University of Cape Town
Department of Computer Science

OFFICE AUTOMATION

by Peter Stutz

A Thesis
Prepared under the Supervision of
Prof. P.S. Kritzinger
In fulfilment of the Requirements for the
Degree of Master of Science in Computer Science.

Cape Town
September 1989.

Abstract

Office automation systems have become an essential tool for the operation of the modern office. With the emphasis of a modern office being placed on efficiency and ease of communication, office automation systems have become the backbone of successful businesses.

COSNET is a prototype office automation system designed and implemented at the Department of the University of Cape Town and runs on Personal Computers that are linked to a NCR UNIX TOWER, which acts as the host.

This dissertation investigates the different facilities supported by some of the office automation systems compared in this thesis, and describes the COSNET features. This prototype office automation system supports many of the facilities that are supported by large office automation systems.

COSNET allows the user to define any MS-DOS based editor or word processor, and uses a *simple editor* for the creation of mail. The *electronic filing* facility allows documents to be *created, filed, retrieved* and *deleted*, and thus provides the users with the necessary features for document exchange. A user may set access permissions to each of his documents and may grant other users either *read* or *write* access to a specific document. The *mail* facility lets the user *read, file, forward, delete* and *print* a message, and supports classification of mail. A *calendar* facility is used as an electronic diary and stores all the user's schedules. These schedules may be viewed in either *daily, weekly* and *monthly* display modes. *Read* and *write* access to the calendar can be set by the user, in order to allow other users to manipulate his schedules. Any MS-DOS based application software can be added to COSNET. This facility allows the COSNET user to configure the office automation system to simulate the office environment.

COSNET thus supports most of the necessary features required by an office automation system.

CR categories - H.4.1, H.4.3.

Keywords - Office automation, electronic filing, electronic mail, electronic calendar, standards, user interface.

ACKNOWLEDGEMENTS

I would like to thank:

My supervisor, Pieter Kritzinger, for his guidance, help and encouragement during this project.

Riel Smit, Helen Steffen, Peter Wood, Graham Wheeler, and Pieter Kritzinger for advice in design, programming and documention of COSNET.

Hilton Lipshitz and Steven Versfeld for their efforts in testing COSNET.

My parents, brother and friends for their continued support.

The FRD who supported my studies with bursaries.

Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 DEFINITION

Office automation systems have become an essential tool for the operation of the modern office. With the emphasis of a modern office being placed on efficiency and ease of communication, office automation systems have become the backbone of successful businesses.

The terms "Office Automation System" (OAS) and "Office Information Systems" (OIS) are used interchangeably, thus sometimes creating confusion as to their meaning.

In this thesis I will define the term "*Office Automation System*" as follows: "An 'Office Automation System' refers to the use of integrated computer and communications systems to support administrative procedures in an office environment and to improve the quality and productivity of office work".

1.2 HISTORY

The history of office automation, according to Mayer [MAYE82], is a long and rich one. Mayer notes a number of innovations and product evolutions occurring in various fields relating to information systems, data processing and office practices which all have contributed to the emergence of office automation. These are summarized below:

1870s the first successful typewriter, essence of the office

1920s invention of the telephone

1930s development of '*scientific management*': it took a rational look at business practices and is the basis for systems analysis

1950s the application of Operations Research to the decision-making process

1950s the emergence of the photocopier

1950s introduction of electronic data processing for conducting business transactions

1960s telecommunications management provides the foundation of telecommunications networking

1960s emergence of Management Information Systems

1970s digital networks emerge as the means of communication between computers

1970s distributed processing as basis for local area networks

1970s application of word processing achieves widespread use

1970s use of behavioural science as a tool for management organizational change

1980s the emergence of the personal computer, bringing the computer to the individual and offering the possibility of completely distributed computing

1980s Decision Support Systems emerge as a tool in the management decision-making process

1980s the emergence of management workstations as new information manipulation tools for managers and professionals

An alternative view of the history of office automation is offered by Panko and Sprague [PANK82]. They maintain that the evolution of office automation can be traced to the early 1960s when the computers were beginning to stake a claim to data processing applications, replacing the older electronic accounting machines. Typewriters and photocopiers became commonplace and a fragmented approach to information management emerged. Panko and Sprague describe four major specializations geared to directing increased spending towards office technology:

1 **DATA PROCESSING, MANAGEMENT REPORTING** - computing invariably meant data/record processing. This new technology necessitated the employment of so many DP specialists, that a powerful and independent DP centre evolved, with its own DP manager. Applications began to extend beyond normal payroll functions, increasing the DP center's hold on the organization.

2 **SPECIALIZED OFFICE PRODUCTS** - These office products included mailing equipment, duplicators and microfilm, normally purchased by the individual offices using them.

3 **TELECOMMUNICATIONS** - Telephone and telex services were supplied by vendors which eased the organizational and administrative burden.

4 **GENERAL OFFICE PRODUCTS** - The outlay on these items such as typewriters, facsimile terminals and answering machines was provided for by annual budgets, but considerable initiative was left to individual departments.

The need for a unified management structure gained momentum towards the beginning of the 1970s. The financial outlay on information technology was increasing to such an extent, that it necessitated tighter control. Also, the technological barriers between the various tools were rapidly disintegrating as new equipment evolved. Recent advances in networking have enabled office products to be linked in decentralized office communication systems and have facilitated the recent trend towards distributed data processing.

In summary, the history of office automation can be thought of in two ways. The long perspective would start with the typewriter and progress through the developments of the telephone and the computer. The alternative view sees the history of office automation as a rather brief one, spanning fewer than three decades. This view suggests that the key to office automation lies in the development of the microprocessor and its application in office products. If the former view could be considered as '*evolutionary*' and the latter might be termed '*revolutionary*' [HIRS85].

1.3 DESIGN

Since automation of office work is one of the fastest growing application areas of information systems [ELLI80], many research programs have been conducted during the last few years to provide some guidelines with systematic foundations for the design of office automation systems [AHLS83] [COOK80] [GIBB82] [HAMM77] [LUM82] [MALO83] [ZLOO77]. The identification and consideration of the different facets and original aspects of the office environment constitute a fundamental issue of such research. The office models are classified either according to different views of the office, or according to the diverse aspects of office work. The resultant different categories are described in Chapter 2.

1.3.1 Standardization

Office systems may differ in several ways because each offers different services and answers the needs of different users. Thus the thread that ties the systems together is information interchange. The main aim is to let dissimilar office systems communicate with one another by creating standard protocols.

Standard protocols are the key to connecting individual networks into one integrated network. Without protocols the internetwork communication would be impossible and world wide network connection difficult. Standardization of protocols allow hardware independent communication between different networks, thus allowing for internetwork communication.

A number of organizations have realized the importance of such standard protocols and have been looking at requirements for international computer-based com-

munication [CUNN83]. The *International Federation for Information Processing* (IFIP), the *National Bureau of Standards* in the United States (NBS), the *Consultative Commitee on Telephone and Telegraph* (CCITT) and the *International Standards Organization* (ISO) have been developing standards for internetwork communication .

The interchange of documents between cooperating tools in office systems necessitates a fundamental, common understanding of the structure of the document. Therefore international standards have been implemented to enable the interchange of documents among open systems [SCHE88].

IBM uses the *Document Content Architecture* (DCA) and the *Document Interchange Architecture* (DIA) to standardize the information interchange between office systems [IBM82]. The DCA describes the form and meaning of content of a document and the DIA specifies the rules and data structures that establish the discipline for unambiguous interchange of documents between office systems.

The *Message Handling System* recommended by the CCITT (X.400), uses the protocols and services of the OSI and is implemented in the Application Level.

The combined ideas and cooperation between these organizations are leading to a single set of standards which will be universally accepted for world wide networks.

1.3.2 User Interface

The user interface forms the outer shell of the OAS and provides the tools necessary to access the OAS functions. It is the sole component of the OAS that the user sees and determines the degree of success of the OAS.

Forms, dialog, and **declarative** user interfaces are all candidates for user interaction with office automation systems.

A **dialog interface** takes the shape of system prompts requesting input from the user (e.g. menu driven systems). The interaction is sequential in nature and takes place in a prespecified order.

Declarative interfaces are those in which the user merely states the input in a non-procedural fashion with the system digesting the information and performing the appropriate commands. The language used for stating intent is usually quite restrictive when compared to natural language.

Electronic forms provide a good interface for the clerical facet of office automation systems such as handling purchase orders. Forms are not suitable for interaction that is unstructured or not predetermined.

Generally, an office automation system will provide one of the above mentioned user interfaces, but it is desirable to provide multiple types of user interface for the convenience of both clerical and nonclerical users [GEHA82].

The user interface must be easy to manipulate and must contain precise self teaching instructions. An extensive HELP facility that explains all the commands and actions is absolutely necessary to guide the newcomer through the system, explaining each step in detail.

1.3.3 Basic Functions

The focus of office automation systems is on systems and facilities to aid the office worker in the more basic aspects of his job. The following tasks have been identified as being part of a typical office automation system [GEHA82] [ELLI80]:

- *Electronic Mail*
- *Filing System*
- *Word Processing*
- *Administrative Support*

All office automation systems studied support these functions in one form or another as will be shown in Chapter 3.

1.4 CONCLUSION

An office automation system improves the efficiency of an office by automating repetitive activities, improving the speed of communication by decreasing the overheads of internal mail and by simplifying storage and retrieval of documents. The system must be easy to use and be able to be operated by people with minimal training. This means that the user interface must be clear, comprehensive and easy to manipulate as it will play a major role in the acceptability of the system. A "*user unfriendly*" system will not be successful.

In this thesis I will first look at the design requirements of office automation systems, thus looking at all the aspects involved in the design process.

This is followed by a comparison of a number of office automation systems, namely Comprehensive Electronic Office (CEO), Profs, All-In-One, Office By Example (OBE), Office Talk Zero, Scoop, Q-Office, Uniplex and COSNET. The discussion highlights the similarities and differences of the systems in relation to the design and implementation features discussed in Chapter 2.

The final part of the thesis introduces and describes COSNET. COSNET is an office automation system designed and implemented at the Computer Science Department of the University of Cape Town and runs on Personal Computers that are linked to a NCR UNIX TOWER, which acts as the host.

Chapter 2

DESIGN OF OFFICE AUTOMATION SYSTEMS

The design of office automation system can be split into two phases, namely the *requirements analysis* and the *requirements specification.*

2.1 REQUIREMENTS ANALYSIS

The requirements analysis involves a detailed analysis of the organization and requires close interaction of the designers with all personnel to determine the functions that should be supported by the office automation system. Before these functions can be determined, the results of four important aspects must be considered:

- *Activity Study*
- *Information Flow Analysis*
- *Management Requirements*
- *Impact on Office Workers*

I will now discuss each of these aspects in detail:

2.1.1 Activity Study

In the early stages of design of office automation system it is important to determine the time spent on different activities in the office. Thus the activities of three groups of employees need to be examined: secretaries, clerical workers and principals (managerial and professional personnel) [ENGE79]. Questionnaires need to be distributed and interviews need to be held to obtain data on the activities of these employees. By analyzing these data one can isolate the common tasks in the organization as suitable for automation.

This activity view of the office is a popular one, as activities are observable and, hence, relatively straightforward to measure [HIRS85]. Moreover, there is

Activities	*Top management(%) only*	*All Principals(%)*
Writing	9.8	15.6
Mail handling	6.1	4.4
Proof reading	1.8	2.3
Searching	3.0	5.6
Reading	8.7	7.3
Filing	1.1	2.0
Retrieving filed information	1.8	3.6
Dictating to secretary	4.9	1.9
Dictating to a machine	1.0	0.6
Telephone	13.8	12.3
Calculating	2.3	6.6
Conferring with secretary	2.9	1.8
Scheduled meetings	13.1	7.0
Unscheduled meetings	8.5	5.4
Planning or scheduling	4.7	4.3
Traveling	13.1	6.4
Copying	0.1	0.9
Using equipment	0.1	4.4
Other	3.2	7.6
	100	100
Total number of principals	76	329

Figure 2.1: Principal Activities Summary

less chance of ambiguity or misinterpretation. It is far easier to see where office technology could be used when the office is conceived in terms of activities instead of more subjective elements such as roles or functions. From an office automation vendor's viewpoint, the office activities view makes the most sense.

Perhaps the most widely quoted office activity study is that of ENGEL *et al* [ENGE79] with the following results:

2.1.1.1 Principals

Activities among the principals generally appeared to be consistent with the levels of the people responding (Figure 2.1).

For example, scheduled meetings, unscheduled meetings and travel, with 13.1, 8.5, and 13.1 percent, respectively, were very prominent activities among the upper-management respondents. In contrast, more administratively oriented activities, such as filing, searching and retrieving were more evident among non-management employees, 13.2 percent.

Upper-level management thus appeared to avoid administrative work and concentrated on communications oriented activities. Their ability to avoid administrative work appeared to depend on the support they received from subordinates and on the percentage of their work delegated to these people.

Even so, there was strong evidence that more work could have been delegated if proper people or systems were available. When asked if there were tasks that they do now that others could do for them, 51 percent of the principals indicated that they had one or more such delegateable tasks. At the time the study was made, principals were spending 14 percent of their own work month on tasks that could be performed by others. The analysis of these tasks showed that many of these task, such as copying, filing and retrieving documents, could be automated. In most cases, trained secretaries or clerical workers could also do 55 percent of the delegateable tasks.

2.1.1.2 Secretarial and Clerical Workers

This large percentage of delegateable tasks led to the analysis of secretarial and clerical activities to determine what this personnel was doing and where savings and automation might be made.

Among the secretaries, typing was by far the number one activity, but, it varied with the number of principals supported by the secretary (Figure 2.2).

Thus, private secretaries supporting a single professional estimated that they spent only 26 percent of their time typing. By contrast secretaries who supported more than four principals estimated that 46 percent of their time was devoted to typing.

With extra time available to them, it appeared that private secretaries did more administrative work, such as conferring with their manager, keeping calendars, taking shorthand and handling mail. Thus the improvement of typing productivity

Activities	*Average Percentage of time*
Writing	3.5
Mail handling	8.1
Bulk envelope stuffing	1.4
Collating/sorting	2.6
Proofreading	3.9
Reading	1.7
Typing	37.0
Telephone	10.5
Copying or duplicating	6.2
Conferring with principals	4.3
Taking shorthand	5.5
Filing	4.6
Retrieving files	2.8
Keeping calendars	2.6
Pick-up or delivery	2.2
Using equipment	1.3
Other	2.0
	100
Total number of secretaries	123

Figure 2.2: Secretarial activity summary

Activites	*Average percent of time*
Filling out forms	8.3
Writing	7.3
Typing	7.8
Collating/sorting	5.2
Checking documents	10.4
Reading	2.9
Filing	5.9
Looking for information	10.2
Telephone	9.2
Copying or duplicating	3.9
Calculating	10.3
Meetings	1.9
Pick-up or delivery	0.8
Scheduling or dispatching	1.2
Using a terminal	6.3
Other	8.4
	100
Total number of clerical staff	115

Figure 2.3: Clerical Activity summary

results in increased time availability to perform administrative duties in the organization.

Clerical activities did not fit any single pattern, other than to show that at least 41.9 percent and as much as 58 percent of the time is spent in paper handling (Figure 2.3).

Additional office activity studies have been performed by Mintzberg [MINT73], Kurke and Aldrich [KURK83], Stewart [STEW67], Poppel [POPP82] and Dodswell [DODS83]. All these studies reached the conclusion that precious time is wasted on repetative tasks involving paperwork that could easily be avoided by automating the these tasks.

2.1.2 Information Flow Analysis

In addition to the activities analysis, a considerable amount of study must be devoted to understanding the paperwork process in the organization. A detailed analysis of origins of the paperwork must be undertaken. An electronic substitute for conventional mail system can be justified, only if the analysis can show that a high percentage of incoming letters and memos originate within the company.

Engel's survey [ENGE79] sampled conventional business correspondence, letters

Copies of outgoing documents	*First two combined*	*First three combined*	*First four combined*
24% to dept. files			
19% routed in dept.	43%		
24% other HQ dept.		67%	
14% other company locations			81%
19% outside company			
Incoming documents			
3% from same dept.			
14% other HQ depts.	17%		
58% other company locations		75%	
25% outside company			100%

Figure 2.4: Document Flow

and memos, to determine where they were coming from and where they were going. The results are shown in Figure 2.4.

The pattern that emerged, on both the incoming and outgoing side, was that a substantial amount of the paper stayed within the company: 75 percent of the incoming letters and memos originated within the company and 81 percent of the outgoing documents remained within the organization.

The survey also showed that a lot of time was spent in copying each original document, six copies were made on average. This was also unproductive time in that it involved travelling to and from copiers, waiting for them to become available, and not unfrequently finding them to be out of service.

2.1.3 Management Requirements

Interviews that are held with the employees of an organization provide the designers of Office Automation Systems with an understanding of the organization and the direction in which the organization wants to move. From the management, in particular, the office system requirements can be determined [ENGE79]:

2.1.3.1 Increase in professional and managerial productivity

The ultimate goal of an Office Automation System would be to increase the managerial productivity. As seen from the discussion above, to increase the principal productivity, it is necessary to first improve the productivity of the secretaries, in order for them to provide better support to the principals.

2.1.3.2 Growth in stages

To achieve maximum value from an Office Automation System, almost all company locations would need access to it, and most employees in those locations would have to become users. Such a system might be potentially expensive, and the

application itself untried, untested and the question of user acceptance unanswered. To minimize such risks the Office Automation System should grow in stages, starting in departments with potentially high value before moving to other departments and locations. In this way, the financial risk would not only be minimized, but as the state of art advances, the company could be able to take advantage of new technological breakthroughs as they occurred, provided the interface was defined.

2.1.3.3 Fit within the existing organization

The new Office Automation System must avoid disruption of the established organization. Thus, the system should fit the users and not the reverse.

2.1.3.4 Tie into data processing applications

An Office Automation System should provide not only access to text documents and communication functions but should also provide integration with data processing applications as well. This could be achieved by having a single user interface to all computer applications, or at the very least by a single physical terminal connected to all systems.

2.1.4 The Impact On Humans In The Office

The implementation of an Office Automation System in a company changes the entire information processing scheme citeFINN83. This implies changes to the daily task and chores of individuals and the rearranging not only of peoples lives but of the physical design of an organization as well. The introduction of an Office Automation System to an office environment would force the office workers to reshape the patterns of their day and assume the new, specific tasks related to the computer systems they must now learn to use. If office automation is to succeed, then the system must not only be functionally complete, but it must be readily accepted by the office workers.

An important question is raised when an Office Automation System is introduced to an office environment: "How will the individual react to the new system?" The answer is attributed to many factors:

2.1.4.1 Aversion To Change

Research in human behavior has indicated that human beings react poorly to change [FINN83], though in varying degrees. Humans tend to view change with a mixture of fear, ambivalence, lack of interest and unwillingness to modify their behaviour. If office automation is to fulfill its promise of increasing productivity and making information processing a viable facet of office operations, then the management must develop a basic understanding of managing change.

Companies that have most successfully brought automation to the office environment while maintaining harmony and good will among office workers in the initial stages have embarked on an active public relations programs to market this

concept to their employees [FINN83]. Such a public relations plan could include several facets, each promoting the acceptance of office automation as a beneficial tool. Newsletters, bulletin boards and posters can be used to advertise Office Automation Systems. Departmental meetings involving all the employees in open discussion about the changes and uses of the system are essential [GRUH79].

2.1.4.2 Fear Of Job Loss

With the introduction of office automation many routine jobs will disappear. Filing, mail sorting, mail delivery, voluminous copying and other repetitive task will be greatly diminished. With this the need to dispel the myth that office automation will create job loss and unemployment is imperative.

Once an Office Automation System is installed, employees generally find that they are freed from the burdens of paperwork and other routine tasks to participate in decision making more effectively. If the task environment is organized appropriately, the training involved in the use of the Office Automation System can represent an upgrading of skills, increased status and job enrichment for clerical and secretarial workers [OLSO82].

2.1.4.3 Change In Stress On Office Worker

Increased workloads due to increased speed of communication may affect the stress of office workers, especially at the secretarial level [OLSO82]. The managerial level could experience increases in time pressure to respond to electronic memoranda that previously would have been typed and transmitted by ordinary mail. However, the extent that electronic mail replaces phone messages, the office worker will have the opportunity to respond to messages without having to react immediately on the telephone. The change in stress experienced by the office workers will undoubtedly have an effect on the attitude towards the new system.

2.1.4.4 Change In Interpersonal Relationships

The change in organizational structure that occurs with the introduction of new Office Automation Systems often affects the interpersonal relationships in the office. The communication features, such as electronic mail can provide a direct substitute for some forms of face-to-face communication. This can lead to a decrease in the amount of personal contact between a manager and secretary, between colleagues and between superiors and subordinates. Social needs play an important part in the motivation of individual workers [OLSO82] and extreme remote work environments should be avoided to maximize social interactions.

2.1.4.5 Ergonomics

Ergonomics is the science that studies the relationship between people and machines. It focuses on the design of computer terminals and the ease with which the user is able to adjust to computers, including a concern for structural and physical design

elements [CONR81]. Problems that have to be addressed include noise from printers and computers, static electricity, illumination, heat generated by computers and the general design of the office. These problems have to be resolved to create a user friendly and comfortable environment to suit the office worker.

2.1.4.6 Training

Proper training methods and documentation of the new system must be provided to ensure the system's success. One could argue, that a way of measuring a system's success is the percentage of time spent on the system by an individual. An adequately trained user will be able to enjoy all the functions supported by the system and will eventually grow dependent on the system.

2.1.4.7 Conclusion

The success of the new office system thus depends primarily on the strategies that the management employs to introduce the system the office workers. It is vital that each individual reacts favorably to the new system for it become widely used and cost effective within the organization.

2.1.5 Components Of Office Automation Systems

Using the management requirements, information flow analysis, and the activity studies as a framework one can now define the application requirements for end users of the Office Automation System. These requirements would become the basis for the system design and development for a prototype Office Automation System. The major functional areas are now described:

2.1.5.1 Word Processing

Nowadays, a word processor in an office environment is a must if the office is to run smoothly and efficiently. Virtually all existing offices have a word processor and hence the inclusion of word processing facilities in an Office Automation System seems automatic.

A word processor is used to create and update documents and messages. To learn to use such a word processor does not take very long, and thereafter, minimal effort is required to transform a type written report or letter into an electronic document.

Usually, the text editor has pre-programmed function keys by which the text is manipulated. A spelling checker should also be included as part of the word processing functions. This compares the text and compares the spelling in the documents with the dictionaries supplied by the system. Once a misspelled word is located, it is usually highlighted and a list of likely corrections is given. Two dictionaries are mostly used to check spelling: a standard dictionary and a private dictionary which contains words that are specific to the user's environment and which can not be found in the standard dictionary.

2.1.5.2 Electronic Mail

Electronic mail is the forwarding of message content by electronic means and is normally associated with communications between people. In the case of traditional postal service, the document is physically delivered to its destination using some transportation method. In electronic mail, the content of the message is transformed into electrical signals and forwarded over communication channels for at least a portion of its path.

An electronic mail system must be able to accept input, transmit the message and output the information. The following objectives can be identified as general requirements of an electronic mail system [ULR280]:

- **low cost** - a cost-effective mail system would be well received by most organizations.
- **reliability** - the mail system must be available when needed and must deliver the messages accurately and dependably.
- **ease of use** - the system must be designed with the user in mind to provide a simple, yet effective user interface.
- **control** - there must be a provision to monitor and control usage.
- **terminal independence** - compatibility with a broad range of user interfaces must exist.
- **security** - privacy and security measures must protect the physical facilities and also guard against unauthorized entry or access of the system.

Businesses are becoming more and more complex and have to deal with an increase in information exchange. There is an increasing need for office support to allow the office worker to deal with this accumulated information swiftly and efficiently. Electronic mail will require an inordinate amount of overhead if installed as a stand-alone function of the office but together, with the objectives described above, there is strong justification to integrate electronic mail and other office functions in today's organizations [O'KE80].

Thus, an electronic mail service has to provide a quick, efficient and inexpensive method for sending and receiving messages, and should improve the productivity of the office worker.

Short messages, such as reminders for appointments, inquiries, etc. can be sent using the electronic mail system. Electronic mail can also be used to send files and any type of document including informal memos, charts, reports, phone messages, spreadsheets and data tables. The mail received electronically by the user is usually delivered to a personal mailbox which can be read at any time. In the remainder of the discussion I shall refer to a message thereby understanding that it may in fact be a file.

Often the mail is divided into different classes, depending on the urgency of the contents:

- The **ordinary mail** will simply be placed into the user's mailbox and will remain there until the user inspects and manipulates the message.
- If the message is **urgent** and requires an immediate response by the recipient, it will be classified as urgent. The recipient will be notified in some way on the workstation screen and will be forced to attend to the mail.
- It is often the case that a user is given the right of access to another user's mailbox (typically a secretary to her manager's mailbox). It may then be required to classify some mail as **confidential**. Such mail will remain inaccessible to any other user scanning the mailbox or at least the user will be denied access to it.

Each user should be notified with some message on the screen whenever a new message arrives in the mailbox. The system should then automatically be placed in a mode which allows the user to read and manipulate the message. This is generally termed an **"interrupt facility"**, because the user interrupts the current activity, performs some other task and eventually resumes the activity at the point where he was interrupted. The user may then decide that the message is not urgent and continue with the interrupted activity straight away.

The standard functions supplied by electronic mail are:

- read the mail
- send mail (also reply)
- file the mail
- forward mail (to another user)
- discard mail
- print mail
- acknowledge receipt of a message

With the mailbox system the messages are sent by user identification, rather than by location, e.g. terminal number, thus allowing for the use of aliases. An alias can be used as well as the original identification of the user. The alias name is mapped onto the real name by the system and the message is then sent.

A user should also have the option to inspect the list of users who can receive mail in order not to omit a user when sending important messages. When sending one message to many users, a name list can be created which automatically sends the message to all the individuals in that list. This is convenient if messages are frequently sent to groups of users.

With an electronic mailing system that provides such features, one can communicate with individuals across a network of computers with ease. The message can be sent at any time regardless of the time of day and whether the recipient is

present or not. This saves a lot of overhead in internal mailing systems and makes communication fast, easy and convenient.

An option frequently supplied with electronic mail systems is to allow for a mechanism which would automatically generate a message when the intended recipient is away from his desk for a few days, notifying the sender to that effect.

To summarize, an electronic mail system provides the office with following benefits:

- the user does not need to locate anyone
- the user need not be interrupted
- all communications are automatically recorded and filed
- one message can go to multiple recipients

Thus, an electronic mail service provides fast and efficient information exchange within the office.

2.1.5.3 Electronic Filing

Most letters and documents in a modern office are created and edited on a word processor. The main advantage of doing this is being able to edit, insert or delete parts of text from a prepared document without having to re-type the entire document or entire pages. The next step is to create an electronic filing environment to store such documents, letter heads and personal messages. Such an electronic filing environment could be arranged to imitate a filing cabinet in the abstract. Such a filing system would be easy to use and will not have as many physical limitations on space as would be the case for the real cabinet .

Access control to personal material can be created to imitate locked and unlocked cabinets which ensures the security of classified material and unauthorized access to such material.

Thus all the required documents, letters, letterheads and other important material are centrally stored and can be electronically retrieved, deleted, updated and stored according to access permission.

Electronic filing allows the user to store and retrieve information by author, subject, date or other categories. Usually the documents are stored centrally in a file server. Different store and retrieval techniques exist. I will look at three methods of filing: the organization of the filing system into a "cabinet - like" structure, the use of filing machines or file servers [SLON81], and the use of message and signature files [TSI282].

Cabinet Filing

The "cabinet" filing structure is a hierarchical organization with drawers and folders which can be created without restriction on the number of items that are to be filed. Public filing drawers as well as a personal filing drawer are allowed on the system.

The personal filing drawer contains all the user's private documents and these cannot be inspected by anyone else. On some occasions the owner might give someone else permission to access the drawer, thereby giving him the full or partial right to manipulate the documents within such a drawer.

The functions for operating on a drawer's contents are:

- read contents
- edit contents
- add contents
- delete contents
- remove (the drawer itself)

The public filing drawer contains the documents which are being shared by many individuals. These may consist of reports, letters, letter heads and other kinds of frequently used documents. Because anyone can retrieve and access information in the public filing drawers, strict safeguard measurements have to be imposed to secure the documents contained therein.

Retrieval of documents can be handled in one of many ways. If the user remembers the exact location of the document, i.e. the drawer name, the folder name and the document name, then the document can be retrieved by specifying these. Other methods by which a document can be located are by:

- specifying document name
- date of creation
- author of document
- a keyword allocated to the document

The display mode is then used to inspect the document and to determine whether it is the desired document.

Filing machines

Due to the increase of information flow in the office, conventional information retrieval systems are not adequate. These systems are mostly designed to run on mainframe computers, are relatively slow and are generally too expensive for general use in business offices. It is therefore necessary, that the document filing and retrieval system should stand apart from any centralized computer [SLON81]. The functions to be performed by the document system are sufficiently important and their volume sufficiently heavy, to justify the use of a dedicated stand-alone machine.

A functionally distributed architecture approach is used for filing machines, allowing for the distribution of functions among several processors to increase the

execution speed of queries. The ideal configuration, thus, is one that permits simultaneous execution of logically subsequent tasks on separate processors. Individual processors are often also duplicated within the system to remove bottlenecks or accommodate changes in the number of users.

There are different strategies to retrieve and store documents and most filing machines make use of high speed caches to store the most frequently accessed documents. Internal algorithms maintain the memory organization efficiently due the dedicated processors.

An Office Automation System using filing machines to implement document filing has a partly distributed architecture which can have drawbacks. Apart from a centralized architecture providing tighter "control" over the Office Automation System, many clients prefer to have a machine that integrates all their desired functions to a distributed system. But this unsubstantiated view is often is outweighed by the efficiency, adaptability, upgradeability and simplicity of filing machines.

Message and Signature Files

A message can be defined to consist of a header and a body [IBM82]. The header contains formatted data representing the most important characteristics of the message and the body consists of a series of words.

A message file is a labeled container of messages which groups messages according to their meaning. This filing capability can be augmented with an access method which can retrieve messages according to contents. In this way, messages can be organized in general files rather than directories. The user specifies some attributes about a message to be retrieved and messages that qualify according to these attributes are retrieved [TSI282].

All messages are stored in large general files, according to user and role. The mechanism only retrieves a subset of messages, which the user then sequentially inspects.

When the user stores a message he will specify one or more logical files in which he wants to store the message. For each logical file, there is a physical file that stores a representation of the message, called the signature. Such a signature contains:

- a pointer to where the message is stored
- the type of the message
- machine representation of the attributes
- signature of the body

The signature of the body is a sequence of bits which approximately represents the significant words of the message body.

To retrieve a message, the user first indicates the logical filename, which contains many messages, and the type of the message. A template for the type is displayed and can be partially or fully filled out by specifying attributes of the message in the

blank slots. A pattern, that may consist of an expression of words relating to the body of the message, can also be completed.

The system then scans sequentially the signature file and checks for the type first. Should the type not match, the next signature is inspected. Once a matching type is found, the attributes and the pattern are matched to those of the signature. Messages that satisfy the query are those that satisfy the conjunction of all attributes specified, and whether the pattern of words appears in the body of the message.

Backup/Deletion of Documents

Many electronic filing systems have safeguards to restore deleted documents. Instead of discarding the document upon deletion, the deleted document is copied into a scrap area from which the document can be retrieved and restored in its original filing medium if need be. This scrap area is often referred to as the " wastebasket" and is periodically cleared. For example, a personal filing drawer would have its own wastebasket for every user and the public filing drawer would have only one wastebasket.

2.1.5.4 Administrative Support

Nearly every person working in a modern office has a diary in which the appointments, reminders and important events are scheduled.

The problem with such a diary is that access to it is either complete or non existent as far as other people are concerned. Either the owner of such a diary keeps the diary safely stored away and does not allow anyone access to the diary or he permits access to the diary, thus exposing all the entries in the diary to the person accessing it. There is no partial access to the diary, access is either complete or non existent.

Thus if a manager wants his secretary to schedule appointments for him, the secretary must have access to the manager's diary or keep a second diary for such purposes. This solution is not really efficient because the manager would have to consult two diaries to determine his daily activities.

An electronic diary can be set up for a user to contain all the usual entries a person requires. Additional persons could be granted limited access to such a diary to schedule or read from the diary without seeing any private entries of the owner of that diary.

The electronic calendar is used to keep track of all personal and business activities such as appointments, lunches, birthdays etc. The calendar can usually be inspected in three modes:

- **daily** - all the daily activities entered in the calendar are displayed with the full description of the activities.
- **weekly** - the activities for the entire week are displayed, usually with the full description

- **monthly** - the calendar for the month is viewed with the activities descriptions abbreviated to keywords only, such as lunch, meeting etc.

Normally the calendar of a user is private and no one else can read it or schedule activities unless given permission by the owner. Even if permission is given, some entries designated as personal cannot be read by anyone else but the owner.

Important entries in the calendar can be combined with the mail system to act as reminder messages. A message would then automatically be sent to the user's mailbox and the user reminded by a message which would appear on the screen. Other less urgent activities would simply be relayed to the mailbox for inspection.

Each user should have the facility to customize his calendar to suit his needs, e.g. if someone has irregular working hours then the daily activities should be scheduled in those specific hours.

2.1.5.5 Summary

Most Office Automation Systems support the basic functions of word processing, electronic mail, electronic filing and administrative support. The systems mainly differ in the method of implementation of these tasks and in the presentation of the user interface, rather than in the functional specification as will be shown in Chapter 3.

2.1.6 User Interface

2.1.6.1 Design Criteria

People generally have a goal in mind when they use an office automation system, such as editing a file or inspecting their mail. The office automation system and its user interface must therefore be designed so that the user can achieve a perceived goal easily. To succeed in this, criteria such as ease of learning, ease of remembering, ease of use, safety and customizability must be considered [SHAR82].

Ease of Learning

Office automation systems have a wide variety of users including those who are just getting started, those who know enough to get by, and those who can use them fluently. The human interface should therefore be designed so that new users can get started quickly and the other users can learn advanced commands easily. As shown in [LEDG80], the experienced users, when faced with a new system, are in much the same position as those who are just learning to use a computer terminal. Therefore, ease of learning is equally important to both. Two characteristics are helpful in this respect:

- Graded Command Structure
- Similarity to Familiar Things

Graded Command Structure

Commands should be designed so that they can be divided into several levels that are suitable for learning by users of different fluency of the office automation system. A new user would begin by learning the first-level commands. These commands must be sufficient to do the basic tasks of the office automation system and must be simple to understand.

As mentioned in Section 2.1.4, humans have an aversion to change and to new environments. The single biggest deterrent from getting started with an office automation system is the amount of complexity new users have to face in the beginning. The above mentioned approach minimizes that and thus eases the transition. More complexity can be tackled when the user is better prepared for it.

Similarity to Familiar Things

Learning new things can be easier if one can draw upon one's past experiences. This is the case if the new things are similar to what is already familiar.

Ease of Remembering

Consistent Command Structure

A command structure is consistent when it has few special cases and when similar commands have similar syntax. This makes the commands easier to learn and remember and the reference manual shorter. Descriptive and meaningful command names should be used as far as possible to improve the learning process.

Mnemonic Command Names

Command names should be mnemonic as far as possible. If all the commands cannot be made mnemonic, then at least the elementary commands should be made so.

Distinct Command Names

Command names that trigger different functions should be distinct. If they are similar to each other, it is difficult to remember which name stands for which command and errors are easily made.

Ease of Use

A program is easy to use when its commands can be put together to perform a task without disrupting the user's thoughts [RICH84]. That is the mechanics of performing a task (namely composing and issuing commands) should not be so complex as to distract the user from the overall goal. The following factors are helpful in making a program easy to use.

- **Power versus Simplicity** - Maintaining a proper balance between power and simplicity is perhaps the most difficult aspect of designing a human interface to an office automation system. Commands should be simple so that they are easy to learn and remember, but they should be powerful enough so that most of the tasks can be done without excessive effort. In designing the first-level commands, simplicity must be given preference over power. More powerful commands should be added to perform special functions, but they must be consistent with the existing commands. In some instances, a tradeoff between power and simplicity has to be made to achieve the desired effect of the office automation system.

- **Feedback** - If a program gives appropriate feedback, it can be learned more easily and used more effectively by its users. Three forms of feedback are discussed below:

    1. **Error Messages** - Users at all levels of expertise, especially beginners, make mistakes in issuing commands. The resulting error message should clearly state what may have gone wrong and suggest a corrective action if possible. For instance, sounding a beep is a common way to communicate

to a user that something has gone wrong in *Vi* [NCR85]. This violates the above guideline.

2. **State Information** - At all times, the current state or mode of the office automation system should be readily apparent to the user.
3. **Miscellaneous Feedback** - The office automation system might have some internal information that is not continually displayed on the screen, but may be needed by the user at times. On-line HELP screens are an example of such information, these have to be displayed in a readable and understandable format.

- **Unobtrusiveness** - An office automation system is obtrusive when it does not let the user do something that he thinks is quite intuitive and obvious. Such instances are undesirable, as they disrupt the user's thinking.
- **Prevent Unnecessary Typing** - The user interface of an office automation system should be designed in such a way, to allow the utilization of full functional power of the system with minimal typing requirements by the user.

Safety

An office automation system is safe to use when it guards the users against unexpected results and, as far as possible, provides for a means to recover from costly mistakes.

The user interface should be responsible for detecting illegal commands and should prevent the system from trying to execute these.

Customizability

If the user interface of an office automation system is customizable, it can be adapted to the needs and tastes of a variety of users. Customizability includes setting parameters, renaming commands, creating new commands (macros), etc. It is designed to accommodate the needs of the new and experienced users alike.

2.1.6.2 Natural Language Interfaces

Natural language interfaces are the most attractive of the three user interfaces discussed, but have a number of drawbacks. Because of the appealing features of natural language interface, a detailed description is given below.

As mentioned before, whenever users decide to communicate something to an office automation system, it is because they have some goal in mind. The user must generate a statement in whatever language is being used, and communicate this statement. The ideal situation would be, if the office automation system could understand that statement. This means that the statement would have to be translated into appropriate actions within the office automation system.

Since understanding is a translation process, a statement is understood only with respect to a particular language and a set of actions. A statement has no single meaning without a target set of actions. This is important, and it is one of the main reasons that building natural languages interfaces is difficult. It cannot be done once and for all; it must be done for each office automation system individually.

One also must realize that, when the term ***natural language interface*** is used, rarely an entire natural language such as English will be used. Instead, a ***subset*** of a natural language is used.

Thus one must consider not only just whether to use a natural language, but also how to choose an appropriate language subset for a particular office automation system.

Perhaps the most attractive aspects of natural language interfaces are ease of learning, ease of use and ease of remembering as described above.

But in addition to the five design criteria, there are at least five factors to consider in deciding whether to use a natural language interface.

1. **Cost of the Interface** - The design and implementation time of natural language interfaces generally cost more than those of more restricted interfaces. Thus, unless there are concrete reasons to use such an interface for an office automation system, it should be avoided [RICH84]. This is an important aspect and a tradeoff between efficiency and implementation cost has to be considered. The reasons for the additional cost of the natural language interface will become apparent later.

2. **Need for Precision** - Many English sentences are ambiguous, and parts of statements are ambiguous even if the full statements in which they occur are unambiguous - in a different context the part could have meant something else.

3. **Need for pictures** - Words are often not the best way to describe concepts such as shapes and positions. To communicate them, other techniques, such as light pens, cursors, or digitizing pads, are better. The natural language interface would possibly need to be integrated with other, graphics-based, communication tools.

4. **Semantic Complexity** - The number of different messages users need to convey to system is so large, that no trivial language can carry the load. Natural languages are concise and efficient when the universe of possible messages and commands is large. Thus, the larger the system gets, the greater becomes the need for a complex language interface.

5. **Promising More Than Can be Delivered** - The close connection between the range and complexity of a program and the range and complexity of the interface to the system exists in the minds of the users, but it leads into a pitfall if there is not a good match between the system and its interface. If the office automation system possesses a seemingly sophisticated interface, users will expect sophisticated behaviour.

Components of a Natural Language User Interface

The process of translating statements from the language in which they were made into program-specific form that causes appropriate actions to be performed is usually broken into three parts: *words* and *lexicon*, *grammar* and *sentence structure*, and *semantics* and *sentence meaning*.

Words and the Lexicon

The first thing that must be done to understand a statement is to divide it into its components. When statements are in a natural language, the obvious pieces are words. Dividing a sentence into words is called *lexical analysis*. For written English, this process is mostly trivial, as it can be done with a single pass over the input in time that increases linearly in relation to the length of the input. Sequences of characters between spaces are words. The list of words that a particular understanding program can recognize is contained in its *dictionary* or *lexicon*.

Design of a natural language interface requires that a lexicon be selected in a two-step process. The first step is to isolate the concepts to be expressed. Firstly, one needs to look at what the target office automation system can do and to decide what concepts it can understand. Secondly, one has to select specific words to represent these concepts. It is in some sense sufficient to select a single word for each concept, but this will make it very difficult for the users to know what expressions are acceptable.

Selecting words that will be acceptable is important because the everyday English lexicon contains so many similar words. Many English words also are ambiguous. The word *can* for example has two meanings:

- "Can he swim?"
- "The can cut him."

Whether a word is ambiguous in the interface depends on the subset of words that is used. Whenever a large subset of words is used, ambiguity can seldom be avoided.

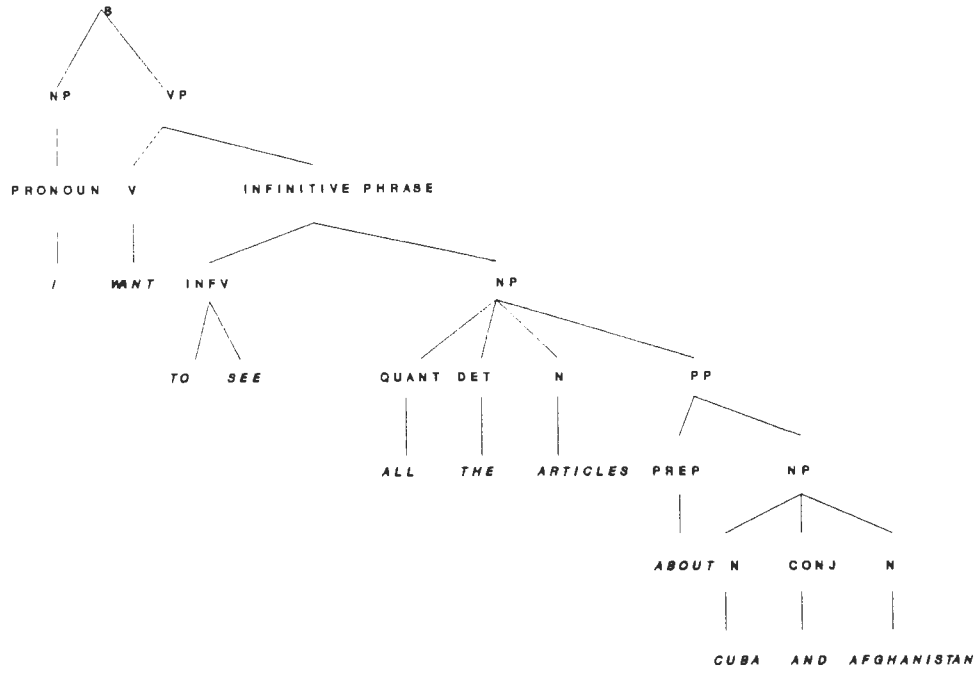


Figure 2.5: Syntactic Parse Tree

Grammar and the Structure of Sentences

Statements in a natural language are, prior to analysis, simply flat strings of words. But they typically stand for structured ideas. A first step toward finding the meaning of a statement is to assign to the statement a structure that will probably correspond in some way to the structure of its meaning. Assigning such a structure to an unstructured object is called *syntax analysis* or *parsing*.

Figure 2.5 shows an example of an English version of a database query that has been parsed by a standard English grammar. The parse tree shows how the components of the statement, S, are formed: NP = noun phrase, VP = verb phrase, V = verb, INFV = infinitive verb, QUANT = quantifier, DET = determiner, N = noun, PP = prepositional phrase and CONJ = conjunction.

Often the grammar is represented explicitly as a set of rules, and the parsing program refers to this set of rules. Then there are two principal ways of conducting the parsing process, and the techniques can be used in combination. The two principal approaches are *bottom-up* and *top-down*. In the bottom-up approach, the tree is constructed from the bottom, starting with the words in the statement. Intermediate constituents are constructed as their components become available, and the parse is completed when the top constituent, representing a complete statement, is formed. In the top-down approach, the search for a complete parse begins at the top, with a complete statement, and lower level constituents that could form a statement are hypothesized. The levels below them are then hypothesized, and this process is continued until actual words are needed. The words are then matched against actual input. There has been a great deal of research on parsing algorithms, much of it in the context of programming language compilers.

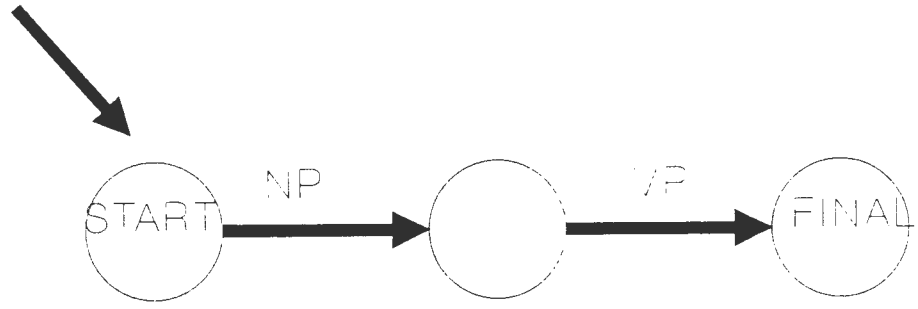


Figure 2.6: Fragment of an ATN

An alternative technique is to encode the grammar in a transition network constructed to correspond to the allowable transitions between the constituents of a sentence. A tiny fragment of such a network is shown in Figure 2.6. When this technique is used, parsing is simulated by walking simultaneously through the input sentence and the network structure. Usually the arcs in the network contain extra information that tells the parser what actions it should take during the parse in order to build the desired structure by the end. Woods calls these networks **augmented transition networks** or ATN's [WOOD70].

The problem of ambiguity has already been discussed in the context with lexical analysis. It must also be considered during syntactic analysis. The ambiguities could be resolved during later processing, as often only one interpretation makes sense. Often though, more than one form makes sense and the safest action then is to consult the user.

Semantics and the Meaning of Sentences

The final step of the understanding process is to assign a meaning to the statement to be understood. In other words, what action the office automation system is supposed to take. Assigning this meaning to a statement is called *semantic processing*. Actually two processes are combined at this stage: semantic processing (determining what a statement means) and pragmatic processing (determining what should be done about it). The techniques used to assign a semantic interpretation to a statement vary widely, depending on the way the rest of the interpretation is conducted, so a discussion will be omitted here.

Natural Language Interface Approaches

At this point, the key components of any natural language understanding program have been introduced. The next three sections describe three approaches to combining these components into a useful natural language interface.

Language Through Windows

When the number of statements that the user will need to make to the office automation system is not very large, one approach to the interface problem is for the system to display the available options to the user, who chooses from among those options and gradually constructs a complete statement, which is then guaranteed to correspond to actions that the target system can perform. An example of an interface built with this technique is Texas Instruments' NXL system. Figure blabla shows an example of a screen that could represent a user interface to a database. The outlined window is *active*, and it is from this window that the next word must be selected. The selected word must be added to the bottom window in which the statement, as it has so far been formed, is recorded. Also, the windows that contain choices for later words must be updated to show those options that can form legal (grammatical and meaningful) statements in the context of what has already been said. And finally, the active marker must be moved to the window that shows the choices for the next word.

A window-based, natural language interface runs relatively efficiently because the options available to the user are so rigorously constrained. But its main usefulness occurs in the domains of low semantic complexity, where there really are few options. In particular, to use this approach, it must be possible to encode in the grammar all the information that is required to determine whether a particular statement can be executed correctly. This decision must be made as the input is being constructed, and it must be made without appeal to the office automation system itself. In semantically rich environments, this is not normally possible.

Semantic Grammars

The window based approach does not really give users control over interactions nor allow them to compose free form statements that must then be understood by the system. Instead the system retains control and presents a list of options to the users, similar to the menu based user interface. When only a restricted number of statements makes sense to the system, this approach is feasible, but as the number of options increases, it becomes less and less practical. Another approach is then called for - one in which users compose entire statements on their own and the understanding system then translates the statements. One implementation of this approach uses a semantic grammar to drive the understanding process [BLAC86].

The basic idea behind the use of a semantic grammar is that a statement can be understood in only two steps, rather than the three previously described. Firstly, lexical analysis separates a statement into words. Then the words are analyzed

for syntax and semantics in a single step. Actually, a small third step may be required to produce the final output, but it is very simple, given the output of step 2. A semantic grammar is in many ways a straightforward extension of the window system to allow a greater number of user options.

Semantic grammars are useful when only a relatively small subset of a whole language needs to be recognized, but they do not capture much of the syntactic regularity of the language. As they are expanded to deal with larger and larger pieces of the language, they tend to get much larger and much more complex. Eventually their *ad hoc* character makes them unusable.

Syntactic Grammars

When a large fragment of a natural language is used as an interface, it is important to capture as much of the regularity of the language as possible in the rules used to understand it. To do so, it is necessary to capture the syntactic regularity of the language being used. This forces a return to a syntactically motivated grammar such as the one that generated the parse tree of Figure 2.5. Then it is possible, for example, to define the structure of a prepositional phrase once, independent of the preposition being used and the role of the phrase in the sentence. This syntactic approach contrasts with the semantic approach in which prepositional phrases may appear in several places.

During the construction of a semantic grammar, one looks at both sides of the translation (both the input language and the office automation system actions) and writes the grammar rules that map, as directly as possible, structures of one into structures of the other. The rules thus appear semantic in that they relate directly to the target actions. But when both sides of the translation are complex, it is difficult to consider both sides at once and to complete the translation in one step. Instead, grammar rules must be written so that they can be used by a parser to assign one structure to each sentence. Typically, a structure is chosen to capture generalities in the input language itself, independent of the target system. Other rules are then written to transform the parsed structure into target actions.

Generally, a large number of constituents are generated. To minimize the number of intermediate constituents that are considered and then rejected, most parsing systems permit grammar rules to be augmented with tests that must be satisfied in order for the rules to fire.

Conclusion

A natural language user interface to office automation systems seems to satisfy most of design criteria discussed above. It is easy to learn, easy to remember and easy to use. Safety and customizability would depend on the actual implementation of the interface together with the functionalities of the office automation system.

The only criterion that could be violated is that more typing is needed to specify a command than, for instance, using a menu driven system.

Unfortunately, constructing a natural-language interface is a time-consuming task, even after the basic structure of the target office automation system is well understood. The lexicon, the grammar, the semantic rules, and the code that uses all of them must be built. This can be a important factor in deciding against a natural language interface.

2.1.6.3 Electronic Forms as a User Interface

Electronic forms provide a very good interface for many clerical facets of office automation systems such as handling of purchase orders and airline tickets. These actions are usually well structured and have a predetermined order and thus are suitable for form based user interaction. The system usually displays an empty or partially filled form on the screen, and requires the user to complete the unfilled fields. Once the significant fields of the form are filled, the form is ready to be processed. Office automation systems such as Office Talk [ELLI80] ELLI80 have successfully implemented forms as the user interface.

Forms have a number of advantages over natural langauge and menu interfaces:

- Forms significantly ease the transition from manual office systems to automated office systems by minimizing the change experienced by the office personnel.
- Paper copies of electronic forms can be readily generated for interaction outside the office environment.
- The user interface provides a "true" environment since both input and display of information have the same format.
- The empty form helps the user by giving a global idea of the information needed whenever data is required.
- The fields of a form can usually be accessed and modified in any order on page forms. Returning back to a field for modification does not require any special instruction. Forms, thus, provide a *random access format* for the input and display of information.
- Many fields, such as the date or data depending on values of other fields, can be automatically filled in by the system thereby increasing the speed and efficiency of completion of the form. This cannot be done cleanly with other user interfaces.
- The display format of a properly designed form will reflect the logical structure or relationship between the various data items in the form.

Figure 2.7 shows an example of a possible form for a stock item. The field *TOTAL STOCK COST* can be filled by the system as soon as the fields *QUANTITY* and *PRICE* have received their values. The system would fill the *DATE* field as soon as the form is fetched.

But there are also a number of problems associated with form based user interfaces that prevent the sole use of this type of interface.

- Forms do seem to be ***unsuitable for interaction that is unstructured or not pre-determined***, as is the case in managerial decision making.

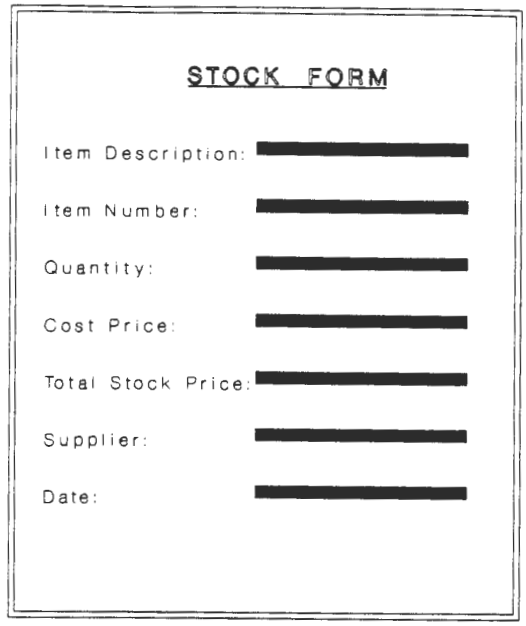


Figure 2.7: Form for a Stock Item

- Problems are encountered whenever new forms need to be added to a system or when existing forms need to be updated. Some prototypes for the definition of new forms in form based office automation systems exist [GEHA82], but these are in early development stages and do not offer a good solution to the problem since it is "tedious to define forms and change form definitions".
- Partially filled forms can create a problem. Often information supplied might not be complete and the form left in an incomplete state. This missing data item might restrict the system from further processing, due to the close relationship between the form and the system applications.
- Forms as a user interface are effective whenever a tight relationship between input data and stored data exists (as is the case in data base oriented systems). This is really a design issue, and it becomes difficult to map the various office applications to a form based user interface if the office automation system does not support such relations.
- Each field entered on a form by the user needs to be verified by the system. This can become an time overhead, especially whenever typing errors occur and the user has to re-enter a particular field. This problem is avoided by systems using selection of a menu item as the user interface.

To conclude, it can be said that a form based user interface satisfies most of the design criteria: it is easy to learn, easy to remember and easy to use. The user interface is safe, provided strict field checking is imposed but modification of the user interface is as yet not well developed.

A form based user interface is justified when applied to facets of an automated office system that are well structured and have a "data base" nature.

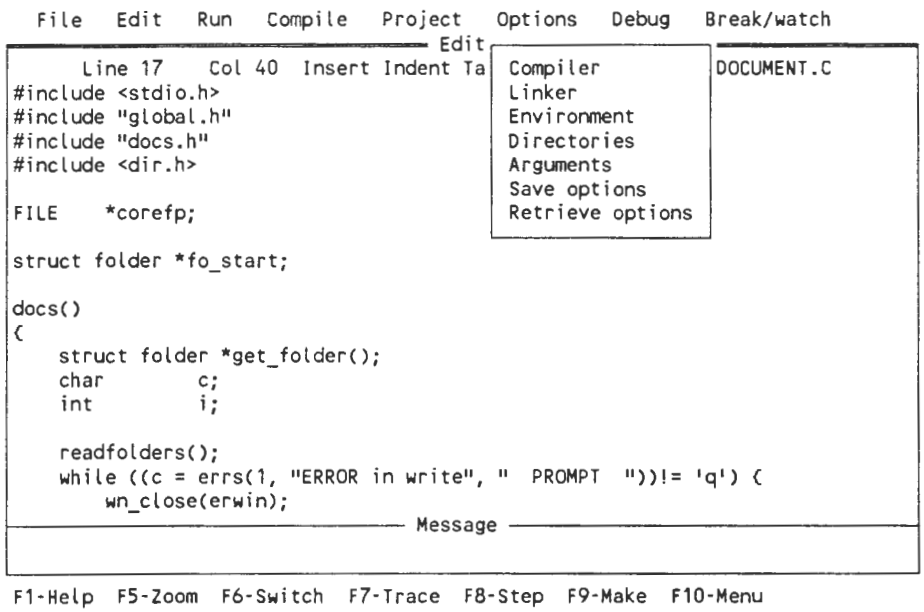


Figure 2.8: TURBO C Pull Down Menu

2.1.6.4 Menu Based User Interface

A menu user interface takes the shape of system prompts requesting the user to select a menu item that is displayed on the screen. The interaction is sequential and takes place in a prespecified order. One can classify three types of menu interfaces : the **Pull Down, Pop Up** and **Full Screen** menus.

Figure 2.8 shows an example of a *pull down menu* from the **TURBO C** program. The text at the top of the screen forms part of the *menu bar*; each word on the bar is one item that can be selected. Selecting a word, either by using the arrow keys or typing the first letter of one of the words, causes a *pull down menu* to appear on the screen.

On this *pull down menu* one can select one of the options by positioning the cursor on the option desired. The *pull down menu* vanishes after the user quits this menu, leaving the screen as it appeared initially.

A *pop up menu* presents an elegant way of displaying data or a *secondary menu* without having to rewrite or erase the contents of the entire screen. *Pop up menus* behave in a similar fashion to the pull down menus. They usually cover only part of the screen, leaving the rest of the screen visible to the user and restore the screen fully upon exit. Figure 2.9 shows a **COSNET HELP** *pop up menu.*

A *full screen menu* extends over the entire screen, thus hiding all the information that was displayed. Such a menu only displays information that is relevant to the current office automation function selected and does not distract the user by having "too much detail" on the screen. *Full screen menus* also create a "logical partition" between the various office automation functions by clearing the screen after the completion of a task. Figure 2.10 shows a PROFS Document Search Menu which

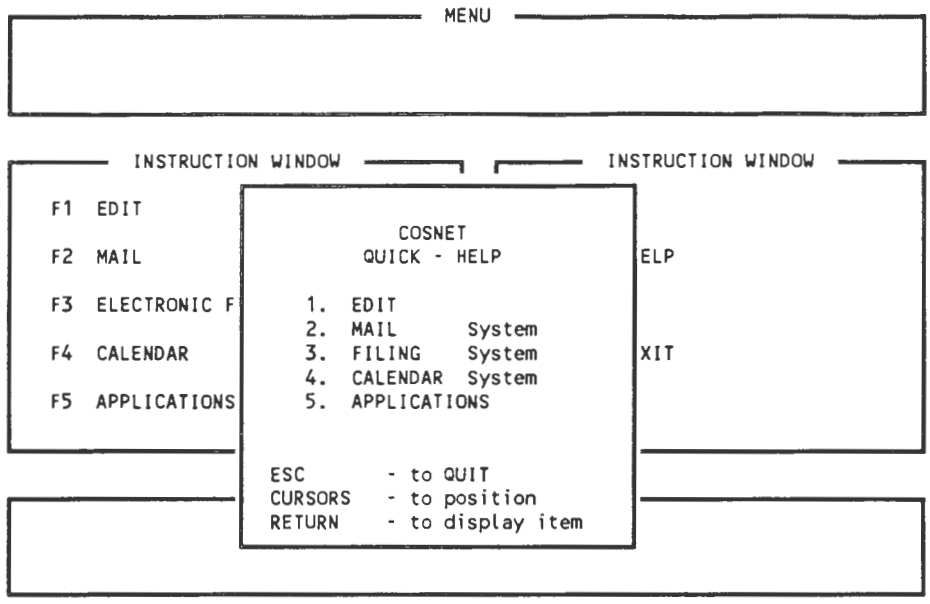


Figure 2.9: COSNET HELP Pop Up Menu

Figure 2.10: PROFS Document Search Menu

is in full screen format.

Many office automation systems such as PROFS, CEO and All-In-1 use one or more of the menu interfaces described above. A menu based user interface is easy to learn to use and to remember since all the options are always shown on the screen. Menus only allow the selection of displayed options and thus provide a reliable and safe interface to the office automation systems.

Customizing the user interface is one area where the menu user interface falls short in meeting the requirements for the user interface design criteria. Although office automation systems such as PROFS, CEO and All-In-1 allow the users to customize the user interface by letting the user redefine keyboard function keys, and change the display of messages or mail according to the users preference, it is not as easy to embody new tasks into the set of main menus.

The menu based user interface to office automation systems seems to satisfy most of the user interface design criteria and provides an effective and attractive interface to the office automation system.

Conclusion

From the discussion above it can be seen that different user interfaces suit different tasks. Since office automation systems support a collection of different tasks it is necessary that different types of user interfaces are provided.

Most office automation systems concentrate on *form* and *menu* based user interfaces (e.g. PROFS, CEO). Others also add the capability to support a restricted *natural language* and *voice* interface (All In One). IBM's *Office Vision* series is predominantly *Icon* based, adding a new type of interface to office automation systems.

But, *menu* and *form* based user interfaces remain as the most common interfaces in office automation systems since many of these elegant user interfaces are very much hardware dependent.

The *menu user interface* has been chosen for COSNET due to the easy, reliable and safe interface qualities provided by this interface. The COSNET user interface will be discussed in Chapter 4.

2.2 REQUIREMENT SPECIFICATION

The **requirement specification** deals with the actual design of the functions of the office automation system. Before discussing the design of these functions, though, the architectures, office models and standards for document and mail systems will be discussed.

2.2.1 Office Automation Architectures and Approaches

If OAS are to be successfully used in organizations, then some strategy of approaching office automation and office automation architectures must be defined [HIRS85]. This will serve as a guideline to the designers and analysts to formalize the structure of the office automation system.

The terms "*approach*" and "*architecture*" first need to be defined to avoid confusion. An "approach" can be seen as "what constitutes office automation in the eye of the organization or the designer" and "architecture" refers to the technical structure in which the approach is implemented.

The three main approaches and their respective architectures can be categorized as follows [HIRS85]:

- Functional approach
- System structure
- Means of integration

2.2.1.1 Functional Approach

This approach looks at the facilities or functions used in the office to form a picture of the office automation system, from which the office automation architecture is eventually established. It does not tend to identify the hardware/software needs of the office, nor does it tend to define the "geographical" structure of the organization at the stage of design.

The following main categories of conceptual office models can be classified, on the basis of the fundamental elements that they take into consideration [BRAC84] [HAMM80]:

- data based models
- process based models
- agent based models
- mixed models

Figure 2.11 and Figure 2.12 depict some well-known conceptual models, which are grouped into the above categories. Most of the models are formal, since they all provide the formal description of at least some of the office elements.

Data-Based Models					
Model	*Year*	*Type of model*	*Required analysis*	*Basic elements*	*Comments*
OFFICETALK-ZERO	1976	descriptive	tech. prot.	forms	minicomputer-based work-station
OMEGA	1980	highly formal descriptive	tech. prot.	forms	knowledge-based system
OFFIS (methodology)	1980	highly formal analytical	tech. prot.	objects attributes relations	system for office system design tech. methodology
OBE Office-By-Example	1981	highly formal descriptive (anal.)	tech. prot.	forms objects	supports many functions
Process-Based Models					
Model	*Year*	*Type of model*	*Required analysis*	*Basic elements*	*Comments*
SCOOP	1977	formal descriptive (anal.)	tech. prot.	procedures transition/ states	conc. asynchronous processes Petri nets + production systems
ICN Information Control Nets	1979	highly formal analytical	tech. prot.	procedures activities repositories	streamlining information struct. control structure
OAM Office Analysis Methodology	1980	analytical	decis. tech. soc.	procedures functions + resource	operational + tech. semistruct. activities methodology based on OSL
OSL Office Specification Language	1980	descriptive	decis. tech. soc.	application domain procedures	analyst oriented
Ticom-II	1981	formal analytical	tech.	task transactions repositories agents	graph-theory auditing methodol. min. cost for internal control
Office MAPS (methodology)	1981	descriptive	tech	functions process flow	organiz. struct. complete methodol. (organizational)
MOBILE-Burotique	1982	formal analytical	tech.	tasks-funct. + hierarchical levels- cost/ benefit an.	meta-methodology set of design tools (socio technical)

Figure 2.11: List of Conceptual Models

Agent-Based Models					
Model	*Year*	*Type of model*	*Required analysis*	*Basic elements*	*Comments*
Structural Model	1980	highly formal descriptive	tech. decis.	agents	office in form of functions+ personal databases of agents

Mixed Models					
Model	*Year*	*Type of model*	*Required analysis*	*Basic elements*	*Comments*
OFS Office Form System	1980	highly formal analytical	tech. prot.	forms-messages procedures	form=relation=message
IML Information Management Lang.	1981	highly formal descriptive	tech.	predicate/transition nets + inscriptions	control structure + data structure same nature
OPAS Office Procedure Automation System	1982	highly formal descriptive	tech. prot.	forms	sepecification through forms
Semantic Models	1982	highly formal descriptive	tech	office objects procedures	semantic model extension of Taxis
OFFICATALK-D	1982	formal analytical	tech. decis. prot.	forms procedures	OFFICETALK-ZERO + ICN
SOS Semantic Office System	1982	formal descriptive	tech decis. prot.	agents-documents dossiers-activities-rules	procedures+form based model rule based control

Figure 2.12: List of Conceptual Models

All models have the goal of describing office elements and activities, but some also offer features for guidance in restructuring activities in the office. The table also distinguishes between descriptive and analytical models. Descriptive models provide specification of office elements and office work, leaving the analyst to restructure office procedures, when necessary. The analytical models provide, together with the office description, some facilities for supporting automatic restructuring of office procedures.

A detailed description of each conceptual model follows.

Data-Based Models

Data based models tend to group data into forms, and hence, the office activities of this model can be seen as a series of operations on data. These forms are similar to paper forms in the traditional office. The basic elements of these conceptual models are types of data and the operation on data, such as storage, retrieval, manipulation and transmission.

An example of a data based model is *OFFICE-BY-EXAMPLE* (OBE), developed by Zloof [ZLOO81][ZLOO82]. OBE is an extension of a well known query language for relational databases (Query-By-Example) and is a language for describing and manipulating office objects of different kinds. The data structures on which OBE is based are two-dimensional objects, in the form of tables, which can be relations, forms, reports, hierarchical structures, documents, menus and so on. In a prototype OBE system [ZLOO82], several basic office functions such as word processing, querying, communication, data processing and document manipulation were defined.

Early conceptual office models were mainly data-based and supported the work of a single user at a time, connecting users through a communications network.

The main purpose of data-based models is to represent the office from the viewpoint of objects manipulated by office workers (agents), in a way similar to traditional offices, where work is primarily based on documents.

Process-Based Models

Process-based models analyze and describe office work by looking at different activities performed concurrently by the users and the system.

Zisman's [ZISM78] *SCOOP* (System for Computerization of Office Processing) is a process-based model based on Petri nets, augmented by production rules that models offices as asynchronous concurrent processes. An ***internal representation*** is a conceptualization of how the machine represents the problem. In addition, an ***external representation*** describes office procedures as activities and documents in a non-procedural programming language for the office analyst. The system, driven by an internal representation as input, tracks instances of procedures and automatically executes portions of them.

The goal of process-based models is that of representing office activities in a coordinated way. Thus, the approach is not founded (as in data-based models) on

operations performed by single users, but instead on an integrated vision of all the activities performed in an office in order to execute certain tasks, with the purpose of a general control of office work.

Agent-based Models

An office can also be modeled from the viewpoint of the functions performed by active elements of the office environment (the office worker or *agent*).

The *Structural Office* [AIEL84] is an agent-based model that describes the office by associating to the different agents a set of functions. The functions associated with the agents are the different roles that they take in performing their tasks, the domain within which they are authorized to act, and the set of relationships that link them to other agents. Every agent also has personal data in a personal database.

Thus, the description of office data and activities is dependent on a third element, besides data and processes, namely the set of office workers and their organizational structure. This model assumes that the actions that are automatically performed by the system are also considered as performed by particular agents.

This approach examines the office workers' roles and the delegation of roles in the office, while data and activities are considered only in relation to their executors. It is also easier, in this type of model, to describe the structural modifications in the office.

Mixed Models

Mixed models are based on more than one type of element as basis for system specification, with defined relationships between them.

The *SOS* (Semantic Office System) [BRAC83], is an example of a mixed model. SOS classifies office elements into three different submodels: The *static*, the *dynamic* and the *evolution submodels*. The static submodel contains specification of data related to office work, with the basic office elements being documents, dossiers and agents. The dynamic submodel contains the specification of operations and activities performed in the office. The evolution submodel specifies the normal evolution of office work and the possible structural modifications of office tasks through two sets of rules. Such rules could typically trigger the automatic execution of some operation, specify static and dynamic constraints on data and determine the authorizations to manipulate the data.

Most of the recent office models belong to the mixed model category, since it provides a more complete specification of different types of fundamental elements in the office and of their interrelationship.

2.2.1.2 System Structure

Instead of focusing on the functional activities of the office, the second approach reflects the general hardware/software structure used by an organization to imple-

ment its conception of office automation. One could argue that there are as many different architectures as there are organizations, but they all are variations of two fundamental elements:

- centralization
- total software system

From this one can state, for example, that a word processing package can be considered as a highly incomplete software system and largely decentralized.

While distributed processing is desirable in terms of speed and efficiency, it is better to model the office by a powerful central machine which can run the most sophisticated software, manage large databases, maintain mass-storage, drive high-speed printers and maintain control over the entire office system.

An alternative to this strictly centralized architecture is to have a more distributed structure where a number of minicomputers support many workstations and terminals. The minicomputers could then be linked by telecommunications facilities to permit sharing of resources and allow information exchange between the minicomputers.

A third alternative is based on the personal computer (PC) and is more decentralized in nature. Each office worker will have access to a PC, which will run a variety of office automation applications. These systems are essentially stand-alone and would have to be linked by a Local Area Network (LAN) to allow for communication between stations and sharing of resources.

This approach is totally opposite to the first method described. The system structure determines the office automation architecture and the hardware/software structure plays the dominant role in the design of the OAS.

2.2.1.3 Means of Integration

This third view of office automation concentrates more on architecture and less on approach, in that it seeks to provide a framework by which tools and technologies of office automation can be linked to provide an organization with an integrated information resource. Two main types of systems arise: ***centralized*** and ***distributed.*** The centralized system resides on a single computer and integration of the office facilities could be, for example, by menus or by forms, and the distributed system normally consists of isolated PC's that are linked together by LAN's to form an effective office system.

This approach combines the two methods described above, to create the most effective approach to OAS design. It is important that the functions and facilities proposed for an OAS are used to determine the OAS architecture, yet the "physical" structure of the OAS proposed cannot be ignored. These two approaches are effectively combined to determine the OAS architecture by concentrating on the integration of the two views to yield a complete OAS.

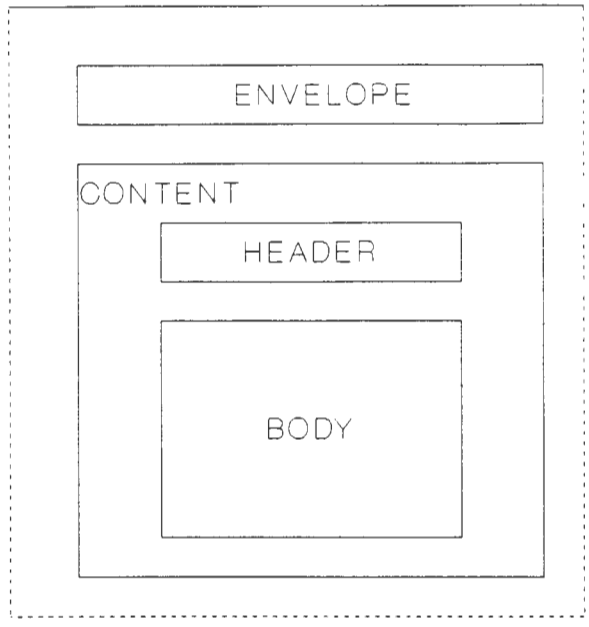


Figure 2.13: Basic Entity of Office Automation System

2.2.2 Standards

As the numbers of different office automation system increases, so does the need to be able to communicate across the multi-vendor products. This led to the development of standards to facilitate communication of office documents between office automation systems.

Presently various standards organizations (CCITT, ISO and ECMA) are investigating standards for interchange of office documents. Close inspection of the various recommendations for these standards shows that there is much similarity between them [HORA85]. The CCITT was the first of the three organizations to produce a draft standard.

Some of these standards are now discussed in detail below, with particular emphasis on the CCITT's *Message Handling Systems*. The work of ISO and ECMA is similar to that of CCITT, and their draft proposals will not be dealt with in great depth.

2.2.2.1 IBM Office Information Architecture

The IBM standards for office documents have been implemented within the framework of the IBM System Network Architecture (SNA). The term "document" used by IBM refers to any user created information, from messages to manuscripts.

A document is defined to consist of an envelope, a header and a body, as shown in Figure 2.13. This concept is the same, as proposed by CCITT.

The header contains formatted data representing the most important characteristics of the document, the envelope has all the delivery control information, and

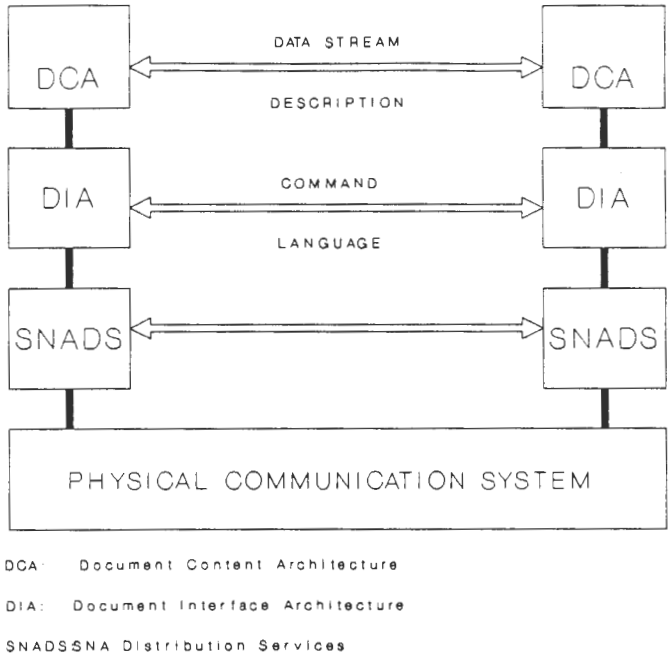


Figure 2.14: IBM Standards

the body consists of any combination of text, facsimile, graphics and other data structures.

The IBM standards have been implemented commercially and have become *de facto* standards. The IBM office automation system for the IBM series-370, DISOSS, conforms to these standards. PROFS, IBM's previous office system does not conform to these standards.

The three layer model of the IBM standards is shown in Figure 2.14. One could compare the model to the conventional letter, envelope and postman. The Document Content Architecture (DCA), which describes the contents of the document, could then be seen as the letter, the Document Interchange Architecture (DIA) as the envelope, and the SNA Distribution Service as the postman.

2.2.2.2 CCITT X.400 RECOMMENDATIONS

The formal name for X.400 is *Message Handling Service*. The scope of the X.400 recommendations is limited to simple messages. A document as defined by IBM standard would be seen as a complex message. Due to this limititation to simple messages, telex and teletex facilities are covered by X.400. The goals of the CCITT recommendations are to:

- promote a single worldwide standard for the Posts and Telecommunication administration
- promote a system for message exchange compatibility that includes computer based message systems, and CBMS and Telex/Teletex services

- allow public and private message systems, as well as interconnection between them

There are eight recommendations by the CCITT:

- **X.400:** System Model and Service Elements
- **X.401:** Basic Service Elements and Optional User Facilities
- **X.408:** Encoded Information Type Conversion Rules
- **X.409:** Presentation Transfer Syntax and Notation
- **X.410:** Remote Operation and Reliable Transfer Syntax
- **X.411:** Message Transfer Layer
- **X.420:** Interpersonal Messaging User Agent
- **X.430:** Access Protocol for Teletex Terminal

X.400 and the Open System Interconnection (OSI) Model

The message handling system fills the top layer of the OSI model (Application Layer). As in IBM's Office Information Architecture, the protocols dealing with the message contents and the transfer of the messages are distinct. The application layer is thus divided into 2 distinct sublayers:

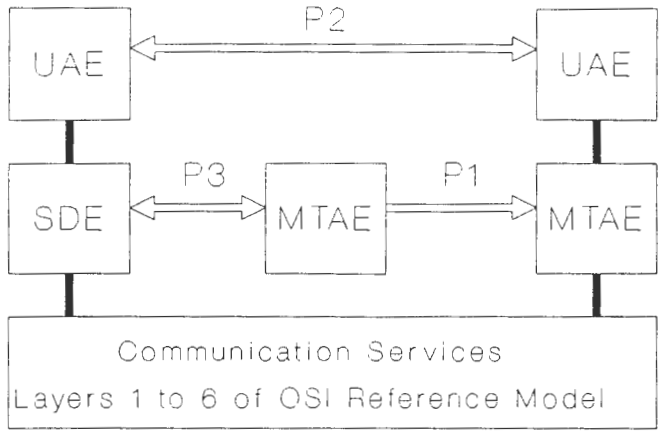
- the *Message Transfer Agent*, which specifies the standards for document interchange
- the *User Agent Layer*, which specifies the protocol for the content of the document.

In the lower layers of the model compatibility with Recommendation S.62 must be provided by the session services, to maintain compatibility with Teletex and Group IV Facsimile services. The Transport protocol class 0 is used, satisfying the requirement for a simple connection oriented transport service.

Message Handling Systems use a similar representation to the OSI seven layer model, but have deviated slightly to depict two protocols P1 and P3 on the same level, as illustrated in Figure 2.15.

DEFINITIONS

Before any of the Message Handling Services are described, a number of definitions and rules need to be discussed:



Protocols	Entities
P1: Relay	UAE: User Agent Entity
P2: Cooperating UA	MTAE:Message Transfer Agent Entity
P3: Submission/Delivery	SDE: Submission Delivery Entity

Figure 2.15: Message Handling Protocol Structure

X.400 System Model and Service Elements

The X.400 defines a Message Handling System that provides a store-and-forward service and allows asynchronous data transfer. The system consists of *User Agents*, *Message Transfer Agents* and the *Message Transfer Service.*

The *User Agent* provides means of creating messages to the user and assists in preparing and storing the messages.

The *Message Transfer Agent* (MTA) is responsible for the transfer of the created messages and a number of MTA's are defined as a *Message Transfer Service* (MTS) (Figure 2.16).

X.401 Basic Service Elements and Optional User Facilities

The Message Handling System service elements are defined as either *Basic* (mandatory) or *Optional* and consist of the Interpersonal Message Service or the Message Transfer Service.

The Message Transfer Service elements deal with submission and delivery of messages, queries, status and information and with conversion of message formats. The optional service elements are usually defined per message or for a certain time span.

X.408 Encoded Information Type Conversion Rules

The Message Transfer Layer is able to convert the content of a message (e.g., from text to facsimile) to be compatible with the recipient's terminal. The rules for converting from one type of format to another are specified by the X.408.

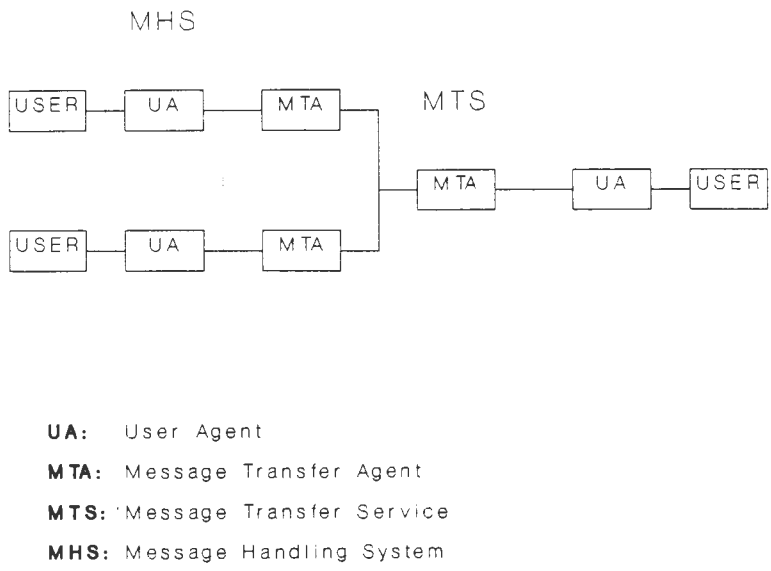


Figure 2.16: Functional View of CCITT's X.400

X.409 Presentation Transfer Syntax and Notation

X.409 defines syntax and notation for Message Protocol Data Units, User Agent Protocol Data Units and Delivery Operations Protocol Data Units. It also specifies the rules for defining the encoding syntax for the binary coded representation of the messages.

X.410 Remote Operations and Reliable Transfer Service

The Message Handling System needs to communicate with the lower layers of the OSI services. The Communication process is defined by the Reliable Transfer Service (RTS).

X.411 Message Transfer Layer

The Message Transfer Layer provides means for delivering a message to one or more User Agents. The services of the layer can be divided into two categories:

- services for transferring messages, and
- services for naming originators and recipients

Transfer of Messages

The message-transfer services are listed by category in Figure 2.17. The basic service is provided for each message submitted by a User Agent and includes message

delivery, the assignment of a unique message identifier and submission time stamp, and notification, should the message not be delivered.

The Message Transfer Protocol, P1 (Figure 2.15), is a protocol that defines the transfer of messages between two Message Transfer entities. In addition, P1 defines the interactions required to provide Message Transfer Layer Services. In this sense, P1 could be compared to IBM's SNADS.

Naming the Originators and Recipients

A prime objective of the message-handling work is to permit a recipient to be identified by name rather than by the address of his work station.

CCITT has specified a set of user attributes by which a potential recipient can be identified. In Figure 2.15, the Remote User Agent Access Protocol is shown (P3). P3 is also called the Submission/Delivery protocol and acts on behalf of the UA when interacting with the Message Transfer Agent Entity (MTAE). Normally, the workstation is remote from the MTA. The remote user is the Submission/Delivery Entity (SDE) and the protocol P3 is required to transfer the message from the MTAE to the SDE. The Submission/Delivery protocol thus performs similar tasks as IBM's DIA.

X.420 Interpersonal Messaging (Content) Protocols

The Interpersonal Messaging (IPM) service builds on the Message Transfer Layer and is provided by means of specific Cooperating User Agent Layer protocol called the IPM protocol or P2. The IPM service also encompasses internetworking with Telex and the Telematic message service as described by X.430 and X.408.

Service elements, as defined in X.400 and X.401 are shown as elements of the IPM Protocol Data Units (P2 Data Units). The X.420 defines two types of P2 Data Units:

- a message content that conveys an interpersonal message; it consists of a header and a body
- and a interpersonal message status report that needs no header.

The requirements of the Interpersonal Message layer are to [CUNN83]:

- provide services for communicating memoranda, the memo being the fundamental office document
- provide a framework for the message body, including messages from CCITT's Telematic services
- provide conversions for the exchange of messages among computer based message systems as defined by X.408.

SERVICE ELEMENTS
Submission + Delivery
Message Transfer
Message Identification
Submission Time Stamp
Delivery Time Stamp
Grade of Delivery Selection
Deferred Delivery
Deferred Delivery Cancelation
Non-delivery Notification
Prevention of Non-delivery Notification
Multidestination Delivery
Disclosure of Other Recipients
Alternate Recipient Allowed Request
Query
Probe
Status + Information
Hold for Delivery
Alternate recipient Assignment
Conversion
Content Type Registration
Original Content Type Indication
Content type conversion prohibition
Implicit Content Type Conversion
Content Type Conversion Indication
Explicit Content Type Conversion

Figure 2.17: Message Transfer Services

X.430 Access Protocol For Teletex Terminals

To maintain compatibility with standard PTT equipment, the CCITT recommendations include provision for Teletex terminals to provide these services to their users. The Teletex Access Protocol defines the standards necessary for using a teletex terminal in a Message Handling System.

2.2.2.3 ISO AND ECMA RECOMMENDATIONS

The recommendations of the ISO (International Standards Organization) and ECMA (European Computer Manufacturers Association) with regard to message handling standards are essentially the same [HORA85].

Both ISO and ECMA have prepared draft proposals for message interchange in 1985, a year after CCITT's X.400. The formal names are *Message Oriented Text Interchange System* and *Distributed Application for Message Interchange*, from ISO and ECMA respectively. Both proposals have as objective to achieve commonalty with the CCITT X.400 standards. In both cases a layered structure similar to the X.400 is adopted. The OSI set of service elements is slightly enlarged and its structure refers strictly to sublayers, unlike the CCITT's recommendation, whose P1 and P3 protocols occupy the same level.

The ECMA's *Message Transfer Services Access* replaces the CCITT's P3 protocol and the Message Transfer Protocol (X.411) and the Interpersonal Message Protocol is implemented with minor differences by ISO and ECMA.

ECMA defines the following sublayers:

- *Office Document Architecture* (ODA)
- *Office Document Interchange Format* (ODIF)

Office Document Architecture

The Office Document Architecture refers to the message content and an ECMA document can be interchanged either in an *Image Form*, to permit its being printed and displayed by the originator, or in *Processible Form*, to permit document editing and layout revision by the recipient. This is similar to IBM's Final and Revisable Forms for documents.

The contents of a document are represented by *Logical Objects*. These are typically titles, figures and footnotes. The document's *Logical Structure* is the grouping of logical objects in an hierarchical order.

Similarly, the *Layout Structure* is an hierarchical and sequential grouping of layout objects. The layout structure takes into account the dimensions of objects, and their type (e.g., title, figure etc.).

Objects are an important facet in ODA, and almost all concepts are represented by logical and layout objects. Objects of the same type are characterized into same object classes (e.g., layout object classes for frame type may be header frame, column frame and footer frame).

Document classes are defined in a similar way. Documents of the same class are grouped together as a document class.

A document may comprise of control characters, spaces and graphic characters as defined by the *Character Content Architecture*. It may have one of the following internal structures:

- *Processible form* - which applies to documents with only a logical structure. The content portions contain logical and shared control functions and may be divided by hard line terminators (carriage return CR and/or line feed, LF) into character sequences. These may be used to separate paragraphs without implementing paragraphs as objects.
- *Image form* - applies to documents with only a layout structure, the content of the basic layout objects contains layout control functions and shared control functions and is divided into lines by hard line terminators.
- *Formatted processible form* - applies to documents with both structures. The content is divided into lines separated by either hard line terminators or soft line terminators (device control string (DCS), carriage return (CR), line feed (LF), and string terminator, (ST)).

Office Document Interchange Format

The formal specification contained in the ODA standard is based on the *Abstract Syntax Notation*, ASN.1, defined in ISO 8824, which is essentially the same as the Standard Notation of the Presentation Transfer Syntax specified in CCITT Recommendation X.409/12/.

In this syntax, each item of information is viewed as a data item of a specific data type with a specific data value. The syntax notation is used to define the format of the data flow and its components as a SEQUENCE or SET of elementary data types that are, in turn, defined by means of more elementary data types. This nested specification ends up with basic data types such as *INTEGER* and *OCNET STRING*.

2.2.2.4 CONCLUSION

Document interchange standards are becoming an important part in the design of office automation systems. Designers of office automation systems must take the new developments into consideration and build their systems to take advantage of the extra network connectability that is becoming available.

The CCITT places large emphasis on international office document exchange and many other standards organizations such as the ISO and the ECMA are developing standards that aim to conform with the CCITT's X.400.

IBM has developed an architecture which standardizes office document interchange and has been successfully implemented. This architecture has been adopted by at least one vendor and although the office automation system products of the

various vendors may not be compatible, the degree of compatibility is increasing, with fully standardized multimedia office communications being the ultimate goal.

Chapter 3

SOME OFFICE AUTOMATION SYSTEMS

Office automation systems are essential to most businesses and as a result there exists a large variety of office automation products. It is impossible to mention all available office automation systems and hence a few have been selected for the purpose of this study.

The systems chosen are *Office-by-Example* (OBE) [ZLOO82], *All-In-One* [DIGI], *Proffesional Office System* (PROFS) [IBM82], and *Uniplex* [UNIP88].

The following sections will attempt to isolate commonalities of these systems and try to establish the areas in which the systems differ. These sections will look at each of these systems concentrating on the design issues mentioned in Chapter 2, namely on

- Type of Model
- System Components
- User Interface
- Standards (where applicable)

3.1 OFFICE-BY-EXAMPLE

Office-by-Example (OBE) is an integrated office system that has been under development at IBM Research. OBE extends the concept of Query-by-Example (QBE) [ZLOO77], a relational database language. It supports various features needed in a typical office environment, such as text processing, electronic mail, menus, database tables, forms, graphics and images. OBE supports the addition of new features to the environment and integrates them through a language called *example elements*. A database manager constitutes the backbone of the OBE system that maps and processes the different features that involve example elements.

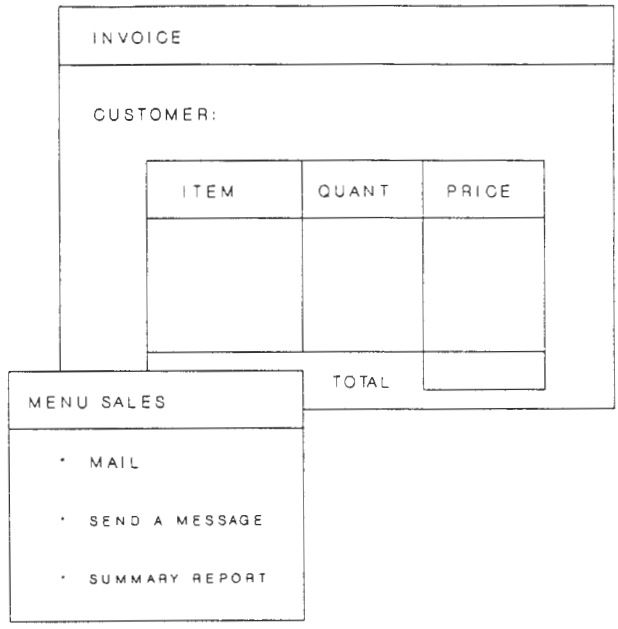


Figure 3.1: Various OBE Objects

3.1.1 Approach and Architecture

OBE is a *data based model* and was designed using the *functional approach* to office automation systems. The language for OBE attempts to integrate the various aspects of word processing, report writing, graphics and electronic mail. The novelty of OBE language is that menus and applications are not preprogrammed by the system developers; they are customized and programmed according to specification by the users. The programming style of OBE deals with programming within two-dimensional pictures of business objects, which include letters, forms, reports, charts and graphs, some of which are illustrated in Figure 3.1.

3.1.2 User Interface

The OBE system supplies the user with a number object and activities manipulation commands. With the help of these commands, menus, forms and tables can be created to present a new higher level interface.

An on-line help facility describes actions that a user can perform. The help system displays a scrollable help box in one half of the screen so that it does not overlap the selected element. The user then can simultaneously see the help information and manipulate the context.

3.1.3 System Components

OBE forms a tightly integrated office environment and many functions are shared by the different components of the system. The various system components of OBE are briefly discussed below:

3.1.3.1 Word Processing

OBE is a two-dimensional, object oriented language that uses the screen manager (SM) to support and manipulate the two-dimensional objects such as *multiple windows* and *multiple objects within a single window*. The screen manager also supports the editing of objects and text within the objects by using predefined function keys **EXPAND**, **ERASE**, **MOVE**, **LOCATE**, **ZOOM** and **PUSHDOWN**.

The limitation on the number and sizes of windows and objects that one can simultaneously display on the screen is that of the virtual address space of the machine. Whenever too many objects fill a single window, one can use the **LOCATE** function key to locate each object separately in sequential order.

The screen manager performs most of the text editing operations by using the function keys described above. Text formatting is also handled by the screen manager and is interactive. Each time the **ENTER** key is pressed the document is reformatted.

3.1.3.2 Mail

If an object is sent to other nodes in the system, or if an object is received from other nodes, the *communication manager* interacts with the underlying operating system to distribute the object according user specifications.

A user may specify the following options:

- acknowledgement
- confidential mail
- hard copy (print option)
- log (write mail to a log file)

The *trigger manager of* OBE may be used to invoke the following actions relating to mail messages:

- send messages at predetermined dates and times
- send acknowledgements
- send a reminder if correspondence is not answered within a given time

The mail can be distributed by specifying the recipient user's ID. Distribution lists may be predefined or can be found by the system upon issuing a query to the database at the time of the distribution. For example a message could be sent to all salesmen who exceeded their quotas in the previous month. Such a mailing list would be produced by a database query.

3.1.3.3 Document Facility

A *report* is regarded to be an *output object* and is stored globally as text. The structure of a report is not stored in the system, thus data cannot be captured via a report and it may not be queried as is the case with *forms*. A text *document* in OBE is regarded as a report containing only text without any formal structure.

The document can be retrieved by name (just like tables, forms, etc), or it can be retrieved by the text content. OBE provides QBE's *partial-example-element* feature for searching by a word or sentence within the body of the text.

One could, thus, request a count of all documents that contain the words "office automation", or request to display these documents on the screen.

Access to documents is determined by the creator, who can specify *read* , *insert*, *delete* and *update* options for documents. Whenever a user issues the request to access or modify a database that may contain tables forms, documents, etc., the screen manager passes the request to the authority manager, which checks its validity. Access to the object may thus be denied by the system.

3.1.3.4 Administrative Support

Users may define their own databases. These databases can be defined either as collections of relations or as hierarchical views of relations.

Although no scheduling facilities have been provided by OBE, they can be created by the end user. A calendar facility can be created by defining a new database with the corresponding objects. Schedules could then be retrieved by issuing a query to the database. Access to the schedules would be defined by the owner of the calendar and controlled by the authority manager .

Management support functions can be implemented by forms and reports, which can be used to generate tables that display sales figures, share prices etc.

3.1.4 Standards

OBE is still in the prototype stage of development and the designers seem not to have concentrated on mail and document standards at this stage.

Although provisions are being made to interface OBE to different database systems, it does not seem that mail can be sent to any environment outside OBE.

The designers do not mention whether the document structure conforms to IBM's *Document Content Architecture* (DCA) and *Document Interchange Architecture*, as mentioned in Section 2.2.2.1. OBE relies on the host operating system (VM/CMS) for file management.

3.1.5 Conclusion

OBE supports office features such as word processing, electronic mail, document storage and database tables. Powerful tools exist to create new menus and databases thereby allowing the users to describe their new applications to the computer. Yet, I feel that because the user interface of OBE office functions first needs to be created

to make the system acceptable to personnel with limited computing exposure, users with no database experience could encounter difficulties with the interface.

3.2 ALL-IN-1

ALL-IN-1 is an office automation system designed by Digital Equipment Corporation. The ALL-IN-1 system combines many of the Digital office automation products into one large, integrated system. The actual components of ALL-IN-1 are selected by the customers according to their office needs. The office automation model thus can simulate the different office structures as closely as possible.

3.2.1 Approach and Architecture

ALL-IN-1 uses the *means of integration* approach to view the office. Both functionality and hardware architecture is taken into consideration to reflect the office structure. The entire system is constructed by integrating the individual facilities. The addition of facilities can happen at any stage, so an office system can be gradually upgraded.

3.2.2 User Interface

The ALL-IN-1 office automation system a *menu* and *form* based user interface to access and manipulate the facilities. An example of such menu is given in Figure 3.2. Simple tools allow the user to customize the menus and forms to suit the user's needs. Extensive online *HELP* facilities give immediate assistance to the user explaining the commands.

ALL-IN-1 also provides a *voice mail* facility. Voice input is converted and stored in a digitized format, thus allowing the user to access the mail system by telephone. The voice commands are not as extensive as the mail features provided by the menu interface, but nevertheless, an extremely usefull service not offered by OBE. More about voice mail will be said in the discussion of ALL-IN-1's mail.

3.2.3 System Components

The different office automation facilities can be selected by the customer and thus no ALL-IN-1 office system is necessarily the same. A brief description of the available facilities is given below.

3.2.3.1 Word Processing

Numerous word processing packages exist, unlike OBE that provides only one word processor. All word processors utilize the same editing interface to minimize retraining, should a word processor be replaced by a new, more powerful product.

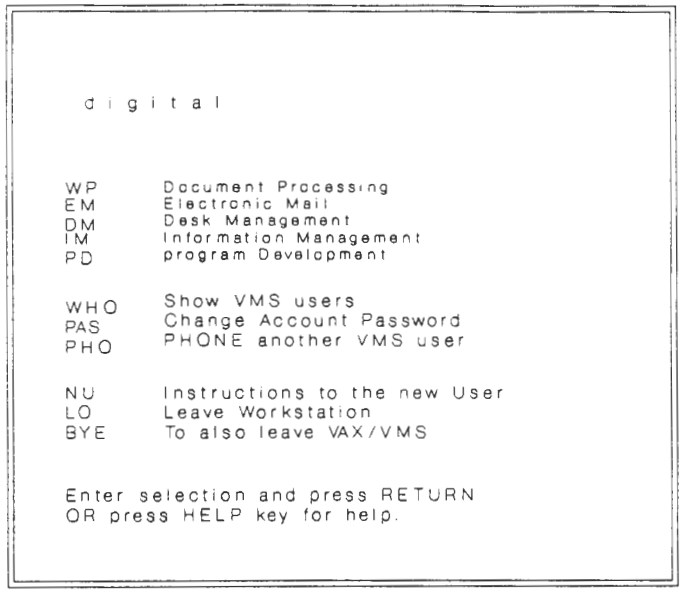


Figure 3.2: ALL-IN-1 User Interface

3.2.3.2 Mail

The ALL-IN-1 electronic mail provides the usual *read, create, send, file* and *print* mail functions, just like OBE. ALL-IN-1 has a few facilities not supported by the two office automation systems discussed before.

- The *answer* facility lets the user send a reply to the message currently being inspected. The system automatically enters the addressing information, such as username and date, and sends the reply after the user completed the message. The user can proceed to inspect further mail items in his mailbox. The answer facility is virtually an acknowledgement as described in Chapter 2, with a message included.
- The ALL-IN-1 electronic mail system also supports an *interrupt* facility which allows the user to execute any other ALL-IN-1 facility, and return to the mail system, to carry on with the task that was interrupted.
- The ALL-IN-1 electronic mail system can be accessed by a telephone: messages can be *read, filed, stored* or *answered* by a pre- determined message. Such a facility provides the user with an access to the mail system without the need of a terminal.

3.2.3.3 Electronic Filing

ALL-IN-1's electronic filing system uses the "cabinet" filing structure. The documents are stored in folders and may be retrieved either by *name* or by *keyword.*

No retrieval by document content is possible as in OBE. Access to documents is determined by the owner of the document; no general access control exists.

Once a document is retrieved by the electronic filing system, the user may

- *distribute* (send by the mail system)
- *edit*
- *delete*
- *print*

the document. These functions are very similar to those of OBE.

The user also has the use of folder operations such as *create* and *delete* that are used to manipulate individual cabinet structures.

3.2.3.4 Administrative Support

ALL-IN-1's administrative has a number facilities that are very useful to office workers and management:

- Scheduling of meetings
- Electronic diary
- Resource Scheduling
- Telephone Directory
- To-do-list

The electronic diary is consulted by the systems whenever a meeting is to be arranged and a convenient time slot is chosen. The users concerned are notified of the meeting via the mail system and an entry is is created in their electronic diary.

OBE does not provide any scheduling or diary facilities.

Other electronic office tools such as *desk calculators, spreadsheets* and *financial planning packages* can be integrated to ALL-IN-1 as part of the office automation system. These tools can be selectively added to the system, depending on the requirements of the office.

An *ICON* based graphics interface can be used for some of the business applications used by ALL-IN-1.

3.2.3.5 Standards

The *Internet* protocols supplied by Digital systems allow the connection to other processors using the same protocol. This currently includes manufacturers from IBM, Central Data Corporation and Unisys. Support is available for Batch BISYNC, Interactive BISYNC and SNA [IBM82] communication.

Documents may be exchanged between ALL-IN-1 and the WANG OIS by means of the *External Document Exchange*.

3.2.4 Conclusion

Because of its ability to cater for the different office environments, ALL-IN-1 is a very efficient office automation system. The voice mail facility adds a very attractive feature to the already rich office automation environment.

3.3 PROFS

PROFS (Professional Office System) is an office automation system developed by IBM. PROFS runs on the VM/CMS operating system and is extensively used at IBM installations throughout the world.

3.3.1 Approach and Architecture

PROFS is a mixed model office automation system, and was developed and designed for the VM/CMS operating system using the *means of integration* approach. PROFS supports a distributed architecture and the PROFS data files are often stored on one central mainframe computer.

3.3.2 User Interface

PROFS is menu, form and command driven. The menus and forms provide the inexperienced user with a simplistic interface to PROFS's facilities; the more experienced user may ignore the menus and use commands to activate the different facilities.

The menu options are selected by pressing the *Program Function* keys on the keyboard. Figure 3.3 shows an example of a PROFS menu screen.

An on-line *help* facility can be activated by pressing the **HELP** key, which is consistent throughout the menus (as in ALL-IN-1).

The main menu of each application displays a calendar of the month, and highlights the current day. Each menu has a level number on the right hand side of the screen to indicate the logical level of the menu.

The display of the menus can be customized by the users to meet their individual requirements.

Personally, I found PROFS menus a little tedious to operate: The menu option contains a detailed description of the function and I feel that a new user might feel intimidated by too much detail on the screen. This can be seen in Figure 3.3. The user can not deduce the different functions presented by the screen just by glancing at the screen, instead each option must be studied carefuly, thus wasting time.

3.3.3 System Components

PROFS supports many of the facilities already seen in ALL-IN-1. Although PROFS does not support the addition of new application to the extent that ALL-IN-1 does,

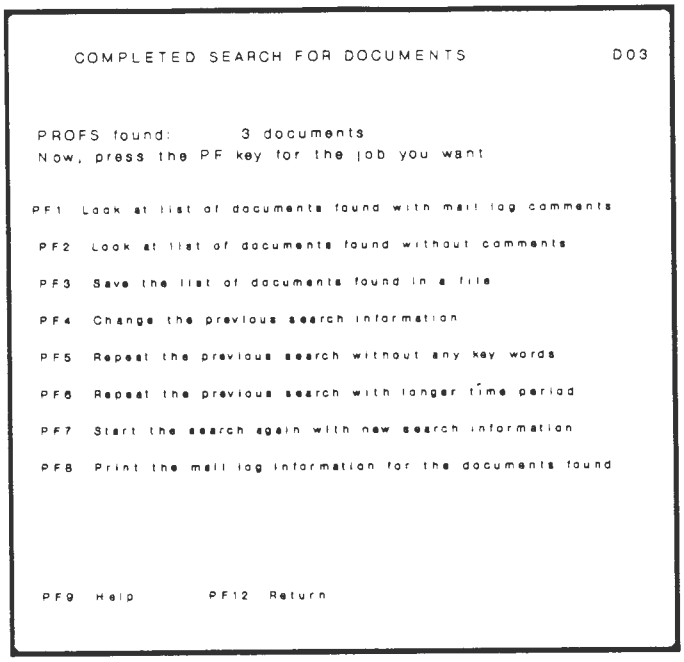


Figure 3.3: PROFS User Interface

documents and data files created at the operating system level can be distributed and stored with PROFS.

3.3.3.1 Word Processor

PROFS provides two types of editors. Firstly, a primitive text editor is used to create notes and messages. This editor does not support many editing commands, as opposed to the PROFS word processor that has all the standard features. The ordinary user who is possibly only interested in sending and inspecting mail, does not need to know the word processing commands in order to send a simple message.

3.3.3.2 Mail

PROFS mail features are similar in most aspects to the features seen in ALL-IN-1.

One major difference is the display of mail: The user may ask the system to display only those mail items that had not been inspected before, i.e. new mail only. Such a feature is very convenient, since many users tend to leave mail items pending in their mailboxes. This causes the old mail being unnecessarily displayed, often causing the new mail to be undistinguishable.

CEO (Comprehensive Electronic Office) informs the user of new mail arrival, only if the message is urgent. PROFS has no classification of mail items, and displays a "new mail" message for every arrival of a mail item. Such a feature can be distracting on large system, since the frequency of newly arrived mail increases with the numbers of users on the system.

3.3.3.3 Electronic Filing

The PROFS filing system differs significantly from that of ALL-IN-1. PROFS distinguishes a *note* from a *document*.

A *note* is considered a short message or article and is stored in the user's directory. Multiple copies of a note are created whenever a note is sent to multiple destinations.

A *document* is considered to be larger than a note, and often contains formatting information generated by the word processor. Whenever a document is distributed, only a pointer to the document's location is passed. The storage technique is similar to signature files and is clearly different to the file organization used by ALL-IN-1.

Document retrieval is also different from the methods used by ALL-IN-1. The conditions for the retrieval of a document can be combined to form a list of constraints which can be either conjunctive or disjunctive, very much like the "query-type" retrieval of documents of OBE.

3.3.3.4 Administrative Support

The administrative functions of PROFS perform the same tasks as those of ALL-IN-1, so I will not discuss them here. The main difference between these two office automation systems is the visual display of the facilities. Having had the opportunity of viewing CEO and PROFS, I prefer the monthly calendar display of CEO. The PROFS monthly display is very cryptive and is not easily understood.

PROFS does not directly support the addition of new applications. PROFS main aim is to provide the user with a mailing, scheduling and document handling facilities.

3.3.4 Standards

IBM standards for office documents have been implemented within the framework of the IBM System Network Architecture (SNA).

Since many installations throughout the world use PROFS, messages and documents can be routed to all the installations connected by a communication network. IBM seem not to have made an extensive effort to conform to other vendor's standards as ALL-IN-1 or CEO have.

3.3.5 Conclusion

PROFS provides the user with an efficient office automation environment with powerful mail, document and administrative features. PROFS does not have the integrational qualities of ALL-IN-1 and CEO, when it comes to new applications.

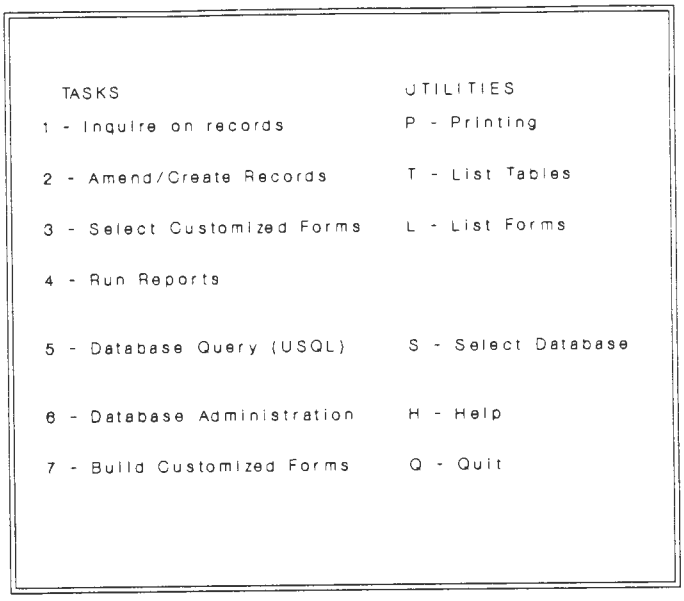


Figure 3.4: UNIPLEX Menu Screen

3.4 UNIPLEX

UNIPLEX [UNIP88] is a UNIX based office automation system, that provides the UNIX user with a set of office automation functions. UNIPLEX relies on the UNIX operating system for file management and uses the existing security features for its document access.

UNIPLEX provides the office automation system user with similar features to those of PROFS, CEO and ALL-IN-1. The reason for the inclusion of UNIPLEX in this study is to be able to compare another UNIX based office automation system with COSNET. I will thus discuss UNIPLEX only in context of the user interface and a few important features.

3.4.1 Approach and Architecture

UNIPLEX uses a functional approach to define the view of the office, and supports a centralized architecture. UNIPLEX runs on any computer under the UNIX operating system and has primitive facilities for networking.

3.4.2 User Interface

The UNIPLEX user interface consists of *popup menu* screens which are divided into *tasks* and *utilities* as seen in Figure 3.4. The TASKS column displays the number of operations that a user may perform from the current menu and the UTILITIES indicate a set of system facilities available.

A menu option is selected by entering the respective number or the first letter of the desired option. The *help* and *quit* keys are consistent throughout all the menus.

3.4.3 System Components

UNIPLEX does not offer any new facilities above those already mentioned. Some minor differences in the presentation of the facilities do exist, however, and will be discussed below.

3.4.3.1 Mail

Two new facilities not supported by the above systems are part of the the UNIPLEX mail system:

- UNIPLEX mail can be assigned a priority level which is used by the system to determine the order in which the messages are delivered
- The *notify* option defines the number of days after which UNIPLEX sends a *non-read* notification. This facility is an alternative solution to an acknowledgement: - it is easier to be notified that mail has **not** been read, then waiting for acknowledgement, since one could easily forget to anticipate the acknowledgement.

3.4.3.2 Electronic Filing

UNIPLEX uses the "filing cabinet" structure for electronic filing. Access to a document is directly based on the UNIX file system security, shown in Figure 3.5. The user may change the read, write and execute permissions for each specific file or folder as shown in Figure 3.5.

Folios are special information files that may be optionally created. A folio file contains information pertaining to the retrieval of the file illustrated in Figure 3.6.

3.4.3.3 Administrative Support

UNIPLEX does not provide any resource or meeting scheduling facilities as seen in PROFS, CEO and ALL-IN-1, but an electronic diary is available.

A financial spreadsheet, a sketchpad and a SQL based database system are all part of the UNIPLEX applications for management support. UNIPLEX is not as flexible as ALL-IN-1 and does not allow new applications to be added to the system.

3.4.3.4 Standards

UNIPLEX currently supports communication with other UNIPLEX distributed machines. This is achieved using the UUCP utility and is not very well supported as it only allows mail transfer. No other communication tools exist for the exchange of mail or documents.

	READ	WRITE	EXECUTE
Owner	[Y]	[Y]	[Y]
Group	[Y]	[Y]	[Y]
Other	[Y]	[Y]	[Y]

Figure 3.5: UNIX File Access Permissions

```
Document        :[________________________]
Title           :[________________________________________]
Folder          :[/usr/current folder_____________________]
Owner           :[jones_____________]
Category        :[__________________]
Filetype        :[WPDOC]Word-processing Document
Created         :08/12/89 : 11:00
Revised         :08/12/89 : 12:00
Keywords        :[_____________________________]
                 [_____________________________]
```

Figure 3.6: UNIPLEX Folio

3.4.4 Conclusion

UNIPLEX offers a full compliment of necessary office automation system functions. It is a centralized system that does not provide any communication tools to non-UNIX systems and makes no provision to include new applications. One must, of course, not forget that UNIX is relatively new when compared to long established mainframe computers that run systems like ALL-IN-1, CEO and PROFS. It should not take too long for systems like UNIPLEX to reach the same level of sophistication as for example PROFS.

3.5 CONCLUSION

From the discussion of the office automation systems above, the following similarities are apparent. All the system provide tools for creating and editing documents, and support electronic mail and document handling facilities.

The large commercial office automation systems supply word processors that support a complete range of document processing facilities, whereas the prototype systems, such as OBE, often provide the user with less powerfull editing facilities. The office automation system should support a *simple editor*, for creating short messages and notes, and at the same time should also provide a *word processor* for the user that needs to create and update documents.

The *document filing* facilities of the office automation systems discussed do not differ significantly. All the systems provide some means for storing and retrieving a document as well as defining access permissions for documents. The concept of sharing documents amongst users is supported by all the systems and strict security control of document access is provided.

The implementation of these document filing facilities is different in each of these systems described, but this is of no consequence to the users, since only the conceptual facilities are visible to them.

Electronic mail should allow the user to *read, file, forward, delete* and to *print* a message, features which are supported by all the office automation systems mentioned above. Additional features like the *classification* of mail items or the *reply* facility exist in some office automation systems and are valuable, additional mail functions.

The *administrative support* requirements vary from office to office; a small business would possibly only require an electronic diary facility and a telephone directory, whereas a large company would need additional facilities such as *scheduling of meetings* and *resource scheduling*.

The administrative functions of the office automation systems discussed above are supported in different levels: OBE provides no basic administrative support, UNIPLEX supports an electronic diary and All-In-1 and PROFS provide the user with a large set of administrative functions.

An important feature that determines the flexibility of an office automation system, is the ability to selectively add new applications to the system. Such a

facility allows the user to customize the office automation system to simulate the office functions more accurately.

The *user interface*, which is one of the most important facets of the office automation system, consists mostly of menu screens, as seen in the above discussions. Such menus give the user a simple and convenient access to the office automation functions. All-In-1 and PROFS are two office automation system that provide a user interface, alternate to the menu user interface.

The need to facilitate communication of office documents and messages between office automation systems is reflected in some of the systems discussed. Communication programs adhering to standard protocols provide the necessary tools to exchange documents and messages between different office automation system. In the future, ideally, it will be possible to exchange documents and messages between all office automation systems.

Chapter 4

COSNET

Before I proceed to discuss COSNET, a short discussion about the computing facilities of the environment for which it was designed has to be included.

Users are connected to a NCR 1632 UNIX TOWER by either dumb terminals or Personal Computers that act as a terminal emulator. Most of the users either have a Personal Computer in their office or at least have access to a Personal Computer.

The UNIX 1632 TOWER is linked to an ETHERNET network and communicates with two NCR 600 TOWERS, also utilizing the UNIX operating system, using XNS protocols. Figure 4.1 shows the topology of the computing network.

The administrative office utilizes Personal Computers to execute all the administrative tasks and no access to to the NCR 1632 TOWER is currently available. I will refer to the NCR 1632 TOWER as the *server*.

From the current computing configuration it can be clearly seen that the administrative office is "isolated" from the other staff members, who can at least use the UNIX mail facility for communication. Similarly, the UNIX file system can be made use of, to share documents between staff members, although no word processing facilities are available.

To improve the efficiency of the administrative tasks, staff members and administrative staff both need access to a mail system and a document filing facility. Since most of the administrative tasks are performed on Personal Computers, we realized that the workstation of the system should idealy be a Personal Computer that must be able to utilize its applications.

An inspection of the hardware configuration suggested two architectures that would suit the office automation system model:

1. The ETHERNET network could be used for a LAN based office automation system

2. The staff machine could be used as an office automation system server

The first proposal required new hardware for each Personal Computer in order to be able to utilize the ETHERNET network.

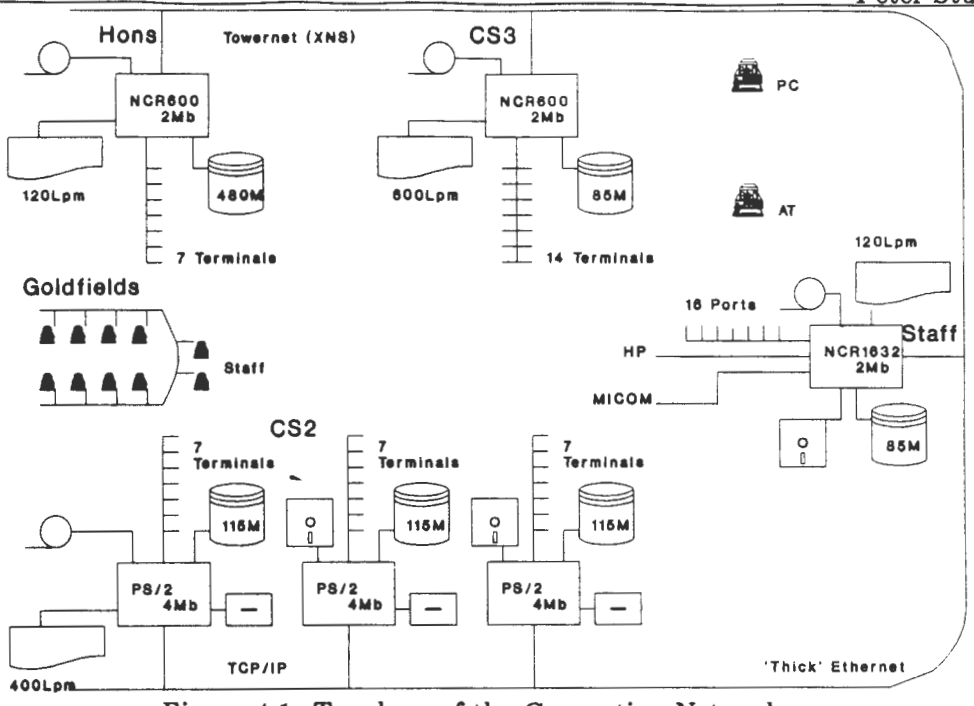


Figure 4.1: Topology of the Computing Network

In the development stages of the prototype office automation system COSNET it was decided to use existing hardware, and thus to use the NCR Tower as the office automation server, and the existing Personal Computers as workstations.

4.1 DESIGN OF COSNET

The design of COSNET was based on a model of a typical academic department.

During the *requirements analysis* of the design of COSNET it became evident that the administrative office needed to improve communication with other staff members. No means of electronic messaging and document exchange was possible and all communication had to be performed telephonically or manually using paper notes.

Although many mail items originate externally to the department, it was observed that an extensive document and message flow between the administrative office and academic staff members existed:

1. Telephone messages were written out on a memo note and distributed by hand to the recipient.

2. Notices of seminars were regularly distributed to all the staff members by hand.

3. Letters that needed to be sent were first created by the sender and later passed on manually to the administrative office, where the letter was edited and mailed.

4. All the course marks were kept in the administrative office. A course chairman had no access to this information after hours, and often time was wasted making copies of the documents.

5. Tutorials, class test and the solutions were kept using a paper filing cabinet and folders in the administrative office. Hence, no easy access was provided to such documents.

6. The secretary managed the Head of Department's schedules. A duplicate set of diaries were kept, often causing inconsistent entries.

These observations suggested that an electronic document, mail, and calendar facilities would improve many manual tasks in such a working environment.

All the staff members, including the administrative staff had "computing experience". Everyone was thus accustomed to working with a computer and adaptation to a new office automation system should not pose too many problems.

4.2 IMPLEMENTATION

Since the Personal Computers required networking cards to be able to connect to the ETHERNET, the idea of a LAN based office automation system could not be considered. This meant that the NCR 1632 Tower had to be used as the office automation system server and to provide the backbone of COSNET, resulting in a distributed architecture.

The UNIX system is used only for storage of documents, calendars, mail items and the office automation system administrative functions. Most of the processing is thus performed by the workstation (Personal Computer). System files are transferred from the server to the workstation, using KERMIT [KERM88], as soon as a connection is established, (i.e. the user logs onto COSNET) and returned to the server whenever the user exits COSNET.

To maintain consistency of all the shared files, the system retrieves application control files only when the specific facility is requested by the user. This mechanism will be discussed in more detail in a later section.

A menu user interface provides simple access to COSNET's facilities. No training is needed to operate COSNET and new staff can quickly adapt to COSNET's user interface, as will be shown in Chapter 5.

4.3 COSNET HELP

COSNET's *help* facility is invoked by pressing the function key **F9** anywhere within COSNET. A *pop-up help menu* appears in the middle of the screen, displaying the help items which are available from that particular help menu. A sample COSNET pop-up help menu is shown in Figure 4.2.

The user can select a help item by positioning the cursor on the desired item followed by pressing **RETURN**.

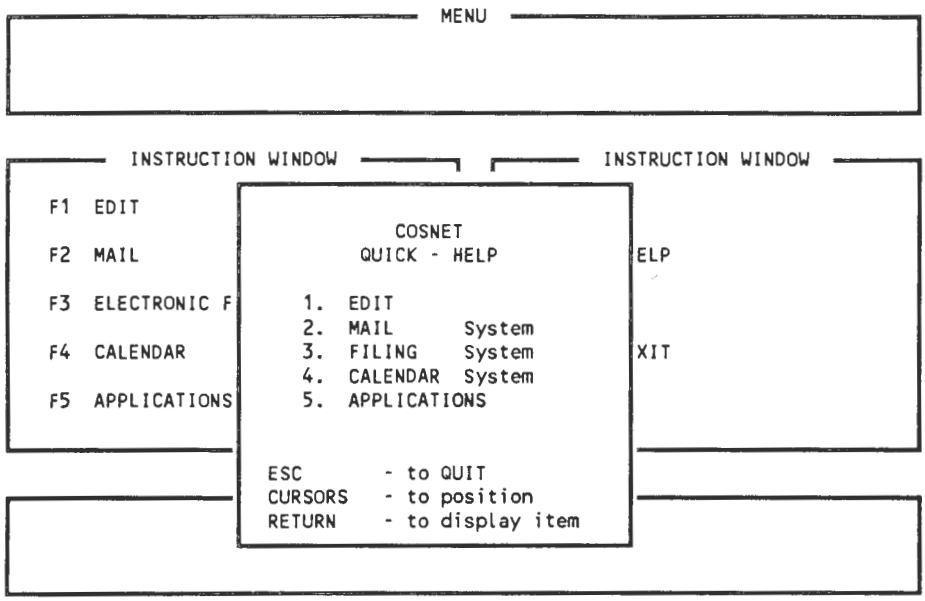


Figure 4.2: COSNET HELP Selection Menu

A *help window* relating to the item selected is then displayed. This help window covers the entire screen and the user may scroll up and down through the displayed text. A command line at the bottom of the help window displays the commands that are available to the user. An example of a help window is illustrated by Figure 4.3. As soon as the user exits this help window, the display screen is restored to show the pop-up help menu. COSNET always restores a screen to its original form after a pop-up window is erased.

The user may request the help facility from anywhere within COSNET, thus allowing access to help information anytime it is needed.

4.4 ELECTRONIC MAIL

COSNET's electronic mail provides the user with five facilities, as can be seen on the COSNET MAIL menu in Figure 4.4. A COSNET user may create and send mail, inspect mail, create nicknames, inspect his mail-log or inspect previously stored mail. These functions are all supported by office automation systems such as All-In-1 and PROFS, as described in Chapter 3. COSNET also supports classification of mail items, very similar the the classes of OBE, described on page 57.

4.4.1 Functionality

COSNET uses the user identification to route messages. All the mail is stored on the server and incoming mail is simply appended to the user's mailfile. COSNET does not make use of the UNIX mail facility.

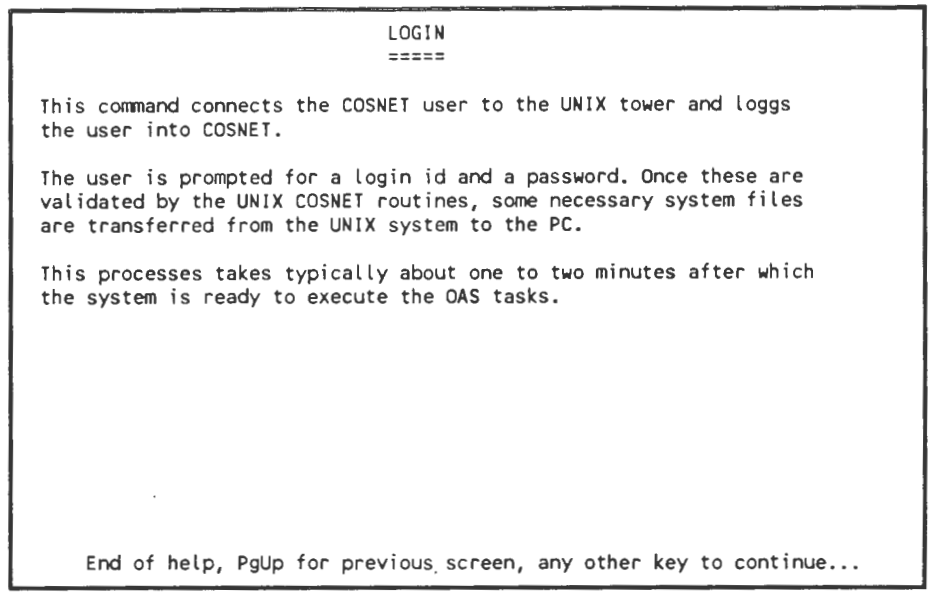


Figure 4.3: COSNET HELP Window

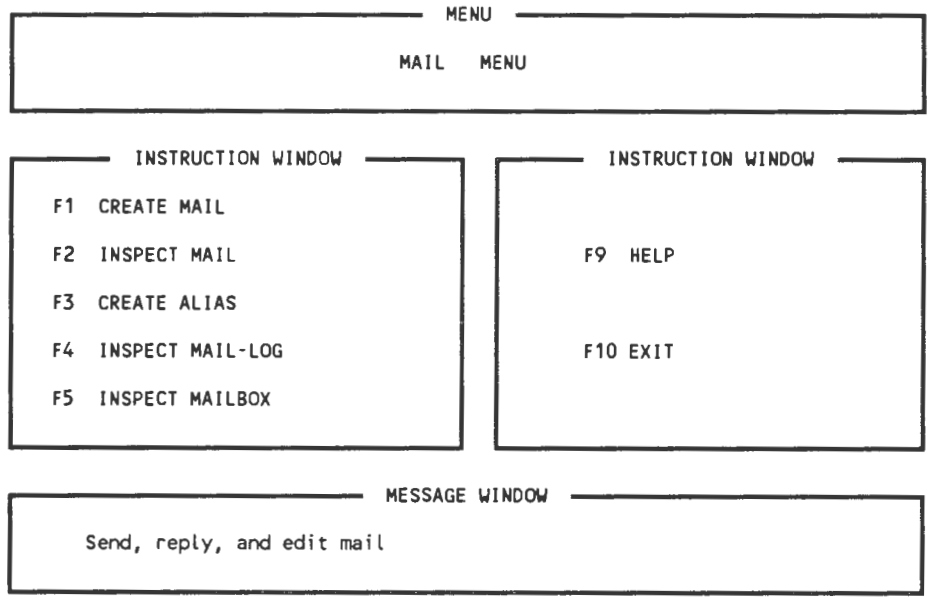


Figure 4.4: COSNET Mail Menu

Whenever a COSNET user decides to inspect his mail, the user's mailfile is transferred to the workstation, where its contents are read into a dynamic data structure. This data structure is a linked list of records, one record for each mail item. The original mailfile on the server is then deleted.

The arrival of new mail during the inspection of the mailfile is detected by COSNET, and a temporary file is created in the user's mail directory. The arrival of new mail is signalled as soon as the mailfile is returned to the server and the user is informed that new mail has arrived. The temporary mail is appended to the mailfile, which is then ready for inspection.

COSNET also tests for the arrival of new mail whenever the COSNET main menu is entered. Communication between the workstation and the server is unidirectional and all communication commands must originate from the workstation. It is thus not possible to send a signal from the server to the workstation to inform a user of new mail; the only way to detect new mail, is to issue the command to inspect the mail directory from the workstation.

4.4.2 Create Mail

COSNET automatically fills some fields, such as the *sender* and *date* fields, in the mail header information form. The user has to enter the other header information, namely the

- subject (blank by default)
- urgent (no by default)
- acknowledgement (default no, if message not urgent)

or simply press **ENTER** for the system default values. Once the header information is completed, the user may type the message using the simple editor supplied by the COSNET mail system. An example of a new mail item is shown in Figure 4.5.

An interrupt facility allows the user to stop editing the mail message, select another task from the main menu, and later return to the *create mail* option. The message is restored exactly to the same state it was in before the interrupt occurred. COSNET assumes that a new message is to be created should the interrupt option not be used, and proceeds to load a blank message form into the editor.

Once complete, a message can be distributed in numerous ways. Figure 4.6 displays the options available to the user.

The message can be sent to a single user at a time, by selecting the first menu option. A list of legal COSNET user names is displayed and the user is prompted to enter a username. COSNET verifies the entry and proceeds to send the mail item to the user specified.

The *groupfile* option can be selected whenever a distribution list needs to be used to send the mail item. Each user may create any number of distribution lists that contain legal COSNET user names. This menu option allows the user to send the mail item using such a list, or to create, delete and update distribution lists.

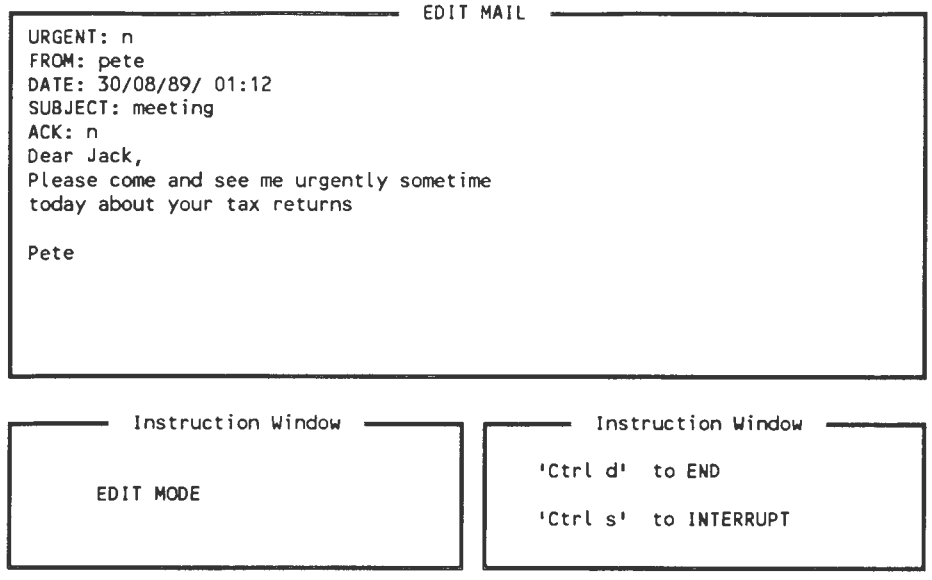


Figure 4.5: COSNET Mail Creation Screen

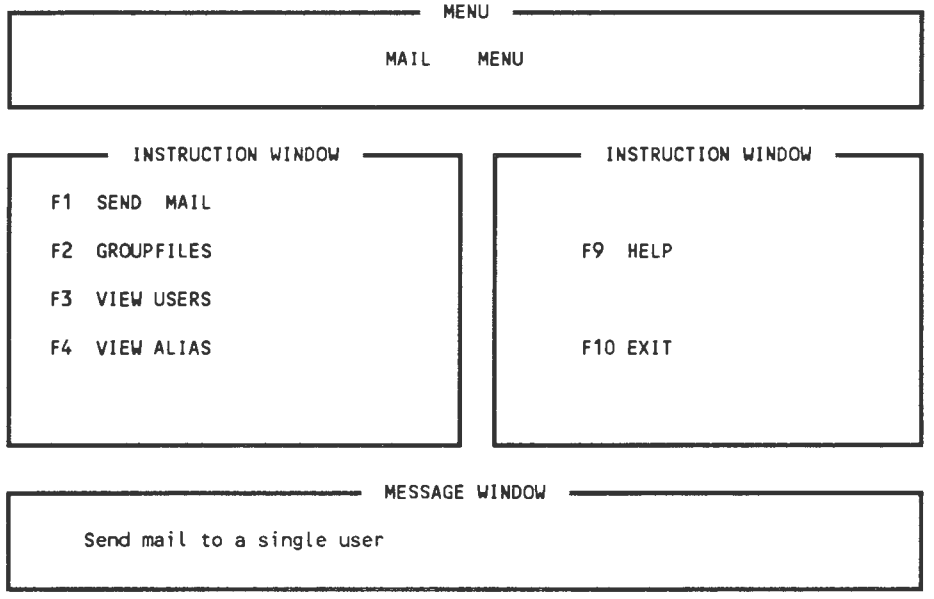


Figure 4.6: COSNET Send Mail Menu

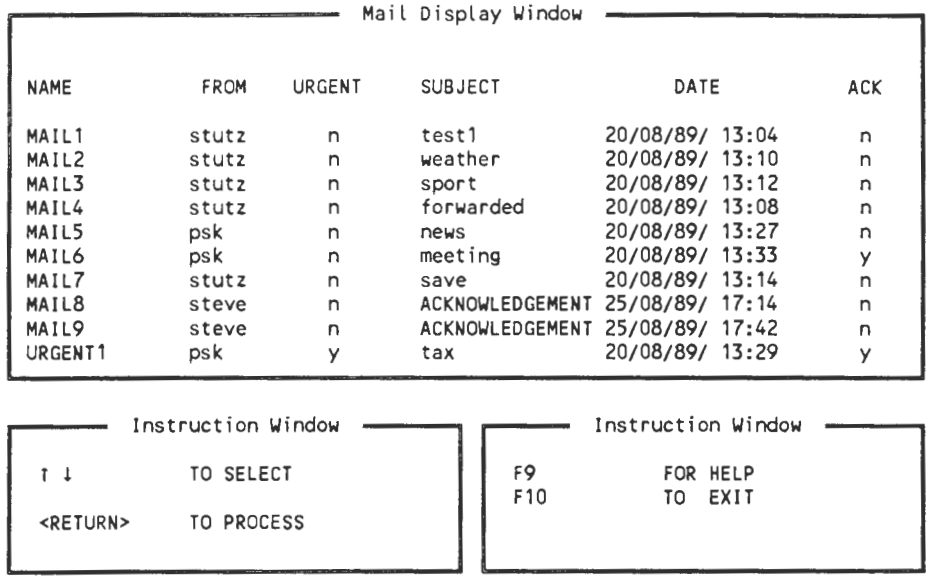


Figure 4.7: COSNET Mail Items Display

COSNET also lets the user inspect legal user and alias names from within the *create mail* option.

4.4.3 Inspect Mail

COSNET scans the user's mail directory to determine if any mail exists. The mailfile is then copied to the workstation where the mail items are read into a dynamic data structure and eventually displayed in the format shown in Figure 4.7.

A mail item is selected by positioning the cursor on the mail display line required, followed by pressing **ENTER**. The mail item is then copied from the data structure record to a window buffer and displayed on the screen as shown in Figure 4.8. Once the user has completed reading the mail item and has quit the file display screen, he may *save, delete, forward*, or *print* the message.

Two types of save options exist:

- The user can append the message to his mailbox, where previously saved mail items are kept. This is the default option.
- The mail item can be saved in the user's filing cabinet just like a document.

COSNET deletes the mail item from the mailing list in both cases.

The *forward* option allows the mail item to be forwarded either to a single user or to users specified by a distribution list. This option is similar to the *send mail* option, except that now an incoming item forms the contents of the message.

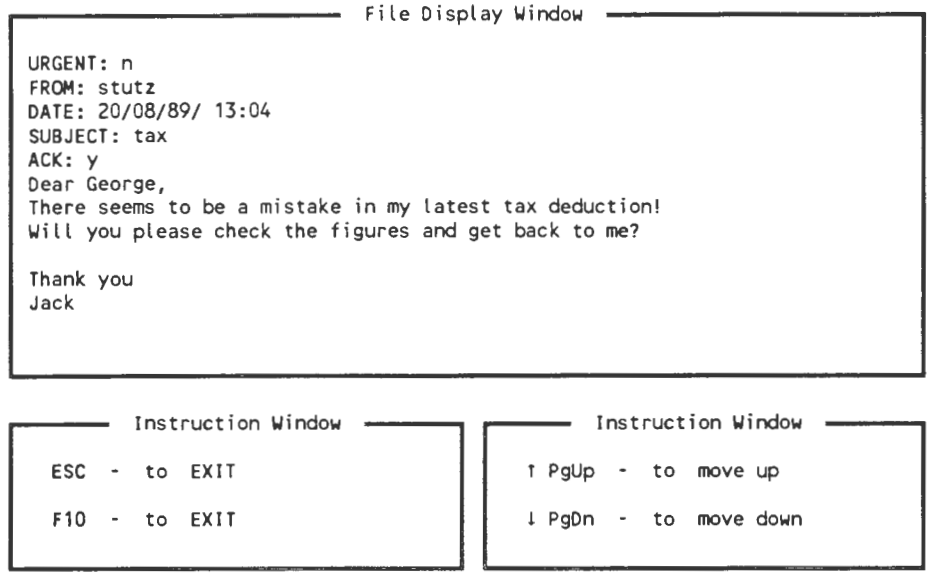


Figure 4.8: COSNET Message Display

4.4.4 Alias

COSNET provides an aliasing facility that permits the definition of one alias per login name. This alias can be used anytime in place of the actual user name.

A user may create a nickname for any legal COSNET username. Often, system names consists of surnames or unique identification numbers, that are be tedious to remember and to type. For example, a login name for the President of the company could consists of his surname and initials, e.g. ***worthingtongim*** for G.I.M. Worthington. This name is not easy to type and will often be misspelled. By creating an alias for ***worthingtongim***, for example ***president***, such problems can be avoided.

The COSNET aliasing functions are:

- Inspect alias definitions
- Create alias
- Delete alias
- Change alias

The alias definitions are stored in an aliasfile which is retrieved from the server when the user logs into COSNET.

4.4.5 Inspect Mail-Log and Inspect Mailbox

The *Mail-Log* is used by COSNET to store a copy of all outgoing messages. This facility provides an easy reference for the sender, since he can inspect the log file

and review all the mail attributes, such as when the mail was sent, who it was sent to, and what the message was.

The mailbox, as mentioned earlier, is used to store a set of saved messages. The user may later view this collection of mail items using this option and browse through them.

Both these viewing options use the message display screen, seen in Figure 4.8, to show the contents of the files.

4.5 ELECTRONIC FILING

COSNET electronic filing facility allows the user to *store, retrieve, update* and *distribute* documents. Documents can be edited by a word processor or an ordinary editor, which must be resident at the workstation. COSNET, just like ALL-In-1, allows the user to choose an editor or word processor, for document updates and creation. In addition to All-In-1's document retrieval by name and keyword, as discussed on Page 60 in Chapter 3, COSNET can retrieve documents by creation date and by author, but can not retrieve a document by its contents, as for example OBE.

4.5.1 Functionality

Each COSNET user has a private "cabinet" which contains all of his documents. This document filing structure was chosen for COSNET since it can be conveniently modeled by the hierarchical UNIX file system structure. A diagram of a user's cabinet hierarchical structure is given in Figure 4.9.

Users may create any number of folders within their cabinets, and each folder in turn may contain any number of folders. COSNET maintains a set of *cabinet information files* for each user. These information files describe the cabinet structure, and have detailed information for each document stored. These files are used to determine file access permissions and are used for retrieval of documents.

The information files are retrieved from the server whenever the user wishes to access the filing cabinet. COSNET then transforms all the data to a dynamic data structure that holds all the cabinet information. The electronic filing routines then can traverse this data structure and retrieve the relevant information.

Whenever a document is retrieved by a user having write permission to that document, a lock is created on the folder containing the document. This prevents any other user from manipulating the folder's contents while the information file is updated. The workstation creates a lock entry in the document's information file and the document is retrieved. Once the document is safely copied to the workstation, the document information file is returned to the server. Now the folder lock can be removed, since the information file describing the retrieved document contains a flag specifying that the document is in use.

The document is not deleted from the folder after it is retrieved. Any other user may copy this document to their workstation, although it might not be the latest

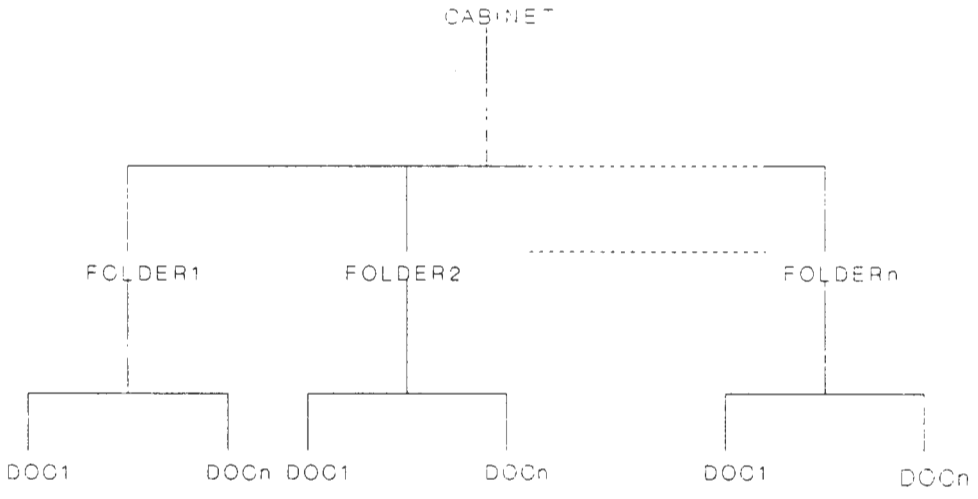


Figure 4.9: Hierarchical COSNET Cabinet Structure

version, but may not restore it in that particular folder.

Once the user has completed updating the document, events similar to the document retrieval procedure occur in the following order:

- A *folder lock* is created and the folder information files are retrieved, since someone else could have changed some other document.
- The *document lock* is cleared from the information file and the information file is returned to the server.
- The document is transferred back to the user's cabinet on the server.
- Finally, the *folder lock* is removed from the cabinet.

This rather elaborate procedure is necessary to ensure that information consistency is maintained. To speed up the document storage and retrieval, one could simply lock the folder from which the document was retrieved, and only remove the lock once the document is returned to the cabinet. This method would effectively restrict access to all the documents in the folder while the document is being updated. It is far more elegant to restrict access to the folder only briefly, and to allow the other documents in the folder to be accessible most of the time.

4.5.2 Retrieve

The retrieval of COSNET documents can be achieved in the following ways:

- by name

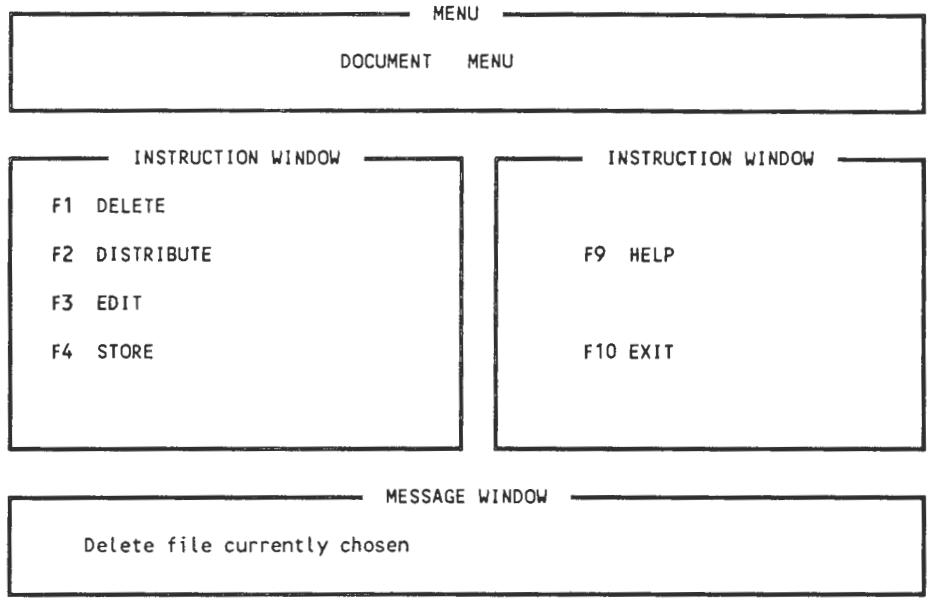


Figure 4.10: COSNET Document Manipulation Menu

- by keyword
- by date

COSNET displays the existing folders when the retrieval by *name* is selected, and prompts the user for a folder name. The entered name is verified and the folder contents, i.e. the documents, are displayed. The user may then select any document displayed in this list.

The user has to enter the retrieval parameters for retrieval by *keyword, author* and *date*. COSNET scans the user's entire cabinet structure and displays the contents of each document that matches the retrieval conditions. The user can retrieve the document displayed or continue the search for the next document satisfying the conditions.

Once a document is retrieved from the server, it can be *edited, deleted, distributed* or *stored* in another folder. The menu presenting these facilities is shown in Figure 4.10. COSNET automatically returns the document to its original folder when leaving this menu, unless the document was deleted.

The *distribute* facility uses COSNET's mail system and sends the document in the same manner in which a message would be sent.

4.5.3 File

A newly created document is initially stored in the working directory on the workstation. This document can later be filed in a filing cabinet on the server. The user has the option to store the document either in his *own* or in the *public* filing cabinet.

4.5.4 Folders

The *folders* option in the electronic filing menu provides the user with a set of folder operations. New folders can be *created* and existing folders *deleted*, provided they are empty. The current version of COSNET allows a user only to create and delete folders in his *own* or in the *public* cabinet.

4.5.5 Restore

Every time a document is deleted from the electronic filing cabinet, a copy of it is created in the user's *wastebasket*. The wastebasket keeps a deleted document for a period of two weeks, after which the document is removed from the system.

Often, documents are deleted by mistake and can not be retrieved, unless a backup version of the document exists. Many systems do backups of the file system and the documents can be restored by the computer operator. COSNET's wastebasket facility allows the user to restore a document deleted accidentally. This facility is not provided by the office automation systems described in Chapter 3, but can be found in systems such as CEO.

A user can restore any COSNET document by selecting the *restore* option: COSNET then displays the contents of the wastebasket, and the user can select a document to restore. The document is then automatically restored to its original folder.

4.6 CALENDAR

COSNET's *administrative support* facility only provides the user with an electronic diary, and, like UNIPLEX, does not have the resource and meeting scheduling facilities of All-In-1 and PROFS.

The COSNET electronic diary allows a user to inspect schedules, enter new appointments and set access permissions to the calendar.

4.6.1 Functionality

All calendar entries are stored in the user's calendar directories on the server. Two files are used by COSNET:

- a file containing the user defined access permissions to the calendar
- the calendar file itself

These files are retrieved from the server to the workstation where the contents are read into a dynamic data structure. Each calendar entry is represented by a record and the entire diary is represented as a linked list of the records. COSNET's calendar functions traverse this list and extract the relevant schedules.

A date is represented by an integer value which is stored as part of the schedule entry. An algorithm transforms this value into the correct day of the week.

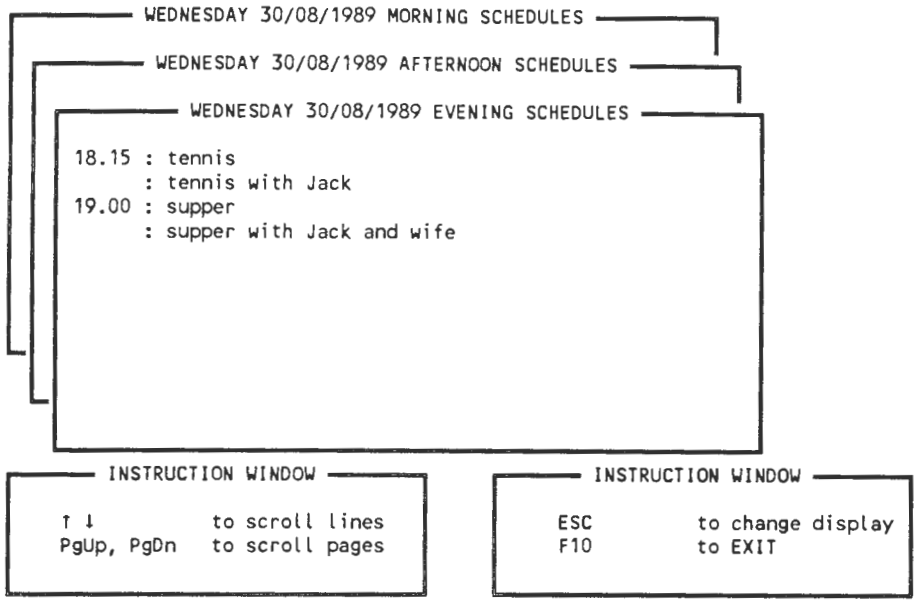


Figure 4.11: COSNET Daily Calendar Display

Access to the calendar is verified before any schedules are displayed. The owner of a calendar has full access to the schedules, whereas users with read or write access to a calendar cannot inspect schedules marked as ***private***.

Only one user may access a calendar at a time and the system displays a warning should anyone try to access a calendar already in use.

4.6.2 Inspect Calendar

COSNET calendar schedules may be viewed in three different formats:

- Daily
- Weekly
- Monthly

The ***daily*** display partitions the schedules into morning, afternoon, and evening sections. Each partition has its own window which contains the schedules, and the user can switch between the windows by using the **ESC** key. Figure 4.11 shows a daily display screen.

A daily schedule consists of the time of the schedule, a keyword to identify the nature of the schedule, and a description giving detailed information pertaining to that schedule.

The ***weekly*** display has one window for each weekday, and only the time and the keyword of a schedule are displayed. The user changes the active display window by using the **ESC** key as in the daily display mode. A sample weekly display screen is shown in Figure 4.12.

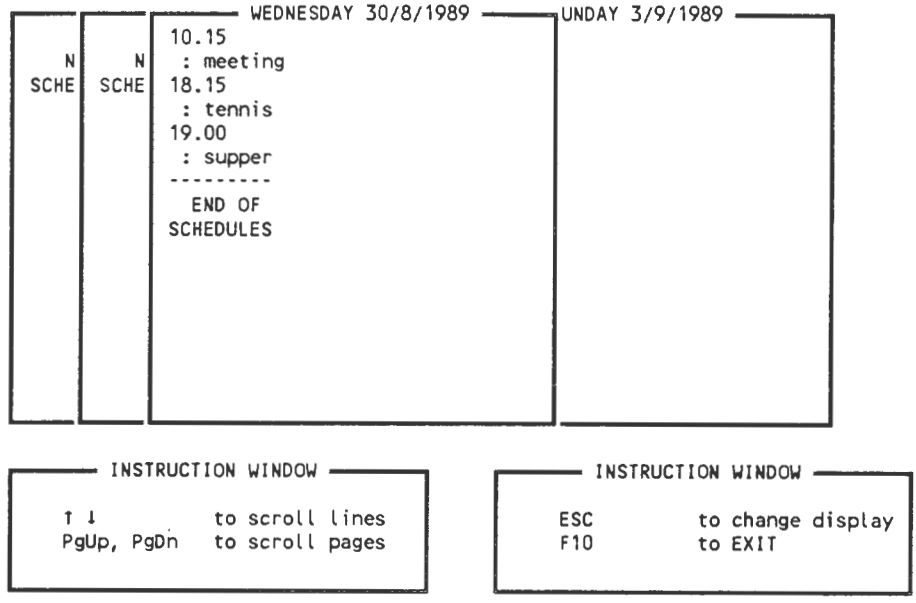


Figure 4.12: COSNET Weekly Display Mode

COSNET also supports a monthly display of calendar schedules. The display starts with the first entry for a particular month, and lets the user *scroll* through all the schedules for the month, instead of trying to fit all the schedules onto one screen, as for example, PROFS.

4.6.3 Modify

The existing schedules for a particular day are displayed before a new entry to the calendar is created. The user may then enter a new schedule which is inserted into the calendar data structure immediately after verification. COSNET always checks the data fields entered by the user, and demands that the data entry be repeated, if an incorrect entry is submitted.

This ensures that only valid data items are passed to the calendar routines. Should a particular calendar slot already be used, then the user may either re-enter the schedule at a different time, or he may change the details of the existing schedule.

4.6.4 Access Modes

Initially, no one has access to a calendar, except its owner. *Read* and *write* permissions can be granted to other users by the owner of the diary. Only valid usernames are accepted by COSNET and new names are added to the access file only after validation.

A user with *read* access may not update an electronic diary, but may inspect the schedules contained within. A user with *write* access may update and inspect the

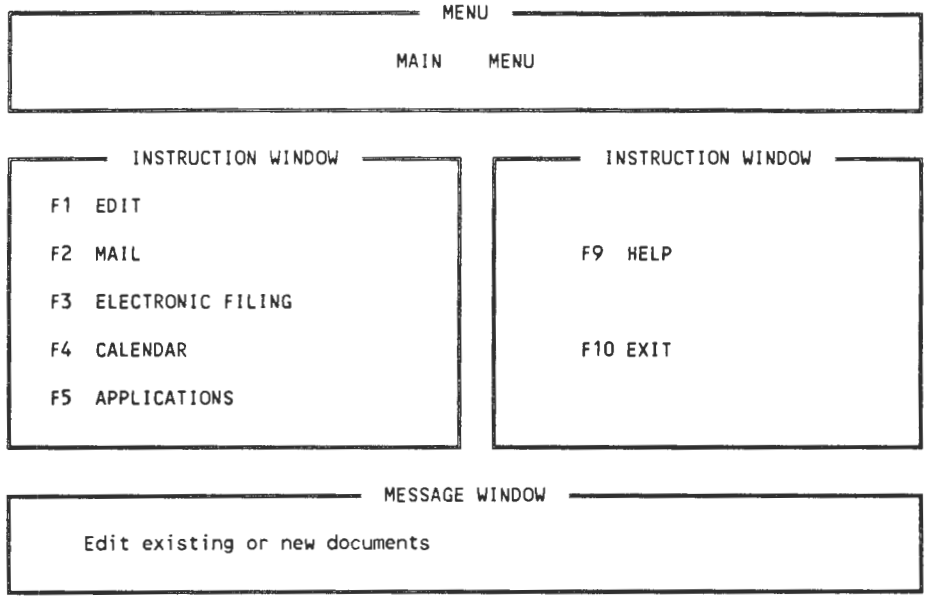


Figure 4.13: COSNET Main Menu

calendar contents. Entries in a calendar marked as *private* can only be inspected by the owner of the diary.

COSNET inspects the read and write permissions whenever a user requires access to a calendar other than his own. Should that username not be found in either of the lists, then access to the calendar is denied to that user.

4.7 APPLICATIONS

COSNET's aim is to provide an office automation system that utilizes Personal Computer software packages. The need to include such applications software originates from the fact that most administrative documents are created by PC - based word processors or spreadsheets.

COSNET permits the definition of ten different applications. These applications are executed by COSNET and the resulting documents can be stored on the server by the electronic filing system. Following the trend of All-In-1, COSNET thus provides means to facilitate the integration of new applications, and in doing so produces a highly flexible office automation system.

The current version of COSNET uses 230Kb of memory on the workstation, so the application has the rest of the workstation's memory available. The pathname of the application must be specified in full in the *Command-Line* field, unless the pathname is defined in the PATH variable in AUTOEXEC.BAT on the workstation.

The application facility can also be used to define the default editor or word processor used by the *edit* option in the main menu (see Figure 4.13) and the

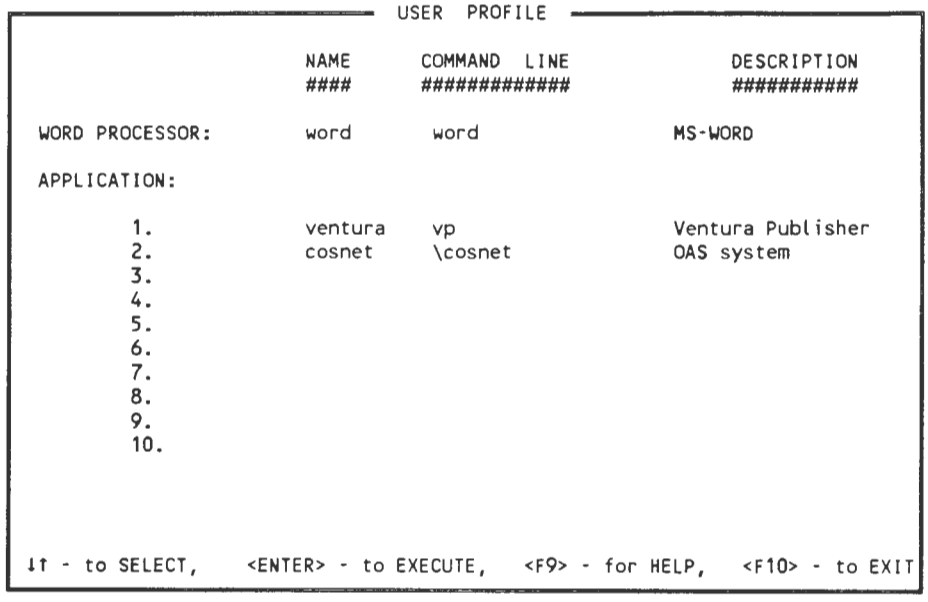


Figure 4.14: COSNET Application Screen

document applications menu in Figure 4.10.

Applications can be *added, deleted, updated*, and *executed.* A sample application screen is shown in Figure 4.14. An application is selected by positioning the cursor on the application folowed by pressing **ENTER**.

4.8 STANDARDS

The current version of COSNET does not provide any means for document exhange other than the transfer of documents between the workstation and the server. COSNET was designed with the aim of providing the Personal Computer user with facilities to share documents and to communicate with other PC users.

All communication is handled by KERMIT [KERM88], [KERM87], which is set to server mode on the office automation system server. KERMIT generally performs two functions, terminal emulation and file transfer. COSNET uses KERMIT for the transfer of files, which is achieved by using the KERMIT file transfer protocol.

All requests for communication originate from the workstation. Although the office automation system server can connect to other remote machines, using XNS, there is no interface that supplies a link to COSNET.

The COSNET document facility relies on the set of information files that describe the cabinet structure. In order to be able to transfer documents external to COSNET, an interface program would have to be written to allow for inter UNIX document transfer.

The mail facility is currently also only restricted to the COSNET office automation system. The COSNET mail does not utilize the UNIX mail system, since many

of COSNET's mail facilities, such as acknowledgement and classification, would not have been possible to implement. An interface to the UNIX mail system would not be too difficult and could be implemented relatively easily.

Once COSNET can utilize the UNIX mail system and can "recognize" external documents, it will be able to utilize UNIX-to-UNIX Copy, UUCP [UUCP86], which is the largest and oldest of the UNIX networks.

UUCP is a group of programs that performs file transfers and command execution between systems. It is the most widely used networking facility for UNIX systems, for the following reasons:

- UUCP is the only standard networking system available for every release of UNIX.
- UUCP is the cheapest network one can have; all that is needed is an asynchronous connection between two UNIX systems.
- For data transfer between remote UNIX systems all that is needed is a modem with dial-out capacity.

After a few minor changes, COSNET would thus be able to utilize these powerful UNIX networking facilities. Three other UNIX networks deserve to be mentioned briefly:

- *RJE* - The RJE (Remote Job Entry) system refers to a collection of programs and associated hardware that allows a UNIX system to communicate with the Job Entry subsystem (JES) on IBM mainframes.
- *NSC* - The Network Systems Corporation Hyperchannel network is a high speed local area network (LAN). It can link several thousand systems over a maximum distance of 5,000 feet and can transmit data at a rate up to 50 million bits/second. It can also interconnect distant systems over leased communications facilities, such as microwave or satellite links.
- *3B Net* - 3B Net is an Ethernet-compatible network that interconnects AT&T 3B computers. It can be used at distances of up to 2.8 kilometers and can connect up to 1,024 systems. It transmits data at up to 10 million bits/second.

These are just a few of the many UNIX networking facilities available. Other networking systems, such as TCP/IP, provide communication facilities between various operating systems.

Thus, COSNET has the potential to utilize one or more of these communication networks for document and message interchange. No UUCP connection exists between the server and other departmental UNIX machines, so no communication facilities have been provided for COSNET.

4.9 SUMMARY

The COSNET prototype office automation system thus supports many of the features that are supported by large office automation systems, such as PROFS or All-In-1 discussed in Section 3.5.

COSNET allows the user to define any MS-DOS based editor or word processor, and uses a *simple editor* for the creation of mail.

The *electronic filing* facility allows documents to be *created, filed, retrieved* and *deleted*, and thus provides the users with the necessary features for document exchange. A user may set access permissions to each of his documents and may grant other users either *read* or *write* access to a specific document.

The COSNET *mail* facility lets the user *read, file, forward, delete* and *print* a message, and supports classification of mail.

COSNET does not provide any resource and meeting scheduling and has no telephone directory. The *calendar* facility is used as an electronic diary and stores all the user's schedules. These schedules may be viewed in either *daily, weekly* and *monthly* display modes. *Read* and *write* access to the calendar can be set by the user, in order to allow other users to manipulate his schedules.

Any MS-DOS based application software can be added to COSNET. This facility allows the COSNET user to configure the office automation system to simulate the office environment.

COSNET thus supports most of the necessary features required by an office automation system, as discussed in Section 3.5. The only facility that the current prototype does not support, is the ability to exchange documents between other office automation systems and COSNET.

Chapter 5

TESTING COSNET

The COSNET facilities were all tested during the various phases of implementation. The results were favourable, and the facilities performed their respective tasks correctly. Unfortunately, a designer tests a system according to how a system was implemented, and what the input requirements are. It is impossible for the designer and implementor to predict the numerous invalid operations that can be applied to the system by users that are unfamiliar with the system. The designer must safeguard the system from illegal input and must provide facilities to recover from error conditions.

The COSNET prototype office automation system was tested by a series of two test. Each test was performed over a day by two users STEVE and HIL, who were paid for this purpose.

A full office environment was created and ten users were added to the system. Each user environment contained a number of documents, mail items and schedules.

The students were given login accounts on COSNET and an agenda of tasks to be performed. None of the students had any previous contact with COSNET and no User Manual was given to them. The only aid provided to them to perform their tasks was the COSNET HELP facility.

The agenda for each user was carefully designed to test the various COSNET facilities and features, and a full listing is given in Appendix A.1. *Bug Report* sheets were given to the users to describe any system faults in detail. These reports were used to correct any COSNET bugs; a sample bug report sheet is illustrated in Figure 5.1.

The two tests were designed with the following goals in mind:

- test all the COSNET facilities
- test the integrity and security of COSNET
- test the user interface
- determine the *user friendliness* of COSNET

The following sections discuss the tests themselves, as well as the results attained.

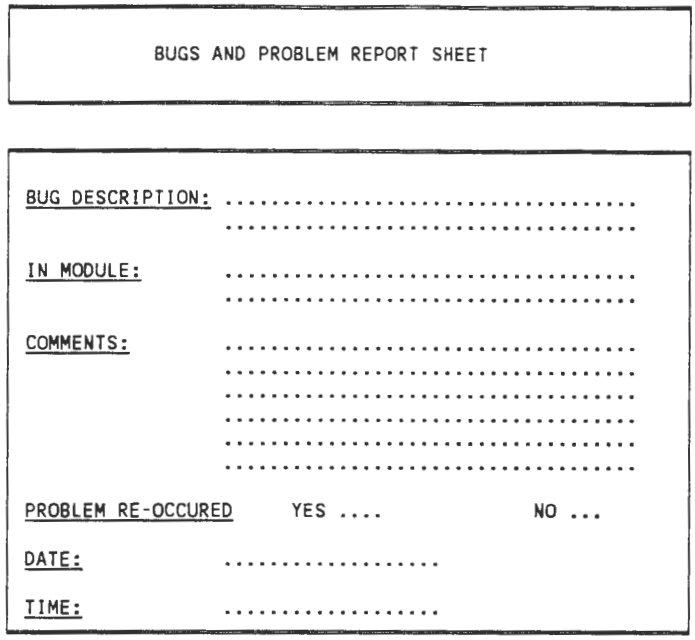


Figure 5.1: A Sample Bug Sheet

5.1 FIRST TEST

A working environment was created for the users, and consisted of a number of mail items, documents, and calendar schedules. The set of instructions supplied on the agenda was designed in such a way, that most of the COSNET functions were executed within the environment.

5.1.1 Applications

No MS-DOS applications were included in the initial environment for the users. Each user was instructed in the agenda to create an application environment by utilizing all the functions supplied by COSNET. The following tasks attempted to test the application facilities:

- *define* a default word processor/editor
- *add* a new application
- *change* the definition of an existing application
- *delete* an application
- *execute* each application

This comprises of all the functions available by the *applications* menu and provides a comprehensive test of the facility. A detailed account of this test is supplied in Appendix A.1.1.1 and Appendix A.1.2.3.

All the tasks were completed without problems, and the expected results were attained, with the exception of one application that could not execute due to insufficient memory on the workstation.

5.1.2 Calendar

Both users had schedules set up in their calendars as part of the environment, at the start of the tests. No access permissions to HIL's calendar were initially set, but HIL had *read* access to STEVE's calendar. The following functions of the calendar facility were submitted to the test:

- set *read* and *write* permissions for the calendar
- *addition* of a *private* schedule
- addition of a schedule using the *system default* time and date values
- *viewing* the calendar in *daily* display mode
- *viewing* the calendar in *weekly* display mode
- *viewing* the calendar in *monthly* display mode
- *viewing* another user's calendar
- *updating* another user's calendar

A problem was encountered, when trying to add an existing user to the calendar access lists. Instead of displaying an error message, garbaged letters appeared on the screen. This problem was corrected before the start of the second test. Otherwise, no problems were encountered.

Private schedules were correctly displayed to the user, and were hidden to anyone else inspecting the calendar. All three display modes displayed the schedules correctly, and access permissions to calendars correctly sustained. A detailed summary of the first calendar test routines is discussed in Appendix A.1.1.2 and Appendix A.1.2.4.

5.1.3 Mail

A number of mail messages had been sent to HIL and STEVE to set up their mail environment. Figure 5.2 illustrates the mail display screen containing these messages. Each user was asked to perform a number of tasks, using these mail items, to test the *mail display* facility.

To test the *distribution* of mail items, each user had to create a short message and send it using various distribution methods. The following list shows the mail attributes subjected to the first test:

- *sending* a mail item to a single user

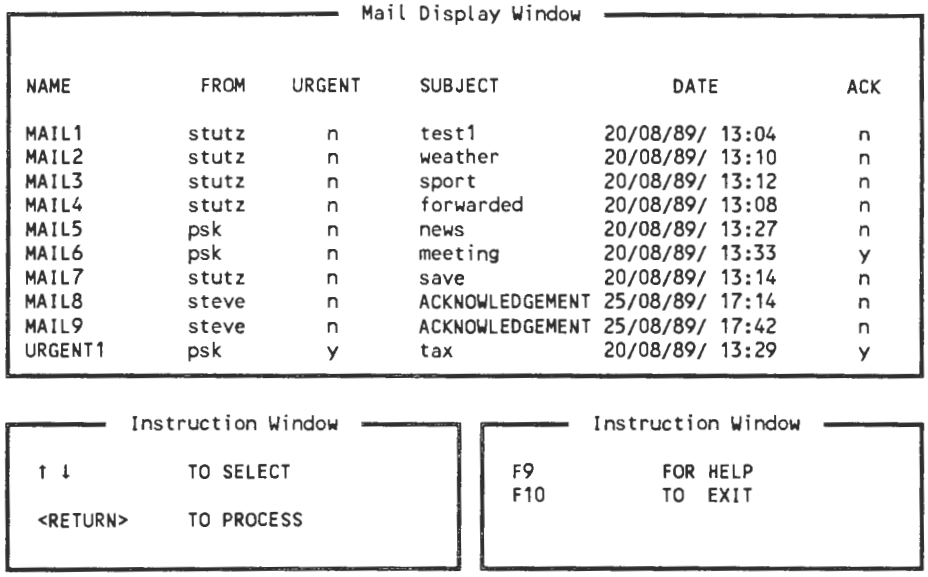


Figure 5.2: Test Mail Environment

- *classification* of mail items
- the use of *distribution* files
- *creation* of *aliases*
- *update* of aliases
- *viewing* of mail items
- *saving* of a mail item in the *mailbox*
- *saving* of a mail item in the *electronic filing cabinet*
- *forwarding* a mail item
- *acknowledgement* of mail inspection

Two system errors were detected by the users:

1. The alias facility did not check the for the legal length of a new alias. An alias containing a space, e.g. "hel lo", was considered as correct by COSNET. This caused various problems, since all the routines using the alias file assume that only valid alias names exist.

2. Whenever a COSNET mail item is saved, the system proceeds to delete the mail item from the mail list. After completion the save option, COSNET should automatically return the user to the mail display menu shown in Figure 5.2. During the test however, this did not occur and the could select the

save option again. This led to a series of errors, since that particular mail item did not exist anymore.

Apart from these two errors which were corrected before the start of the second test, all functions behaved as expected; the detailed description of the COSNET mail tests can be found in Appendix A.1.1.3 and Appendix A.1.2.1.

5.1.4 Electronic Filing

To test the *electronic filing* facilities of COSNET, a number of folders and documents were created for each user. The folders JUNK, MSWORD and WORK, were created in each of the user's filing cabinet. Each folder was set up to contain a number of documents.

The following list summarizes the electronic filing functions included in the first test:

- *creation* of new folders
- *deletion* of a folder
- *retrieval* of a document by *name*
- *update* of a document
- *automatic storing* of a retrieved document
- *deletion* of a document
- *retrieval* of documents from other user's cabinets
- *access permissions* to documents
- *restoring* of a deleted document

This set of functions covers just about the full set of document handling facilities offered by COSNET, and a detailed summary of the tests performed on the document facilities is given in Appendix A.1.1.4 and Appendix A.1.2.2.

Folders were successfully added and deleted, and COSNET correctly displayed the updated cabinet structure. Retrieval, update and deletion of documents within the cabinet structure were correctly reflected by COSNET.

Access permissions to *read* and *write* documents belonging to other users correctly allowed the manipulation of these documents, and no errors were reported.

The only error that was detected was in the "*restore*" procedure and was corrected before the start of the second test. COSNET did not correctly restore a deleted document to its original folder. On closer inspection it was found that COSNET did actually physically restore the documents to the correct folder, but did not update the information files (Section 4.5.1) correctly.

5.1.5 Summary

Both users reacted to the user interface very favourably and commented on the easy manipulation of the menus. They found no problem in completing their tasks, especially due to the usefull online help.

The COSNET features executed predictably, with a few exceptions that are noted below:

- Error condition when an existing *read* or *write* access permission was added to the calendar
- Problems occurred when invalid alias names were added
- COSNET allowed a user to save an already deleted mail item
- COSNET did not restore a deleted document correctly
- COSNET menu windows behaved erratically on two occasions

These COSNET errors were corrected and a second test arranged, to ensure that these problems had in fact been solved, and to test other features of COSNET that were not part of test one.

5.2 SECOND TEST

A new test agenda was drawn up for the second test, which contained tasks to test the errors discovered during TEST1. The test also served the purpose to ensure that the mail *interrupt* facility and the retrieval by *date, author* and *keyword* work correctly. The following lists states the objectives of TEST2:

- Test the *aliasing* facility
- Test the *saving* of mail items
- Test the *restore* option of electronic filing
- Test the display of menu windows
- Test the *interrupt* facility
- Test the retrieval by *name*
- Test the retrieval by *date*
- Test the retrieval by *author*
- Analyze critical comments of the users

The detailed list of the test instructions can be found in Appendix A.2.2 and Appendix A.2.1.

The second test proved to be very successful. All the objectives set for the testing of the document facilities were achieved and the only error reported was screen related, as discussed on page 120. The documents were correctly retrieved, deleted and restored in the electronic filing cabinet.

The alias facility worked perfectly for alias names of up to eight characters long. Alias names longer than this limit, which is the maximum length of a COSNET user name, were correctly created, but mail could not be sent using this alias. This was due to the COSNET's automatic field checking, that only accepts user names with the length of eight or less characters.

The mail *interrupt* facility let the users create a mail item, exit mail to use another COSNET facility, and return to complete the mail item, correctly.

The users were also asked to supply objective comments about the facilities of COSNET and this will be discussed in detail in Section 5.3.

5.3 SUMMARY

One must bear in mind, that the two tests were designed not only to test the correctness of the COSNET facilities, but also to exploit any COSNET weaknesses. The users were instructed to "abuse" the system by entering incorrect data, performing unpredictable tasks, and attempting to crash the system.

A number of bugs were found with the testing procedures and duly corrected. All the facilities of the current COSNET prototype, as described in Chapter 4, work correctly.

5.3.1 User Interface

Both users found COSNET's user interface attractive, and easy to use. As mentioned before, none of the users had any previous contact with COSNET, and no user manual was given to the users.

In spite of this, both users had no difficulty in performing the tasks described in the tests and found COSNET's *help* facility very useful.

One criticism concerning the design of the mail user interface was raised. The user was initially confused by the difference between the "F2 INSPECT MAIL", "F4 INSPECT MAIL-LOG" and "F5 INSPECT MAILFILE" options. This confusion would have been avoided, if the User Manual had been made available.

The COSNET user interface thus provides an effective mechanism to manipulate the facilities by providing a simple, attractive and comprehensive interface to the office automation system.

5.3.2 COSNET Facilities

The users found the navigation through the different facilities simple and quick and found that the global structure of the system was comprehensive in its support of

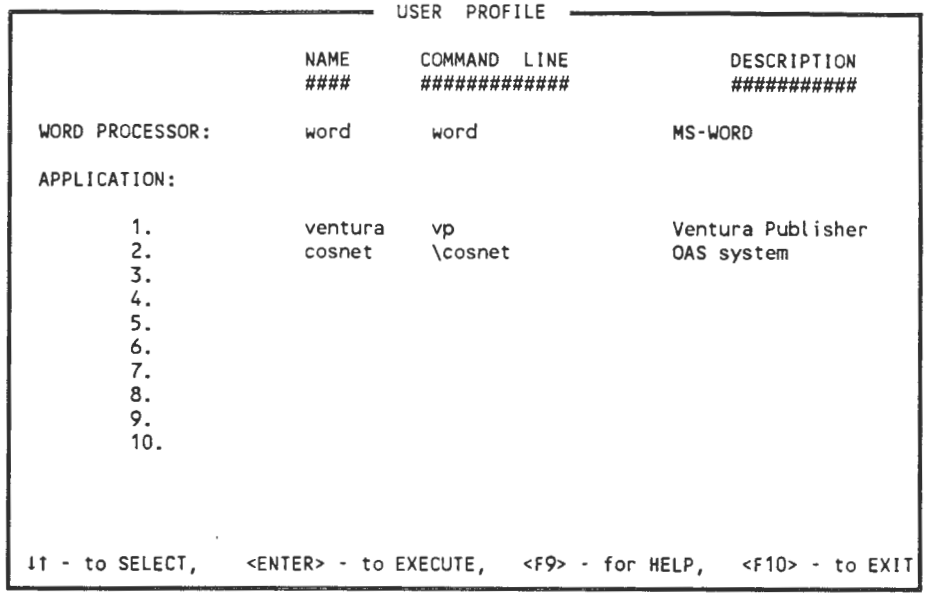


Figure 5.3: Application Screen

an office environment.

One user mentioned that he would prefer to select and execute a menu option by simply pressing the required function keys. A point was raised that some commands do execute immediately after the respective function key is pressed, causing an inconsistency in the menus. The *application* screen illustrates such an example in Figure 5.3. It is true that the command is executed by simply depressing the function key, without the need of the RETURN key. But this action is always preceded by the selection of an item, such as positioning the cursor over the required application. Thus, effectively to keyboard actions generally take place, as in the selection of a menu option.

Another criticism was that the length of the field in the system pop-up prompts did not reflect the maximum length of the input item. This is correct; a date that consists of two digit input, is entered via a prompt window having a size of ten characters, as shown in Figure 5.4. Although each prompt window is called with an argument to determine its size, it would not be possible to display a heading, such as "ENTER DATE" in Figure 5.4, for a prompt window having the size of two characters.

It was also suggested that the system should ask for confirmation whenever a document or application is to be deleted. This seems unnecessary, since a deleted document can be restored by the *wastebasket* facility, and a deleted application can be added back without difficulty. A summary of critical comments is Appendix A.2.2.6 and Appendix A.2.1.6.

A feature particularly liked by the user is the ability to define a personalized default editor. Since a wide variety of word processors are available for the Personal

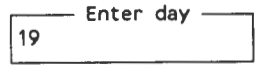


Figure 5.4: Date Prompt Example

Computer, this facility becomes particularly valuable.

The users thought that COSNET's error messages and system prompts are concise and intelligent, and convey the message to the user effectively.

A summary of COSNET features enjoyed by the test users is given in Appendix A.2.2.4 and Appendix A.2.1.5.

Generally, it was concluded that COSNET supports the facilities and features of an effective and comprehensive office automation system.

Bibliography

[AHLS83] Ahlse'n, M., Bjonerstedt, A., Britts, S., Hulten, C., and Soderlund, L., **A Survey of Office Information Systems**, SYSLAB Working Paper No. 44, (Stockholm, January 1983).

[AIEL84] Aiello, L., Nordi, D. and Ponti, M., **Structural Office Modelling**, Proceedings 2nd ACM Conference on Office Information Systems, (Toronto, June 1984).

[BAIL81] Bailey, and Gerlach, **Internal Accounting Controls in the Office of the future**, IEEE Computer, Vol. 14, 5 (May 1981), pp.59 - 70.

[BLAC86] Black, W.J., **Intelligent Knowledge Based Systems**, Van Nostrand Reinhold (UK), 1986.

[BOGG80] Boggs,D.R., Schich, J.F., **PUP: An Internetwork Architecture**, IEEE Trans. Commun. COM-28, Vol. 4, (April 1980), pp. 612 - 634.

[BRAC83] Bracchi, G. and Perninci, B., **SOS: A Conceptual Model for Office Automation Systems**, Proceedings of ACM SIGMOD Database Week Conference, (San Jose, Calif. May 1983).

[BRAC84] Bracchi, G. and Perninci, B., **Design Requirements of Office Systems**, ACM Transactions on Office Information Systems, Vol. 2, No. 2, April 1984, pp. 151 - 170.

[BROT83] Brotz, D., **Message System Mores: Etiquette in Laurel**, ACM Trans. on Office Automation, Vol. 1, No. 2, April 1983, pp. 179 - 192.

[CONR81] Conrath, D.W, Higgins, C.A., Thackenhary C.S., and Wright, W.M., **The Electronic Office and Organizational Behaviour**, Computer Networks, Vol. 5, 1981, pp. 401 - 407.

[COOK80] Cook, C., **Streamlining Office Procedures**, AFIPS, (May 1980), pp. 555 - 565.

[CUNN83] Cunningham, I., **Message - Handling Systems and Protocols**, Proceedings of the IEEE, Vol. 71, No. 12, Dec 1983, pp. 1425 - 1450.

[DATA] Data General, **A CEO Preview**.

[DIGI] Digital., **ALL-IN-ONE: Office and Information Systems.**

[DODS83] Dodswell, J., **Office Automation**, J. Willey, 1983.

[ELLI80] Ellis, L. and Nutt, G., **Office Information Systems and Computer Science**, ACM Computing Survey, Vol. 12, 1, (March 1980).

[ENGE79] Engel, G. and Grappuso, J., **An Office Communications System**, IBM Systems Journal, Vol. 18, No. 3, 1979.

[FINN83] Finn, N.B., **Interfacing People with their Machines**, AFIPS, Vol. 52, 1983, pp. 353 - 359.

[GEHA82] Gehani, N.H., **The Potential of Forms in Office Automation**, IEEE Trans. on Commun. COM-30, Jan 82.

[GIBB82] Gibbs, S., **Office Information Models and the Representation of 'Office Objects'**, Proceedings ACM SIGOA Conference on Office System, (Philadelphia, June 1982).

[GRUH79] Gruhn, A.M. and A.C. Hohl., **A Research Perspective on Computer-Assisted Office Work**, IBM System Journal, Vol 18, No. 3, 1979.

[HAMM77] Hammer, M., Howe, W., Kruskal, V., and Wladlawsky, I. **A Very High Level Programming Language for Data Processing Applications** Commun. ACM 20, Nov. 1977.

[HAMM80] Hammer, M. and Kunin, J.S., **Design Principles of an Office Specification Language**, AFIPS, Vol. 49, 1980, pp. 541 - 547.

[HIRS85] Hirschheim, R. A., **Office Automation: Concepts, Technologies and Issues**, Addisson-Wessely 1985.

[HOLD80] Holden, J.B., **Experiences of an Electronic Mail Vendor**, AFIPS, Vol. 49, 1980, pp. 493 - 497.

[HORA85] Horak, W., **Office Document Architecture and Office Document Interchange Formats: Current Status of International Standardization**, IEEE Computer, Oct 1985, Vol. 18 (10), pp. 50 - 60.

[IBM82] **Office Information Architectures: Concepts**, IBM, GC23-0765-0.

[JANS83] Jansen, P., Svobodova, L. and Mahle, E., **Filing and Printing Services on Local Area Networks**, Commun. ACM, Vol. 26, No. 4, 1983, pp. 211 - 220.

[KAIV87] Kaivers, R., **Communication Analysis in the Company**, Computer Networks, Vol. 14, (1987), pp. 199 - 205.

[KERM87] da Cruz, F., and Fischer, H., **C-Kermit User Manual**, 1987.

[KERM88] da Cruz, F., and Gianone, G., **MS-DOS Kermit User Guide**, 1988.

[KONS82] Konsynski, B., Bracker, L. and Bracker, W., **A Model for Specification of Office Communications**, IEEE Trans. Commun., COM-30, (Jan. 1982), pp. 27 - 36.

[KURK83] Kurke, L. and Aldrich, H., **Mintzberg was Right!**, Management Science, Vol. 29, No. 8, Aug. 1983.

[LEDG80] Ledgard, H.F., Whiteside, J.A., and Singer, A., **An Experiment on Human Engineering of Interactive Software**, IEEE Trans. Software Eng., Vol. SE-6, Nov. 1980.

[LUM82] Lum, V., Choy, D. and Shun, N., **OPAS: An Office Procedure Automation System**, IBM System Journal, Vol. 21, 3, (1982), pp. 327 - 350.

[LIEB87] Lieberman, D. **Office Automation Strives to Juggle Competing Demands**, Computer Design, Vol. 26, No. 21, Nov. 1987.

[MALO83] Malone, T. **How Do People Organize Their Desks? Implications for the Design of Office Information Systems**, ACM Trans. Office Inf. Syst. 1, Jan. 1983.

[MAYE82] Mayer, N., **Office Automation: A Progress Report**, Office: Technology & People, Vol. 1, No. 1, March 1982.

[MINT73] Mintzberg, H., **The Nature of Managerial Work**, Herper & Row, 1973.

[NCR85] NCR, **VI User Manual**, 1985.

[NUTT81] Nutt, G.J. and Ricci, P.A., **Quinault: An Office Modelling System**, IEEE Computer, May 1981.

[O'KE80] O'Kelley, H.E., **Electronic Message System as a Function in the integrated Electronic Office**, AFIPS, Vol. 49, 1980, pp. 499 - 502.

[OLSO82] Olson, M.H. and Lucas, H.R. Jr., **The Impact of Office Automation on the Organization**, Commun. of the ACM, Vol. 25, No. 11, 1982.

[PANK81] Panko, R.R., **Facing Basic Issues in Office Automation**, Computer Networks, Vol 5, 1981, pp. 391 - 399.

[PANK82] Panko, R.R., and Sprague, R., **Towards a New Framework for Office Support**, Proceedings ACM SIGOA Conference on Office System, (Philadelphia, June 1982).

[POPP82] Poppel, H., **Who needs the Office of the Future?**, Harvard Business Review, Nov. 1982.

[RICH84] Rich, E., **Natural-Language Interfaces**, IEEE Computer, Sep. 1984, pp. 39 - 47.

[REIC87] Reichardt, O.A., **Optical Disk Technology: Moving Toward the Paperless Office**, Computer Design, Vol. 26, No. 21, Nov. 1987.

[SCHE88] Scheller, A., **Document Standards: Availability and Products**, Computer Networks, Vol. 16, (1988/1989).

[SCHI82] Schicker, P., **Naming and Addressing in a Computer Based Mail Environment**, IEEE Trans. Commun., Vol. COMM-30, No. 1, 1982.

[SCHO82] Schoch, J.F., Dalal, Y.K., Redell, D.D., and Crane, R.C., **Evolution of the ETHERNET Local Computer Network**, IEEE Computer Vol. 15, Aug. 1982.

[SHAR82] Sharma, D.K., and Gruchacz, A.M., **Display Text Editor TED**, IEEE Trans. Commun., Vol. COMM-30, No. 1, 1982.

[SLON81] Slonim, J., MacRae, L.J., Mennie, W.E., and Diamond, N., **NDX-100: An Electronic Filing Machine for the Office Of the Future**, IEEE Computer, May 1981, pp. 24 - 36.

[SOLO82] Solomon, M., Landweber, L.H. and Neuhengen, D. **The CSNET Name Server**, Computer Networks, Vol. 6, No. 3 (July 1982), pp. 161 - 172.

[STEW67] Stewart, R., **Managers and Their Jobs**, Macmillan, 1967.

[TAMI87] Tamine, J. **Successful Implementation of an Office System**, Computer Networks, 1987, 279 - 282.

[TSI182] Tsichritzis, D. **Form Management**, Commun. ACM, Vol. 25, No. 7, July 1982.

[TSI282] Tsichritzis, D., Rabbitti, F.A. and Hogg, J., **A System for Managing Structured Messages**, IEEE Trans. Commun., Vol. COM-30, (Jan. 82), pp. 66 - 73.

[TSIC83] Tsichritzis, D. and Christodoulakis, S., **Message Files**, ACM Trans. on Office Information Systems, Vol. 1, No. 1, Jan 1983, pp. 89 - 98.

[TSIC84] Tsichritzis, D., **Message Adressing Schemes**, ACM Trans. on Office Information Systems, Vol. 2, No. 1, Jan. 1984.

[ULR180] Ulrich, W.E., **Introduction to Electronic Mail**, AFIPS, Vol. 49, 1980, pp. 485 - 488.

[ULR280] Ulrich, W.E., **Implementation Considerations in Electronic Mail**, AFIPS, Vol. 49, 1980, pp. 489 - 497.

[ULRI83] Ulrich, W.E., **Current Issues on Electronic Mail - Heralding a New Era**, AFIPS, Vol. 52, 1983, pp. 361 - 365.

[UNIP88] UNIPLEX User Manual, 1988.

[UUCP86] Honeyman, P., Nowitz, D.A., and Redman, B.E., **Experimental Implementation of UUCP**, 1986 UNIFORM Proceedings.

[WHAN87] Whang, K.-Y., et al., **Office-by-Example: An Integrated Office System and Database Manager**, ACM Trans. on Office Information Systems, Vol. 5, No. 4, (Oct. 1987).

[WOOD70] Woods, W.A., **Transition Network Grammars for Natural Language Analysis**, Commun. ACM, Vol. 13, 1970.

[ZISM78] Zisman, M., **Use of Production Systems for Modeling asynchronous, concurrent processes**, Pattern Directed Inference Systems, Waterman and Hayes-Roth, Eds., Academic Press, New York, 1978.

[ZLOO77] Zloof, M. and De Jong, S., **The System for Business Automation (SBA): Programming Language**, Commun. ACM, Vol. 20, No.6, (June 1977).

[ZLOO81] Zloof, M., **QBE/OBE: A Language For Office and Bussiness Automation**, IEEE Computer, Vol. 14, No. 5 (May 1981).

[ZLOO82] Zloof, M. **Office-by-Example**, IBM System Journal, Vol. 21, No. 3, (1982).

Appendix A

TESTING

A.1 FIRST TEST

A.1.1 User name HIL

A.1.1.1 APPLICATIONS

At the start of the test session, no PC Applications or word processors are defined for the user:

1. Define the word processor "WORD" as the default word processor

   Purpose: Test the definition of a default word processor

   Result: OK.

2. Add the application "NU" (Norton Utilities)

   Purpose: Test the addition of a new application

   Result: OK.

3. Add the application "B" (Brief Editor)

   Purpose: Test the addition of a new application

   Result: OK.

4. Add the application "VI" (Another optional editor)

   Purpose: Test the addition of a new application

   Result: OK.

5. Change the application definition of "NU" to be "NI" (Norton Integrator)

   Purpose: To test the update of an existing application

   Result: OK.

6. Delete the application "VI"

   Purpose: To test the deletion of an existing application

   Result: OK. - Comment: "The system should possibly get confirmation of the user's intentions before deleting the application".

7. Execute each application

   Purpose: To test the execution of the defined application

   Result: OK.

A.1.1.2 CALENDAR

Although both user have schedules set up in their calendars at the start of the tests, no ACCESS specifications to their calendars exist. This implies that no one but the owner may inspect the calendar.

1. Set WRITE permissions for users FRED, STEVE, PSK, and PETE

   Purpose: Test the setting of WRITE permissions for users: - (Note that FRED is not a legal COSNET user)

   Result: Problem creating user FRED (not a legal owner); Wrong HELP file was displayed from this menu; the menu screens had a different format after completion of the task.

2. Set READ permissions for users STUTZ and JVD

   Purpose: Test the setting of READ permissions for users

   Result: OK.

3. Add a PRIVATE schedule using the default time and date

   Purpose: To test the addition of a private schedule using the date and time supplied by the system as default

   Result: OK.

4. Add a schedule using the default day, at 15:00.

   Purpose: To test the addition of a schedule

   Result: OK.

5. Add a PRIVATE schedule using the default day, at 16:00.

   Purpose: To test the addition of a PRIVATE schedule

   Result: OK.

6. View the calendar schedules using the WEEKLY display mode (current week)

   Purpose: To test the WEEKLY display of schedules

   Result: OK.

7. View the calendar schedules using the MONTHLY mode (current month)

   Purpose: To test the MONTHLY display of schedules

Result: OK.

8. Inspect STEVE's calendar schedules for the current day

   Purpose: To test test the inspection of another user's calendar

   Result: OK.

9. Add a schedule for STEVE at 15:30 today

   Purpose: Test the access to another user's calendar: HIL should not be allowed to enter schedules; only the READ permission is set for HIL.

   Result: OK.

10. Add a schedule for STEVE at 9:00 tomorrow

    Purpose: Test the access to another user's calendar: HIL should not be allowed to enter schedules; only the READ permission is set for HIL.

    Result: OK.

11. View STEVE's calendar using the WEEKLY display mode (current week)

    Purpose: To test the WEEKLY display of calendar

    Result: OK.

A.1.1.3 MAIL

To test the COSNET MAIL SYSTEM messages had been sent to the test users HIL and STEVE. Figure A.1 shows the mail display screen containing these messages. Each test user then performed a number of tasks to test the MAIL applications:

Send Mail

The following activities are designed to test the delivery of messages to users on COSNET:

1. Send a message to PSK

   Purpose: To test mail system

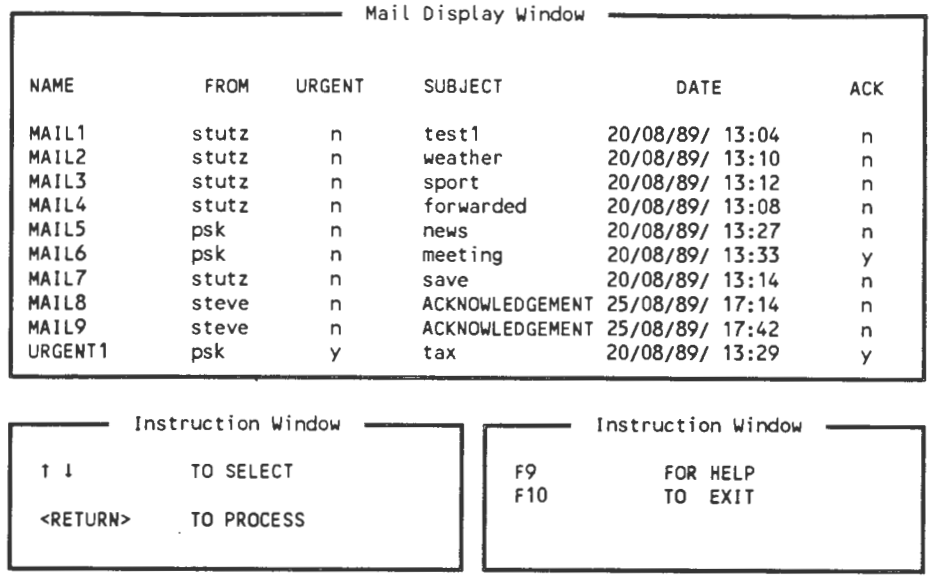


Figure A.1: Test MAIL Screen Display

Result: OK.

2. Create an urgent message (acknowledgement required) and then create a GROUPFILE containing users PSK, STUTZ, GEORGE and PETE

   Purpose: To test creation of groupfiles (GEORGE is not a user on COSNET)

   Result: OK.

3. Send this urgent message to the users specified by the groupfile.

   Purpose: To test the delivery of messages using distribution mailfiles

   Result: OK.

4. Create an alias for user PETE and send a message using this alias name

   Purpose: To test mail delivery using an alias instead of the user name

   Result: OK.

5. Create alias for users PSK, STUTZ, JACK, JVD; change PSK's alias to a new alias name

   Purpose: To test the COSNET aliasing system

   Result: System crash on creation of an alias.

Inspect Mail

The following tasks have been designed to test the manipulation of mail messages in the MAIL SYSTEM. Ten different mail items have been sent to user HIL:

1. View the first mail item, and leave it unchanged.

   Purpose: Test the display of messages and the "ignore" option of the mail system. This is the default if options such as SAVE, DELETE, FORWARD or PRINT are not chosen.

   Result: OK.

2. View the second item and save it to the default mailbox

   Purpose: Check the default save mail option. - This should append the current message to the user mailbox.

   Result: OK.

3. View the fifth mail item and save it in any file other than the default mailbox.

   Purpose: To test whether the system saves the message in a specified file

   Result: The system saved the message, HIL then tried to save the message again, causing the system to crash.

4. View the 7th mail item, forward it to user STEVE and leave the item unchanged.

   Purpose: To test the forward option of COSNET MAIL, and to ensure that the automatic acknowledgement of messages works (As seen in Figure A.1, mail item 7 requires an ACK to PSK).

   Result: OK.

5. Exit mail display and inspect mailbox

   Purpose: Ensure that COSNET did save the mail item in the mailbox correctly.

   Result: OK.

A.1.1.4 ELECTRONIC FILING

The following actions have been designed to test the document handling functions of COSNET:

1. Create any 2 new folders

   Purpose: Test the creation of new folders within the own cabinet

   Result: OK.

2. Attempt to delete folder JUNK

   Purpose: Ensure that non-empty folders do not get deleted

   Result: OK.

3. Delete one of the folders just created

   Purpose: Test the deletion of empty folders

   Result: OK.

4. Retrieve "ONE.DOC" from the "MSWORD" folder, edit it and return the document to the "MSWORD" folder.

   Purpose: Test the retrieval, updating and filing of documents

   Result: OK.

5. Retrieve "TWO.DOC" from the "MSWORD" folder and delete it

   Purpose: Ensure that documents can be deleted from the COSNET filing cabinet

   Result: OK.

6. Retrieve the saved mailfile from its folder and delete it

   Purpose: Check that the document was correctly saved and that it can be deleted.

   Result: OK.

7. Retrieve from PSK's cabinet folder "JUNK" the document "JUNK", edit it and return it to its origin.

   Purpose: Test the retrieval of documents from another user's filing cabinet. The system should prohibit user HIL to return the document to the cabinet, since he has no write permission.

Result: OK, not allowed to re-store.

8. Now store this document it own cabinet folder "JUNK"

   Purpose: HIL may restore the document anywhere in his own cabinet: so let's put it in "JUNK"

   Result: OK.

9. Restore the deleted mailfile document

   Purpose: Test the "WASTEBASKET" system. The mailfile should automatically be restored to folder "JUNK"

   Result: Problem with restoring the document in its original folder.

A.1.2 User name STEVE

A.1.2.1 MAIL

The following activities are designed to test the delivery of messages to users on COSNET:

1. Send a message to PSK

   Purpose: To test mail system

   Result: OK.

2. Create an urgent message, (with acknowledgement) and create a GROUPFILE containing users PSK, STUTZ, GEORGE and PETE

   Purpose: To test creation of groupfiles (GEORGE is not a user on COSNET)

   Result: OK.

3. Send the urgent message to the users specified by this groupfile.

   Purpose: To test the delivery of messages using distribution mailfiles

   Result: OK.

4. Create an alias for user PETE and send a message using this alias name

   Purpose: To test mail delivery using an alias instead of the user name

   Result: System error when using a blank username; OK the second time with correct name.

5. Create alias for users PSK, STUTZ, JACK, JVD; change PSK's alias to a new alias name

   Purpose: To test the COSNET aliasing system

   Result: OK for PSK, STUTZ and JACK (illegal user). STEVE then entered two words instead of one alias name, causing system to crash.

The following tasks have been designed to test the manipulation of mail messages in the MAIL SYSTEM. Ten different mail items have been sent to user STEVE:

1. View the first mail item, and leave it unchanged.

   Purpose: Test the display of messages and the "ignore" option of the mail system. This is the default if options such as SAVE, DELETE, FORWARD or PRINT are not chosen.

Result: OK.

2. View the second item and save it to the default mailbox

   Purpose: Check the default save mail option. - This should append the current message to the user mailbox.

   Result: OK.

3. View the fifth mail item and save it in any file other than the default mailbox.

   Purpose: To test whether the system saves the message in a specified file

   Result: OK, but screen not entirely cleared after executing the task.

4. View the 7th mail item, forward it to user HIL and leave the item unchanged.

   Purpose: To test the forward option of COSNET MAIL, and to ensure that the automatic acknowledgement of messages works (As seen in Figure A.1, mail item 7 requires an ACK to PSK).

   Result: OK.

5. Exit mail display and inspect mailbox

   Purpose: Ensure that COSNET did save the mail item in the mailbox correctly.

   Result: OK.

A.1.2.2 ELECTRONIC FILING

The following actions have been designed to test the document handling functions of COSNET:

1. Create any 2 new folders

   Purpose: Test the creation of new folders within the own cabinet

   Result: OK.

2. Attempt to delete folder JUNK

   Purpose: Ensure that non-empty folders do not get deleted

   Result: OK.

3. Delete one of the folders created in (2)

Purpose: Test the deletion of empty folders

Result: OK.

4. Retrieve "ONE.DOC" from the "MSWORD" folder, edit it and return the document to the "MSWORD" folder.

   Purpose: Test the retrieval, updating and filing of documents

   Result: OK.

5. Retrieve "TWO.DOC" from the "MSWORD" folder and delete it

   Purpose: Ensure that documents can be deleted from the COSNET filing cabinet

   Result: OK, but document transfer script file displayed the transfer information, thus affecting menu screens during transfer. The screen was refreshed immediately after the transfer.

6. Retrieve the saved mailfile from folder "JUNK" and delete it

   Purpose: Check that the document was correctly saved and that it can be deleted.

   Result: OK, as above.

7. Retrieve from PSK's cabinet folder "JUNK" the document "JUNK", edit it and return it to its origin.

   Purpose: Test the retrieval of documents from another user's filing cabinet. The system should allow user STEVE to return the document to the cabinet, since he has write permission.

   Result: OK, STEVE had MS-WORD as default editor: as a result, the system did not display the filename in the directory listing that looks for "*.doc" files. The file was loaded into MS-WORD by loading "JUNK.".

8. Restore the deleted mailfile document

   Purpose: Test the "WASTEBASKET" system. The mailfile should automatically be restored to folder "JUNK"

   Result: COSNET did not restore the file properly into its original folder.

A.1.2.3 APPLICATIONS

At the start of the test session, no PC Applications or word processors are defined for the user:

1. Define the word processor "WORD" as the default word processor

   Purpose: Test the definition of a default word processor

   Result: OK.

2. Add the application "NU" (Norton Utilities)

   Purpose: Test the addition of a new application

   Result: OK.

3. Add the application "B" (Brief Editor)

   Purpose: Test the addition of a new application

   Result: OK.

4. Add the application "VI" (Another optional editor)

   Purpose: Test the addition of a new application

   Result: OK.

5. Change the application definition of "NU" to be "NI" (Norton Integrator)

   Purpose: To test the update of an existing application

   Result: OK.

6. Delete the application "VI"

   Purpose: To test the deletion of an existing application

   Result: OK.

7. Execute each application

   Purpose: To test the execution of the defined application

   Result: OK, some facilities called up from NI did not have enough memory to execute.

A.1.2.4 CALENDAR

Although both user have schedules set up in their calendars at the start of the tests, no ACCESS specifications to their calendars exist. This implies that no one but the owner may inspect the calendar.

1. Set WRITE permissions for users FRED, JVD, PSK, and PETE

   Purpose: Test the setting of WRITE permissions for users: - (Note that FRED is not a legal COSNET user)

   Result: OK.

2. Set READ permissions for users STUTZ and HIL

   Purpose: Test the setting of READ permissions for users

   Result: OK, "garbage" error message, when trying to add a user that already existed.

3. Add a PRIVATE schedule using the default time and date

   Purpose: To test the addition of a private schedule using the date and time supplied by the system as default

   Result: OK.

4. Add a schedule using the default day, at 15:00.

   Purpose: To test the addition of a schedule

   Result: OK.

5. Add a PRIVATE schedule using the default day, at 16:00.

   Purpose: To test the addition of a PRIVATE schedule

   Result: OK.

6. View the calendar schedules using the WEEKLY display mode (current week)

   Purpose: To test the WEEKLY display of schedules

   Result: OK.

7. View the calendar schedules using the MONTHLY mode (current month)

   Purpose: To test the MONTHLY display of schedules

   Result: OK.

8. Inspect HIL's calendar schedules for the current day

   Purpose: To test test the inspection of another user's calendar

   Result: OK.

9. Add a schedule for HIL at 15:30 today

   Purpose: Test the access to another user's calendar

   Result: OK.

10. Add a schedule for HIL at 9:00 tomorrow

    Purpose: Test the access to another user's calendar

    Result: OK.

11. View HIL's calendar using the WEEKLY display mode (current week)

    Purpose: To test the WEEKLY display of calendar

    Result: OK.

A.2 SECOND TEST

A.2.1 User name STEVE

A.2.1.1 MAIL

The following activities are designed to test the delivery of messages to users on COSNET. When you log on to COSNET, you should have new mail; the acknowledgement to mail sent in the first test from either PSK, STUTZ or JVD:

1. View a mail item and save it in a file in your JUNK directory (NB remember the keyword).

   Purpose: To test whether the system saves the message in a specified file, other than "mailfile".

   Result: OK.

2. Inspect acknowledgement sent from the user(s).

   Purpose: Ensure that COSNET delivered the mail item to the users correctly and sent the acknowledgement when the mail item was read.

   Result: OK.

3. Create an alias for user PETE. Create a new mail item, use the INTERRUPT facility to change the default editor to "b" (BRIEF) and return to "CREATE MAIL". Finish the message and send it, using the alias name just created.

   Purpose: To test interrupt facility and mail delivery using an alias instead the user name

   Result: Interrupt facility worked well, the inconsistant length of alias names allowed an aliasname longer than 8 characters to be created. But the system prompt anly accepts 8 characters entered for any username used in ***send mail***. Thus only alias names shorter than 8 characters worked correctly.

4. Create alias for users PSK, STUTZ, JACK, JVD; change PSK's alias to a new alias name

   Purpose: To test the COSNET aliasing system

   Result: OK.

A.2.1.2 ELECTRONIC FILING

The following actions have been designed to test the document handling functions of COSNET:

1. Retrieve the document saved using the mail menus by its keyword and delete it.

   Purpose: Test retrieving a mail item saved in the cabinet

   Result: Ok, apart from "F1 RETRIEVE" display corrupted.

2. Try to retrieve the mail item from folder JUNK, first by its name and then by its keyword.

   Purpose: Ensure that documents are deleted from the COSNET filing cabinet. The saved mail item should not be in the folder.

   Result: OK.

3. Restore the deleted mail item document

   Purpose: Test the "WASTEBASKET" system. The mailfile should automatically be restored to the original folder.

   Result: OK.

4. Retrieve the mail item document from folder JUNK and edit it.

   Purpose: Test the "WASTEBASKET" system. The mailfile should automatically be restored to the original folder.

   Result: OK.

A.2.1.3 CALENDAR

Although both user have schedules set up in their calendars at the start of the tests, no ACCESS specifications to their calendars exist. This implies that no one but the owner may inspect the calendar.

1. Set WRITE permissions for users FRED, JVD, PSK, and PETE

   Purpose: Test the setting of WRITE permissions for users: - (Note that FRED is not a legal COSNET user)

   Result: OK.

2. Set READ permissions for users STUTZ and HIL

Purpose: Test the setting of READ permissions for users

Result: OK.

3. Inspect PSK's calendar schedules for the current day

   Purpose: To test test the inspection of another user's calendar. (PSK is currently busy with the calendar)

   Result: OK, system message that calendar is in use.

A.2.1.4 COMMENTS

The following section discusses the reactions of STEVE to the general COSNET functionality:

A.2.1.5 What I liked about COSNET

User STEVE liked the following features of COSNET:

- The windowing feature of COSNET - i.e. the *user interface*
- COSNET's ability to choose own default word processor
- The general variety of COSNET features
- COSNET's HELP facility
- COSNET's design of navigating through the set of facilities

A.2.1.6 What I did not like about COSNET

The following list notes the items which STEVE did not enjoy about COSNET:

- STEVE found the mail menu confusing: He could not understand the difference between *inspect mail, inspect mail-log* and *inspect mailbox.*
- STEVE also mentioned, that he would prefer the function keys to automatically execute the option instead of needing to press RETURN.
- STEVE did not like the inconsistency in the user interface; "in some options, when you press a key, the function automatically executes and at other times RETURN has to be pressed".
- STEVE would prefer the field lengths to be equal to the maximum length of the input item; e.g. 8 characters for a user name.

A.2.2 User name HIL

A.2.2.1 MAIL

The following activities are designed to test the delivery of messages to users on COSNET. When you log on to COSNET, you should have new mail; the acknowledgement to mail sent in the first test from either PSK, STUTZ or JVD:

1. View a mail item and save it in a file in your JUNK directory (NB remember the keyword).

   Purpose: To test whether the system saves the message in a specified file

   Result: OK.

2. Inspect acknowledgement sent from the user(s).

   Purpose: Ensure that COSNET delivered the mail item to the users correctly and sent the acknowledgement when the mail item was read.

   Result: OK.

3. Create an alias for user PETE. Create a new mail item, use the INTERRUPT facility to change the default editor to "vi" and return to "CREATE MAIL". Finish the message and send it, using the alias name just created.

   Purpose: To test interrupt facility and mail delivery using an alias instead the user name

   Result: OK.

4. Create alias for users PSK, STUTZ, JACK, JVD; change PSK's alias to a new alias name

   Purpose: To test the COSNET aliasing system

   Result: OK.

A.2.2.2 ELECTRONIC FILING

The following actions have been designed to test the document handling functions of COSNET:

1. Retrieve the document saved using the mail menus by its keyword and delete it.

   Purpose: Test retrieving a mail item saved in the cabinet

   Result: OK.

2. Try to retrieve the mail item from folder JUNK, first by its name and then by its keyword.

   Purpose: Ensure that documents are deleted from the COSNET filing cabinet. The saved mail item should not be in the folder.

   Result: OK.

3. Restore the deleted mail item document

   Purpose: Test the "WASTEBASKET" system. The mailfile should automatically be restored to the original folder.

   Result: OK.

4. Retrieve the mail item document from folder JUNK and edit it.

   Purpose: Test the "WASTEBASKET" system. The mailfile should automatically be restored to the original folder.

   Result: OK.

5. Retrieve any document by DATE and AUTHOR and edit it.

   Purpose: Test the retrieval by date and author.

   Result: OK, The first retrieval displayed an empty file, after which the retrieval worked.

A.2.2.3 CALENDAR

Although both user have schedules set up in their calendars at the start of the tests, no ACCESS specifications to their calendars exist. This implies that no one but the owner may inspect the calendar.

1. Set WRITE permissions for users FRED, JVD, PSK, and PETE

   Purpose: Test the setting of WRITE permissions for users: - (Note that FRED is not a legal COSNET user)

   Result: OK.

2. Set READ permissions for users STUTZ and HIL

   Purpose: Test the setting of READ permissions for users

   Result: OK.

3. Inspect PSK's calendar schedules for the current day

Purpose: To test test the inspection of another user's calendar. (PSK is currently busy with the calendar)

Result: OK.

A.2.2.4 COMMENTS

The following section discusses the reactions of HIL to the general COSNET functionality:

A.2.2.5 What I liked about COSNET

HIL noted the following points in COSNET's favour:

- Good user interface
- Readily available help
- Intelligent system prompts
- Good error and system messages
- Customization of editor is good

A.2.2.6 What I did not like about COSNET

HIL did not like the following features:

- Slow response from the server
- No verification on the *delete* options

Appendix B

USER MANUAL

COSNET

USER MANUAL

Department of Computer Science
University of Cape Town
Rondebosch, South Africa 7700.

Contents

List of Figures

Chapter 1

GETTING STARTED

1.1 What is COSNET?

COSNET is an office automation system designed and implemented at the Computer Science Department of the University of Cape Town. COSNET improves the efficiency and quality of office work by providing the office worker with a set of automated office functions. Improved interpersonal communications and the provision of electronic filing and calendars are necessary features that an electronic office must support. COSNET not only supports these features, but also allows the office worker to integrate any MS-DOS based application to the office system.

1.2 COSNET Features

COSNET supports the following features:

- Word Processing
- Mail
- Electronic Filing
- Electronic Calendar
- MS-DOS Application

Documents can be created and updated by using an MS-DOS **editor or word processor**. This editor is defined by using the MS-DOS application facility.

Mail can be sent and received to/from any COSNET user at any time. The recipient needs not to be present to receive mail and can inspect the mail at any convenient moment.

Documents can be stored centrally in an **electronic filing** cabinet and may be shared by other users, provided they have permission to do so. Documents may also be electronically distributed to any COSNET user.

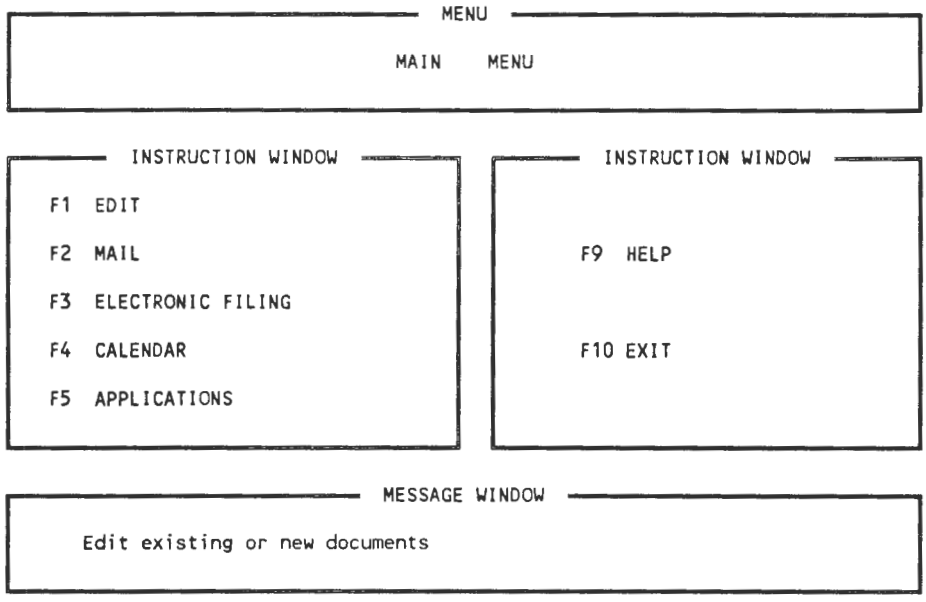


Figure 1.1: COSNET Main Menu

COSNET provides a **calendar** utility to handle a user's appointments and schedules. Each user has a private calendar which can be manipulated by other users that have been granted access to the calendar by the owner.

MS-DOS applications can be integrated with COSNET to enhance the office environment. These applications can be added to COSNET by the application menus.

These functions can all be accessed from the COSNET main menu as seen in Figure 1.1.

1.3 COSNET Installation

COSNET has to be installed on both the PC, and the UNIX TOWER. Two sets of diskettes are supplied for this purpose:

1. PC Installation diskettes
2. UNIX Installation diskettes

The following sections describe the COSNET installation process and requirements.

1.3.1 Hardware Requirements

COSNET minimum requirements consist of one PC and a NCR 1632 UNIX TOWER.

The PC should have a minimum of 250Kb memory, the current version of COSNET uses 230Kb. It is recommended though, to have 640Kb available, to be able to utilize large MS-DOS software application packages. COSNET provides menus for monochrome and color screens. A hard drive is strongly recommended, and at least 500Kb of disk space should be available for COSNET, which is loaded onto the PC from 360Kb floppy diskettes.

The UNIX installation diskette contains a UNIX file system and can currently only be read on a NCR 1632 TOWER Floppy Drive. COSNET UNIX installation requires 500Kb disk space on the UNIX file system initially.

A serial connection between the PC and the UNIX Tower is required, namely a standard asynchronous connection. The communication speed can be selected: - consult your UNIX administrator to set your terminal line speed.

1.3.2 PC Install

Install COSNET on your PC by following these steps:

- Insert the **PC Install Diskette** into **Drive A**.
- Set the default drive to Drive A - (if it is currently not the default, type *a:* at the DOS prompt and press *Enter*).
- Type *install* and press *Enter*.
- Now, follow the instructions displayed by **COSNET Install** on your screen.
- Once the **install** program has installed COSNET on your PC, you will want to add the COSNET *pathname* to **AUTOEXEC.BAT**.

The DOS *path* command tells your computer where to look for commands it does not recognize. DOS only recognizes programs in the current (logged) directory, unless there is a path to the directory containing pertinent programs or files.

The *PATH* variable in *autoexec.bat* tells DOS which directories to search for a command file, should it not be in the current working directory. Autoexec.bat must have "`\cosnet\bin`" as part of the *path* variable.

1.3.3 UNIX Install

Your UNIX. administrator should install COSNET on the UNIX TOWER. The following steps should be followed:

- Insert the **UNIX Install Diskette** into the floppy drive and mount the floppy device. E.g. "*/etc/mount /dev/fd70 /mnt/fd70*".
- Change the working directory to the mounted file system. E.g. *cd /mnt/fd70.*
- Type *install* and press *Enter*.
- Users may be added to COSNET as soon as the *install* program is complete.

- Type `a_users` *username* for each user, with the *username* not exceeding 8 characters.

COSNET is now ready to be executed.

1.3.4 COSNET Communications

COSNET uses KERMIT [KERM88] for communication between the workstations and the server. The communication of the workstation has to match the speed of the terminal line as defined on the server .

This is done by editing the file "`cosnet\bin\mskermit.ini`" and changing the "*set baud 9600*" statement. This procedure needs to be done once only during COSNET installation, or whenever the UNIX administrator decides to change the baud rate of the terminal line.

Chapter 2

COSNET FACILTIES

2.1 STARTING COSNET

To start COSNET at the workstation, just enter *cosnet* at the DOS prompt. As long as the path variable was set in *autoexec.bat* at COSNET installation, the program will be executed and a *login* menu is displayed, as illustrated in Figure 2.1.

This menu allows you to log into COSNET, display information about COSNET, call the HELP facility of this menu or simply exit to DOS.

2.1.1 F1 LOGIN

Login first inspects the status of the connection to the server and resets it if possible. Should the office automation server be inaccessible, a message is displayed to inform you.

Once the connection to the server is cleared, you will be prompted for your login name, as shown in Figure 2.2. You must type in your login name, as supplied to you by your COSNET administrator. The system accepts any input after you press **RETURN** or **ESC**.

You will then be prompted for your password. If you do not have a password, simply press **RETURN**, otherwise enter your password, followed by **RETURN** or **ESC**.

NOTE: Your password will not echo on the prompt screen, so you will not be able to see what you type. This feature is designed to prevent anyone from discovering your password.

After you have entered your login name and password, the system will automatically connect you to the server and access the files needed for COSNET operation. This process typically takes one to two minutes, during which time the system displays messages to you.

Once the system has established a connection to the server and all the necessary system files have been transferred, COSNET will inspect your mailbox. If new mail has arrived since you last used COSNET, you will be informed, and given the option to inspect your mailbox, as shown in Figure 2.3.

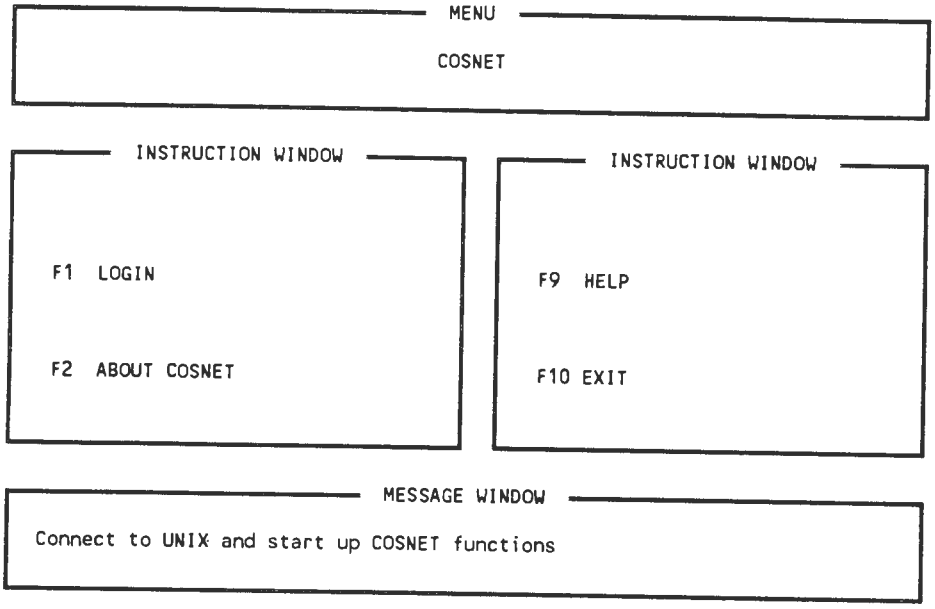


Figure 2.1: COSNET Login Menu

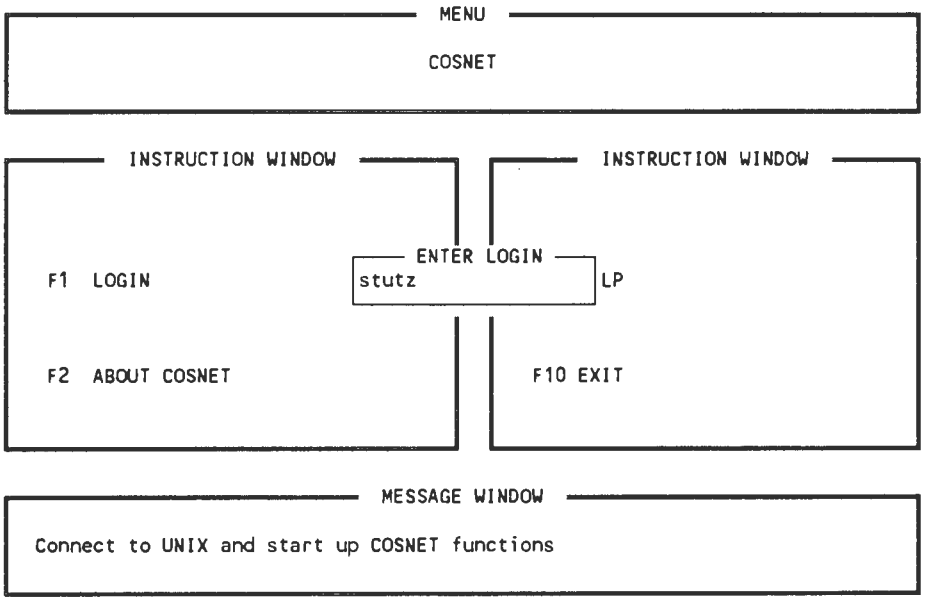


Figure 2.2: COSNET Login Prompt

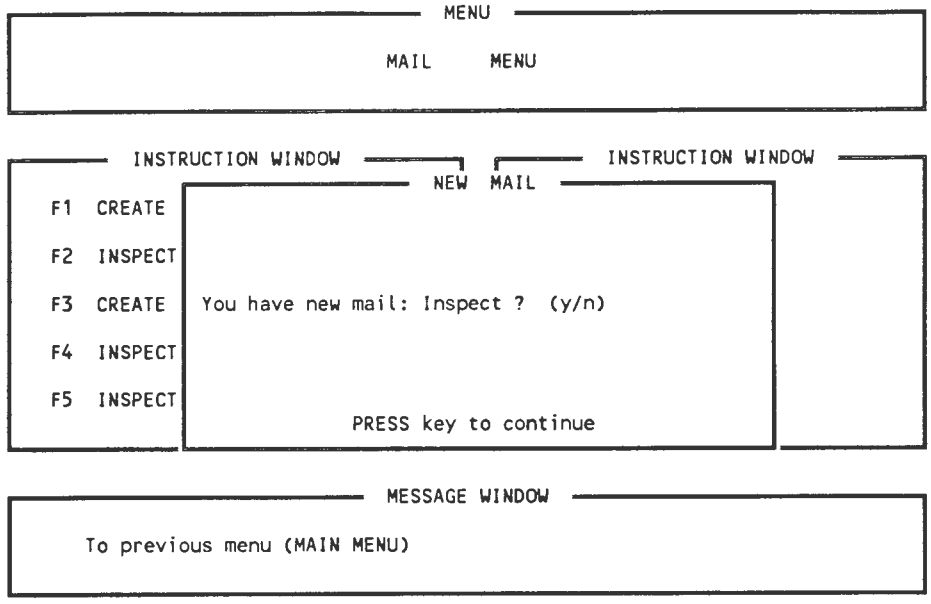


Figure 2.3: COSNET Prompt Menu for New Mail

If no mail exists, or if you decline to inspect your mail, then COSNET will display the **Main Menu**, as illustrated in Figure 2.4.

2.1.2 F2 ABOUT COSNET

Should you, as a newcomer, like to find out more about COSNET, this is the menu option you should choose. You will be able to inspect the COSNET "*QUICK HELP*" facility, which gives a brief description of all the available COSNET facilities.

A pop-up help menu appears on your screen, from which you can select any item to inspect. The screen is illustrated in Figure 2.5.

An item is selected by positioning the highlight bar on the required item, followed by pressing **Enter**.

The help information about the selected item is displayed in a *help window*. This help window covers the entire screen and you may scroll up and down through the displayed text. A command line at the bottom of the help window displays the commands that are available to the user. An example of a help window is shown in Figure 2.6. The pop-up help menu is restored as soon as you exit this help window.

2.1.3 F9 HELP

The *HELP* option of this menu enables you to inspect the help information to the facilities available from this menu. The format and operations on the help windows are the same as discussed in Section 2.1.2.

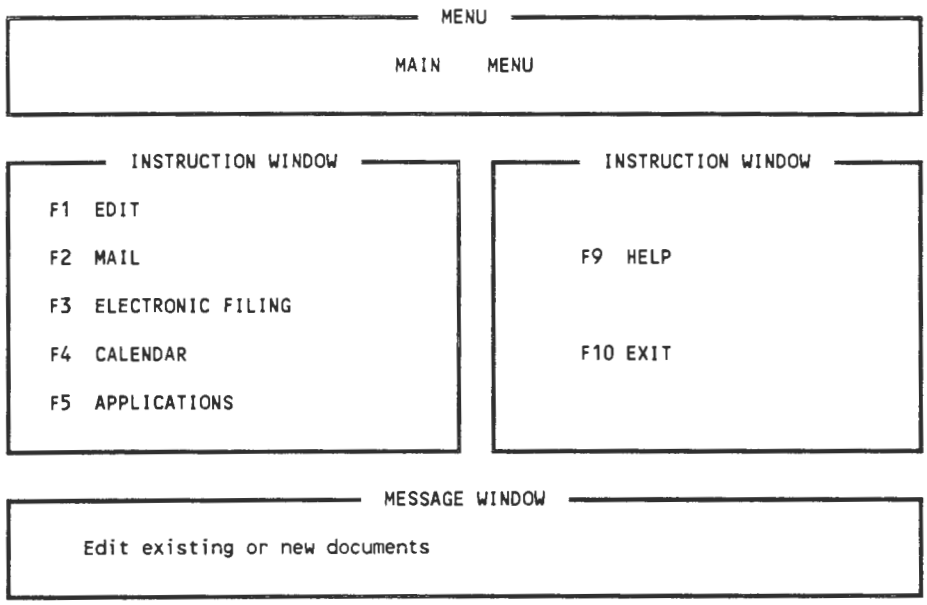


Figure 2.4: COSNET Main Menu

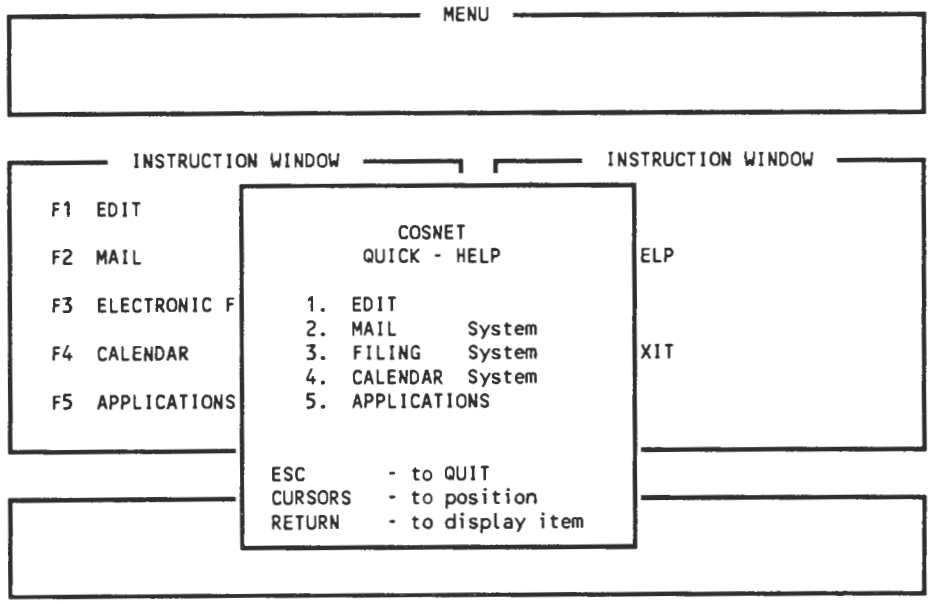


Figure 2.5: COSNET Pop Up Help Menu

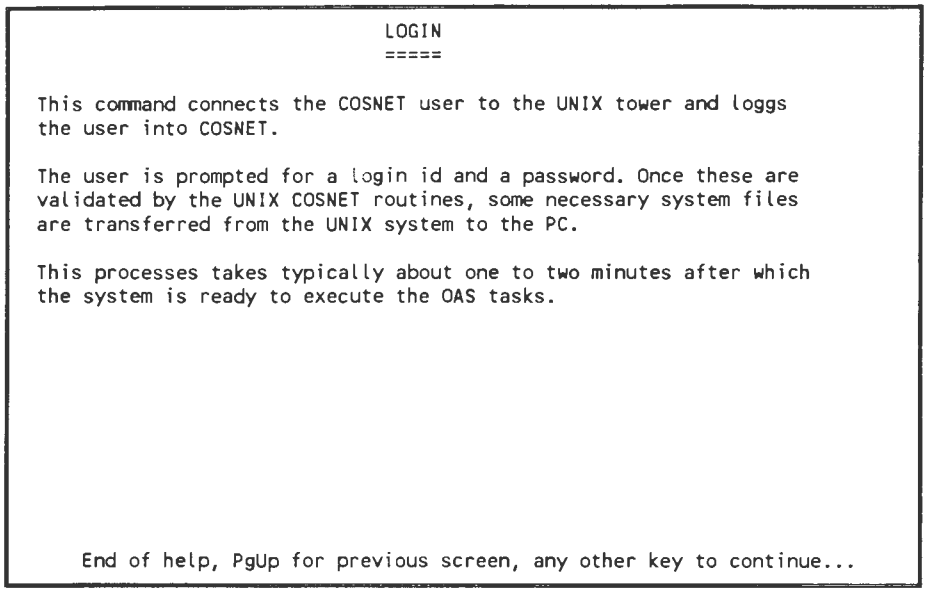


Figure 2.6: COSNET Help Window

The help facility is consistent in all COSNET menus and facilities and can be selected with the function key **F9**.

2.1.4 F10 EXIT

The *EXIT* option is also consistent in all COSNET menus. Generally, you will be returned to a calling menu when you select this option; in this particular case you will be returned to the DOS prompt.

2.2 MAIN MENU

The following sections describe the facilities, as illustrated in Figure 2.4, that are available from the main menu.

2.2.1 F1 EDIT

This option is used to invoke an editor or word processor. The system displays the defined default editor in its prompt, which you can accept by pressing **RETURN**, or you may type the name of another editor that you may wish to use. The editor must be resident on your workstation and its home directory must be defined by the *path* environment in AUTOEXEC.BAT as described in Section 1.3.2.

Next, COSNET displays a list of the documents in your COSNET working directory, as shown in Figure 2.7. You may choose any one of these files, by typing its name in the prompt window, or you may decide to create a new file. This is achieved by typing in a name that does not exist in the file display window.

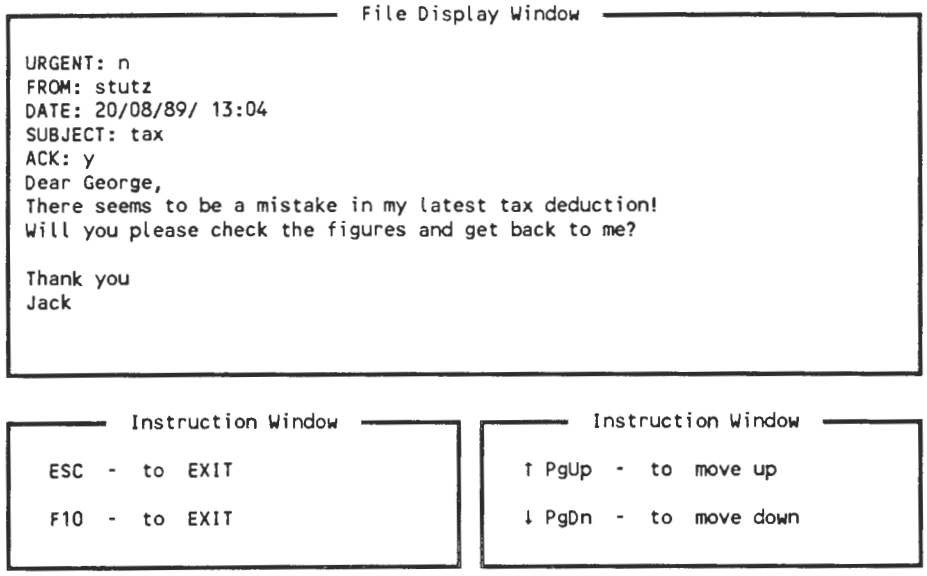


Figure 2.7: File Display Window

After the file is selected, it will be placed in the editor and you may proceed to update the document.

COSNET will automatically return you to the main menu after you quit the editor.

2.2.2 F2 MAIL

The COSNET mail facility allows you to create and inspect mail, manipulate alias, inspect mail-log and mailbox. The menu for the mail facilities is shown in Figure 2.8.

2.2.2.1 F1 CREATE MAIL

This menu option is used to create new mail messages that can be mailed to one or more COSNET users. Figure 2.9 illustrates a typical mail creation screen. The system prompts you to enter the *URGENT, SUBJECT* and *ACK* parameters, and completes the *FROM* and *DATE* options automatically. You will then be placed into *edit* mode as seen in Figure 2.9 and can proceed to write your message.

A simplified version of an editor is used to create mail and you may only change the contents of the current line. The editor does not permit you to scroll back and edit lines previously created.

There are two ways of leaving the mail editing facility, as can be seen in Figure 2.9.

You may want to interrupt the editing of the message and perform some other COSNET task, and to complete the message at a later stage. To achieve this

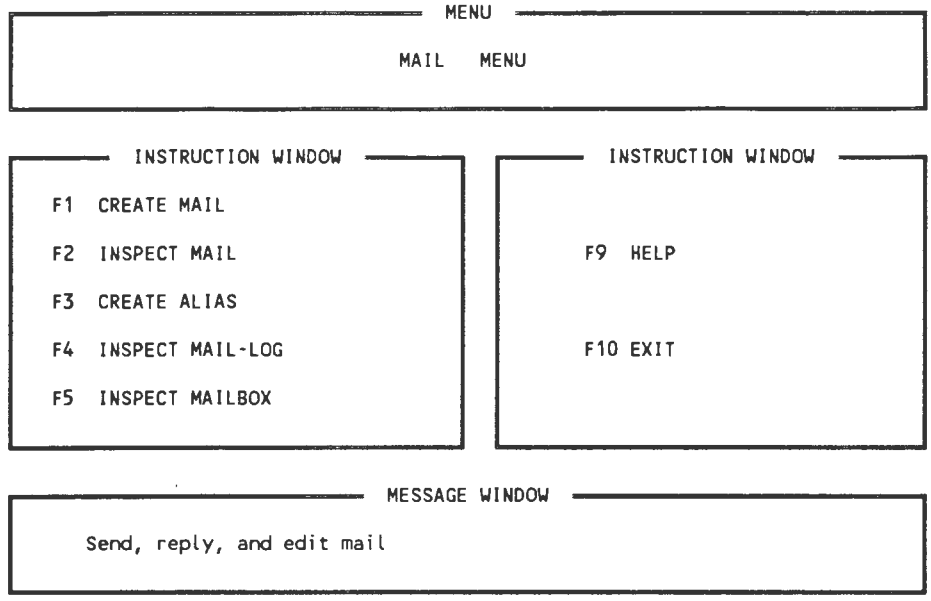


Figure 2.8: Mail Menu

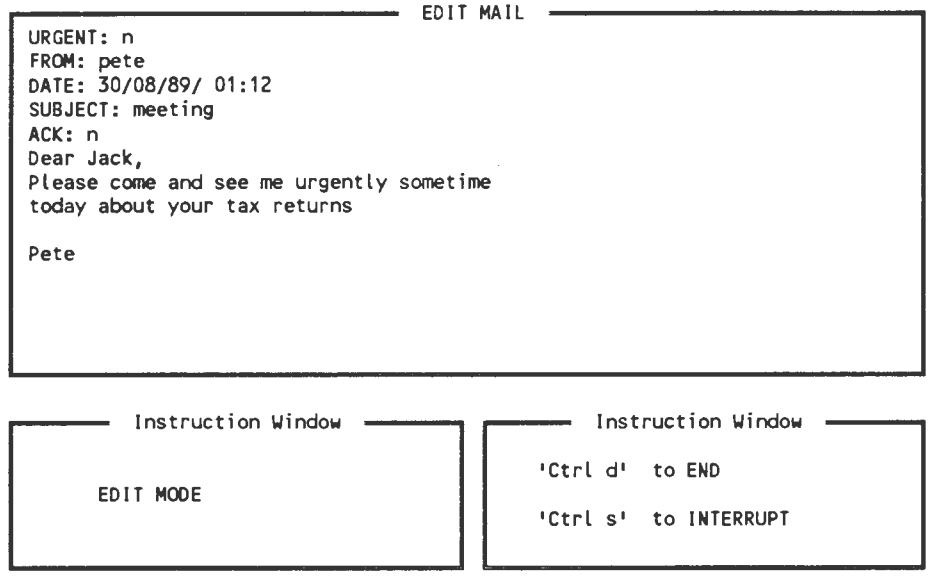


Figure 2.9: Mail Creation Screen

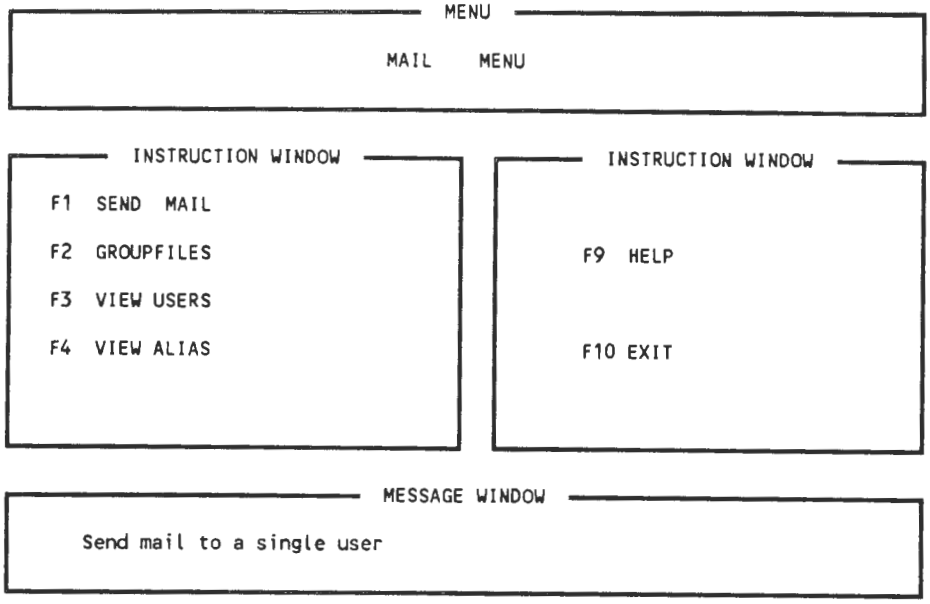


Figure 2.10: COSNET Mail Send Menu

you would have to press *Ctrl-S*, as shown in Figure 2.9; i.e. press *Ctrl* and *S* simultaneously. COSNET will then call the Main Menu as shown in Figure 2.4. You can now perform any task available, such as inspecting calendar schedules, and at any later stage return to the *CREATE MAIL* menu. Your message is restored to the exact state it was in when the interruption occured, and you may proceed to complete it.

Once you have finished editing the message and are ready to send your mail item, you may quit this screen by pressing *CTRL-D*; i.e press *Ctrl* and *D* simultaneously.

The system now displays the *Send Mail Menu*, as shown in Figure 2.10. If you are not too sure who the COSNET users are, or what alias you have defined, you can inspect the current user and alias definitions by selecting options *F3 VIEW USERS* and *F4 VIEW ALIAS* from the Send Mail Menu shown in Figure 2.10.

F1 SEND MAIL

Most of the time you will not need to send messages to more than one user. Once you have selected this option, COSNET will prompt you to enter a user name. The message that you have just created will be sent to the recipient, provided you have entered a legal user or alias name.

A copy of the letter and the addressing information is saved in a file called *mail-log* every time you send a message. You can inspect this file and read its contents to follow up any queries you might have regarding sent mail.

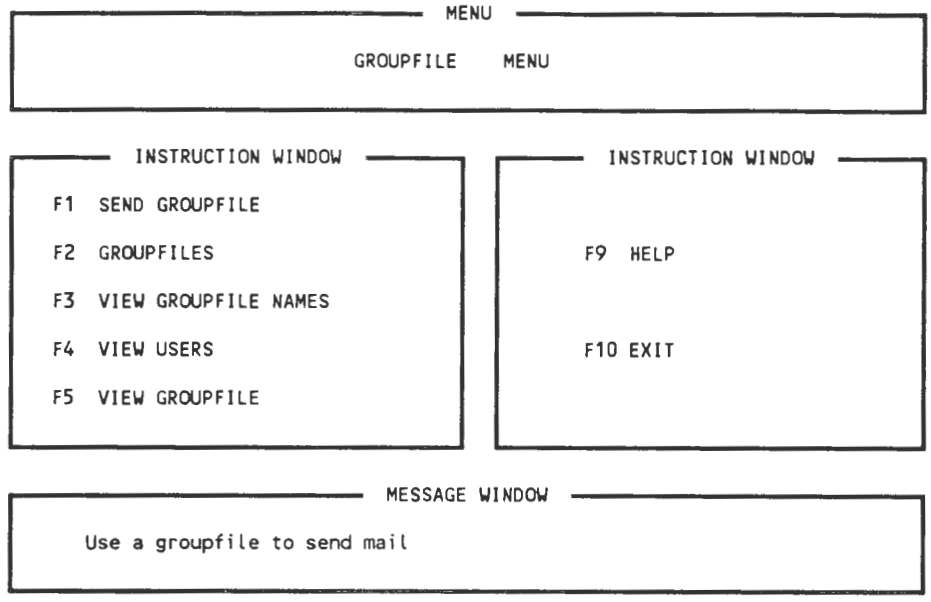


Figure 2.11: COSNET Groupfile Menu

F2 GROUPFILES

This option allows you to send mail using a distribution list, or *groupfile*. A groupfile is a collection of user names to which you may periodically want to send messages. Instead of typing each user name as for *F1 SEND MAIL*, COSNET will use such a groupfile that you have created, and send the message to each user listed in the file. The groupfile menu is shown in Figure 2.11.

F1 SEND GROUPFILE - (Groupfile Menu)

COSNET displays the current groupfiles and requests you to enter a groupfile name. The display of the file names is the same as shown in Figure 2.7.

Once you have entered a correct groupfile name, COSNET commences to send the message to each of the users listed in the file.

F2 GROUPFILES - (Groupfile Menu)

To *create, update* and *delete* groupfiles, you will need to choose this menu option. The menu relating to these functions is shown in Figure 2.12.

F1 UPDATE GROUPFILE

The *update groupfile* option lets you change the contents of an existing groupfile; you can either add a new name to a groupfile or delete a name from the file.

COSNET first displays all the current groupfile names and prompts you to enter a name. The system verifies if the name you entered exists, and then displays the

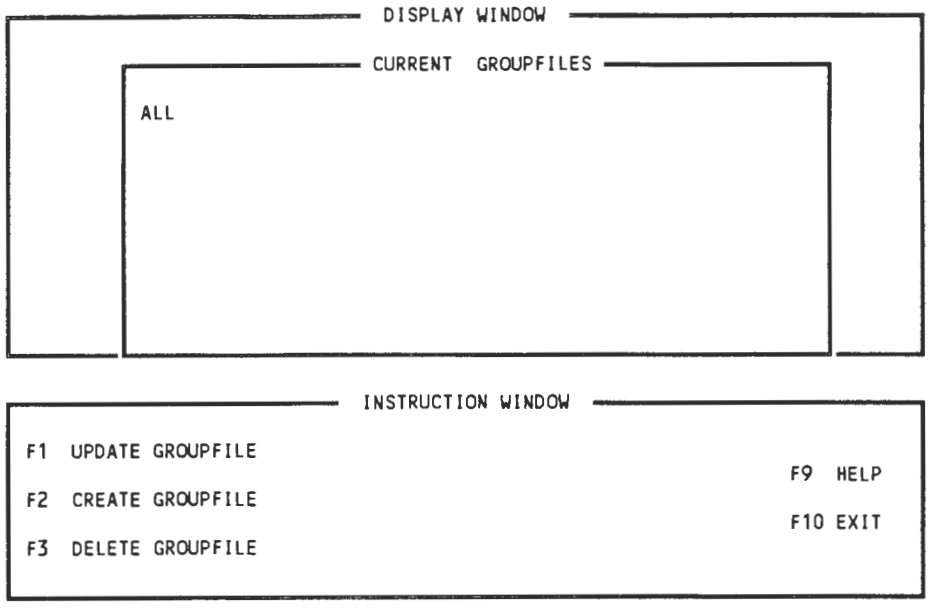


Figure 2.12: Groupfile Manipulation Menu

contents of the groupfile.

You can then enter the user name that you would like to add/delete. An example of the windows is shown in Figure 2.13. Once again, your input is verified, and the task is executed. The contents of the updated groupfile are displayed immediately after the update occurs.

You can also view the list of current users on COSNET, as well as a list of your alias names, before updating your groupfiles.

F2 CREATE GROUPFILE

The *create groupfile* option lets you create a new mail list.

COSNET first displays all the current groupfile names and prompts you to enter the name of the new groupfile. If no groupfile with such a name exists, you will be promted to enter a user name or alias.

After each new name that you add to the groupfile, the contents of the file are displayed, as illustrated in Figure 2.13. To stop adding names to the groupfile, simply press **RETURN** or **ESC** when prompted for a new user name.

F3 DELETE GROUPFILE

The *delete groupfile* option allows you to delete an existing groupfile.

COSNET will firstly display all the current groupfile names and will prompt you to enter the name of the groupfile to be deleted. If a groupfile with such a name exists, it will be deleted, and the updated list of groupfiles will be displayed, as shown in Figure 2.7.

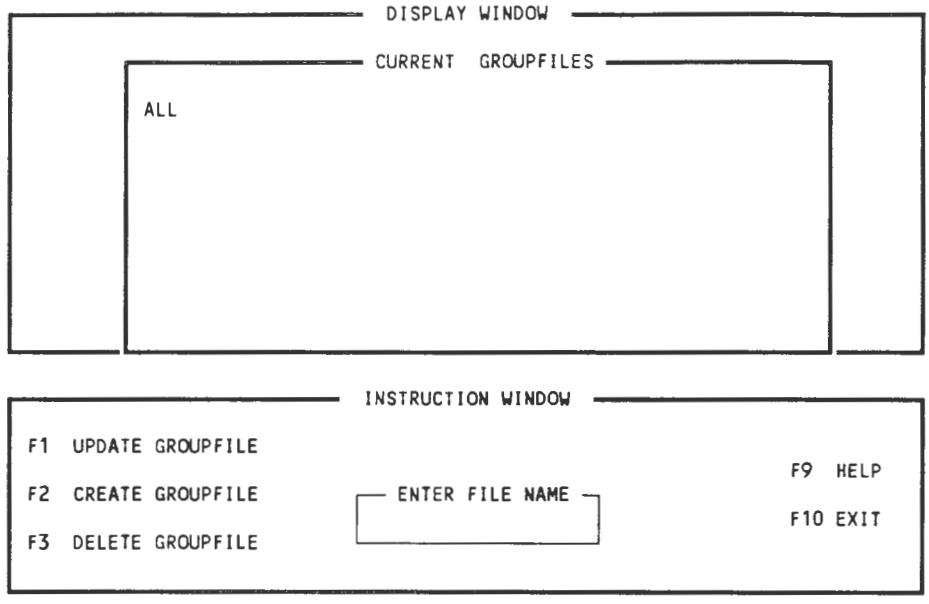


Figure 2.13: Creation of Groupfiles

F3 VIEW GROUPFILE NAMES - (Groupfile menu)

Should you forget how many groupfiles you have and what their names are, you can inspect the list of your groupfiles with this option. Figure 2.7 illustrates the format of the display.

F4 VIEW USERS - (Groupfile Menu)

You can also inspect the current COSNET user using this option; this might be necessary if one of your groupfiles contains users that do not exist anymore.

F4 VIEW GROUPFILE - (Groupfile Menu)

You might also want to inspect the contents of a particular groupfile, before you choose to use it to distribute your message.

COSNET first displays all the current groupfile names and prompts you to enter a file name. The system verifies whether the name you entered exists, and then displays the contents of the groupfile, as shown in Figure 2.7.

2.2.2.2 F2 INSPECT MAIL

This option is used to look at your current mail items. These could consist of new messages, or pending mail items that you have previously not inspected or ignored.

COSNET first displays a messsage in the message window, informing you that your mailfile is being retrieved from the server. After the completion of the mailfile transfer, COSNET will display all your mail items in the *Mail Display Window*, as

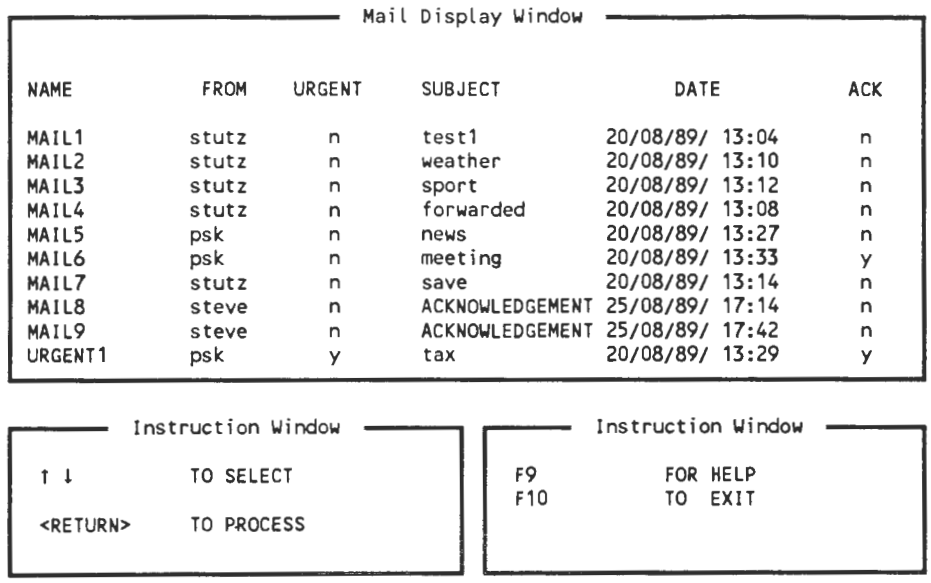


Figure 2.14: COSNET Mail Display Window

illustrated in Figure 2.14. COSNET informs you who the message is from, the class of the message (urgent or not), subject of the message, date of creation and whether it needs to be acknowledged or not.

From this display window you can select any mail item that you would want to inspect. This is done by selecting a mail item using the up and down arrows on your keyboard followed by **RETURN**. COSNET then displays the contents of the message in a *Message Display Window*, as shown in Figure 2.15. You can read the mail item by scrolling through its contents, but you cannot edit it. To quit from this display you can either use **F10** or **EXIT**. When you have finished reading your message, you can *save, delete, forward* or *print* the message. The options available are shown in Figure 2.16.

F1 SAVE

The message can be saved in the following two ways:

- You can save the message in a file called *mailbox*. Each time you save a mail item in this file, the message is appended to the contents of the mail mailbox. You can then inspect the contents of the mail box as shown in Section 2.2.2.5.
- COSNET also gives the option to save the mail item in your filing cabinet. This is very useful, since you will often receive documents via the mail system. For the procedure of saving a document in your filing cabinet refer to *F2 FILE* in Section 2.2.3.2.

NOTE: COSNET will delete your mail item the moment you are using one of the two above-mentioned mothods, and will return you to the mail display screen

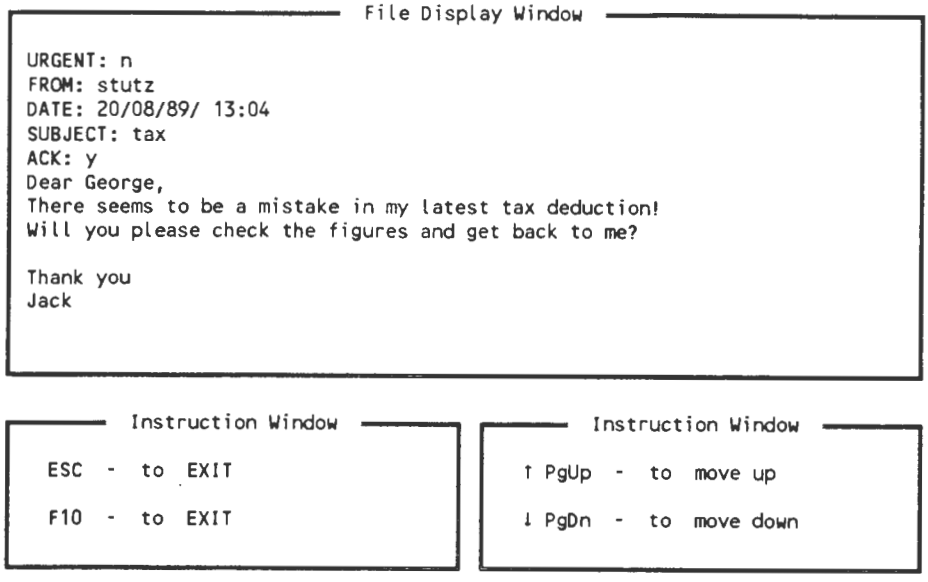


Figure 2.15: COSNET Message Display Window

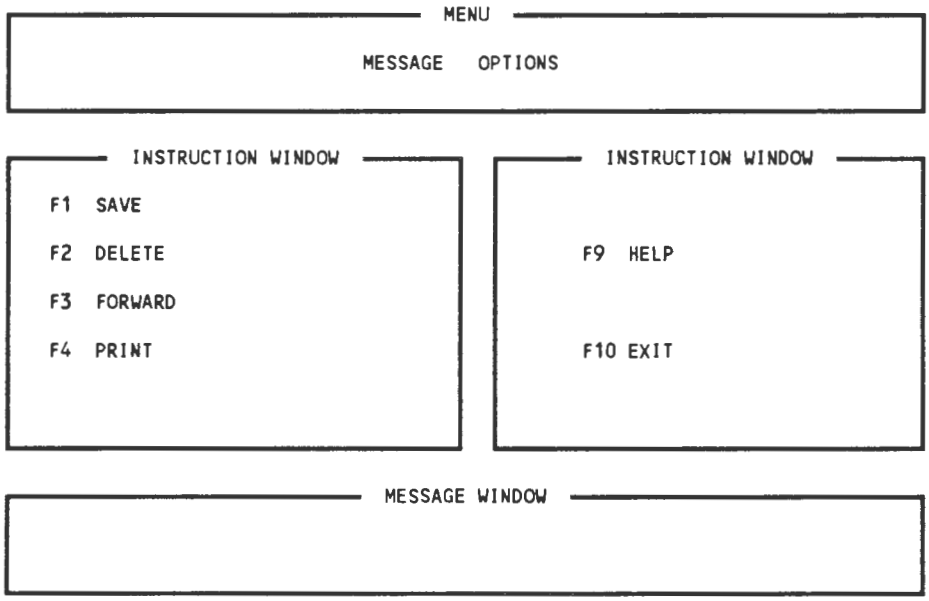


Figure 2.16: Mail Options Menu

illustrated in Figure 2.14.

F2 DELETE

This option simply deletes the mail item from the mail display list and returns you to the mail display screen, as shown in Figure 2.14.

F3 FORWARD

You can send this message to another user in exactly the same form as you received it. The message will indicate that it had been forwarded from you when the recipient inspects it. Forwarding a mail item is performed in the same way as sending mail, described in Section 2.2.2.1. The message is not deleted if being forwarded, and the mail option menu, as seen in Figure 2.16, is displayed after completion of this task.

F4 PRINT

COSNET allows you to print the message on your default printer, provided you have one. The message is not deleted if being printed, and the mail option menu, as seen in Figure 2.16, is displayed after completion of this task.

2.2.2.3 F3 ALIAS

The alias facility allows you to associate a nickname with a COSNET user name. You may, for instance, want to create an alias for user "worthingtonagm" to avoid typing this long name. By creating an alias "andy", for example, you would need only to type "andy" every time you would want to send mail to "worthingtonagm". COSNET lets you *create, delete* and *change* your alias definitions. From the alias menu, as illustrated in Figure 2.17 you can also select options *F1 DISPLAY USERS* and *F2 DISPLAY ALIAS* to inspect the latest users and alias names.

Before deleting, updating and creating alias names, the current alias names are displayed. COSNET will then prompt you to enter the necessary data for the deletion, creation or update of the alias names.

2.2.2.4 F4 INSPECT MAIL-LOG

Mail-log is a file updated every time you send a message via COSNET mail. The details of the message are appended to this file for future reference. You can inspect this file's contents, which are displayed using the *file display window*, as illustrated in Figure 2.15.

2.2.2.5 F5 INSPECT MAILBOX

Your mail is saved in your *mailbox* by default, if you do not specify an alternate file name upon saving a message. Your message is appended to the contents of this file, containing other previously saved mail items. You can inspect this file's contents, which are displayed using the *file display window*, as illustrated in Figure 2.15.

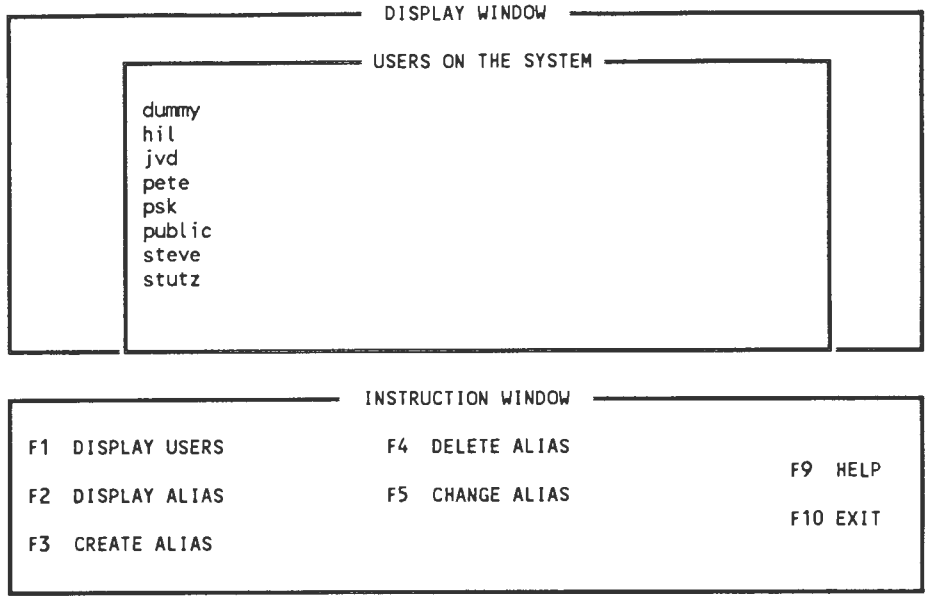


Figure 2.17: COSNET Alias Menu

2.2.3 F3 ELECTRONIC FILING

You have a private filing cabinet which consists of a number of folders, in which all your documents are stored. You may allow other users to manipulate your documents, but they may not create new documents in your cabinet.

COSNET's electronic filing facility lets you ***retrieve, file, distribute*** and ***restore*** documents. You may also create and delete folders within your own filing cabinet. The Electronic Filing Menu is shown in Figure 2.18.

2.2.3.1 F1 RETRIEVE

Once you have chosen this option, you will have to specify whether you want to retrieve a document from *OWN, PUBLIC* or *OTHER* filing cabinet. Once you have selected the source filing cabinet, you must decide on the ***method*** of retrieval, as shown in Figure 2.19.

F1 NAME - (Retrieval Method Menu)

COSNET will firstly retrieve your cabinet information files from the server, and display all the folder names that exist in your cabinet, as shown in Figure 2.20. You must enter the folder name from which you want to retrieve the document.

COSNET checks if you have entered a valid folder name, and then displays all the document names that exist within that folder. You may then choose the document that you wish to retrieve. The document names are displayed in the same format as the folder display shown in Figure 2.20. COSNET then "fetches" the document from the server once you have made your selection, and displays the Document

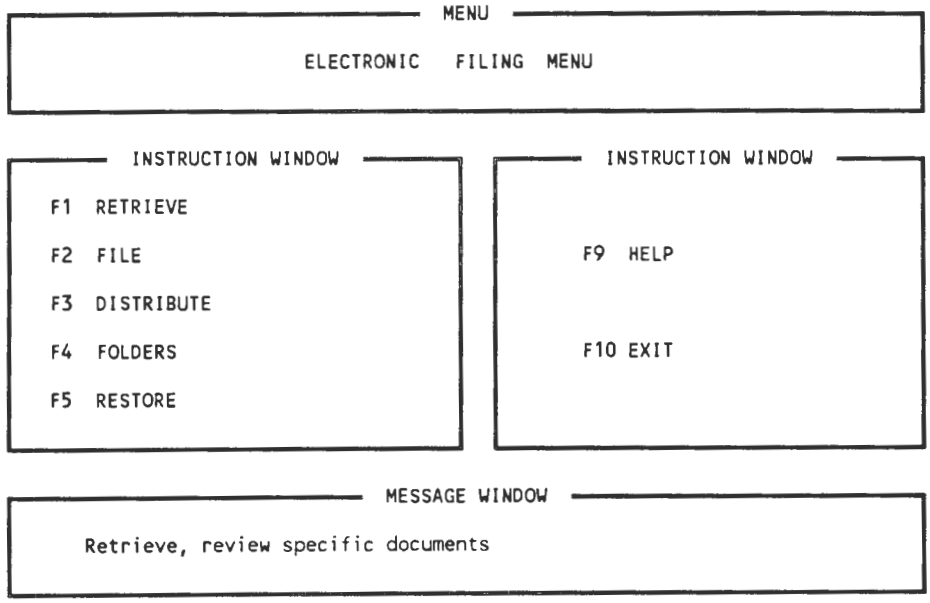


Figure 2.18: COSNET Electronic Filing Menu

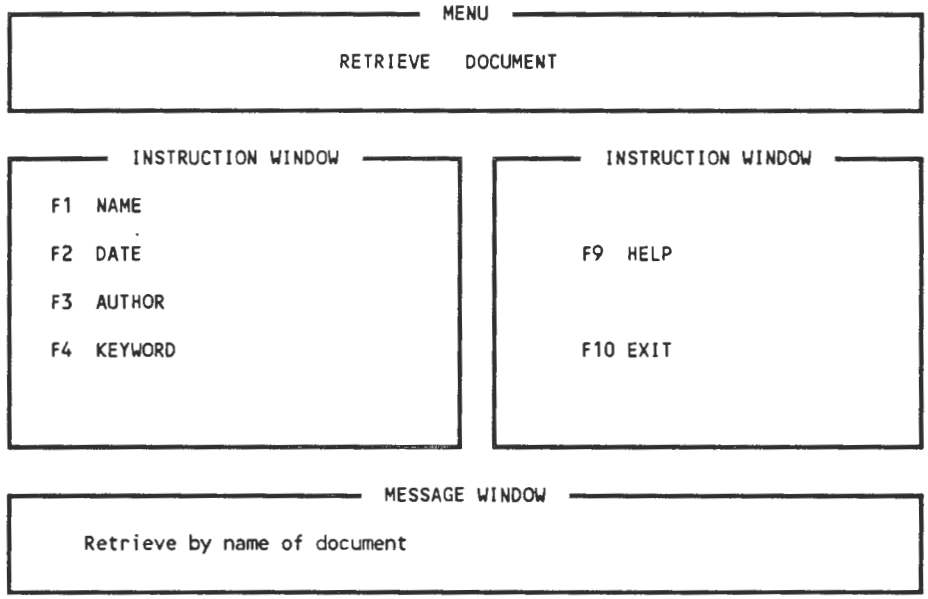


Figure 2.19: Retrieval Method Menu

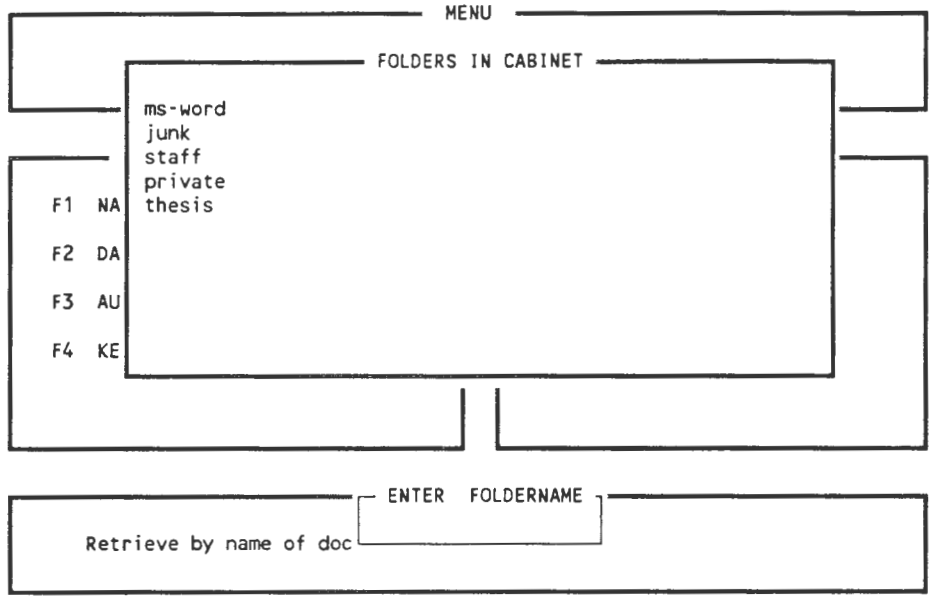


Figure 2.20: COSNET Folder Display

Handling Menu, as illustrated in Figure 2.21. You may *delete* the document from the cabinet provided you have the permission to do so. You can delete all your personal documents, but you will need *write* permission to a document not owned by you, should you want to delete it.

The document is deleted from the folder and is stored in the "wastebasket". You can later *restore* this document to its original folder should you wish to do so. This process is described in Section 2.2.3.5.

You can *distribute* the document to other one or more users. The procedure is exactly the same as sending mail. For details see Section 2.2.2.1.

If you decide to *edit* the document the system will give you the option to choose an editor; the default editor name appears in the prompt window and you can either accept this by pressing **ENTER** or **ESC**, or choose a new editor by typing its MS-DOS command name.

The document is then loaded into the editor and you may proceed to change its contents. The Document Handling Menu, as shown in Figure 2.21 will be restored when you exit the document.

You will need to select option *F4 STORE* **only** if you want to store the document in a folder other than where the document was retrieved from. COSNET automatically returns the document to its original folder when choosing *F10 EXIT* in the Document Handling Menu.

See Section 2.2.3.2 for procedure of storing the document in another folder.

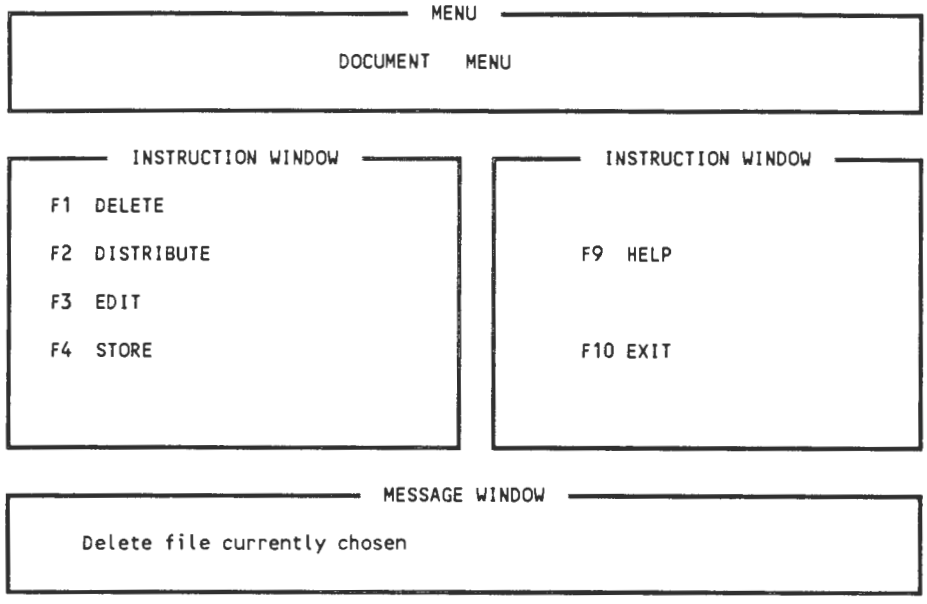


Figure 2.21: COSNET Document Handling Menu

2.2.3.2 F2 FILE

If you have created a new document you will want to store in a cabinet on the server.

You must first select the document from the list displayed to you, as in Figure 2.7. Then you must choose the cabinet that you want to store the document in, ie. *OWN, PUBLIC* or *OTHER*. The current version of COSNET only allows you to store the document in OWN and PUBLIC cabinets. COSNET then retrieves the information files pertaining to the chosen cabinet, and displays the cabinet's folders, as shown in Figure 2.20.

COSNET asks you to enter a folder name in which the document will be stored. If the document has no information file, you will be asked to fill in the fields on the information file form, as shown in Figure 2.22. This information is needed by COSNET for document retrieval. The fields *FILENAME, DATE* and *AUTHOR* are automatically completed by COSNET and you must supply the document's keyword and ***read*** and ***write*** users. If you decide not to enter any read and write users, no-one but you will be able to manipulate the document.

COSNET will file the document, once you have completed the information form, and will update the cabinet's information files to reflect the new entry.

2.2.3.3 F3 DISTRIBUTE

You may also choose to distribute a newly created document. The sequence of action is the same as described in Section 2.2.2.1.

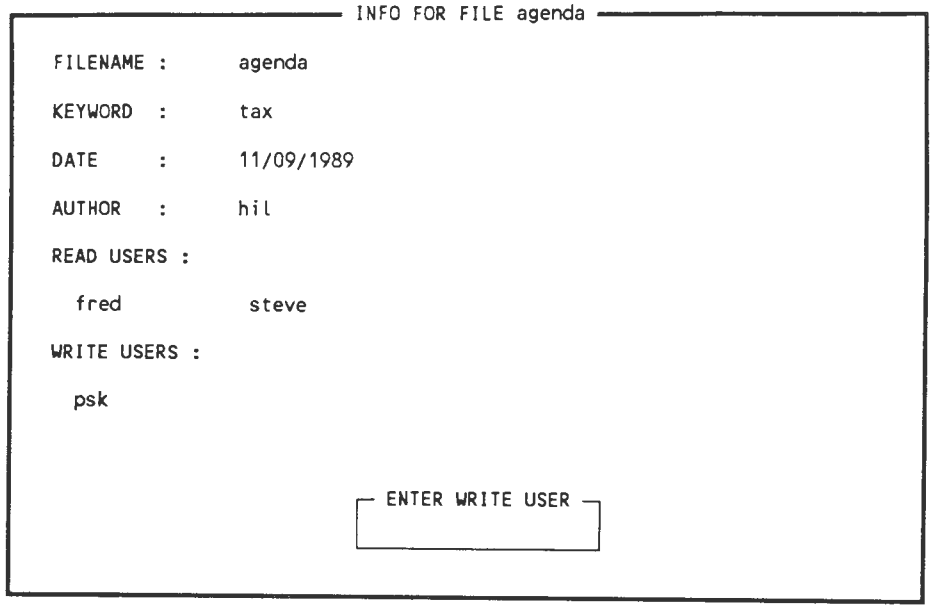


Figure 2.22: COSNET File Information Screen

2.2.3.4 F4 FOLDERS

COSNET allows you to *create* and *delete* folders in your OWN or in the PUBLIC cabinet.

A list of the current folders in the cabinet is displayed before folder creation or deletion, as shown in Figure 2.20. You may only delete a folder that is empty.

2.2.3.5 F5 RESTORE

If you realize that you have accidentally deleted a document, then you may *restore* it from the *wastebasket*. Every document that is deleted, is copied into a folder called wastebasket.

When you choose this menu option, COSNET displays the wastebasket contents and lets you select a document. The selected document is then automatically restored to the folder where it was deleted from.

COSNET automatically updates the cabinet structures and deletes the document from the wastebasket.

2.2.4 F4 CALENDAR

After you have chosen the calendar option from the Main Menu, you need to inform COSNET whether you would like to inspect your own calendar, or the calendar of another user. Should you choose to manipulate someone else's schedules, a list of COSNET is displayed, from which you may choose a name.

COSNET then retrieves the calendar and calls the Calendar Menu, as illustrated in Figure 2.2.4.

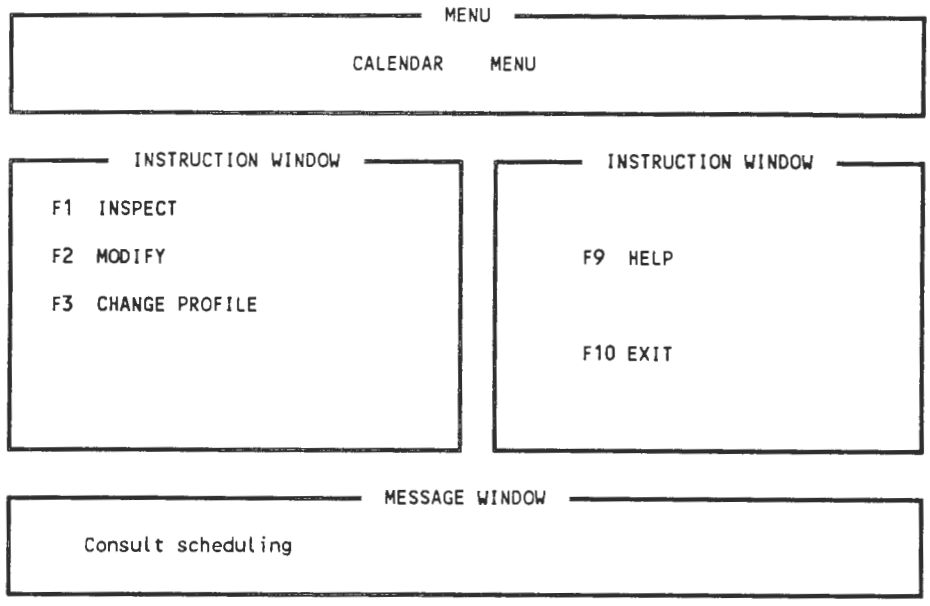


Figure 2.23: COSNET Calendar Menu

2.2.4.1 F1 INSPECT

You can inspect your calendar schedules in three different modes; the *daily, weekly* and *monthly* modes.

The daily mode is a detailed display of your schedules. You must first enter the date of the day that you want to inspect. The system default day is the current date.

Your schedules for the particular day are then displayed, as shown in Figure 2.24. The day is divided into three parts, *morning, afternoon* and *evening* and each part of the day has its own display window. You can select between the windows by pressing the **ESC** key. A daily schedule consists of the *time* of the schedule, a *keyword* to identify the nature of the schedule, and a *description* giving detailed information pertaining to that schedule.

The *weekly* display has one window for each weekday, and only the time and the keyword of a schedule are displayed. You can change the active display window by using the **ESC** key, as in the daily display mode. A sample weekly display screen is shown in Figure 2.25.

COSNET also supports a *monthly* display of calendar schedules. The display starts with the first entry for a particular month, and lets you scroll through all the schedules of the month. Two display windows are used to fit more information on one screen, as shown in Figure 2.26.

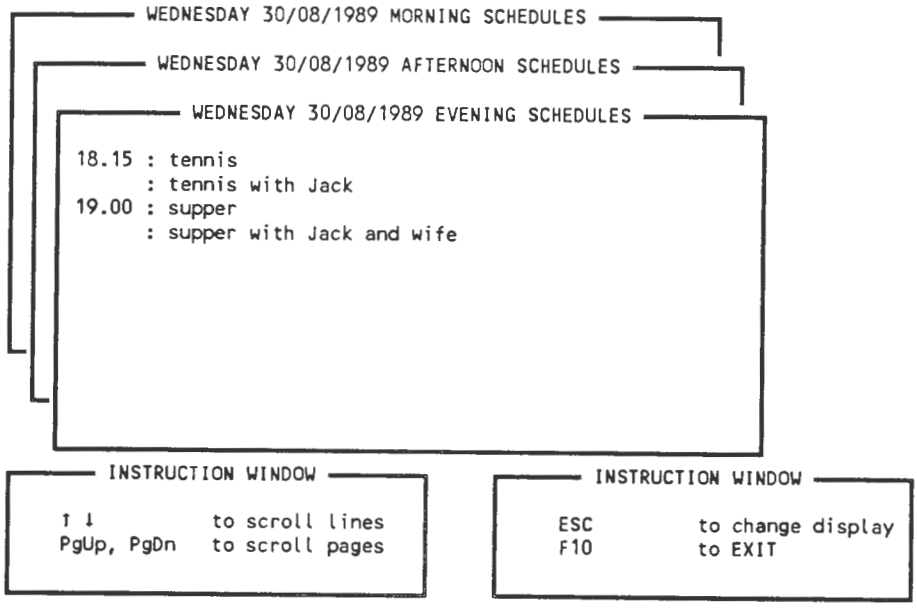


Figure 2.24: COSNET Daily Calendar Display

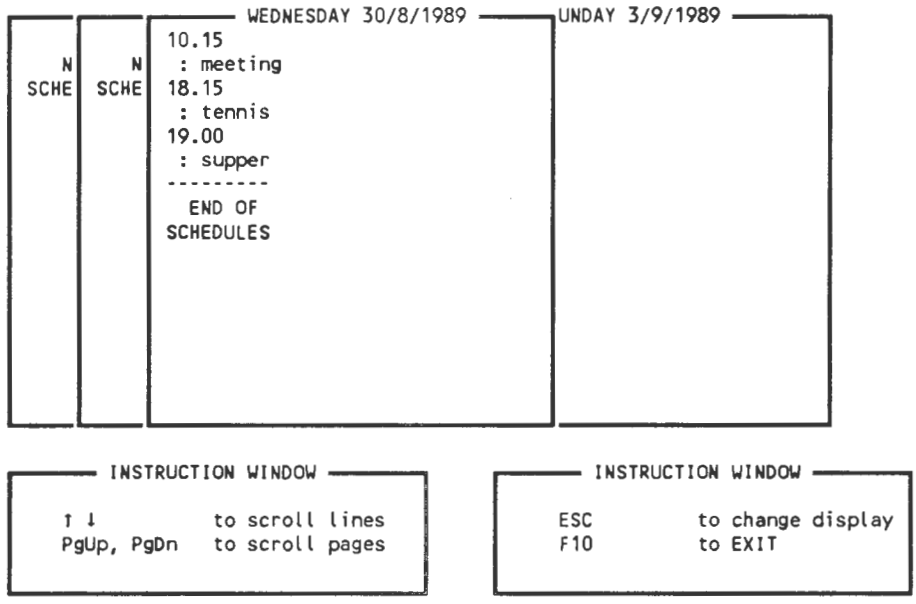


Figure 2.25: COSNET Weekly Calendar Display

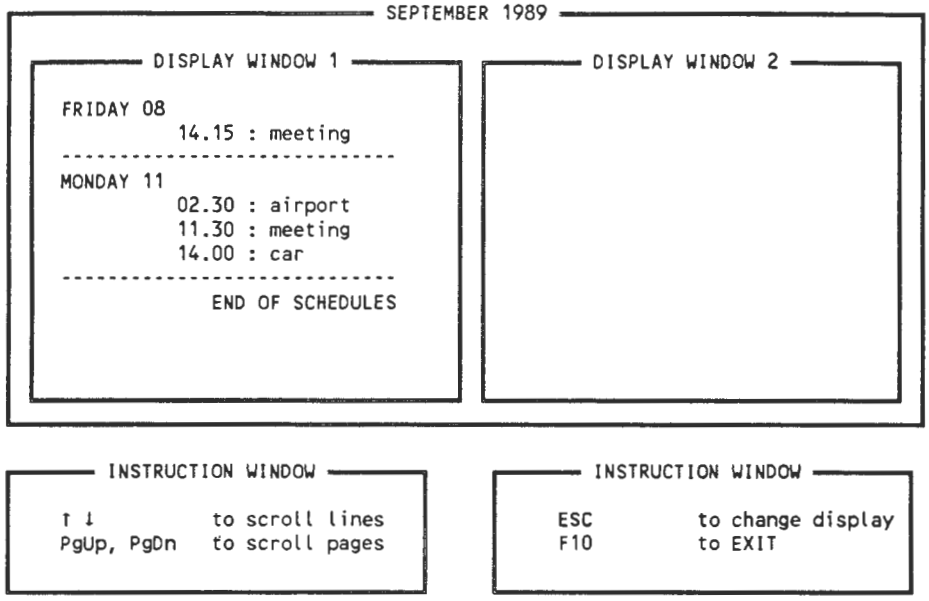


Figure 2.26: COSNET Monthly Calendar Display

2.2.4.2 F2 MODIFY

Use this option to update the schedules of the calendar that you have retrieved. The schedules for the day that you have chosen are displayed in the daily format, as in Figure 2.24, before COSNET prompts you to enter the new schedules. COSNET always checks the data fields that you enter, and demands that the data entry is repeated, if you submit an incorrect value.

COSNET calendar uses 15 minute slots for each schedule. Should a particular calendar slot already be in use, then you may re-enter your schedule at a different time, or you may change the details of the existing schedule.

2.2.4.3 F3 ACCESS MODES

Two different access modes to a COSNET calendar exist; the *read* mode and the *write* mode. Initially, no one else has access to your calendar, but you may selectively grant access to other COSNET users.

A user with *read* access may not update your calendar, but can inspect the schedules contained within.

A user with *write* access may update and inspect your schedules, but will not be able to read your schedules marked as *private*.

For both access modes you can *add* and *delete* a user, as shown in Figure 2.27. COSNET will display the read or write users that you have currently defined and prompt you to enter a user name to either add or to delete.

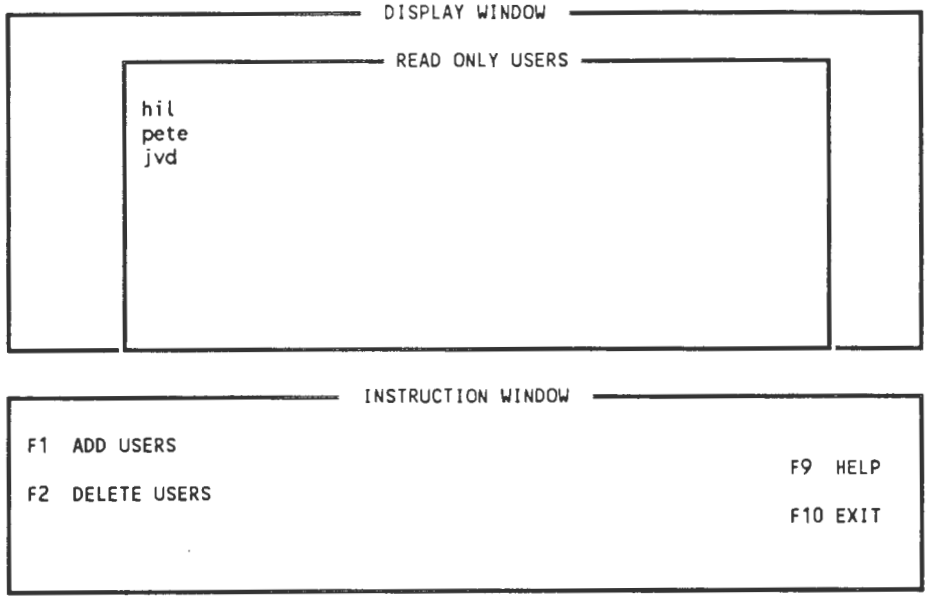


Figure 2.27: COSNET Update Access Mode Screen

2.2.5 F5 APPLICATIONS

The *applications* menu option is used to manipulate MS-DOS application software. You can add up to 10 MS-DOS applications to COSNET. The applications menu is shown in Figure 2.28. The current version of COSNET uses 230Kb of memory on the workstation, so the application has the rest of the workstation's memory available. The pathname of the application must be specified in the *Command-Line* field, as seen in Figure 2.29, unless the pathname is defined in the PATH variable in AUTOEXEC.BAT on the workstation.

The application screens of *F1 DISPLAY APPLICATION, F2 ADD APPLICATION, F3 DELETE APPLICATION, F4 UPDATE APPLICATION* and *F5 EXECUTE APPLICATION* are the same, except for the instruction line at the bottom of the window. The *ADD APPLICATION* screen is shown in Figure 2.29. An application is selected by positioning the cursor over the application, followed by pressing **ENTER**.

You may also use the application facility to define the default editor or word processor, which is used by the *edit* option in menus shown in Figure 2.4 and Figure 2.21.

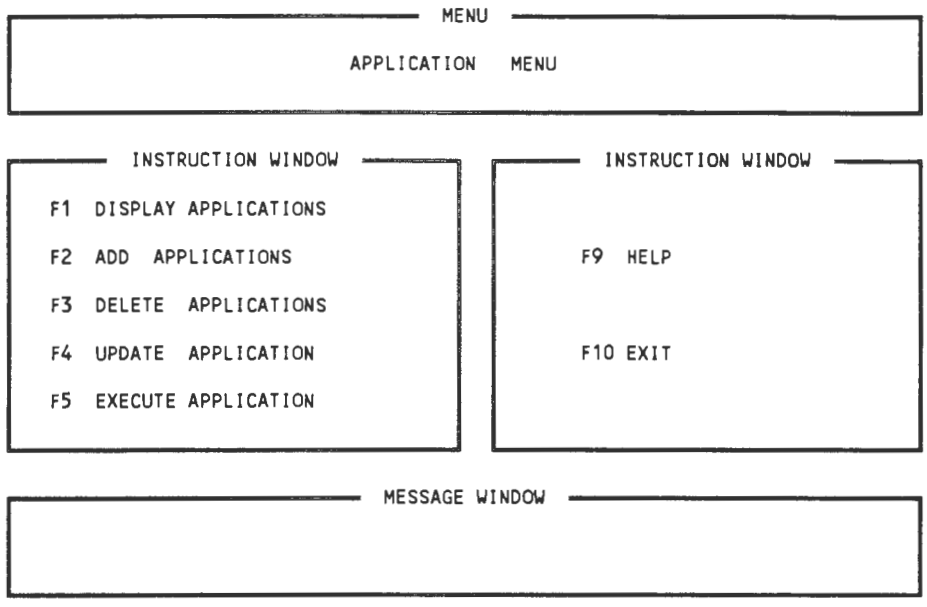


Figure 2.28: COSNET Application Menu

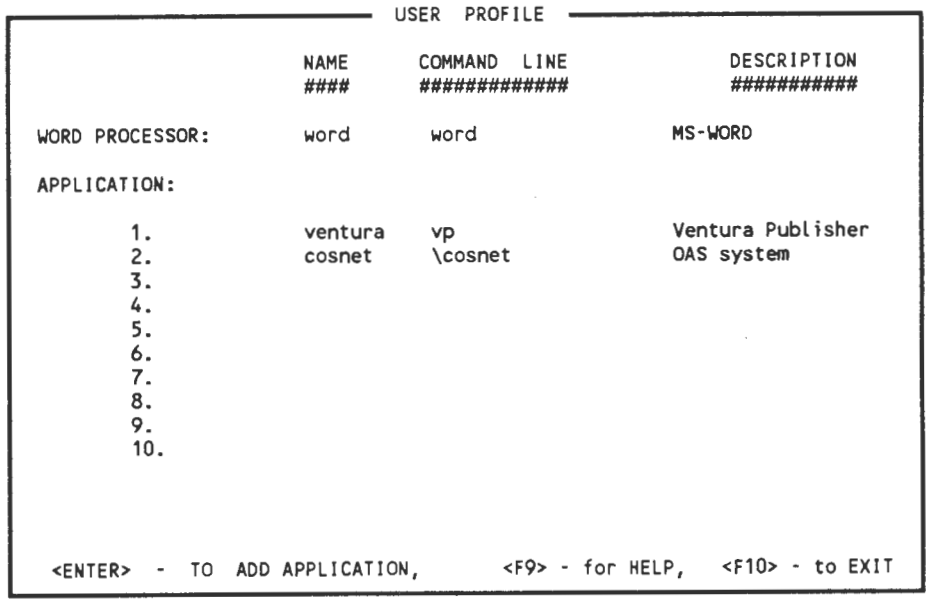


Figure 2.29: COSNET Add Application Screen

Chapter 3

COSNET MESSAGES

3.1 SYSTEM MESSAGES

This Section contains a summary of COSNET's system messages:

- **Another schedule for** *dd/mm/yy*? **(y/n)** :- You may add another schedule to the calendar schedules for the displayed day.
- **Checking UNIX connection** :- COSNET always check the status of the communication line to the server before logging you on.
- **Do you want acknowledgement? (y/n)** :- You may specify acknowledgment to a mail item.
- **Fetching cabinet information ....** :- The cabinet information files are being retrieved from the server.
- **Fetching calendar schedules from UNIX ....** :- The calendar files are being retrieved from the server.
- **Fetching mailbox from UNIX ....** :- The mailbox contents are being retrieved from the server.
- **Fetching mail-log from UNIX ....** :- The mail-log contents are being retrieved from the server.
- **Fetching system files from UNIX ....** :- The COSNET system files are being retrieved from the server.
- **File exists - Do you want to overwrite? (y/n)** :- This document already exist, and it may be destroyed.
- **Forwarding mail to** *user* :- The mail item that you are currently inspecting is being forwarded to *user*.

- **Inspecting mail on UNIX - Please wait ....** :- COSNET is inspecting your mail directory on the server to determine if any mail exists.
- **Logging onto UNIX - Please wait ....** :- You are being connected to the server and COSNET will execute the startup functions as soon as the connection is established.
- **New mail has arrived - inspect? (y/n)** :- You have received new mail since you last inspected your mailbox and may inspect the mail items.
- **No mail exists** :- You have no mail in your mail directories on the server.
- **Private schedule entry? (y/n)** :- To prohibit anyone else from inspecting this schedule, you may choose to classify it as *private*.
- **Restoring calendar on UNIX - Please wait ....** :- Your calendar files are being returned to the server.
- **Restoring mailfile on UNIX - Please wait ....** :- Your mail file is being returned to the server.
- **Restoring system files on UNIX ....** :- The COSNET system files are being returned to the server.
- **Sending acknowledgement - Please wait ....** :- An acknowledgement is automatically being sent whenever the acknowledgement bit on a mail item is set. After the acknowledgement is completed you may manipulate the mail item.
- **Send document to another user? (y/n)** :- You may distribute the document to other users.
- **Sending document to** *user* :- The document selected is being sent to *user*.
- **Sending mail to** *user* :- The mail item is being sent to *user*.
- **Send mail to another user? (y/n)** :- You may send the current message to another user.
- **Storing mail in mailbox** :- The mail item that you have inspected is being saved in your mailbox and will be deleted from the mailing list.
- **Urgent mail? (y/n)** :- You may classify the message to be *urgent*. The mail item is classified as not urgent by default.
- **Would you like to forward to another user? (y/n)** :- You may forward the current mail item to another user.
- **Would you like to inspect own calendar? (y/n)** :- You may choose to inspect your own calendar schedules or the schedules of another user.

- **You have new mail - re-inspect? (y/n)** :- While you were inspecting your mail, new mail has arrived. COSNET gives you the option to re-inspect your mail that includes the newly arrived message(s).

3.2 ERROR MESSAGES

. This Section describes COSNET's error messages:

- **Access to calendar denied!** :- You have no permission to inspect the calendar schedules.
- **Alias** *aliasname* **does not exist!** :- You are trying to delete or update an alias that does not exist.
- **ERROR -** *user* **already exists!** :- The username that you are trying to add to the calendar access file already exists.
- **ERROR - cannot find userfile!** :- The system file containing the COSNET user information cannot be found, or is corrupted.
- **ERROR -** *user* **does not exists in alias file!** :- You are trying to delete or update a user name that has no alias defined.
- **Groupfile** *filename* **does not exist!** :- The name of the groupfile you have chosen does not exist.
- **Illegal username!** :- The user name you have chosen does not exist.
- **Illegal date!** :- The date entered by you is incorrect.
- **No alias exist!** :- You have no alias defined.
- **No documents with author were found!** :- COSNET could not retrieve any documents created by the author name that you have chosen.
- **No documents with date were found!** :- COSNET could mnot retrieve any documents that were created on the date that you have selected.
- **No groupfiles exists!** :- You have currently no distribution lists.
- **No schedules for** *dd/mm/yy exist!* :- There are no schedules for the selected date in the calendar.
- **No such username or alias exists!** :- You are trying to delete a user or alias name that is not defined in your alias file.
- **Sorry - can't create printfile!** :- There is a problem with your printer; COSNET can't create the printfile.

- **Sorry - wrong password!** :- You have entered the wrong password. COS-NET does not allow you to proceed and you must restart the login procedure.
- **User** *username* **does not exist in groupfile!** :- You are trying to delete a user that does not exist in the distribution list.
- **User** *username* **has alias - change (y/n)** :- The user that you are trying to create an alias for, already has an alias. You may change this alias if you like.

Index