AT/25.1/031

The *EMPEE* Program

J.D. Argyros
12 Jan 87

1.0 Introduction

The *EMPEE* program is used to communicate to a remote node the list of points to be used during the next monitoring phase. The protocol was originally developed for use to the antenna control computer or the ACC but is expected to apply equally to the correlator control computer or the CCC and to any other node remote from the Synch computer.

Whenever a monitor point list needs to be dispatched to a remote node, then the program *EMPEE* will be run. This program will read a *master file* specification and use the contents of the files listed there to build up the tables of points to be monitored. The program *EMPEE* performs several additional functions:

(i) generates *requests to monitor* for the remote node;
(ii) partitions the global common areas used to hold the monitor point values in the Synch computer;
(iii) generates a description of each monitor point and where it is to be found in the Synch computer.

Once these operations have completed the program terminates. This means that if a list of monitor points needs to be changed, then all of the monitor point lists need to be reloaded into the remote node. The one exception to this is the points specified for the special maintenance mode for which items (ii) and (iii) above do not apply.

The *master file* specification is assigned to the logical name *EMPEE$MONITORPOINTS*.

2.0 The Monitor Points

Monitor points will be referred to by a descriptive name. This will be used to perform database accesses which will provide a complete description of the monitor point. This description helps locate the monitor point within the AT by providing it with an address. The address of a monitor point consists of three parts:

(i) the site (each antenna is considered a different site);
(ii) the *dataset* at each site;
(iii) the monitor point within the *dataset*.

The AT database has the name *ATDBASE*. Its CDD path is *CDD$TOP.AT.ATDBASE*. It is physically located by the logical *AT$DBASES*.

Other parts of the monitor point description would allow status and diagnostic displays to be built and to allow different processes within Synch to use the values returned for the monitor point.

In addition to the monitor points, their location, and their datatype, it is important to know the *class* that any one monitor point falls into. The *class* of a monitor point will determine its sampling frequency and what is done with the

data values returned to the Synch computer. The valid *classes* have been described in AT/25.1.1/039. In summary, these are

A: fast monitor

B: slow monitor

C: bit mask monitor

D: special maintenance mode monitor

S: Synch computer software monitor

We note the introduction of *class S* which has not been described previously. This class will be used to monitor software-generated values by programs run in the Synch computer. Because these monitor points are local, no monitor point request packets will be required. However by allowing the definition of the monitor point in the database, it will be possible to allow this class of monitor point to appear in the request files of monitor points and to have valid records generated for them in the *MPD* files (section 7).

3.0 Request files of monitor points

All monitor points will be requested in the same way. Each monitor point that is required will be entered into a text file, one name per line as follows:

monitor_point_name location_code

On the first line of the file appears the *class (A to D, S)* which describes how the monitor points included in this file are to be handled. In turn, the complete file specification is entered into a *master file* which contains a list of file specifications, one per line, to be used to specify the monitor points to be used.

There is no naming convention for either the *master file* or for the secondary files containing the list of monitor points. However it would be expected that the file names be appropriately descriptive.

In the *master file* only file names of the secondary files are listed, there being one file specification per line. The *EMPEE* program terminates when it detects an *end of file* conditions while reading the *master file*. Secondary files which are not found do not cause the program to terminate prematurely, rather an INFORM message is written to *OPER*.

Each secondary file has, as its first line, the *class* into which the following monitor points are to be placed. Monitor points of a single *class* only are allowed in a single file. However it is possible to have several files, each containing monitor points of the same *class*. Under normal operating conditions, it would be expected that at least three secondary files would be active with monitor points of *classes A, B* and *C*. When the program *EMPEE* detects the *end of file* condition on a secondary file, it finishes processing it and searches for the next secondary file.

The ! character is treated as a comment character. Any text following it, to the end of line, is ignored. The ! character can appear in any column.

Completely blank lines are ignored.

4.0 The database lookups

Every monitor point is described in the *ATDBASE* database. The information held there is used by the *EMPEE* program as follows:

(1) to build the monitor point request data packets used to describe the points to be monitored at the remote node;
(2) to allocate space in the global common areas for each of the monitor points returned to the Synch computer;
(3) to build the monitor point description file.

Only the following relations held in *ATDBASE* are used:

(1) monitor
(2) data_set_monitor
(3) monitor_limits
(4) data_sets
(5) data_set_type
(6) module_item

5.0 Building the monitor point request packets

Each request packet describes one monitor point only.

The following information is supplied in each monitor point request packet:

(a) general characteristics:
  (i) - 1- the letter **M**, the monitor point code;
  (ii) - 1- the *class* to which the monitor point belongs translated into the code recognized by the remote node: the letter **S** for slow monitor and bit mask monitor, the letter **R** (Regular) for the fast monitor, and the letter **H** for the special maintenance mode;
  (iii) - 8- the abbreviated name for the monitor point;
  (iv) - 1- the type of monitor point: the letter **D** for a monitor point in a dataset or the letter **G** for a software generated monitor point;
  (v) - 1- the data type for the monitor point: the letter **I** for a 2 byte integer, the letter **R** for a 4 byte F-floating real, the letter **C** for a variable number of characters;
  (vi) - 2- the number of bytes required by the data type as a 2 byte integer;

(b) the monitor point address:
  (i) - 2- the monitor point address as a 2 byte integer;

(c) the monitor point description:
  (i) - 4- the monitor dataset point type as characters;
  (ii) - 4- the scale factor as F-floating real;
  (iii) - 4- the offset added to the data after scaling, again as F-floating real;
  (iv) - 2- the length in bits of the data as a 2 byte integer;
  (v) - 2- the bit position (low order bit) where the datum starts as a 2 byte integer;

(d) the monitor point checking:
  (i) - 4- scaled upper check limit as F-floating real: 1E20 means no check;
  (ii) - 4- scaled lower check limit as F-floating real: -1E20 means no check;
  (iii) - 1- the action to be taken on a limit failure: the letter **N** indicates no action; **I** indicates an informative message is sent to Synch; **U** indicates an urgent message is sent to Synch;
  (iv) - 1- the process (type of checking) to be done: the letter **F** indicates absolute and fixed limits; the letter **D** indicates variable limits in degrees; the letter **V** indicates variable limits for non-angular measures; the letter **N** indicates no check;

(e) the remote identification for the monitor point which is returned with the data:
  (i) - 4- the *mode* to be used to locate the monitor point remotely using the items following, as characters;
  (ii) - 4- the *buffer code* as characters specifying into which buffer the monitor point is to be located;
  (iii) - 4- the *index* as a 4 byte integer indicating which 512 byte area of the buffer is used to locate the monitor point;
  (iv) - 4- the *offset* as a 4 byte integer (1 to 483) giving the offset into the area of the buffer where the datum is loaded (low byte).

6.0 Partitioning the Global Common Areas

None of the global common areas are partitioned before run-time. When the *EMPEE* program runs it allocates space in a global common area for it. This information is then communicated both to the remote node that has an interest in the monitor point and, via the *monitor point description* file, to any process in Synch requiring that knowledge.

The following algorithm is used to allocate the space in the global common areas. It is assumed that each common area is divided into a number of sub-areas, each of equal area, and of the order of 500 to 1000 bytes long. The special maintenance mode monitor points are **not** allocated any room in the global common areas. Fast monitor points are allocated space in a sub-area which is different from the sub-areas used by the slow monitor and the bit mask monitor which are allocated room contiguously in the same sub-areas.

For each *class*, space for a monitor point is allocated in the appropriate sub-area contiguously with the previous one. When one sub-area has been completely allocated, then another sub-area is located and the process continues. For the case when the monitor points for a *class* are distributed between several files, the allocation of space within the global common area proceeds with no break, i.e. the space continues to be allocated contiguously between the last point of one file and the first point in the next file.

7.0 The *monitor point description* file

Every time the *master file* is processed, another file is built which describes the monitor points that have been requested. This is the *MPD* file.

One *MPD* file is produced for each master file processed. It is an indexed file with the monitor point name and its location as the primary key. The name of the master file will be used to name the *MPD* file.

The extension (or file type) that will be taken by the *MPD* file is *mpd*. Other processes would then open the *MPD* file READONLY to obtain the information on the currently available monitor points.

The following information will be available in each record of an *MPD* file:

(a) monitor point name,
(b) location of the monitor point,
(c) abbreviated monitor point name,
(d) *class* of the monitor point,
(e) the datum type,
(f) the number of bytes occupied,
(g) the mode of the point as specified by "load common" (AT/25.1/022)
(h) the abbreviated global common area name of Synch where it is located,
(i) the sub-area within the global common area in which it is located,
(j) the offset of the first byte of the sub-area from the beginning of the data area with the first byte set to 1,
(k) the offset of the datum from the beginning of the sub-area with the first byte set to 1.

The primary key is formed by the "monitor point name" and the "location of the monitor point". The secondary key is formed by using the "location of the monitor point" and the "abbreviated monitor point name". The tertiary key is the "class" of the monitor point. It should be noted that the order of the first three fields of the record structure of the *MPD* file is important as the "location" is used in both the primary and secondary keys.