DOCUMENT RESUME

ED 092 011 HE 005 571

TITLE	OASIS General Introduction.
INSTITUTION	Stanford Univ., Calif.
PUB DATE	73
NOTE	27p.
EDRS PRICE	MF-$0.75 HC-$1.85 PLUS POSTAGE
DESCRIPTORS	*Computer Programs; *Computers; Data Processing; *Educational Administration; *Higher Education; Information Processing; Management Systems; *Programing Languages
IDENTIFIERS	*OASIS; Project INFO; Stanford University

ABSTRACT

Recognizing the need to balance generality and economy in system costs, the Project INFO team at Stanford University developing OASIS has sought to provide generalized and powerful computer support within the normal range of operating and analytical requirements associated with university administration. The specific design objectives of the OASIS system are: (1) responsiveness to information needs, current and projected, of major university administrative offices; (2) support with both volume batch and teleprocessing requirements with reasonable efficiency from the same file; (3) fast terminal response and simultaneous batch program activity; (4) minimal system overhead and a favorable cost/performance ratio; (5) hardware and software reliability features appropriate to the demands of an online environment. This document covers the executive control services, data base services, terminal services, and generalized services of the OASIS system. Appendixes include the OASIS file structure components, network design, development plans, and command and reserved words. (Author/PG)

OASIS

GENERAL INTRODUCTION

STANFORD 1891 UNIVERSITY

Project INFO

STANFORD UNIVERSITY

OASIS has been developed by the staff of Project INFO with the assistance of other members of the Management Systems Office of Stanford University and major support from a grant by the Ford Foundation. It is available without charge to tax exempt educational institutions for their own non-profit purposes. License for other uses may be obtained.

Detailed documentation on OASIS is contained in three volumes as shown below. An OASIS Newsletter is published quarterly and is available upon request.

Volume I—OASIS User's Guide (to be published in 1974)
Volume II—OASIS Application Programmer's Guide
Volume III—OASIS System Maintenance Guide

Address all communications to Director, Project INFO, Encina Hall, Room 30, Stanford, California 94305.

CONTENTS

INTRODUCTION

System Objectives & Features. The development of computer technology has been characterized since its inception almost three decades ago by an exponentially decreasing cost of providing electronic machine performance balanced against an exponentially increasing range of requirements and complexity in computer systems. Historically, computer systems have been tailored to rather specialized functional requirements which made the most efficient use of their potential. However, as the use of computers expanded within organizations to embrace not only most major clerical functions but also a number of important management support functions, this fragmentation of systems has become a serious handicap. Input and output format requirements are rigid and expensive to change, reports and analyses needing cross-functional gathering of information generate excessive one-time programming requirements, and identical items of information become embedded in multiple machine-readable files.

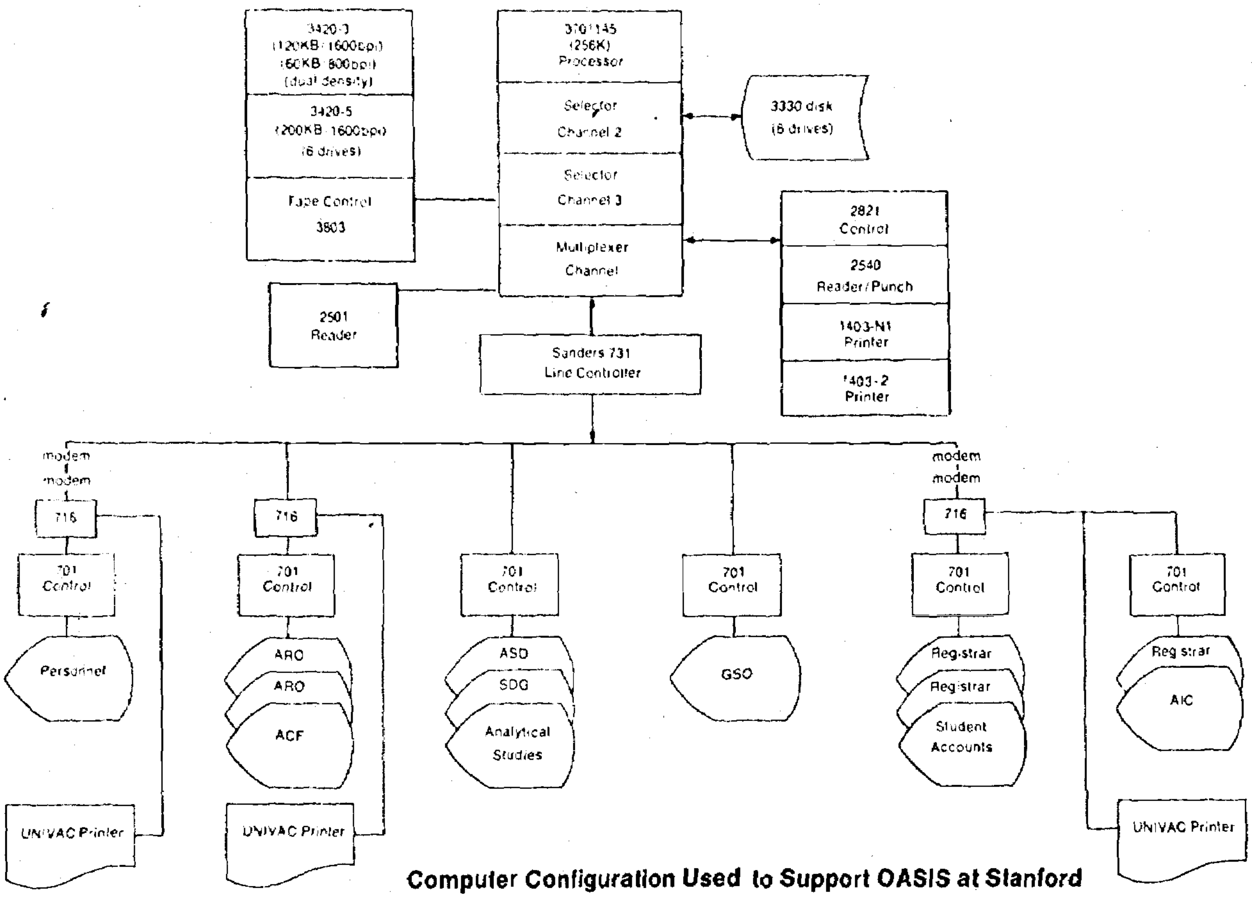
There have been a number of major generalized system development efforts carried on in recent years in an attempt to deal with the need for a unified approach to organizational information processing requirements. The results of these efforts have frequently been mixed, either because of attempts to provide a breadth of system capability beyond the economic reach of the using organization, or because the system failed to address sufficiently current information system problems. Recognizing the need to balance generality and economy in system costs, the Project INFO team developing OASIS has sought to provide generalized and powerful computer support within the normal range of operating and analytical requirements associated with university administrative needs.

The specific design objectives of the OASIS system are:

- Responsiveness to information needs, current and projected, of major university administrative offices.
- Support for both volume batch and teleprocessing requirements with reasonable efficiency from the same file.
- Fast terminal response and simultaneous batch program activity.
- Minimal system overhead and a favorable cost/performance ratio.
- ware and software reliability features appropriate to nands of an online environment.
- Modular and flexible system design to meet changing requirements.
- Support for integration of data files and for consistent naming and definition of data elements.
- Inclusion of features oriented to the non-technical user.

These objectives have been realized in a data base management system with four logical components: Executive Control Routines, Terminal Services, File Services, and Generalized Services. Major features of the system include:

- Online multitasking within a single program partition, with each task protected from the activities of other tasks.
- A terminal command language which allows easy user access to all system features.
- Terminal communication facilities available to the application programmer that simplify the design and execution of online programs.
- A comprehensive set of services, including routines for defining, building, accessing, and restoring files. File access code is resident in the OASIS online partition and is reached by application programs via standard calling sequences. Access calls include retrieval, addition, and deletion services. Variable length record segments provide a significant degree of data compaction. All data may be referenced by name via dictionary facilities, thus allowing the implementation of extensive data security down to the data element level.
- Dynamic allocation of memory to online program modules and data buffers which conserves real memory and supports sharing of reentrant code where possible.
- A file enqueue/dequeue facility which allows concurrently executing batch and online OASIS programs to access information from the same files.
- An inquiry language, QUERY, which permits retrieval of selected file contents based on data element name specifications and selection criteria, and supports the arithmetic operators plus, minus, multiply, divide, and the special functions COUNT, SUM, MIN, MAX, and AVG. An ORDERED option in QUERY provides an internal sorting capability to produce lists in ascending or descending sequence based on data element values.
- Multiple entry to file contents is possible through the specification of varying levels of indexing of elements contained in the file.



Computer Configuration Used to Support OASIS at Stanford

- An interactive Terminal Report Generator allows the composition, specification, and testing of report programs which may be subsequently named, cataloged, and executed either online or in a batch mode.
- *Support for online application programs written in ANS* COBOL in modular format.

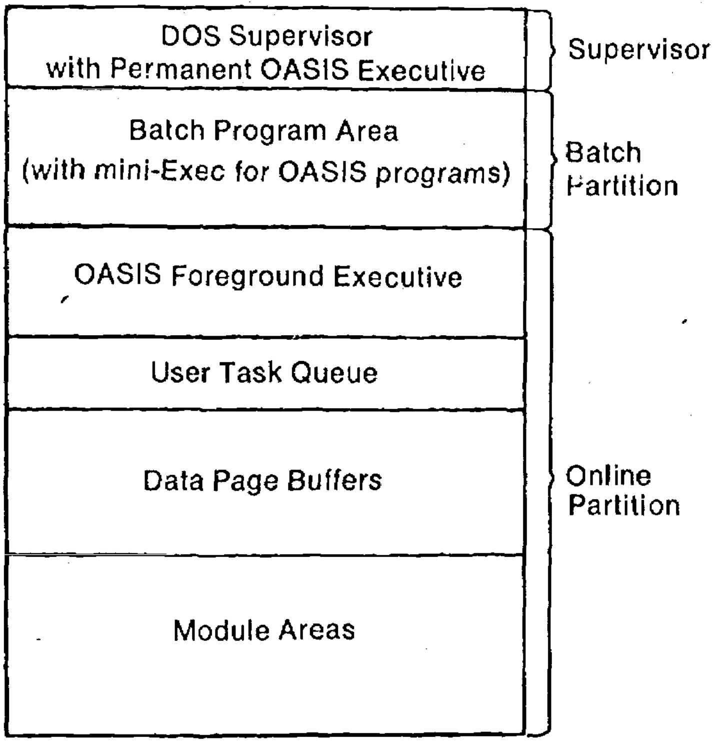
Operating Environment. OASIS is designed for use on medium scale IBM 360/370 computers under control of the Disk Operating System (DOS) and currently supports the attachment of Sanders 720 series CRT displays to the processor. (See Appendix for discussion of work now in progress to support OASIS under the full Operating System and with IBM 3270 series CRT terminals.) Varying hardware configurations are possible, depending on the specific requirements of an installation and the total load of OASIS and non-OASIS work to be performed. The machine configuration currently used in the Stanford Administrative Computing Facility is shown above. The OASIS online system is operated eight hours a day at Stanford and occupies 150K bytes of memory during that time. Less would be required if fewer terminals were to be supported. The minimum size in which several terminals could be driven using all OASIS services is approximately 100K bytes. During the evening operating period, OASIS files are available to batch programs in the same manner that standard IBM disk and tape files are attached to partitions and programs.

The relationship between DOS and OASIS is that OASIS appears in all major respects just like a normal job step. Certain modifications have been made to DOS to allow OASIS to maintain its integrity against inadvertent job cancellations and to enable inter-partition communications. These modifications are self-contained and release-independent. A tape drive for system logging messages is continuously attached to OASIS during both online and batch operations to provide restart and error recovery services.

It is important to maintain the distinction between the OASIS system itself and applications which use it. OASIS is written in IBM assembly language and its programs perform an array of services which, in the aggregate, comprise an online data base management system. Application designers may use this system to perform a broad variety of functions, depending upon specific needs of an administrative office. For instance, a small and relatively static file of specialized information, such as a university space inventory, could be placed online primarily to allow use of the generalized QUERY service to answer non-routine requests for data. On the other hand, a major student records application may require development of online data capture, file maintenance, and extensive printing of documents such as grade slips and transcripts, in addition to providing immediate online inquiry into the file to determine student directory information or academic performance status. In this case, application programs would be written in networks of 2K COBOL modules to handle the online requirements, and larger COBOL programs would be written to execute in batch partitions to support specialized printing and reporting needs. QUERY and the Terminal Report Generator would be used to deal with standard inquiry and reporting requirements. (See Appendix for discussion of online network programming.)

EXECUTIVE CONTROL SERVICES

OASIS Executive Control. The OASIS Executive Control Services perform a number of important functions for the system, including task scheduling, allocation of memory to tasks and I/O buffers, and communications between partitions and the DOS Supervisor. These services are divided into three physical sections of code, the first of which is resident within the DOS Supervisor and is called the *permanent Exec*. The second section controls the online partition and is called the *foreground Exec*. The third section is link edited into batch programs which access OASIS files and is called the *mini-Exec*. The first two sections are required for all terminal operations; the third is needed only when OASIS batch programs are executing. The memory layout of OASIS under DOS is shown below.



lemory Layout of 370/145 When OASIS Is Operating

During system IPL procedures, the OASIS permanent Exec is inserted into the DOS Supervisor, requiring approximately 2K bytes of memory. The principal function of this program is to intercept and analyze all Supervisor Call (SVC) and program check interrupts. Most SVC requests are passed on to DOS; however, a number of them have been designed for use by OASIS and their processing is done within OASIS. The permanent Exec also performs initial processing before routing program checks to the appropriate DOS error routines.

The foreground Exec is a collection of programs, most of which are continuously resident in main memory, that supervise and coordinate the operation of multiple user tasks. Each terminal is considered a task by OASIS and when active may be executing either user-written program modules, the OASIS Generalized Services, or code within the Exec itself. Associated with the foreground Exec is a table of task information blocks called the task queue. Each of these blocks is associated with one of the terminals and contains status information which is maintained by the task scheduler and used to determine execution priorities, monitor errors, and allocate resources to tasks. OASIS is an I/O-driven time sharing system; once a task commences execution, it normally is allowed to continue to execute until I/O is requested or termination occurs, at which point the scheduler saves the current task status, requests the appropriate I/O from DOS if necessary, and branches to the task next in line for execution. OASIS is not a "swapping" system and does not write to disk the modules being used by a task between active execution periods.

The Exec also contains a polling routine to handle communications with the CRT terminals. The processor interval timer is set to a value which is dependent upon the total number of terminals connected to the system. When the timer elapses and causes an interrupt, the polling routine takes control and determines whether any terminal is seeking to communicate with OASIS. If a response is found to the poll, the status of that task is updated to request processing. Certain other status checking is also done at this time and control is then returned to the active task at the point where the timer interrupt occurred.

The foreground Exec operates in Supervisor state with a memory protect key of zero. This allows the setting of protect keys for all task modules operating in the online partition, and prevents an active task from modifying information in modules of another task or in the system itself.

As can be seen from the diagram on the previous page, a major portion of the online partition is allocated to a program module area and a data page buffer area. The former is separated into 2048-byte blocks because this is the minimum area to which the protect key may be applied. The latter is divided into 1696-byte blocks to match the capacity of the disk storage devices. Allocation of module pages and data buffers to tasks is handled dynamically by the scheduler.

When communications are needed from within a task program to an OASIS Exec service, those requests are made via a standard CALL statement of the form: CALL 'Service' [USING param1, param2, . . . paramN] where the parameters are dependent on the service called. There are three classes of service which may be accessed by a CALL statement:

- File Services requests—those activities related to disk and tape files.
- Terminal Services requests—those activities related to the terminals.
- Control Services requests—all other activities (most of these services are concerned with module linkages and backup/restart activity).

When a CALL is made from a task program, control is transferred to the Exec at the entry point of the particular service. (The service routine acts as an extension of the task.) For those services which include no terminal or file I/O activities, the above procedures are executed with no interruptions. For those services which do include I/O activity, however, the task is retired from active state while DOS processes the I/O request.

When the active task wishes to transfer control to another of its network modules, it issues a CALL request to indicate the name of the new module and the desired form of loading function. OASIS then loads and link edits the requested module from the OASIS loader file and transfers control to the entry point within that module. Four linkage services are provided:

- OLINK—a new copy of the module is loaded into one of the module areas.
- RLINK—the module areas are searched for an unused copy of the module; a load occurs only if no old copy is available.
- LINKR—the module areas are searched for a copy of the module, even if it is in current use.
- OREPLACE—the new module is loaded on top of the calling module.

The OASIS Generalized Services use the LINKR linkage service in order to minimize memory utilization and load execution time.

Command Language Processor. The Command Language Processor (CLP) associated with the OASIS Exec allows the user to communicate with the system. Through the use of keywords, he can call the Generalized Services, utilize the debugging facilities, load and execute his programs, perform system functions (such as LOGON, LOGOFF, and CLOCK), or send messages to the operator's console. The operator's console has additional privileged operations related to the status and condition of the system.

The user communicates his requests via keyword commands or their optional abbreviations. (See Appendix for keywords and their meanings.) Certain features of the CLP are standard for all requests:

- When ready for another user-specified command, the system will prompt "COMMAND?" The user may respond with any of the CLP functions.
- An arrow (↑) may be typed in as a user command at any time. The user will then be permitted to enter any command.
- If an error has been detected in a user command, the system will respond with "RETRY" and an appropriate message.

Each command must contain a keyword. Most commands also require one or more parameters. No special formatting is required and no command terminator is used.

The time between user LOGON and LOGOFF is considered to be a *session*. To activate his terminal, the user types LOGON followed by his password. A successful request results in a display of the date, time of day, and message-for-the-day; the user may then enter further requests. When the user desires to terminate a session, he types in LOGOFF. The system responds with the length of his session and the terminal is returned to its inactive state. When a temporary exit from a problem program occurs, a session break results. The user may enter any desired commands, then return control to the problem program by entering the command CONTINUE. An abnormal session break occurs when a problem program interrupt occurs. The system will respond "PROGRAM CHECK INTERRUPTION" with the actual location, condition code, and type of error.

Once the user has successfully performed LOGON, certain functions are always available. To find the date, time of day and current session duration, the user types CLOCK. He may send a message to the operator's console through the use of the MSG command. Certain other commands (QUERY, REPORT, DEFINE) may be input to the CLP and will result in the execution of Generalized Services programs.

Problem programs are loaded from the OASIS loader file and put into execution through use of the LOAD and EXECUTE commands, to which are appended program names. (The load and execute processes may be entered independently in order to facilitate debugging functions.) In response to such a request, the system loads the first module of the requested program into a 2048-byte area, or returns appropriate error messages if the program cannot be loaded. Execution of the program commences directly if an EXEC command was entered. Upon normal completion of program execution, the system will respond "NORMAL TASK TERMINATION".

To facilitate terminal debugging of problem programs, the system provides special debugging commands, which are available only to specified terminals. As part of these capabilities, the terminal user may perform program tracing (TRACE and RESET commands) or specify and clear his own 'break points' by means of the BREAK and CLEAR commands. When a specified location is reached by the program, a session break results and control is transferred to the user. The programmer can check the status of his program or attempt to correct a program check interruption through use of the SHOW and PATCH commands. If he prefers, he may have selected core location values printed out on the printer via the DUMP command for later reference. Displays and modification of his user module areas, registers, and program status are possible.

The system also provides a set of restricted commands which may be used only at the operator's console. Three of the commands are concerned with the activation and deactivation of terminals. Before a user can LOGON from a given terminal, the system must be alerted that communications may be coming from that terminal. The system normally is alerted to this condition at IPL time. However, additional terminals may be brought up through use of the ATTACH command. A terminal may be deactivated through use of the DETACH command. Information on the number of active terminals and the nature of their activities may be displayed through use of the COUNT and LIST commands. The STATUS command will yield information concerning the status of an individual terminal. The operator may enter a message-for-the-day which will appear to each user at LOGON time through use of the DAYMSG command, or send a message to a selected terminal via the SENDMSG command.

Backup and Restart Facilities. The OASIS Data Base is serviced by a complex set of backup routines for all update activities, whether they originate in the batch or online partitions. The backup trail is maintained on the OASIS logging tape and consists of a variety of record types which record all update requests at the I/O services level and also maintain the status of transaction cycles and task initiations/terminations. (The logging tape also contains records used in producing system statistical reports.) The backup trail is recorded in such a manner that restart procedures will involve only those files which must be restored, although all changes are recorded for all files.

Overriding all other considerations in a system like OASIS is the need to keep the online system in operation as much as possible, with few interruptions. The interruptions that occur must be as short as possible. This need to maintain a stable terminal environment makes demands on the scheduling of batch jobs, the design of online jobs, and on the nature of the backup system. The design of OASIS backup reflects this need, in that:

- The response time is affected as little as possible: relatively few tape records of short length are produced for backup.
- Every attempt is made to avoid restart when a crash occurs; a complex set of switches is checked prior to requesting restart.

- Restart time is minimized by replaying only the records in those files known to require processing.

Under OASIS, it is not possible to produce 30-second recovery procedures. The file structure makes it impractical to record 'before' and 'after' snapshots on every modification to the files. The time required to write out the logging records would impact the response time at the terminals. As designed, the tape logging time is essentially absorbed within the disk I/O times.

Because OASIS restart procedures require restoring parts of the data base and reprocessing successful updates subsequent to that time, attention should be given to design of programs and procedures to reduce the number of situations which would necessitate restart procedures.

Periodically, the OASIS Data Base should be dumped to tape and a new logging trail initiated. (No updates prior to that point in time will ever have to be replayed.) The dump is performed by executing a special OASIS batch utility which selectively dumps to tape the active pages on the Data Base. The utility permits dumping of the entire Data Base or selective dumping by file. Ordinarily, regularly scheduled full dumps should be taken, then dumps of individual files taken prior to large batch maintenance jobs.

The length of time to perform restart is primarily dependent on three things:

- Number of files requiring restart
- Volume of updates since the last dump of the Data Base
- Time span since the last backup.

The following factors are counterbalanced against the restart time:

- The probability of a crash
- The availability and cost of the execution time required for backup
- The consequences of a crash/restart at a given point in time.

Under OASIS, restart is not necessarily required each time a partition crashes. Only one condition requires a mandatory restart. All other restart requirements are program-induced.

[illegible]datory restart is brought about when a partition aborts and at least one task was *in the middle of an update service activity*. 'Task' is defined as a batch job or one of the terminal programs. 'Update service activity' means that the task made a request to a File Services updating function and the I/O processing for that function was not completed. Thus, the cause of the crash is relatively unimportant. What is critical is the state of the tasks that were currently in operation. If an update function is left incomplete, that file must be restored to insure that its indices and data records are preserved. It does not matter whether that particular task caused the crash.

Although only an incomplete update activity forces a restart, two programming conditions may also cause a restart if the program is abnormally terminated:

- A file maintenance program with no internal restart facility.
- A file maintenance program which requires that transactions be completed in sets of multiple update functions.

Although most programs can be designed to avoid the restart requirement, there will probably be circumstances when one of the above conditions cannot be avoided. The presence (or lack) of program-induced restart is specified to the OASIS backup system via certain program CALL requests. In addition to these requests, OASIS provides a service which will punch a restart parameter card(s) for a batch program to enable that program to avoid reprocessing when an abort occurs.

When OASIS detects that a file has been invalidated, a console message notifies the operator that restart processing must be done. Although the system will continue to operate for inquiry purposes, no further updating will be permitted until the restart procedures are complete.

The OASIS restart program will list the file(s) that must be restored and request the most recent backup tape on that file and the corresponding logging tape(s) from that point in time. Processing is then automatic: OASIS restores the active pages for the necessary file(s) and spins the logging tape to find and replay any successful updating records against that file. Additional tapes will be requested if needed and a message will notify the operator when the Data Base is again intact. Online OASIS and update processing may then be reinitiated.

DATA BASE SERVICES

File Structure. Employing a hybrid file structure, the OASIS system provides services which allow the user to define and maintain an integrated Data Base. The file structure:

- Allows flexible definition of file characteristics (e.g., indexing, security, segmentation) which can be tailored to application requirements.
- Provides for quick entry into the Data Base via indices.
- Disengages data descriptions from programs.
- Provides for sequential processing based upon a particular key.
- Provides for online or batch update of the Data Base, including modification, addition, and deletion.
- Allows for access to information in one file through its related files.
- Accomplishes the above with minimum sacrifice to file compactness.

The Data Base is physically resident on a disk storage device which is divided into a collection of physical blocks called pages. There are four 1692-byte pages per track on an IBM 2314/2319 Disk Storage Device, seven pages per track on an IBM 3330 Device. (See Appendix for a pictorial description of the Data Base.)

The OASIS Data Base is a collection of files which may or may not be related. The number of files which may be included in any one OASIS Data Base may vary from 1 to 30,000. A *file* is a physical collection of variable-length *records*, each of which is subdivided into a physical collection of *segments*, which may contain one or more *elements*. The element (or field) is the smallest definable unit of data or information. A collection of elements may be defined as a *group*, meaning that they are logically (rather than physically) related. Elements in related files may be referenced by means of *indirect references*. A pictorial description of the OASIS file structure is shown on the next page.

An element is a contiguous series of bytes in one of four data types: character, binary, packed decimal, and zoned decimal. Numeric elements may specify scaling factors. Segments may be subdivided into one or more elements. Each segment type within a given file has a fixed length, although the length for differing types will usually vary. A data record is a collection of one or more logically related segments. Within any one file there can be a maximum of 254 different segment types. There is no relationship between segments with the same type number in different files. In general, a segment may occur any number of times in one data record (including zero times). However, during file definition each segment type may be restricted in one or both of the following ways:

- The segment may be required to appear at least once in every record in the file.
- The segment may be limited to no more than one occurrence in each record in the file.

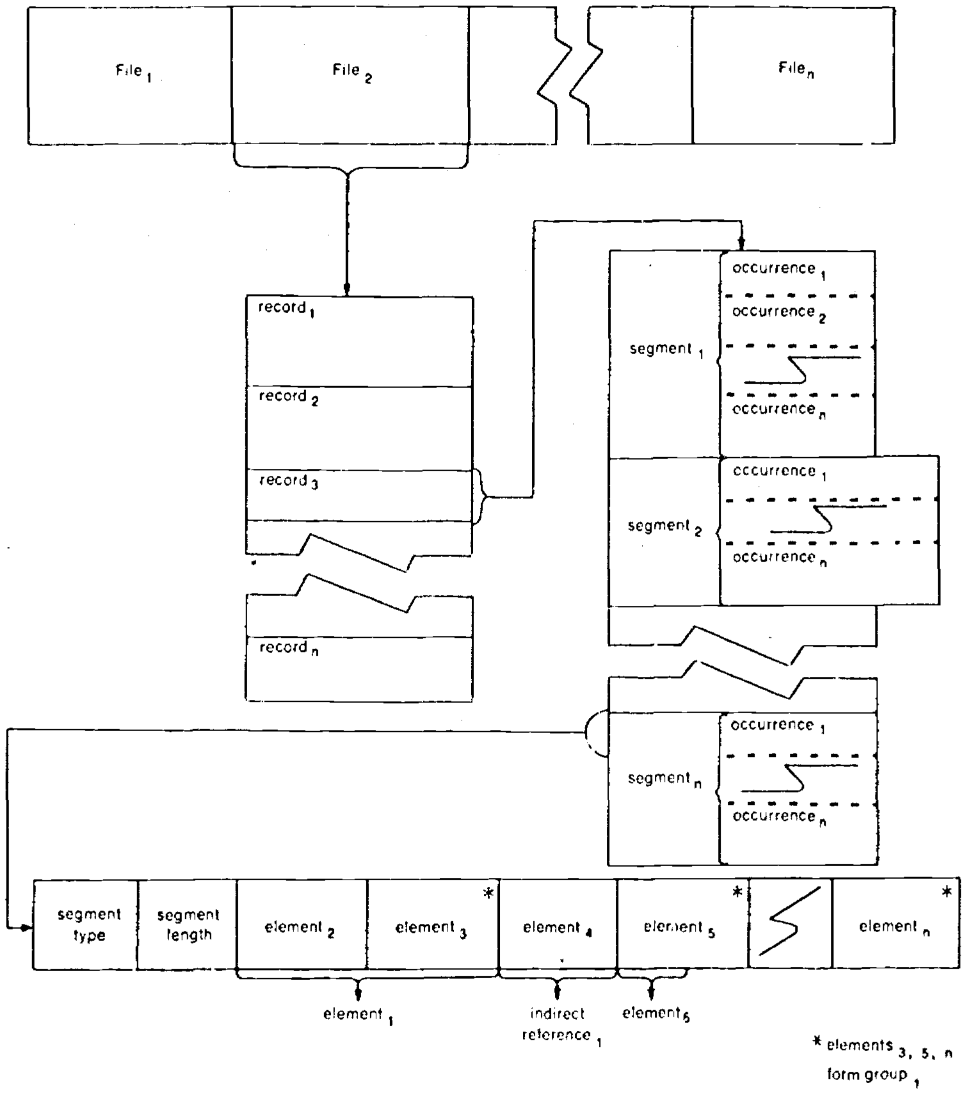
Indirect references make it possible to reference data in a secondary file indirectly through reference to a primary file, just as though the data had been contained in the file through which they were requested. A common usage of indirect references occurs as a form of table lookup, in which an abbreviated code is carried in the primary file and the expanded translation in a related secondary file so that the full field occurs in the Data Base only once.

Data Base Dictionary and Data Names. Each data item to be accessed by a user must have an OASIS name by which the information can be specified at the user level. These names are specified to the system during file definition and then stored in the OASIS Dictionary. OASIS edit pictures are also specified for use in edited displays by the Generalized Services and by application programs as desired. The information in the Dictionary may be accessed by a user program by means of the DESCRIBE service.

Valid OASIS names and passwords:

- are from one to sixteen characters in length, with the permissible characters being the letters A through Z, the digits 0 through 9, and the period.
- must begin with a letter.
- must *not* end with a period.

OASIS edit pictures are used to specify how element data are to be edited for display purposes. The rules for formation of these pictures are extremely broad and permit inclusion of almost any printable character.



OASIS File Structure

The OASIS Dictionary is used by File Services as a master directory in manipulating the data within the records of the Data Base according to requests from application programs and Generalized Services programs. Because OASIS names need not be unique within the Data Base, qualification with a particular file would often have to be performed. Since each data name may be as long as sixteen characters, much space would be required by File Services to identify the requested information. Instead, a unique 4-byte *compact reference* is assigned to each element, group, and indirect reference. This compact reference is then used for all File Services routines.

However, since the full names are more natural and meaningful to the user, translation services are provided to convert names to compact references (NAMETOCR service) and back again (CRTONAME service). This approach also makes it possible for the user of File Services to separate his logic for testing the validity of a data request from the logic for actually making the data request.

Security Structure. Maintaining confidentiality of data in computer systems requires detailed attention to many different facets of information handling. Included are questions of the physical location and security of computer processors, files, and terminals; handling of machine readable data in forms such as disk packs and magnetic tape reels; procedures for user offices to safeguard information in hardcopy form, and the measures taken in the data base system to provide access only to authorized individuals. The discussion below focuses on the security features which are embedded in OASIS itself; however, adequate protection of sensitive information will require policies and procedures that embrace the whole range of forms and uses of data stored in the system.

All information stored in OASIS data files has associated with it two security *codes*, an access code and a modify code, which control the use of the information. All users of the OASIS system will also have two security *clearances*, an access clearance and a modify clearance, which determine the files and information in the files which the user is permitted to access and modify. (Modify clearance is required both to change the value of old data and to create new data.)

Each data file in the OASIS system has its own set of security codes, which do not relate to those of any other file. Each element has its own access and modify security codes. Groups do not have their own security codes but are controlled by the individual security codes for elements in the group. Each segment type has its own access and modify security codes, which default to the lowest codes which cover all the elements in the segment.

For an OASIS user to either access or modify data in an OASIS file, he must have a security clearance for that file which is either equal to or greater than the security code of that data. The security clearance for a given user is determined by the password he provides during LOGON. Each password has associated with it the access and modify clearances for one or more files which the holder of the given password has been authorized to use.

Data base security levels of three kinds are provided in the OASIS system:

- A main hierarchy
- *26 exclusive hierarchies*
- An absolute security.

The main hierarchy for each OASIS file consists of 32 security levels, known as levels 1 through 32. There are also 26 exclusive hierarchies, denoted as hierarchies A through Z. There is no significance to the alphabetic order of the exclusive hierarchies; they are all separate, equal, and identical in form and operation. The final level of security is called absolute security, denoted ABS, and is greater than all other securities.

Each exclusive hierarchy contains eight levels, 1 through 8; these are denoted A,1 through A,8, B,1 through B,8, and so forth. Within each exclusive hierarchy the levels have the obvious relationship, so that D,5 is higher than D,4; thus D,8 is the highest level *in the D hierarchy*, while D,1 is the lowest. The exclusive hierarchies have no relationship to each other, except that they are mutually exclusive in the sense that a user can access information which has a security code in the same exclusive hierarchy as his particular security clearance but cannot access information from any of the other 25 exclusive hierarchies. A security code of D,3 is neither higher nor lower than a code of K,6 because the two codes belong to different hierarchies.

The main and exclusive hierarchies are related so that each level within each exclusive hierarchy covers a part (or all) of the main hierarchy. Each level in an exclusive hierarchy covers four times as many levels in the main hierarchy as it does in the exclusive. For example, security F,3 covers (is greater than or equal to) not only its own securities F,1, F,2, and F,3 but also securities 1 through 12 in the main hierarchy. However, while each level in each exclusive hierarchy covers a part of the main hierarchy, no level in the main hierarchy covers any part of any of the exclusive hierarchies.

If an OASIS file contains only one kind of information, only the main hierarchy need be applied, using as many of the 32 levels as are needed.

If an OASIS file contains information which has been gathered from many independent sources, the exclusive hierarchies may be used. Such gathering of dissimilar information into one file may reduce the storage requirement and eliminate the problem of updating separate records of addresses and other common information.

If a file has two kinds of common information (one to be shared by everyone, and one to be shared only by certain groups of users), an overlapping of the main and exclusive hierarchies may be used. The truly common information may be placed at the bottom of the main hierarchy, say at access level 1. The common information to be shared by some of the users may be placed in the middle of the hierarchy, say at level 17. All of the other information is placed in various exclusive hierarchies. The users who are not to be given access to the partially confidential information are given access to their exclusive hierarchies, but they are restricted by convention to the exclusive levels 1 through 4, which keeps them from accessing the data at main level 17. The groups of users who are to share all of the common information use their exclusive hierarchies; however, by convention, they are assigned levels 5 through 8, so that they are always given access clearance to the common information at level 1 *and* level 17.

File Access and Modification. OASIS File Services provides the capability to create, maintain, and access files. Creation of a file requires the execution of four batch programs:

- *File Definition.* An updated OASIS Dictionary is created, which contains information on the names and structures to be used with the new file.
- *Space Allocation.* An area within the Data Base is allocated for storage of the new file and its indices.
- *File Build.* The new file is built on disk from tape input data, and an intermediate tape file is produced which will be used in creating index tables.
- *Index Build.* The intermediate tape file is used to produce the Record Number Index and Value Index Tables required by the file.

Once the four creation programs have been successfully completed, the user may access and update his file by means of the Generalized Services or through application programs which utilize the File Services routines. These routines include user services to:

- Obtain descriptive information about the Data Base (DESCRIBE, NAMETOCR, CRTONAME, GETVAL)
- Access, sort, and subset files (ATTACH, DETACH, SELECT)
- Access and modify data at the record and segment levels (GETSEG, RPLSEG, ADDSEG, DELSEG)
- Access and modify data at the element level (GETDATA, RPLDATA)
- Log online transactions on tape (TAPEWRIT).

All of the services may be called from COBOL and ALC programs by means of a standard COBOL CALL statement. The requests include a return-code argument, which is set at the end of each request.

There are two general methods of accessing OASIS files:

- Direct read from the file, either by a given element value search or according to a given ordering of the file by one indexed element
- Indirect read, via a list of record numbers which were preselected according to a set of selection criteria and/or ordering specifications.

The first method of file access is designed to be used in the following three situations:

- Random retrieval according to one specific indexed element value
- Retrieval of all the records in the file, in the order of a specified indexed element
- Retrieval of all the records in the file, in the order they were added to the file.

The second method of file access is used to select records out of a subset of records from a file, or to request records from a file (or file subset) in a given order other than that of a single indexed element. This method of file access requires calling the SELECT service prior to other access functions. SELECT constructs a list of record numbers of all records in the file that meet the user-specified selection criteria, sorts the list if necessary, and returns a pointer by which the user may indicate the location of the resultant list to the I/O routines. SELECT is activated by an application program request or a QUERY or TRG specification, which may contain as many as 20 conditional expressions connected by OR, AND, or TAND* and up to 10 sort fields. Valid operators are EQ, NE, GR, LS, LE, GE, and RN (range).

In order to access data in the Data Base, the user must specify the file(s) containing the desired data. File Services can then clear the user's security before any I/O requests are made. This service is called ATTACH. The opposite service, DETACH, removes user access from the file(s). It is necessary to attach a file, since all of the I/O services check to see if the file has been properly attached and examine the associated security levels before returning any data to the user or modifying any data in the file.

The File Services routines include record-level, segment-level, and element-level functions. Four services are provided to retrieve and maintain data records at the physical segment level:

GETSEG Retrieves a particular occurrence of one segment type.

RPLSEG Replaces a particular occurrence of one segment type. This service essentially overlays data fields on an already existing physical segment. (Record size does not change.)

ADDSEG Puts a new physical segment into a data record. The service is requested in such a way that the new segment automatically becomes a particular occurrence of one segment type. (The logical record size will be increased.)

DELSEG Removes a particular occurrence of one segment type from a data record. (The logical record size will be shortened.)

Two services are provided to retrieve and maintain data records at the physical element level:

GETDATA Retrieves data on an element, group, and indirect reference basis. The service can retrieve one occurrence of one or more elements and/or groups, or all of the occurrences of one element. (GETDATA can fetch data out of a single file or automatically make indirect feferences to other files.)

RPLDATA Changes the value of a single element.

Additional file-accessing features are available through special usage of the I/O services. These features are signalled by non-standard use of certain of the parameters:

- All occurrences of a given element may be retrieved at once (special GETDATA)
- All occurrences of a given segment may be retrieved at once (special GETSEG)
- An entire record may be retrieved (special GETSEG)
- An entire record may be added to a file (special ADDSEG)
- An entire record ~ay be deleted (special DELSEG).

File Design Considerations. The design of an OASIS Data Base involves the entire process of systems analysis. However, certain characteristics of OASIS should be considered when making such designs. Because OASIS provides the capability for creating an integrated data base, it is critical that the designers of the first files are cognizant of the characteristics of the entire data pool. Under OASIS, no file exists in a vacuum; each may relate to other files and other information. Before any file design is attempted, enough analysis should be performed in all areas proposed for OASIS support in order to determine where data interactions and overlap occur. An effective means of gathering this information is the creation of a data element dictionary.

Under OASIS, the manner in which a piece of data is stored and specified during file definition can be of great assistance during access and modification activities. Since the element

is a special connector to permit comparison of data within n segment occurrence.

is the basic unit used by the Generalized Services, its manner of usage should usually be the first consideration, the space requirement the second. The way a user will want to 'get at' his data plays an important role, since much of OASIS calls for user specification of data. Elements should be defined in familiar units; their names should be descriptive from the viewpoint of the users; the edit picture should be one to which they can relate.

The indexing of elements is probably the most easily abused aspect of the OASIS file structure. It is tempting to index every element that is an access point into the file. However, although indexing does markedly improve access speeds on data items, the space required for the index and execution time needed to maintain it may outweigh the worth of occasional accesses. In determining whether to index an element, one should consider the fact that no element need be indexed, since file data can be accessed according to the value of any element, in the order of that element, regardless of indexing. Indexing simply makes the access faster.

The actual set of indices for a given file should be based on the predicted access to the file. If need be, the indexed items may be changed after the file has been used for a period of time sufficient to evaluate the index usage. The usual types of elements to consider indexing are:

- Primary keys
- Unique or near unique values that are accessed frequently
- Very common selection criteria
- Flag-type information
- Lookup elements for indirect references.

The OASIS file structure permits very efficient utilization of file space because of the optional and recurring qualifications on segment types. By isolating each piece of data into its own optional segment, presumably no file space would be left idle. However, two factors must be taken into account when considering one-element segments:

- A two-byte header is appended to each segment.
- More time is required to access a logical group of elements in separate segments than when they occur in a single segment.

In determining the segment structures for a given file, the first consideration should be the need for file space economy. Usually, the bigger the file the more emphasis on eliminating 'dead' space. Within the boundaries of economy, the elements should be placed into logical groups, oriented more toward modification than toward access. The occurrence patterns of the elements should also be taken into consideration. For instance, recurring items should not be combined with nonrecurring ones.

The concept of the group has been incorporated in OASIS primarily as a convenience; instead of specifying a list of elements each time they are needed, a single group name can be used to produce the same results. Because groups are formed on a logical basis and do not affect the physical structure of a file, they may be added, altered, and deleted as their use (or misuse) becomes known.

Indirect references should be used as space-saving devices. They should be used sparingly, in that their access timings are much slower than direct accesses. When needed, indirect references save so much file space that the reduced accessing speed is counterbalanced.

TERMINAL SERVICES

Terminal Communications. A series of subprograms has been developed to facilitate communication with the Sanders 720 terminals from OASIS application programs. Collectively called Terminal Services, the subprograms are available in two versions:

- the online OASIS version
- a special stand-alone version, which may be accessed through standard DOS linkage editing and utilized by programs in a standard DOS operating environment.

The stand-alone version is useful primarily when first installing OASIS. The terminal equipment may be tested and programmer familiarity with the terminal services may be acquired even before online OASIS is fully installed.

The chief difference between the stand-alone and OASIS online versions is the method of polling terminals: the OASIS Exec automatically handles polling, whereas the stand-alone version requires the specification of a list of terminals which should be polled when requested by a program. The difference in polling methods necessitates special functions for the stand-alone version. Certain standard functions of the Exec, however, can be utilized in order to provide additional services to the online environment which are not possible in the stand-alone version.

Included in Terminal Services are eleven basic functions, which may be divided into three groups.

The standard functions are:

TERASE —fill specified block with blanks (clear block)

TGET —receive response from attached terminal (terminal read)

TPAGE —retrieve format and send to terminal buffer (display page)

TPGPUT —send specified program contents to terminal buffer (display program page)

TPUT —send information to specified block (display block)

The stand-alone functions are:

PTOPL —put terminal on poll list

FPL —remove terminal from poll list

TPOLL —search list of terminals for one which has sent information

The online OASIS functions are:

TASKNUMB —report the terminal number

TRESTORE —redisplay format page as saved

TSAVE —save current format page

These eleven functions may be grouped into five basic types of operation dependent on their form of communication within a user program:

- Ready the device (TASKNUMB, PTOPL, TPOLL)
- Initialize the screen with a format (TPAGE, TPGPUT)
- Receive communications from the user (TGET)
- Alter the contents of the screen (TPUT, TERASE, TSAVE, TRESTORE)
- Release the device (RTFPL).

All of the services may be requested from application programs through a standard calling sequence. Each is specified with a set of user parameters.

Terminal Screen Formats. The formatted terminal display is utilized by terminal networks to facilitate the display and retrieval of file data and is the basis for the interactive nature of the OASIS system. The primary consideration in the design of such displays is the user, the person who will have to interact with that format.

In designing terminal screen formats, it must be remembered that the screen is usually a replacement for printed forms. Initially, for many users, the replacement will seem awkward, in that the printed form required only pen or pencil, or at most a typewriter. The terminal will seem strange and the interaction through typed actions a poor substitute for the better understood and more controllable printed forms. Because of the natural reservations, it is important that care be exercised in designing terminal display formats.

The actual creation of a terminal format involves the design, implementation, and cataloging of the layout. The design of a terminal format is very user-dependent and should consider the manner in which the information will be used.

A knowledge of the file design may be used to reduce the amount of reformatting between file and terminal I/O commands. The most common display formats will involve the following four types of blocks:

- Headings and constant information
- Error-message/action-message area
- Input user-response area
- Output program-supplied areas.

Having established a format design, the actual format is created on a terminal to insure that:

- the spacing can be accomplished, using the control characters
- the format does not exceed the buffer positions
- the layout is balanced and well-positioned when viewed on the live screen.

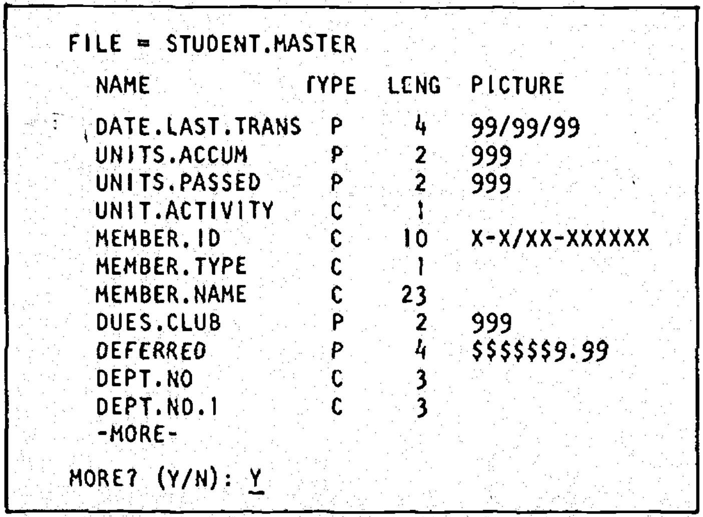
Any adjustments may be made to the format at the terminal. Subsequently it is cataloged in the OASIS loader file by means of the FORMATPG program, which is called from online OASIS terminals. This program may also be used to view, copy, and modify existing terminal formats.

Hardcopy Output. OASIS provides the capability for printing out the contents of the terminal buffer. The user requests this service by typing a backward arrow (←) anywhere on the screen. (The arrow itself will not be printed on the hardcopy device.) The type of printer (if present) may vary and is specified for each terminal through an installation parameter.

Hardcopy printouts may also be requested directly from an application network through a special set of OASIS CALL requests.

QUERY. Generalized file inquiry allows the terminal user to retrieve information from the Data Base by responding QUERY to the Command Language Processor. There are two major types of query requests: descriptive queries and WHERE-clause queries.

Descriptive queries allow the user to review names and specifications of the data to which he has access. Allowable responses are FILES, file-name, group-name, and element-name. A descriptive query follows.



```
    FILE = STUDENT.MASTER
      NAME              TYPE  LENG  PICTURE
      DATE.LAST.TRANS    P      4   99/99/99
      UNITS.ACCUM        P      2   999
      UNITS.PASSED       P      2   999
      UNIT.ACTIVITY      C      1
      MEMBER.ID          C     10   X-X/XX-XXXXXX
      MEMBER.TYPE        C      1
      MEMBER.NAME        C     23
      DUES.CLUB          P      2   999
      DEFERRED           P      4   $$$$$$9.99
      DEPT.NO            C      3
      DEPT.NO.1          C      3
      -MORE-
    MORE? (Y/N): Y
```

First Page Descriptive QUERY for Student Master File

If the query request is FILES, the user receives a list of files to which he has access according to the OASIS security standards. If the request is a file name to which the user has authorized access, he receives a list of element descriptions for that file. The descriptions are composed of element name, type, length, and OASIS picture. If the request is a group name in one of the user's files, a list of elements in the group is produced. If the request is an element name in one of the user's files, a list of the values currently in the file is displayed.

WHERE-clause queries allow the user to select and view portions of the data in his files. Certain functions and arithmetic operations may be incorporated to display modified data. Automatic columnar formatting is performed, and the user is provided an option to view his data in rows instead. Standard headings are included and the data is edited according to the picture specifications in the OASIS Dictionary.

Prior to the use of a WHERE-clause query, the user must specify a primary file that he intends to access. Specification is done by the response QFILE. file-name. The named file is then attached and remains the inquiry file until the user again responds with the QFILE form.

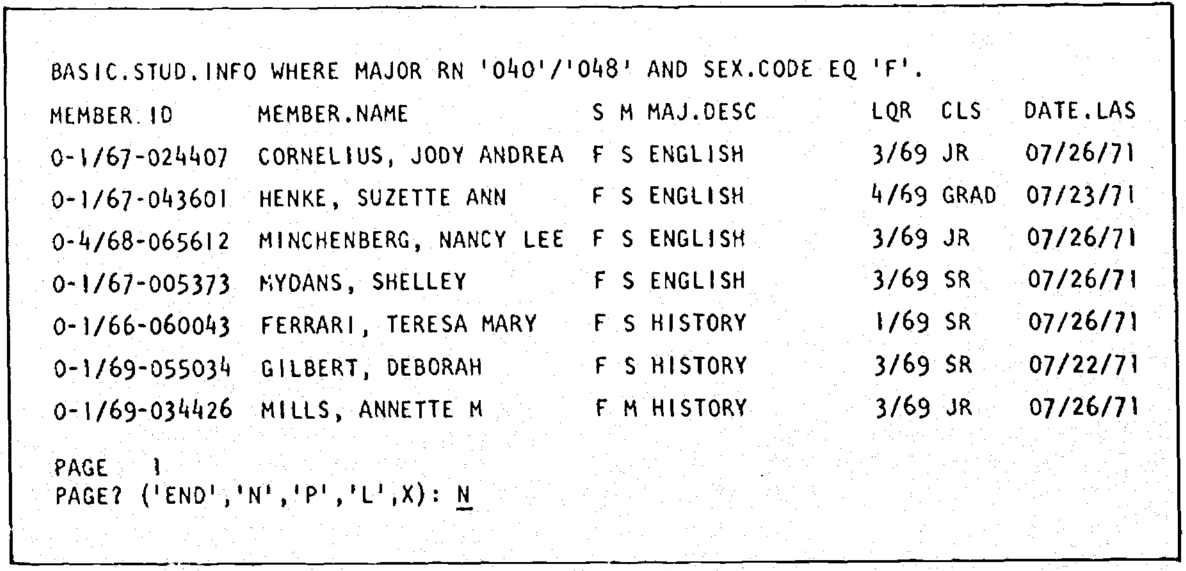
Having specified a file, the user may then proceed with his requests. All WHERE-clause queries are of the form:

$$[\text{RECORDS}]\ \left[\begin{Bmatrix}\text{VERTICAL}\\ \text{COLUMNAR}\end{Bmatrix}\right][\text{list-term } [\ldots, \text{list-term}]]$$
$$[\text{WHERE selection-clause}].$$

If the "WHERE selection-clause" option is chosen, data will be selected according to the user specification. If no WHERE-clause is included, data will be selected from all the records in the file. A specification of RECORDS will result in a count of records meeting the selection criteria.

A *list-term* may be an element name, a group name, an indirect reference name, a function, or an arithmetic expression. (Group names will be converted automatically to a list of element names.) A function consists of a function code followed by an element name. Allowable function codes are: SUM, COUNT, AVG, MIN, and MAX. (COUNT provides the total number of occurrences of a given element, including multiple occurrences in recurring segments.) An arithmetic expression consists of single numeric element names, functions, or literal constants separated by the single operators + − / *. One WHERE-clause query may contain a maximum of twenty literal constants and/or elements. A maximum of five functions and five arithmetic expressions may be used in one query.

QUERY output is displayed one page at a time. Each page begins with a statement of the query request and element headings and ends with the current page-number. The user may then respond according to which page he wants to see next: none, next, previous, last, or a particular one. If he specifies a number greater than the total number of pages, he automatically receives a display of the final page of output. He is free to page back and forth through his output



```
BASIC.STUD.INFO WHERE MAJOR RN '040'/'048' AND SEX.CODE EQ 'F'.
MEMBER.ID          MEMBER.NAME                S M MAJ.DESC        LQR  CLS    DATE.LAS
0-1/67-024407      CORNELIUS, JODY ANDREA     F S ENGLISH         3/69 JR     07/26/71
0-1/67-043601      HENKE, SUZETTE ANN         F S ENGLISH         4/69 GRAD   07/23/71
0-4/68-065612      MINCHENBERG, NANCY LEE     F S ENGLISH         3/69 JR     07/26/71
0-1/67-005373      MYDANS, SHELLEY            F S ENGLISH         3/69 SR     07/26/71
0-1/66-060043      FERRARI, TERESA MARY       F S HISTORY         1/69 SR     07/26/71
0-1/69-055034      GILBERT, DEBORAH           F S HISTORY         3/69 SR     07/22/71
0-1/69-034426      MILLS, ANNETTE M           F M HISTORY         3/69 JR     07/26/71

PAGE    1
PAGE? ('END','N','P','L',X): N
```

First Page of QUERY Showing Information on Female Students in Several Departments

until he responds END. At that point, he may request a new query generation. EXIT terminates QUERY execution.

Generation of data is according to the OASIS security standards. Blank fields will be generated in place of data restricted from the user. Error messages are included to assist the terminal user. Informational assistance is provided in answer to the request HELP.

The first page of the user's query is displayed automatically. The actual generation and display of additional output pages depends entirely upon the requests of the user. Output is generated in consecutive page order; however, only the pages prior to and including a requested page are generated. If the user asks for page 2 and then page 20, pages 3 through 20 are generated even though only page 20 is displayed on the screen. The pages are stored so that repeated requests for a given page do not require regeneration of the output.

Sample queries are shown on these two pages. The first is an example of how the entry of a single group name can be used to retrieve an entire line of data. The processing of recurring elements and the automatic formatting features of QUERY are displayed in the other example.

Terminal Report Generator. Generalized Services also provides a Terminal Report Generator (TRG), by which the OASIS user can define and generate reports using the information in the Data Base. Report definition is performed online at the terminal. Report generation may be initiated from the terminal or via batch input. Reports initiated from the terminal may be displayed page by page on the screen or may be spooled to the logging tape. The user also has the option of generating an online proof of his report.

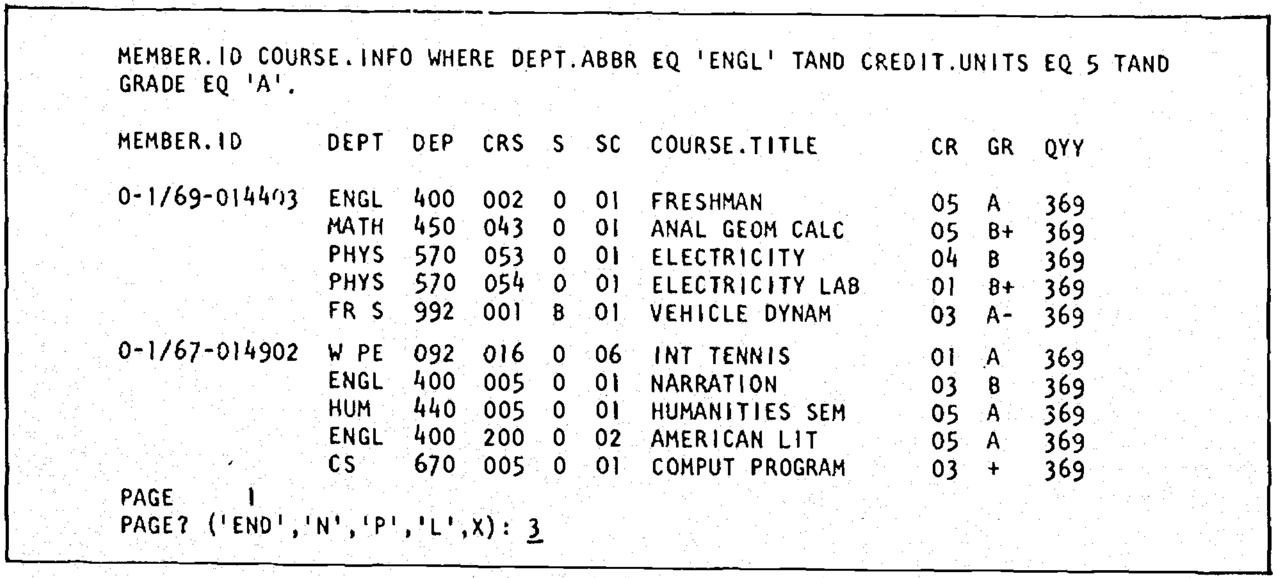
Report definition is performed by responding DEFINE to the Command Language Processor. A series of six questions will appear. As with all requests in the report definition phase, a response of MORE will yield an expanded description of the question. Error messages are provided if the user responds illegally or unrecognizably.

The initial six questions provide general specifications for the user report:

- *Report Name.* Report name is a 1-8 character alphanumeric [illegible] which will be used to identify the given report definition. If the name duplicates an already existing report definition, the user will be asked if he desires to replace the original report or if he wishes to select another name.
- *Output Mode.* The user has a choice of output mode. If the report will always be displayed on the terminal, he responds TERMINAL. If the report will always be printed out (except for proofs), the user specifies BATCH; his line definitions may then cover up to 132 characters. If a report may be displayed on the terminal or printed out, the user responds EITHER.
- *File Name.* File name is the name of an OASIS file from which the data are to be extracted.
- *Report Title.* The user has a variety of options concerning the first line to be printed or displayed on each page of his report. By responding NONE he will receive no standard title line. A string of text enclosed in single quote marks will yield a heading line with that text centered on the line. PAGE will produce automatic page numbering. DATE will yield the date of actual report generation on each page of the report. The user may select any combination of options.
- *Selection Criteria.* The user may specify a sorting sequence for his report and has the capability to produce the report on a subset of his file. Such specification is done by means of a SELECT WHERE-clause. If the selection criteria for report generation will vary, the user may opt to provide selection criteria each time the report is produced.
- *Control Breaks.* The user may select elements which he wants monitored so that total and summary functions may be performed when the value of one or more of these elements changes. During line definition, these control breaks are identified by number: the major break is 1, the next 2, and so on.

Having completed the specifications concerning overall production of his report, the user is ready to define the format and content of the particular types of lines to appear. A report may be composed of three types of line: data, total, and text lines. Each serves a different function and is therefore defined in a unique way.

Data lines are the detail lines of a report and are composed of individual record data, interspersed text, and/or arithmetic expressions. If desired, automatic heading lines will be produced. Another specification will cause the headings to be produced each time a new record is read, appearing immediately above the first occurrence of the associated data line.



```
MEMBER.ID COURSE.INFO WHERE DEPT.ABBR EQ 'ENGL' TAND CREDIT.UNITS EQ 5 TAND
GRADE EQ 'A'.

MEMBER.ID      DEPT  DEP  CRS  S  SC  COURSE.TITLE        CR  GR  QYY

0-1/69-014403  ENGL  400  002  0  01  FRESHMAN            05  A   369
               MATH  450  043  0  01  ANAL GEOM CALC      05  B+  369
               PHYS  570  053  0  01  ELECTRICITY         04  B   369
               PHYS  570  054  0  01  ELECTRICITY LAB     01  B+  369
               FR S  992  001  B  01  VEHICLE DYNAM       03  A-  369
0-1/67-014902  W PE  092  016  0  06  INT TENNIS          01  A   369
               ENGL  400  005  0  01  NARRATION           03  B   369
               HUM   440  005  0  01  HUMANITIES SEM      05  A   369
               ENGL  400  200  0  02  AMERICAN LIT        05  A   369
               CS    670  005  0  01  COMPUT PROGRAM      03  +   369
PAGE     1
PAGE? ('END','N','P','L',X): 3
```

QUERY Showing Academic Program of Students with a Particular Course and Grade

Total lines are summary lines and may include functions (such as SUM and COUNT), detail data, interspersed text, and arithmetic expressions. Production of a total line is determined by a change in the value of one of the specified control breaks. The user indicates by number which break controls the line. Grand total lines may also be defined for production at the end of the report. All functions and expressions are computed automatically and reset after each production of the associated line.

Text lines are used for a variety of reasons. They may be used to obtain additional report heading lines. Comments, notes, and informational messages may be embedded in the content of a report via text lines. By defining text lines which contain only blanks, the user can vary the spacing in his report.

Each time the user specifies a data or total line, he must also define the information (or fields) to be contained in that line. Many types of field content are possible and the types may appear in varying combinations:

- *Data Fields.* Element, group, and indirect reference names may be included. At generation time the element-level security will be checked and, if the user is not cleared for the information, blanks will fill the field.
- *Interspersed Text.* The user may embed text anywhere within his lines by enclosing the desired literal in quote marks. The text will appear just as typed (without the quotes) beginning at the next available position of the line. This feature makes possible headings to the left or right of the data fields and explanatory titles on total fields.
- *Functions.* The functions SUM, COUNT, MIN, MAX, and AVG may be used. When a function appears in a total line, the calculation is performed from production of that line until next production of the line, at which time the value is reset. When a function appears in a data line, the value is calculated based on *all* the records to be processed.
- *Arithmetic Expressions.* In a manner similar to that used in QUERY, functions, numeric literals, and numeric data fields may be combined with operators to form expressions.

Each time the user inputs a new field he receives a display [illegible]ne in its current state; i.e., he watches his line being [illegible]e also sees the associated heading line, if present, and the previous data and total lines. Edit masks are prefixed by a dot, which will not appear on the generated report but is used to distinguish text from data fields. The user may modify the displayed lines by shortening or lengthening edit masks or headings and by shifting them to the left or right.

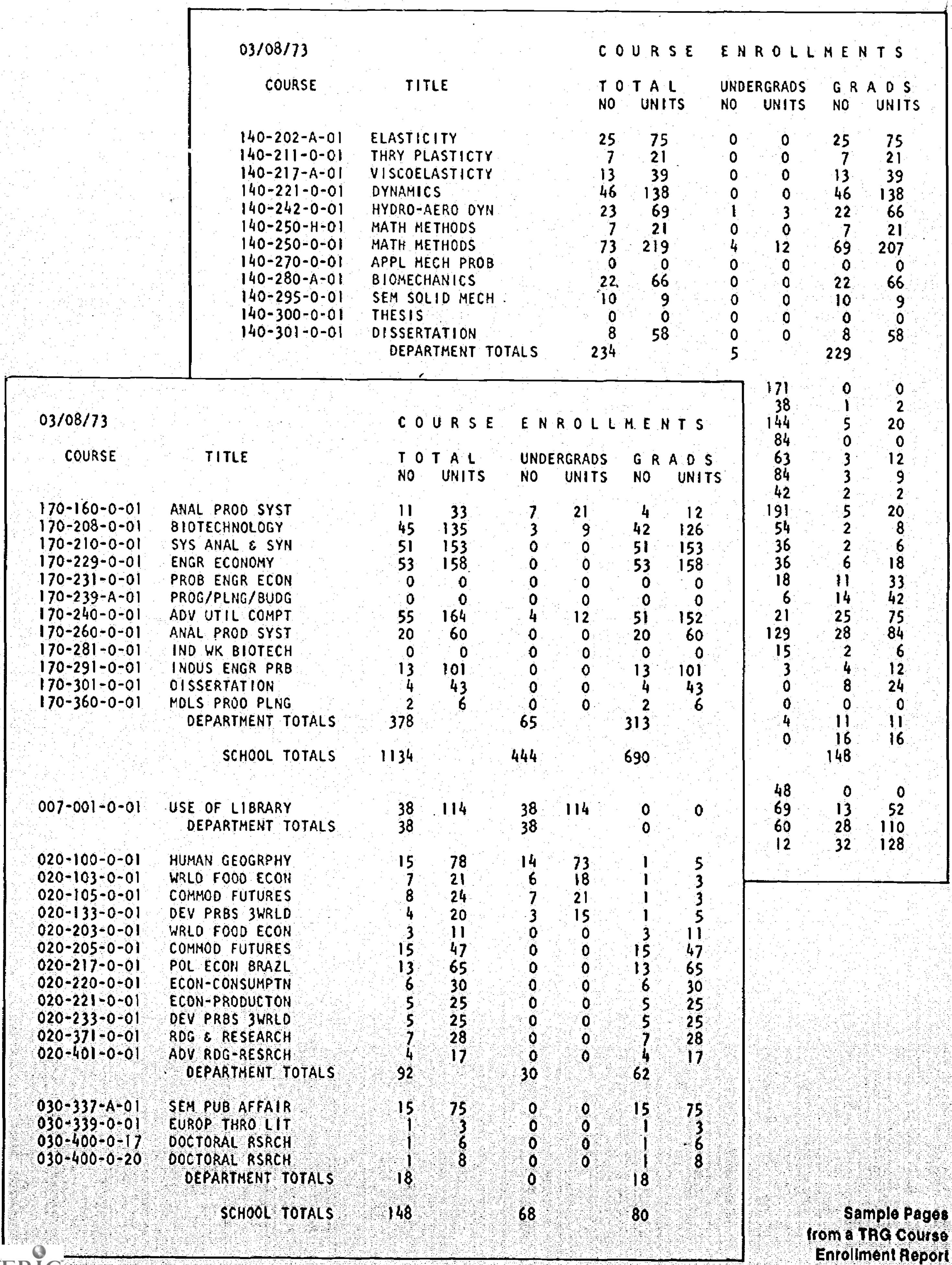
When the user has completed definition of his report, object code is produced and the complete specifications are saved in the OASIS Report File. The user need not, however, stop processing; he may choose to generate a proof of his report or to begin actual report generation.

Ordinarily the user will want to test his report definition to see if it will yield the desired results. Especially for batch reports with complex selection criteria, processing of the full report population would be too costly and time-consuming. The proof option permits the user to specify a simplified selection criteria and to see the report displayed on the terminal, produced online, even though designed for batch production.

The user may begin report generation directly from the DEFINE program or by responding REPORT to the Command Language Processor or by calling REPORTB from the batch card stream. Once the mode and selection criteria have been established, report generation will proceed according to the definition specifications.

Generation of the terminal display, terminal spooled, and batch reports is basically identical except for the execution of the output routine. The write routine employed in the terminal display report is similar to that used in WHERE-clause queries: the report is produced one page at a time; paging is possible; a response of END causes job termination. The spooled report is written onto the logging tape with sorting prefix and line control characters appended. The data are processed straight through and the job terminated when all lines have been produced. Batch programs are provided for sorting out the individual reports and for printing them in batch production. The batch report is produced on a direct line-by-line basis with immediate printout. The job terminates with standard batch end-of-job. Card input provides information as to report name and processing options.

A sample TRG report follows. This particular example involved the definition of a data line and two total lines.



03/08/73 COURSE ENROLLMENTS

COURSE	TITLE	TOTAL NO	TOTAL UNITS	UNDERGRADS NO	UNDERGRADS UNITS	GRADS NO	GRADS UNITS
140-202-A-01	ELASTICITY	25	75	0	0	25	75
140-211-0-01	THRY PLASTICTY	7	21	0	0	7	21
140-217-A-01	VISCOELASTICTY	13	39	0	0	13	39
140-221-0-01	DYNAMICS	46	138	0	0	46	138
140-242-0-01	HYDRO-AERO DYN	23	69	1	3	22	66
140-250-H-01	MATH METHODS	7	21	0	0	7	21
140-250-0-01	MATH METHODS	73	219	4	12	69	207
140-270-0-01	APPL MECH PROB	0	0	0	0	0	0
140-280-A-01	BIOMECHANICS	22	66	0	0	22	66
140-295-0-01	SEM SOLID MECH	10	9	0	0	10	9
140-300-0-01	THESIS	0	0	0	0	0	0
140-301-0-01	DISSERTATION	8	58	0	0	8	58
	DEPARTMENT TOTALS	234		5		229	

171	0	0
38	1	2
144	5	20
84	0	0
63	3	12
84	3	9
42	2	2
191	5	20
54	2	8
36	2	6
36	6	18
18	11	33
6	14	42
21	25	75
129	28	84
15	2	6
3	4	12
0	8	24
0	0	0
4	11	11
0	16	16
	148	
48	0	0
69	13	52
60	28	110
12	32	128

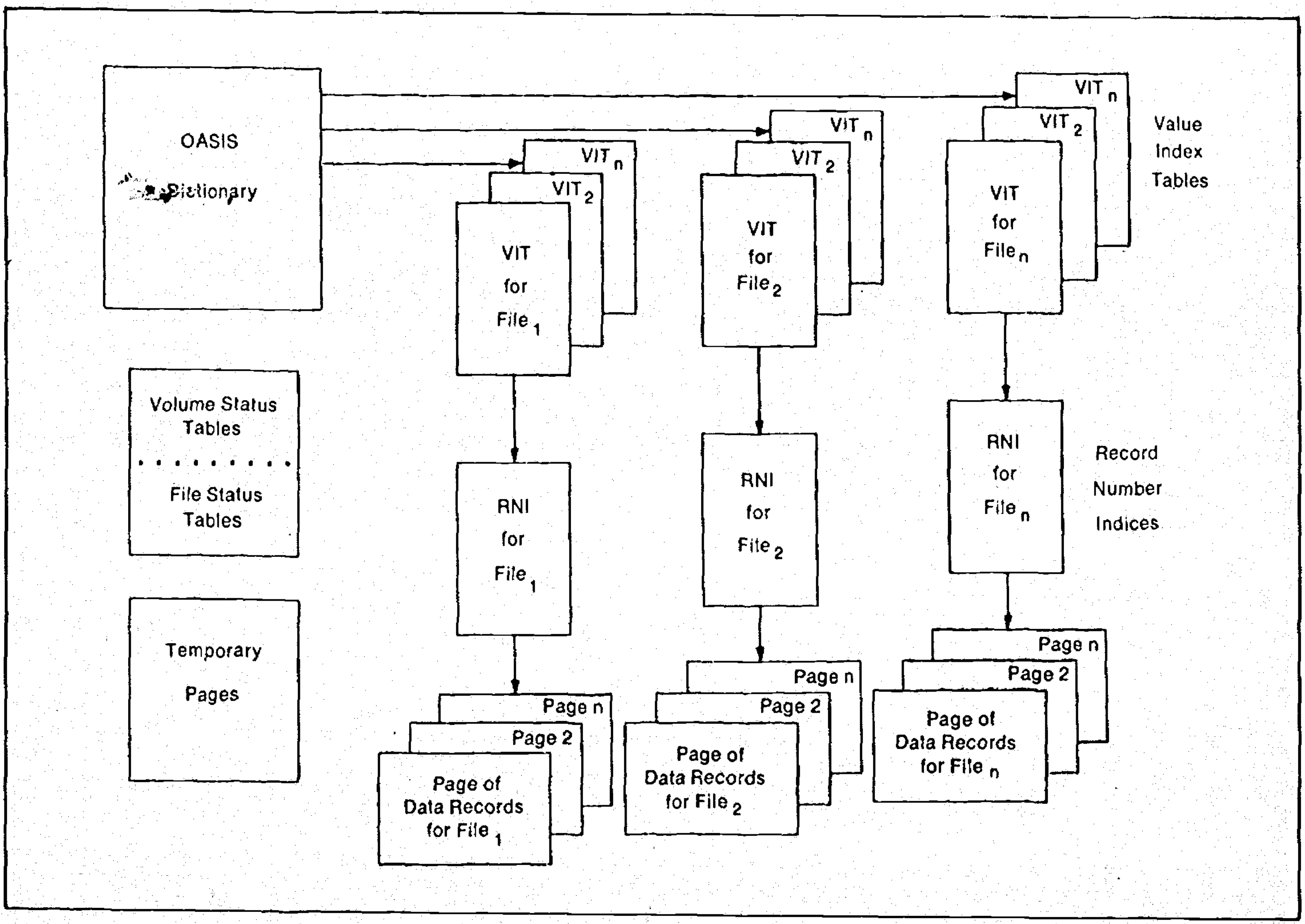
03/08/73 COURSE ENROLLMENTS

COURSE	TITLE	TOTAL NO	TOTAL UNITS	UNDERGRADS NO	UNDERGRADS UNITS	GRADS NO	GRADS UNITS
170-160-0-01	ANAL PROD SYST	11	33	7	21	4	12
170-208-0-01	BIOTECHNOLOGY	45	135	3	9	42	126
170-210-0-01	SYS ANAL & SYN	51	153	0	0	51	153
170-229-0-01	ENGR ECONOMY	53	158	0	0	53	158
170-231-0-01	PROB ENGR ECON	0	0	0	0	0	0
170-239-A-01	PROG/PLNG/BUDG	0	0	0	0	0	0
170-240-0-01	ADV UTIL COMPT	55	164	4	12	51	152
170-260-0-01	ANAL PROD SYST	20	60	0	0	20	60
170-281-0-01	IND WK BIOTECH	0	0	0	0	0	0
170-291-0-01	INDUS ENGR PRB	13	101	0	0	13	101
170-301-0-01	DISSERTATION	4	43	0	0	4	43
170-360-0-01	MDLS PROD PLNG	2	6	0	0	2	6
	DEPARTMENT TOTALS	378		65		313	
	SCHOOL TOTALS	1134		444		690	
007-001-0-01	USE OF LIBRARY	38	114	38	114	0	0
	DEPARTMENT TOTALS	38		38		0	
020-100-0-01	HUMAN GEOGRPHY	15	78	14	73	1	5
020-103-0-01	WRLD FOOD ECON	7	21	6	18	1	3
020-105-0-01	COMMOD FUTURES	8	24	7	21	1	3
020-133-0-01	DEV PRBS 3WRLD	4	20	3	15	1	5
020-203-0-01	WRLD FOOD ECON	3	11	0	0	3	11
020-205-0-01	COMMOD FUTURES	15	47	0	0	15	47
020-217-0-01	POL ECON BRAZL	13	65	0	0	13	65
020-220-0-01	ECON-CONSUMPTN	6	30	0	0	6	30
020-221-0-01	ECON-PRODUCTON	5	25	0	0	5	25
020-233-0-01	DEV PRBS 3WRLD	5	25	0	0	5	25
020-371-0-01	RDG & RESEARCH	7	28	0	0	7	28
020-401-0-01	ADV RDG-RESRCH	4	17	0	0	4	17
	DEPARTMENT TOTALS	92		30		62	
030-337-A-01	SEM PUB AFFAIR	15	75	0	0	15	75
030-339-0-01	EUROP THRO LIT	1	3	0	0	1	3
030-400-0-17	DOCTORAL RSRCH	1	6	0	0	1	6
030-400-0-20	DOCTORAL RSRCH	1	8	0	0	1	8
	DEPARTMENT TOTALS	18		0		18	
	SCHOOL TOTALS	148		68		80	

Sample Pages from a TRG Course Enrollment Report

APPENDIX

OASIS FILE STRUCTURE COMPONENTS



OASIS Data Base Components

OASIS Dictionary—At the heart of File Services processing is a set of entries defining each element, group, indirect reference, file, and password in the Data Base. Information describing the particular item follows each name and is used by File Services to locate and process user data.

Value Index Table (VIT)—Each indexed element in the Data Base has an associated Value Index Table to facilitate retrieval of data by element value specification. A pointer in the OASIS Dictionary locates the beginning of a given VIT. Each VIT is ordered by individual element value according to a modified collating sequence. Each element value is followed by a list of symbolic numbers of records containing the particular value. The VIT's are automatically updated during File Services maintenance routines.

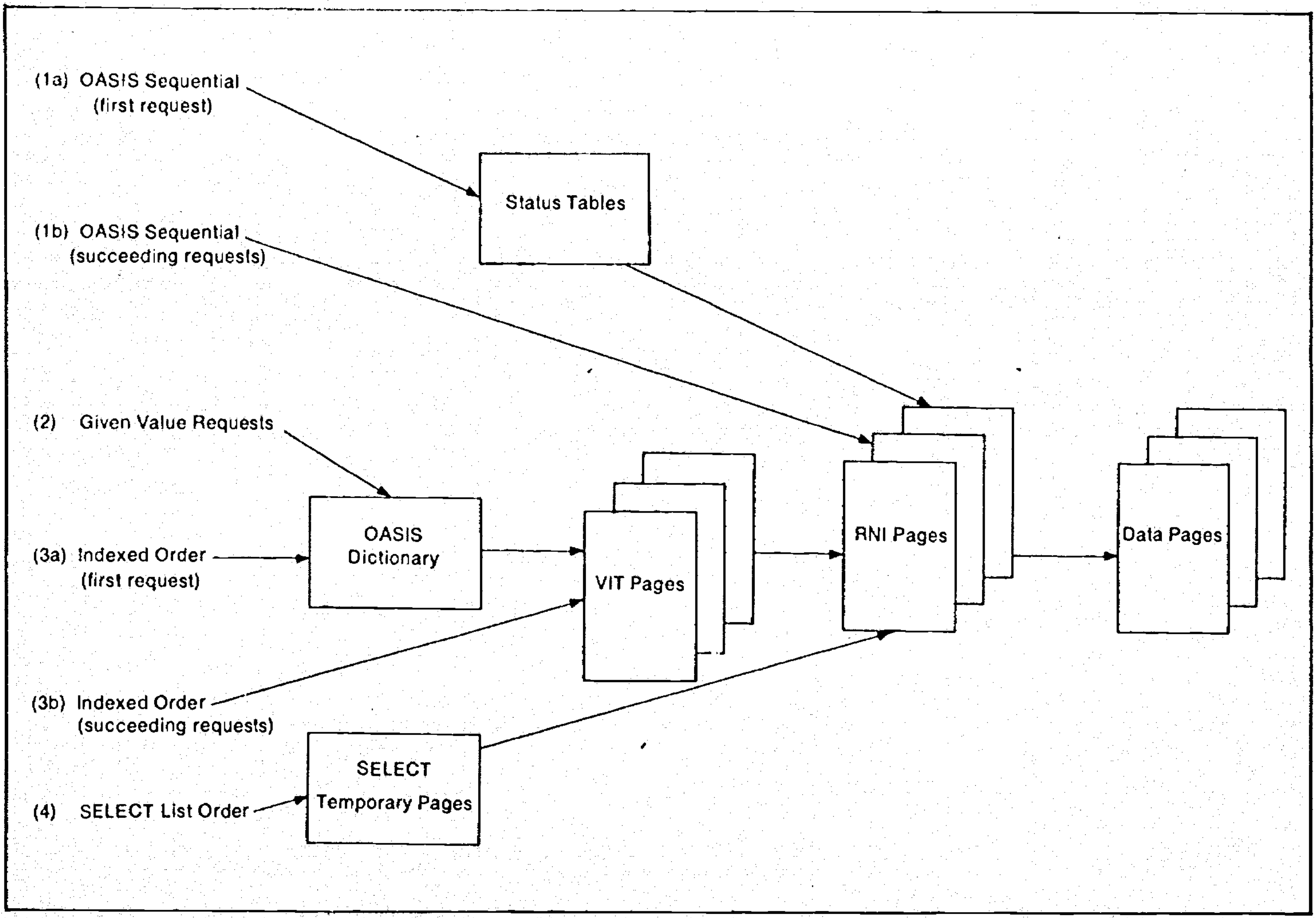
Record Number Index (RNI)—Each record is assigned a symbolic record number when added to the file. The number itself has no significance. However, each file in the Data Base has associated with it a table containing the actual physical locations corresponding to the symbolic record numbers. This table acts as a buffer between the data records and VIT's: relocation of a record during file maintenance requires one [illegible] in the RNI rather than modifications to every VIT for

Data Records—All data for a given record are contiguous, in segment type order. All records for a given file are located together, with unused areas, called 'holes', interspersed to allow for file growth. No specific file order is maintained, although pointers do exist to permit chained sequential processing. Pointers are also maintained within each 'hole' to facilitate chaining through these open areas during file maintenance. The disk space is divided into 1692-byte blocks, called pages; an I/O request brings one page at a time into one of a series of memory buffer areas.

Volume Status Tables—A list of the unassigned areas on disk is maintained in the Volume Status Tables for use in allocating space during file creation, deletion, and maintenance.

File Status Tables—Information concerning the physical status of each file is contained in the File Status Tables and referenced and updated during file maintenance. Included, by file, are the following: last symbolic record number assigned; location of first disk page; location of first 'hole'; chaining sequence of pages for the various VIT's.

Temporary Pages—Certain of the OASIS services, including SELECT and the Generalized Services, use the Temporary Pages file for intermediate storage of data.


(1a) OASIS Sequential (first request)
Status Tables
(1b) OASIS Sequential (succeeding requests)
(2) Given Value Requests
OASIS Dictionary
(3a) Indexed Order (first request)
VIT Pages
RNI Pages
Data Pages
(3b) Indexed Order (succeeding requests)
SELECT Temporary Pages
(4) SELECT List Order

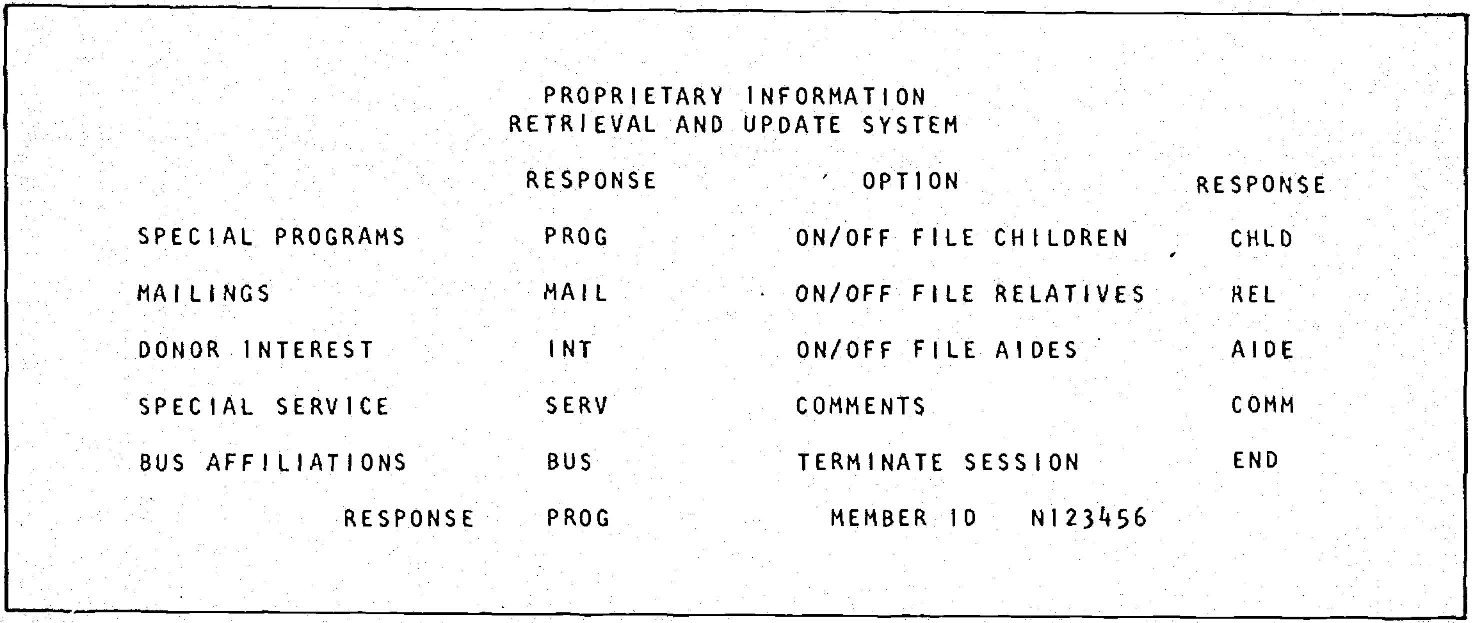
OASIS Data Retrieval Processes

OASIS Sequential. On the first request (1a), the Status Tables are accessed to find the address of the first RNI page for the appropriate file. This step is not necessary on subsequent requests (1b). From the RNI page, the physical address of the data page containing the desired record is obtained and the page can be read in from disk. Since about 400 record addresses are contained on each RNI page, an actual read of an RNI page is usually required only once every 400 requests. A reduction in requirement for actual reads of data pages is also effected by the fact that several records are often contained on each data page of 1692 bytes.

Given Value and Indexed Order. On all Given Value requests (2) and on the first Indexed Order request (3a), the OASIS Dictionary is accessed to find the address of the VIT for the appropriate element. This step is not necessary for subsequent Indexed Order requests (3b). Each VIT page contains lists of RNI's indicating records in which the specific index value may be found. Once these RNI's are found, processing proceeds in the same manner as in Sequential retrieval. As in the case of RNI pages, an actual read of a VIT page is not necessary for all requests, since as many as 540 instances of RNI's may be stored on one page.

SELECT List Order. The SELECT feature, which is used by both the Generalized Services and by user written programs, creates a list of RNI's stored in the Temporary Pages file. Processing by SELECT list order (4) accesses the RNI's in the temporary pages in the same manner as in Sequential retrieval. Approximately 550 RNI's are stored on each temporary page.

TYPICAL OASIS NETWORK DESIGN



```
                       PROPRIETARY INFORMATION
                     RETRIEVAL AND UPDATE SYSTEM

                        RESPONSE           OPTION                    RESPONSE

SPECIAL PROGRAMS        PROG          ON/OFF FILE CHILDREN          CHLD

MAILINGS                MAIL          ON/OFF FILE RELATIVES         REL

DONOR INTEREST          INT           ON/OFF FILE AIDES             AIDE

SPECIAL SERVICE         SERV          COMMENTS                      COMM

BUS AFFILIATIONS        BUS           TERMINATE SESSION             END

            RESPONSE    PROG            MEMBER ID    N123456
```

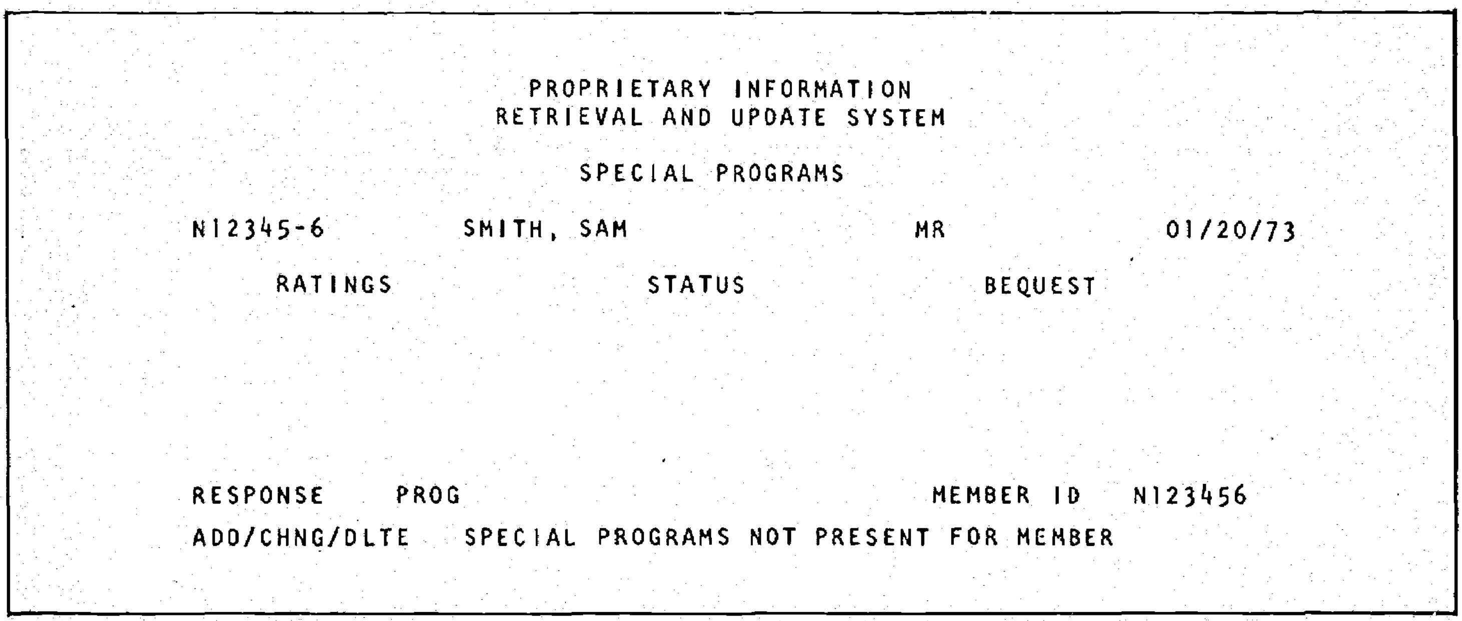
Terminal Display Resulting from Execution of Network Module AA

Unlike programming in a batch environment, online programming under OASIS has a limit of 2048 bytes per module. This concept necessitates a network system design which can perform all the functions of a batch program while maintaining a rigorous modularity. GSO502 is an online retrieval and update network which is used in this appendix to illustrate how these conditions can most effectively be satisfied. Throughout the execution of this network, no more than two levels of 2K modules occupy memory at any given time. In all, there are over 112 modules involved in processing including the root module which is always resident during execution. In OASIS networks, the first module to be linked to an active terminal is known as the root module. It normally contains the data areas used by other modules, processes messages and data from the terminal, and handles routing to other modules. It is sometimes difficult to find sufficient space for the data areas (i.e. COBOL Working-Storage) in the root module. This can be remedied to a major extent by making sure that Working-Storage is redefined as much as possible with common information contained in the first portion of working storage. That not needed for common usage in the network can be defined as one large area, which subsequent modules can use through redefinition for specific needs.

In most terminal networks two types of routines can be identified: those which emphasize common functions needed by all phases of the network and those which are unique to a given series of modules. Networks should be written in such a way that these two kinds of routines are independent of each other. The common routines are those which involve activities such as a GETSEG (retrieve data from file) or TGET (accept response from terminal). Specialized routines, on the other hand, involve processing that is peculiar to a given module or modules, such as editing a user response during an update.

A network designer must be aware of what the terminal user is seeing and responding to. Uncertainty by the user as to what is happening should be minimized through frequent display of information and error messages. Through both retrieval and updating cycles of a network, it is beneficial to inform the user as to the result of information he has requested or entered; i.e. if information requested cannot be found or if invalid input data are rejected, an appropriate message should be displayed to the user. Also, it is helpful during network development to display the result of any error conditions which result from the File Services or Terminal Services calls. The same area that is to be used by the terminal operator when the network becomes operational can be used by programmers during the development phase. OASIS provides a return code area that carries the result of all file requests. If the request is in error it is not necessary to terminate the session at the terminal: simply display the return codes and continue with the next response. This technique will assist the programmer during the debugging phase of a network.

The following material illustrates application of the network concept to a typical function required by the terminal user: the retrieval of a specified segment. Each procedure is called a cycle, and requires loading and execution of a number of modules, which have been assigned two-letter identifiers. A cycle is initiated by a "Send Block" request from the terminal for service, and concludes with the display of new in-



```
                         PROPRIETARY INFORMATION
                       RETRIEVAL AND UPDATE SYSTEM

                            SPECIAL PROGRAMS

      N12345-6        SMITH, SAM                  MR              01/20/73

          RATINGS                  STATUS                BEQUEST




      RESPONSE    PROG                          MEMBER ID   N123456

      ADD/CHNG/DLTE   SPECIAL PROGRAMS NOT PRESENT FOR MEMBER
```

Display Results at Completion of Network Execution Cycle

formation on the terminal screen. The example is from the portion of an Alumni/Gift network dealing with special program information. After each module completes execution, it branches back to the root module and releases its memory space for the next module required.

(1) MODULE-AA: This module performs various initialization and termination routines such as attaching and detaching the user files and displaying the signoff message whenever the end of a terminal session occurs. It begins the retrieval process by displaying the options available to the user. In the display on the previous page, the user wishes to look at special program information and specifies "PROG" plus the identification number of the person sought.

(2) MODULE-AB: This module analyzes the user response and sets up File Services parameters for member ID, which is in segment 1. If an invalid response option is detected, the parameters are set up to display this error condition to the user.

(3) MODULE-AC: This module retrieves segments from the file via the GETSEG (retrieve data from file) call. The parameters for the service are already set up when this module is called. When the last occurrence of a given segment is reached, an indicator is set. Subsequent modules test this indicator and follow the appropriate paths.

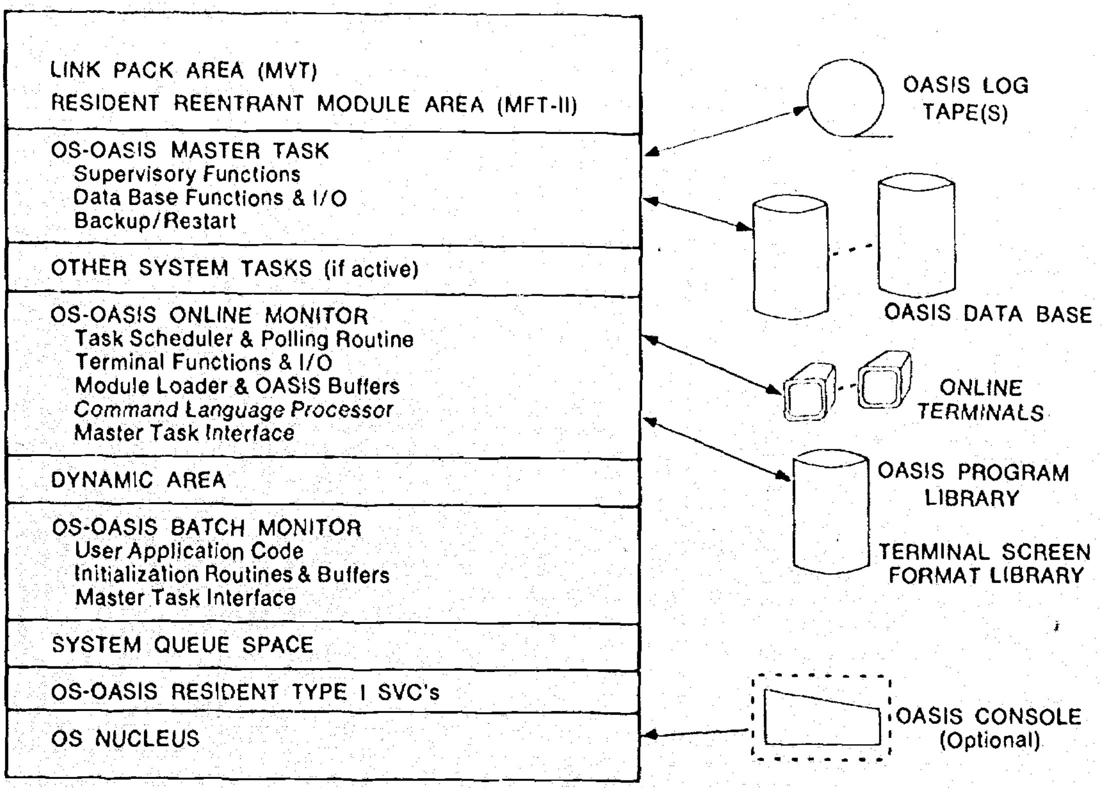
(4) MODULE-AD: This module checks the return code area from the previous GETSEG call. If the requested ID was found ile, the ID, name, and title from segment 1 are formatted in the terminal display area. Once this is done, the correct module is selected to retrieve the segment containing the information requested by the user via his response option; i.e., one of ten module names will be set at this time. If the ID is not found, an error message is issued.

(5) MODULE-BA: Since "PROG" was the option chosen by the user, MODULE-BA is entered. Because special programs (segment 102) can occur only once for a given member, the GETSEG parameters for segment 102 occurrence 1 are set up here. The module-name-save area is set to "BA", since this module must be reentered to check the results of the GETSEG call. Module name is set to "AC" (common GETSEG). If segment 102 is found for this member, the codes in that segment are placed in the display area with their appropriate expansions. If segment 102 is not found, the message "SPECIAL PROGRAMS NOT PRESENT FOR MEMBER" is displayed. In this example, segment 102 is not present for the member.

(6) MODULE-BI: Since this particular network has ten different display formats, MODULE-BI analyzes the response the user has requested and preformats the terminal screen. The completed display is shown above.

(7) MODULE-AH: This module performs all terminal display functions for the retrieval phase of the GSO502 network. The screen format consists of from 1 to 10 blocks of information. Upon exit from this module, the TGET indicator is turned on, because a response from the user will be expected. Exit from the module ends the cycle.

OASIS FUTURE DEVELOPMENT PLAN



Processor Memory Layout for OS-OASIS

OS-OASIS. As a result of expressions of interest from many institutions, an effort was initiated in 1972 to develop a design specification for a version of OASIS that would run under the IBM full Operating System (OS). With assistance from IBM, such a specification was produced, and the actual conversion work is under way at a non-Stanford location. It is not yet known when the completed and tested system will be available. Although the design is considered to be compatible with the newly announced virtual memory Operating Systems (VM, VS1, VS2), further development work will be needed before an informed judgment can be made.

The diagram above shows a general view of the structure of OASIS in an OS (MFT or MVT) environment. Two differences that are immediately evident are the incorporation of an OASIS Master Task, and elimination of the need to share the log tape and Data Base devices between Online OASIS and Batch OASIS. Multiple Console Support, an optional feature of OS, can be used to provide separation of OASIS console functions from OS console functions if desired.

The concept of an OASIS Master Task allows two significant advantages. First, control over the Data Base and the log tape is improved, since their associated physical devices are not shared between problem program partitions. Second, by centralizing OASIS service routines a saving in storage requirements is achieved since these routines appear only once in storage instead of once for Online OASIS and once ch OASIS. The OS-OASIS Master Task is designed ate as an OS System Task—that is, a privileged program that operates as though it were a part of OS (similar to an OS Reader/Interpreter, Writer or Initiator).

The OS-OASIS Online Monitor operates as a conventional OS job and incorporates some special facilities. The functions it provides include terminal polling, terminal I/O, user program loading, and scheduling of OASIS terminal tasks. In addition, the Online Monitor manipulates storage and PSW protect keys (by use of a user written SVC) to implement the OASIS storage protection feature. Another user written SVC is used to notify the Master Task of a request for service from the Online Monitor or from a Batch Monitor.

The OS-OASIS Batch Monitor is designed to provide an interface between batch applications and the Master Task. It contains entry points used to resolve each service CALL that can be made from a batch application program.

IBM CRT Terminals. OASIS currently allows the attachment of only the Sanders 720 series CRT terminals. The OASIS Generalized Services and Terminal Services make use of a number of hardware features of this terminal that are not available in other vendors' equipment. Advancements in terminal capability since the OASIS design was frozen, particularly the IBM 3270 series CRT's, have made it desirable to expand the range of terminals that may be used. The OASIS development team at Stanford is currently designing support for the IBM terminals and expects to complete this work by the end of 1973.

OASIS COMMANDS AND RESERVED WORDS

CLP Commands

ATTACH
BREAK
CLEAR
CLOCK
CONTINUE
  CONT
COUNT
DAYMSG
DEFINE
DETACH
DUMP CORE
END FOASIS
EXECUTE
  EXEC
HOLD
LIST
LOAD
LOGOFF
LOGON
MSG
MSGR
NEWVOL
NOHOLD
PATCH CORE
PATCH GPR
PATCH IA
QUERY
REPORT
RESET
SENDMSG
SHOW CORE
SHOW GPR
SHOW MAP
STATUS
TEST
TRACE

File Services Commands

ADDSEG
ATTACH
CRTONAME
DELSEG
DESCRIBE
DETACH
FREEALL
FREELIST
GETDATA
GETSEG
GETVAL
NAMETOCR
RPLDATA
RPLSEG
SELECT
TAPEWRIT

Terminal Services Commands

PTOPL
RTFPL
TASKNUMB
TERASE
TGET
TPAGE
TPGPUT
TPOLL
TPUT
TRESTORE
TSAVE

OASIS Reserved Words*

A
ALL
AND
AVG
COLUM
COLUMNAR
COUNT
D
EQ
FILES
GE
GR
HELP
IGNORE
LE
LS
MAX
MIN
MORE
N
NE
NONE
NULL
OR
ORDERED
RECORD
RECORDS
RN
SUM
TAND
VERT
VERTICAL
WHERE
Y

* Not to be used as data names.