Hartebeesthoek 26m

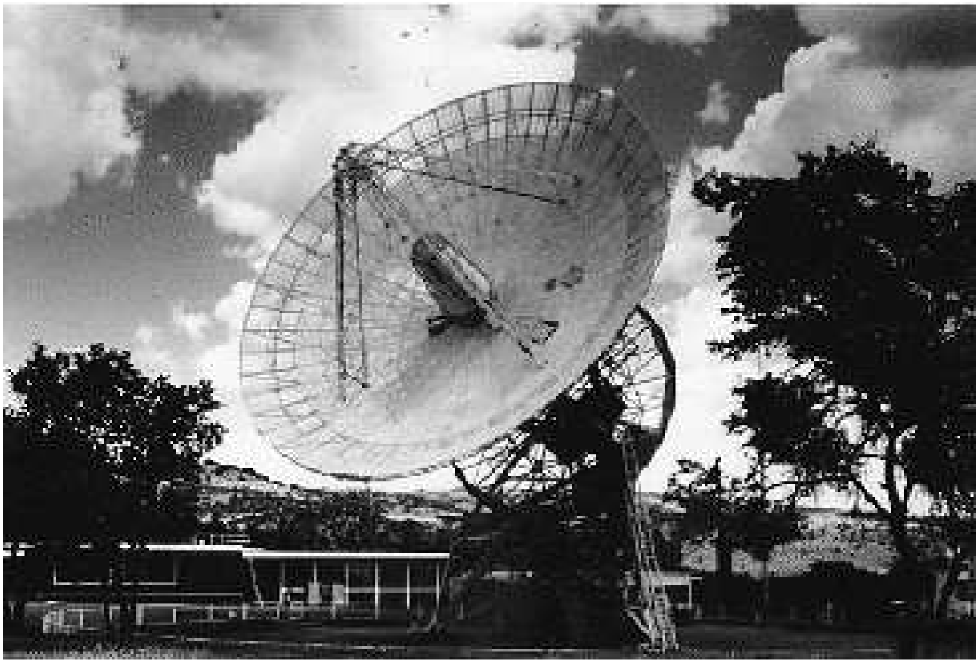
New Computer Control System (NCCS)

REQUIREMENTS & SPECIFICATIONS

release 1.1

Andy Götz,
Mike Gaylard,
Jonathan Quick

November 28, 1995



REQUIREMENTS & SPECIFICATIONS
REQUIREMENTS & SPECIFICATIONS
REQUIREMENTS & SPECIFICATIONS
REQUIREMENTS & SPECIFICATIONS
REQUIREMENTS & SPECIFICATIONS
REQUIREMENTS & SPECIFICATIONS
REQUIREMENTS & SPECIFICATIONS

Contents

1 Introduction

The Hartebeesthoek 26m telescope is at present controlled by a suite of computer programs running on an HP 1000 E series computer. This computer was bought more than 13 years ago and is now out of date. The HP 1000s are not manufactured anymore and are becoming increasingly difficult and expensive to maintain. Proof that the HP 1000 has served a good purpose is that it is still running and controlling the telescope today. However it is limited in CPU power, memory, disk space and networking capabilities. It also does not support modern software and hardware standards. New technology is available on the market today which offers much more disk space, cpu and memory and a modern working environment at affordable prices. The new technology is mainly in the form of low-cost PC's. A decision has therefore been taken to replace the HP 1000's with PC's and at the same time rewrite the telescope control system.

This document presents the requirements of the new computer control system (NCCS). It will serve as the basic requirements list for the followup document - the Design Document. Where a solution and/or standard has already been adopted it will be clearly stated so. Consequently this document is more than simply a list of requirements. It also includes a list of specifications and has therefore been called the - *Requirements and Specifications Document.*

The information presented in this document has been obtained by the authors after discussions with other HartRAO staff and users and browsing through the existing documents. In drawing up the document the authors have tried to be as precise as possible.[1]

The authors welcome and need all constructive feedback which results from reading this document. Readers who find inaccuracies or features which are missing and would like to see them incorporated should inform the authors. Not doing so could or will result in features NOT being implemented in the first version and maybe NEVER being implemented because the design of the system excludes them. This is therefore a desparate plea to all readers and future users of the NCCS to provide the authors with feedback.

In writing this document a distinction has been made between requirements and specifications which are mandatory (marked with a $\surd$) and those which are desirable (marked with a $\heartsuit$). The number of $\heartsuit$s indicates the level of desiredness (more $\heartsuit$s means more desirable). In the text the same distinction is made between these two classes of requirements and specifications by employing the words *must* and *should* respectively. Deadlines are marked with a $\clubsuit$ and are to be taken very seriously.

2 Main Requirements

The main requirements and goals of the NCCS is :

$\surd$ **DEVELOP A NEW CONTROL SYSTEM FOR THE 26M HARTEBEESTHOEK TELESCOPE TO COMPLETELY REPLACE THE EXISTING SYSTEM (HARDWARE AND SOFTWARE) RUNNING ON THE HP 1000 COMPUTERS,**

$\surd$ **THE NEW SYSTEM MUST OPERATE THE TELESCOPE AND RELATED HARDWARE IN A STABLE AND RELIABLE MANNER.**

[1] the authors apologise for the software bias of the present document but that is due to the fact that this document deals mainly with the software of NCCS

The aim of this is to be able to switch off the HP 1000s and forget about them. These goals must be achieved in the first version of the NCCS. Any other requirements such as extending remote control to more hardware or improving the control system are secondary to these requirements.

3 Hardware

The hardware refers to the devices with which the control system has to interact in order to move the telescope, setup a particular observation, acquire data, perform some degree of realtime analysis and detect faults. These devices are the sensors and actuators of the control system. It is important to identify all devices which need to be interfaced by the control system. The following list enumerates all known devices which have to be controlled by the NCCS :

3.1 PC-STEER

Refers to the computer which actually drives the telescope. It presently consists of a PC running a program written in Turbo Pascal under DOS. It runs in slave mode and interfaces to the outside world via an HPIB interface responding to commands sent via the HPIB bus.

PC-STEER understands 22 ASCII type commands (cf. *HartRAO Antenna Control System*, page 20) and returns a status string everytime it is addressed in listen mode (idem. page 23).

PC-STEER also has a graphical interface with basic user interaction via the keyboard.

This system is working today and is used during VLBI observations and the Rhodes mapping program. However in order to replace HP 1000s a number of mandatory improvements must be made :

- √ radiometer digital voltmeter (DVM) readings be acquired synchronously with telescope coordinates,
- √ the hardware interface's known design deficiencies (power supply is instable, A/D conversion is done a long distance from the power supply etc.) have to be solved,
- √ UTC time as generated by the station clock must be read by PC-STEER and used to synchronize and set the computer time,
- √ replace the HPIB interface with Ethernet,
- √ apply time-of-day dependent hour angle (HA) and declination (DEC) error correction as on HP 1000, this is implemented as a table of offsets stored as a 2D function of UT and day of year (eventually to be replaced by a model),
- ♡♡♡ convert PC-STEER from Turbo Pascal running under DOS to C running under Linux.

Q: A FUNDAMENTAL QUESTION TO BE ASKED IS DOES PC-STEER HAVE TO BE CONVERTED TO LINUX AND C FOR THE FIRST PHASE OF THE NCCS OR WOULD IT BE POSSIBLE TO MAKE ONLY MINOR MODIFICATIONS TO PC-STEER AND CONCENTRATE ON THOSE PARTS OF THE SYSTEM WHICH DO NOT EXIST IN THE BEGINNING ?

A: THE ANSWER TO THIS WILL DEPEND LARGELY ON THE TECHNICAL FEASABILITY OF RUNNING PC-STEER UNDER LINUX AND THE HUMAN WORKLOAD INVOLVED IN DOING SO, A COMPREHENSIVE SET OF BENCHMARKS AND A DETAILED ANALYSIS ARE REQUIRED TO DETERMINE THIS.

In addition to the necessary changes the following changes are desirable but not mandatory (cf.*A Proposed Configuration for the New Control Computer System (NCSS)*) :

- ♡ control tilt and focus of subreflector (currently analog input and outputs),
- ♡ read angles from a second DEC encoder (using multiplexed encoder inputs) to improve reliability of DEC encoder readings,
- ♡ apply independent error maps to the two DEC encoders (note the error maps do not exist yet),
- ♡ check quality of data from encoders in low and high speed,
- ♡ apply refraction correction separately from the structural error map (formula available from VLBI field system),

3.2 Instrumentation Control

The NCCS must control the following hardware :

- √ select calibration noise diode (8 bit expandable to 12 non-multiplexed),
- √ master computer fail alarm reset (1 bit),
- √ select input signals to DVM #1 (4:16 multiplexed),
- √ select input signals to DVM #2 (4:16 multiplexed),
- √ select input signals to Pulsar Timer (3:8 multiplexed),
- √ turn 18cm and 13cm noise adding diode off/on, to be generalised to include any noise adding diodes (see below),
- √ select input signal to General Purpose Radiometer (GPR) (4:16 multiplexed),
- √ select GPR IF filter bandwidth (3:8 multiplexed),

At presently Serial Digital Input Output (SDIO) is used by the VLBI field system as an instrumentation bus to control some of the above hardware. SDIO is a general purpose interface for doing input and output which can be daisy-chained to make up a bus. It communicates with the main computer via an RS-232 serial link. SDIO cards have been developed for doing digital input/output and for doing analog input. This system has been developed by Justin Jonas of Rhodes University and has been maintained and adapted by Jonathan Quick at HartRAO. It suffers from the limitation that if one of the interface boards goes down then the whole bus goes down.

The NCCS must continue to use the SDIO. The following improvements have to be made however :

- √ select input signal to general purpose mixers,

- √ convert the RS-232 link to RS-422 so as to be able to cover long distances and be immune to noise,
- √ group devices together logically on independant SDIO buses to prevent faults on a single interface from affecting all devices connected to SDIO,

No major additional development should be put into SDIO. Where SDIO is is found to be lacking in functionality Ethernet, HPIB-488, RS-422 or an industrial solution should be used.

In addition to the above mandatory hardware to control it is desirable that the NCCS controls the following hardware (probably via SDIO) :

- ♡ configure all receivers i.e.
  - ♡♡ select noise-adding noise diode and modulation on/off,
  - ♡ select dual/single beam,
  - ♡ select Dicke switch on/off,
  - ♡ select polarisation mode.
- ♡ configure 3.5cm, 6cm, 13cm, 18cm radiometers (e.g. mode is noise adding/Dicke/total power).
- ♡♡ select input signals to present chart recorders (4 x 3:8 multiplexed)

Note the old radiometers have not been designed for computer control and will need to be modified in order to accept computer control.

3.3 Data Acquisition

Data acquisition refers to all data which has to be acquired by the NCCS. This can be divided into fast data acquisition, slow data acquisition and specialised data acquisition. Specialised data acquisition refers to those subsystems which generate their own data and which have to be read and stored by the NCCS e.g. correlator and pulsar timer but not the VLBI terminal.

3.3.1 Fast Data Acquisition

Fast data acquisition refers mainly to the radiometers and the correlator square-law detector. The radiometers are interfaced via two digital voltmeters (DVMs) which can be switched to take any of the radiometer outputs or various other sources as inputs. The DVMs are currently Fluke 8840A's which are interfaced via HPIB. The most stringent requirement is that they have to be read synchronously with the telescope positions and correctly time-stamped. Synchronization has to be to within a few milliseconds. The sampling frequency on the old system was a maximum of 50 Hz. The NCCS must be capable of guaranteeing this as well.

The most logical place to read the DVM's is to do so in PC-STEER. This is because PC-STEER has the telescope coordinates and time locally available.

The acquisition system must offer standard data treatment options for the radiometer readings e.g. averaging, median filtering.

The HPIB-based DVMs are 9 years old and showing signs of aging - they should be replaced as standard continuum data acquisition DVMs. The DVMs could be replaced either with new intelligent DVMs *á la* Fluke's or with A/D interfaced directly to a PC.

3.3.2 Slow Data Acquisition

Additional slow values which have to be read e.g. humidity, temperature, wind speed, will be read by Serial DVMs (SDVMs).

3.4 Correlation Spectrometer

The existing correlator is largely under manual control with a few limited functions under computer control via the DIO on the HP 1000.
The functions under computer control which must be reproduced in the NCCS are :

- √ start correlator (output)
- √ read data ready signal (input)
- √ read 16-bit parallel data from the correlator (input),
- √ "fix" data as is done by ACFIN program on HP 1000 (to produce an auto- or cross-correlation function)
- √ transmit correlator data (raw and/or fixed) to master computer and convert correlation function to spectrum.

These functions are presently implemented via a dedicated interface consisting of 2 control lines and 16 data lines.
The new correlator interface will be based on a PC. It will look like a black box to the master computer and will be interfaced via Ethernet or RS-422 or HPIB.
In addition to the above mandatory commands the new system should accept the following commands (cf. *A Proposed Configuration for the New Control Computer System (NCSS)*) :

- ♡ select video signal bandwidth,
- ♡ select correlator bandwidth,
- ♡ set video signal level via computer-controlled attenuators,
- ♡ reset correlator,
- ♡ set most significant bits of the correlator count (i.e. replace thumbwheel switches),
- ♡ setup correlator quadrant configuration,

3.5 Pulsar Timer

Presently running the Mark I pulsar timer but to be replaced in the future by the Mark II pulsar timer. The Mark I is interfaced via HPIB to the main computer. The NCCS must be able to control the Mark I Pulsar Timer i.e. it must be able to :

- √ SAMPLE - select number of samples and period,
- √ SETUP TIME - set the pulsar timer clock,
- √ PRINT TIME - output the time on every second,
- √ OUTPUT DATA - output the accumulated data,

- √ SETUP τ - select the post-detection time constant,
- √ GRAPH DATA - output accumulated data sample to oscilloscope,
- √ LED TEST - turn on the front-panel LEDs for three seconds,
- √ RAM TEST - test memory chips used for data storage,
- √ D/A TEST - output a ramp signal to both analogue outputs,
- √ INPUT VIEW - sample input and write to analogue meter,
- √ BUS TEST - output A...Z on the HPIB.

The Mark II pulsar timer is presently under design and it has been decided to interface the pulsar timer to a PC running Linux. The PC will communicate with the NCCS via the network using TCP/IP. The exact communication protocol still has to be decided upon.

- ♡ interfacing the Mark II pulsar timer is considered desirable but not mandatory for the first version of the NCCS.

3.6 Frequency Synthesiser (HP8662)

Used for frequency control of the local oscillator chain. Interface is via HPIB.

The following functions must be computer controlled :

- √ select frequency,
- √ set output power.

3.7 Frequency Synthesiser (HP3336)

Used for pulsar timing sampling frequency and for frequency control of the local oscillator chain. Interface is via HPIB.

The following functions must be computer controlled :

- √ select frequency.
- √ set output power.

3.8 Mark IV VLBI Terminal

Consists of a PC running the Field System and controlling the Mark III VLBI Data Acquisition Terminal (DAT). Presently running version 8 of the field system under Venix but will soon be replaced by version 9 which will run under Linux.

During VLBI observing the field system is master of the telescope and bypasses the HP 1000. It sends commands to PC-STEER and has control of all necessary equipment via SDIO (cf. Instrumentation Control section above).

Because the NCCS is supposed to provide an integrated control system with common log files, user interface, file formats etc. VLBI should be better integrated into the control system so that it runs as one of the observing programs in the same way as spectral lines, continuum etc.

The VLBI field system should coexist with the NCCS but be in control during VLBI. Other programs (e.g. Vela monitoring) must not be able to run during VLBI unless called by SNAP commands.

4 Time

Timing is vital to controlling a telescope for doing astronomical observations. It is important that the NCCS implements timing correctly and does not compromise on any of the hard or soft timing constraints. The following discussion on timing will be separated into hardware and software timing.

4.1 Hardware

Refers to the station clock, maser, 1PPS chain. The chain consists of MASER-CLOCK-TRANSMITTER-RECEIVER-COMPUTER. It is assumed that everything is implemented entirely in hardware and the signals are distributed directly via hardware to wherever they are needed.

4.2 Software

Software timing refers to the timing as seen and used by the control computer(s). This has to be synchronised with the station time via a mixture of hardware and software. Computers are bad at keeping time. This has however been solved for the present system (HP 1000 and PC-STEER) and the NCCS should learn and/or copy from these solutions.

The HP 1000's internal clock is synchronised to the maser frequency standard via a 1 MHz signal which feeds directly into the HP 1000. Days, hours, minutes and seconds are setup manually and then synchronised to the 1PPS via the pulsar timer.

PC-STEER resynchronises the internal software clock once a second to be synchronous with the 1PPS signal. Days, hours, minutes and seconds are setup manually and then synchronised to the 1PPS.

Time must be kept to sufficient precision to ensure that position calculation is accurate to at least 1/10 of a beamwidth at the highest observed frequency (22 GHz) i.e. 12 arcseconds. PC-STEER presently does all its calculations to 1 arcsecond accuracy and this should be maintained. Entry of leap seconds must also be taken account of. For pointing the telescope time must be kept accurate to within 1/10 of a STEER cycle (10ms) for the hardware and 1/2 STEER cycle (50ms) for software.

Ideally all computers on the network should have the same time. This helps for sharing of files via NFS and correlating events recorded by different computers. A freely available network protocol exists for Unix called Network Time Protocol (ntp) which is capable of synchronising computers over Ethernet to within millisecond precision. All computers should be time synchronised via ntp.

5 Technical Monitoring System (TMS)

This system (proposed by Mike Ainley) will monitor station engineering parameters e.g. power supply voltages, cryogenic temperatures, vacuum levels, etc. and log them. It will display the status of each sub-system visually and be able to inform the control computer (on request) on the status of each sub-system.

At present a basic proposal exists using commercially available hardware and software modules. The TMS must coexist with the NCCS. The NCCS will control all parameters which are essential for observing i.e. setting up the radiometers, frequency etc.

This system should be the basis of an intelligent fault detection and diagnosis system.

A minimum configuration based on Turbolink and SCADA has been proposed by Mike Ainley (cf. *DATA ACQUISITION AT HartRAO*) and should form the basis of the initial installation. Exactly which modules are bought will depend on how much money is made available.

6 Observing

HartRAO supports a variety of observing programs much the same as in other observatories. HartRAO is unique however in the fact that a number of different programs can be scheduled simultaneously. Conflicts are avoided by the scheduler and by a kind of gentle(wo)man's agreement. This means that long term monitoring programs can be scheduled simultaneously with short term programs . It also means that telescope time is used efficiently. This unique "multi-tasking" feature of the HartRAO observing scheduler has to be supported by the NCCS. Another unique feature of the HartRAO system is the large amount of unattended observing which is done. The NCCS has to support this feature as well.

This section will explain the main observing modes used at HartRAO and try to list the main features which are necessary and have to be supported by the NCCS.

They must be written in a pseudo control language which resembles a mixture of C and English. The algorithms must be broken down into short atomic actions. Control flow must be indicated by means of the following control statements :

```
if ( condition ) then { statement(s) } else { statement(s) }

while ( condition ) do { statement(s) }

repeat { statement(s) } until ( condition )

for ( i = 0 ; i < N ; i++ ) do { statement(s) }

switch ( condition ) { case (1) : { statement(s) ; break }
                                  .
                                  .
                                  .
                       case (n) : { statement(s) ; break }
                       default  : { statement(s) }
                      }
```

Comments can be specified in the standard C or C++ style i.e.

```
/* this is a short comment */

// the whole line is a comment
```

Here is an example of using this language to specify an algorithm one of the authors uses for waking up in the morning :

```
repeat
{
   sleep
   alarm rings
   switch off alarm
   if ( first time )
```

```
   then
   {
      sleep
   }
   else
   {
      wakeup
   }
}
until ( awake)
wash face
brush teeth
while ( not really awake )
{
   make coffee
   drink coffee
   meditate on life
}
if ( dirty )
then
{
   take a shower
}
get dressed
go to work
```

The NCCS must support the observing algorithms below either as they are described here or in an equivalent manner.

♣ **ALL ALGORITHMS MUST BE SUBMITTED BY 1/12/95**.

6.1 Continuum

Today a variety of observing programs exist for doing point source observations. These should be treated in a unified manner. The option of using stepping, drift and scanning should be incorporated into the flux measuring program.

Observing programs for continuum can be divided up into a number of different programs :

6.1.1 Stepping

Flux measurements of point sources. Makes use of STEER to drive the telescope and radiometers to acquire data.

JQ/GDN to provide the algorithm for observing strategies.

6.1.2 Drift Scans

GDN/MJG to provide algorithm for observing strategy.

6.1.3 Scanning

Measure flux or T_B distribution via orthogonal scans.

MJG/GDN to provide algorithm for observing strategy.

6.1.4 Mapping

Mapping program to make continuum maps.
JJ to provide algorithm for observing strategy.

6.1.5 Sun/Moon

Makes maps of the sun and moon.
JQ to provide algorithm for observing strategy.

6.2 Spectroscopy

Observing program to take spectra of astronomical sources. Uses STEER to drive the telescope, correlator to take the spectra and radiometer to check the system temperature.
MJG to provide algorithm for observing strategy.

6.3 Pulsar Timing

Pulsar timing program to measure the time structure of pulsars. Makes use of STEER to drive the telescope, pulsar timer to measure the timing profile and radiometers to check the system temperature.
CF to provide algorithm for observing strategy.

6.4 VLBI

VLBI observations are run using the field system. Each observation has a SNAP schedule which sets up the Mark IV terminal and send commands to PC-STEER to move the telescope.
JQ to provide observing algorithm.
Support for Mark II VLBI will be continued via the field system and manual control of the recorder.
The NCCS will not support the old Mark III VLBI running on the HP 1000.

6.5 Test Programs

The following test programs exist for the HP 1000s and must exist in an equivalent form in the NCCS :

- √ ANGLE - displays the angle encoder readouts and tachometer sign and magnitude signals five times per second, as a diagnostic for encoder or servo faults.
- √ CALT - tests the receiver systems and the reading of the radiometer outputs by the Fluke voltmeters.
- √ CORRTEST - test the correlator in a variety of modes.
- √ SRDIO - test the serial digital input and output respectively (to be written).
- √ DODECDN - show current value of dec pointing offset, which changes seasonally.

- √ FDVM - sample HA, DEC, DVM1, DVM2 at specified intervals and write results to a disk file.
- √ HADC - read and capture HA and DEC.
- √ HPIB - test any instrument or device on the HPIB interface.
- √ HYP - reads and displays the hyperbola tilt and focus.
- √ MANDR - provides direct control of the antenna via the computer system.
- √ MONI - this provides information updated once per second on the antenna position commanded by STEER, the actual antenna position, the position error, and the present siderial time and approximate right ascension of the antenna.
- √ RADSP - samples the signal from a radiometer and Fourier transforms it to obtain the power spectrum of signal.
- √ TEMPS - monitors system parameters read thru the DVMs.
- √ THUM - reads the outside air temperature and humidity and displays the results.
- √ TSETP - automatically scheduled when the computer is rebooted, in order to set the computer time accurately. It uses the pulsar timing system which has access to the station 1 PPS signal for this purpose. It can be run at any other time to reset the computer clock.
- √ ZE - park antenna at zenith now.

The following utility programs exist for the HP 1000s and must in an equivalent form in the NCCS :

- √ AZEL (utility, standard fortran) - convert HA,Dec to/from Az,El at HartRAO.
- √ DPRC2 (utility) - precess an input 1950.0 epoch Right Ascension and Declination to the present epoch.
- √ PJ2000 (utility, standard fortran, also on PCs and Sun) - precession to/from J2000 coordinates.
- √ STNOW (utility) - current siderial and universal times are written to the screen once a second by this program, accompanied by a beep. This enables the computer time to be checked easily against other UT or ST clocks.
- √ TSTPLN (utility, author JQ?) - gives the current coordinates of the planets, from MICA.
- √ UPDN (utility, standard fortran) - calculate when an object rises and sets

6.6 Scheduler

The NCCS must provide at least the same scheduling capabilities as the present system.

A description of the present scheduler is as follows :

- The present system makes a distinction between the master observing program (MOP) and routine observing programs (ROP). The philosophy of the present system (as implemented by SHPRM/SCHDL on the HP 1000s) is that the telescope is managed as a global resource. The current program has control of the telescope until such time as it decides to release it. At this time if the scheduler has work to do (this has been calculated previously by looking at the input parameters) it wakes up and schedules the program which is supposed to run. The program which is supposed to run is calculated based on the present sidereal time, the requested source's coordinates and the mode of routine observation requested. The following modes are supported:

  √ calculated hour angle (maximum of 13 observations per day),

  √ at a given hour angle (maximum of 7 observations per day),

  √ immediate scheduling (after 1 minute).

  Every program has to specify the observing frequency, radiometer mode (if continuum) and source name/coordinates. 5 optional parameters can be specified to the scheduler to specify anything else the observing program wants to use the optional parameters for.

  Once a ROP has gained control of the telescope it keeps it until such time as it decides to release the telescope back to the scheduler.

  The list of ROPs to be scheduled and their parameters are kept in a file which is read by the scheduler each time it starts up. This ensures that the list of scheduled programs is maintained over system crashes and that it can be modified during runs and the scheduler will take the modifications into account.

The NCCS must offer all the above options.
Additional options which are desirable are :

♡ The ability for the NCCS scheduler to interrupt an observing program and if necessary suspend it and schedule a new program. The notion of priorities of observing programs will have to be introduced in order to do this. Programs with a higher priority will have precedence over programs with a lower priority. Programs should only be interrupted at the end of an observation. The definition of what is an observation will depend on the kind of observation i.e. continuum, spectral lines, mapping, pulsars. Priorities will be distributed by the operations manager but it should be possible for a program to modify its priority dynamically if necessary.

♡ Support the notion of Background Observing Programs (BOPs). These are low priority programs which sit in the background and are only scheduled when neither a MOP or ROP are running. The BOPs are there to soak up any idle time of the telescope and ensure maximum utilisation of the telescope. Typical example of BOPs are pointing programs.

6.7 Safety

Safety is a very important aspect of any control system. The NCCS will only be involved in equipment safety and not personal safety. It is assumed that all aspects of personal safety (e.g. fire, danger of moving the telescope while somebody is on it) are taken care of by systems and procedures independant of the NCCS.

Therefore the main safety requirement of the NCCS is :

√ **THE TELESCOPE'S SAFETY MUST BE INSURED AT ALL TIMES.**

This means during normal operations and during a crash of the NCCS. The hardware interfaces to the telescope of the NCCS must stop the telescope in a predefined and safe way if the NCCS crashes or starts to malfunction.

6.8 Faults and Diagnostics

An intelligent fault detection and logging system is required. Two sources of fault detection can be identified - (a) faults which can be directly detected by the NCCS based on what the observing thinks should be happening and (b) faults which the TMS detects based on the its knowledge of the engineering parameters.

The two systems of fault detection are complementary and should not duplicate each others knowledge bases and rules.

All faults and diagnostics must be time-stamped and logged to file. A browser must be provided for looking at the contents of this file.

6.9 Alarms

The NCCS must support the detection, triggering, broadcasting and logging of alarms.

The present system supports alarms in the following way :

- √ alarms can be triggered directly by an observing program (by switching a bit),
- √ standard alarms patterns i.e. slow (STEER alarm), fast (OBS alarm), continuous (CPU alarm),
- √ alarms can be triggered via a watchdog timer (different time constants supported are 5s, 19m, 38m, 1h16m and infinity).

The NCCS must provide the above alarms.

In addition to the above requirements the NCCS must provide :

- √ a log of all alarms which keeps track of the time and origin of when they occurred,
- √ support for the present buzzer alarm,
- ♡ a synthesised voice alarm (in addition to the present buzzer alarm) which is linked up to the public address (PA) system and which also phones the appropriate staff member on duty,

6.10 Logs

On the HP1000 observing logs are kept by the spectral line and by the field system on the VLBI field system. This must be extended to all observing programs. A system wide observing log must be kept (no matter what program is running i.e. this includes VLBI). The observing log should be in an easily readable format. A log browser must be provided for browsing, printing and commenting the log files. It must also be able to generate overall statistics of observations, implying that predefined formats are needed for each type of report.

The observing log must store at least the following information (all entries must be time stamped) :

1. beginning of each observation :

- √ project name,
- √ proposal number,
- √ principal investigator,
- √ observation number
- √ type of observation (e.g. spectral line, VLBI, pulsar etc.),
- √ class of observation (e.g. ROP, thesis, student, etc.),
- √ start and end time of observation,

2. during observations :

- √ any changes to system configuration,
- √ the system state e.g. Tsys, receiver stability, weather
- √ source name and coordinates,
- √ any faults detected,

6.11 Configuration

Receiver and system setups must be defined and stored in configuration files. The present format of these files on the HP 1000 (which contain feed offsets, noise diode cal values etc.) is not adequate and needs to be refined to make it easily extendable. The NCCS must support the notion of configuration files. Standard and non-standard setups must be read by the NCCS from these files. It must be possible for observers to modify existing configurations and define their own configurations using a simple ASCII-based editor. Configuration files must be kept under revision control so that it is possible to go back to old configurations.

6.12 Remote Monitoring

The present telescope control system allows some limited remote observing in that it is possible to login to one of the Sun file servers (either via modem or Internet) and then connect to the HP 1000 via kermit to look at what is running by running the `whzat` program or viewing the observing log file.

The NCCS should provide support for remote monitoring via eavesdropping. Eavesdropping means that an astronomer can be in a remote place and can find out what is presently being observed and look at the status of the observations (see *Remote Observing and Experience at ESO* for a definition of remote observing terms). Eavesdropping does NOT mean moving the telescope from a remote place. Eavesdropping must be possible via modem and Internet. It must be possible to find out the status of the present observing program, follow in quasi-realtime the present observation, take a look at the data and query the observing and system logfiles.

7 User Interface

It is important that the NCCS be designed with a homogeneous and easy-to-use user interface. Guidelines must be given to program developers on how to develop such a system with a class library available which allows users to develop interfaces which have the same look and feel. Prototype interfaces must be made available which observers can modify to create their own interfaces.

The user interface can be broken down according to the type of user - observer, developer, test, analyser and non-expert. The user interface must cater for all these groups.

The user's interface can be split up into two parts - an ascii based interface and a graphical users interface. Experience shows that a graphical interface is not sufficient on its own and a high level extendable language is very useful especially for doing rapid prototyping of a new observing program.

7.1 Ascii interface

Observers must have a high level interpreted language for interacting with the system. This language must allow the user to do the following :

- √ setup an observing schedule in a simple manner,
- √ support most flow-control constructs (e.g. `if then else`, `while do`, `repeat until`, `for do`)
- √ be easily extensible,
- √ support functions/subroutines,
- √ support the definition of global and local variables,
- √ the language must be ascii based.

Examples of such a language are TK/TCL, OBS, SPEC, IDL and interpreted C to mention a few.

The observing language has to interact with the scheduler in the same manner as all other observing programs. It must be possible to have more than one interactive session running simultaneously without any conflicts.

7.2 Graphical interface

In addition to an ascii based interface the NCCS must also provide a graphical user interface. This interface will be used for doing standard procedures such as setting up the telescope, displaying the state of the control system, "manually" driving the telescope etc..

8 Development Environment

In order to be able to maintain existing observing programs and develop new observing programs the NCCS has to provide support for experts to extend the NCCS at system level, modify existing hardware and add new hardware. In order to avoid duplication of code there should be only one way of doing the same thing e.g. setting frequency, acquiring data and communicating with hardware. The development environment should provide a set of system routines stored in dynamically loaded shared libraries for doing system related things. All developers must use these routines. It should be possible to extend the set of system routines.

8.1 Graphics

Graphics must be able to output to X11 and Tektronix screens. A hardcopy device must be supported for doing occasional screen dumps. The main output devices will be screen and disk and not to printers however A plotting package such as PGPLOT must be supported.

8.2 Analysis

A very small set only of analysis programs for continuum, spectral lines, pulsars and mapping will be supported. These programs will be used mainly to take a "quick look" at the data and provide a figure of merit for quality assurance. The majority of analysis will be done offline on PCs and workstations.

Q: what analysis programs exist and which ones should be supported in the NCCS ?

8.3 Data formats

A common data format must be defined and all observed and analysed data stored in this format.

Q: what format should be used - a locally developed one or an international standard ? Why not simply use FITS (compressed if necessary) ?

9 Computer System

This section will describe the system computer requirements of the NCCS. The main requirement is that the system should be modern, low-cost, easy to maintain and reliable. Care should be taken to ensure that the system is not out-of-date before it is finished. This should be done by making sure that the solution adopted is in line with market trends.

9.1 Hardware

The hardware platform should be PC compatible. Where possible high-end PC's i.e. 486/Pentium should be used, and only in those cases where it is not technically and/or financially possible would one consider using lower end PC's e.g. 286.

9.2 Software

Software should follow modern standards and trends. Where the requirements point to an obvious choice for any of the system aspects this will be mentioned and should be considered as part of the requirements.

1. **Operating System**

   The operating system must have the following characteristics :

   - √ it must be multi-tasking,
   - √ it must support the IEEE POSIX[2] standards for operating system calls,
   - √ it must provide a symbolic compiler and debugger,

[2] POSIX is an internationally accepted standard for operating systems

- √ it must provide support for networking,
- √ it must be readily available and low-cost,

Based on the above list of requirements `Linux` (a free version of Unix for Intel platforms) is an obvious choice as operating system and has been specified by the observatory. Where low-end systems are installed which cannot support Linux e.g. 286's, DOS should be the second alternative not because it fulfills all the above requirements but because it is well known, low cost and has an installed base in house already.

2. **Programming Language**

   For efficiency reasons a single language should be adopted as the main programming language to be used for coding. The language chosen must have the following characteristics :

   - √ be an ANSI standard language,
   - √ be well supported by freeware and industry,
   - √ have a good compiler which produces efficient code,
   - √ have a good debugger,
   - √ be portable,
   - √ POSIX compliant,

Based on the above requirements the obvious choice is `ANSI C`. This is the *lingua franca* of Unix and therefore Linux as well. The best compiler available on the market for C today is the GNU C compiler (`gcc`) which comes with Linux.

`C++` should be used for programming projects which are suited to object oriented programming techniques. `C++` has the advantage that it has a high degree of compatibility with C code.

Very limited support must be provided for `Fortran 77`.

It will only be supported for backwards compatibility where debugged mathematical routines exist. All observing programs written in `Fortran 77` must be rewritten in a combination of C and the high-level observing language.

Programming in `assembler` should be strictly forbidden unless absolutely necessary. As far as possible drivers and system software must be written in ANSI C.

Other programming languages should not be supported and if a piece of software has to be included which has been written in another language it should be treated as a black box.

9.3 Code Development

Code development must be done using modern tools :

- √ use `make` to manage compilation and dependancies,
- √ use the Revision Control System (RCS) to provide version control of files,
- √ use the GNU C compiler and graphics debugger to compile and debug.

9.4 Software Technology

Maximum use must be made of modern software technologies typically used for control systems. This means that the NCCS should be a **distributed control system** based on the **client/server** model. The advantages of object-oriented technology i.e. data hiding, encapsulation and inheritance, are well-known today (cf. *Object-orientation: A software technology base for the 1990s*) and where possible **object oriented (OO)** design and programming techniques should be used. The NCCS must be message based. All commands between control components of the NCCS must be passed as messages. Data structures must be kept private to modules.

9.5 Network

As the NCCS will be a distributed control system it will have to rely enormously on the network[3].

The network must have the following characteristics :

- √ be an industry standard well supported by commercial vendors and software,
- √ have sufficient bandwidth to support quasi-realtime control i.e. it must support a sustained traffic between two nodes of a minimum of 10 Hz for small messages with 50 Hz being desirable for data capture from radiometers,
- √ transmission speeds must be predictable,
- √ it must be secure from the outside world,
- √ it must support the Internet protocol standard TCP/IP,
- ♡ because networks are evolving all the time the network chosen should be easily upgradeable to accommodate new standards.

Based on the above requirements the obvious choice for the network is *Ethernet* using TCP/IP as protocol. In order to make Ethernet predictable and secure the Ethernet control segment should be a dedicated segment separate from the rest of the observatory Ethernet segment(s). Sustained traffic load should not exceed a maximum of 30% to ensure predictability. Access to the control segment should only be allowed to the computers dedicated to control. One of the control computers should have two Ethernet cards so that it can act as (IP) router between the control network and the rest of the world. Ethernet can be upgraded to 100 MHz when necessary.

9.6 Disk Space

Data for offline analysis will not be stored on the NCCS and must be transferred to other machines.

9.7 Performance

The NCCS should be at least as performant as the existing HP 1000 control system. No official figures exist for the HP 1000 but from discussions with staff this implies guaranteeing a minimum of a 10 Hz control cycle with 50 Hz being desirable. The

[3] as a well known company once said - "*the network is the computer*"

new system must also be capable of generating logfiles and network access at that speed as well.

Number crunching requirements will be determined by calculations required for the different observing programs e.g. doppler (VLSR), pulsar ephemerides, planet positions.

9.8 Backups and Archiving

The NCCS must offer a possibility of doing regular backups of the disk and be able to archive data offline. A database of archived data should be kept and it should be easy to retrieve old data. The medium chosen for backups and archiving should be an industry standard and well supported, it should also be low cost. An export facility should also be supported so that visiting astronomers can export their data to their home institute. The export medium should be widely supported by the majority of collaborating institutes.

The NCCS must support the following media for internal use :

- √ Digital Audio Tape (DAT),
- √ network backup,
- √ 1.44 MB 3.5" stiffy.

The NCCS must support the following media for exporting data :

- √ Digital Audio Tape (DAT),
- √ Internet,
- √ 1.44 MB 3.5" stiffy disks.

9.9 Redundancy

The system must be redundant. Minor failures should not last longer than 1 hour. It must be possible to recover from major failures in less than 24 hours with a minimum of intervention. Due to the geographical locality of the institute and the lack of funds for an expensive maintenance contract the backup solution must be maintained in house and always be available.

- √ **A SECOND CONTROL COMPUTER (REFERRED TO AS THE "BACKUP" CONTROL COMPUTER) WHICH WILL BE DEDICATED TO SERVE AS A BACKUP IN THE CASE OF FAILURE OF THE MAIN CONTROL COMPUTER AND WHICH CAN BE USED FOR TEST PURPOSES MUST BE BOUGHT**
- √ **SPARES MUST BE BOUGHT AND KEPT AVAILABLE FOR ALL CRITICAL SUBSYSTEMS OF THE NCCS (e.g. PC-STEER)**

The backup computer must be have an identical motherboard and CPU as the main computer and must have identical versions of the main interface cards (to be defined). It must be possible to shutdown the main computer and boot up the backup computer running identical versions of the software. The backup computer must be reserved for this purpose. During tests of new versions of the software (operating system, field system etc.) it must be possible to use the backup computer for testing. The same is true for all backed up critical sub-systems.

9.10 Starting, Stopping and Checking

The NCCS must be easy to startup and shutdown and it must be easy to check to see whether it is running. It must be possible to do so from a single login point. Startup must be completed in a short time (maximum 5 minute with 1 minute being desirable) A log must be kept of all startups, shutdowns, crashes. It must be posible for an authorised non-expert to startup, shutdown and check that the system is running.

9.11 Security

The NCCS must be security conscious. All control protocols should be locally defined. It should have at least level C Unix security. A system log must be kept of all logins, bad logins and shutdowns. Passwords must be regularly changed.

10 Budget

A breakdown of the budget requirements of the NCCS must be made. The proposed expenditures must be listed in order of priority and then submitted to the Director. Money will be allocated as a function of available funds and priorities.

11 Documentation

Documentation must follow modern software documentation standards.

11.1 Offline

The following "offline" documents must be produced and maintained :

- √ *Requirements and Specifications Document* - will cover all aspects of the NCCS which must be implemented, it will also contain a wish list of desirable but not mandatory requirements. Where certain technological choices are clear they will be specified as part of the requirements,
- √ *Design Document* - will present the software and hardware design of the NCCS,
- √ *Budget* - will break down all proposed expenses for the NCCS ordered according to priority,
- √ *Tasklist* - break down work to be done into a prioritised tasklist,
- √ *Planning Chart* - will break down the tasks which need to be done in order to implement the NCCS and propose a plan for them,
- √ *Users Guide(s)* - will describe how to use the NCCS from a user's point of view,
- √ *System Description Document* - will describe the NCCS from a systems and maintenance point of view.

All documents must be written in English using Latex as text processing system (unless another in house standard is stipulated). A HartRAO specific Latex style must be developed.

11.2 Online

Online help must be provided. The "offline" documents must also be available online and users must be able to access them easily.

All online documents must be written in or converted to HTML (Hyper Text Markup Language) so that they can be viewed using the Mosaic and/or Netscape browsers.

12 Testing 1-2-3

The NCCS must be thoroughly tested before being handed over to users. The tests must be done as much as possible in parallel with the running of the old system. Down time must be kept to a minimum.

Q: what is the maximum overall time allowed for testing the NCCS ?

Q: what is the maximum time allowed for testing the NCCS in any one session ?

13 Deadlines

It is important to have deadlines for the NCCS project so that a check can be made on progress and to ensure that it will be implemented within a finite time. The following deadlines have been initially defined but can and should be revised if necessary :

- ♣ a working version of the Requirements and Specifications Document must be available by **1 November 1995**,
- ♣ the Observing Algorithms should be finalised by **1 December 1995**,
- ♣ the Design Document should be finalised by **16 December 1995**,
- ♣ a first version of the basic system should be ready for tests by **1 March 1996**,
- ♣ an operational version of the first version of the NCCS should be ready by **1 June 1996**.

14 Acknowledgements

Many thanks to the following people who provided valuable information, documents and good ideas - Mike Ainley, Claire Flanagan, Keith Jones, Peter Stocker and especially to George Nicolson and Justin Jonas.

15 References

1. *A Proposed Configuration for the New Control Computer System (NCSS)*, M.J.Gaylard, internal HartRAO document.
2. *Serial Digital Input/Output Interface*, J.Quick, internal HartRAO document.
3. *HartRAO Antenna Control System*, J.Jonas, internal HartRAO document,
4. *Linux Installation and Getting Started*, M.Welsh, Internet, 1995,
5. *The Linux Operating System*, S.H.Bkhari, Computer, August 1995,
6. *Where is Client/Server Software Headed ?*, T.D.Lewis, Computer, April 1995,
7. *Distributed Coordination Models for Client/Server Computing*, R.M.Adler, Computer, April 1995,
8. *Data Acquisition at HartRAO*, M.Ainley, internal HartRAO memo.
9. *Object-orientation: A software technology base for the 1990s* by T.Hoverd, in Software Engineering for Technical Managers edited by R.J.Mitchell, Peter Perigrinus Ltd (1990).
10. *http://fits.nrao.edu/FITS.html*, D.C.Wells, W.C.Greisin, R.H.Harten,
11. *http://stdsbbs.ieee.org/products/catalog/drafts.html* to order a copy of the Posix standard,
12. *Remote Observing and Experience at ESO* by A.A.Zijlstra, J.Rodriguez and A.Wallander, The Messenger, No. 81 September 1995.