technical memorandum Daresbury Laboratory

DL/CSE/TM15
(Revised)

GRANT MONITORING SYSTEM - MANAGER'S GUIDE

by

G.D. FIRTH and E.M. BAILEY, Daresbury Laboratory.

NOVEMBER, 1983

Science & Engineering Research Council

Daresbury Laboratory

List of Contents

1. INTRODUCTION

GMS is a computerised database system containing details of grants awarded for use of the NAS 7000 and CRAY-1S computer at Daresbury Laboratory. Details of all computer users and the resources they use are maintained.

GMS uses the RAPPORT database management system on the Daresbury Laboratory NAS 7000 computer. The RAPPORT software was obtained from LOGICA Limited, however certain machine dependent routines have been written by the Daresbury Systems Group.

RAPPORT provides facilities for maintaining and interrogating a relational database. The data stored in the GMS database may be added to, deleted, modified, used to generate reports and for 'one off' type enquiries. These last are catered for by the RAPPORT QUERY language which allows interactive access to the database. Subsequent sections of this report describe the elements and structure of the database and the command procedures which may be used to access it both in batch mode and interactively. The information in the database is updated on a monthly basis, further reports etc being generated as required.

Further details on RAPPORT are available from the following manuals (issued by LOGICA, obtainable from the User Interface Group)

RAPPORT - Fortran user Manual
RAPPORT - Designing and using a database
RAPPORT - Interactive QUERY language user manual.

Requests to draw on information held in the database should be addressed to the GMS database manager.

2. DISCLAIMER

The system in its present state is in no way seen as a final solution to the accounting problem on Daresbury's computers, rather it is seen as a framework to which further facilities can be added.

3. THE AWARD AND ADMINISTRATION OF GRANTS

The SERC maintains powerful computing facilities for use on approved projects of an experimental or theoretical nature. All use of these computing facilities must be authorised in an appropriate manner. Authorisation may result from:

a) An award of computer time as part of a research grant;

b) An award of "pump-priming" facilities for conducting an appraisal or feasibility study with the intention of sub-writing a research grant application;

c) Approval of a research programme, coordinated by an SERC laboratory (as opposed to an outside organisation as in (a)) for which an annual computing allocation is made;

d) Approval of an application by other body for access to SERC facilities upon repayment.

After authorisation, users must register at the appropriate establishment, for each facility to which they have been authorised access. A person wishing to apply for authorisation to use SERC computing facilities will first of all need to fill in form AL54. This has to be signed by a representative from the applicant's local computer centre to verify that the facilities sought cannot be provided locally or regionally. The form will then be passed on to the appropriate SERC laboratory, either by the applicant himself or by the local SERC representative. Where approval by an SERC committee is required (e.g. a research grant submission), the form will be suitably endorsed by the facility management and returned to the applicant who should sent it to SERC central office with a completed grant application (form RG2) or other appropriate form. The form will then be presented to the appropriate committee which will decide whether any award should be made or not. The awarding committees are mainly grouped into four boards, as can be seen from the following diagram:-

Science & Engineering Research Council Grant Awarding Committees

SERC Committees: i) Co-operative Grants Committee
ii) Joint SERC/SSRC
iii) Energy

(1)	(2)	(3)
Astronomy, Space & Radio Board	Engineering Board	Nuclear Physics Board
Committees:-	Committees:-	Committees:-
i) Astronomy I	i) Information Engineering	i) Nuclear Structure
ii) Astronomy II	ii) Engineering Processes	ii) Particle Physics
iii) Solar System	iii) Machines and Power	
	iv) Materials	
	v) Environment	
	(4)	
	Science Board	
	Committees:-	
	i) Biological Science	
	ii) Chemistry	
	iii) Laser Facility	
	iv) Mathematics	
	v) Neutron Beam	
	vi) Physics	
	vii) Science Based Archeology	
	viii) Synchrotron Radiation	

When an award is made, the applicant will be sent an announcement letter, a copy of which will also be sent to the appropriate SERC laboratories. Details of the grant are not entered into the database until a copy of the announcement letter has been received. Otherwise problems can occur (if the committee award a different amount of computing time than was requested for example). The announcement letter is the only document which contains the final details of the award.

Where approval by an SERC committee is not required for an application (e.g. a request for "pump-priming" time, an application to the NERC etc.) then notification of the award made is usually in the form of a letter from the appropriate person.

4. DATABASE STRUCTURE

The database consists of 8 RAPPORT files, as follows:-

- NAME - User details
- ACCOUNT - ID/account combinations
- GRANT - NAS grant details
- CGRANT - CRAY grant details
- USAGE - NAS grant details
- CUSAGE - CRAY grant details
- POST - details of people who receive monthly statements on NAS grants
- CPOST - details of people who receive monthly statements on CRAY grants

Details of the contents of each file are given in the following tables:-

Note that:-

a) PKEY indicates the field is a prime key (i.e. uniquely identifies an entry in the file)

b) the figures in brackets under the NOTES column refer to expanded comments which follow the table

c) the lengths of the fields are in words

d) type is either CHAR for character or BIN for binary

e) the field column refers to the field name within the DDF file

FILE NAME: NAME

RECORDS 2101 CHANNEL 16

FIELD	QUERY NAME	TYPE	LENGTH/ REPETITION	NOTES
LUID	ID	CHAR	1	PKEY (1)
LUINIT	INITIALS	CHAR	2	(2)
LUNAME	SURNAME	CHAR	5	INDEX (3)
LULOC	PLACE	CHAR	5	(4) Users location

FILE NAME: ACCOUNT

RECORDS 2801 CHANNEL 15

FIELD	QUERY NAME	TYPE	LENGTH/ REPETITION	NOTES
LTID	ID	CHAR	1	PKEY (1)
LTACCT	ACCT	BIN	1	PKEY, INDEX (5) Users Account Number

FILE NAME: GRANT

RECORDS 1001 CHANNEL 17

FIELD	QUERY NAME	TYPE	LENGTH/ REPETITION	NOTES
LGBORD	BOARD	CHAR	1	(6) issuing board
LGCTEE	COMM	CHAR	1	(7) Committee, INDEX
LGRANT	NUMBER	CHAR	3	PKEY (8)
LGDACT	ACCT	BIN	1	INDEX (5) Account number for Grant
LGSAD	START	BIN	1	(9) grant start date
LGSTDT	STOP	BIN	1	(10) grant stop date
LGTUNS	TOTAL	BIN	1	(11) Total award
LGDUNS	TIME	BIN	1	(12) Award at DL
LGHOLD	SURNAME	CHAR	5	(13) grant holder, INDEX
LGHLOC	PLACE	CHAR	5	(14) Holders location
LGCOH1	SURNAME2	CHAR	5	(15) grant co-holder
LGCOH2	SURNAME3	CHAR	5	(15) grant co-holder
LGCONT	CONTACT	CHAR	5	(16) Contact name
LGCLOC	LOCATION	CHAR	5	(17) Contact location
LGCODE	CODE	CHAR	1	(18) Code
LGSUM	USE	BIN	1	(19) CPU time Total
LGSEQ	SEQUENCE	BIN	1	PKEY (20) Sequence number
LGSPAR	-	BIN	7	Unused

FILE NAME: CGRANT

RECORDS 601 CHANNEL 17

FIELD	QUERY NAME	TYPE	LENGTH/ REPETITION	NOTES
LRBORD	BOARD	CHAR	1	(6) issuing board
LRCTEE	COMM	CHAR	1	(7) INDEX, committee
LRRANT	NUMBER	CHAR	3	PKEY (8)
LRDACT	ACCT	BIN	1	INDEX (5) account number
LRSAD	START	BIN	1	(9) Grant start date
LRSTDT	STOP	BIN	1	(10) grant stop date
LRTUNS	TOTAL	BIN	1	(11) Unused
LRDUNS	TIME	BIN	1	(12) award at DL
LRHOLD	SURNAME	CHAR	5	INDEX (13) grant holder
LRHLOC	PLACE	CHAR	5	(14) Holders location
LRCOH1	SURNAME2	CHAR	5	(15) grant co-holder
LRCOH2	SURNAME3	CHAR	5	(15) grant co-holder
LRCONT	CONTACT	CHAR	5	(16) Contact name
LRCLOC	LOCATION	CHAR	5	(17) Contact location
LRCODE	CODE	CHAR	1	(18) Code
LRSUM	USE	BIN	1	(19) CPU time total
LRSEQ	SEQUENCE	BIN	1	PKEY (20) Sequence number
LRSPAR	-	BIN	7	Unused

FILE NAME: USAGE

RECORDS 2301 CHANNEL 18

FIELD	QUERY NAME	TYPE	LENGTH/ REPETITION	NOTES
LAID	ID	CHAR	1	PKEY (1)
LAACCT	ACCT	BIN	1	PKEY (5)
LATSOS	TSOS	BIN	12	(21) TSO Sessions
LATSOE	TSOE	BIN	12	(22) TSO elapsed time
LACPUT	CPU	BIN	12	(23) CPU time
LAJOBS	JOBS	BIN	12	(24) number of batch jobs

FILE NAME: CUSAGE

RECORDS 1301 CHANNEL 18

FIELD	QUERY NAME	TYPE	LENGTH/ REPETITION	NOTES
LCID	ID	CHAR	1	PKEY (1)
LCACCT	ACCT	BIN	1	PKEY (5)
LCTSOS	TSOS	BIN	12	(21) Not applicable
LCTSOE	TSOE	BIN	12	(22) Not applicable
LCCPUT	CPU	BIN	12	(23) CPU time
LCJOBS	JOBS	BIN	12	(24) Number of batch jobs

FILE NAME: POST

RECORDS 1001 CHANNEL 18

FIELD	QUERY NAME	TYPE	LENGTH/ REPETITION	NOTES
LPRANT	NUMBER	CHAR	3	PKEY (8)
LPNUM	SEQUENCE	BIN	1	PKEY (25)
LPNAME	SURNAME	CHAR	5	(26) Person requiring information
LPLOC	LOCATION	CHAR	5	(27) Person's location

FILE NAME: CPOST

RECORDS 701 CHANNEL 18

FIELD	QUERY NAME	TYPE	LENGTH/ REPETITION	NOTES
LORANT	NUMBER	CHAR	3	PKEY (8)
LONUM	SEQUENCE	BIN	1	PKEY (25)
LONAME	SURNAME	CHAR	5	(26) Person requiring information
LOLOC	LOCATION	CHAR	5	(27) Person's location

NOTES

(1) ID

This is the ID which is used for identifying TSO sessions and batch jobs. It may be up to 4 characters long e.g. XBD, EM, USE, OPER

(2) Initials

A user's initials (and possibly title) e.g. GD, EM, DE PROF. Up to 8 characters may be used.

(3) Surname

This is the user's surname, e.g. FIRTH, BAILEY, SMITH and may be up to 20 characters long.

(4) Location

This is the site at which the user is based e.g. DARESBURY for Daresbury Laboratory, BIRKBECK for Birkbeck College London. See Appendix A for list of sites. Up to 20 characters may be used.

(5) Account

The account number a user uses for TSO sessions and running batch jobs. Up to 4 digits e.g. 30, 4320, 410.

(6) Board

This is the board responsible for issuing the grant e.g. SB for Science Board. See Appendix B for a full list of abbreviations. Up to 4 characters long.

(7) Committee

This is the committee responsible for issuing the grant e.g. BS for Biological Sciences. See Appendix B for a full list of abbreviations. Up to 4 characters long.

(8) Number

This is the grant reference number e.g. GR/A/12345. It may be up to 12 characters long.

(9) Start Date

The date from which a grant is to run in YYMM format e.g. 8004 represents April 1980. NB: the grant will run from the first day of the month. The date must consist of 4 digits, the first 2 representing the year, the second 2 representing the month.

(10) Stop Date

The date on which the grant is to terminate in YYMM format e.g. 8303 represents March 1983. NB: the grant runs until the last day of the month. The date must consist of 4 digits, the first 2 representing the year, the second 2 representing the month.

(11) Total Units

This is the total number of CPU hours for which the grant is awarded. Note that some of this grant may be used on computers other than those at Daresbury Laboratory, e.g. at the Rutherford Laboratory. For the IBM or NAS this figure is given in 195 hours, where one 195 hour is equivalent 2.0 AS/7000 hours.

(12) Daresbury Units

The number of 360/195 equivalent CPU hours to be used on the NAS at Daresbury, or the number of CRAY CPU hours.

(13) Grant Holder

The surname of the person to whom the grant is issued, which need not be the person actually doing the work, e.g. SMITH. Up to 20 characters long.

(14) Place

The site at which the grant holder is situated, e.g. BIRKBECK, LIVERPOOL. See Appendix A for a list of sites. Up to 20 characters long.

(15) Co-holder

Some grants are held by more than 1 person, so co-holder names must be stored, e.g JONES. Up to 20 characters long.

(16) Contact Name

The person actually doing the work e.g. BROWN. Up to 20 characters long.

(17) Contact Location

The site of the person actually doing the work e.g. LIVERPOOL. See Appendix A for a list of sites. Up to 20 characters long.

(18) Code

This will be blank in the majority of cases but can take the following values:

E for expired grants

D gor discretionary grants

C for grants with an allocation of CRAY time but no NAS time. Anyone with a grant of this type may need access to the NAS in order to submit jobs to the CRAY, so an entry is put in the GRANT file with both the 'total units' and the 'Daresbury units' fields set to zero and the code set to C. An entry is made in the CGRANT file in the normal way and the CODE field left blank.

(19) Use

This is the total amount of CPU time used against the grant to date, stored in 1/100^{ths} of a second.

(20) Sequence number

Most grants have a unique account number associated with them and in this case the sequence number is set to 1. A few of the 'in house' projects have several account numbers under one allocation of time in which case there is one entry for each account number with the same grant number but a different sequence number for each entry. The first entry will have the 'Total units' and 'Daresbury units' fields filled in with the appropriate allocations of time and will have a sequence number of 1. Each successive entry will have both 'total units' and 'Daresbury units' set to zero and will have a sequence number one greater than the previous entry for the grant.

(21) TSO Sessions

There is 1 entry for each of the last 12 months, each entry containing the number of TSO sessions completed under this ID/account combination during the month. NB: these are calendar months.

(22) TSO elapsed time

There is 1 entry for each of the last 12 months, each entry containing the total time the ID/account combination was logged on to TSO during the month in 1/100^{ths} of a second. The months are calendar months.

(23) CPU time

There is 1 entry for each of the last 12 months, each entry containing the total CPU time used by the ID/account combination during the month. The figure is in 1/100^{ths} of a second and includes time for batch jobs and TSO sessions. The months are calendar months.

(24) Jobs

There is 1 entry for each of the last 12 months, each entry containing the number of batch jobs submitted by the ID/account combination during the month. The months are calendar months.

(25) Sequence

This is an integer greater than or equal to 1 which, together with the grant number, uniquely describes the entry. For example if two people both wanted information on grant GR/B/1111 then there would be two entries in the file, one with the grant number set to GR/B/1111, the 'name' and 'location' fields set to the appropriate name and location and the 'sequence' field set to 1, and one with the fields as described above except for the 'sequence' field which would be set to 2. If another request for information on this grant was received then another entry would be made with the 'sequence' field set to 3 and so on.

(26) Surname

The surname of the person requiring information on a given grant. Up to 20 characters long.

(27) Location

The site at which the person requiring information is based. Up to 20 characters long. See Appendix A for a list of sites.

5. CLISTS TO MAINTAIN THE DATABASE

Access to the database is by means of the clists listed below. Each clist is described in detail in the following chapters.

GUS.CLIST(MONTHLY) : To submit the jobs which update the database files.

GUS.CLIST(REPORT) : To submit a batch job which produces reports from the database.

GUS.CLIST(DUMP) : Submits batch job(s) to backup specified RAPPORT files.

GUS.CLIST(RELOAD) : To submit batch job(s) which restore all or part of the database from backup.

GUS.CLIST(GUQUERY) : To enable an interactive QUERY session to be established.

GUS.CLIST(GEDIT) : To edit the NAS GRANT and POST files.

GUS.CLIST(CGEDIT) : To edit the CRAY GRANT and POST files.

GUS.CLIST(GUMAKE) : To make the NAS files used by the database.

GUS.CLIST(GUEMPTY) : To EMPTY RAPPORT files before use.

GUS.CLIST(GUQPREP) : To prepare the QUERY questions and messages files for use.

GUS.CLIST(RAPPCL) : To preprocess, compile and link programs.

GUS.CLIST(CRAYREP) : To submit a batch job which generates a report on CRAY grant usage for use by boards and committees.

GUS.CLIST(NASREP) : To submit a batch job which generates a report on NAS grant usage for use by boards and committees.

GUS.CLIST(CRAYACC) : To submit a batch job which generates statements of CRAY usage to send to users.

GUS.CLIST(NASACC) : To submit a batch job which generates statements of NAS usage to send to users.

6. MONTHLY UPDATE OF THE DATABASE GUS.CLIST(MONTHLY)

GUS.CLIST(MONTHLY) contains a clist to provide for update of the database with more recent information. The clist submits 9 batch jobs which have the following steps:
(Note that CLEAR is a RAPPORT command to delete all records in a file)

Job 1 has 2 steps:

a) To extract information for the NAME and ACCOUNT files from the USERREG file.
b) Clear and load the NAME and ACCOUNT files.

Job 2 has 6 steps:

a) To extract information for batch jobs and TSO sessions from the NAS SMF tapes.
b) To sort the information from a) to ID/account order.
c) To accumulate the NAS usage for each ID/account combination.
d) Update the NAS USAGE file.
e) Accumulate the NAS usage under the correct ID/account combinations.
f) Update the usage totals in the NAS GRANT file.

Job 3 has 4 steps:

a) Set all CRAY usage to 0 for the new month.
b) Print a report on the current database.
c) Dump the whole database.
d) Copy the dump to a TSO pack from where it can be archived.

Job 4 has 2 steps:

a) Dump the POST file.
b) Dump the CPOST file.

Job 5 has 4 steps:

a) Dump the NAME file.
b) Dump the ACCOUNT file.
c) Dump the GRANT file.
d) Dump the CGRANT file.

Job 6 has 5 steps:

a) Dump the CUSAGE file.
b) Dump the USAGE file.
c) Dump the DDF file.
d) Copy the individual file dumps to a TSO pack.
e) Write the number of the month being updated into a dataset.

Jobs 7, 8 and 9 all produce different reports which should be sent to the people who's addresses are given in comments in the JCL.

The jobs are run once a month, as soon after the end of the month as possible.

The NAME and ACCOUNT files are cleared and reloaded in their entirety every month. However the USAGE and CUSAGE files are treated in a different manner. Usage is stored for the last 12 months on a 'rolling' basis. Usage figures for the oldest month in the database are cleared to zeros and then overwritten with the latest month's figures.

The clist has the following keyword parameters, default parameters being shown in brackets.

NAS1(*) }	The volume serial numbers of the 2 SMF tapes containing
NAS2(*)	the month's NAS usage figures.
YEAR(*)	The year used in defining the period for which information is to be extracted from the SMF tape.
MONTH(*)	The month for which usage is to be updated (also used to define period for which information is to be extracted from the SMF tape).
PRINT(01)	Set to 01 if reports of the extract from the USERREG file and the accumulations of usage under ID/account combinations are required. Set to 00 to switch off these reports.
JOB(M)	Appended to the parameter ID and suffixed by a number from 1 to 9 this gives a jobname to the job submitted. Note that the jobs submitted are numbered from 1 to 9, corresponding to the numbered descriptions above. Each job releases the subsequent job.
ID(GUS)	Used in running the jobs.
A(0030)	The account number the jobs are to run under.
N(GUS)	Parameter to specify the name field on the job card.
CLASS(A)	The sysout class for output.

7. GENERATING REPORTS FROM THE DATABASE: GUS.CLIST(REPORT)

GUS.CLIST(REPORT) contains a clist to provide for the generation of reports from the database. The particular report(s) to be generated is/are specified in a control file which is a member of PDS GUS.DATA. A single step batch job is submitted.

Details of the control file:

Each control card either generates a report or acts as supplemental information to the previous control card, e.g. in some cases a second card is used to provide a report heading. The keywords listed below each indicate a specific report to be generated. The keyword must start in the first column of the card. Only the first four characters need to be specified, although additional characters may add clarity if the file is to be kept for further use.

USER - report from NAME and ACCOUNT files
GRANT - report on NAS grants
CGRANT - report on CRAY grants
USAGE - report on NAS usage
CUSAGE - report on CRAY usage
TUSAGE - totals report on NAS usage (see below)
TCUSAGE - totals report on CRAY usage (see below)
STOP - optional last card to indicate end of control file

Note that when TUSAGE or TCUSAGE is specified a second card containing a group heading is also expected.

After the keyword has been specified subsequent fields are all numeric and must be separated by one or more blanks.

With the exception of the TUSAGE and TCUSAGE cards (see below) the second field on the card specifies the order in which the report is to be printed, the default being order 1. The following options are available.

Keyword	Value of Parameter	Sort order
USER	1	ID order
	2	Account order
	3	Location order
	4	Name order
GRANT	1	Account order
	2	Board/Holder order
CGRANT	1	Account order
	2	Board/Holder order
USAGE	1	Account order
	2	ID/account order
CUSAGE	1	Account order
	2	ID/account order

For USAGE/CUSAGE reports the following further options are available.

field 3 : The first month's details to go into the report
field 4 : The last month's details to go into the report
* field 5 : Minimum CPU time for details to be reported
* field 6 : Maximum CPU time for details to be reported
* field 7 : Minimum TSO elapsed time for details to go on report
* field 8 : Maximum TSO elapsed time for details to go on report
field 9 : If set to 1 only totals for each ID/account combination will be printed. The default, 0, gives full details

The fields marked * must be entered in 1/100^{ths} of a second.

Note that the default wil give the full range of months, CPU times and TSO elapsed times.

All fields up to and including the last one which is not to take the default value must be specified. If one of the fields which is to be specified is to take the default value then it must be given a value of zero (except the sort field) e.g. to obtain a USAGE print in order 1 giving totals only the card would be of the form

USAGE 1 0 0 0 0 0 0 1

When the TUSAGE or TCUSAGE keyword is specified the following parameters are required:

field 2 : the first account number of the range for which usage is to be totalled
field 3 : the last account number of the range for which usage is to be totalled.

TUSAGE or TCUSAGE card should be followed by a card containing a heading to go with the accumulated totals e.g. DARESBURY INTERNAL PROJECTS TOTALS. The heading may be up to 40 characters in length.

The clist has the following keyword parameters, default values being shown in brackets.

JOB(PRINT) : Appended to the ID parameter this gives a jobname to the job submitted.
A(0030) : The account number the job is to run under.
ID(GUS) : Prefixed to the JOB parameter this gives a jobname to the job submitted.
T(0) : The number of minutes CPU time the job is to be given. The seconds field is set to 59.
DC(PRINT1) : The member of GUS.DATA containing the control cards.

8. DUMPING THE DATABASE GUS.CLIST(DUMP)

GUS.CLIST(DUMP) contains a clist which submits batch jobs to dump specified database files to different members of one partitioned dataset (which may then be copied and archived to tape) or to dump the full database to a direct access file which may be copied to a sequential file and then archived to tape if required.

It is envisaged that the facility to dump individual Rapport files will be of use if it becomes necessary to enlarge or alter a Rapport file or an NAS file. The file may be dumped, the physical structure of the database altered, and the database returned to its previous state by re-loading the file (see section 9 reloading the database). The facility to dump the full dtabase is necessary to enable the database to be restored to a consistent state, should for instance, the machine crash while the database is being updated. The facility to archive copies of the database will enable a historical record of each stage of the development of the database to be maintained.

All parameters are keyword, default values being given in brackets (where applicable).

ALL : dump all the individual failes and the whole database.
DDF : dump the DDF file (used to define the database structure)
NAME : dump the NAME file
ACCOUNT : dump the ACCOUNT file
POST : dump the POST file
CPOST : dump the CPOST file

GRANT : dump the GRANT file
CGRANT : dump the CGRANT file
USAGE : dump the USAGE file
CUSAGE : dump the CUSAGE file
DB : dump the whole database
ARCH : copy the dump to a new file from where it can be archived
MONTH(*) : to be used in the name of the archive file
YEAR(*) : to be used in the name of the archive file
ID(GUS) : used to identify files and in the jobname of submitted jobs
N(GUS) : the name field of the job card
A(0030) : the account number the job is to run under
CLASS(A) : sysout class for output
JOB(DUMP) : used to form the jobname

It is perfectly possible to specify more than one of the keyword by name parameters so that several jobs may be submitted by a single execution of the command.

When an individual file is dumped it is stored in a partitioned dataset called ID.GU.RAPPORT.DUMP.DATA with a member name the same as the file name, e.g. if the NAME file is dumped, the dump is stored in ID.GU.RAPPORT.DUMP.DATA(NAME). When the whole database is dumped it is stored in the dataset ID.GU.RAPPORT.DUMPDB.DATA.

If one or more individual file has been dumped and the ARCH keyword was specified then ID.GU.RAPPORT.DUMP.DATA is copied to a TSO dataset called ID.GU.RAPPORT.DUMP.Dmonthyear. Once the job has run the dataset should be arhived using the TSO command
ARCHIVE 'ID.GU.RAPPORT.DUMP.Dmonthyear' REPLACE. If the whole database was dumped and the ARCH keyword was specified then the dataset ID.GU.RAPPORT.DUMPDB.DATA will be copied to a TSO dataset called ID.GU.RAPPORT.DUMPDB.Dmonthyear. Once the job has run the dataset should be archived using the TSO command
ARCHIVE 'ID.GU.RAPPORT.DUMPDB.Dmonthyear' REPLACE.

N.B. When the ARCH keyword is specified the MONTH and YEAR parameters must also be specified. The MONTH and YEAR parameters should correspond to the date of the latest information in the database at the time of the

dump, e.g. if the database is up to date to the end of August 1981 then the parameters will be MONTH(08) YEAR(81) and the datasets produced ready to be archived will be:

(1) ID.GU.RAPPORT.DUMP.D0881
(2) ID.GU.RAPPORT.DUMPDB.D0881

If the ARCH and DB parameters are specified but no individual file names then only the 2nd dataset will be produced. If the ARCH parameter and one or more file names are specified but the DB parameter is left out then only the 1st dataset will be produced. If the ARCH and DB parameters and also one or more file names are specified (or the ALL keyword is used) then both datasets will be produced.

9. RESTORING AND RELOADING THE DATABASE GUS.CLIST(RELOAD)

GUS.CLIST(RELOAD) contains a clist which submits batch jobs to reload specified database files from a partitioned dataset or to restore the full database from a direct access file. Both the partitioned dataset and the direct access file may be restored from the TSO archive system if required.

It is envisaged that the facility to reload individual RAPPORT files will be of use if it becomes necessary to enlarge or alter a RAPPORT file or an NAS file. The file may be dumped, the physical structure of the database altered and the database returned to its previous state by re-loading the file. The facility to restore the full database will enable recovery to a consistent state should a database transaction be interrupted. Archived copies of the whole database and of individual files may be restored from tape to permit access to older versions of the database.

All parameters are keyword, default values being given in brackets (where applicable).

NAME : load the NAME file
ACCOUNT : load the ACCOUNT file

POST : load the POST file
CPOST : load the CPOST file
GRANT : load the GRANT file
CGRANT : load the CGRANT file
USAGE : load the USAGE file
CUSAGE : load the CUSAGE file
DDF : load the DDF file (used to define the structure of the databases)
DB : load the full database
REST : load the full database and/or one or more individual files from a dataset which has been restored from archive
MONTH(*) : defines which dump has been restored from archive
YEAR(*) : defines which dump has been restored from archive
ID(GUS) : used to identify files and in the jobname of the submitted jobs
N(GUS) : the name field of the job card
A(0030) : the account number jobs are to run under
CLASS(A) : the sysout class for output
JOB(LOAD) : used to form the jobname of submitted jobs

It is acceptable to specify more than one of the keywords by name parameters so that several jobs may be submitted by a single exeution of the command.

If one or more individual file is to be loaded from an archived version then it is first necessary to restore the dump from tape using the TSO command

RESTORE 'ID.GU.RAPPORT.DUMP.Dmonthyear'

Once the dump has been restored the RELOAD clist can be executed and, provided the REST YEAR and MONTH parameters are specified as well as the file names the restored dataset will be copied to ID.GU.RAPPORT.DUMP.DATA from where the specified file(s) will be re-loaded to the database. The restored dataset, ID.GU.RAPPORT.DUMP.Dmonthyear, should then be deleted (a copy will remain in archive).

If the whole database is to be loaded from an archived version then it is first necessary to restore the dump from tape using the TSO command

RESTORE 'ID.GU.RAPPORT.DUMPDB.Dmonthyear'

Once the dump has been restored the RELOAD clist can be executed and, provided the REST, YEAR and MONTH parameters are specified as well as the DB parameter, the restored dataset ID.GU.RAPPORT.DUMPDB.Dmonthyear will be copies to ID.GU.RAPPORT.DUMPDB.DATA from where the database will be loaded. The restored dataset, ID.GU.RAPPORT.DUMPDB.Dmonthyear, should then be deleted (a copy will remain in archive).

N.B. When the REST keyword is specified the MONTH and YEAR parameters must also be specified. The MONTH and YEAR parameters should correspond to the the data of the latest information in the database to be restored, e.g. if the database is up to date to the end of September 1981 but it is necessary to restore a version which is only updated to the end of July 1981, the MONTH and YEAR parameters will be MONTH(07) YEAR(81) and the datasets to restore from archive will be

(1) ID.GU.RAPPORT.DUMP.D0781
(2) ID.GU.RAPPORT.DUMPDB.D0781

If it is required to load the full database then it is only necessary to restore the second of these files. If it is required to load one or more individual files and not the full database then it is only necessary to restore the first file.

If it is required to load both the full database and one or more individual files (although this is unlikely) then both the above files will need to be restored.

10. ACCESS TO THE DATABASE USING QUERY GUS.CLIST(GUQUERY)

GUS.CLIST(GUQUERY) contains a clist which enables an interactive QUERY session to be set up. The clist allocates the GMS RAPPORT files to the terminal session and then calls the QUERY Program. Refer to the Grant Monitoring system - user guide (Daresbury Technical Memorandum DL/CSE/TM19) for full details of the command sequence lists which are available for interrogating the database. If these CSLISTS are not sufficient to deal with a posed problem then the QUERY commands detailed

in LOGICAS 'Interactive Query Language user manual' may be used.

Note that a 650K TSO region must be used to run the interactive QUERY language.

There are no parameters to this clist.

11. UPDATING THE NAS GRANT AND POST FILES GUS.CLIST(GEDIT)

GUS.CLIST(GEDIT) contains a clist which runs an interactive program to update the GRANT and POST files of the database. The clist has no parameters. At the start of an edit session the standard RAPPORT information about files, blocks, etc. is displayed. It should be checked for the presence of an error code. If an error occurs, the GMS database manager should be consulted. After the RAPPORT messages the editor produces the message

'GU810 GMS GRANT AND POST EDITOR DATE'

followed by a prompt for input. Throughout the edit session an asterisk '*' is used as a prompt for input. It should be noted that the @Y command is trapped by the editor and can be used to interupt and terminate a command.

The following commands are available:

a) GRANT
b) POST
c) SHOW
d) STORE
e) HELP
f) STOP

Use of the commands is detailed in the following sections. For all commands, the keyword used must be entered, starting in column one of the screen. Any abbreviations for a command is permitted provided the first four characters of the commands are unaltered. Only the GRANT and POST commands have subcommands. If an invalid command is entered, appropriate help information is displayed and the command may then be re-entered.

a) The GRANT command

There are two forms of the grant command, one to create a new grant entry and one to edit an existing grant.

i) Creating a new grant record

The format of the command is:

GRANT GRANT-NUMBER NEW

e.g. GRANT GR/A/1111 NEW

If one or more entries for the grant already exist then the 'sequence' field in the new entry is set to the next highest value and a message will be displayed informing the user of this. For example, suppose there are already three entries for grant GR/B/1234 and the user types in GRANT GR/A/1234 NEW, then the 'sequence' field of the new entry will be set to 4 and a message will be displayed to this effect.

If there are no existing entries for the grant number given then the 'sequence' field is set to 1 for the new entry.

After the GRANT command has been entered, the user will be prompted for each of the fields in the grant in turn. For example the first prompt is

'BOARD A4'

The user should respond by typing in a board abbreviation (a table of permitted boards and committees is given in appendix B of the Guide). For character fields, a maximum size is indicated in the prompt, for numeric fields any number up to the maximum size integer which may be stored in a word is permissible. All fields entered should start in column 1 of the screen.

Please note: in most cases there will be one account number uniquely associated with a grant and in this case there will only be one entry in the GRANT file and this will have the 'sequence' field set to 1. However, for a few of the 'in house' projects there are several account numbers associated with one allocation of CPU time. In these cases there will be more than one entry for the given grant, in which case the 'total units' and 'daresbury units' fields should be filled in only for the entry with sequence number one. In all other entries for the grant these fields should be set to 0.

ii) Editing an existing grant record

The format of the command is: GRANT GRANT-NUMBER e.g. GRANT GR/A/1111

If more than one entry exists for the given grant, the user will be asked which sequence number he wants.

The user will now be prompted for a subcommand.

With the exception of SHOW, HELP and END the subcommands are used to change the values of specified fields. The format of the subcommand is

FIELDNAME VALUE

where 'field name' is taken from the list below and 'value' is the new value to be assigned to the field

BOARD - the board issuing the grant
COMMITTEE - the committee issuing the grant
ACCT - the account number associated with the grant
START - the start date of the grant (YYMM format)
STOP - the stop date of the grant (YYMM format)
TOTAL - the total award of the grant
TIME - the Daresbury award
NAME - the grant holder
PLACE - the grant holders location
NAM2, NAM3 - grant co-holders
CONTACT - the person working on the grant if not the holder
LOCATION - the contact location
CODE - the code associated with the grant
SUM - the total CPU usage of the grant

e.g. to set the grant holders name to SMITH enter

NAME SMITH

To set a field to blank enter an @ symbol. The other subcommands available are:

SHOW - this takes the form SHOW GRANT/POST and will display details of either the current GRANT record or the current POST record depending on which keyword is specified

HELP - to obtain help information
END - to terminate the edit of this grant record

Please note: the GRANT command does not update information in the database, it only alters values in a buffer. To update the database from the buffer it is necessary to use the STORE command.

b) The POST command

This works in exactly the same way as the GRANT command and the format is

POST GRANT-NUMBER (NEW)

where NEW is an optional keyword used to specify whether a new entry is to be made or an existing entry is to be updated e.g.

POST GR/C/1234 NEW

would cause the user to be prompted for each of the fields in the POST record and a new entry would be created which could be put into the database using the STORE command

POST GR/C/1234

would enable the user to edit the existing entry for grant GR/C/1234.

c) The SHOW command

The show command is used to display all of the details in either a GRANT record or a POST record. The format of the command is:

SHOW POST/GRANT

e.g. SHOW POST would give details of the POST reord currently held in the program buffer

SHOW GRANT would give details of the current GRANT record.

Before any given grant or post record can be displayed it must first have been retrieved or created by means of the GRANT or POST command.

d) The STORE command

The STORE command is used to update the database. The format of the command is

STORE POST/GRANT

e.g. STORE POST will cause the POST record currently stored in the program buffer to be put into the database. If an entry already exists with the same grant number and sequence

number then it will be overwritten. If there is no entry with the same grant number and sequence number then a new entry will be created.

STORE GRANT will store the current GRANT record in the same way.

e) The HELP command

The HELP command lists all the commands available to the user.

f) The STOP command

This terminates the GEDIT program.

12. UPDATING THE CRAY GRANT AND POST FILES GUS.CLIST(CGEDIT)

This is identical to updating the NAS GRANT and POST files except that when a new grant is input there is no prompt for 'total award' as this is always set to 0 for a CRAY grant.

13. MAKING THE NAS FILES GUS.CLIST(GUMAKE)

GUS.CLIST(GUMAKE) contains a clist which EXECS clist DAMAKE (in GUS.CLIST(DAMAKE) - see below) once for each of the NAS files in the GMS system which is to contain RAPPORT files. There are no parameters for GUMAKE, but in each execution of DAMAKE the name and size of the file are specified. The files made by DAMAKE are standard fortran direct access files, each of which may contain one or more RAPPORT files and/or indices.

The main use of GUMAKE is when it is necessary to create all the files within the database from scratch. This should only happen in exceptional circumstances once the system has been implemented (for instance if it were decided to reduce the number of NAS files, to move indices to different channels from their associated files, or to change the database size to cope with a different volume of records). A more likely situation is that it should become necessary to change the size of one NAS file but that the rest should be unchanged. This can be achieved by the direct execution of DAMAKE, but GUMAKE should be altered to reflect the change. This is the second use of GUMAKE, to record the size and names of all the

NAS files within the system.

GUS.CLIST(DAMAKE) contains a clist which submits a batch job to create a fortran direct access file on a specified volume with 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.

There is one positional parameter:

FILE The name of the file to be created.

The remaining parameters are keyword, default values being given in brackets.

SIZE(10)	The number of blocks of data in the file
CLASS(A)	SYSOUT class for output
ID(GUS)	The ID for the job and file created
ACCT(0030)	The account number for the job
DISK(DL0238)	The disk the file is to be created on
UNIT(3330-1)	The unit type of the disk
BYTES(6220)	The number of bytes per block
WORDS(1555)	The number of words per block
NAME(GUS)	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.

14. EMPTYING THE RAPPORT FILES GUS.CLIST(EMPTY)

GUS.CLIST(EMPTY) contains a clist which will empty one or more of the NAS files in the database system which contain RAPPORT files. Every RAPPORT file must be emptied (i.e. set to zeros) before it is used for the first time after being created. When the clist is executed, the message 'RAPPORT-3 EMPTY' will appear on the screen, followed by 'DATABASE COMMON FILE (DCF)' and then 'FORMATTED READ ON CHANNEL 9 OPENED'. The prompt 'ENTER CHANNEL NUMBER, $ALL, $HELP OR $EXIT' will then be displayed. If a channel number is entered, the file on the given channel will be emptied and the program will issue the prompt again. If $ALL is entered, all files will be emptied. $HELP causes information about the number of

blocks on each channel to be displayed on the screen and £EXIT causes the program to terminate with the message 'EMPTY PROCESSING COMPLETE'.

The main reason for using EMPTY is when one of the database files has become too small and has been increased in size. The NAS file has then to be re-created and emptied before it can be used again.

The clist has no parameters.

15. PREPROCESSING THE DATA DEFINITION FILE GUS.CLIST(PREPDDF)

GUS.CLIST(PREPDDF) contains a clist which preprocesses the data definition file (DDF) and forms the DCF file. The DDF contains information about the structure of the database and the preprocessed version (the DCF) is used by the RAPPORT software when accessing the database.

The clist has no parameters.

16. ENLARGING A DATABASE FILE

When a database file becomes 90% full the next attempt to store a new record will cause the error number -7 to appear. The new record will be stored successfully, but the file should now be enlarged before the stage is reached where the file is 100% full. To enlarge the file the following steps are necessary:

1) Backup the database
2) Edit the DDF and increase the number of records for the appropriate database file. Note which channel it is allocated to.
3) Preprocess the DDF
4) List the DCF and at the bottom there will be a table detailing how many blocks are required on each channel. Make a note of the number required for the channel allocated to the file which is being enlarged
5) Re-create the appropriate NAS file using GUS.CLIST(DAMAKE). The value of the size parameter will be the number of blocks described in the DCF

6) Empty the NAS file using GUS.CLIST(EMPTY)
7) Re-load all database files which are stored in the NAS file that has been recreated (even those which have not been enlarged)

17. QUERY PREPARATION GUS.CLIST(GUQPREP)

GUS.CLIST(GUQPREP) contains a clist which interactively loads the RIQLOM, RIQLOS and RIQLOC files of the database which are used by the interactive QUERY language. This program must be run after the NAS file which will contain these RAPPORT files has been made and emptied (see sections 13 and 14). It enables the QUERY language to be used. Once the system has been implemented it will only be necessary to reload these files in exceptional circumstances, for instance if the NAS file on which they reside is enlarged to enable the RIQLOC file (where command sequence lists are stored) to be expanded.

The clist has no parameters.

18. PREPROCESSING, COMPILING AND LINKING GUSC.CLIST(RAPPCL)

GUS.CLIST(RAPPCL) contains a clist which submits a batch job to preprocess compile and link a fortran source containing RAPPORT subroutine calls. For details of the preprocessor refer to the RAPPORT user manual. The compile and link steps use the standard NAS facilities.

The job preprocesses the Fortran Source (i.e. converts RAPPORT statements into Subroutine calls), compiles the preprocessed source program and link edits the compiled program with the RAPPORT library. The load module is stored in GUS.LOAD with a member name the same as the program name which is specified when the clist is executed.

The clist has one positional parameter:

PROG The program to be compiled (up to 5 characters)

The remaining parameters, listed below, are keyword their default values being given in brackets.

ID(GUS)	ID of the job to be submitted
ACCT(0030)	The account number the job is to run under
OUT(A)	Sysout class for output
MIN(0)	
SEC(29)	
JOB(SPCL)	

19. CRAY COMMITTEE REPORTS GUS.CLIST(CRAYREP)

GUS.CLIST(CRAYREP) contains a clist which submits a batch job which uses information stored in the CRAY grant and usage files (CGRANT,CUSAGE) to create a usage report.

There are three steps to the job. The first extracts the information from the database, the second sorts it and the third prints it. The report produced is in board, commitee, holder order, totals being produced for each board and committee.

Usage figures are given by quarter. The parameter LM allows the user to determine which of the months within the database will correspond to the first month of the first quarter.

The clist has the following keyword parameters, defaults being given in brackets.

ID(GUS)	The ID the job is to run under
A(0030)	The account number the job is to run under
JOB(PRINT)	The jobname
T(0)	The number of CPU minutes allocated to each step of the job. The Seconds Field for each step is 59
CLASS(A)	The sysout class for output from the job
LM(01)	The first month of the first quarter on which usage figures are based
N(GUS)	The name field of the job card

20. NAS COMMITTEE REPORTS GUS.CLIST(NASREP)

GUS.CLIST(NASREP) contains a clist which submits a batch job which uses information stored in the NAS grant and usage files (GRANT,USAGE) to create a report for use by boards and committees. There are three steps to the job. The first extracts the information from the database, the second sorts it and the third prints it. The report produced is in holder within committee within board order, totals being produced for each board and committee.

Usage figures are given by quarter. The parameter LM allows the user to determine which of the months within the database will correspond to the first month of the first quarter.

The clist has the following keyword parameters, default values being given in brackets.

ID(GUS)	The ID the job is to run under
A(0030)	The account number the job is to run under
JOB(PRINT)	The jobname
T(0)	The number of CPU minutes allocated to each step of the job. The seconds field for each step is 59
CLASS(A)	The sysout class for output
LM(01)	The first month of the first quarter on which usage figures are based
N(GUS)	The name field of the job card

21. CRAY USAGE STATEMENTS GUS.CLIST(CRAYACC)

GUS.CLIST(CRAYACC) contains a clist which submits a batch job to create detailed reports about the latest months CRAY usage. One or more reports may be produced for each grant depending on how many people have requested information about a given grant. Each individual report contains information about one grant and also the name of the person it is to be posted to.

There are two positional parameters:

MONTH — the number of the month to be reported on. Must be two digits (e.g. 05 represents May)

YEAR — The year to be printed at the top of each report. Must be two digits (e.g. 83).

N.B. This parameter does not affect what information is retrieved as the database only contains 12 months worth of data.

The remaining parameters are keyword, default values being given in brackets.

ID(GUS) — The ID the job is to run under

A(0030) — The account number the job is to run under

JOB(MONTH) — The jobname

CLASS(A) — The sysout class for output

22. NAS USAGE STATEMENTS GUS.CLIST(NASACC)

GUS.CLIST(NASACC) contains a clist which submits a batch job to create detailed reports about the latest months NAS usage. One or more reports may be produced for each grant depending on how many people have requested information about a given grant. Each individual report contains information about one grant and also the name of the person it is to be posted to.

There are two positional parameters:

MONTH — the number of the month to be reported on. Must be two digits (e.g. 05 represents May)

YEAR — The year to be printed at the top of each report. Must be two digits (e.g. 83).

N.B. This parameter does not affect what information is retrieved as the database only contains 12 months worth of data.

The remaining parameters are keyword, default values being given in brackets:

ID(GUS) — The ID the job is to run under

A(0030) — The account number the job is to run under

JOB(MONTH) — The jobname

CLASS(A) — The sysout class for output

23. DATABASE INTEGRITY ASSERTION GUS.CLIST(DIA)

If a program which is accessing the database should terminate abnormally (e.g. run out of time etc.) then the closedown of the database will not be done correctly and everytime the database is accessed thereafter, the error message -169 will be displayed. This is to indicate that the database may not be consistent. However, if the program which was accessing the database at the time of the abend was only reading from the database files there will be no problem with consistency and the clist GUS.CLIST(DIA) should be run. Running DIA causes the database to be closed down correctly and the -169 error message will no longer be displayed when the database is accessed. If, however, the database was being updated at the time of the abend the files may be inconsistent and the clist GUS.CLIST(REFORM) (see section 24) should be used to validate the database before GUS.CLIST(DIA) is run.

When GUS.CLIST(DIA) is executed, the database is opened in the usual way and following some introductory text the user is prompted for a reply which should be one of the following:

1) $OKAY if it is known that the database is consistent

2) $EXIT to terminate DIA

N.B. If there is any doubt about consistency, always run GUS.CLIST(REFORM) first.

24. VALIDATING THE DATABASE GUS.CLIST(REFORM)

GUS.CLIST(REFORM) is used to validate the database if it has not been

properly closed down due to the abnormal termination of a program which was acccessing it. When the clist is executed, the database is opened in the usual way and RAPPORT error -40 will occur. If any incorrectly closed files require validation a list of their names will be displayed followed by the message 'enter names of rapport files affected by altered hashing terminated by $GO:' The names of all files which were listed as requiring validation should now be entered, followed by $GO. If no files were listed then $EXIT should be entered.

N.B. Following the prompt for file names it is possible to enter $ALL instead of individual file names. However this should be avoided if at all possible since the REFORM utility is very slow on the AS/7000 and it takes a long time to run it on all files.

When the database has been validated the data will be consistent and the clist GUS.CLIST(DIA) (see section 23) should be run and then the data should be checked and any updates or insertions which were lost in the abend should be re-done.

APPENDIX A - USER SITES

Abingdon
Bangor
Bath
Belfast
Bidston
Birkbeck
Birmingham
Bracknell
Bradford
Brighton
Bristol
Bury
Cambridge
CERN
Chelsea College
Copenhagen
CRAY
Daresbury
Dundee
Durham
East Kilbridge
Edinburgh
Exeter
Geneva
Glasgow
Grenoble
Hull
IBM
Imperial
Keele
King's College
Lancaster
Leeds
Leicester
Liverpool
LOGICA
London
London Poly
London School of Economics
Loughborough
Mancaster
Manchester
NAS
Newcastle
Oxford
Portsmouth
Queen Mary College
Reading
Royal Holloway College
Rutherford
St. Andrews
Salford
Sheffield
Shell
Southampton
Surrey
Sussex
Swindon
Teeside Poly
Tennessee
Texas
ULCC
Ulster
UMIST
University College
Wallingford
Warrington
Warwick
Wormley
Wokingham
York

APPENDIX B - BOARDS AND COMMITTEES

Abbreviation	Title
ASR	Astronomy Space and Radio
AI	Astronomy 1
AII	Astronomy 2
SS	Solar System
EB	Engineering Board
EC	Environment
EPC	Engineering Processes
IEC	Information Engineering
MC	Materials
MPC	Machines and Power
NP	Nuclear Physics Board
NS	Nuclear Structure
PP	Particle Physics
SB	Science Board
BS	Biological Sciences
C	Chemistry
M	Mathematics
P	Physics
SBA	Science Based Archeology
NBRC	Neutron Beam Research
LFC	Laser Facility
SRFC	Synchrotron Radiation Facility

APPENDIX C - GMS DATASETS

The following datasets reside on TSO packs, unless otherwise stated:

GUS.GU.RAPPORT.DDF.DATA	The database DDF
GUS.GU.RAPPORT.DCF.DATA	The database DCF
GUS.GU.RAPPORT.ACCOUNT.DATA	Contains the ACCOUNT file of the database
GUS.GU.RAPPORT.GRANT.DATA	Contains the GRANT and CGRANT files of the database
GUS.GU.RAPPORT.USAGE.DATA	Contains the USAGE, CUSAGE, POST and CPOST files of the database
GUS.GU.RAPPORT.USER.DATA	Contains the NAME file of the database
GUS.GU.RAPPORT.QUERY.DATA	Contains the QUERY files of the database
GUS.GU.RAPPORT.DUMP.DATA	PDS for individual file dumps (resides on DL0238)
GUS.GU.RAPPORT.DUMPDB.DATA	Full database dump (resides on DL0238)
GUS.GU.BOARD.DATA	Contains the board/committee abbreviations and expansions (resides on DL0238)
US.GMS.LATEST.DATA	Contains the number of the month which was last updated
USG.USERREG.MAIN.TEXT	Contains details of users names, ids and account numbers

APPENDIX D - GMS PROGRAMS

Program	Load Module	Description
GU010	GU010	Extracts information for the NAME and ACCOUNT files from USERREG
GU030	GU030	To accumulate usage for each ID/ACCOUNT combination after it has been taken from the SMF tape
GU040	GU040	To extract information on TSO sessions and batch jobs from the SMF tapes
GU050	GU050	To accumulate CRAY usage for each ID/ACCOUNT combination
GU110	GU110	To clear and load the NAME and ACCOUNT files. (The input is from GU010)
GU120	GU120	To clear and load the GRANT file
GU130	GU130	To clear and load the USAGE file
GU140	GU140	To clear and load the CGRANT file
GU150	GU150	To clear and load the CUSAGE file
GU160	GU160	Update the CUSAGE file
GU170	GU170	Update the USAGE file
GU180	GU180	Updates the usage total in the GRANT file
GU190	GU190	Updates the usage total in the CGRANT file
GU195	GU195	To clear and load the POST file
GU196	GU196	To clear and load the CPOST file
GU200	GU200	Print program control module (load module GU200 created from all GU200*GU290 programs)
GU210	GU200	USER prints (from NAME and ACCOUNT files)
GU220	GU200	GRANT prints
GU230	GU200	USAGE prints
GU240	GU200	
GU250	GU200	
GU260	GU200	CGRANT prints
GU270	GU200	CUSAGE prints
GU280	GU200	TUSAGE prints
GU290	GU200	TCUSAGE prints
GU310	GU310	Dump the NAME file
GU320	GU320	Dump the ACCOUNT file
GU330	GU330	Dump the GRANT file

GU340	GU340	Dump the CGRANT file
GU350	GU350	Dump the USAGE file
GU360	GU360	Dump the CUSAGE file
GU380	GU380	Dump the full database
GU390	GU390	Dump the DDF
GU395	GU395	Dump the POST file
GU396	GU396	Dump the CPOST file
GU410	GU410	Load the NAME file
GU420	GU420	Load the ACCOUNT file
GU440	GU440	Restore the full database
GU470	GU470	Reload the DDF
GU510	GU510	Extract GRANT/USAGE information for Committee report
GU520	GU520	Print Committee report
GU530	GU530	Extract CGRANT/CUSAGE information for Committee report
GU610	GU610	Accumulate usage under valid ID/ACCOUNT combinations
GU810	GU810	To update the NAS grant and post files
GU820	GU820	To update the CRAY grant and post files
LATEST	LATEST	To overwrite US.GMS.LATEST.DATA with the number of the month being updated
IDATE	IDATE	Returns the data as an integer of the form YYMM
NASACC	NASACC	To print NAS monthly statements
CRAYACC	CRAYACC	To print CRAY monthly statements

All the above programs are stored in the TSO archive system and (with the exception of GU040, LATEST and IDATE) have names of the form: GUS.R3.program.FORT . The program IDATE is stored in GUS.IDATE.ASM, GU040 is stored in GUS.R3.GU040.ASM, and LATEST is stored in GUS.GU.LATEST.FORT.

APPENDIX E - NECESSARY STEPS IN CREATING THE DATABASE

The following list gives a brief outline of the steps taken in creating a RAPPORT database.

1) Create the DDF
2) Preprocess the DDF to give the DCF
3) Create the NAS direct access files on which the database is to reside (GUMAKE)
4) Empty these NAS files (EMPTY)
5) Load the QUERY messages and questions (GUQPREP)

The database is now ready for use by both batch jobs and via the interactive QUERY language, the next step being to load some data into it.