United Kingdom Infra Red Telescope

Joint Astronomy Centre

Royal Observatory Edinburgh

UKIRT Scheduler: Preliminary Design

Alan Bridger and Gillian Wright

orac021-schp Version: 01 **Original: 1 February 1999 Modified: 22 February 1999**

This document sets out a preliminary design for the Observation Scheduler, part of the Observatory Control System, part of the ORAC (Observatory Reduction and Acquisition Control) project.

1.0 Introduction

1.1 Purpose

This document is intended to outline a preliminary design of the Observation Scheduler task in the UKIRT Observation Control System which will meet the requirements set out in [O-20]. It is intended to be read by anyone with an interest in the UKIRT Observatory Control system. It is assumed that the reader is familiar with the ORAC project, and in particular the OCS component. An overview can be found in [O-12].

Note that much terminology, and probably no small part of the design, in this document is undoubtedly and unashamedly influenced by other projects, especially the Gemini project.

1.2 Scope

The main function of the Observation Scheduler is to create and possibly maintain an optimised schedule for observing that could be used at the telescope. For short-term (nightly) scheduling this will be a queue of individual observations. For longer term (weeks, semester) scheduling this will more likely be a schedule of Observing Plans or Science Programs. In addition the Scheduler must be able to provide the “next” appropriate observation in a plan to clients requesting it (typically the Chooser). Finally the Scheduler may also provide other functions that aid in the preparation of observing plans - *e.g.* simple sanity checks on the plan.

1.3 Overview

The rest of this document will briefly describe the UKIRT Observation Scheduler and its relationships to other systems. A comparison with previous similar systems will also be made. The system context will be described and the major functions of the system outlined. A preliminary system design will be presented.

2.0 General Description

2.1 Relationship to Other Projects

The Observation Scheduler is part of the UKIRT Observatory Control System, which in turn is one part of the UKIRT Observatory Reduction and Acquisition Control (ORAC) project. References [O-1] and [O-12] give an overview of this project and summarises the relationships.

2.2 Predecessors

There is really no predecessor to the Scheduler in the existing UKIRT system. A similar system is being designed contemporaneously by the Gemini Project [G-18] and there are other similar systems either planned for or in use by other telescopes. These are considered in more detail in Section 8.0.

2.3 Function and Purpose

In summary the main functions of the Observation Scheduler are:

1. To construct a schedule for a nights observing using information given in each observation and environmental information.
2. To execute this schedule if requested.
3. To allow the schedule to be edited by the Observer.
4. To allow the importance of the parameters used to generate the schedule to be modified.
5. To notify the Observer of any problems discovered in the schedule.
6. The Scheduler will be required to perform medium or long-term “strategic” scheduling.

It should be noted here that the Observation Database is not defined as part of the Scheduler. The design of the Observation Database is tackled in [O-19]. In addition it should be noted that the Scheduler is not responsible for updating the Observation Database with information regarding the status of Science Programs or their Observation Definitions. It is responsible only for creating plans *using* the information in the Observation Database.

2.4 Environmental Considerations

The Scheduler must run at the UKIRT telescope for interactive observing (in service modes, queue-scheduled modes and possibly classical observing modes) but part of it must also be able to run anywhere as an aid to planning a nights observing. Portability is desired as is remote access to the Observation Database. Long term scheduling tools, should be available at the TAG meetings.

2.5 Relationship to other systems

The Scheduler will interact with the Observation Database and with the Chooser. When in use to drive observing the Scheduler will supplant the Science Program/Observation Selection part of the Chooser in determining an observation to be executed.

2.6 General Constraints

As well those are set out in Section 2.4 the system should, where possible, be written in Java. This satisfies portability requirements, but also fits in with the use of Java in the Chooser and the Observing Tool. However, if use is made of other scheduling tools (see Section 8.0) then this constraint may need to be relaxed.

3.0 Model Description

In meeting the requirements set out above it would seem that the Scheduler might best be implemented as a number of separate but interconnected tools.

Figure 1 shows a conceptual overview of how the Scheduler fits in with the other systems at UKIRT. Some details of the connections between components are deliberately omitted from this figure so as to reduce the confusion. In particular the essential *feedback* that will form a part of the function of most systems is omitted.

This figure allows us to get an impression of the other observatory systems that the scheduler will interact with and what the scheduler must be able to do. It should not, however, be taken as a formal diagram in any sense.

Figure 1 also demonstrates that the Observation Database (ODB) will perform a central role in the overall system. The Observation Database is considered in a separate document [O-19] so the detailed contents of the ODB are not considered, only a broad overview.

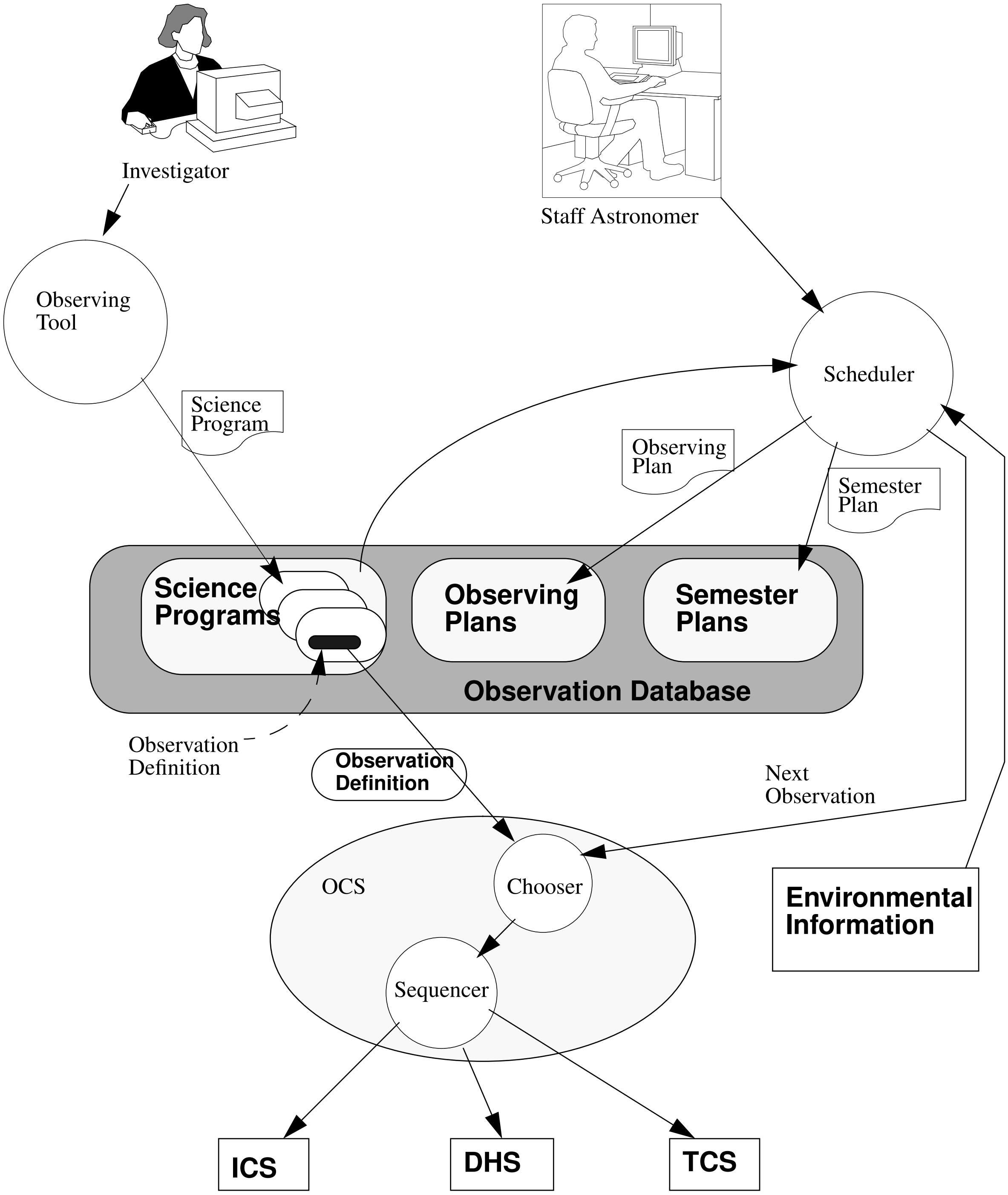
In Figure 1 Investigators (visiting observers, PIs, staff astronomers...) create Science Programs using the ORAC Observing Tool. These will get stored in the ODB. For the purposes of discussion at the moment it will be assumed that these documents also include Phase 1 (proposal) information and TAG gradings and other qualities. Each Science Program will consist of a number of Observation Definitions, describing how to configure the systems and take the data for an Observation.

A staff astronomer (though this could also be an investigator) will then use the Scheduler. The Scheduler can use the information available in the Science Programs, along with Environmental Information (current or predicted conditions) to create Observing Plans. An Observing Plan is a short term (1 or very few nights) plan or schedule of the observations to be taken. It consists of a list of Observation Definitions taken from the Science Programs (details of this will be tackled in [O-19]).

A staff astronomer could also use the Scheduler in a similar (but different in important details) manner to create Semester Plans, or a medium- to long-term schedule of Science Programs and Observing Plans matched to days.

The Observatory Control System (specifically the Chooser) can be run in a mode where it automatically requests the next Observation to be executed from the Scheduler, so the Scheduler must be able to provide the OCS with the next scheduled observation. Omitted from the figure is the idea that the Chooser may also be controlled by an Observer selecting Observation Definitions one by one. The Chooser is then responsible for providing the Observation to the Sequencer which will actually execute it, interacting with the main Observatory Systems (see [O-12],[O-14] for details).

FIGURE 1. Overview of the ORAC Scheduler environment



This figure and the description above illustrates the likelihood that the scheduler will comprise a number of different systems: A user interface to interact with the Staff Astronomer and obtain current Observatory Status and weights; a tool that monitors or requests environmental information; a tool that implements an algorithm to create an optimal schedule of observations; a tool to implement a strategic schedule of Observing Plans; a tool to provide the next observation to the Chooser; a tool that must interact with the observation database in order to make queries and extract information about Observation Definitions, Science Programs and Observing Plans.

These tools are examined in more detail in Section 4.0.

4.0 Design

4.1 System Context

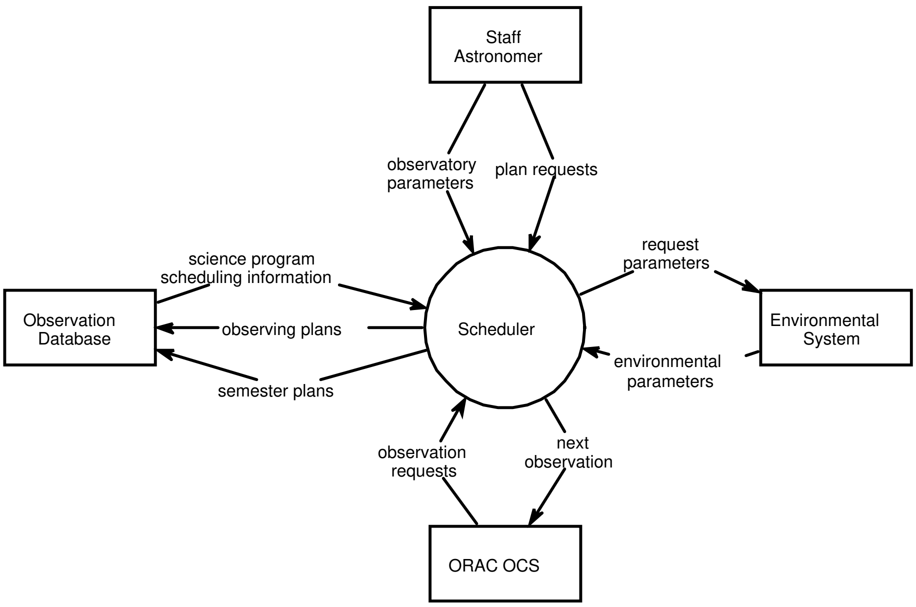


FIGURE 2. Context Diagram for the ORAC Scheduler

Figure 2 shows a context diagram for the Scheduler system. In the diagram are identified the following external entities:

Staff Astronomer. This represents the astronomer who is responsible for deriving schedules. This will most likely be a staff astronomer, but it could be a visiting observer using the system to optimise their own observing plans.

Environmental System. This is system that monitors the environmental conditions at the telescope.

ORAC OCS. This is the Observatory Control System, responsible for coordinating and controlling the observatory systems in order to execute observations.

Observation Database. This is a database comprising: Science Programs as prepared by the Observation Preparation System (ORAC-OT) (Science Programs also define Observation Definitions); Observing Plans created and maintained by the Scheduler,; Semester Plans created and maintained by the Scheduler.

and the following data flows:

observatory parameters. This represents the set of parameters that can be enter as inputs for the scheduler.

plan requests. A command requesting that the Scheduler calculate a new plan, based on the parameters input.

request parameters. This represents a request to the Environment System for the values of current environmental attributes.

environmental parameters. This is the set of current environmental conditions at the telescope.

next observation. This contains a link to the next Observation Definition that should be executed.

observation requests. This is a request for the next observation to be executed.

science program scheduling information. This is the scheduling information from the Science Programs stored in the Observation Database.

observing plans. The flow of observing plans calculated by the Scheduler that are to be stored in the Observation Database.

semester plans. The flow of semester plans calculated by the Scheduler that are to be stored in the Observation Database.

4.2 User Interface

The user interface is what allows the Staff Astronomer to modify parameters in the Scheduler and calculate new Observing Plans and Semester Plans. The user interface will provide the following functionality:

1. Access to the ODB. This shall allow browsing of the ODB and selection of Science Programs for scheduling. It will also allow storage and recall of Observing Plans or Semester Plans to and from the ODB. Access will be authenticated.
2. Input of parameter values to be used in the schedule calculations.
3. Save and restore of parameter configurations.
4. Enable the user to request the calculation of an Observing Plan.
5. Enable the user to request the calculation of a Semester Plan.
6. Provide a display of the calculated plan.
7. Provide crude editing abilities for the plan?
8. Allow an Observing Plan or Science Program to be checked for “sanity”, and highlight any resulting problems.

4.3 Information Input and monitoring

This module gets information about the current or predicted environmental conditions (including local HA), the current observatory setup (instrument availability etc.) and the weighting parameters that should be applied to the scheduling qualities used to decide on a schedule. The details of the parameters to be obtained are outlined in requirement FM2. As can be seen the values of some of these parameters can be obtained from the Observatory System, though most will be input by the user, using the user interface.

4.4 The Program/Plan Handler

This module forms a core part of the system in that it is responsible for handling everything to do with Science Programs and also with the Observing Plans and Semester Plans created. Much of this module will be identical to the similar module in the Observing Tool, though clearly extensions will be required to handle the Observing and Semester Plans.

Another likely extension will be the provision of methods to estimate the time required for a given Observation Definition.

4.5 The Query Handler

This package will be responsible for interrogating the Observation Database using a Query language in order to locate Science Programs or Observation Definitions meeting the appropriate conditions.

4.6 Observing Plan Creation

This will use the Science Programs or Observation Definitions found along with the other relevant information to determine an Observing Plan for the current or requested night. This is the main Schedule Calculation engine. At this stage the algorithm to be used is not decided. There are three main choices:

1. Develop and implement a new algorithm.
2. Re-implement an algorithm from an existing Scheduling package to meet our own interfaces and programming choice (Java).
3. Develop an interface layer so that an existing Scheduling package can be used with little modification (some may be necessary).

Superficially the third of these looks like the most attractive (lowest cost) option, but as always use of existing software must be looked at carefully. Developing the interface may take along time, and this approach may result in a fragile product. The approach to be taken here is:

1. Examine the UKIRT scheduling requirements in some detail and judge each system against them.
2. If one or more systems fully meets them (or goes further) then consider what is involved in interfacing to the system from the Scheduler.
3. If this interfacing looks reasonably tractable for a system then investigate further with a view to using it.
4. If 2 and 3 have not resulted in a system then examine those that fall short in requirements to see if interfacing is tractable.
5. If so then consider extending the system to meet the UKIRT requirements.

6. Finally if there is no candidate at this stage then select the algorithm that best meets the requirements and will be simplest to re-implement and do so.
7. If there is no such system then write our own with the simplest possible algorithm.

Some of the scheduling tools in use are considered in Section 8.0.

4.7 Semester Plan Creation

This will use the Science Programs or Observing Plans found along with the other relevant information to determine a Semester Plan. This will be very similar to Observing Plan creation, except for the following considerations:

1. The Scheduling Quantum will be a "whole night" (an Observing Plan) or a Science Program. Representative Scheduling requirements will need to be derived.
2. Classically scheduled blocks could be handled as "chained nights" or remain as just a Science Program.
3. Queue-scheduling nights could be handled as Observing Plans, or perhaps more sensible as "tied" nights with resource availability noted.
4. Some events will need to be fixed to be specific nights.
5. The rules will be very similar to that for Observing Plan creation, but will differ in detail.

4.8 Observation Provision and Queue Handling

The result of creating an Observing Plan will be a time ordered list of Observations, usually for one night. At the telescope this *queue* will form the basis for automatic observing. This module will handle this queue and must do the following:

1. Respond to a request from a "client" (the Chooser) to send the Next Observation that meets some simple criteria defining which Observatory resources it requires (i.e. it must use a certain instrument and does or does not require the telescope - to be extended to other facility systems like the polarimetry waveplate).
2. Then remove that Observation from the queue, or mark it (in the plan?) as passed for execution. Note that the ODB is not updated at this stage - the Chooser will update the ODB.
3. This module will not wait for the final status of executing that Observation. It will only respond to requests for a Next Observation.
4. The queue must be periodically updated to reflect changing conditions, in particular the time. If its performance is high enough then it is likely that this update will be performed at the time of a request for a Next Observation.
5. If no Next Observation meeting the criteria is available then the Scheduler must return a "None Available" status.
6. Solutions to covering the "None Available" state mentioned above are discussed in Section 7.0.

4.9 Observing Plan sanity checking

Any Observing Plan, and indeed most Science Programs should satisfy some criteria for them to make scientific sense. In particular:

1. Calibration frames will be required. Exactly what depends on the observation type. Most spectroscopy will require a flat field, many will require dark frames, a few will require a bias. It is suggested that the rules for what is required are defined in an instrument specific configuration file.
2. Standard star observations are usually required for spectroscopy or photometry. This is more difficult to automate (ones person's standard star is another person's object...) and it may require that standards are somehow marked as such in the Science Program definitions.
3. There are also some obvious things to catch here too: target visibility, instrument configuration? (should be caught at OT level?)

The sanity checker will run through a given Plan or Program and check each observation to ensure that these criteria are met. Where they are not met the user will be warned of the problem but not prevented from pressing on regardless (to allow the user to continue to have full control).

4.10 Modifications to the Chooser

The design of the Scheduler presented here has some implications for the functionality of the Chooser. These are outlined here:

1. The Chooser should have an Automatic mode in addition to its existing (Manual) mode. The user must be able to switch from one to the other.
2. In the Automatic mode the Chooser will be constantly executing observations. As an Observation completes the next on that Sequencer is requested from the Scheduler. In this mode the Chooser should have a Sequencer running for each instrument available and be able to request the Next Observation for each Sequencer, defining simple criteria of the Instrument to be used and whether or not the telescope is required (plus other facility systems, e.g. the polarimetry waveplate).
3. If no Next Observation is available then when should the Chooser next ask for one?
4. Add to or implement writing back of status information to ODB. It is expected that basic status information writing ("Observation completed" status, date, ...) will have already been implemented, but more may be required in this area.

5.0 The Observation Database

The Observation Database is considered very briefly here, in the context of the Scheduler, but a more detailed description may be found in [O-19].

Briefly, the Observation Database is the repository for all of the Science Programs submitted and for the Observing Plans and Semester Plans created by the Scheduler. Its specification also encompasses the management system that must be used to interact with the database. It will reside at the JAC, possibly at the telescope. The database forms a crucial part of the system and will be divided into a number of areas: The Science Program database, the Observing Plan database and the Semester Plan database. It may also encompass a Phase 1 database. In addition at least the Science Plan database will be further divided into "pre-active", "active" and "retired" sections, indicating the status of the Science Program. During its stay in the active database a Science Program will be updated by the ORAC-OCS with information on its current status.

The Observation Database will be accessible by all UKIRT users and staff, but access will be restricted, requiring authentication to see Programs or Plans. Appropriate staff will normally have rights to see all of the database.

6.0 Accounting

Accounting is not really a function of the Scheduler, however it is briefly covered here in the context of the Scheduler. It is considered in more detail in [O-19].

Briefly in a flexibly scheduled environment (queue-scheduled, service-mode, reactive) the quantum of observing is the Observation (with some extensions to cover observation chaining and grouping) In this environment there is no guarantee at all that all of the observations in an investigators Science Program will be executed together. In fact that will be the exception rather than the rule. Coupled with the fact that the investigator will probably not be present for the execution of most of the observations this means that systems must be in place to track the status of Observations: have they been executed? Was the execution successful? When was it done? Where is the data? The approach taken in ORAC is as follows:

1. The information about the status of the Observation is stored with it, in the Science Program.
2. The handling of the Science Programs, and thus the status information, is the responsibility of the Observation Database and its associated Observation Database Management System. This is described further in [O-19].
3. The Chooser, being the system that has responsibility for executing an Observation, must inform the Observation Database Management System when an observation has been executed, and its completion status. Note that this will also affect the “Program Completion Level” of a program.
4. The Scheduler does not write any of this information. However, it will make use of it when Plans are to be calculated.

7.0 UKIRT Scheduling

The requirements set out in [O-20] outline the scientific requirements for constructing observing plans and semester plans for UKIRT. In this section we outline a general approach to how the problem of constructing plans for UKIRT should be solved.

There are two principal types of scheduler - dispatch schedulers and optimum schedulers. Dispatch schedulers operate by having a group of allowed observations (those that satisfy the current constraints) available at any given time in the night and, on request, selecting the “best” observation from among these. Dispatch schedulers work well when a schedule is well-loaded, that is that there are a lot of observations available more than filling the parameter space that the night will cover. If this is not the case then simple dispatch schedulers become very non-optimum [X-19].

Optimum schedulers take a different approach. They attempt to calculate an optimum schedule for the night (or longer period), in advance of executing the observations. This, certainly in the case of a lightly filled parameter space, will typically create a much more optimum schedule. However, they are typically more computationally intense and they begin to fail when conditions are changing relatively rapidly.

UKIRT's requirements would appear to demand a system much closer to the optimum scheduling paradigm, but when conditions are changeable it is clear that a backup approach must be in place. It should be noted that "changing conditions" might not just refer to the external environment, it could also refer to instrument failure or even human error. Thus some elements of dispatch scheduling would seem to be desirable.

There are a number of possible approaches to this problem including, for instance, deliberately overloading the schedule. However, an approach that has found success for the JCMT and has been tested in simulation by Gemini [G-19] is the creation of more than one "queue" of observations for the night. In implementing the UKIRT Scheduler it would seem that this approach would be the best. In implementing this it would seem that the choices are either to create more than one plan, separated probably by priority, automatically, or to require that the user to create more than one plan. Exactly the approach taken is something that will need to be discussed with UKIRT operations, and also with the board and the TAG.

Finally the actual mathematical algorithm used to create a schedule will depend to some extent on the Scheduling engine chosen. However, early discussions seem to indicate that the use of a simple product of weighted attributes and boolean functions is a model to aim for.

8.0 Other Scheduling Systems

In this section some current systems for scheduling telescopes are considered, particularly with regard to their possible suitability for use as the scheduling engine in the ORAC Scheduler. The requirements set out for the UKIRT Scheduler should be borne in mind, as should the size of the project. Any scheduler chosen for UKIRT should implement a simple and straight-forward algorithm.

8.1 SPIKE

SPIKE [X-23] is a scheduling tool developed by the STSci for long term scheduling of HST observations and subsequently adapted for other space missions, for short-term scheduling and for ground-based telescopes. It is based around the idea of satisfying constraints and is written using CLOS (Common Lisp Object System). It is possible to add customised algorithms (set out as a constraint satisfaction problem). SPIKE is undoubtedly a very powerful scheduling tool and should comfortably meet UKIRT's requirements. However, there is a concern that it would be overkill, and in addition it would introduce a large body of Lisp code and implies a requirement to interact with said code from a Java tool. These concerns indicate that adopting SPIKE might not be the best approach for the UKIRT Scheduler. There may also be some performance concerns if SPIKE were to be used to recalculate schedules frequently during a night, particularly for large numbers of observations. Test runs for the ESO-NTT [X-22] provide numbers of about 30secs for a "typical" Unix workstation for a night's worth of between 30 and 90 "Observation Blocks". An Observation Block is very approximately equivalent to an Observation Definition, i.e. potentially many data frames. So these tests might indicate that performance is not a serious concern.

8.2 ATIS and Automatic Photometry Telescopes

Automatic Photometry (or Photoelectric) Telescopes (APTs) have used simple dispatch-scheduling algorithms for many years. These algorithms operate by executing at each (effective) request for “next observation”. A robust algorithm chooses the next observation to execute by a simple selection procedure from among those that could be done. Though robust and quick the algorithm is often not optimum, especially if the available observations do not sufficiently “load” the night (provide a wide coverage in parameter space). Many of the APTs also use the Automatic Telescope Instruction Set (ATIS) to define their observations and also to encode the selection rules for the observations. ATIS is an IAU standard and is used by many APTs. Though ATIS is worth investigation our commitment to the use of markup languages (whether SGML or XML) would seem to eliminate it as a sensible contender, and basic dispatch scheduling does not seem to be appropriate to the UKIRT requirements.

8.3 The NASA-AMES UKIRT Scheduler Prototype

This tool has been used by UKIRT for scheduling Service mode nights. It was produced by a group at NASA AMES in response to an approach from UKIRT (John Davies). The algorithmic approach is unknown though may be a variant of the one found in the Associate Principal Astronomer [X-18]. The core of the tool is written in C, with shell and perl scripts providing some additional functionality. Access is via CGI on the WWW. UKIRT has found the tool to be of significant value in scheduling service nights, being quite happy with the results. The language of the tool should present no problems (retaining the core as C or porting to Java) and it should clearly be investigated further. Further tests of the tool are planned for upcoming service nights and this testing is very much encouraged.

8.4 The Gemini Scheduler

The approach taken to scheduling the Gemini telescopes [G-19] is to use a set of weighted factors to create a weighting function or “policy” for each observation. This is essentially analogous to the suitability functions used to implement the constraints in SPIKE. Implementation is likely to be in Java. The overall functionality is likely to be far more than is required by UKIRT, but the algorithms may be more suited than those found in the core SPIKE as the tool is aimed at a similar problem. Thus “overkill” might be a worry, but probably less so, and the language used is not a concern. Maintenance may also be less of a concern as the UK is a Gemini partner. The Gemini tool does not yet exist, so the approach here will be based on their timescales and may be collaborative.

8.5 The ESO VLT

The baseline ESO Scheduling tool (SCHED) is based around the SPIKE scheduling engine. Status of the tool at present is unknown, but a prototyping experience using the NTT [X-22] seems to have been positive. In that prototype some ESO-specific modified algorithms were added to SPIKE and a tcl/Tk interface was produced. ESO have plans to re-implement the interface tools in Java (which may be relevant to one of our concerns about SPIKE listed above) but will retain the lisp-based core. If SPIKE were to be chosen as the UKIRT scheduling engine it would be prudent to examine the ESO approach in more detail.

8.6 The Liverpool Telescope

The Liverpool Telescope is a robotic telescope that will be used on La Palma [X-21]. They intend implementing a dispatch-type scheduler [X-20]. As with the APTs mentioned above this type of scheduling is not what UKIRT has planned. However, the scheduling algorithm planned (a weighting function not unlike the Gemini approach) is closer to those used in optimum scheduling and it should be possible to modify such a scheduler to meet UKIRT's requirements for short-term scheduling. The tool is likely to be implemented in Java and is likely to be of a level of functionality reasonably matched to UKIRT's needs. Investigation is clearly worthwhile and collaboration might be possible.

9.0 Cost Estimates

9.1 Manpower

The table below indicate estimated manpower requirements for the project, including the

Task	Effort (dsy)
Work to CDR	0.3
GUI	0.4
Information Handler	0.2
Program/Plan Handler	0.2
Scheduler Engine (including investiga-tion +interfacing)	0.8
Sanity Checker	0.2
Query Handler	0.2
Queue Handler	0.1
Observation DBMS	0.5
Accounting	0.2
Chooser Modifications	0.1
Integration and Testing	0.2
Commissioning	0.1
TOTAL	3.5

Manpower requirements

Observation Database system described in [O-19]. It must be emphasised that these are approximate and key parts are highly dependent on the approach to the scheduling engine, interface and integrate, rewrite or create from scratch, and indeed whether or not collaborations are entered into. Scientific input and project management are absorbed into the costs.

Given the uncertainties in a number of areas a guess might place the size of the error bars at as much as 50%.

The effort has not been profiled across financial years as it is highly dependent on the number of staff available to work. If the ATC alone allotted 1.5 staff then the project would take 2.3 years. In the scenario of the JAC providing 0.5 dsy over this period then the duration would be closer to 1.8 years.

9.2 Requisitions

The following are identified as the likely non-manpower costs of the project - note JAC travel costs are not included. Note that the costs here are split between FY 1999/2000 and

Item	FY 1999/2000 (UKP)	FY2000/2001 (UKP)
Team Meetings travel	2000	8000
Collaboration/ investigation travel	1000	2000
Purchase of tools.	1000	0
DBMS	0	20000
Other	500	500
TOTAL	4500	30500

Project Requisitions

FY2000/2001. This is partly to show the effect of purchasing a database management system if required.

The cost of the DBMS is very much a ballpark figure, and could be much smaller, or even zero. It is not likely to be much larger.

10.0 Project Plan

A formal project plan at this stage is not simple to produce as there is too much uncertainty regarding the detailed plans and the staffing levels. However, some milestones can be chosen. Here I work on the assumption of 1.5 staff years being available from the ATC and 0.5 staff years from the JAC to produce a very approximate schedule.

November 1999. Project begins.

February 2000. Project CDR, recommendation on Scheduling engine and database technology.

January 2000. Installation of ODB for beta test.

May 2001. Begin integration and test.

July 2001. Delivery of scheduler tool to UKIRT.

11.0 Descope Options

The Scheduler requirements and preliminary design describe the modules that are required for a full system. Given potentially limited funding and effort it may be necessary to descope this, or to prioritise the items with the less essential aspects provided significantly later.

Key advantages of using software to automate the scheduling are the “sanity checking” and calculating of nightly observing plans based on rules. The latter is probably not useful without some of the former. If other software can be adapted for the latter then they are also perhaps the easiest to provide. There is little in these that can readily be descoped, and there is no point in the “keeping track” functions of the database or long term scheduler without the observing plans. Rather than descope the alternative is probably to go for doing everything manually.

A major cost saving descope would be to simplify greatly the Observation Database, so that its functionality is similar to that of the current simple directory structure used by UKIRT-PREP. This means eliminating all the accounting requirements from the system. The Chooser does not write anything when Observations are completed, the Scheduler has no information available about previous attempts at Observations, and there is no user interface to simplify checking the completion status of programmes. The implication of this is that all “keeping track” has to be done by UKIRT staff by hand. For example the observing log could be checked each day, completed observations removed from the database and the Scheduler run to create the next nights plan. Meanwhile a record of whose observations had been done would also need to be kept by hand, so that completion of programmes can be monitored. This would be quite a bit of “detailed attention paying”, especially if almost all nights were flexibly scheduled, but in theory it could be done.

Another area of descope is to drop any consideration of medium and long term scheduling tools for the Scheduler. The only functionality would be to calculate a Plan for a single night. This means that the allocation of time would have to be done using “rules of thumb” -e.g only ~2 nights worth of high priority good condition programmes per month and a large number of low priority backup programmes. Some of the latter are needed anyway, but there would be less chance to fine tune allocations. The other implication is that there would be no ability to “predict” schedules for a few days ahead to get an overview of what the likely scheduling of various programmes was.