National Radio Astronomy Observatory
Green Bank, West Virginia

300-FOOT CONTROL COMPUTER MEMO NO. 10

MEMORANDUM November 26, 1984

To: Addressee

From: R. Fisher

Subj: Summary of Meeting to Discuss Telescope Control Computer Uniformity among NRAO Sites

In attendance: Bignell, Burns, Farris, Fisher, Hvatum, King, Payne, Seielstad, Stobie, Vance

This meeting was organized to discuss how and to what extent NRAO can satisfy the users' request that telescope control at all NRAO sites look as similar as possible to the observer. This request was given considerable emphasis at the November 15 Users Committee meeting.

R. Fisher opened the meeting with a statement of the problem and a number of points for discussion:

The only objective under discussion is that telescope control at all sites look the same to the user as far as possible. Uniformity of both detail and style are important to this objective. The implementation of telescope control at different sites is not what is intended to be uniform, only the appearance of each system to the user. Style was defined by example: the control language and its syntax, editors for assembling an observing program, and whether the control language is used in a batch or interactive mode. Detail might include actual parameter and function names. Portability of computer code used to implement a control system was also said not to be part of the objective under discussion even though it may be useful for other reasons.

A method for establishing control system uniformity between sites was presented. Since Green Bank is in the process of designing a new control system for two telescopes, the suggestion was made that this design be presented to other sites for review and requests for change with the aim of establishing a design whose appearance to the user is acceptable to all sites, and as each site found the effort to change its control system the user interface standards which were derived in this design would be adopted in the new systems. This approach would avoid much of the effort that would be required to design new standards with equal contributions from all sites, and this savings in effort would more than compensate for the fact that the result would be different and possibly further from optimum than an equal input design. The degree to which uniformity between sites can be achieved without affecting efficiency of use of the various telescopes will be determined in the design process. A likely outcome will be a system with very good uniformity in style and a large number of specific functions with each site having extensions which make good sense at only one or two telescopes. Also, after systems which conform to the user interface standard are installed, each will evolve in response to observer requests, and some mechanism would have to be set up for keeping all systems up to date on relevant changes without introducing a lot of coordination overhead or constraining development of new features.

Betty Stobie made the point that her system at Tucson is expected to be changed on a timescale roughly equivalent to the one at Green Bank, and as a result she will have to work quite closely with the Green Bank designers to assure that her commitments are met. She also suggested that it would be very useful for one or more Green Bank designers to visit Tucson so that system differences can be accounted for in the design without a large amount of detailed coordination between Tucson and Green Bank.

Carl Bignell said that he had not realized the extent to which the single dish systems would like to operate in an interactive control mode. The VLA allows little or no observer interaction with the observing schedule once it has been assembled. He thought that it should be possible to accommodate both methods of operation and still make the systems appear the same to the user, and he suggested that Barry Clark's memo of 24 January 1983 be used as a starting point.

Hein Hvatum asked for a clarification of what we meant by a control language, and Rick Fisher offered the definition of any statement which the user sends to the control system to cause actions by the telescope and associated receiver electronics. This is very different from the computer language used to implement the control system. HH, "Do you mean a command language?" RF, "Yes."

Bob Vance expressed concern about the intent to allow the observer to change the computer code. R.F. stated that what was meant by an interactive system is the ability to change the observing program while observing. There is no intention to allow the observer to change the code in which the system is written.

Bob Burns used the analogy of writing a scientific paper to describe the design process. One member of a team writes the paper and the other authors suggest changes and additions. He suggested that the design of the control systems proceed with more nearly equal inputs from the different sites than this analogy imples. R.F. said that he intended an approach which concentrates the first draft responsibilities at one site.

Carl Bignell mentioned that there are several people at the VLA who would be involved in the review of a control program design, and he wanted to be sure that there would be adequate time to consider each submission from the design group.

George Seielstad said that he hopes the coordination does not get stuck in a lot of meetings and debates over design, and he thought that the approach being formulated at this meeting would work.

Harry Payne expressed smypathy for Betty's problem of having to get a new control system on line in a fairly short period of time without much help and hopes that a shared design would be of some help to her.

Hein urged that we formalize the coordination procedures with a memo series preferably on the computer.

Rick said that it was his intention to extend the existing 300-foot control computer memo series to cover the coordination process, and promised to send copies of all old memos to everyone not currently on the distribution list. He also promised to continue as the shepherd of the coordination process unless anyone can suggest a more appropriate organization.

Barry Clark's memo and Carl Bignell's memo on "The New Observe Program" are attached to this summary to make them a part of this series.

JRF/cjd

Enclosures
  Clark and Bignell memos

May 10, 1984

To: Addressee
From: C. Bignell
Subject: The New Observ Program

I have started to put together a proposal for a new OBSERV program. Although it is not a complete description in any way, recent comments suggest that there may not be any strong desire to begin a "different" program.

In order to avoid potentially wasting more time on this subject I would like you to read what I have put together (I have also enclosed Barry's memo at the end) and let us discuss if this or a similar approach is worth additional effort. We could bring it up for discussion at one of the Computer meetings.

A PROPOSAL FOR A NEW VLA OBSERVE PROGRAM

During the past year there have been two different proposals for a new VLA OBSERVE program. The two proposals were put forward for two different reasons: (a) to incorporate screen oriented editing capability and (b) to allow throught macro generation a means of "automatically" generating sequences of observations. The following is a description of a new program which incorporates both the earlier suggestions and is designed to aid in a few other ways.

The user interface outlined below is different from the orginal example and is an attempt to aid in learning the program at the same time providing more power and greater ease of use. An extensive on line HELP capability would also be supported. The specific interface currently suggested is one that has become very popular on sophisticated applications software written for microcomputers. Before proceeding, it should be noted that it will not be possible to satisfy every persons esthetics or desires, however it is hoped that this approach is a reasonable compromise.

A general description of the user interface will be presented, followed by the specific proposal for the OBSERVE program.

GENERAL DESCRIPTION

OF USER INTERFACE

The basic user interface is very similar to many programs used on microcomputers. In this approach, commonly used words are used as commands and commands are structured in a tree type organization. Each command at the top tends to group many "related" functions together while those at the lower levels perform the individual functions. The user is generally either in the command mode which will eventually select a function or interacting with the program itself (such as editing, setting parameters, etc). This scheme is aimed at reducing the amount of information the user has to somewhat arbritrarily memorize.

It will be assumed that any terminal using this program will have (a) the capability of cursor addressing, (b) reverse video or highlighting or both, (c) 24 lines by 80 columns and (d) a control (ctrl) and an Escape key (ESC). It will also be necessary for the user to remember a small number of commands.

These include:

Key	Alternative	Action
ESC	ctrl-[	Enter/exit command mode or cancel action.
Up arrow	ctrl-k	Move cursor up one line.
Down arrow	ctrl-j	Move cursor down one line.
Left arrow	ctrl-h	Move cursor left 1 character.
Right arrow	ctrl-l	Move cursor right 1 character.
INS	ctrl-v	Toggles between insert and overwrite character mode.
DEL	ctrl-d	Delete character under cursor.
Page Up	ctrl-p	Page towards top of text one screen

		full of lines.
Page Down	ctrl-q	Page towards bottom of text one screen full of lines.
TAB	ctrl-i	Tab right over number of spaces or move to the next field on the right.
LEFT TAB	ctrl-o	Tab left over number of spaces or move to next field on the left.

Most keys are for moving the cursor around while the others are designed for limited editing.

Screen Layout

The first three lines are reserved for the command processor.

```
 ________________________________________________________________
/COMMAND line when invoked                                       \ <-Note 1
¦Expanded description of command/Error or general messages       ¦ <-Note 2
¦Boundary line indicates miscellaneous information               ¦ <-Note 3
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦ <-Note 4
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦
¦                                                                ¦
\________________________________________________________________/
```

Note 1

This line will list, up to about a maximum of ten words, the commands available at this level when the command mode has been invoked (through the use of the ESC key). example of such a line might be

EDIT FILE CHECK TERMINAL QUIT.

When in the command mode the particular command being pointed at will be in reverse video. This selected command can be avtivated by pressing the return key. The right and left cursor keys are used to move the command pointer between different commands. Commands may also be selected alternativ by pressing the first character of the command itself, independent of the location of the command pointer. The ESC key is used to get out of the command as well. When not in the command mode this line will be left blank.

Note 2

This line has several funtions. Firstly, when the user is in the command mode (activated by the method given in the above note) this line will contain an expanded descriptio of the command under the command pointer. Secondly, if there are any error messages they will be displayed on this line unless more than one line is needed. Thirdly, if there are any short user inputs required (such as answers to questions, etc) they will be requested on this line.

Note 3

The third line is used to demarcate the top three li from the rest of the screen. It will be entirely in rever video. This line will also contain some specific information. Namely, the left most part of the line will indicate current level and its path (e.g. MAIN/EDIT/FILE). In addition there will be at least two other pieces of information: where the ESC command will take the user and the help commands

Note 4

The area of the screen below the top three lines is available to the user programs. It will also be used for extensive help explanations.

Data Entry

There are times when the user must choose or select options for a passive parameter. If only one parameter is being choosen then line 2 may be used for this purpose. On the other hand if there are many parameters to be set then the screen area below line 3 should be used. In general the manner in which the options will be choosen will be form fill in. For example:

```
 --------------------------------------------------------------
/CMD1 CMD2 CMD3 QUIT                                            \
!Set blat parameters                                            !
!TOPLEV/MIDLEV/CMD1  Press ESC to return to commands. ?=HELP    !
!                                                               !
! Parameter1 (hex,dec,oct,*):  ____                             !
! Parameter2 (0 <= nn <= 99):  __                               !
! Parameter3 (on,off):         __                               !
! etc                                                           !
!                                                               !
!                                                               !
!                                                               !
!                                                               !
!                                                               !
!                                                               !
!                                                               !
!                                                               !
!  [More parameters on next page. Use curor keys to advance     !
!   forwards to next (or backwards to previous page)]           !
\______________________________________________________________/
```

The cursor and edit keys would be used to move around, insert and delete the parameter settings. For instance the return key would be used to tell the program "I have finished typing the option for this parameter. Go away and check the validity of the entry and if valid move on to the next field. If not valid give me an error message and leave me in this feild so I can edit it.". Using the cursor keys to move off this feild onto another would also invoke the same action. When all parameters are set the user can escape back to the command level by pressing the ESC key.

Help

There will be two basic types of help available. The first is the ? key which when invoke either while the command cursor is pointing to a particular command when the program is in the command mode or to the feild of a parameter while trying to set the option of a parameter will present an expanded description of the command or parameter whichever is appropiate. The second method of obtaining help is to type CTRL ? . At the top level of the program a general intorduction to the program would be given. At other levels of the "tree" structure, a description for that level is given. These help description will "grow" into the topic, eventually giving the details. The entire screen will be used when necessary and the HELP discussions will probably be about 1 or 2 pages long for individual commands and parameters and probably many pages for the extensive level helps.

THE OBSERV PROGRAM

The OBSERV program would be invoked in one of two ways:

```
OBSERV
or
OBSERV filename.
```

In the first instance the program would be started with no referenc to a particular observing file. While the second option would bring the user up with the observing file (and all the program defaults) filename.

```
 ----------------------------------------------------------------
/TERMINAL EDIT GLOBAL VERIFY LIST SOURCELIST QUIT                \
|Set type of terminal to be used.                                |
|HELP: Ctrl ? = General assistance, ? for command.               |
|                                                                |
|                                                                |
|                                                                |
|                       O B S E R V                              |
|                           1.0                                  |
|                                                                |
|     NRAO Observing file preparation program.                   |
|                                                                |
|                                                                |
|     IF terminal is not a XXXX then type T to set               |
|     program to proper option.                                  |
|                                                                |
|                                                                |
|                                                                |
\________________________________________________________________/
```

A brief summary of these commands follows.

TERMINAL This command is used to tell the program which type of terminal OBSERV is talking to.

EDIT To edit, create or change a set of observing instructions. Any MACROS will be defined, modified and implemented in this option as well.

GLOBAL Specify which NRAO instrument, observing date, observing file, etc.

VERIFY Check the validity of the observing file.

SOURCELIST Create or modify a list of sources. This will tell OBSERV the positions of the sources and other information unique to the source.

LIST List the observing file or the MACROS file on the CRT or screen in a format specifiable.

QUIT This will exit the OBSERV program. All program options will be stored with the particular file.

TERMINAL

```
 ____________________________________________________________
/SPECIFY OTHER QUIT                                          \
!Indicate a specific terminal type.                          !
!OBSERV/TERMINAL                                     ?=HELP  !

!                                                            !
\____________________________________________________________/
```

SPECIFY A list of terminal types is presented. The user will point to the specific terminal using the cursor keys and select the appropiate option by hitting the space bar. If the appropiate terminal is not listed the OTHER option should be used. To leave this option the ESC key is pressed.

OTHER If the list of terminals available under SPECIFY does not contain the appropiate type, then this menu form fill-in will allow the user to spell out the terminal characteristics of the particular terminal to be used.

QUIT Return to the main menu.

The option which remains in effect is the last one choosen with from the SPECIFY or OTHER option.

QUIT

```
 -----------------------------------------------------------
/                                                           \
:                                                           :
:                                                           :
:                                                           :
:                                                           :
:                                                           :
:                                                           :
\-----------------------------------------------------------/
```

Several options are possible.

(a) If the file has been edited since the last time it was saved, the program will respond:

DO you wish to save this file? (Y/N):

> If the user responds Y, then the file will be saved under its old name with the old version backed up and the program will then gracefully exit to the operating system. If the observer has not specified or set a filename the program will proceed to request one before saving it.
>
> If the user responds N, the program returns to the main menu.

(b) If the file has not been edited since it was last saved the program will gracefully exit.

GLOBAL

The basic parameters for specifying the observing files, type of instrument, etc are changed by the form fill procedure.

```
 ---------------------------------------------------------
/                                                          \
!                                                          !
!OBSERV/GLOBAL   Press ESC to quit                 ?=HELP  !
!                                                          !
!  Instrument                                              !
!   (VLA,VLBA,BOTH,etc..)   .  VLA___                      !
!  OBSERV filename . . . . .  ______________               !
!  User source file name . .  __________________________   !
!  Type of observation                                     !
!   (CONTINUUM, SPECTRAL LINE)CONTINUUM___                 !
!  Programmer number . . . .   _____                       !
!  Observing date  . . . . .   _______                     !
!  Start time (& type) . . .   _________ LST               !
!                                                          !
!                                                          !
!                                                          !
!                                                          !
!                                                          !
!                                                          !
\__________________________________________________________/
```

Whenever possible the specific options allowed in each feild should be spelled out whenever possible.

EDIT

```
 ____________________________________________________________________
/WINDOW MOVE COPY DELETE FILE VERIFY GOMACRO TRANSFORM   **\
|Choose screen layout.                                      |
|OBSERV/EDIT                                     ?=HELP     |
                                                            
|                                                           |
|                                                           |
\___________________________________________________________/
** other commands include: PARAMETERS QUIT
```

WINDOW — This option is used to select what parameters will be used for creating source cards and which parameters will be dispplayed. In addition part of the screen may be set aside to display currently defined MACROS. If this option is not selected then the screen layout is either the program default or the last one used for the specified file.

MOVE — A block or line of text may be moved from one region of the file to another. The cursor is placed at the begining of the block to be moved, command mode is activated and this command is selected. The program responds with "Move cursor to end of text to be moved and press return.". After the cursor is moved and the return key pressed, the program responds with "Move the cursor to position where the text will be moved and press return.". After the user responds to the appropiate instructions the text is moved.

COPY — This command works exactly the same as the MOVE option except that text is copied from one location to another instead of being moved. Another special key (CTRL-R) can be used to help facilitate coping the same text many times. If a block of text that was last copied needs to be copied to another location the cursor is placed at the new location and CTRL-R is pressed.

DELETE — A block or line of text may be deleted with this command. Again the cursor is moved to first part or line, the command is activated, the cursor is moved to the end of the text and the return key is pressed.

FILE — The current file and all of its options can be saved with this command.

VERIFY

The validity of the observing file may be checked now instead of returning to the main menu. The user is allowed to access this program option either from this level or from the main level of the program. The command will be described later.

GOMACRO

Macros may be defined, isted or edited with this option. When activated the screen is cleared if no MACRO screen was selected with the WINDOW commandotherwise the cursor jumps to the MACRO window. The MACROS may be viewed, edited or listed. Return to the Edit window is acheived by activating this command a second time.

QUIT

Return to the main menu.

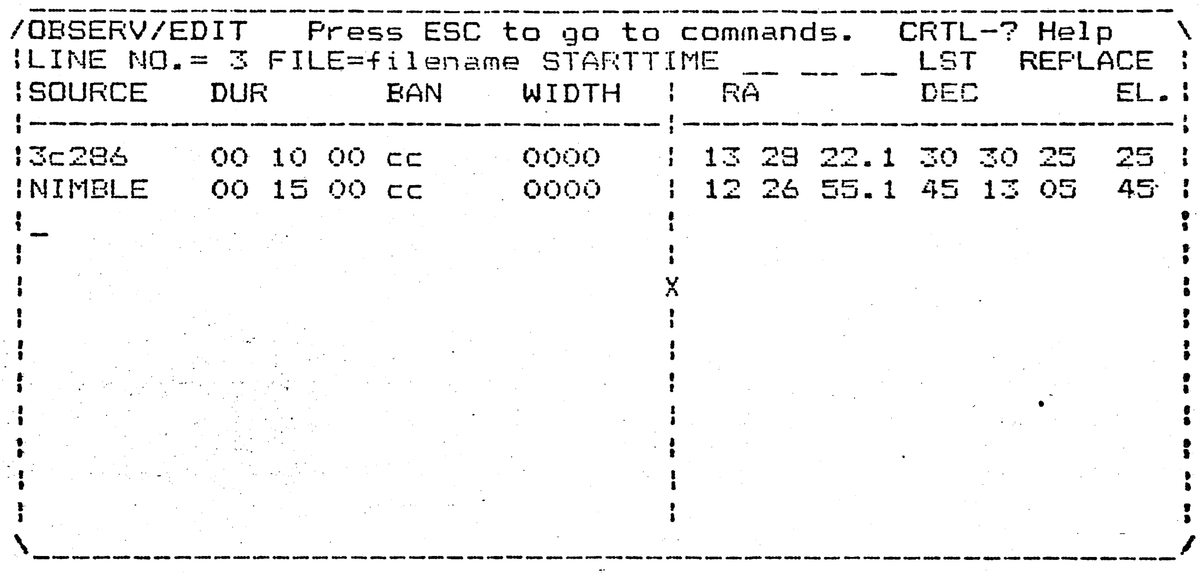
TRANSFORM

Convert display containing macros in Edit Window to a display with the macros expanded (i.e. the final observing file). Activation of the command a second time will return the Edit Window to the original state.

PARAMETERS

Set some of the instrument values (such as observin frequency, etc) to a given state. These parameters are particular to the instrument and remain in effect until changed.

The screen will normally look like the following when not in the command mode.



```
 ----------------------------------------------------------------------
/OBSERV/EDIT   Press ESC to go to commands.   CRTL-? Help     \
!LINE NO.= 3 FILE=filename STARTTIME __ __ __ LST   REPLACE !
!SOURCE    DUR        BAN     WIDTH   !  RA           DEC          EL.!
!-------------------------------------!-------------------------------!
!3c286     00 10 00 cc       0000     ! 13 28 22.1 30 30 25   25 !
!NIMBLE    00 15 00 cc       0000     ! 12 26 55.1 45 13 05   45 !
!_                                    !                               !
!                                     !                               !
!                                     X                               !
!                                     !                               !
!                                     !                               !
!                                     !                               !
!                                     !                               !
!                                     !                               !
!                                     !                               !
!                                     !                               !
\_____________________________________________________________________/
```

```
              ^                                 ^
              ^                                 ^
         Edit Window                     Display Window
```

The cursor can be moved around the edit window only. All the control keys specified in the GENERAL DESCRIPTION OF USER INTERFACE apply for cursor movement and editing. Note that a new line can be inserted between othe lines by terminating the last field of the Edit Window by pressing the return key while the INSERT mode (INS) is on.

After each field entry is terminated with the return or TAB keys, any syntax and validity checking possible are carried out.

Macros (defined with the GOMACRO command) may be utilized on any line in the Edit Window. There are some rules for doing so. The first field must contain a marco name (all macros start with the @ symbol). This capability allows the user to create an observing file by setting the standard options and using macros. This approach makes it easy to specify an observing file in the usual manner at the same time allowing the power and flexibility of utilizing macros. It is also be possible to create an observing file in the Edit Window using macros only.

The X in the screen display indicates at which point the parameters have been set. They can be changed as often as is necessary or desirable.

Bignell

To: Scientific Staff, Programming Staff

From: B. Clark

Subj: A possible alternative OBSERV

Date: 83jan24

In the meeting to discuss OBSERV, it was suggested that we have a look at possible radical revisions. This memo discusses such a possibility.

The DEC-10 command scanner does not contain macro capability, despite repeated efforts on the part of Jerry Hudson to sneak them in. Basically, the reason we do not perceive them as useful is that each "go" command does so much (that is, takes so long) that we are willing to organize each independently, and the convenience of macros is not sufficient to warrent the effort to set them up. It is possible that such is not the case for OBSERV; the convenience of macros, each describing, say, a calibrator-source-calibrator sequence, may be considerable.

The problem that immediately arises when one conceives of fitting macros into OBSERV is that one has two types of object--the macros and the macrocode that is expanded to the observing program, and the Modcomp style observing program itself. The problem is to keep these two types of objects separate but associated. The easiest way I can see around this is to keep only the macro-style program on the DEC-10 (or whatever), and to have the FETCH program decode it into the Modcomp style list which is sent to the Modcomp. The macro decoder would then be a module in the FETCH program and an identical module in OBSERV, or rather in OBSCHK. The paradigm for this module is a language processor, rather than an editor. The observers would prepare their programs with the system editor--EDT or SOS (or TECO or whatever). The OBSCHK program would have only commands for INFILE, OUTFILE, LISTOPTIONS, MINTIME, and MINELEVATION.

As a compromise between readability and conciseness, I suggest that the standard variable names be unique in four characters, and that these four characters (or any number beyond four) would abbreviate the variable name. Macro names would be forbidden to begin with the four characters of a standard variable. (We could take a pure MINMATCH approach, but I think it might well be annoying in a langauge processor, where it is not in an interactive program.)

In conformity with language processors, I suggest it is less confusing to have variables preserve their values until changed, rather than looking up defaults separately for each observation.

The language processor should have good diagnostics, and must not quit on syntax errors, but must continue to try to make the best of things and catch all errors in a single pass.

The formal definition of the language is given in the Bachus-Naur Form in the appendix at the end of the memo. A program consists of four sections in fixed order. The first section, the ID section, generates the Modcomp ID card, and, in addition, supplies the starting and ending times. The second section generates the

Modcomp local defaults for LO setting, etc, as a function of band. The third section defines the macros to be used to generate the program. The fourth section is the observing program itself.

There are slight extensions to the current observe commands, as well as the macro facility, to make the language more powerful. The first is the WHILE statement. This causes repetition of the "while group" of observations until a "while condition" is no longer satisfied. "While conditions" take two forms. The first is a list of source names and a limiting elevation. The "while group" is repeated until an observation of one of the named sources falls below the limiting elevation. The second is a time condition. The while group would terminate at the given time, or at the end time (given in the ID section) less a specified interval.

The second extension is the addition of a third way of specifying the stop time, ON_D[URATION], which adds the move time to get a DURATION.

There are several ways of specifying some parameters; for instance, STOP, DURA[TION], and ON_D[URATION] would specify the same thing in the end; the last one encountered would govern. This can be quite confusing when it is groups of things being set by one command, as when AFRE[QUENCY] sets BAND, L6A, and AFINE frequencies. I don't see much way out of this, except to let the last one govern.

Macros have no parameters, except for a few psuedomacros. <IAU_name> = CALI[BRATOR] causes the appropriate calibrator list (J2000 or B1950, as specified in the ID section) to be searched for the named calibrator; mentioning the IAU-name subsequently would then expand to NAME=<IAU_name> RA=<ra_from_list> DEC=<dec_from_list> GAINCODE=<gaincode_from_list>. (I think that there is sufficient latitude in calibrator observations that it is possible to pick a single gain code that will work at all bands). As mentioned above, mentioning AFREQUENCY is equivalent to BAND, L6A, and AFINE parameters.

I have been a little careless in the format definition about the names of the various parameters. I want to discuss the principle of a language processor, without thinking too much about parameter names, or of ensuring that I get all parameters.

An interesting, but I think inferior, variation of this scheme is to have the language structure as defined below as the input to a realtime compiler, which writes output in the Modcomp format, but which writes the macros into a separate file (say with the same name but a different extension) and with an editing facility.

Apendix B. Formal definition

```
<observe_program> ::= <id_section> [<defaults_section>] [<macro_section>] <program_section>

<id_section> ::= <id_list> ENDID <delimiter>
    <id_list> ::= <id_item>
              ::= <id_item> <id_list>
     <id_item> ::= <id_identifier> = <constant> <delimiter>
         <id_identifier> ::= PROG[RAM_CODE]
                         ::= USER[NUMBER]
                   ::= DATE[_OF_OBSERVATION]
                   ::= STAR[TTIME]
                   ::= ENDT[IME]
                   ::= COOR[DINATES]
                   ::= TIME[_TYPE]
         <delimiter > ::= <blank>
                ::= ,
                ::= <carriage return>
         <constant> ::= <command_scanner_constant>

<defaults_section> ::= [<defaults_block> <delimiter>] [<defaults_block> ...]
    <defaults_block> ::= DEFAULTS "2_character_block_name>" <default_list> ENDDEFAULTS "2_character_block_name" <delimiter>
     <default_list> ::= <default_item> <delimiter>
                 ::= <default_item> <delimiter> <default_list>
     <default_item> ::= <default_identifier> = <constant> <delimiter>
         <default_identifier> ::= PLUN[GE]
                     ::= WRAP
                     ::= PMRA[RATE]
                     ::= PMDE[CRATE]
                     ::= PMHO[RIZONTALPARALLAX]
                     ::= BIAS[DIGITS]
                     ::= FIRS[TLO]
                     ::= AL6L[O]
                     ::= BL6L[O]
                     ::= AFIN[ETUNING]
                     ::= BFIN[ETUNING]
                     ::= AFRE[QUENCY]
                     ::= BFRE[QUENCY]
                     ::= ARES[TFREQUENCY]
                     ::= BRES[TFREQUENCY]
                     ::= AVEL[OCITY]
                     ::= BVEL[OCITY]
                     ::= RCVR[FILE]
                     ::= SUBR[FILE]
                     ::= DSCH[ANNELS]
                     ::= DSST[ARTCHANNEL]
                     ::= DSIN[TEGRATION_TIME]

<macro_section> ::= MACROS <macro_definitions> ENDMACROS
    <macro_definitions> ::= <macro>
                  ::= <macro> <macro_definitions>
     <macro> ::= <macro_name> = {<macro_code>} [<delimiter>]
          ::= <IAU_name> = CALIBRATOR <delimiter>
         <macro_code> ::= <substitution_string_without_{_or_}>

<program_section> ::= <observation_block> [<observation_block> ...]
    <observation_block> ::= <observation_set>
               ::= WHILE {<while_conditions>} <observation_set> ENDWHILE <delimiter>
     <while_conditions> ::= [<elevation_condition>] [<time_condition>]
         <elevation_condition> ::= (<name_list>) > <angle_constant>
          <name_list> ::= <source_name> <delimiter>
```

Appendix A. Example

The example below uses the system default LOs, does sets of calibrator source calibrator at L and C bands, switches calibrators when the first gets too low, starts off with 3C286, and ends with 3C48.

```
PROG = AP46
  DATE = 52054
  START = 14:30  ENDT = 23:
  COORDINATES = B1950
  ENDID
MACROS
  1328+307=CALIBRATOR
  1730-130=CALIBRATOR
  1937-101=CALIBRATOR
  0134+329=CALIBRATOR
  MINE = {NAME=MINE RA=18:28:27 DEC=-17D30'17.3"}
  SUBSET = {ON_D=0:2 1730-130;
     MINE ON_D=0:10;
     1730-130 ON_D=0:2;}
  SET = {BAND = LL SUBSET BAND = CC SUBSET}
  SUBSET2 = {ON_D=0:2 1937-101;
     MINE ON_D=0:10;
     1937-101 ON_D=0:2;}
  SET2 = {BAND = LL SUBSET2 BAND = CC SUBSET2}
  ENDMACROS

DURA = 0:10 BAND = LL 1327+397 ;
DURA = 0:2 BAND = CC;
WHILE {(1730-130,MINE)>10d TIME < 0:10}
  SET
  ENDWHILE
WHILE {(1937-101,MINE)>8d TIME < 0:10}
  SET2
  ENDWHILE

DURA = 0:8 BAND = LL 0134+329;
WHILE TIME < 0:
  BAND = CC 0134+329;
  ENDWHILE
```

```
                   ::= <source_name> <delimiter> <name_list>
         <time_condition> ::= TIME = <time_constant>
                    ::= TIME < <time_interval_constant>
     <observation_set> ::= <observation> ;
                 ::= <observation> ; <observation_set>
         <observation> ::= <parameter_item>
                 ::= <parameter_item> <delimiter> <observation>
          <parameter_item> ::= <parameter_name> = <constant>
              <parameter_name> ::= NAME
                         ::= QUAL[IFIER]
                         ::= STOP[TIME]
                         ::= DURA[TION]
                         ::= ON_D[URATION]
                         ::= RA
                         ::= DEC
                         ::= EPOC[H]
                         ::= BAND[S]
                         ::= GAIN[CODE]
                         ::= WIDT[HS]
                         ::= MODE
                         ::= CALC[ODE]
                         ::= <default_identifier>
```

Remarks: some discression must be given to the program about terminating WHILEs. Probably the best rule would be that if the last observation has less than half of its specified duration (or on_duration), it should be scrubbed.

It is probably worth the while to have the program explicitly forbid macro recursion, rather than merely going into an infinite loop.
I think I would also forbid nested WHILEs. There is no obvious application without macro arguments or counters.

Solar system bodies could be added as additional psuedo macros, although unlike the others, they must be evaluated at program generation time, rather than at macro time. They could identify the object by name for the planets, by osculating elements for asteroids and commets, and by name and elements for planetary satelites.

To alleviate confusion with semicolons in macros, it might be well to have the sequence " ; ; " generate only one observation, rather than two with the same parameters. One then needs a special character, say "#", which doesn't do anything, but merely separates semicolons. " ; # ; " would generate two identical observations.

Note that the formal definition permits any mixing of STOPTIMEs, DURATIONs, and ONDURATIONs. However, the language processor would convert all of them to STOPTIMEs. The Modcomp OBSERV program could be kept sufficiently alive to convert STOPTIMEs to DURATIONs for those few occasions that it is really needed.

It might also be possible to have the processor optimize wrap choices. However, it is my impression that this consumes too much CPU time if a brute force approach is taken. I had a try a few years ago, and the program ran forever and produced little.

bgc:bgc