DL/CSE/TM18

technical memorandum Daresbury Laboratory

DL/CSE/TM18

# RAPPORT USER GUIDE

by

G.D. FIRTH and C.M. JEFFCOCK, Daresbury Laboratory

DECEMBER, 1981

Science & Engineering Research Council

Daresbury Laboratory

Daresbury, Warrington WA4 4AD

# RAPPORT USER GUIDE

by

G.D. FIRTH and C.M. JEFFCOCK

## CONTENTS

## 1. INTRODUCTION

The aim of this guide is to help RAPPORT users by:-

i) indicating the steps involved in creating a RAPPORT database.

ii) making available details of installation dependent parameters which are not given in the RAPPORT manuals.

iii) documenting the conventions adopted by the Daresbury RAPPORT Users Group to permit the development of generalised and exchangeable software.

iv) documenting the general purpose procedures, clists and programs which have been developed, so that new users do not need to 're-invent the wheel'.

v) providing a list of existing RAPPORT applications so that new users may discuss problems with experienced users.

## 2. DISCLAIMER

When the Daresbury RAPPORT User Group was set up it was felt that a guide should be written so that new users of RAPPORT could benefit from the existing expertise in its use. This document has been produced in response to the demand. However it is recognised that as further experience with RAPPORT is gained a revision of the guide may be desirable and comments and criticisms to this end would be welcomed.

## 3. RAPPORT REFERENCES

Information on RAPPORT may be obtained from the following sources:-

i) The manuals written by LOGICA:-

RAPPORT - Designing and using a database
RAPPORT - User manual
RAPPORT - Interactive QUERY language user manual

These may be obtained from the User Support Group.

ii) The Daresbury RAPPORT Users Group.
The group was established in order to exchange ideas and software related to RAPPORT. Any user of RAPPORT is welcome to attend the user group meetings. For further details contact G.D. Firth (Secretary).

iii) The RAPPORT Newsletter, published at intervals by LOGICA.
Circulation of the Newsletter has been arranged by the RAPPORT User Group.

iv) The RAPPORT User Guide - a Daresbury technical memorandum.

The user requiring an overview of RAPPORT is recommended to read the following sections of the manuals listed in (i) above.

(a) From the user manual read:-

| | | |
|---|---|---|
| Section 1 | - | introduction (in full) |
| Section 2.1 | - | layout of data definition file |
| Section 3.1 | - | use of files, records and fields |
| Section 3.2 | - | general rules for RAPPORT commands |

(b) From the designing and using a database manual read:-

| | | |
|---|---|---|
| Section 1 | - | introduction (in full) |
| Section 2 | - | logical database design (in full) |
| Section 3.1 | - | general (physical design) |
| Section 3.2 | - | how hashing works |
| Section 3.3 | - | how indices work |
| Section 4.1 | - | introduction (to maintenance) |

(c) From the interactive QUERY language user manual read:-

| | | |
|---|---|---|
| Section 1 | - | introduction |
| Section 2 | - | general features |

## 4. ADMINISTRATOR AND MANAGER

In any database project coordination and management are of great importance. In order that adequate control of RAPPORT projects is maintained two levels of database control have been identified as being applicable. These are the RAPPORT - wide level and the application level.

Control of RAPPORT at the first level is to be carried out by a database administrator, responsible for ensuring RAPPORT software is properly maintained, new releases are made available, controlling and coordinating use of the multi-user nucleus (see Section 6 below), providing advice on database projects and persuading users to adopt the conventions given in this guide. The database administrator should be aware of current and proposed applications and ensure users with areas of joint interest make contact with each other.

Control at the second level will be achieved by nominating a database manager for each RAPPORT application. The database manager should preferably be someone who will have a high level of involvement with operational running of the database and be responsible for coordinating modification and development of the particular database. Requests to draw on information stored in a particular database should be addressed to the appropriate database manager.

Once it has been decided that RAPPORT will be considered for use in a project the future database manager should discuss the proposed application with the database administrator. This is of particular importance if the multi-user nucleus is to be used but should also serve to let new users benefit from existing experience at the earliest possible date.

## 5. CREATING A SINGLE-USER RAPPORT DATABASE

The terms single-user and multi-user are used to refer to the RAPPORT nucleus being used. When the single user nucleus is used information in the database may only be updated when access to the database is limited to a single task. When the multi-user nucleus is used several tasks may update the database simultaneously while the nucleus ensures data is not corrupted. A single-user database may be accessed by several tasks if access is limited to referencing existing data. See the section on concurrent access to the RAPPORT user manual for a full discussion.

The following steps are required when creating a single-user database:-

i) create a DDF
ii) preprocess the DDF to give a DIL file
iii) create the direct access files
iv) EMPTY the direct access files
v) load the QUERY questions and messages files

### i) Create a DDF (DDF = Data Definition File)

Full details of the DDF format are given in Section 2 of the RAPPORT user manual. Before a DDF is created it is recommended that Section 8 of this guide on reserved channels is read. The DDF file is of the standard format created by the TSO editor when a type of DATA is used. It is used to define the file names, sizes, channel numbers, indices and fields of the database. If it is possible that the database will later be used with the multi-user nucleus then the following section on creating a multi-user RAPPORT database should be read as certain restrictions on field names, exist for the multi-user nucleus.

### ii) Preprocess the DDF to give the DIL file

The DIL file is produced as output from the RAPPORT preprocessor, together with a compileable FORTRAN source, and must be attached to every job and TSO session where the database is accessed. The information held in the DIL file is derived from the information held in the DDF. By listing the DIL file it is possible to determine what size the direct access files to be created in the next step will be. The information is given in a table at the end of the DIL file in the format:-

| | | |
|---|---|---|
| number of blocks required on channel | X | abc |
| number of blocks required on channel | Y | efg |
| number of blocks for database dump | | hij |

The clist RAPPREP, detailed in section 11 of this manual may be used to preprocess the DDF file and give the DIL file.

### iii) Creating the Direct Access Files

Each RAPPORT database resides on one or more direct access files, each of which may contain one or more RAPPORT files. The RAPPORT files which will reside on each direct access file are determined from the information held in the DDF. The clist RAPMAKE, detailed in section 11 of this guide, may be used to create the direct access files. In order to make best use of the general purpose clists and procedures developed the files should be named in accordance with the conventions given in section 7 of this guide.

### iv) EMPTY the Direct Access Files

Before a direct access file can be used to contain RAPPORT files it must first be emptied. Emptying a direct access file creates a check block in it containing details of file size etc. Each time the database is opened, by subroutine DBDEF, the information in the check block of every direct access file belonging to it is read and the information in it tested against the information held in the DIL file, thus determining whether the database is consistent. The direct access files can be emptied using the RAPEMPTY clist detailed in section 11 of this guide.

### v) Load the QUERY Files

This step is only necessary if it is intended to access the database via the interactive QUERY language. When using the interactive QUERY language 3 RAPPORT files must be present, the QUESTIONS file, the MESSAGES file and the COMSEQ file. The COMSEQ file is used for storing command sequences created by the user, but the QUESTIONS and MESSAGES files are used for storing information supplied by LOGICA to 'talk' to the interactive QUERY user. These files must be loaded by running the program IQPREP. The information for the QUESTIONS file is held in the IQPREP program, but the information for the messages file is held in the file DNPL.RAPPORT.MESSAG.

The following statements must be included in the DDF if the interactive QUERY language is to be used:-

```
FILE QUESTIONS
RECORDS 751; CHANNEL 14
FIELDS
JQUNUM
JQUIN PKEY; HMULT 31
JQUOUT
KQUMOD CHAR; PKEY; UNHASH
KQUSYM(7) CHAR; PKEY; HMULT 11
JQUDAT
JQULIS
JQUSAV
INDEX
RECORDS 701; HMULT 7
JQUIN
FILE MESSAGES
RECORDS 301; CHANNEL 14
FIELDS
JMEMOD PKEY; UNHASH
JMENUM PKEY; HMULT 23
KMESS(15) CHAR
JMEPAR PKEY; UNHASH
FILE COMSEQ
RECORDS 501 CHANNEL 14
FIELDS
KCSNAM(3) CHAR; PKEY
JCSLIN PKEY; HMULT1
KCSPCK(17) CHAR
JCSEND
```

Notes

- The sizes of some arrays are varied from the sizes given in the RAPPORT interactive QUERY language user manual. This is due to certain machine dependencies and the sizes should not be altered.
- If it is required that the QUERY files occupy less space the BITS option may be used for certain fields, as detailed in appendix C of the RAPPORT Interactive Query language user manual. However it should be realised that use of the BITS option implies a computational overhead file on access.
- different channel numbers may be used for the files.
- the number of records in the COMSEQ file must be determined by the user to reflect the number of command sequences to be used. As the COMSEQ file is like another RAPPORT file if the initial estimate of its size is too small it may be increased at a later date.

## 6. CREATING A MULTI-USER RAPPORT DATABASE

Using the multi-user nucleus enables several different tasks to access a database concurrently without risk of losing data through multiple updates etc. At present there is only one multi-user database at Daresbury and all multi-user applications will become part of it. As this will involve data from a wide range of applications being present in the database it is obvious that great care must be taken in all operations concerning the database which could lead to its possible corruption. This means that such tasks as changing the DDF must be performed by a single person, the nominated database administrator. It is envisaged that applications will be developed as single-user databases and then added to the multi-user database when operational.

The convention that all field names for a given application will start with a given letter (determined by the database administrator) has been adopted.

The following steps are involved in creating a multi-user RAPPORT database:-

i) Contact the database administrator and determine what file names may be used and what letter fields for the application are to start with.

ii) Develop the application as a single-user database.

iii) Contact the database administrator and arrange for files etc to become part of the multi-user database.

## 7. FILE NAMING CONVENTIONS

The following conventions have been adopted:-

i) programs to be given names of the form

ID.PROG.FORT (where PROG is the program name)

ii) all other files have names of the form

ID | database name | file description

where database name may be in a qualified or unqualified form, for example SRSCAT or GU.RAPPORT.

For files specific to a users application (e.g. a file to hold details of grants of machine time at Daresbury) the file description is chosen by the user. However there are certain files which will be common to many applications in terms of purpose. A list of standard file descriptions for these is given below:-

DDF.DATA - The database definition file.

DIL - The preprocessed database definition file (COMMON file).

AIF - File holding logged After-Images for rolling through when recovering a dumped database.

BEF - File holding logged Before-Images for backing out of transactions.

CUR - File holding the current block of After-Images.

SORT - Work file for sorting records.

DUMP - File holding a dumped database.

QUERY - To hold the files associated with using QUERY when all three files are grouped on a direct access file.

Examples of file descriptions specific to a users applications are:-

STATIONS, GRANT.DATA

Note that the description maybe qualified as in the second example.

iii) Any necessary link edit statements should be stored in a member of a PDS:-

ID.LINK.DATA

## 8. RESERVED CHANNEL NUMBERS

A list of channel numbers used by RAPPORT is given below. Although it is not necessary that the DIL file is attached to channel 9 a convention that channel 9 is used has been established. It can be seen from the list below that a particular database may not use all the reserved channel numbers, however it is recommended that these channel numbers are not used for any purpose other than those listed below.

| Channel | Use |
|---|---|
| 2 | used to attach the DDF file when loading the QUERY questions and messages file. |
| 3 | terminal. |
| 4 | terminal, printer. |
| 6 | terminal, printer. |
| 7 | After image file with multi-user nucleus. |
| 8 | Before image file with multi-user nucleus. |
| 9 | DIL file. |
| 10 | Current block file with multi-user nucleus. |
| 11 | Before image file with single-user nucleus. |
| 12 | SORT file. |
| 13 | DUMP file with single-user nucleus. |

## 9. INSTALLATION PARAMETERS

When RAPPORT was installed it was necessary to set values for parameters used by the RAPPORT program. Many of these are machine dependent constants as well as dimensions of arrays used in common blocks. However certain of the parameters may be varied by the database administrator, to whom any comments on the parameters should be made, with the aim of improving any new release of RAPPORT. The parameters may be described under the following headings:-

Nucleus parameters

Machine dependent parameters

Preprocessor parameters

Backup and recovery parameters

Multi-user locking parameters

Interactive QUERY language parameters

Details of parameters which will be useful in developing RAPPORT applications are given below:-

Nucleus parameters:

MAXCON(20) number of conditions allowed in one SEARCH, FETCH or DELETE statement.

MCHNS(99) Highest channel number available for database files and indices.

MFILE(31) maximum number of files +1.

MLIST(61) maximum number of lists (files + indices) +1.

MPROBF(360) size of program buffer in words.

MWCOMP(240) maximum number of words in name key and index key for all files.

MWFORM(480) maximum number of fields referred to in all files and indices.

MWTEST(360) maximum number of files and fields.

MSKEY(20) maximum number of fields in an ORDER statement.

MDEPT(5) maximum depth of nesting of sorted retrievals.

Machine dependent parameters:

MZIMIN (-2147483647) largest negative integer.

MZIMAX (2147483647) largest positive integer.

MAXPOW($10^{75}$) power of 10 of largest positive real.

MINPOW($10^{-75}$) power of 10 of smallest positive real.

MBWOR(780) number of words in a block (record).

Preprocessor parameters:

MFIELD(420) maximum number of fields in the database (not counted twice if in indices).

MFLD(30) maximum number of fields in any file.

MIND(10) maximum number of indices in a file.

MSRCH(10) maximum depth of nesting of SEARCH loops.

MRHS(40) maximum number of characters on the right hand side of a condition.

MINFLD(10) maximum number of fields in an index.

Interactive QUERY language parameters:

MACHS(200) maximum number of characters in all values for arguments in nested command sequences.

MLILEN(80) maximum length in characters of a line read from the terminal.

MARGS(30) maximum total number of arguments of nested command sequences.

MCSTXT(68) maximum number of characters in the text of a command sequence line.

MCSCHAR(24) maximum number of characters in a command sequence name.

MWHOLE(12) field width in characters used to output all real numbers (includes MFRAC).

MFRAC(5) number of decimal places used to output all real numbers.

MINT(11) field width in characters used to output all integer numbers.

## 10. CATALOGUED PROCEDURES

The catalogued procedures listed below have been created to perform RAPPORT functions. For those marked with an * a clist exists to submit a job which uses the catalogued procedure, details of the clists being given in section 11 of this guide.

i) RAPPREP* to preprocess a RAPPORT/FORTRAN source.

ii) RAPPCL* preprocess, compile and link (NCAL).

iii) RAPPCLL* preprocess compile and link.

iv) RAPPCLG preprocess, compile, link and go.

v) RAPLINK* link edit to give executable RAPPORT program.

i) RAPPREP

This procedure preprocesses a RAPPORT/FORTRAN source. This will be of use when a single-user database has been altered such that a new DIL file must be prepared.

Parameters are:-

| | | |
|---|---|---|
| ID | = '*' | Used to identify files. |
| DBNAME | = '*' | The database name (see section on naming conventions). |
| PROG | = '*' | The program to be preprocessed. |
| CLASS | = A | Sysout class for output. |

Note that the preprocessed source is written to the dummy writer.

ii) RAPPCL

This procedure preprocesses, complies and links (NCAL) a RAPPORT/FORTRAN source. A non-executable module is placed on the library. This is of use when several program sources are being used to give one executable module or when RAPPORT routines are linked to the users routines in a link and go job.

Parameters are:-

| | | |
|---|---|---|
| ID | = '*' | Used to identify files. |
| DBNAME | = '*' | The database name (see section on naming conventions). |
| PROG | = '*' | The source program. |
| MEMB | = TEMP | Name of the NCAL module. |
| LIB | = '&&LIBRARY' | Library the NCAL module goes on. |
| CLASS | = A | Sysout class for output. |

iii) RAPPCLL

This procedure preprocesses, compiles, links (NCAL) and then links all routines (including RAPPORT ones) to give an executable load module from a RAPPORT/FORTRAN source. This is used to form an executable load module.

Parameters are:-

| | | |
|---|---|---|
| ID | = '*' | Used to identify files. |
| DBNAME | = '*' | The database name (see section on naming conventions). |
| PROG | = '*' | The source program. |
| MEMB | = TEMP | Name of the load module. |
| LIB | = '&&LIBRARY' | Library the load module goes on. |
| CLASS | = A | Sysout class for output. |

iv) RAPPCLG

This procedure preprocesses, compiles, links and executes a RAPPORT/FORTRAN source. Note that you will need to include your database files by means of override statements.

Parameters are:-

| | | |
|---|---|---|
| ID | = '*' | Used to identify files. |
| DBNAME | = '*' | The database name (see section on naming conventions). |
| PROG | = '*' | The source program. |
| CLASS | = A | Sysout class for output. |

v) RAPLINK

This procedure links standard routines, RAPPORT routines and previously linked (NCAL) modules to give an executable load module. This is of use when several sources programs are used to give one executable module.

Parameters are:-

| | | |
|---|---|---|
| ID | = '*' | Used to identify files. |
| LINK | = '*' | Linkedit statements in PDS &ID.LINK.DATA(&LINK). |
| MEMB | = TEMP | Name of the load module. |
| LIB | = '&&LIBRARY' | Library the load module goes on. |
| CLASS | = A | Sysout class for output. |

## 11. CLISTS

General purpose RAPPORT clists are detailed in this section. The clists reside in RA.CLIST. If they are to be used frequently it is suggested that you include RA.CLIST in the concatenation of clists produced by your start-up command.

An assist command has been created in RA.CLIST to provide assistance in using the available clists. The command is of the form:-

ASSIST CNAME

or EXEC 'RA.CLIST(ASSIST)' 'CNAME' if you have not concatenated RA.CLIST into your command libraries, where CNAME is the name of the clist on which information is required.

Available clists are:-

| | |
|---|---|
| i) RAPEMPTY | - to empty a RAPPORT file. |
| ii) RAPLINK | - link edit to give an executable RAPPORT program. |
| iii) RAPMAKE | - to create the direct access files which will contain a RAPPORT database. |
| iv) RAPPCL | - preprocess, compile and link (NCAL). |
| v) RAPPCLL | - preprocess, compile and link. |
| vi) RAPPREP | - to preprocess a RAPPORT/FORTRAN source. |

i) <u>RAPEMPTY</u>

The clist is used to empty an NAS file for use by RAPPORT. Emptying a file causes the check block used by the DBDEF routine to be created and the remainder of the file to be set to zeros, which is necessary before it can be accessed by RAPPORT routines. Before using the command the disk on which the NAS file resides must be on line and a 350K TSO region in use. The clist is interactive and starts by outputting the name of the DIL file and the file to be emptied to the screen. It then checks whether the file to be emptied exists - if not the command aborts with the message: 'file does not exist - use RAPMAKE to create it'. The existence of the DIL file is also checked and the user is then asked if it is 'O.K. to proceed'. An answer 'no' causes the clist to abort; an answer of 'yes' causes the program RDEMPTY on RAP.LOAD to be executed. The program prints the name of the file to be emptied on the screen and asks for a channel number or an 'end'. The user must enter the channel number, previously specified in the DDF. The question is repeated to which the user must again reply with the channel number. When the question is repeated once more the user should enter 'end'.

Note that the command must be repeated for each NAS file in the database - it is not possible to empty more than 1 NAS file at a time.

Positional parameters are:-

| | |
|---|---|
| DIL | The preprocessed DDF, or DIL file. |
| FILE | The file to be emptied. |

The one keyword parameter (default in brackets) is:-

| | |
|---|---|
| ID(*) | The ID prefixed to the file names. |

ii) <u>RAPLINK</u>

The clist submits a batch job which links standard routines, RAPPORT routines and previously linked (NCAL) modules to give an executable load module.

The positional parameter is:-

| | |
|---|---|
| LINK | The linkedit statements stored in &ID.LINK.DATA(&LINK) |

Keyword parameters follow, default values being given in brackets

| | |
|---|---|
| MEMB(TEMP) | The name of the load module. |
| LIB(&&LIBRARY) | The library the load module goes on. |
| ID(%U) | The ID the job will run under. |
| ACCT(%G) | The account number for the job. |
| NAME(%U) | The name field in the job card. |
| JOB(P1) | The jobname. |
| CLASS(A) | Sysout class for output. |

iii) <u>RAPMAKE</u>

The clist submits a batch job which creates a Fortran direct access file suitable for use with RAPPORT files on a specified volume with a specified size etc. Should the file already exist the user will be given the option of causing the file to be deleted and recreated or abandoning the command.

The positional parameter is:-

| | |
|---|---|
| FILE | The name of the file to be made (ID is prefixed to this) |

Keyword parameters follow, default values being given in brackets

| | |
|---|---|
| SIZE(10) | The number of blocks of data in the file. |
| CLASS(A) | Sysout class for print. |
| ID(%U) | The ID the job will run under. |
| ACCT(%G) | The account number for the job. |
| DISK(*) | The disk the file is to be created on. |
| UNIT(3330-1) | The unit type of the disk. |
| BYTES(3120) | The number of bytes per block of the created file. |
| WORDS(780) | The number of words per block of the created file. |
| NAME(%U) | The name field in the job card. |

Note that BYTES=WORDS*4, and that the values of these parameters were fixed when RAPPORT was installed at Daresbury.

iv) RAPPCL

The clist submits a batch job which preprocesses, compiles and links (NCAL) a RAPPORT/FORTRAN source. An object module is placed on the library.

The positional parameters are:-

| | |
|---|---|
| PROG | The source program. |
| DBNAME | The database name. |

Keyword parameters follow, default values being given in brackets

| | |
|---|---|
| MEMB(TEMP) | The name of the NCAL module. |
| LIB(&&LIBRARY) | The library the NCAL module goes on. |
| ID(%U) | The ID the job will run under. |
| ACCT(%G) | The account number for the job. |
| NAME(%U) | The name field in the job card. |
| JOB(P1) | The jobname. |
| CLASS(A) | Sysout class for output. |

v) RAPPCLL

The clist submits a batch job which preprocesses, compiles and links to give an executable load module from a RAPPORT/FORTRAN source.

The positional parameters are:-

| | |
|---|---|
| PROG | The source program. |
| DBNAME | The database name. |

Keyword parameters follow, default values being given in brackets

| | |
|---|---|
| MEMB(TEMP) | The name of the load module. |
| LIB(&&LIBRARY) | The library the load module goes on. |
| ID(%U) | The ID the job will run under. |
| ACCT(%G) | The account number for the job. |
| NAME(%U) | The name field in the job card. |
| JOB(P1) | The jobname. |
| CLASS(A) | Sysout class for output. |

vi) RAPPREP

The clist submits a batch job which preprocesses the RAPPORT/FORTRAN source. A new DIL file is prepared.

The positional parameters are:-

| | |
|---|---|
| PROG | The source program. |
| DBNAME | The database name. |

Keyword parameters follow, default values being given in brackets

| | |
|---|---|
| ID(%U) | The ID job will run under. |
| ACCT(%G) | The account number job will run under. |
| NAME(%U) | The name field in the job card. |
| JOB(P1) | The jobname. |
| CLASS(A) | Sysout class for output. |

## 12. UTILITY PROGRAMS

The following programs of general applicability to RAPPRORT databases have been developed:-

i) general housekeeping routine
ii) dump a RAPPORT database
iii) restore a RAPPORT database
iv) unload the COMSEQ file
v) reload the COMSEQ file
vi) report generation routines
vii) a program for editing CSLISTS

for further information on the above contact G.D. Firth. Full documentation for the above will be made available if demand warrants it.

## 13. USING QUERY

The following is an example of a clist to run the interactive QUERY program. Obviously the channel numbers and file names are those selected by the user when establishing the database.

```
FREEALL
FREE         FILE(FT03F001,SYSIN,SYSPRINT,SYSHELP)
ALLOC        DA('XBD.GU.RAPPORT.DIL.DATA') FILE(FT09F001) OLD
ALLOC        DA('XBD.GU.RAPPORT.SORT.DATA') FILE(FT12F001) OLD
ALLOC        DA('XBD.GU.RAPPORT.GRANT.DATA') FILE(FT14F001) OLD
ALLOC        DA('XBD.GU.RAPPORT.REST.DATA') FILE(FT15F001) OLD
FREE         ATTR(VDU)
ATTR         VDU LRECL(80) RECFM(F B A) BLKSIZE(81)
ALLOC        DA(*) FILE(FT04F001) USING (VDU)
ALLOC        DA(*) FILE(FT06F001) USING (VDU)
ALLOC        DA(*) FILE(FT03F001)
CALL         'SYR.LOAD(RDIQRY)'
END
```

When the above clist is executed the user will be asked whether backup if required. Reply Y for yes, N for no. Note that if the BAKOUT command is to be used then backup is required and the appropriate file must be allocated in the clist. After this a RAPPORT logo will be displayed and a prompt to enter a QUERY command will appear. The QUERY commands are detailed in the 'RAPPORT - interactive QUERY language user manual'.

## 14. THE USER SUBROUTINE

This is unavailable at present. Once a suitable way of implementing this facility has been developed the appropriate documentation will be made available.

## 15. CURRENT APPLICATIONS

RAPPORT has been used in the following applications:-

i) GMS - Grant Monitoring System, developed to help administrate use of the NAS and CRAY computers. The database is now operational, using a single-user nucleus. Authors G.D. Firth and E.A. Bailey.

ii) SRSCAT. This system is being developed to catalogue the data produced by the SRS and will use the multi-user nucleus. Authors E. Pantos and F. Rake.

iii) Personnel records - under development by G.D. Firth.

## 16. COMMON DATA SETS

The following datasets are available

DNPL.RAPPORT.MESSAG - contains the messages to be loaded when the QUERY files are set up

RA.LINK.DATA - RAPPORT link edit control statements