NATIONAL RADIO ASTRONOMY OBSERVATORY

Charlottesville, Virginia

October 11, 1974

VLA COMPUTER MEMORANDUM #116

# SYSTEM UPDATING AND SECURITY

Barry Clark

## I. Updating

This memo places into effect the recommendations of Jim Torson, VLA Computer Memorandum 107, for the maintenance and security of the synchronous system files. There are minor changes suggested here, so this memorandum replaces that.

There will be five disk partitions set aside as the programmers personal libraries, each capable of holding about 8000 lines of code. Each programmer may do what he likes with these libraries, using the source editing program SEDIT. Do not maintain the libraries with SMC -- that format is annoyingly difficult to backup. The usual logon procedure, when a programmer sits down at the console, will be to type /B/DEFAULT USL=DSx, when DSx is the programmer's library name.

There will be also a system source library, GSL, which will contain all of the source code necessary to completely recreate the system. The procedure is thus for a programmer to write the code and perform initial debugging using his own source library. When he feels it is ready to be used by others, he will fill out a system change form and submit it.

At regular intervals--until further notice at 10 a.m. Monday mornings--outstanding system change forms will be examined and the system updated. The sequence of events will be: subroutines will be assembled, those without assembly errors will be added to or replaced in the system subroutine library, as specified. Second, main programs will be assembled and linkedited. Those without error will be replaced or added to the background overlay library or foreground task library as specified. Third, those source modules successfully used in the first two steps will be added to the system source library.

The protected area of the disk will not be written by anyone other than the person charged with the responsibility of the system changes, with one exception. It will be permitted to write new systems, for testing purposes in the fourth file of DLS (reached by typing $AVF BO,4). Extreme care in doing so is enjoined.

## II. Backups

Six different backup tapes will be kept, so that all likely disasters may be recovered in the most expeditious possible way.

1. Immediately after the system update, a DTC dump of the first part of the disk will be taken, up through the programmer source libraries. This gives protection against something happening to the physical disk, when we would want to recreate the same disk very quickly.

2. This will be done on alternating tapes, so that if we discover that today's system contains an unexpected bug, we may very quickly go back to yesterday's.

3. Also immediately after the system update, a partition dump of the disk will be made, using for each partition the COPY command of the appropriate processor. This backup will be used, for instance, to restore an accidentally bombed source program in the programmer's library.

4. This, too, will be done on alternating tapes, so that a single module of yesterday's system may be restored to today's easily and quickly.

5. When the disk is first made up (a new disk will be made up whenever it becomes clear that the current partitions need to be revised) a partition dump will be made. Then, each system update, the source code will be written to a transaction tape as well as to GSL. This backup will be used for answering questions like "What version of program x was in use when Joe Blow was observing last year?"

6. Each system update, all programs will be written to a transaction tape. This backup takes care of the situation when it is suddenly discovered that a capability that a processor had last year, and is suddenly needed now, has been lost somewhare along the line.

## III. System Change Form

The top part of the form consists mostly of checkoff boxes describing the common things you might wish to do. You first enter your nane and the date. When the update is made, a code number will be assigned by the system updater, for filing and reference identification. Then, to identify the module underdiscussion, list its alias name, (the name by which it is known to SEDIT), which will be used to fetch the module. Also list the name by which it identifies itself to the link editor, the argument of PGM or of PROGRAM found in the code.

Next, check whether it is a subroutine (to be put in the system library and linkedited into other load modules), an overlay (like the background processors, to be loaded into the core of an established task), or a task (always including a task resource block). A task or overlay may be flagged as priviledged or unpriviledged. You should indicate that the program is a revision if this source is to replace an old source module in GSL, and if the object module is to replace a previous version. The system updater will fill in the source size (lines), the subroutine size (words) and the size of the linkedited program.

The usual case will be to find the source deck either in the programmers personal library, or, when he specifies recompiling an unchanged main program to include a new version of a subroutine, in the system library, GSL. The usual place for the system updater to put things will be DLB, the system subroutine library, for subroutines, DBB, system background overlays, for unpriviledged overlays, and DFF for tasks and priviledged overlays. If it is to go elsewhere, for instance to DLT for programs for Cora, this should be specified.

The source code will generally go to GSL. No code modules will be put into the system unless the source is available in GSL unless absolutely unavoidable.

The usual reasons for making a system change are to backup your own library (to make this version of your code available to you only, after the week that the usual system backup tape is saved), to put the source code into GSL (which makes it easily available for others to look at, steal, and possibly, use). Substantive changes include correcting errors, extension of function (with existing users noticing no difference), and major modifications. Major modifications are distinguished, for subroutines, into those that require a change in calling sequence, and those which don't. If you make changes in subroutines, you will likely want to make a new linkediting of one or more main programs to incorporate the new subroutine. In this case, you may submit a change form for each main program with the "link to new subroutines" box checked, and no further documentation required.

The lines for INTERNALS added/deleted and EXTERNALS added/deleted are so that a file may be maintained listing, by subroutine, the programs which call the subroutine.

On the reverse side of the system change form, is found space to describe the change made to the module. It is anticipated that this will not be sufficient space for an initial submission, for which you should attach additional sheets. A word or two about test procedures is requested to make sure that we do not stick naive users with insufficiently debugged code.