CERNET File Manager Services Implemented on IBM (- compatible) Computers

by

D. Wiegandt, CERN - DD - Division

1. Introduction

One of the main reasons for the creation of the CERN-wide packet switching network, now known by the name of CERNET, was the need for communication between minicomputers used in particle physics experiments and computer centre machines. This need resulted from the physicists' desire to pass samples of raw experimental data through the analysis chain of programs on the centre machines to permit a change of parameters of the experiment should the analysis of the events indicate such a modification.

Two methods were to be used:

- on-line sample analysis, performed by two collaborating processes, one in the minicomputer, the other one in a computer centre machine, hence the need for process-to-process communication.

- deferred sample analysis, in which case the minicomputer program writes a sample file on a computer centre disk and then starts up the chain of analysis programs, hence the need for file access and remote job entry.

2. The File Managers

It is the deferred sample analysis which, among other reasons, gave rise to the concept of CERNET file managers and also had a strong influence on their design.

The requirement to continue writing the same file after a restart following a breakdown of some vital part of the system led to the design of a file access system (as opposed to a file transfer system). File access is performed by two collaborating processes: the file requester process and the file server process or file manager.

The file requester is a user program or a utility program that runs in any of the computers connected to CERNET. There is a subroutine package called the UIP (User Interface Package) that implements the file access protocol on the requester side. Three basic data types are supported:

- Text

  transmitted through CERNET as ASCII code without parity. Successive blanks are replaced by an escape sequence. Code translation is performed at the originating computer and at the receiving computer when necessary. The unit of information a user program handles is a line of text.

- Blocked binary

  transmitted as blocks of information with an internal structure of variable length records. No code conversion is performed. The unit of information a user program handles is a record.

- Unblocked binary

  transmitted as block of information with no internal structure, stored on disk as received, padded with binary zeros to a byte or word boundary when necessary. The unit of information a user program handles is a block.

These three data types are mapped onto file formats supported by the host file system, i.e. the system the file manager runs on. In case of the IBM computers, text is stored in FB (fixed, blocked) data sets (accessible with a FORTRAN formatted READ), blocked binary in VBS (variable, blocked, spanned) data sets (accessible with a FORTRAN unformatted READ), and unblocked binary in Format U (undefined) data sets (not easily accessible from FORTRAN).

Quite a few additions were made to the capabilities of the file manager during the last years (see Fig. 1). Some of them were requested by users, e.g. support for PDS, special file formats, access to catalogue information, others were put in because new hardware had to be supported or new facilities had become available, e.g. HSM support, support of the 3850 Laser Printer, etc.

3. The design of the file manager in retrospective

Some principles that guided the design from the very beginning have retained their validity:

- Exploit the facilities of the operating system as far as possible but avoid everything that requires any modification of the operating system.
- Do not use dirty tricks, e.g. do not switch to supervisor mode in order to modify control tables used by the operating system.
- Aim at fault resistance but do not attempt to recover from fatal errors in the underlying operating system.

- Do early checks, i.e. catch user errors before sending erroneous or superfluous commands over the network.

- Return error messages in comprehensible English, neither "Error IE206I2" nor "File Error".

Deficiencies of the design which are now recognised are:

- the compression of successive blanks, meant to reduce transmission overheads created encoding/decoding overheads severely degrading transmission rates for minicomputers, and prevented block translation of the source text.

- the way in which a user can supply command parameters is unfortunately not very systematic, i.e. some have to be given as binary numbers, others as strings.

There are also some deficiencies in the IBM implementation of the file manager:

- only limited usage is made of subtasking. Most of the commands are handled by a fairly big main task (≃ 64k bytes). More parallelism might have been possible to separate file manager actions for different requesters more clearly.

- error recovery is more complicated when you have to keep the main file manager task alive in cases where a subtask could just have crashed and notified the main task.

4. Some "not too obvious" problems

- A problem which is fairly obvious is that of character translation, especially when you have to deal with character sets of different size. Decisions to map some special characters in one set onto "undefined" in a smaller set (e.g. CDC display code) may be disastrous for some applications.

- a less obvious problem is the handling of special characters in text lines, e.g. tabulator stops or line delimiting characters, e.g. carriage return, line feed, new line, or binary zero.

  They may be meaningful or mandatory on one computer and meaningless or even illegal on another.

- we have spent endless time on sorting out problems caused by the reverse ordering of bytes on DEC computers.

There is no way of successfully transferring a mixture of binary data and text between a DEC computer and another non-DEC computer on the network without detailed knowledge of the contents of what is to be sent so that you can selectively apply byte swapping.

- another puzzling problem was that the file manager would not prevent two requesters from simultaneously opening the same file for output. This is due to a deficiency in the file reservation mechanism in MVS that allows the same task (in this case, the file manager) to open the same file any number of times for output ("you have it anyway, so why bother").

5. Conclusion

The CERNET file managers have proven to be versatile tools to satisfy the needs of physicists for access to data and program storage and for providing utility functions. They have given a reliable service for many years already and the rate of usage is still growing. Many experiments at CERN depend critically on the availability of services provided by CERNET file managers, which shows the feasibility of remote file access and, at the same time, the level of confidence that has been reached.

Additions to the Original File Manager Design:

- Support for Partitioned Data Sets
- Access to Catalogue Information
- Support for Special File Formats
- Alias Data Types
- HSM support
- Date command
- Remote Job Handling
- File Archiving
- List Offline
- Simplified File Transfer

Fig. 1