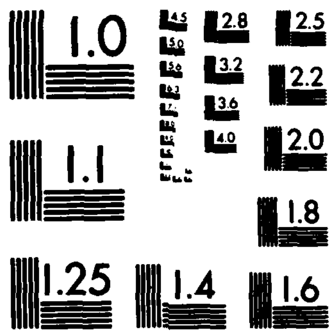
AD-A142 847 A SOFTWARE SCHEME FOR MANAGING MAGNETIC TAPE STORAGE SPACE(U) ELECTRONICS RESEARCH LAB ADELAIDE (AUSTRALIA) J C GWATKING ET AL. AUG 83 ERL-0285-TR 1/1

UNCLASSIFIED F/G 9/2 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(12)

ERL-0285-TR

AR-003-696

AUSTRALIA

AD-A142 847

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

ELECTRONICS RESEARCH LABORATORY

DEFENCE RESEARCH CENTRE SALISBURY
SOUTH AUSTRALIA

TECHNICAL REPORT

ERL-0285-TR

A SOFTWARE SCHEME FOR MANAGING MAGNETIC TAPE STORAGE SPACE

J.C. GWATKING and R.W. COLLIER

Approved for Public Release

AUGUST 1983

The official documents produced by the Laboratories of the Defence Research Centre Salisbury are issued in one of five categories: Reports, Technical Reports, Technical Memoranda, Manuals and Specifications. The purpose of the latter two categories is self-evident, with the other three categories being used for the following purposes:

Reports	:	documents prepared for managerial purposes.
Technical Reports	:	records of scientific and technical work of a permanent value intended for other scientists and technologists working in the field.
Technical Memoranda	:	intended primarily for disseminating information within the DSTO. They are usually tentative in nature and reflect the personal views of the author.

AR-003-695

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

ELECTRONICS RESEARCH LABORATORY

TECHNICAL REPORT

ERL-0285-TR

A SOFTWARE SCHEME FOR MANAGING MAGNETIC TAPE STORAGE SPACE

J.C. Gwatking and R.W. Collier

S U M M A R Y

This report analyses a problem which has been facing the DRCS central computing facility for a number of years: the continual growth in demand for magnetic tape storage within the computer room. It summarises the various techniques which have been proposed and implemented to reduce the rate of growth and finally describes in detail a software scheme which has alleviated the problem. The basis of the scheme is to categorise all tapes as either active or inactive, permitting the latter category to be stored outside the computer room.

**POSTAL ADDRESS: Director, Electronics Research Laboratory,
Box 2151, GPO, Adelaide, South Australia, 5001.**

DOCUMENT CONTROL DATA SHEET

Security classification of this page: UNCLASSIFIED

1 DOCUMENT NUMBERS		2 SECURITY CLASSIFICATION	
AR Number:	AR-003-695	a. Complete Document:	Unclassified
Series Number:	ERL-0285-TR	b. Title in Isolation:	Unclassified
Other Numbers:		c. Summary in Isolation:	Unclassified

3 TITLE

A SOFTWARE SCHEME FOR MANAGING MAGNETIC TAPE STORAGE SPACE

4 PERSONAL AUTHOR(S):

J.C. Gwatking and
R.W. Collier

5 DOCUMENT DATE:

August 1983

6 6.1 TOTAL NUMBER OF PAGES: 21

6.2 NUMBER OF REFERENCES: 11

7 7.1 CORPORATE AUTHOR(S):

Electronics Research Laboratory

7.2 DOCUMENT SERIES AND NUMBER

Electronics Research Laboratory
0285-TR

8 REFERENCE NUMBERS

a. Task: DST 80/076

b. Sponsoring Agency:

9 COST CODE:

228753/135

10 IMPRINT (Publishing organisation)

Defence Research Centre Salisbury

11 COMPUTER PROGRAM(S) (Title(s) and language(s))

12 RELEASE LIMITATIONS (of the document):

Approved for Public Release

Security classification of this page: UNCLASSIFIED

Security classification of this page: UNCLASSIFIED

13 ANNOUNCEMENT LIMITATIONS (of the information on these pages):

No limitation

14 DESCRIPTORS:

a. EJC Thesaurus Terms

Magnetic tapes
Storage
Availability
Organising

b. Non-Thesaurus Terms

15 COSATI CODES:

09020

16 SUMMARY OR ABSTRACT:
(if this is security classified, the announcement of this report will be similarly classified)

This report analyses a problem which has been facing the DRCS central computing facility for a number of years: the continual growth in demand for magnetic tape storage within the computer room. It summarises the various techniques which have been proposed and implemented to reduce the rate of growth and finally describes in detail a software scheme which has alleviated the problem. The basis of the scheme is to categorise all tapes as either active or inactive, permitting the latter category to be stored outside the computer room.

UNCLASSIFIED

TABLE OF CONTENTS

The boxes marked T represent either tape units or tape controllers and the box marked C represents the operator console on which tape requests are displayed. The racks used to store tapes requiring long term retention are marked R and the rack containing scratch tapes and short term usage tapes supplied by users for specific jobs is marked S. The areas marked A and B are largely unoccupied. It can be seen that operators have fairly quick access to the tape storage area from their normal position near the console.

2.2 Increased demand for tape storage space

The number of tapes requiring permanent storage in the racks R in figure 1 had been growing steadily and by the end of 1978 numbered about 1270 and occupied 5 racks. There was no sign of a decline in the demand for extra tapes and no more racks could be accommodated in the area adjacent to the tape units. It was therefore clear that tape storage space would soon become a problem.

2.3 Reduction of tape usage

One obvious answer to the space problem was to curb the use of magnetic tape. During the next couple of years (1979 and 1980) users were encouraged to convert tape based applications to use disk where practicable and to allow the archiving system(ref.1,2) to handle data requiring long term retention.

In addition, two software tools were introduced to provide better control over tape data sets. The first(ref.3) automatically transfers all small tape data sets to disk and the second(ref.4) provides a procedure for transcribing several data sets belonging to the same user on to a single tape. Both schemes have proved effective not only in releasing tapes but also in helping to educate users in good tape usage practices, and both are still in use. The success of the transcription process, in particular, is apparent from the fact that the 2610 tapes in the pool at the end of 1982 contained approximately 6500 data sets. Considering that the tape handling rules and practices encourage a one data set per tape situation(ref.4), this means that nearly 4000 tapes have been saved. However, this has not been without cost. The transcription process does incur considerable machine and operator overhead and for this reason it is not scheduled during periods of high activity.

2.4 Additional storage options

The introduction of the tape saving measures slowed down the rate of increase, but the number of tapes had still reached about 2000 by 1980 and the overflow was accommodated by placing extra racks in area A (see figure 1). However this was not a satisfactory long term solution because of the distance of these racks from the operators' normal location and because of the prospect of having to overflow still further into area B. A rearrangement of the computer equipment to more centrally locate the tape units was not possible because of other overriding considerations.

The idea of storing tapes in a compactus instead of fixed racks was considered. A compactus is a cabinet containing a number of movable racks which are normally "closed" to conserve floor space. Access can be gained to a particular rack by sliding the others to one side to form a passage way. Therefore by eliminating the need for permanent spacing between racks the compactus could perhaps double the number of tapes that could be stored adjacent to the tape units. However the access time to the tapes would be substantially increased because the compactus would firstly have to be opened at the correct rack. Simultaneous access to different racks would also be precluded. In addition it was felt that this would be a temporary solution at best since it would only double the number of tapes that could be accomodated. For these reasons the idea was rejected.

2.5 Tape usage patterns

Experience indicates that many of the data sets stored on magnetic tape in a research and development environment will seldom or never be accessed again. However there is a requirement to keep such data as a record of a completed experiment, for instance, or because the owner simply cannot be sure that it is no longer needed. This means that many, if not most tapes become relatively inactive after an initial period of activity. An analysis of the usage at the end of 1978 revealed that 50% of 1270 tapes had not been accessed for 100 days or more, 30% had not been accessed for 200 days and 20% had not been accessed for a year. These percentages were expected to increase as the number of tapes increased and the data sets aged, and this is confirmed by the figures at the end of 1982, which showed that 65% of the 2610 tapes had not been accessed for 100 days or more, 55% had not been accessed for 200 days and 45% had not been accessed for a year.

3. TAPE MANAGEMENT SCHEME OVERVIEW

The usage trends described in the preceding section led to the formulation of a tape management scheme to address the storage problem. The basic idea of the scheme is to limit the number of tapes stored in the computer room by dividing them into two groups: those "recently" used and those not "recently" used. Only the recently used ones would be stored in the computer room and the remainder would be stored in a compactus in an adjacent room, which has the capacity for storing a very large number of tapes if required.

It was estimated that only about 1000 tapes would need to be kept in the computer room and they would be stored in racks with the slots numbered consecutively, starting at 1. Each tape would have its slot number written on a small removable label placed on its external casing close to the tape's serial number so that both could be seen when the tape was placed in the storage rack. Exactly the same technique would apply to the tapes stored externally, except that the slots in the compactus would be numbered consecutively starting at 5000.

The concept of a unique slot number for each tape is an extremely important part of the scheme. Since tapes are no longer stored in serial number sequence the slot number provides the only way to determine the location of a tape.

To support this storage arrangement the tape management scheme must include software to provide three basic functions:

(a) It must modify tape mount messages for all tapes participating in the scheme to include the slot number.

(b) It must automatically record the last access date for all participating tapes.

(c) It must include facilities for the operators to interrogate and manipulate the scheme.

Further details of these functions and the manner in which they are implemented are included in the sections that follow.

4. OPERATOR FACILITIES

Although the operators should rarely need to interrogate or affect the operation of the tape management scheme the following facilities need to be available at an operator console.

(i) Should errors occur in assigning tapes to slots a conflict will sooner or later arise. To deal with such problems procedures must be supplied to interrogate the system to determine which slot a particular tape should reside in and which tape a particular slot should contain.

(ii) Since the usage patterns of tapes do change a procedure is required to shift tapes between the two storage areas (the computer room racks and the compactus), although it would only need to be invoked infrequently. It would identify those tapes in the compactus that have been accessed more recently than some of those in the computer room, and would produce a report informing the operators of the slots involved in the swap and their new occupants. The tapes would be swapped in pairs, so that the external slot labels on the casings could also be swapped. An extra feature of this procedure would be the ability to leave a certain number of computer room slots empty for subsequent "import" operations (see (iii)).

(iii) Periodically operators might identify a tape from the compactus that is becoming more frequently used. A procedure is needed to transfer a single tape into the computer room (to "import" it). If there is a free slot in the computer room the tape would be assigned to it. If not, the tape in the computer room with the oldest access date would be selected and the two tapes swapped. This import procedure would also be used to introduce new tapes into the system, if necessary displacing inactive tapes from the computer room into new or empty slots in the compactus.

(iv) A procedure is required to produce a cross-reference listing, indicating the occupants of each slot, in slot sequence, and the location of each tape, in tape sequence. This report would normally be produced after each reorganisation and kept for reference should some form of failure of the tape management software prevent the slot numbers from being inserted in the tape mount messages.

(v) A facility is needed to increase the number of slots in the computer room. The new number would be recorded by the software but would have no effect until the next reorganisation.

(vi) Normally only features (ii), a major reorganisation and (iv), a cross-reference report, need to produce printed output. The remainder only require responses to be displayed on the operator console. However an option should be available for the operator to request printed output from all procedures.

(vii) The existing tape erase program(ref.4) selects and erases tapes which no longer contain permanent data and reassigns them as scratch tapes. It should include a new option to confine its operation to the tapes in the computer room or to those in the compactus. Limiting the selection to the tapes already in the computer room would not cause any reassignment of slot numbers, and this would be the mode used on most occasions. However, periodically, the operators would select tapes from the compactus for erasure and reuse, and those tapes would have to be returned to the computer room. In this case the erase program would perform an "import" operation on the selected tapes, swapping them with inactive tapes from the computer room.

5. DETAILS OF SOFTWARE IMPLEMENTATION

5.1 Modifying tape mount messages

The operating system in use on the DRCS central computer is MVS. It issues several different formats of mount messages to the operators, depending on the type of event that caused the allocation of the tape. For example, the allocation may be initiated by a request in a JCL statement, it may be a dynamic request from either a batch job or a time-sharing user, or it may be caused by end-of-volume processing on a previous tape. The different message formats are produced by different operating system modules, some of which are relatively complex.

All of the mount messages which request a specific tape naturally include the tape volume serial number. For example, a typical format for message IEF233A is as follows:

```
IEF233A M 470,911082,,JCGJOB1,STEP2
```

where 911082 is the serial number. It was decided to insert the slot number immediately after the serial number in each of the messages, making it sufficiently prominent to easily catch the eye.

Using the above example again, and assuming that tape 911082 resides in slot 570, the new format of the message would be:

```
IEF233A M 470,911082(SLOT   570),,JCGJOB1,STEP2
```

One technique for implementing the changes to the message formats is to modify the operating system modules which actually produce them. This was evaluated but soon discounted due to the number of modules involved, the complexity of the task and the general policy of Computing Services Group to only make changes to operating system code when absolutely necessary. Instead an alternative technique was adopted, based on inserting a new module into the operating system as a pre-processor and/or post-processor to an existing module. This technique had been used previously in an experimental nature by one of the authors and is used by at least one major IBM program product, the Hierarchical Storage Manager(ref.5). The reasoning is that if there is a single module through which all of the requests of interest pass, then it should be possible to modify the request before passing it to the module and/or modify the output on return from the module.

Tape mount messages, like all other operator messages, must be processed by the Write-to-Operator SVC (SVC35)(ref.8) to actually place the information on the operator consoles. Module IEAVVWTO is the normal entry point to SVC35. However a new module, IGG035DU, was created and made the new entry point. The new module performs the following steps:

(i) It checks the identifier of the message and if it is not one of the four mount messages (IEC501A, IEF233A, IEF233D or IEF455D) lets it pass unaltered.

(ii) It extracts the tape volume serial number from the appropriate offset in the mount message.

(iii) It passes the serial number to an external task in a separate address space and receives back the corresponding slot number (or zero if the tape does not participate in the scheme).

(iv) If the slot number is zero the message is not altered. Otherwise the slot number is inserted in the message and the amended message is passed to module IEAVVWTO for display on the console.

The task that receives control in step(iii) above is described in detail in Section 5.4.

5.2 Recording last access date

The last access date of a tape is recorded by the external task when it is invoked to determine the slot number. However there is one other situation which must be handled by the new IGG035DU module to ensure that all accesses are recorded. This occurs when a scratch tape is requested. When the operator has mounted a tape the operating system issues message IEF705I to identify it. IGG035DU extracts the serial number and passes it to the external task to ensure that, if the tape participates in the scheme, the access is recorded.

5.3 Accessing tape and slot information

As mentioned previously an external task, operating in a separate address space, is invoked to maintain and retrieve information about slots, tapes and access dates. This arrangement was selected after considerable experimentation.

Initial attempts to make module IGG035DU perform the processing it requires itself, which would be a much simpler method of operation, were unsuccessful. The reason is that the module must be able to access a permanent data set (see Section 5.4.5), which contains the tape and slot information, in order to perform the necessary processing. However this was found to be extremely difficult because of the environment in which the module is invoked. The underlying operating system activity is allocation. In other words, the operator request to mount the tape volume occurs as part of the process of allocating a tape data set at the request of a user, and the allocation environment is still in effect when the Write-to-Operator function is invoked. Because the allocation environment is not recursive, the IGG035DU module cannot directly access the data it needs, since to do so requires that the data set containing the information also be allocated and opened.

Variations to the above technique were also tried. They involved creating and executing programs in other partially independent environments, though still within the same address space, to allocate and process the data set. The two techniques tested were subtasking and synchronous exit processing(ref.7). The idea was for IGG035DU to establish the environment for the other program, pass control to it, supplying the tape number, and then wait for it to execute and return the slot number. However, neither of the techniques gave enough independence from the calling environment to enable the allocation of the data set to succeed. Since the programs were still part of the same address space they were still under the influence of the underlying allocation request.

As a result of these tests it was decided to adopt the more complicated technique of using a separate address space to allocate and process the data set. The extra complexity is introduced because of the difficulties in communicating between address spaces and the need to synchronise the processing of, and control the access to, the "slave" address space from the one or more requesting or "master" address spaces which could be performing tape allocation simultaneously. This slave address space was given the name TAPEMAN.

Appendix I describes how the address spaces interact with each other.

Standard MVS techniques for address space communication, such as service request blocks (SRBs) and subsystem interfaces(ref.7) were considered but rejected either because they were unsuitable or would have required more effort to implement.

5.4 The TAPEMAN task

The TAPEMAN task is the focal point of the entire tape management scheme. Details of its operation and implementation are described below.

5.4.1 Requests from other address spaces

The primary role of TAPEMAN is to obtain slot information and record tape accesses on behalf of other address spaces. The reasons for this and the manner in which the requests are made have already been covered.

5.4.2 Operator requests

Besides handling requests from other address spaces, TAPEMAN also communicates with the operators and processes requests from them (see Section 4). The only exception is that the tape erase procedure is initiated independently and is not under the direct control of TAPEMAN.

The TAPEMAN address space is started automatically at each system initialisation (IPL). As soon as it begins execution it issues a console message requesting input, and then waits for the first request for service, either from the operator or from another address space, via the IGG035DU module. The operator will not reply to the console message until there is a need to interact with TAPEMAN. The reply is then in the form of a request for one of the services described in Section 4, or a request to terminate, which is the normal action just prior to closing the computer system down. The exact formats of these requests are defined in Appendix IV.

After it has processed an operator request TAPEMAN redisplays the console message, indicating that it can now accept further operator input, and then waits for the next request for service.

5.4.3 Batch operation

TAPEMAN also has an option that allows it to run as a normal batch job. In this mode it cannot accept requests from either the operator or from other address spaces. Instead it reads and processes commands, in the same form as the operator input requests described in Appendix IV, from a data set. When it has processed the last request, or when it encounters a termination command in the input, TAPEMAN terminates. This mode of operation is particularly convenient for testing changes to TAPEMAN.

5.4.4 Emergency procedure

In the event of TAPEMAN becoming unusable, due to program logic errors or corruption of its data, for example, a fall-back is available. The operators firstly terminate TAPEMAN and start the cleanup procedure, TAPEMANC, to reset fields in the communications work element (see Appendix II). When TAPEMANC finishes they then start the emergency procedure, named TAPEMAND, in place of TAPEMAN. TAPEMAND simply returns a "tape not found" condition to all work requests from other address spaces, which causes the IGG035DU module to let the tape mount messages pass unaltered for display on the consoles. The messages therefore indicate the tape number but not the slot number. The operators then use the current cross-reference report, which they should regenerate each time slots are reallocated, to locate the tape. Any operator input causes TAPEMAND to terminate.

5.4.5 Data storage

The information concerning tapes and slots has to be stored in some form of permanent data set where it can be accessed by TAPEMAN, and there are several requirements of the storage organisation used. In particular, the relationship between tape volume serial number and slot number must be recorded, as must the last access date of each tape. In addition the information needs to be accessed directly, via the tape serial number for some functions and via the slot number for others. Still other TAPEMAN functions require sequential access, in both tape number and slot number sequence. A VSAM key-sequenced data set(ref.6) keyed on tape number and with an alternative index keyed on slot number was considered the most suitable organisation to satisfy all of these requirements. The format of the records in the data set is shown in Appendix V.

5.4.6 Software

The TAPEMAN software consists of a driving program which extracts and interprets input commands from the operator and requests for service from other address spaces. There is also a series of subroutines to handle the various types of requests. These subroutines all open the VSAM data set containing the tape and slot information every time they receive control and close it again on completion. This allows each subroutine to select the combination of processing options (input or update, direct or sequential, tape serial sequence or slot sequence) suited to its task.

The code is written almost entirely in PL/I, the exceptions being two IBM System/370 assembler language routines required to invoke operating system functions not accessible from the PL/I language.

6. EXPERIENCE WITH TAPEMAN

6.1 Initialisation

When the time came to put TAPEMAN into operation the number of tapes to be retained in the computer room had to be determined. Obviously the number had to be large enough to give acceptable performance, which had been arbitrarily set at a maximum of three mount requests per day for tapes residing in the compactus. Previous analysis of SMF data(ref.10) had indicated that, with 50% of tapes stored in the compactus, this requirement could easily be met. The other consideration was the number of slots that could be comfortably accommodated in the computer room. A figure of around 1000 seemed to be the limit, which matched the other criterion quite well, given that there were just over 2000 tapes at that stage. The decision was therefore to use all 1000 slots from the outset, in order to optimise performance.

SMF data was again used to determine the 1000 most recently used tapes, and these were assigned as the initial occupants of slot numbers 1 to 1000. The remainder were assigned to the compactus slot numbers, starting at 5000. A report was produced indicating the assignments and this was used to physically relocate all of the tapes. All of the slots had been previously numbered and a removable slot number label was also affixed to the outer casing of each tape during the relocation, in a position where it could be seen when the tape was in position in the rack.

6.2 Operating characteristics

In the two years or so since the tape management scheme came into operation it has been necessary to use the TAPEMAND emergency procedure only once, while the VSAM data sets were restored from the backups. They had become corrupted due to an operator swap request which was not actually carried out.

The "import" function has been the operator service most frequently used, but not for the intended reason, which was to move selected high activity tapes from the compactus into the computer room. Instead the function has been used when transcribing the contents of tapes held in the compactus on to a single output tape (see Section 2.3). As a result extra scratch tapes have been made available in the computer room, displacing infrequently used tapes into the compactus. The tape erase procedure has the same effect when it operates on the tapes in the compactus (see Section 4 (vii)).

Many inactive tapes have also been forced into the compactus through the introduction of new tapes into the tape management scheme, without increasing the numbers stored in the computer room. The import function also controls this process.

The use of both the import function and erase procedure fluctuates with the consumption of scratch tapes by users. There have been periods when they have been used quite regularly (more than once a week), and other periods of several months when they have not been used at all. However together they maintain a sufficient flow of tapes between the two storage areas to keep the number of accesses to the compactus within acceptable levels. As a result the function designed specifically for this purpose, the "swap" function, has rarely been used. In fact it has not been used since the first few months of operation of the tape management scheme, during the settling-in period. The disadvantage of the swap function, unless it is used fairly regularly, is that it can cause a major upheaval by requiring a large number of tapes to be physically relocated. However, by using import and erase, the process is much more controlled, with relatively small numbers of tapes being involved each time. In addition a large number of unnecessary movements, caused by isolated accesses to tapes in the compactus, is avoided.

6.3 A design deficiency

One minor deficiency has so far been found in the scheme. The design does not include provision for reducing the number of slots in the computer room, only for increasing it. Although this is not a serious omission there has been one occasion when the facility would have been useful.

6.4 Performance statistics

The latest check of tape mount statistics to assess the performance of the tape management scheme was made over a period of seven working days in February 1983. All tapes were assigned to one of three categories. These were:

(i) Tapes which participate in the scheme. These are the tapes used for the storage and retention of user and archived data. There were 2610 tapes in this category, with 1069 in the computer room and 1541 in the compactus (soon after the scheme was introduced the initial allocation of 1000 slots in the computer room was increased to 1069, for no justifiable reason).

(ii) Other tapes which are stored permanently in the computer room, occupying a number of the racks R in figure 1. These tapes are used primarily for the daily backup of disk data sets and as log tapes for IMS (Information Management System)(ref.11).

(iii) User tapes which are supplied for specific jobs, and are then returned. These tapes are mainly used for data interchange between the central computer and various other computers and recording devices. While in the computer room these tapes are stored temporarily in the rack S in figure 1.

The statistics were:

(a) In the 7 days there were a total of 1194 tape mounts, or an average of 171 per day.

(b) The total number of requests for tapes in category (i) was 819 (average of 117 per day), for category (ii) 137 (20 per day), and for category (iii) 238 (34 per day).

(c) In the category of primary interest, category (i), only 7 of the total of 819 mount requests, or an average of 1 per day, were for tapes stored in the compactus. This represents about 0.85% of tape mounts in category (i), and about 0.6% of all tape mounts.

These statistics are representative of other observations that have been made and indicate that the tape management scheme is operating well within acceptable limits and is successfully performing the job it was designed to do.

7. SUMMARY

A scheme to alleviate tape storage problems within the computer room while minimising the distance operators are required to walk has been described. The scheme was proposed and implemented during a period of sharp increase in tape usage, and has operated successfully for over two years since then. In this period the rate of increase in tape usage has declined somewhat. This has been partly due to the influence of other tape management techniques, to user education and to the increase in direct access storage capacity of the computer system. Nevertheless the demand continues to grow.

An address space communication mechanism has also been developed to support the implementation of the tape management scheme. This mechanism can simultaneously service other applications and has also been used in the creation of an experimental address space to support automatic retrieval of data sets from the archives.

REFERENCES

No.	Author	Title
1	Gwatking, J.C.	"A Data Migration Scheme for the 370/168 Computer at WRE". WRE Technical Report 1870(A), June 1977
2	Gwatking, J.C.	"An Efficient Application of Disk Storage to the DRCS Data Migration Scheme". Technical Report ERL-0009-TR, April 1978
3	Rugless, P.A.	"Management of Catalogued Tape Data Sets on the IBM 370 Computer at DRCS". Technical Report ERL-0076-TR, May 1979
4	Gwatking, J.C. and Collier, R.W.	"Management Procedures for Controlling Data Storage on the IBM System 370 Computer at DRCS". Technical Report ERL-0055-TR, January 1979
5	IBM Corporation	"OS/VS2 MVS Hierarchical Storage Manager Logic". LY35-0026
6	IBM Corporation	"OS/VS2 Access Method Services". GC26-3841
7	IBM Corporation	"OS/VS2 System Programming Library: Supervisor". GC28-1046
8	IBM Corporation	"OS/VS2 System Programming Library: Debugging Handbook". Volume 1. GC28-0708
9	IBM Corporation	"OS/VS2 System Programming Library: Debugging Handbook". Volume 2. GC28-0709
10	IBM Corporation	"OS/VS2 MVS System Programming Library: System Management Facilities (SMF)". GC28-1030
11	IBM Corporation	"IMS/VS General Information Manual". GH20-1260

APPENDIX I

ADDRESS SPACE COMMUNICATION

This appendix describes the software written to allow communication between TAPEMAN (a slave address space) and other address spaces (master address spaces) requesting service from it.

I.1 Communications work element

The mechanisms which control the communication between master and slave address spaces were designed to be general purpose, so they could be used by other applications with requirements similar to those of the tape management scheme. They revolve around the concept of a communications work element (see Appendix II for details), which describes a request from the master to the slave. Each application, TAPEMAN being an example, has a single work element which is linked to that of the other applications via a chain emanating from an available word in the IBM control block known as the CVT (Communications Vector Table)(ref.9). Figure I.1 shows the structure of a work element chain for three applications. The work elements are all stored in that part of the addressing range known as the CSA (Common Systems Area), which means that they are accessible to all suitably authorised address spaces.

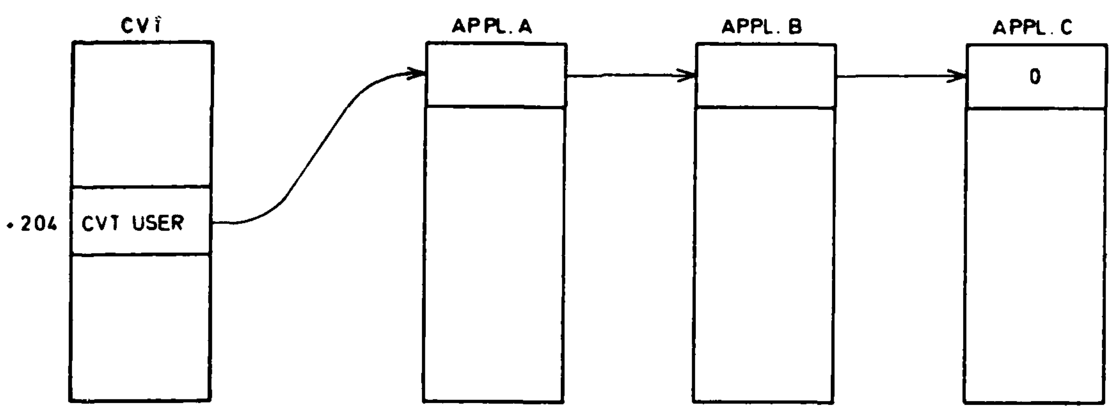


Figure I.1 Work element chain

The CVT is also available to all address spaces and its virtual address is known. Hence the addresses of each of the work elements can also be found.

I.2 Resource control

Since there can be more than one master address space attempting to communicate with a single slave simultaneously some form of control is necessary. The three resources to which access needs to be controlled in the environment described above are:

(a) the CVT, which is common to all applications,

(b) an application itself, and

(c) an application's work element.

The reason for representing an application as two resources ((a) and (b)) will be explained in the sections to follow.

These three resources are managed by use of the operating system ENQ/DEQ facility(ref.7) called from four communications interface subroutines, one of which is invoked by the application code in the master address space, (from IGG035DU in the TAPEMAN application), and three from the application code in the slave address space.

The ENQ/DEQ facility allows an address space to reserve or lock a resource for its exclusive use. If the resource is not available when requested the address space will wait until the current holder releases it.

I.3 Communications interface subroutines

Descriptions of the four communications interface subroutines follow. Their exact calling sequences are described in Appendix III.

I.3.1 Subroutine COMMINIT

This subroutine is called once from the slave address space during its start-up processing. It reserves the CVT and the requested application, and either resets an existing work element or constructs a new one and appends it to the work element chain (or starts the chain if no other applications are currently active). It then frees the CVT and application resources for other users.

COMMINIT will not create a new work element if there is already one for the same application. This may happen if the application's slave address space has been stopped (which does not remove the work element) and is now being restarted. In such cases COMMINIT instead reuses the old work element, resetting and updating its fields.

From this point the slave address space is ready to accept requests for work.

I.3.2 Subroutine COMMWAIT

COMMWAIT is called from the slave address space immediately after initialisation and subsequently after it has completed each work request. It places the address space in a wait state, pending a work request arriving from either the operator or from another address space via the application's communications work element. When a request arrives the operating system signals COMMWAIT, which immediately reserves the work element resource and passes details of the request back to the application program in the slave address space for processing. Because the work element resource is still reserved any other requestor must wait (see subroutine COMMWORK below).

I.3.3 Subroutine COMMPOST

This subroutine is also invoked by the slave address space after it has completed processing a work request but before it calls COMMWAIT to signal its availability for accepting a further request. If the request just completed was initiated by a master address space, COMMPOST will pass the results back to it via the work element and signal to the address space that processing has been completed. Finally COMMPOST will free the work element resource, at which point another requestor may acquire it and initiate a further work request.

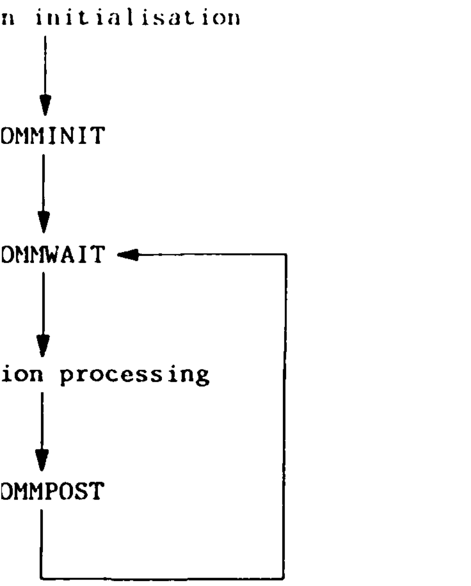
I.3.4 Subroutine COMMWORK

The final subroutine, COMMWORK, is invoked by a master address space when it wishes to communicate a request for work to a particular slave address space. COMMWORK firstly attempts to locate the application's work element on the chain emanating from the CVT. If it cannot find the required work element, or if the slave address space has been stopped, COMMWORK informs the operator that there is work waiting for the application and that it should be started. The subroutine then waits for one minute and repeats the check.

Once it has established that the slave address space is active COMMWORK reserves both the application and work element resources. It then places the information supplied by the caller to describe the request in the work element and signals the slave that it has some work to do, which is detected by the COMMWAIT routine. After this it frees the work element resource and waits for the slave to signal back that it has processed the request, via the COMMPOST subroutine. Once this completion signal has been received COMMWORK is reactivated and it frees the application resource for use by another requestor.

The need for a two level lock on the application (a lock on the application itself and on its work element) can be appreciated by examining what happens during a typical request for service. Initially a requestor obtains the application resource (via COMMWORK), which ensures that any other requestor has to wait. Then it acquires the work element resource while it sets information in the work element, after which it frees this resource and passes control to the slave, which in turn acquires the same resource for the duration of its processing. During this period (while the requestor is waiting for the slave to finish execution) the requestor holds the application resource and the slave holds the work element resource. Suppose that the requesting address space is cancelled before the slave finishes. This would free the application resource for another requestor. However, that requestor could still not proceed because it could not acquire the work element resource held by the slave. The slave can therefore complete the request initiated by the cancelled address space without interference. Without this second level of locking there would be no control over the work element and the second requestor and slave could update it asynchronously, causing one or both to fail due to inconsistent information.

Any future applications requiring address space communication can easily make use of these four subroutines to provide all the execution control that is necessary within both the slave and master address spaces. Figure I.2 illustrates where the subroutine calls need to be placed within the application code of the slave address space.



trol in the slave address space

APPENDIX

FORMAT OF COMMUNICATIO

This appendix describes the format of th
between requesting (or master) address spa
Appendix I for more details). In the tap
address spaces are those requesting a tape
is the TAPEMAN task.

The single work element for each applicati
Common Systems Area (CSA), to enable acces
The work element length is 40 bytes.

Offset	Length	Fi
0	4	address of ne
4	8	application i
12	4	application A
16	8	application j
24	4	ECB for appli
28	4	requestor ASC
32	4	ECB for reque
36	4	parameter add

Notes

(1) This is the name given to the appl
others and to uniquely identify its worl
management application the name is TAPEM

(2) The ASCB (Address Space Control B
defines the characteristics of a parti
also stored in a part of the addressi
authorised address spaces, so that one
of another. A requesting address spa
these two fields (ASCB address and jo
address space is still active. Having
via its ASCB, that there is some work fo

(3) An ECB (Event Control Block) is a
an address space that an awaited even
slave address space has nothing to do i
work request, and this particular ECB
happened.

(4) When a requesting address space ha
pauses and waits for the work to be c
uses the requestor's ASCB address and th

(5) If the requesting and slave address
define the request or to indicate the r
it via the address in this field. The

storage that contains the information in a pre-determined format. This area must also reside in the portion of the addressing range that is available to all authorised address spaces.

APPENDIX III

CALLING SEQUENCES OF COMMUNICATIONS SUBROUTINES

This appendix defines the calling sequences for the four communications interface subroutines described in Appendix I. Although developed in conjunction with TAPEMAN these routines have broader applicability and could be used for other applications requiring inter-address space communication. All the routines are written in IBM System/370 assembler language.

III.1 COMMINIT

This subroutine is called from the slave address space to initialise communication. The calling sequence is:

CALL COMMINIT(APPLID)

where APPLID is the 8-byte name identifying the application.

III.2 COMMWAIT

This subroutine is called from the slave address space to wait for the next request for work. The calling sequence is:

CALL COMMWAIT(APPLID,WORKELT,ECB)

Where:

(i) APPLID is the 8-byte name identifying the application.

(ii) WORKELT is a 40-byte area in which the work element describing a work request from a master address space will be returned.

(iii) ECB is an extra event control block on which COMMWAIT will also wait. This is application dependent and may be used for operator communication.

III.3 COMMPOST

This subroutine is called from the slave address space to inform the requestor that the work has been completed. The calling sequence is:

CALL COMMPOST(APPLID,WORKELT,RETCODE)

Where:

(i) APPLID is the 8-byte name identifying the application.

(ii) WORKELT is a 40-byte area containing the work element which was passed to COMMWAIT.

(iii) RETCODE is a full-word binary integer containing a return code from the slave address space to be passed back to the requestor. The use of the return code is entirely under the control of the application. For example, one possible use could be to pass back the result of the request (if it can be conveyed in four bytes).

III.4 COMMWORK

This subroutine is called from the requesting address space to signal the slave that it has some work to do and to await its completion. The calling sequence is:

CALL COMMWORK(APPLID,PARM)

Where:

(i) APPLID is the 8-byte name identifying the application.

(ii) PARM is the address of an area containing parameters and any other information required to define the request to the slave address space and to return the results.

APPENDIX IV

FORMAT OF OPERATOR REQUESTS

Sections 4 and 5.4.2 describe the manner in which operators interact with the tape management system and the facilities that are available. This appendix defines the input formats that may be specified in response to the outstanding operator message. They are as follows:

(i) TAPE=t or T=t

This returns the numbers of the slots containing the indicated tapes (t). t may be a single volume serial number, a range of serial numbers (specified as t_1-t_2), or any combination of the two forms.

eg TAPE=911014,911497-911499,911663

(ii) SLOT=s or S=s

This returns the serial numbers of the tapes residing in the indicated slots (s). As in (i), s may be a single slot number, a range of numbers, or any combination of the two.

eg SLOT=10-14,129,2314,2316-2319

(iii) IMPORT=t or I=t

This option is used to move tapes (t) into the computer room. They may be new tapes or ones from the compactus. Again t may be a single volume serial number, a range of numbers, or any combination of the two.

eg IMPORT=910930-910932,911333

(iv) REPORT or R

This option produces a cross-reference report.

(v) SWAP(,EMPTY=n) or SW(,E=n)

This forces a complete reorganisation, with inactive tapes from the computer room swapping positions with more active ones from the compactus. The optional parameter specifies the number of slots to be left free in the computer room.

(vi) LIMIT=n or L=n

This option increases the number of slots residing in the computer room. However, the change is not affected until the next reorganisation (IMPORT or SWAP).

(vii) PRINT or P

This causes hard-copy output to be produced for all requests. Normally it would be produced only for SWAP and REPORT.

(viii) END or E

This option terminates TAPEMAN.

APPENDIX V

DATA FORMAT

This appendix shows the format of the records in the VSAM data set containing tape and slot information (see Section 5.4.5).

Offset	Length	Field
0	6	tape volume serial number (primary key)
6	5	slot number (alternate key)
11	5	last access date in Julian form
16	4	unused

Figure V.1 Record format

There are records in the data set for all slots, including empty ones. These records have a null last access date and a tape volume field which contains the character 'Z' followed by the slot number. None of the participating tape volumes have serial numbers beginning with 'Z', so the records defining empty slots occupy their own unique key range, which simplifies access to them.

In addition there are two special records in the data set which contain parametric and statistical information. These records have unique keys to enable rapid access. One has the value 'SLOT1' in both the tape volume serial number and slot number key fields, and the other 'SLOT2'. The contents of these records are shown in figure V.2.

Offset	Length	Field
0	6	'SLOT1 '
6	5	'SLOT1'
11	5	highest assigned slot number
16	4	number of empty slots

Offset	Length	Field
0	6	'SLOT2 '
6	5	'SLOT2'
11	5	lowest compactus slot number
16	4	highest computer room slot number

Figure V.2 Special record formats

DISTRIBUTION

	Copy No.
EXTERNAL	
DEPARTMENT OF DEFENCE	
In United Kingdom	
Defence Science Representative, London	Title page
British Lending Library Division, Boston Spa, Yorks	1
In United States of America	
Counsellor, Defence Science, Washington	Title page
Engineering Societies Library, New York, NY 10017	2
Cambridge Scientific Abstracts, Riverdale, Md 20840	3
In Australia	
Chief Defence Scientist Deputy Chief Defence Scientist Superintendent, Science Programs and Administration Controller, External Relations, Projects and Analytical Studies	4
Director, Joint Intelligence Organisation (DSTI)	5
Document Exchange Centre Defence Information Services Branch for:	
Microfilming	6
United Kingdom, Defence Research Information Centre (DRIC)	7 - 8
United States, Defense Technical Information Centre	9 - 20
Canada, Director, Scientific Information Services	21
New Zealand, Ministry of Defence	22
Director General, Army Development (NSO), Russell Offices for ABCA Standardisation Officers	
UK ABCA representative, Canberra	23
US ABCA representative, Canberra	24
Canada ABCA representative, Canberra	25
NZ ABCA representative, Canberra	26
Defence Library, Campbell Park	27

Library, Aeronautical Research Laboratories	28
Library, Materials Research Laboratories	29
Library, H block, Victoria Barracks, Melbourne	30
Library, Royal Australian Navy Research Laboratory	31
DEPARTMENT OF DEFENCE SUPPORT	
Deputy Secretary (Manufacturing)	32
Deputy Secretary (Materiel and Resources)	
Controller, Defence Aerospace Division	
Controller, Munitions Division	
Library, DDS Central Office	33
Director, Industry Development Regional Office, Adelaide	Title page
WITHIN DRCS	
Director, Electronics Research Laboratory	34
Superintendent, Optoelectronics Division	35
Principal Officer, Computing Services Group	36
Mr P. Tomlinson, Computing Services Group	37 - 39
Authors	40 - 49
DRCS Library	50 - 51
Spares	52 - 57

END

FILMED

8

DTIC