UNIVERSITY OF EDINBURGH

Edinburgh Regional Computing Centre

EMAS 2900 : Concepts

A brief introduction to the design philosophy of the multi-access operating system EMAS 2900.

1st Edition
May, 1978

EDINBURGH REGIONAL COMPUTING CENTRE

EMAS 2900: CONCEPTS

A brief introduction to the design philosophy
of the multi-access operating system EMAS 2900

1st Edition
May, 1978

EMAS 2900 CONCEPTS AND FACILITIES

CONTENTS

CHAPTER 1: HISTORY AND OBJECTIVES

History

Between 1966 and 1970 a large University/Manufacturer team attempted to write a multi-access operating system for the most powerful computer in the English Electric Leo Marconi range, the System 4-75. This project, the Edinburgh Multi Access Project (EMAP), which was organised on conventional lines with managers, team leaders, designers and programmers, together with a selection of technical and coordinating committees, failed to implement a system. Sufficient code was working at the end of the project to give an idea of the characteristics of the system: EMAP would have supported a small number of users (8-10) with poor response and reliability. The reason was that the system was far too big, and little room was available in the machine for user programs. Also, in spite of the size of the code, the monitor and diagnostic features were poor or nonexistent; consequently, "bugs" were not susceptible to analysis and could only be found by heavenly inspiration.

Thus in 1970 Edinburgh University was left with a lot of hard earned experience and a System 4-75, but no operating system. A small team varying between 3 and 7 set out to implement a system to the original specification (minus certain "marketing" additions), salvaging anything relevant from EMAP and writing the rest. Within six months a pilot system was running and in a year it was the prime service vehicle. In two years EMAS (Edinburgh Multi-Access System) was setting new standards of facilities and reliability among the System 4 Universities; the lessons of the EMAP project had been learned.

Eight years after the delivery of the 4-75 the University received an ICL 2970 to augment its computing capacity and start the phased replacement of old equipment. This machine came with a large virtual memory operating system with all the design aims of EMAP and more besides; it also came with a strict injunction from the Computer Board that the manufacturer's operating system must be used. This system supported a small (1-2) number of users with poor response and reliability. The reason was excessive size and complexity of system software together with lack of diagnostic aids and of facilities for system recompilation. The University mounted a major effort to improve the performance of the system throughput which produced, among other items, three compilers (ALGOL(E), FORTRAN(G) and IMP), the Scientific Jobber and the Scottish Network Augmentation Package (SNAP). After eighteen months the University effort expended to improve the manufacturer's operating system had exceeded the effort involved in writing EMAS, yet there was still a vast chasm between EMAS and the system that was intended to replace it.

At this stage it was clear that the most economical way of obtaining a suitable operating system was to write one. The new system, EMAS 2900, was to replicate the system interface of EMAS as closely as possible, to allow the subsystems and application packages to move with minimum effort. At the same time certain internal changes were introduced in the light of experience. The most difficult and extensive changes were in the area of communications. None of the items marketed by ICL as communications hardware was suitable for the demanding task of EMAS Front End Processor (FEP). A PDP 11/34 was obtained and attached to a New Range Peripheral Interface via local hardware. Once this was operational progress was rapid and by January 1978 EMAS 2900 could support 30 terminals on a 1 megabyte 2970.

Objectives

EMAS is a system dedicated to terminal access. An on-line terminal system is clearly more convenient than a batch system for the user and should be more economical in computer time since the user can intervene to abort runaway computations, prevent large amounts of repetitive output from being printed, and generally use computer time with more precision than can the remote batch user.

In practice, however, when interactive terminals are added to a paged operating system, much less effective use is made of computer time, and management is left with the unenviable choice of either severely restricting terminal access or purchasing more hardware. The EMAS designers have made a study of this problem and found the following reasons why batch systems perform poorly in interactive mode:

a) Too much software resident

In a batch environment a comparatively small amount of space can support enough user work to saturate the CPU - it does not matter if a large proportion of main store is occupied by system software. In an environment where most jobs execute for a short period before stopping to interact with the terminal, it is important to slim down the system software to enable many user jobs to be resident simultaneously, a necessity if the CPU is to be fully utilized. Elephantiasis of the operating system is fatal.

b) Core occupancy by data base and buffer areas

All systems need a data base for each job running, to describe the virtual memory and to hold other relevant information. In a batch system, where a small number of jobs run to completion, it is obvious to make the area - frequently many kilobytes in size - resident. Batch systems also have a buffer pool for real devices and tend to allocate interactive device buffers from this pool. This means that a user who is scratching his head at his terminal or answering the telephone may be occupying several pages of main store while requiring no services.

c) Program delay by channel contention

It often happens that a user who is trying to run has difficulty getting at his data files because the data paths to the discs are occupied in paging out another user who has just entered a wait state.

d) Hidden loss of main store during paging

Most systems let user programs demand page their working set. While a program is demanding page N there are (N-1) pages occupying part of main store, which is therefore not available for other pages. In a batch system the time taken to acquire the working set is small compared with execution time and this loss is unimportant. However an interactive program will have to re-acquire its working set following each occasion that it has had to wait for input from the terminal, and the time taken to do this can be much longer than its execution time.

EMAS 2900 is designed to provide cheap interaction on ICL 2900 (P series) architecture, and makes its prime aim that of solving the problems posed by a) - d) above. Amongst its features are

1) Unification of secondary storage (for paging) and file storage. All secondary storage is in files and all file access is via the virtual memory. This arrangement is sometimes called "virtual files" or "mapped files"; it minimises paging software.

2) A three-part Supervisor, consisting of

   a) A minimal resident Global Controller.

   b) A Local Controller for each process; the process data base is in the Local Controller data space and is paged with the process.

   c) A paged supervisor (called Director) to provide non-time-critical user functions. Director is described in Chapters 6 and 7.

3) A specialised Communications Controller which arranges all terminal and RJE traffic. It also handles local card readers and line printers. Transfers are done directly into virtual memory using the standard paging routines - no resident buffering is required.

4) A four-level memory hierarchy, consisting of

   Archive store - tape

   Disc store

   Drum store (desirable but not essential)

   Main store

   The last three levels are managed by the Supervisor in a way that is transparent to the user.

5) Retention of working sets

   The Local Controller attempts to maintain the working set between terminal interactions and then preload it when the wait state is left. This reduces the cost of providing good response to repetitive actions like text editing or data vetting.

6) Specialist device drivers

   The principal paging device is handled by a specialist routine. This ignores the hardware queueing and maintains priority order queues. Transfers are scheduled based on demand pages; page out requests and lower priority pre-page requests are then used to occupy sector windows that would otherwise be unused. This reduces the cost of pre-page transfers and prevents them causing additional channel congestion.

7) No catalogue contention

EMAS has no system file catalogue. Instead each accredited user has his own catalogue, which is a standard file (his "File Index"). His Director can arrange access to other File Indexes for the sharing of material, provided suitable permissions have been given. System software is arranged in one or more "MANAGR" processes (see Chapter 2), with the material distributed as the System Manager thinks fit. Catalogue contention can never become a limitation in EMAS.

CHAPTER 2: SYSTEM APPEARANCE AND STRUCTURE

Appearance

Before a potential user can access EMAS 2900 he has to be accredited to the System by the System Manager, who has a special paged process (the MANAGR process) to provide this and other management facilities. Once the user is accredited, EMAS 2900 provides him with four basic facilities:

1) A virtual memory, or addressing space, of 32 megabytes.

2) File storage space which the user can manipulate freely, subject to limits set by the System Manager at accrediting time.

3) A virtual processor which can execute non-privileged instructions from the 2900 order code at a protection (ACR) level determined by the System Manager.

4) A procedural interface (the Director interface) which provides facilities for the user to manipulate his files and virtual memory. It also allows him to communicate with other processes on the System. The most important of these are the Spooling process for slow device input and output, and the Communications Controller for terminal operations.

It should be noted that the EMAS System itself does not provide any user-orientated features such as command interpreters, editors, compilers, loaders or diagnostic aids. The user has to write, or otherwise acquire, a suitable set of such facilities to provide himself with an interface to those System features which are relevant to his problems. Such a collection of facilities is known as a "Subsystem". The Edinburgh Subsystem (Chapter 9) is only one of many possible Subsystems and, like any Subsystem, allows only a selection of the System features to be used.

Structure

EMAS 2900 is organised as a collection of processes which communicate by means of messages passing between them. The structure can therefore be viewed as in Figure 1, which shows a central message switching facility with message paths radiating to the processes.

Processes fall into two classes. First, "Supervisor processes", resident in main store and implemented in the form of IMP routines, which control primarily resource allocation and peripheral device driving. Secondly, "paged processes", running in the virtual processors mentioned above; these are created dynamically, for instance when a user logs in. Certain paged processes perform Supervisor functions which are not time-critical. These are the "paged System processes", described in Chapter 8.

Each paged process is associated with a separate incarnation of the Supervisor process known as the "Local Controller", and each incarnation operates on behalf of that paged process alone. Its main functions are to procure resources for the paged process and to deal with page faults, program errors, etc.

"Global Controller" is the name given to a collection of Supervisor processes which allocate resources such as CPU, main store and "active" file space to Local Controllers, for the use of their associated paged processes.

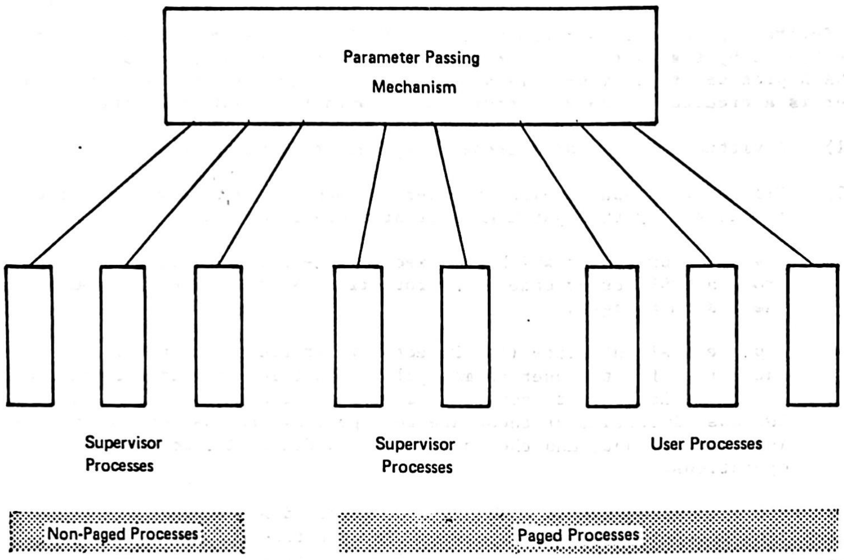


Figure 1: Parameter passing between processes

All user contact is with "Director" routines; this procedural interface hides the Local and Global Controllers from the user. The structure of the outer components of the System is shown diagrammatically in Figure 2.

Director (Chapter 6) provides a set of routines for terminal I/O and a set of routines to allow the user to create, destroy and manipulate files in the file store. The only method of file access allowed on EMAS is "connection": when Director is asked to connect a file it associates it with a free area of virtual memory and reports its size and address. Accessing the addresses accesses the file directly - the file and the address space are one and the same thing. For languages designed with this sort of storage in mind (e.g. PL/1's "based" storage or IMP's "mapped" variables) file access is extraordinarily efficient. For more primitive languages it is trivial to interpose a routine to pretend that the file is a sequence of 80-byte card images - or whatever is required. Similarly the more complicated access methods can be arranged provided the file can fit into the address space. For files too large for the address space it is possible to connect the file a section at a time; this means that access is less convenient and efficient except for strictly sequential reading and writing.

Access to the Director routines is via the 2900 "System Call" feature. This has a built-in ACR check so that the System Manager can vary the available facilities, on an individual user basis, by selecting the ACR level at which a user's process runs. This is invaluable in a teaching environment.

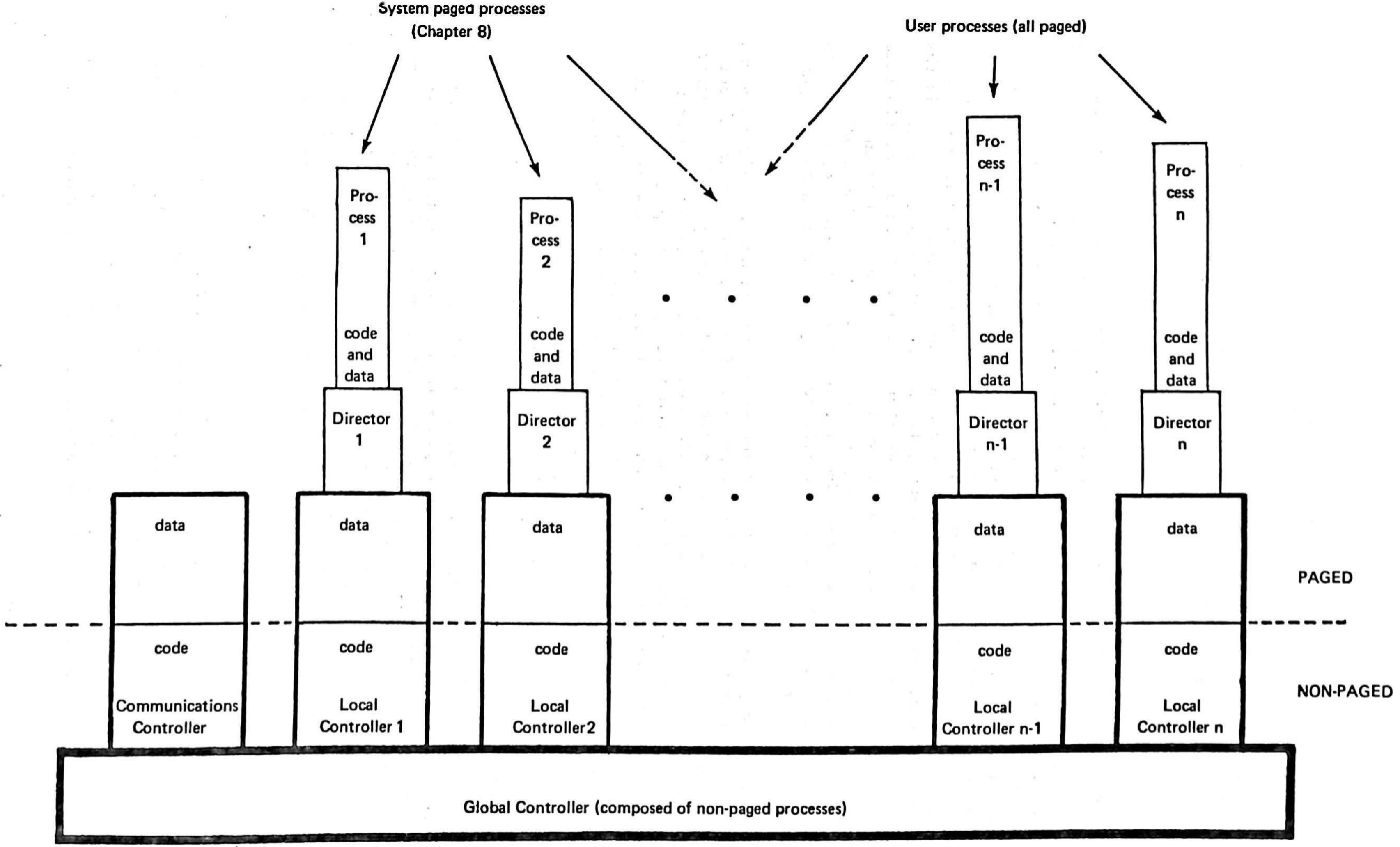


Figure 2: EMAS 2900 System structure above the Global Controller

CHAPTER 3: THE GLOBAL CONTROLLER

The Global Controller consists of a number of Supervisor processes, the most important of which are the "Scheduler", the "Active Memory Manager", the "Paging Manager" and the device drivers.

The Scheduler

The Scheduler allocates CPU time and main store to Local Controllers. It does this on a democratic basis so that requests for the allocation of small amounts will be satisfied sooner than requests for large amounts. This has the effect of ensuring good response for small tasks, for example interactive text editing, while larger tasks may have to wait longer. In compensation, however, the latter get a good bite at the cherry when they get their allocation. The scheme is based upon a set of categories, each defining varying combinations of CPU time and main store allocations. As processes change their characteristics the Scheduler automatically finds the most suitable category. For instance, if the Scheduler finds that the process is only using a small amount of store and a small amount of CPU time between console interactions, then it will assign the process to an appropriate small CPU/small store category. Thereafter, if the process starts to use more store and more CPU time in order, for example, to compile an edited source file, the Scheduler will reassign the process to a larger CPU/larger store category.

Each category has a priority, varying from high for small CPU/small store to low for large CPU/large store categories. Processes waiting to run but without a CPU time or store allocation are queued according to the priority of their current category. The Scheduler selects processes to load from higher priority queues more often than from lower priority queues. The scheme is not pre-emptive, however, and low priority processes cannot get held up indefinitely, even if the System is very busy. The net effect of the scheme, when the System is busy, is to vary the apparent speed of a process's virtual CPU inversely as its core requirement. Thus a program whose working set is 100 Kbytes will find that its virtual CPU has an instruction rate about twice that of a program whose working set is 200 Kbytes. This is true for all processes regardless of their degree of privilege.

Whenever main store becomes available, the Scheduler attempts to load further processes. Loading may take place in up to three stages depending on how much store is available. First, the pages corresponding to the Local Controller stack are brought into store; secondly, an allocation of store pages is made, corresponding to the size of the process's "working set" of pages; and finally an allocation is made to complete the requirements of the category of the process. No more processes are loaded until this final allocation has been completed. This avoids the problem of "thrashing".

The Active Memory Manager

The Active Memory Manager controls and organises those portions of file space which are currently "active", i.e. in use. This is a subset of the file space which is currently "connected" (see Chapter 6). It corresponds closely to the space available on drum storage (for those installations which have drums). The use of drum storage very much improves the ability of the System to respond quickly to requests for access to pages. Areas of file space, i.e. disc storage areas, are mapped into Active Memory Table areas in the Global Controller upon request from a Local Controller. All requests for paging activity thereafter from Local Controllers are made in terms of the Active Memory Table indexes. Simultaneous sharing of files between processes is achieved by this central facility. Each Local Controller is given a limit to the amount of active memory space it may request. As the active memory space becomes full this limit is reduced, and as it empties the limit is increased again.

The Paging Manager

The Paging Manager is a global service provided for Local Controllers to use. Its functions are to bring pages into store and to remove them when they are no longer required. As this service is part of the Global Controller, it can take account of sharing of pages in main store. In other words, if a previous Local Controller has already requested a certain page, the next Local Controller to ask for it can be told that it is already in main store. This phenomenon occurs very frequently, particularly for editor and compiler pages, and helps to make good use of the available main store.

When a page in store is finished with, it will either be paged out to disc if it was written to whilst in store, or just discarded if it was only read. The drum is used as a "cache" in that it always duplicates the (active) information on disc. This is achieved by writing to both the disc and the drum when page-outs are performed.

The Device Drivers

The device drivers are processes provided to control the principal rotating and general peripherals. Their differing structures reflect the importance of the peripherals to the System.

The Drum Driver drives the Sector File Controllers (SFC) as paging devices; this is fundamental to the Supervisor and no provision is made for non-Supervisor use. The Drum Driver appropriates all SFCs and Drums at IPL time and allocates appropriate communication and queueing areas. Queueing is a software function since it is necessary to distinguish priority (i.e. demand pages) and non-priority transfers. The aim of the Drum Driver is to minimise the time for priority transfers rather than maximise throughput. The Drum Driver is equipped with sophisticated error recovery and the ability to discard doubtful tracks or, in extremis, doubtful files.

The GPC Driver is a much simpler piece of software, since slow peripherals are not central to the System. The GPC Driver allows peripherals to be claimed by other (usually paged) processes, and provides simple facilities for executing command chains constructed by those processes. It is thus possible to perform on-line testing of GPC devices, and for paged processes to drive new and/or unsupported peripherals.

The Disc Driver is intermediate in complexity between the GPC and Drum Drivers. All EMAS 2900 file system discs are appropriated by it at IPL time and driven on behalf of the System. Other discs are available for processes to use at the simple command chain level, for testing, diagnostic and other purposes. The file system discs are driven to obtain maximum throughput using all available hardware facilities. Queueing is arranged by software to minimise head movement and although some priority is given to demand page requests this is much less dramatic than in the case of the Drum Driver.

The Disc Driver and Director co-operate to note any bad or suspect tracks on an EMAS file system disc. When space for user files is allocated, known bad tracks are always avoided and suspect ones are avoided if possible. This results in a reliable and robust file system.

CHAPTER 4: THE LOCAL CONTROLLER

Each paged process is controlled by a separate incarnation of the Local Controller, whose functions are to manage the virtual processor in which the process runs and to provide various services to the process. Each incarnation is distinguished by having its own stack of working storage, which is paged in and out as the process is loaded and unloaded. The code is shared by all Local Controllers and is permanently resident in main store (compiled as a subroutine of the Global Controller).

The mapping of file space into virtual memory ("connection") is done by Director: it writes the mapping information into data structures on the Local Controller stack, to which it alone has access under APF level privilege. These include the Local Segment Table, which is used by the address translation hardware and which resides at the bottom of the Local Controller stack pages. When a page fault occurs, that is, when a required page of virtual memory is not present in main store, the Local Controller consults the mapping information and its own Local Active Memory Tables in order to locate the missing page on backing store. The Global Controller is then called to bring the page into store.

The Local Controller ensures that the process remains within its current allocation of CPU time, main store and active memory space. If the process runs out of CPU time or needs more main store, it asks the Scheduler for more, but unless the System is lightly loaded this will involve the process being paged out of store and put on a priority queue for the new resources it needs. If the process needs access to a new active memory area which would take it over its active memory space limit, the Local Controller releases some of the space least recently used by the process before asking for the new space.

The main store allocation is in terms of a number of pages, and the CPU time allocation is in terms of "time-slices". A time-slice is the largest unit of time for which a single process is allowed to occupy the CPU without interruption. After that time, the process must temporarily relinquish the CPU to another process. Processes which are in store and are waiting for a CPU time-slice take turns at the CPU in round-robin fashion; this again is in an effort to ensure good response to small interactive processes. Time-slicing is achieved by the Local Controller setting the Local Interval Timer Register to cause an interrupt after the appropriate time, and then putting itself on the back of the round-robin queue when the interrupt occurs. This illustrates a convenient feature of the 2900 architecture: interrupts can be taken directly into the Local Controller of the process currently on the CPU. This is done for Interval Timer interrupts and also for page fault and OUT (Supervisor Call) interrupts.

The OUT mechanism is used to provide services for the process: these handle console communications in part, disconnection of files, inter-process message passing and the manipulation of alternate stacks, together with a number of other items. An important alternate stack is that used by the Director "Signal" mechanism: asynchronous messages for the process, such as single-character interrupts from the user's console (see Chapter 9), cause the Local Controller to switch to this stack and to activate the Signal context. The previous context is preserved while performing Signal activity.

CHAPTER 5: COMMUNICATONS

As the structure of the communications software in EMAS 2900 is rather novel, a summary of the design aims is given. Thereafter the actual approach adopted is described, followed by a description of the software components involved.

Design Aims

All communications traffic (i.e. interactive, inter-machine, RJE and transaction) should be transported between a Front End Processor and the ICL 2900 mainframe using a common logical interface, independent of the make, model or number of Front End Processors or the physical interfaces by which they are attached to the mainframe.

Within an ICL 2900 mainframe system there should exist a single common communications software interface. This requirement is made in order to avoid the use of different internal transport mechanisms for each type of communications traffic, and so to minimise the amount of software required. This does not necessarily imply a single user interface for all communications traffic.

To make the handling of communications efficient there are two principal requirements:

* The effects of the inherent data unit mismatch between variable length messages, which are the unit of transfer between Front End Processors and the mainframe, and pages, which are the unit of storage allocation and transfer within the mainframe, should be minimized.

* A paged virtual process should be invoked to handle communications traffic only when enough information has been transferred to trigger computational activity in the virtual process; they should not be invoked for book-keeping operations. This means that an interactive process should be invoked only upon the reception of a complete input message for which it has been waiting, but not upon the reception of message fragments. Likewise, it implies that the Spool Manager should be invoked only upon the completion of a file transfer rather than upon the transfer of a block.

Approach

The critical new feature in EMAS 2900 is that communications traffic is transferred directly between non-paged devices (Front Ends, card readers, line printers, etc) and files, rather than between these devices and real memory buffers.

In the case of interactive traffic, the files involved are also connected into the appropriate user's paged virtual memory, in communications mode. Input is written by a Front End Processor into a file, which is connected in communications input mode, and is read from that file by programs running in the associated user's virtual process. In a similar fashion, output is written within a user's virtual process to a file connected in communications output mode, and is read from this file by the associated Front End Processor.

In the case of RJE traffic, a sequence of spool files is associated with each RJE device stream by the System's Spooling Manager (see Chapter 8).

For I/O transfers to take place to or from a file, the relevant pages of that file must be resident in main memory. This does not imply that these pages should be permanently resident in main memory, only that they should be resident for the duration of the data transfer. As the time required for a data transfer is small and the period between transfers is large compared to page transfer times, the pages containing or receiving the data are transferred to main memory from secondary (drum or disk) storage only as required. These transfers are performed under the control of a special paging manager called the Communications Controller.

In most existing computer systems, the inclusion of a special paging manager for this purpose would be difficult, as such systems possess only a single paging manager to control the use of all memory resources. It is not a problem in EMAS 2900, in which multiple paging managers (called Local Controllers) are provided on a one-to-one basis for each virtual process in the System. The Communications Controller is merely a special form of Local Controller; a special form is required because 2900 peripheral devices do not generate page fault interrupts on missing pages. The Communications Controller must generate these interrupts by software and await the arrival of the pages before allowing the transfer to start.

In addition to data transfers to or from devices, there is a requirement for passing control information between 'Device Servers' and the user. (A Device Server is the module that controls the device to which the user is connected.) In the case of local peripherals such as line printers and card readers, the Device Servers in each case are embedded in the Interface Adaptor Module for that device. In the case of devices connected through a Front End Processor, such as interactive or RJE terminals, the Device Servers reside in the Front End Processor itself. The identity and format of this control information need only be agreed between a user and a Device Server. Control information is therefore transparent to the Communications Controller, and the introduction of a new interface or device only requires the introduction of a new Adaptor Module - no other changes to the System are required.

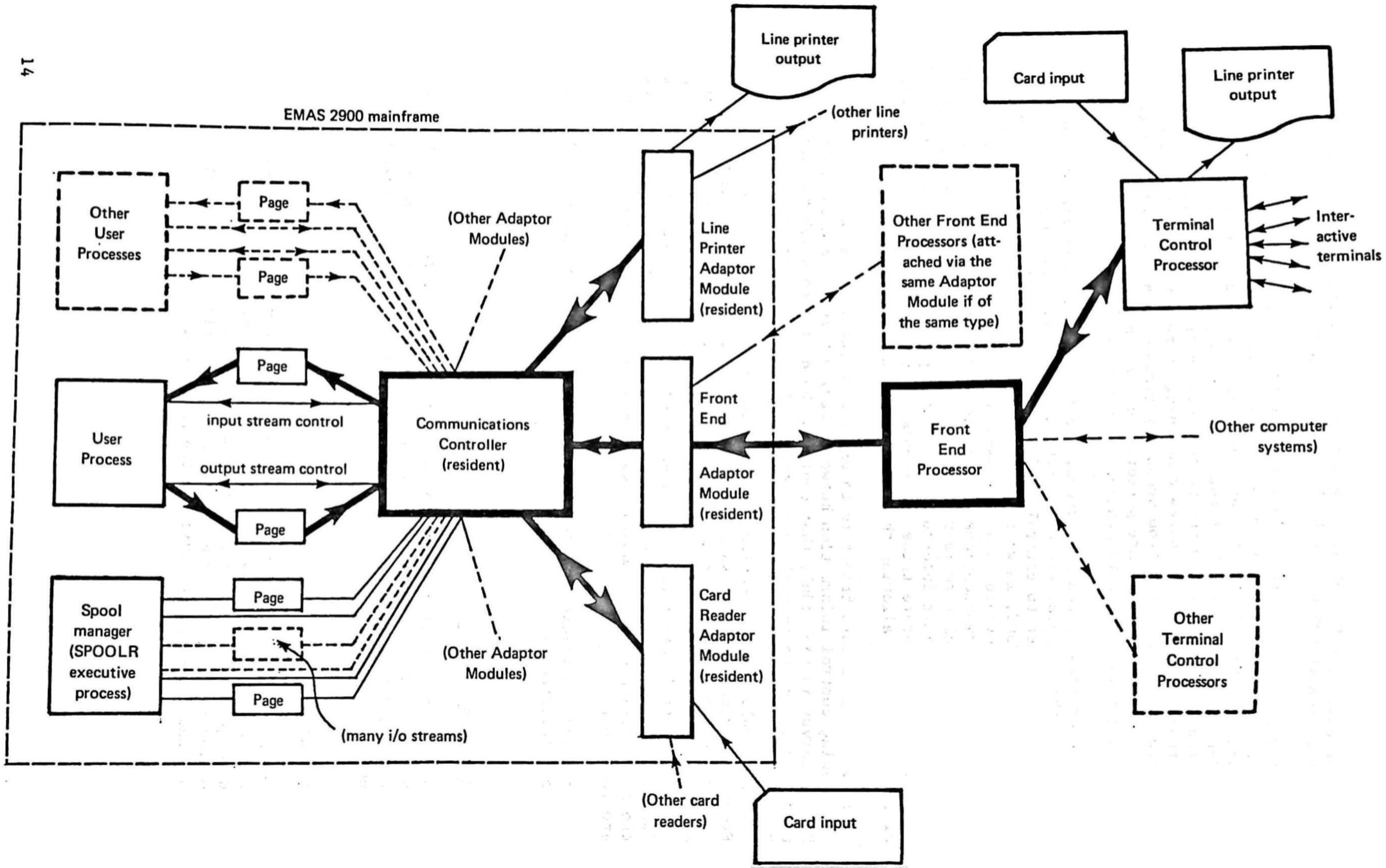
The structure resulting from this approach is shown diagrammatically in Figure 3 (overleaf).

Software Architecture

The preceding section described the approach adopted for the handling of communications traffic within EMAS 2900. In this section the software modules that mechanize the handling of this traffic within the mainframe are examined.

Interface Adaptor Modules

These modules mechanize control and data transfers between Front End Processors or local slow peripherals (card readers, line printers, etc) and communications streams. The identity of an Adaptor Module depends upon the identity of the physical interface employed to connect each Front End Processor or device to the mainframe.


EMAS 2900 mainframe
Line printer output
(other line printers)
Card input
Line printer output
Other User Processes
Page
Page
(Other Adaptor Modules)
Line Printer Adaptor Module (resident)
Other Front End Processors (attached via the same Adaptor Module if of the same type)
Terminal Control Processor
Interactive terminals
Page
User Process
input stream control
output stream control
Page
Communications Controller (resident)
Front End Adaptor Module (resident)
Front End Processor
(Other computer systems)
Spool manager (SPOOLR executive process)
Page
Page
(many i/o streams)
(Other Adaptor Modules)
Card Reader Adaptor Module (resident)
Other Terminal Control Processors
(Other card readers)
Card input

The Communications Controller

This module mechanizes communications streams and handles page faults on behalf of Interface Adaptor Modules. There is only one Communications Controller in the System.

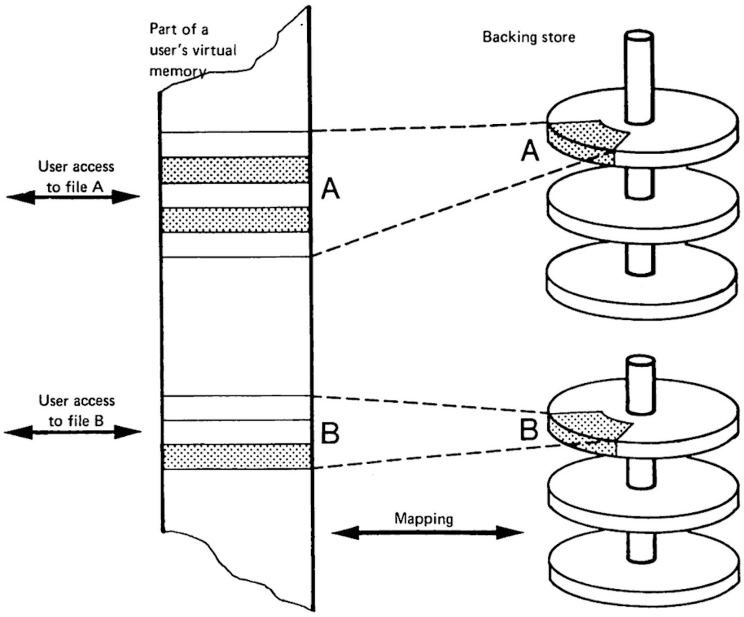
Each stream is identified by a sixteen-bit stream address, with even addresses identifying input streams and odd addresses output streams. A Front End will usually have a number of streams with contiguous addresses assigned to it, while a line printer, for example, has only one. All streams operate in simplex mode with data transfers taking place in only one direction, though control information may pass in both directions. If a duplex stream is required, as for example with interactive traffic, it is constructed from two consecutive streams (input stream "N" and output stream "N+1"). Stream clusters required to support RJE terminals may be similarly configured from multiple streams.

Interactive Interface Modules

These modules are part of the Director of each virtual process. They support the System's interactive communications interface by fielding users' interactive I/O commands, and generating the appropriate calls on the Communications Controller and attached Interactive Terminal Server.

The Spooling Manager

This is an executive virtual process (more fully described in Chapter 8) that accepts output files from user processes, and generates the appropriate calls on the Communications Controller to transfer the contents to a local or remote line printer, card punch, paper tape punch or other output device. The Spooling Manager also generates new files into which jobs or files from local or remote devices (card readers, paper tape readers, etc) are read under the control of the Communications Controller, before being transferred to the user process in which they are to be stored or executed.



Files A and B are connected into the user's virtual memory. The user accesses them as though they were part of contiguous main store – it is one of the functions of EMAS 2900 to sustain this fiction.

Each file is divided into fixed-size pages. A page of a file is only brought into "real" main store if it is referenced by a program or if code within that page is to be executed. The pages shaded in the diagram of the virtual memory represent those actually in main store; they will be copied to backing store, if they have been altered, when the main store space they occupy is required for other pages (probably by other users).

A file can be shared by two or more users, i.e. it can be connected simultaneously in two or more virtual memories. If a page of such a shared file is required simultaneously by two or more users, nonetheless only *one* copy of that page is held in main store.

Note that the diagram above is considerably simplified, and does not, for example, show the two levels of backing store which may be used.

Figure 4: File connection in EMAS 2900

CHAPTER 6: THE FILE SYSTEM AND DIRECTOR

In EMAS 2900, the Local Controller knows nothing of "files" - it merely organises transfers of pages of data within the storage hierarchy, in response to virtual store interrupts (page faults) generated as each process executes. A user's file on disc is not accessed using conventional read/write primitives, but by becoming part of the virtual memory of the user's process, i.e. by "connection" into the user's virtual memory. Connection does not involve any copying or data transfer: the disc addresses of the sections of the file are placed in a table. Thereafter access takes place through the normal paging mechanisms.

In order to provide to the user the means of allocating, naming and connecting units of disc storage (files), a set of procedures is available in each virtual memory. These procedures run at a higher level of privilege than the user programs and are shared by all processes. They share a data base comprising all users' "File Indexes" and form an important part of the EMAS Director. The Director procedures reference the File Indexes through the same mechanism as user programs reference files: they are connected into the virtual memory. Since files are described "by user", and to the Local Controller index references are indistinguishable from file references, there is no question of "catalogue references" forming a potential bottleneck, as can be the case with a conventional "volume table of contents" approach.

Each user's files and File Index are allocated on a single disc pack. Thus, if a disc drive fails or a pack is damaged, only those users allocated on that pack need be interrupted. Each Index provides a one-level naming scheme for that user's files. Files may be shared by many users; that is, they may be simultaneously connected in many virtual memories (at potentially different virtual addresses). Sharing is under the control of the user, in that he calls a Director procedure to give general, individual or group permissions to one of his files. The permission information is stored in the user's File Index. In order to connect and reference another user's file, a program specifies the owner's name and the owner's filename to the Director CONNECT procedure (Connection is illustrated in Figure 4). Usernames have six characters, and filenames have up to eleven characters.

The main Director file system procedures are:

CREATE	a file with the specified name
DESTROY	the file named (must belong to the user)
RENAME	the specified file with the specified new name (must be the user's own file)
PERMIT	other users to access the specified file (must be the user's own)
CONNECT	the specified file into the user's virtual memory
DISCONNECT	the specified file from the user's virtual memory
GIVE FILENAMES	i.e. give a list of the names of all the user's own files

OFFER	the specified file to another user (must belong to the user making the offer)
ACCEPT	the specified file from another user (file must have previously been offered)
NEWGEN	(new generation) of a file. This procedure provides a means of introducing a new version of a file, even though the current version may be connected in other users' virtual memories. Any subsequent connection of the file specified in the call yields the new generation of the file; the old generation is destroyed when it is finally removed from all virtual memories.

The maximum permissible file size is determined for each user by the System Manager, but in no case would it be expected to exceed one quarter (say) of the amount of virtual memory allocated to a process, currently 32 megabytes. Users requiring extremely large data structures would be expected to organise their data in units of the locally-determined maximum file size.

CHAPTER 7: OTHER DIRECTOR FUNCTIONS

Apart from implementing the file system, Director provides procedures for handling contingencies, process timing, accounting, interactive terminal I/O, file I/O from and to local or remote devices, record I/O from and to magnetic tape, the "detaching" of jobs for background execution, and the "detaching" of jobs to other computer systems on the communications network. In addition, Director procedures allow a program to run sub-programs securely at a lower level of privilege than itself.

Contingency Handling

Process contingencies may be either synchronous (e.g. program errors) or asynchronous (e.g. interactive terminal "INT:" messages, initiated by the user pressing the ESCAPE key). In order to allow the process to diagnose the contingency or to change the course of the computation, a Director procedure PRIME CONTINGENCY may be called in order to specify a procedure to be executed in the event of a process contingency. When a contingency occurs the process state is preserved for the contingency procedure to analyse; the procedure READ INTERRUPT DATA provides the contents of the machine registers at the time of the contingency. The contingency procedure may elect to resume or abort the computation, or to diagnose the error. In any event, the Director procedure RESUME provides the means of exit from the contingency procedure to the chosen environment. Asynchronous events may be inhibited for short periods of time if required; Director will queue a small number of events until uninhibited.

Process Timing

A Director procedure allows a machine instruction count to be specified: after this number of instructions has been executed an "instruction-count-exhausted" program error contingency is signalled. Procedures also exist to specify process timing in terms of CPU time used.

Accounting

Each user's Director maintains, both for the current session and cumulatively, totals of instructions executed, CPU time used, page-turns invoked and kilobytes of data transferred to and from real devices. User programs may read these totals, and privileged accounting programs (not Director procedures) may be supplied to read and reset the totals and to produce charging documents based on algorithms chosen by the System Manager.

Interactive Terminal I/O

In order to initiate interactive terminal I/O, each process must connect, in "communications mode", one file for input and one file for output. The Communications Controller autonomously places input data in the input file and updates a register, visible to the user program, each time a complete message has been transferred. When the user program has read all such data and requires more, a Director procedure REQUEST TERMINAL OPERATION is called which causes a "prompt" message to be dispatched for display at the terminal, if no data has been input in the meantime. The process is then suspended until an input message is available. Data to be sent to the terminal is written into the output file and REQUEST TERMINAL OPERATION is called to specify the extent of the data. Unless a very considerable amount of data has been specified, processing may then continue immediately.

The interactive terminal input and output streams form a full duplex pair. In addition to them, a control stream is available to the user by which interrupt-type messages may be sent independently from the terminal to the process. Multi-character messages sent thus are placed by the Local Controller in an area visible to the user program. Single-character control messages initiate an asynchronous interrupt of the user program (see "Contingency Handling", above).

It should be noted that the global sharing operates to eliminate the additional paging that appears to be required by this scheme. Writing terminal output to the file causes the page to be moved into main store; the REQUEST TERMINAL OPERATION causes the Communications Controller to obtain the page before it is paged out of main store. When the transfer is over the page is written out in the normal manner. A converse system operates for input.

Local and Remote Device I/O

I/O to and from real devices, except magnetic tape, may not be performed by user processes. Instead a privileged executive process called "Spooler" accepts input files for users and queues output files for local or remote devices. Director procedures are called by user programs to specify output filenames, device mnemonics and other parameters peculiar to the file or device type, if necessary. Delivery information, previously given to another Director procedure and recorded, is attached. The Director procedure then notifies the Spooler process, which queues and eventually dispatches the data. See also Chapter 8.

Non-Interactive Jobs

Jobs for background (non-interactive) execution, or for dispatch to other computer systems via the communications network, are similarly handed by a Director procedure to the Spooler process, along with job limits (CPU time, output limit), priority, job-ordering data, job output destination, number of magnetic tape decks required, and so on.

Magnetic Tape

A set of Director procedures allows for the claiming and releasing of magnetic tape volumes and for the reading and writing of data blocks of user-determined size. At installations having small numbers of magnetic tape drives, such use of magnetic tape by individual users would be exceptional and would clearly be at the discretion of the System Manager. Jobs requiring magnetic tapes might for example be required to be submitted for non-interactive execution.

Privilege

On 2900 series computers, procedure calls may optionally invoke a hardware, microcoded or software sequence to perform certain checks and change the level of privilege currently obtaining for the process. In order that user programs may run sub-programs securely at lower levels of privilege, Director procedures are available to set up or modify the System Call Tables, which control the hardware in making the change of privilege.

CHAPTER 8: THE PAGED SYSTEM PROCESSES

Introduction

The EMAS paged System processes (sometimes called "executive" processes) perform specific System functions which are not so time-dependent as those performed in the resident Supervisor. There are currently five such processes:

1. Volumes (VOLUMS)
2. Spooler (SPOOLR)
3. Engineer (ENGINR)
4. Manager (MANAGR)
5. Jobber (JOBBER)

The continuously running processes Volumes and Spooler are started automatically during the IPL sequence, and run without an interactive terminal. They are normally controlled by commands from operator consoles. The Manager and Engineer processes are not continuously running processes and are therefore normally started from interactive terminals. The Jobber process is started from the operator console when it is required to process jobs. All of these processes are significant in that they run at a higher level of privilege than normal user processes.

1. The Volumes Process (VOLUMS)

The Volumes executive is a privileged user process which is automatically started by the System after an IPL. Volumes maintains a File Index on each of the on-line file system devices. In these indexes it keeps lists of requests for transfers of files to and from tape.

The facilities provided by Volumes include the allocation of non-System devices (e.g. magnetic tapes) to users, the maintenance of the Backup and Archive file stores, and the transfer of files between the file system and these file stores.

Input to Volumes is derived from three sources:

* messages from the resident device control routines
* messages from users' Directors
* messages from operator consoles

Output from Volumes is sent to operator consoles and consists of replies to operator queries, requests to the operator for action, and error reports.

It is appropriate here to describe the use and maintenance of the Backup and Archive file stores.

Backup

"Backup" is the name given to the procedure which copies users' files currently resident on the on-line disc store, i.e. the "immediate file store", to magnetic tape storage. Its purpose is to ensure that important user files are protected against loss or corruption due to hardware or software failure. The Backup procedure must be carried out regularly, to minimise the loss of users' work should any files need to be restored to the file system as a result of corruption.

A number of options are available, to back up different classes of user files. These are:

* all changed files

* all files

* all changed cherished files (see below)

* all cherished files

A System Management decision is normally made either to back up all user files, or only those files nominated by users (i.e. files which have been "cherished"). Changed files are normally backed up daily (an incremental backup) and all files are normally backed up once a week (a consolidation backup).

If the System Management decision is that files are not to be automatically protected by the Backup system, it is left to the user to decide the relative importance of each of his files and to specify which are to be backed up. If a file is to be so protected, a Director service must be called to set a bit in the file descriptor in the user's File Index. The file is then said to be "cherished". A Director service is also available if the user wants to remove the cherish status of a file.

In the event of a file or set of files or a complete file system being destroyed or corrupted, it is possible to restore, to the on-line system, individual files, groups of files, or complete magnetic tapes of files.

Archive

"Archive" is the name given to the magnetic tape-based extension to the on-line file system. Its purpose is twofold: it allows users to "own" more files than their on-line File Index will allow, in a cheap and secure manner; and it serves to keep the amount of allocated on-line disc space in a steady state, despite a continuing demand for file space, by holding files which have been transferred from disc because they have been unused for some specified time.

When files with the cherish bit set in their file descriptors are removed to archive storage, it is normally because they have not been accessed within a specified period. Files which do not have the cherish bit set are normally deleted from the on-line file system if this specified period has elapsed since they were last accessed. This period is chosen by the System Manager to keep the file system in a steady state.

A user may explicitly request that a file be moved to archive storage, by calling a Director service to set the "to be archived" bit of the file descriptor in the user's File Index.

Each user has an on-line Archive Index of his files, which is accessible to him. Once a file has been copied to magnetic tape and a record added to the Archive Index, the disc copy of the file is destroyed. A file may be restored to the on-line file system, by calling a Director service to retrieve the file from archive.

2. The Spooler Process (SPOOLR)

The Spooler executive is a privileged user process which is automatically started by the System after an IPL. Spooler maintains a File Index on each of the on-line file system devices. In these Indexes it keeps queues of batch jobs to be run, and queues of files to be output to local peripherals and remote job entry (RJE) stations.

The facilities provided by Spooler include the control of local slow peripherals and communication with the Front End system to handle remote traffic. It accepts card, paper tape and RJE input to the System and carries out the spooling of all output to line printers, card punches, paper tape punches and RJE stations. It also controls the queueing of batch jobs, either detached from interactive terminals or "true" batch work entered on cards or from RJE stations.

Input to Spooler is derived from four sources:

* messages from the Communications Controller
* messages from users' Directors
* messages from operator consoles
* messages from RJE stations

Output from Spooler is sent to local and remote operator consoles and consists of replies to operator commands, requests to the operator for action, status reports and error reports.

3. The Engineer's Process (ENGINR)

This privileged process runs with an interactive terminal and is based upon components of the Edinburgh Subsystem (Chapter 9). It is permitted to access real devices at the physical handling level, so long as they are not being used by the System. ENGINR comes with utilities for labelling, formatting and testing discs, and for reading and interpreting System dump tapes. It is hoped to provide an interface for some of ICL's hardware testing and error logging utilities in this process.

ENGINR is an appropriate place for the user to run disc transcription programs to copy data from foreign discs, 7-track tapes and other devices not supported by EMAS 2900.

4. System Manager's Process (MANAGR)

This privileged process runs with an interactive terminal and is based on the Edinburgh Subsystem (Chapter 9); some user facilities have been withdrawn and additional System management utilities added. The System Manager can accredit and discredit users and change control parameters for particular users: passwords, File Index size, etc. There are also facilities for extracting accounting information from users' File Indexes, for passing to a billing program.

More detailed information about MANAGR is available in the EMAS 2900 System Management Manual, to be published towards the end of 1978.

5. The Jobber Subsystem (JOBBER)

This Subsystem is derived from the ICL product "Scientific Jobber", described in ICL Technical Publication TP6878. It provides a fast cards in/lines out service for student work and program testing.

The Jobber Subsystem can be started either automatically or by the operator, and runs without an interactive terminal. It requests input from the Spooler process; when no input is available it goes into a wait state and makes no demands on the System. Input can be submitted via any card reader, local or remote; alternatively an input file for the Jobber can be passed to the Spooler from an interactive process. Cards in/lines out jobs can be run for non-accredited users, but anyone using file facilities in the Jobber must be accredited and must have been given any necessary access permissions.

CHAPTER 9: THE EDINBURGH SUBSYSTEM

The EMAS 2900 Edinburgh Subsystem is based on the three Edinburgh written compilers - IMP, ALGOL(E) and FORTRAN(G) - together with a selection of editors, utilities, loaders and language library routines, etc. The central component of the Subsystem is the Command Language Interpreter (CLI) which is very simple yet general. Each command is assembled; the CLI then searches for a procedure whose entry point is the same as the command and then calls the procedure formally, passing the text string as the parameter. Each user can select whether the Subsystem files are to be searched first (efficient and the default), or whether his Directory structure is to be searched first (very general and allowing all Subsystem commands to be redefined to taste). This approach allows an attractive two-way symmetry: any command can be called by program and any procedure able to process a text string parameter can be called as a command. Commands may be written in any language, but IMP's text handling features make it the obvious choice.

The CLI will accept commands from a text file; this allows simple "macros" to be provided and is also the basis of background jobs. More complicated "macro commands" involving loops, jumps and conditions are usually written in IMP, taking advantage of the ease of calling commands from programs. The CLI will also accept a command or file of commands for execution at log-on time. This provides a way of extending the Subsystem by preloading additional modules.

A key feature of the Subsystem is the Directory structure, which is used both for loading commands and for satisfying references in programs. A Directory is a file which maintains the names of entry points and object programs containing them. The association is made when the user "inserts" the compiled program into his current Directory. The user can choose to search other Directories, perhaps belonging to other users, if an entry point (command) being sought cannot be found in the current Directory. An "aliasing" facility is also available for procedure (command) names.

These features make it easy to tailor the Edinburgh Subsystem to an installation's or user's requirements. However, it is necessary to maintain a balance between adding facilities and minimising searching and loading times; when substantial additions are required it is better to write a new Subsystem.

The following is a summary of the principal standard commands of the Edinburgh Subsystem:

ACCEPT(file,newname)

Accept file OFFERed by another user; optionally rename it to newname.

ALERT

List information on system faults etc.

ALGOL(source,object,clist,outdev)

Compile ALGOL 60 program from file source into file object (or .NULL) with listing in clist, and optionally extra list of fault messages on outdev.

ARCHIVE(ownfilelist)

Mark files for archiving.

CHERISH(ownfilelist)

Mark files for backing-up and automatic archiving after installation-defined period of non-use.

CLEAR

Clear definitions set up by DEFINE.

CONCAT(controlfile)

Read names of files to be concatenated from terminal or controlfile - one to a line; list terminates with .END. Followed on the next line by name of output file.

COPY(file,ownfile)

Copy file into ownfile; this is the standard method for adding a member to a pd file or copying a member out, i.e. either file or ownfile may be a member of an existing pd file.

CPULIMIT(time)

Set time limit for each subsequent command to time (mins,secs). At least one of mins and secs must be specified. CPULIMIT(?) gives current setting.

DEFINE(file/outdev,size)

Set up data file definition.	
size	integer giving the maximum size of the file in Kilobytes. Default 255.

For input, file may be concatenated using +, and pd members may be included, e.g. A+AOLD+ABC_X

DELETEJOB(joblist)

Delete jobs from background job queue.

DELIVER(address)

Set delivery information (for line printer output, etc) to address. DELIVER(?) prints current delivery information.

DESTROY(ownfilelist)

Destroy files or members of pd files.

DETACH

Send JOB to background job queue or to a remote processor.

DISCONNECT(filelist)

Disconnect files from virtual memory and copy back to backing store.

EDIT

Edit a character file, either within itself or to a new copy, using a number of single-letter commands to locate, insert, remove or move text in the file.

FILEANAL(file,out)

Give information about size, type and access permission of file.

FILES(mask,group,out)

List ownfiles in specified group which fit the mask.

mask	Up to three fields, where a field is either a set of explicit characters or the symbol '*' (representing any set of characters).
group	I immediate store files (the default) C CHERISHed immediate store files H HAZARDed immediate store files A ARCHIVEd files E causes extra information to be printed S specifies single spacing of file names

Any combination of these letter codes may be given.

FINDJOB(job)

Give information about job in background job queue; if job omitted, about all jobs in background job queues.

FORTE(source,object,clist)

Compile FORTRAN IV program from file source into file object (or .NULL) with listing in clist.

HAZARD(ownfilelist)

Un-CHERISH: files will not be backed up and will be destroyed if unused for four weeks. Also has the effect of revoking a request to ARCHIVE a file.

HELP(param)

Print information about Subsystem facilities. Param can be: omitted for general information; name of command for information about the command, e.g. HELP(RENAME); or output line printer, e.g. HELP(.LP).

IMP(source,object,clist,outdev)

Compile IMP program from file source into file object (or .NULL), with listing in clist, and optionally extra list of fault messages on outdev.

INSERT(objectfilelist)

Put information about object files into current Directory.

LINK(controlfile)

Read names of object files to be linked from .TT or controlfile, one to each line. List terminated with .END. Followed by name of output file.

LIST(file,outdev,copies)

List character file or data file on outdev. Files may be concatenated using the '+' operator, e.g. A+HOLD_V2+BC_V3
Copies: default 1, max 15.

LOOK(file)

Invoke Editor to examine, but not alter, file.
Default for file is SS#LIST.

METER

Print date, time and for this session, CPU time used, connect time, page turns and charge.

NEWDIRECTORY(newname)

Create a new directory file.

NEWPDFILE(newname)

Create a new, empty partitioned file.

NEWSMFILE(file,size)

Create file of size bytes for use for direct mapping.

OBEY(file,out)

Obey file containing sequence of commands and data, optionally putting output in out.

OFFER(ownfile,user)

Offer ownfile to another user. If user is omitted then revoke outstanding offer on ownfile.

PARM(parmlist)

Set compile time options. Remains in effect until next call or log off. PARM with no parameter resets the default options; the alternatives to the defaults are described in the relevant language manuals.

Note that PARM(?) gives list of current compile time options.

PASSWORD(foreground,background)

Set foreground and/or background passwords - each must be four printable characters other than comma.

PERMIT(ownfile,user,permission)

Set access permission for ownfile.

user	can be 6 character username, or null (meaning everybody else), or can include ? characters to indicate group of users.
permission	can be

```
E   Execute ⎫ or
R   Read    ⎬ any
W   Write   ⎭ combination
N   None
```

RENAME(ownfile,newname)

Change name of ownfile to newname.

RESTORE(ownfile,date,perm)

Transfer ownfile from archive store to immediate store. Format for date is dd/mm/yy. If date omitted use most recent copy on archive. Use perm to set other users' access permissions, coded as in PERMIT; alternatively WP restores with permissions in force when file was archived.
Default: NONE.

RUN(objectfile)

Run program which has been compiled into objectfile.

SEND(ownfile,outdev,copies)

List (and destroy) ownfile on outdev (default .LP). Copies: 1-15, default 1. Outdev cannot be .TT.

STOP

Print date and time and log off.

USERS

Print number of active users.

This list includes most of the standard commands. However the precise details of parameters have not been finalised.

Appendix: Glossary of Terms

This appendix provides brief definitions of a number of terms and abbreviations used in this manual and elsewhere in relation to EMAS 2900. It should be appreciated that some of the terms do not have generally accepted definitions and so may be used for different purposes with respect to other systems.

The first occurrence of a term is underlined in a definition if it itself is defined in the Glossary.

Access Permission	An attribute of a file, which can have several associated access permissions. Each one specifies a user, a group of users, or all users, and a corresponding permitted level of access - one of read unshared, read shared, write unshared, write shared, execute (applies to object files only). Access permissions thus determine which connect modes are possible. See also APF.
Accredit	To register a user on the EMAS 2900 System. This is done by the System Manager, and entails giving the user a File Index and access to it by means of a foreground password and a background password.
ACR Level	(Stands for "Access Control Register".) The ACR level is a number in the range 0-15. Each process currently executing in the System has an ACR level. When a process attempts to make use of a segment within the System, the Access Protection Field (APF) associated with the segment is compared with the ACR level of the process. If the ACR level is higher then the process is denied access to the segment. This mechanism thus provides a means of implementing levels of privilege and protection within the System, and is used, for example, to ensure that critical parts of the System software are not modified or corrupted by non-privileged processes. A process's ACR level can be altered, but this itself is a privileged operation if the ACR is being lowered.
Active Memory	A subset of the material connected in processes' virtual memories which is currently in use (see Chapter 3). Active memory tables are the means by which material is shared between processes. On machines with drum storage, some of the active memory will correspond to drum storage, but not necessarily all of it.
Address	An integer value which uniquely specifies a byte location in a store (memory). A "physical address" refers to a location in the computer's main store; a "virtual address" refers to a location in a virtual memory. A user process operates exclusively with virtual addresses, applying to its own virtual memory. Addresses of data items which are larger than a byte specify the leftmost byte of that data item.
APF	Access Protection Field. Files are handled by the System in terms of segments, and each segment has an associated Access Protection Field. It is used in conjunction with a process's ACR level to determine whether the process may operate on or make use of that segment.

Archive Index	Each user has an index of those of his files which are held in the archive store (cf. File Index). See also VOLUMS (Chapter 8).
Archive Store	A long-term store for users' files, held on magnetic tape. The executive process VOLUMS (Chapter 8) manages the archive store.
Asynchronous Interrupt	An event which can occur at any time, irrespective of the progress of any process. Asynchronous interrupts can be caused, for example, by a peripheral device, or a user at an interactive terminal pressing the ESC key. Their processing is often time-critical. They are distinguished from Synchronous interrupts. See also "Contingency Handling", in Chapter 7.
Background	A synonym for batch.
Backup	The procedure of copying users' disc files to magnetic tape, the purpose being to enable their restoration should they be corrupted or destroyed through hardware or software malfunction. This is distinct from archiving, in which disc files are copied to magnetic tape and the disc copy then deleted. See also Chapter 8.
Batch System	A method of using a computer non-interactively. The term originally came from the practice of collecting a set of jobs in a batch (the "batch stream") and transferring them to magnetic tape so that the delay in starting the processing of a job after the previous one was completed could be minimised. The term no longer implies a specific ordering of jobs. In EMAS 2900, batch (or "background") work can run concurrently with interactive work.
Block	The unit of active memory allocation, currently up to 16 epages.
Byte	The smallest addressable unit of store used in EMAS 2900. It contains 8 binary digits (bits).
Chopsupe	A rudimentary supervisor loaded at IPL time to check and investigate the hardware before loading the Supervisor.
Communications Controller	A resident part of the EMAS System by means of which all communications between the mainframe and the outside world are managed. This includes I/O traffic of all kinds, including interactive. The Communications Controller is described in Chapter 5.
Connection	The only method provided by EMAS 2900 for gaining access to files. It is a simple concept central to the provision to each user of a virtual memory. See Chapter 6 for details.
Console	A keyboard device connected to a computer system and permitting dialogue with it. A "user console", or "interactive terminal", is normally an electric typewriter or video device. An "operator's console", or "Oper", is used by a computer operator to control the running of the EMAS 2900 System. A single mainframe may have more than one Oper, each comprising one or more videos.

Contingency	The occurrence of an event which is reported to a process by means of an interrupt. Contingencies, like interrupts, are classed as synchronous (e.g. program errors, instruction counter overflow) or asynchronous (e.g. user console "INT" messages). The Director provides facilities for specifying a procedure to be executed when a process contingency occurs, and for giving such a procedure access to relevant information when it is activated.
Core	An anachronistic synonym for main store, which used to be constructed from ferrite cores threaded on wires. Main store is normally constructed from semi-conductors in modern machines.
Core-resident	A synonym for resident.
CPU	(Stands for "Central Processor Unit".) A piece of hardware central to the computer which coordinates and controls the activities of all the other parts and performs all the arithmetic and logical operations to be applied to data. The basic machine instructions are executed by this unit. The term "CPU" is not strictly applicable to 2900 Series machines, although it is sometimes used as a synonym for OCP.
Demand Paging	The policy of bringing a required page into main store only when a page fault has occurred for lack of the page, and not before. This is distinguished from pre-paging a process's working set when the process is reactivated.
Device Driver	A resident piece of software devoted to controlling the operation of a particular type of I/O device. In general a device driver is resident in the computer to which the device in question is directly attached. This may be the 2900 mainframe, or the Front End Processor, or some remote computer. See also Chapter 5.
Director	A collection of routines, forming part of the EMAS 2900 System. Each user process has its own Director (more strictly its own incarnation of Director, since the actual code is shared). If EMAS 2900 is seen as an onion-like structure, Director is the level directly below the Subsystem. All the services provided for a user by EMAS 2900 are requested by calls on Director routines, which thus form a procedural interface between a user process and the rest of EMAS. Normally, however, there is another layer between this interface and the user, viz. the Subsystem. Director is paged; it is contained in the associated user's virtual memory. The user cannot corrupt it because of ACR protection. See also Chapters 6 and 7.
Disc Drive	An I/O device with a rotating spindle onto which a disc pack can be mounted, together with read/write heads for each recording surface, a mechanism for positioning the heads, and electronics for controlling the reading and writing of data. In some designs the read/write heads are part of the disc pack, not the disc drive.
Disc Pack	A magnetic storage device consisting of a number of flat circular plates, each coated on both surfaces with some magnetizable material. The plates are mounted on a central

column. On each surface there is a series of concentric tracks and when the disc pack is mounted on a disc drive and rotated at high speed, a read/write head (one per surface) can be moved radially to a required track for accessing or changing the information stored there. EMAS 2900 currently supports EDS 200 disc packs (capacity 200 Mb) and EDS 100 disc packs (100 Mb) to hold users' disc files.

Disc File — See store.

Disc Store — See store.

Drum — A rotating magnetic storage device, similar in principle to disc storage. The primary difference is that there is one read/write head associated with each track so that the heads do not move. Drums are sometimes called "fixed-head discs". See also active memory, and store.

Edinburgh Subsystem — The name given to the principal Subsystem currently used in Edinburgh, and provided with EMAS 2900. Some details of the facilities, standard commands and design philosophy are given in Chapter 9.

Epage — (Stands for "extended page".) See page.

Executive Process — (Also called a "paged System process".) Similar to a normal user process but belonging to the System. The executive processes are used to carry out Systems tasks which are non-time-critical, such as spooling files, managing the archive store, running Engineer test programs, etc. They differ from user processes in that: 1) they may have access to facilities not given to users; 2) in some cases they are not run from an interactive terminal; 3) in some cases they may not make use of the Edinburgh Subsystem.
They are described in Chapter 8.

File — A collection of information held within some storage device in the computer system. EMAS 2900 operates, from the user's point of view at least, in terms of files, and extensive facilities are provided by Director for using them, keeping them, sharing them, etc. Director does not impose any conditions or conventions regarding the interpretation or use of the contents of files. The Edinburgh Subsystem provides facilities for working with particular types of files (see the EMAS 2900 User's Guide, to be published in 1978), although other Subsystems need not adhere to the conventions adopted by it. A user's disc files are all kept on a single disc pack.

File Index — Each accredited user of EMAS 2900 has a File Index which is maintained by his Director; it can be referred to and modified only by means of calls on Director routines. It keeps information about all the disc files belonging to that user, including their sizes, storage sites on disc, access permissions to other users, etc. The File Index itself is a file, and is kept on the same disc pack as all that user's files. It is connected into the user's virtual memory when he logs on.

Filename	(Sometimes written as two words.) The name given to a file by a user when it is created or renamed. The name must be unique within that user's File Index, but other users may have files of the same name; see also full filename. A filename can contain up to 11 upper case letters or digits, the first character being a letter. This restriction is imposed by Director. A user's Subsystem may impose further conditions, although the Edinburgh Subsystem does not do so.
File System	Each user of the System has a File Index, which contains information about all his disc files. A series of disc packs is used to implement this storage level, and each user's File Index and associated files are held on a single disc pack. A file system consists of the files and File Indexes held on a single disc pack, together with other information stored there to manage the space allocation and to ensure the self-consistency and integrity of the disc pack. The file system is the collection of all the disc pack file systems. Note that the term "file system" refers only to on-line files, i.e. those kept on disc.
Foreground	A synonym for interactive.
Front End Processor	(Sometimes called "FEP".) The only computer which is directly connected to the EMAS 2900 mainframe. Its job is to manage communications with remote processors and terminals. On the Edinburgh 2970 the FEP is a PDP 11/34, and is connected to a PDP11/45 which controls an interactive network of terminals, although logically this task can be undertaken by the FEP itself. The purpose of having a FEP is to devolve from the mainframe a considerable part of the task of communicating with the outside world, thus enabling it to spend more time on user tasks. See also Chapter 5.
Full Filename	A filename together with the owner's username. Under the Edinburgh Subsystem the two parts of the full filename are joined by a '.' to form a single string: thus "username.filename". The full filename is unique to that file and must be specified by users other than the owner when referring to it.
Global Controller	A collection of processes, forming the central part of the EMAS 2900 Supervisor. It is completely resident. It arbitrates between the various Supervisor and user processes for resources, such as CPU time and main store (although it should be noted that much of this task is delegated to the Local Controller). It also controls the sharing of active memory, and contains the device drivers for devices directly attached to the mainframe. See also Chapter 3.
GPC	(Stands for "General Peripheral Controller".) Part of the 2900 hardware which controls, at the hardware level, the slow devices (magnetic tape units, card reader, line printer, Operator console), and the Front End Processor. It is itself controlled by a procedure called "GPC", which forms part of the Global Controller.
Immediate Store	A synonym for disc store, no longer used.

IMP	A block-structured problem-oriented high level programming language of the ALGOL type. It was developed in Edinburgh from Manchester University's Atlas Autocode, primarily for implementing system software - hence its name. Apart from a very small amount of machine code, the EMAS 2900 System software is written entirely in IMP; this includes the Edinburgh Subsystem and the compilers currently available (ALGOL(E), FORTRAN(G), IMP). For details of the language refer to the "IMP Lanugage Manual", published by ERCC (1974).
Incarnation	A term used to describe a single activation of a program or routine which may be being shared and executed by several processes simultaneously. It can also be used of recursive routines. With respect to EMAS 2900, there exist at any time several incarnations, for example, of the Local Controller, of Director, and perhaps of the compilers.
Inhibited	A process can indicate to the message passing mechanism that it is not prepared to deal with certain classes of message for the moment. Such messages are then described as "inhibited". They are queued until the process is prepared to deal with them.
Interactive	Descriptive of the type of computer use which permits a user to respond to the output as it is being generated, and thus to stop or change the course of a computation. For such interaction a user console is required. With the Edinburgh Subsystem three levels of interaction are permitted to a user: 1) after each command has been completed, successfully or unsuccessfully, the user is asked for his next command; 2) during the execution of a program, the user may be asked for input data, depending on how the program was written; 3) the user can interrupt any program running in his process and redirect or abort it. The standard alternative to interactive operation is batch operation.
Interrupt	A signal which may be generated externally, for example by a peripheral device, or from a running program, and which normally causes the process currently executing to be suspended until the contingency implied by the interrupt is dealt with. Interrupts are of two types: synchronous and asynchronous. Interrupts can occur during the handling of previous interrupts; the System's behaviour is then dependent on the relative priority of the interrupts in question.
I/O	(Stands for "Input/Output".) Information travelling to or from a computer system.
I/O Device	A piece of hardware attached to a computer; it can read, store or write information in some form.
IPL	(Stands for "Initial Program Loading".) Strictly a hardware function involving reading in a minimal control program to an otherwise empty machine. In the case of EMAS 2900 the term covers the initialisation of the mainframe for an EMAS session. This entails discovering the configuration, initialising store and the peripheral controllers, carrying out a consistency check of the File Indexes, and starting the continuously running System processes (Chapter 8).

Local Controller	A resident part of EMAS 2900 of which each user process has a separate incarnation. The Local Controller manages the virtual processor in which the user process runs, and acts on its behalf to obtain System resources from the Global Controller. See also Chapter 4.
Mainframe	Strictly speaking, the central processor unit (CPU) and main store of a computer. It is often used to distinguish a large, powerful computer from a smaller satellite computer (e.g. a Front End Processor).
Main Store	The part of the computer hardware which can store instructions or data to be executed or read or written directly by the OCP. With respect to the EMAS 2900 System, main store is divided into "page frames", each able to hold a page. It is allocated thus to user processes, although EMAS 2900 manages main store, active store and disc store in such a way that they appear to form a contiguous main store much larger than the pages of main store actually allocated. See also store, and virtual.
Mapping	In EMAS 2900, the correspondence of an area of disc store with an area of virtual memory.
Memory	A synonym for store.
Message	EMAS 2900 is composed of a large number of interdependent processes, some associated with the Supervisor, some executive processes, and some associated with users. These processes have to communicate with each other, in order to ask for actions to be performed, or to inform that an action has been performed, and on occasion to achieve synchronisation (when, for example, two processes might try to access a shared file simultaneously). This communication requirement is fulfilled by a message passing mechanism which is administered by part of the Global Controller.
Non-paged	Often loosely used synonymously with resident. Strictly, descriptive of a segment which is not referenced via page tables. Thus, strictly, a segment could be resident although paged.
Non-time-critical	Descriptive of tasks or events requiring to be done or acted upon, but for which it is not important if there is an arbitrary delay in the response. In view of this, the code required to do the task is normally non-resident, and may be part of an executive process. See also time-critical.
OCP	(Stands for "Order Code Processor".) A piece of hardware central to the computer which performs all the arithmetic and logical operations to be applied to data. The basic machine instructions are executed by this unit.
On-line	1) Of peripheral equipment: logically connected to the EMAS 2900 System, the alternative being "off-line". 2) Of a user: having successfully carried out the standard log-on procedure via an interactive terminal, so that the associated user process has been started up and interactive working can be undertaken. The alternative is batch (or "background") use of the System, in which a terminal is not normally involved.

Oper	The standard abbreviation for the operator's console.
Order Code	The set of codes used to represent the basic operations carried out by a 2900 Series OCP.
OUT	A software-generated synchronous interrupt, used in EMAS 2900 to implement the message passing mechanism, and to enable Director to make requests for services from the Local Controller.
Page	The basic unit of store in 2900 Series hardware, of size 1K bytes. EMAS 2900 works with "extended pages" (epages), which are defined in units of pages. The size of an epage is currently 4 pages, although this could in principle be altered. Main store is divided into epages, and information is transferred between main store and backing store in units of epages. From the point of view of a user or a Subsystem, however, information is handled in terms of files, the paging operating transparently.
Paged	A term used to describe a segment which is accessed via a "page table". It means that when it is connected in a virtual memory, it is subject to the effects of paging. See also non-paged.
Paged Process	A process whose virtual memory is subject to paging.
Page Fault	An interrupt which arises when reference is made by a process to a page of a virtual memory, and the page in question is not in main store at the time. The paging operates, transparently to the process, to locate the page in active or disc store and bring it into main store. The process making the reference is suspended until the page is in main store.
Page Turn	A term used to describe the transfer of a page to or from main store. Page turns are recorded, and the total number of page turns caused by a process can be used as accounting information.
Paging	Descriptive of the task carried out by the Paging Manager, which is a component of the Global Controller. When a page fault occurs, the Local Controller associated with the process which caused it will make a request to the Paging Manager, which will initiate the transfer of the page to main store or inform the Local Controller if it is already there. As this is a global service, account can be taken of pages which are shared between processes. The Paging Manager also has the task of returning pages to backing store (if they have been modified) or discarding them (if they have not been modified), when the page frames they occupy in main store are required more urgently for other pages.
Pre-paging	A technique used by EMAS 2900 to bring pages into main store before they have been referenced (which would have caused a page fault). The pages which are pre-paged are the process's working set; the alternative approach is to let the process demand page them. See also Chapter 1.

Privileged	Descriptive of a program or process which has access to files, programs or facilities which are not accessible to user processes. This is achieved by the use of the ACR level mechanism.
Process	An autonomous activity which can compete for the resources of the System. It is the basic unit of scheduling. The main function of EMAS 2900 is to provide a suitable environment for user processes; the appearance of this environment is described in Chapter 2. Each user process or executive process can communicate with other processes by means of messages, but is autonomous in the sense that it apparently has a virtual processor and virtual memory to itself. The services provided by EMAS 2900 are implemented in terms of processes: a typical Supervisor or executive process will be activated via the message passing mechanism when there is a task for it to do, and will be suspended when it has completed all its current tasks. In a single-processor computer only one process will actually be executing at any instant, but the resources are shared in such a way that all active processes appear to be executing. Two types of process are distinguished: a virtual process, which is paged and whose virtual memory comprises segments normally referenced via a Local Segment Table; and a Supervisor process, which is resident and whose virtual memory comprises segments referenced via the Public Segment Table.
Resident	Permanently residing in main store. See also non-paged; the distinction between the two terms is somewhat artificial.
Resources	1) In EMAS 2900, low-level scheduling is governed by a process's use of OCP time and by its working set size. 2) The System Manager can control the allocation of global resources, such as cumulative CPU time, file space, interactive terminal connect time.
RJE	(Stands for "Remote Job Entry".) The technique of using a computer, non-interactively, from a remote site. With respect to EMAS 2900, RJE terminals (usually, but not necessarily, comprising a card reader and a line printer) are connected to the mainframe via the Front End Processor, which handles all communications traffic.
Segment	The segment is the unit of memory protection in 2900 Series computers, and has a maximum size of 256 Kb. A process's virtual memory is defined by segment tables (the single Public Segment Table and the process's Local Segment Table), which contain the access control information and the information required to effect the mapping between virtual store and main store. All segments referenced via Local Segment Tables are paged. A connected file comprises one or more segments.
SFC	(Stands for "Sectored File Controller".) A part of the hardware of the computer, which controls the operation of the fixed head discs (drums).

Spooler Process An executive process (named SPOOLR) which controls the transfer of files between the mainframe and 1) local slow peripherals, or 2) remote devices (via the Front End system). All input files destined for, and output files generated by, user processes are transferred via the Spooler process, which also handles batch jobs, however submitted. See also Chapter 8, and spooling.

Spooling (Stands for "simultaneous peripheral operation on-line".) A technique originally used to avoid the inefficiency of keeping most of a computer system idle while a job was being read in (e.g. from a card reader) or results were being printed out (e.g. to a line printer). It also prevents contention between processes for physical devices.
The technique is to read in jobs (or files) and keep them in a queue on backing store, at the same time as earlier jobs are executing. When another job is to be executed, the operating system (which controls the spooling process) chooses a job from among those in the input queue. A similar technique is used for spooling output: for example, a file to be sent to the line printer is initially kept in a line printer queue on backing store, until a line printer is available for it. Spooling is normally transparent to the end user, since the intervention of the operating system is automatic when an I/O operation is requested.

Stack An area of store which operates on a last-in first-out basis and is accessed by the machine stack registers. Each process has a stack, which is intrinsic to it.

Store Four types of physical storage media are distinguished in EMAS 2900: archive store (magnetic tape), disc store, drum store (optional), and main store.
A file is either archived (held on magnetic tape) or it is a disc file. For disc files the following logical storage hierarchy applies: 1) a disc file may be connected; 2) part of a connected file may also be in active memory; 3) a page in active memory may also be in main store.

Subsystem The facilities provided by EMAS 2900 are made available to users via the Director procedural interface. These facilities however, are at a basic level and are intended as the primitives from which a higher-level user interface is to be constructed. Such a user interface is known as a Subsystem, which in EMAS 2900 is expected to provide such things as text editors, compilers, program loaders, subroutine libraries, run-time support, etc. Each user has his own Subsystem, although in practice this is an incarnation of a standard Subsystem. The only comprehensive Subsystem currently available is the Edinburgh Subsystem.

Supervisor That part of the EMAS 2900 System below the Director. The term encompasses the Global Controller, the device handlers, the Local Controller, and the Communications Controller.

Synchronous Interrupt An event generated from within the computer system by a process executing an instruction. Synchronous interrupts can arise either unexpectedly (e.g. a program error such as accumulator overflow), or be deliberately provoked (e.g. by a "System Call" instruction).

System	"Computer System" is the term used to describe the composite of a computer and the software available with it; it may also include satellite computers (e.g. Front End Processors) and associated software. The "EMAS System" (or "EMAS 2900 System"), is sometimes used to mean the software up to the Director interface. More usually it implies a 2900 Series computer with EMAS software, up to and including Subsystems and user programs.
System Manager	The person who is reponsible for the day-to-day running of the EMAS 2900 System, and who takes the management decisions which arise (such as those relating to the accrediting of users).
System Process	Normally means paged System process, which is also called executive process, or "privileged user process". Strictly, any process within the EMAS Supervisor, paged or non-paged.
Terminal	Synonym for console, usually when referring to "user console".
Thrashing	Descriptive of the undesirable state of a paging operating system when it is spending all or nearly all of its time transferring pages to or from backing store, and thus doing little, if any, useful work. The design philosphy of the Local Controller minimises thrashing (or at least localises it to exceptional individual processes).
Time-critical	Descriptive of a task or event which must be acted upon within a finite time of its appearance (e.g. an interrupt from a fast device). A System component used to handle such events is normally resident.
Time-slice	A short period of CPU time (of the order of a tenth of a second) used in the scheduling of running processes. The time-slice is the maximum period of time for which a single process may occupy a processor without interruption.
Track	On a disc or drum, the part of the surface which is accessible from a single read/write head position.
User Process	See process.
Username	A unique six-character alphanumeric name assigned by the System Manager to a user when he is accredited, and used to distinguish him from other users. The user's process is given this name, and the full filename of each of his files starts with this name. He must also specify it when he is logging on to EMAS 2900.
Virtual	The primary function of the System is to maintain for each process the illusion that the process has a complete machine at its disposal without interference from other users. Most other features of EMAS 2900 follow logically from this objective. The components of this illusion are described as "virtual", to distinguish them from their "real" counterparts in the actual computer. Thus, each user has a virtual processor with a virtual memory, and in this processor there runs a virtual process; the latter only differs from a real process in that it is periodically suspended.

Volumes Process	An executive process responsible for the backing up, archiving and restoring from archive of users' files. See also Chapter 8.
Wait State	The condition of a device or process when it has been suspended from running to await the occurrence of some event.
Working Set	The set of pages referenced by a process within a short period of time. The System anticipates that the pages will be required again shortly thereafter. The working set is dynamic, and will alter as different parts of the virtual memory are referenced. Note that: a) The working set is itself a subset of the process's active memory; b) it is chosen by the System, not the process.