problems it is of course of no surprise that the Gemini Observing Tool (essentially the equivalent to the UKIRT Preparation System) has already solved more or less the same problems in more or less this way. The Gemini Observing Tool is a Java-based system that will be used by Gemini observers to prepare observations for their Gemini Observing.

Thus in the interest of saving effort in the ORAC project and of achieving some degree of comonality in observing on different telescope available to UK observers it seems obvious to consider, with permission from the Gemini project and OCS group, use of the Gemini Observing tool as the UKIRT Observation Preparation System. There are some differences in the concepts behind the two systems, some of which could affect the way observing at UKIRT is handled, and so careful consideration of how the tool could be used on UKIRT is required. This will be covered in a separate document.

3.0 Specific Requirements

3.1 Functional Requirements

FP1. Replaces current functionality of UKIRT_PREP, except where modified or extended by other requirements.

FP2. Checks that sensible values for parameters are entered, including name.

FP3. Must run at UKIRT.

FP4. Can also be run from observers home institution with minimal software requirements.

FP5. Produces output that can be fed directly into the UKIRT observing system.

FP6. Must be compatible with all current and planned future instruments.

FP7. Values of all items must be viewable, including "hidden" items.

FP8. Ability to keep local copies of definitions and descriptions.

FP9. Format of stored components tbd. (ASCII?)

FP10. Use "look and feel" system that is in common use.

FP11. All components will have both "standard" (observatory maintained) and "user" varieties.

FP12. Must be sufficiently modular to allow addition of target and scheduling/priority information and data reduction recipes for OCS at a later stage.

FP13. Extensible to read in parts of observation definition from another source, *e.g.* perhaps a PATT proposal.

FP14. Ensure that all observations defined are possible with the instrument (within reason).

FP15. Enables both automatic and manual specification of various instrument-specific information in the Instrument Configuration.

FP16. Support use of rules to determine "automatic" values for items.

FP17. Detailed specification for each instrument will be written - Instrument Configuration items which may be determined automatically and the rules for doing so tbd. Some examples are: optimum orders based on wavelength and grating, exposure time based on magnitude.

FP18. Auto-specification “rules” to be changeable by a support scientist.

FP19. Enables the selection of standard Observing sequences without the use of a text editor or knowledge of a scripting language.

FP20. Supports user creation of non-standard observing sequences (*e.g.* for large mosaics) without needing knowledge of scripting language, using a Visual Editor.

FP21. Selects appropriate data reduction recipe for observing sequence chosen - *i.e.* tie standard observing sequences to standard recipes (by the time Michelle is delivered).

FP22. Supports creation of new data reduction recipes from building block components

3.2 Security Requirements

The following security issues will be addressed in the design and appropriate solutions found:

SP1. Determine who will be allowed to access and run the preparation system (and how).

SP2. Determine who can access the preparation database (which may contain sensitive co-ordinate information). (And how to prevent unauthorised access.)

SP3. Determine who can access the astronomical data.

3.3 Interface Requirements

The preparation system interfaces explicitly with the external entities outlined in Section 2.5, but it also interfaces indirectly with other systems, specifically the telescope control system, the instrument control systems and the data reduction system, through the output files it creates. These interfaces must be defined by this system, in conjunction with the other systems, but as some backwards compatibility is required further requirements are placed.

IP1. Interface to OCS must be defined, in conjunction with OCS developers. (Format of stored files + observation sequence and scheduling information).

IP2. Interface to DR system must be defined, in conjunction with DR system developers. (This is how data reduction recipes are defined, stored and addressed).

IP3. Interface to telescope control system must be defined, in conjunction with TCS support team.

IP4. Interface to instrument systems must be defined, in conjunction with current support teams and present instrument developers.

IP5. There must be an option to store observation sequences in the current format, for use by CGS4 and IRCAM as they stand.

IP6. There must be an option to store IRCAM3, CGS4 and UFTI configs in their current format.

3.4 Documentation Requirements

This is a list of the documentation requirements for the system. Similar requirements are placed on other parts of the ORAC project, and the numbering is the same where they are duplicated.

DP1. A programmers guide is required both on paper and html.

DP2. A user guide is required on paper and html.

DP3. User guide to be accessed from "help" in GUI.

DP4. Link to instrument cookbook written by support scientist (*e.g.* CGS4 manual).