ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION STUDIES FOR AFRICA

DEVELOPING A USER FRIENDLY COURSEWARE:

A CASE STUDY WITH INVENTORY MANAGEMENT

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF

THE REQUIREMENT FOR THE DEGREE OF MASTER OF

SCIENCE IN INFORMATION SCIENCE

BY

ALEMAYEHU MOLLA

May 1995

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION STUDIES FOR AFRICA

DEVELOPING A USER FRIENDLY COURSEWARE:

A CASE STUDY WITH INVENTORY MANAGEMENT

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF

THE REQUIREMENT FOR THE DEGREE OF MASTER OF

SCIENCE IN INFORMATION SCIENCE

BY

ALEMAYEHU MOLLA

May 1995

ADDIS ABABA UNIVERSITY

School of Graduate Studies

Development of User-Friendly Courseware: A Case Study with Inventory Management

by

Alemayehu Molla Adankew

School of Information Studies for Africa (SISA)

Approved by Board of Examiners

	Signature
Ato Getachew Birru, Chairman, Examining Board	
Dr. Taye Tadesse, Avisor	
Ato Tesfaye Birru, Advisor	
Dr. R.Gibson, External Examiner	
Dr. G.G. Chowdhury, Internal Examiner	

ACKNOWLEDGEMENT

I would like to thank Dr. Taye Tadesse and Ato Tesfaye Biru who have been very encouraging and stimulating supervisors; Ato Ocubasellase Tekleab of AACC for assisting in organizing the Inventory Management course material used in this study, and Borland International for using some of their program utilities.

I am also indebted to Dr. G.G. Chowdhury and Prof. G Battacharyya for encouraging me to work on this topic; My families (Abebech, Etabezahu, Samrawit and Tihitna) for the care and endurance they have given me throughout my study; Sisay Fisseha for his help while writing the program; the Academic and support staffs of SISA and to all my friends.

ABSTRACT

These days, the use/implementation of computer in education is becoming increasingly attractive. There is now a general understanding among workers in this area that the application of computers in education can be categorized as **TUTOR (CAI), TOOL and TUTEE .** The **TUTOR** application (which is the subject of concern here) in particular, relates to the use of computers as educational delivery systems where as the **TOOL and TUTEE** applications refers to the use of computer for performing certain tasks and for programming respectively.

A typical CAI/**TUTOR** system consists of a **COURSEWARE** (a specially designed software for classroom instructional use), along with the hardware and system software required to run such software.

As with any other software, courseware development requires analysis of the requirements first, in particular, answer to such basic questions as **When to use a courseware ? how to design (develop) a user friendly courseware** and **what factors to consider in the design of such a courseware.**

This study attempts to address these questions by way of demonstrating the design and development of a user friendly courseware using the available resources at SISA. A literature survey of definitions, types and design considerations of a courseware along with the main reasons behind the use and current development trends is presented. Among the development models suggested for the use in developing a courseware, the **ISD** model is

described in great detail. The design and development of a microcomputer based prototype courseware: **I**nventory **M**anagement **C**ourseware (**IMC**), using the ISD model and turbo vision programming language on the basis of a course under offering at the Addis Ababa Commercial College-Addis Ababa-Ethiopia (work place of the worker) is described at length.

The result of the study indicates:

1. Use of ISD model together with other tools and techniques lead to the development of a user friendly courseware within a reasonable period of time.
2. The possibility/feasibility of using high level programming languages such as Turbo Vision to develop an acceptable quality courseware even in the absence of specialized authoring software.
3. The team approach of designing a courseware, as it facilitate representing of different expert opinions, exchanging of ideas and identification of problem areas that might be missed otherwise results in producing a better quality courseware than individual based approach.

TABLE OF CONTENTS

CHAPTER THREE

APPROACHES TO COURSEWARE DEVELOPMENT

CHAPTER FOUR

THE COURSEWARE DEVELOPMENT PROCESS

CHAPTER FIVE

IMPLEMENTATION

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 BACKGROUND

Today, the pervasive influence of computers is being felt in all areas of application not excepting education. Indeed, computers appear to be revolutionizing and improving every aspect of our lives, i.e., the stores in which we shop, the offices where we work, the cars we drive, the banks that handle our money, the games, schools, kitchen appliances, health centres, e.t.c. (Williams 1989; Stern 1982; Bruce 1992; Sanders 1985; and Ihara 1993).

Since the introduction of the low-cost microcomputers in the early 1980's, the use of computers has,in fact, gradually spread throughout the educational system (Augenstein 1993 and Longhorne et al. 1989). There are many different ways in which a computer can be used in education.

One application of computers in the area of education is the use of computer as educational delivery systems. That is, where computers are programmed to deliver educational content to individual students in a situation where there are computer facilities.

The general term used to denote such use of computers in education (especially for the programs which deliver the education) is **COURSEWARE** (Bunderson 1981, Watts 1981, Coburn et al. 1985, and Uffing and Radev in Tagg and Lovis 1988).

1.2 STATEMENT OF THE PROBLEM

Despite the computers tremendous capacity in storing, retrieving, analyzing, and communicating data and information, computers seem to not being adequately utilized in 'developing' countries (Boyad-Barreft 1990 and Rahel 1992). In the educational system, in particular, the potentiality of computers does not seem to be well recognized. In Ethiopia, for instance, some studies (Rahel 1992), indicate that wordprocessing is the single largest area where computers seems to be widely used.

The need to change this trend and make necessary steps in increasing the utilization of computers in all sectors in general and education in particular of these 'developing' countries with the objective of improving the service and efficiency of the sectors, has been (and still is) the major issue (Boyad-Barreft 1992, IDRC 1988, ITU 1983, NSTIDC 1991, and PADIS 1992, Rahel 1992).

The use of computers in education, especially for the purpose of educational delivery along side lectures and text books requires the acquisition of coursewares through purchase or in house development. Due to their specialized nature, limitation of resources, degree of customization required and other reasons, such softwares are difficult to procure through the purchase option for countries such as ours. Therefore, for the reasons outlined above, it is here assumed that, an attempt has to be made to acquire such softwares through the second option - that of inhouse development.

1.2 STATEMENT OF THE PROBLEM

Despite the computers tremendous capacity in storing, retrieving, analyzing, and communicating data and information, computers seem to not being adequately utilized in 'developing' countries (Boyad-Barreft 1990 and Rahel 1992). In the educational system, in particular, the potentiality of computers does not seem to be well recognized. In Ethiopia, for instance, some studies (Rahel 1992), indicate that wordprocessing is the single largest area where computers seems to be widely used.

The need to change this trend and make necessary steps in increasing the utilization of computers in all sectors in general and education in particular of these 'developing' countries with the objective of improving the service and efficiency of the sectors, has been (and still is) the major issue (Boyad-Barreft 1992, IDRC 1988, ITU 1983, NSTIDC 1991, and PADIS 1992, Rahel 1992).

The use of computers in education, especially for the purpose of educational delivery along side lectures and text books requires the acquisition of coursewares through purchase or in house development. Due to their specialized nature, limitation of resources, degree of customization required and other reasons, such softwares are difficult to procure through the purchase option for countries such as ours. Therefore, for the reasons outlined above, it is here assumed that, an attempt has to be made to acquire such softwares through the second option - that of inhouse development.

As a software, commonly used systems development methodologies could be used/applied in courseware design and development. As a specialized software, however, customized models of the systems methods are being widely recommended.

This study is, therefore, an attempt to demonstrate the development of a courseware by applying one of the suggested customized models. By so doing, the study, particularly attempts to seek explanations/solutions for the following questions

1. **What are the factors that have to be considered in designing a courseware?**
2. **What characteristic features should a courseware posses ? and**
3. **How does one go about that task of designing a courseware?**

The case problem selected for the demonstration is the Inventory Management course given at the Addis Ababa Commercial College (AACC).

1.3 JUSTIFICATION

As in the case elsewhere, higher learning institutions in Ethiopia, are charged with the responsibility of producing skilled manpower in the fields of economics, business, adminstration, technology, etc. In the process of fulfilling the above responsibility, however, these institutions are facing a number of problems. According to Azeb (1986) and Asmerom et al. (1989) the attrition rate from higher learning institutions is increasing from time to time. The quality of graduates from the institutions is not satisfactory resulting in low job performance. The reasons for such problems according to the workers are generally

categorized in to two: problems related to pre-college factors and problems related to the institutions' teaching methodology.

Azeb (1986) and Asmerom et al. (1989) have also tried to suggest solutions to improve the situation. They indicate that the teaching process could be improved or could become 'effective' if the following conditions are in place:

1. Following student centred teaching
2. Recognizing the nature of difference among students
3. Meeting of student needs by diagnosing learning difficulties
4. Active student interaction
5. flexibility
6. Questioning
7. pace,

Institutions of higher learning are also influenced by the change in the environment in which they are operating. This environment is undergoing a tremendous change due, mainly, to the introduction and application of new technologies such as microcomputers. This change, in turn, influences the type and quality of manpower demanded from these institutions (Longhorne et al. 1989 and Tagg and Lovis 1988, 559). It also poses a challenge in delivering and managing the education process (Tagg and Lovis 1988, 679-682).

Educational technologists, starting from the early 80's, have been proposing the utilization of computers in delivering the information needed to teach a student a particular course (subject) mainly to bring about 'effective' teaching. It is also said that, (Longhorne et al.

1989 and Tagg and Lovis 1988, 679-682), the utilization of computers in delivering the information enables to effectively respond to the challenge posed by the trend of computerization in the environment.

As a result, these days, it appears that computers are in use in nearly every school and in all higher learning institutions in developed countries (Coburn et al. 1985, and Bear 1984) and in some of the institutions in the developing countries (Boyad-Barreft 1990).

Currently, most of the higher learning institutions in Ethiopia are acquiring computers and incorporating at least basic computer application training in their curricula. There is also an increasing trend in using the computers as a tool for facilitating their activities including wordprocessing, imparting application software training, numerical and statistical analysis, and library house keeping (Rahel 1992).

In view of the potentials of computers and the problems faced by the institutions, such level of utilization, though encouraging, could not be taken as suffice. As evidenced elsewhere, these institutions could bring significant improvement in the performance of students and in the process of teaching if they implement the computers in delivering the information required to teach courses.

To take advantages of computers in general and courseware in particular, however, the situations in which they are likely to be beneficial have to be considered (Alessis and Trollip 1991, Blaise 1989, Longhorne et al. 1989, and Salisbury 1984). The situations where courseware are to be of assistance are where:

1. Extensive individual student practice is needed
2. Students future job involves the daily use of computers
3. Safety is a matter of concern (laboratories)
4. Students motivation is a matter of concern
5. The material is very hard to teach by other methods

The course chosen as a case area/problem for the current work - Inventory Management, (PSM 252) has been under offering at the **Addis Ababa Commercial College**[1] in the Purchasing and Supplies Management stream since 1982.

As an information intensive course, definitions, principles, rules, methods, examples, practices and interpretations related to management, statistics, material management, accounting are needed in order to help students understand the subject of inventory management.

The selection of the course as a case problem for this study is mainly based on the following reasons:

1. As a student of a similar course in college and as a tutor of this same course (i.e., Inventory Management) at AACC, the worker has adequate knowledge/exposure with the concepts, principles and models involved.

[1] The Addis Ababa Commercial College, established in 1943, is one of the oldest higher learning institutions in the country which has grown from a high-school to a full fledged college. The college currently offers a two year diploma programme in six disciplines: Secretarial Science and Office Management, Accounting, Banking and Finance, Purchasing and Supplies Management, Marketing Management and Personnel Management.

2. Experience in the conduct of the course exhibits that usually students face difficulties in comprehending the subject, applying the models and executing the exercise with the lecture and textbooks methods only
3. The details of the course do not change frequently

In addition, the following are also additional reasons for selecting inventory management as a case area.

1. Inventory management is one of the important areas where business organizations can make significant improvements in their cost structure.To this end, regardless of its size, type, objective and orientation every modern organization has to deal with inventories, therefore, it has universal applicability and
2. Some organizations have an automated inventory management system (or the trend is towards that) which can be best simulated through computer assisted instructions.

In view of the foregoing considerations, it is believed that Inventory Management course is an area suitable for demonstrating the purpose and intent of this study- design and development of a courseware.

What is more, the experience gained in developing this courseware could be used to develop other similar coursewares in the future there by reversibly the application of computers in education.

1.4 OBJECTIVES

The study has the general objective of **DEMONSTRATING THE DESIGN** and **DEVELOPMENT** of a courseware by applying one of the suggested models for this purpose using existing resources and facilities at The School of Information Studies for Africa (SISA)

Specific objectives are to:

- create an awareness among instructors about the potential of using computers as educational delivery systems
- describe courseware types and respective characteristics
- highlight current trends and developments in courseware design
- describe some of available courseware development methodologies
- demonstrate the development of a courseware using one of the suggested methodology and using the inventory management course under offering at the AACC as a case area
- give demonstrations of some of the features of the designed prototype courseware

1.5 SCOPE

As the main purpose of the study is to demonstrate the design and development of a courseware, the study, doesn't attempt to make evaluation and comparison of suggested methodologies except brief description with the aim of finding the best method.

CHAPTER TWO

REVIEW OF RELATED LITERATURE

2.1 COMPUTERS IN EDUCATION

The technological breakthrough in the silicon chip has contributed for the development and construction of smaller, handy and cheeper microprocessors and the introduction of microcomputers (Coburn et al.1985). This development of microcomputers, in turn, has dramatically changed the nature of business operation by revolutionizing the way of generating, storing, processing and transmitting information. In fact, it is not only the business environment that faces dramatic change by the implementation of microcomputers, but also almost all other activities of the society (Cheever et al. 1986).

In the area of eduction, in particular, although the potential use of computers has been well documented more than three decades earlier (cheever et al. 1986), the computer didn't seriously influence the practice of education until the introduction of microcomputers. Since 1980, however, computers are posing a gradual change in the educational systems. Educators are now taking the advantages of the power of computers to make significant improvements in the conduct and effectiveness of education.

According to Taylor, uses of computers in education can be generally accommodated in one of the three modes: **Tool, Tutor, Tutee** (Boyd-Barrett 1980, Cheever et al. 1986, Merrill et al. 1986)

The **TOOL** aspect related to those applications in which students use computers to accomplish a single purpose or a related group of purposes-the computer is used by students and teachers primarily for carrying out certain tasks. Examples of such application include, word-processing, numerical analysis, statistical analysis, information processing, laboratory simulation, graphics and music synthesis .

The **TUTEE** mode of application relates to the use of computers for programming and computer literacy. The computer is a **TUTEE** when the student directs it by writing programs which describes procedures the student wishes the computer to execute.

In the **TUTOR** mode of application, the computer is used in presenting the materials to be learned by the students, i.e, the computer present a material, the student responds, then the computer evaluates the response and determines the next action.

Among the three stated applications of computer in education , the **TUTOR** mode is a matter of concern in this study.

In the literature (Cheever et al. 1986, Dean and Whitlock 1983, Kearsely 1983, Merrill et al. 1986, and Offir et al. 1993) the terms **C**omputer **M**anaged **I**nstruction (**CMI**), **C**omputer **B**ased **T**raining (**CBT**), **C**omputer **B**ased **E**ducation (**CBE**), **C**omputer **A**ided **L**earning (**CAL**) **C**omputer **A**ided **T**eaching (**CAT**), **Computer Aided Instruction (CAI)** are used sometimes synonymously and sometimes with marginal difference to represent the application of computers as a tutor.

For reasons of currency (Coburn et al. 1985, Meera and Rahel 1994, Offir 1993 and Tagg and lovis 1988, 29-30, 559-563), however, the term **CAI** is given preference inhere to represent the application of computers as a tutor.

2.2 COURSEWARE: Defined

The growth of the computer industry and the pervasive application of computers over the past years results in the introduction of the convenient terms **hardware** and **software.** Like any other application of computer, a typical CAI basically consists of hardware and software.

But with the advent of CAI, a need comes for a distinction to be made between the computer software that supports the general functions of the hardware and the software that is tailored for providing instructional interactions. Such a need results in the coinage of the term **courseware** (Bunderson 1981, Watts 1981, Coburn et al. 1985, Uffing, Radev in Tagg and lovis 1988).

Bunderson (1981) reported that he have heard the term from Frank Haas in about 1967. He also reported that, through some process of natural selection, the term **courseware** emerged triumphant over such competing terms as **teachware and lessonware** and has now come in to fairly common use with little exception.

Brief review of published literature presents a number of, but related, definitions.

Bunderson (1981) defines courseware in the following way.

"... courseware typically refers to whatever combination of on line and off line materials which comprise a training course or program."

Watts (1981) considers coursewares as

"... those packages of software and ancillary materials specifically designed for classroom instructional use"

Coburn and his associates (1985) refers courseware as

"... educational software which is usually accompanied by a range of ancillary materials including teachers manuals; ditto sheets; work books and tests"

Software packages which help teachers to improve their teaching in the classroom is also referred by Jaques Hebentett as "courseware"(Tagg and Lovis 1988, 7)

In addition Uffing Shouten (Tagg and Lovis 1988, 517) refers courseware to represent the development of CAL as *"... the development of CAL also known as courseware "*. While Radev uses the term to refer to

"... the preparation of teaching programs or teaching courses,i.e., coursewares" (Tagg and Lovis 1988, 549)

Brief review of published literature presents a number of, but related, definitions.

Bunderson (1981) defines courseware in the following way.

"... courseware typically refers to whatever combination of on line and off line materials which comprise a training course or program."

Watts (1981) considers coursewares as

"... those packages of software and ancillary materials specifically designed for classroom instructional use"

Coburn and his associates (1985) refers courseware as

"... educational software which is usually accompanied by a range of ancillary materials including teachers manuals, ditto sheets, work books and tests"

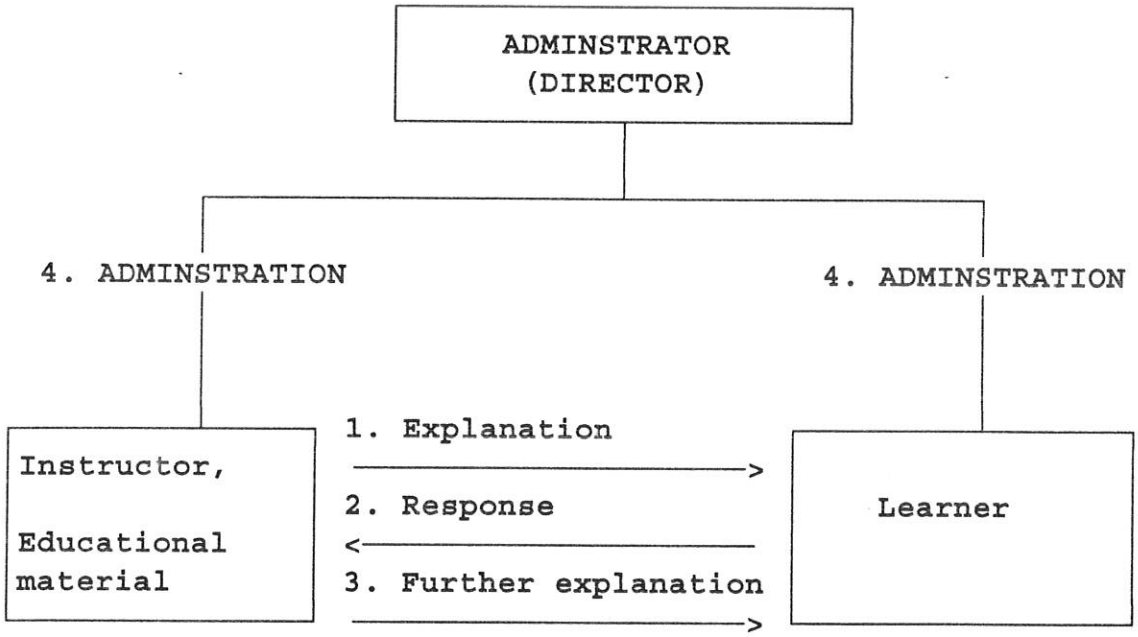
Software packages which help teachers to improve their teaching in the classroom is also referred by Jaques Hebentett as "courseware"(Tagg and Lovis 1988, 7)

In addition Uffing Shouten (Tagg and Lovis 1988, 517) refers courseware to represent the development of CAL as *"... the development of CAL also known as courseware "*. While Radev uses the term to refer to

"... the preparation of teaching programs or teaching courses,i.e., coursewares" (Tagg and Lovis 1988, 549)

As can be seen, from the various definitions presented above, although the wordings differ, one can clearly see that the concept emphasized is more or less the same.

Generally, assuming the following model as the representation of the basic concept of education,



one can state without loss of generality that CAI (courseware) is a system where a keyboard, CRT display, other computer controlled equipments (optional) and documntal references (optional) could be used for the purpose of performing (1), (2), and (3) from the above.

It is worth noting, however, that the use of a courseware can not replace a teacher. It is one form of education where the simulation, animation and other characteristics of computers are used in the process of instructing a student through a certain course material with supportive materials (online or offline) of how to use the program.

2.3 REASONS FOR USING COURSEWARE

Coursewares, inherently provides a model for an interactive and individualized learning. The learner must continually do something - answer a question, select a topic ask for a review and so on.

Generally almost all of the applications of courseware are driven by the need to improve the efficiency or effectiveness of teaching, which in turn means getting better instruction results (i.e. improved student achievement or job performance) with fewer resources.

Increasing efficiency / effectiveness benefits of a courseware are more expressed in terms of (kearsley 1983):

- increased control
- reduced resource requirement
- individualization
- improved student performance
- increased learning satisfaction, and
- convenience

Each of these benefits are briefly described as follows

2.3.1 Increased Control

Lack of control over instructional activities is one of the biggest failings in educational systems (Kearsley 1983). Institutions spend a considerable amount of time and money for

developing a course. Such course materials, however, may be modified by instructors which may produce different outcomes than were originally intended. This could lead to lack of standardization. In addition, except for certain check points, real controlling over the learning process to ensure if students have learned good troubleshooting or diagnostic skills is impossible.

Use of courseware is reported to have alleviated such problems (Keresley, 1983) and allows increased control in terms of improved utilization or completion of learning materials, increasing standardization of instruction, or monitoring of student progress.

2.3.2 Reduced Resources

The most significant resource reduction in teaching as a result of using courseware is the possibility of representing real world problems and events within the computers. In particular, use of computer based simulations (which are one type of coursewares) have displaced the need for actual equipment. In addition to providing the kind of practice- hands-on- equipment - needed without the cost (risk) associated in the actual equipment, with the advent of computer graphics technology, computer-based simulations are often providing the student with a better understanding of the equipment than is possible with the actual equipment. For example, David Singer and Uri Ganiel (Tagg and Lovis 1988, 55) have reported how use of a simulation reduced the resource required in teaching Optics and Romer's experiment.

Coursewares have also the potential to enable students to take care of the routine aspects of the instructions by themselves through online and offline self study and making the instructor needed only to help students with problems.

2.3.3 Idividualization

One of the central focus of coursewares is allowing each student to learn in his/her own pace and in a fashion most suited to his/her particular learning style (Offir et al. 1993 and Sarah 1992).

Use of courseware allows students not only to proceed at their own pace, but to skip over materials in which they already have competence, and also repeat lessons as necessary. This results in significant time savings over conventional classroom instruction which tend to be geared to the pace of the slowest learner.

Coursewares also allow students to have some control (sometimes free and sometimes conditional) over the order in which they learn topics, (i.e., the sequence of instruction). It is also reported that, the use of courseware enables each student to complete the training with the same acceptable level of competence (Kearsley 1983).

Von Solms, Bosschere Koenrad and Berg et al. (Tagg and Lovis 1988, 79 - 106, 243- 249) have made reports on their experience of designing and using coursewares **EDUBASE** (an educational file management system); **EDULAN** (a tool for software educators) and a courseware for teaching biology respectively. In their reports, they emphasized, among

other things, the benefits of courseware in individualizing instructions and providing the learner control over the learning process.

2.3.4 Improved Student Performance

One of the major reason for using a courseware is to improve student performance or reduce performance problems (Kearsley 1983). Courseware's capability to provide interactive and individualized instruction provide it with inherent potential to improve the quality of teaching. According to Kearsley (1983), Offir et al (1993) and Dean and Whitelock (1983), due to the one way communication in many inances, instructional approach based on conventional methods (i.e, classroom lectures) result in little actual learning as the student spends most of the time not attending to the instruction. In using courseware, however, the student spends a very high percentage of the time interacting with the computer and doing something that the courseware instructs him/her to do. Such continuous interaction and two way communication, according to Kearsley, Offir and Dean, results in actual learning and improved student performance.

For example, the benefit of improved student performance in using courseware for Mathematics, English and Biology have also been reported by Coburn (1985) and Tagg and Lovis(1988, 511- 521).

other things, the benefits of courseware in individualizing instructions and providing the learner control over the learning process.

2.3.4 Improved Student Performance

One of the major reason for using a courseware is to improve student performance or reduce performance problems (Kearsley 1983). Courseware's capability to provide interactive and individualized instruction provide it with inherent potential to improve the quality of teaching. According to Kearsley (1983), Offir et al (1993) and Dean and Whitelock (1983), due to the one way communication in many inances, instructional approach based on conventional methods (i.e, classroom lectures) result in little actual learning as the student spends most of the time not attending to the instruction. In using courseware, however, the student spends a very high percentage of the time interacting with the computer and doing something that the courseware instructs him/her to do. Such continuous interaction and two way communication, according to Kearsley, Offir and Dean, results in actual learning and improved student performance.

For example, the benefit of improved student performance in using courseware for Mathematics, English and Biology have also been reported by Coburn (1985) and Tagg and Lovis(1988, 511- 521).

2.3.5 Increased Learner Satisfaction

The interactive nature of courseware has a common outcome of improved learner satisfaction and enjoyment. Students typically find using courseware more motivating than other forms of instruction (Kearsley 1983 and Offir et al. 1993).

The feedback provided on responses and the capability of assessing the student progress available in most of the coursewares creates an environment where by students develop a sense of achievement while learning. This leads to increased satisfaction during training which is one of the most important factors that contributes to students achievement and satisfactory completion of the course (Barker and Manji 1992). Poor motivation rather than lack of ability or skill often causes learning problems. Thus, the capability of coursewares to improve motivation is an important factor in reducing course failures.

The experiance of Xavier and Sereno (Tagg and Lovis 1988) in developing and using coursewares for accounting and biology inteachings respectively indicates that coursewares have greater advantages in improving student performance.

2.3.6 Convenience

Considering some of the reasons discussed above, convenience may appear to be a relatively weak rationale for use of coursewares. However, as computer systems become more ubiquitous and widespread, it is becoming a very strong and compelling argument for courseware.

In so far as students have access to a computer system when they go to work, which is most likely to be the case, and to a courseware package, use of courseware can make a good deal of sense. In addition, the courseware can always be used for refreshing previous knowledge.

While these are some of the main arguments generally cited to encourage the use of courseware, increased development time, Problem of facilitating man machine communication, Problem of cost efficiency and that of user acceptance are, however, the main reasons cited to discourage its use.

2.4 TYPES OF COURSEWARE

Coursewares may be considered along a spectrum. At the one end is receptive coursewares which simply provide information to be received by the student. Examples are drills and one way tutorials. At the other end of the spectrum are coursewares which students and teachers operate using different methods and applications in the different subject areas according to their own interest, inclination, and ability.

Generally five types of coursewares are identified: drill and practice, tutorial, demonstration, simulation and instructional games (Cheever et al. 1986, Coburn et al. 1985, Merrill et al. 1986, Kearsley 1983, and Watts 1981). Brief discussion of each follows.

2.4.1 Drill and Practice

Drill and practice is the most common, best known and most "disparaged" type of courseware. It is the simplest way of using a computer for instruction. It consists of presenting a question or problem, accepting a student response, and branching to another question or problem based on the accuracy of the answer or the mastery criterion for the instruction.

The critics of this type of courseware, (Coburn et al. 1985), include pedagogical narrowness (lack of stimulus - response) and the possibility of reinforcing incorrect learning. But advocates of drill and practice type of courseware (Coburn et al. 1985 and Merrill et al. 1986) acknowledge that such criticism is due to poor design of the programs by the persons (educators) involved in the development rather than the inherent nature of drill and practice. They suggest that with the additional features of graphics, sound and animation drill and practice can provide useful educational experiences.

It has been reported that (Merrill et al. 1986) Drill and practice types of courseware are generally appropriate if the expected learning outcome is intellectual skill, (especially discrimination).

2.4.2 Tutorial

Unlike the drill and practice, tutorials coursewares involve the presentation of information. Such coursewares typically consist of discussions of concepts or procedures with

interspersed questions or a quiz at the end of the discussions. The primary purpose of tutorial applications is to teach new information. In this regard, tutorial coursewares, by taking the full instructional burden, guides a student to the achievement of a specified set of objectives.

Like the drill and practice, the critics of the tutorial coursewares point out its "limited and limiting pedagogy" (Coburn, 1985). Tutorials are also criticized by their nature of trivializing concepts, i.e., by narrowing the range of possible responses, tutorial coursewares keep students from exploring the complexities of a given concept.

Advocates of tutorial courseware, however, respond to this criticism by pointing out that the same criticism can be applied in the conventional instructional methods such as the lecture, textbooks etc (Merrill et al. 1986). With the advent of intelligent tutorials which make the representation of knowledge very possible and an easy task, such criticism is loosing its significance.

Tutorials, as reported (Merrill et al. 1986), are mostly applicable if the expected learning outcome is intellectual skill especially concept and motor skill.

2.4.3 Demonstration

Demonstrations says Coburn (1985), are one of the main features of a traditional teaching repertoire in such subjects as science and mathematics. Demonstration coursewares can raise demonstrations to undreamt highest of sophistication and at the same time make them fail

proof and easy enough to use. Among the factors that contributed a lot to the rapid development and application of such courseware packages are graphics, sound and colour features of a computer.

Computer demonstrations have a much richer potential to be interactive than do chalk-board or overhead projector demonstration (Merrill et al. 1986). Variables are more easily manipulated. Effects are instantaneous. The graphics are clearer than chalk marks and the working surface cleaner than a board.

The design of elaborate graphics with high degree of interaction, however, requires a sophisticated and time consuming programming. This makes development of a demonstration courseware usually a difficult task (Coburn et al. 1985). Demonstration coursewares are said to be most appropriate to aid the instructors in giving instruction and the student in reviewing the material.

2.4.4 Simulations

Simulations are based on a model of some process, mechanism or activity. Such model, allows a student to relate input or changing parameters to output or outcomes; initiates a real or imaginary system based on the model's theory of the operation of the system. Since reality can not be fully represented by any model, simulations focus student's attention on certain aspects of the process under investigation. Simulations, however, are also powerful learning tools in studying events that could not be otherwise examined owing to danger, expense, or lack of time (Coburn 1985).

Although simulation have such a profound use, the design and programming of good computer simulation is very difficult. The main critics in this line is the inherent difficulty that exist in representing real world problems with the model (Coburn 1985).

Simulations are best suited in an environment where mastery of skills, learning of content, concept development, promoting enquiry are the educational goals. Or more generally where the learning outcomes expected are intellectual skills (especially rule sorting), cognitive strategies and motor skill.

2.4.5 Instructional Games

Instructional games involve an element of fantasy which is not possible in the other types of coursewares (Coburn et al. 1985). They are designed to be fun for students and thereby increase the chances of the student learning the concept, knowledge or skill embedded in the game. Game coursewares are most helpful and effective in a situation where student motivation is a concern (Merrill et al. 1986).

The most successful instructional computer games have challenge, fantasy and curiosity characteristics.To provide such features, the development of instructional game coursewares require sophistication in both curriculum design and computer programming.

2.5 COURSEWARE DESIGN CHARACTERISTICS

Due to the appealing advantages of coursewares, as described in the previous section, there are many individual and group endeavour in developing courseware all over the world. As a result, it seems that, almost every day a new catalogue of courseware is published.

In order to evaluate and choose among these coursewares and/or develop a courseware to pass the evaluation, identification of the basic features that should be possessed by a courseware is essential.

Coburn and his associates (1985) and The Open University (Boyd-Barrett 1990) provide basic characteristic features that should be present in a particular courseware.Generally, these different characteristic features of a courseware could be broadly categorized into four: factors related to **program content, pedagogy, program operation, and program documentation**.

The **PROGRAM CONTENT** characteristic feature of a courseware include:

- presentation of information about the content and back ground;
- suitability of the content of the material for students' use;
- compatibility of the content with the learning outcome(s) expected;
- educational significance of the content;
- statement of objectives and aims;
- statement of the intended type of use and audience; and
- availability of instructions for running the program.

The **PEDAGOGY (FRIENDLINESS AND FLEXIBILITY OF THE PROGRAM)** characteristics are:

- provision of immediate feedback which may be passive, active or interactive;
- clear definition of the learning theory underlying the courseware, i.e., behavioral, cognitive, logical, e.t.c ;
- self continence of the program, i.e., allowing students to learn by their own without intervention when needed;
- effective screen presentation,i.e, usage of colour, sound, graphics, and animation to enhance student learning with out filling the program with unnecessary sound and fury; and
- motivation (motivating) of students in using the program.

The **PROGRAM OPERATION** characteristics include

- clarity of the program from bugs and breaks;
- clarity and acceptance of the directions and instructions;
- synchronized usage of graphics, sound, and colour capabilities;
- informative and clear user error handling system;
- versatility,i.e., allowing the student control (conditional or unconditional) over the program operation; and
- portability, that's ability to transfer the program to a different computer.

The **DOCUMENTATION** characteristics of a courseware are

- information about machine requirements;
- information about the model used;
- information about the program structure;
- listing of the program code; and
- user documentation to use the package.

Generally if one expects to gain the benefits of using a courseware, the courseware has to satisfy all or most of the above characteristic features.

2.6 CURRENT TRENDS AND DEVELOPMENTS

Owing to the evolution of technologies, courseware development has made appreciable progress in recent years. In these regard the application of technologies like object oriented programming, hypertext and artificial intelligence efficiently contribute to the design and pedagogical maintenance of interactive and intelligent coursewares. A brief description of how this three technologies contribute to the design of courseware is presented as follows.

2.6.1 Object Oriented Programming

One of the promising approach to the courseware development challenges is object oriented programming. This technology for software development enables the programmer to arrange groups of code (objects) to build an application rather than building each application from scratch.

Systems built around object orientation methodology are built as collections of classes where by every class represents a particular abstract data type implementation or group of implementations.

Generally, object orientation can be characterized through the most fundamental aspect of its paradigm: **abstract data typing (encapsulation), inheritance and object identity** (Parodi and Ponta 1993 and Parsaye et al. 1989)

The first concept, abstract data typing, refers to the attributes and behaviour of an object. It enables to keep critical information together, hides details, minimize dependency and allows new objects to be easily added. The concept of inheritance refers to the nature of all the objects responding consistently to the same set of commands that are inherited from the original objects. The third concept ,i.e, identity allows easier maintenance by packaging different objects with the same commands which enables to respond to a consistent message.

Object oriented programming, generally, allows a more direct representation of the real world model in the code. As a result, the normal radical transformation from systems requirements (defined in user's term) to system specification (defined in computer term) is greatly reduced. It provides better concepts and tools to model and represent the real world as closely as possible.

Application of object orientation is increasingly becoming popular in software engineering. In Particular object oriented programming is found to be suitable for meeting such software engineering goals as **modifiability, maintainability, understandability, portability,**

abstraction, information hiding, reusability etc (Parsaye et al. 1989, Scholtz et al. 1993 and Arano et al. 1993).

The main features of the object oriented model mentioned earlier, i.e, encapsulation, identity and inheritance provide the facility to design software which could be easily modified maintained, understood and portable without any effect on the rest of the system.

Courseware development (engineering) basically being an activity of software engineering, which is differentiated from the later only by its educational content, draws all the benefits of using object orientation for software engineering. Especially, Parodi and Ponta (1993) has reported that, the application of object orientation methodology in courseware development enables an incremental development and software reuse. It also efficiently contributes to the design and pedagogic maintenance of interactive coursewares.

The implementation of object oriented languages enables the integration of all the courseware features into a homogeneous parts where all the different features of a lesson are built as "objects" and as such can be efficiently manipulated and combined. In fact, instead of building each application from the beginning, it enables the courseware programmer to accomplish this task by the arrangement objects.

Generally Cox and Hunt point out that "**... what is truly revolutionary about object oriented programming is that it helps programmers reuse existing code, just as Silicon Chip help circuit builders reuse the work of chip designers**" (Parsaye et al. 1989)

2.6.2 Hypermedia

One of the main obstacles that holds back the production and diffusion of educational courseware according to Ponta and Parodi(1993) is the excessive amount of time necessary to develop a quality courseware. The hypermedia approach provide a solution to both problem of quality and development time.

Hypertext can simply be defined as the creation and representation of links between discrete pieces of data. When these data are graphics or sound as well as text or numbers, the resulting structure is referred as **hypermedia** (Parsaye et al. 1989). **A Hypermedia** environment is the user interface that allows to work with different elements as a whole: text, graphics, images, sounds, simulation software, external hardware, databases, computer networks, etc., on the basis of specific application.

Conceptually the notion of hypertext is closely related to the idea of semantic networks with text and data being represented in nodes. Hypertext incorporates the notion of linking pieces of information, allowing users to navigate through a network of chunks of information. It offers a very powerful way of organizing and accessing information and hence has a potential role to play in the development of coursewares where knowledge representation is a primary concern (Parodi and Ponta 1993). With hypertext, users can smoothly move from one document to another along a particular chain or path of ideas (Mcknight 1991). Hypertext corresponds to the associative way in which people think. It is a flexible knowledge structuring principle that can be used to implement and embed other knowledge structures.

2.6.2 Hypermedia

One of the main obstacles that holds back the production and diffusion of educational courseware according to Ponta and Parodi(1993) is the excessive amount of time necessary to develop a quality courseware. The hypermedia approach provide a solution to both problem of quality and development time.

Hypertext can simply be defined as the creation and representation of links between discrete pieces of data. When these data are graphics or sound as well as text or numbers, the resulting structure is referred as **hypermedia** (Parsaye et al. 1989). **A Hypermedia** environment is the user interface that allows to work with different elements as a whole: text, graphics, images, sounds, simulation software, external hardware, databases, computer networks, etc., on the basis of specific application.

Conceptually the notion of hypertext is closely related to the idea of semantic networks with text and data being represented in nodes. Hypertext incorporates the notion of linking pieces of information, allowing users to navigate through a network of chunks of information. It offers a very powerful way of organizing and accessing information and hence has a potential role to play in the development of coursewares where knowledge representation is a primary concern (Parodi and Ponta 1993). With hypertext, users can smoothly move from one document to another along a particular chain or path of ideas (Mcknight 1991). Hypertext corresponds to the associative way in which people think. It is a flexible knowledge structuring principle that can be used to implement and embed other knowledge structures.

Any pieces of information whether it be text, graphics sound or numerical data can be linked to any other pieces of information. This flexibility makes it possible to construct a qualitative courseware because one is able to represent knowledge; browse, carry out structured searches and make inferences all within the same environment (Parodi 1993).

Hypertext also enhances cooperative writing among multiple authors. It allows one not to commit him/her self to a particular linear sequence of ideas when writing. Ideas can grow and evolve naturally over time while a consensus built on how they should be linked. It enables to avoid most disputes over material organization and primacy by allowing variety of orders to be constructed and tested.

Generally, the realization of multimedia application coupled with usage of an object oriented environment avoids the need for writing a long list of program statements. Instead it is possible to build applications by creating and/or modifying the objects that embody the different parts and functions for those application. The hypermedia environment provides a facility for developing a courseware which can facilitate individualization and interaction (Ahmed-Ouamer 1993).

2.6.3 Expert Systems

As Dunn and Morgan (1987) state, the subject of Artificial Intelligence (**AI**) provides the most promising source of new ideas for courseware. The particular branch of Artificial Intelligence which is currently making a significant impact in education relates to **Expert Systems** or **Intelligent Knowledge Base Systems**.

Alberico and Micco (1990) define an expert system as

" **computer programs along with knowledge, information and databases which act together to simulate the problem solving and decision making processes of human expert within a relatively narrow domain**"

Expert systems provide the tools for knowledge representation which are referred to as knowledge base, inferencing and higher level programming capabilities. Expert system environment also includes a number of tools for helping the various peoples who build or use the expert system.

The knowledge base is the part of the expert system that carries out long term memory of facts, structures and rules that represent expert knowledge about the domain of experts. Where as the inference engine is the part of the expert system that carries out the reasoning function.

In courseware development it is not the expertise of an expert system that is used to make improvement but the usage of expert systems as a higher level programming tools. In particular it is the approach to expert system where they are seen primarily as high-level programming environment for information and knowledge-intensive tasks which dominates the advantage of expert systems in courseware design and development. That is, expert systems technology are used as a general programming environment for putting intelligence in to coursewares.

Such inclusion and representation of knowledge using the expert systems results in the development of intelligent coursewares which are highly interactive and have the capability of modelling the students knowledge in order to discover the reason(s) of his/her mistakes. As such expert system application increases the adoption of the courseware to a given student (Boufaida and Barril 1993). It also facilitate development of courseware packages which can respond flexibly to individual learner needs and progress (Ford 1991).

CHAPTER THREE

APPROACHES TO COURSEWARE DEVELOPMENT

The benefit from the use of courseware depends on the availability of a high-quality courseware (Barker and Manji 1992). This in turn requires the initiation, control and coordination of a range of development processes which contributes to the production of operational courseware. The search for a development model which would increase the likelihood of quality courseware, therefore, is one of the important issues in courseware design.

3.1 DEVELOPMENT MODEL

There are various courseware development models which are in use in developing courseware. A method suggested by Pistorius et al. (1992) consists of five phases.

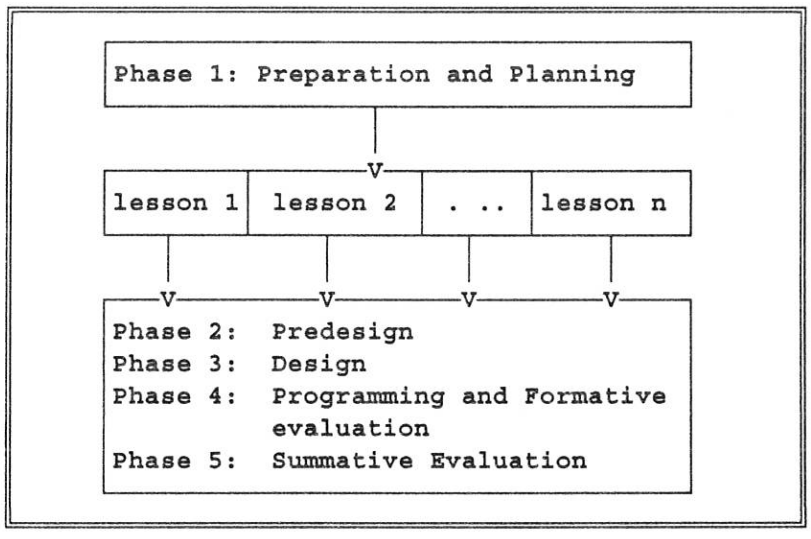


Fig. 1: COURSEWARE DESIGN MODEL

Each of the five phases is made of the following detailed activities.

Defining a course where CAI could be used

Identification of the target population

Drawing up a project plan

Appointment of a courseware development team

Setting up instructional and design strategies

Choosing authoring software

Fig. 2: Phase 1: Preparation and Planning

Identify topics for CAI

Identify support material

Discuss ideas with design team

Fig. 3: Phase 2: Predesign

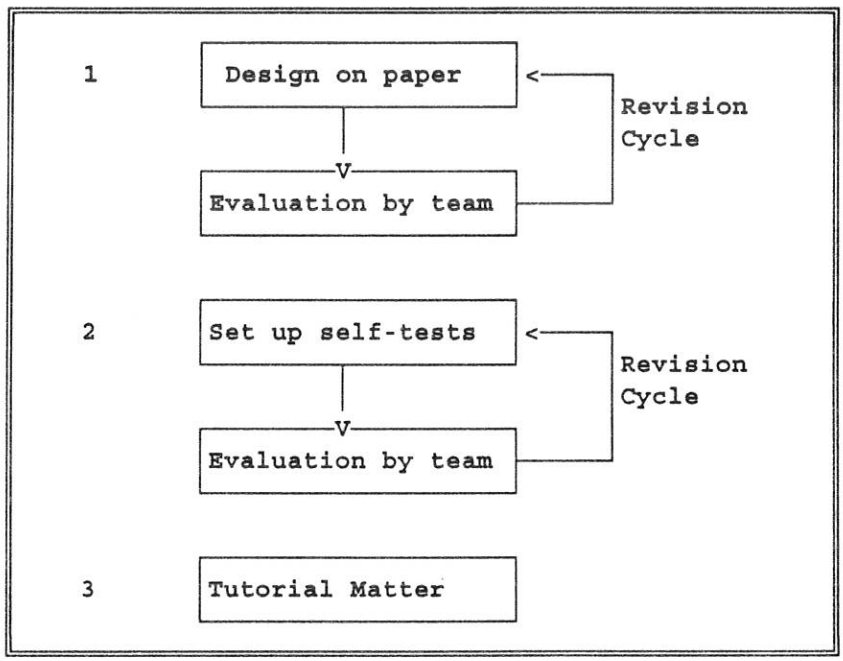


Fig. 4: Phase 3: Design

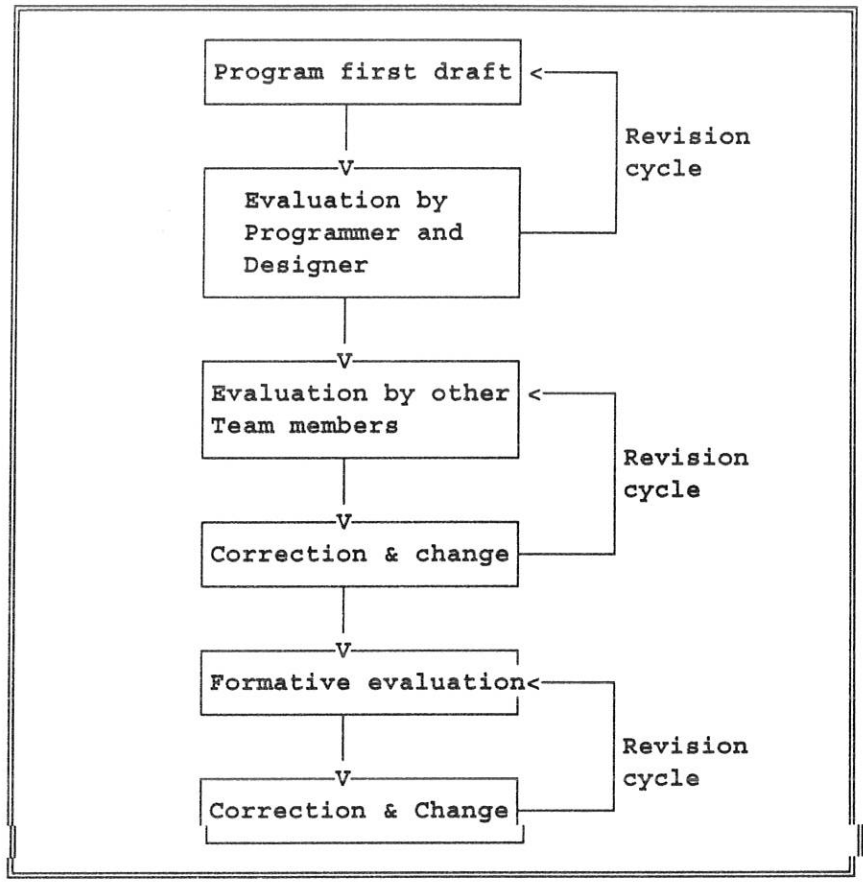


Fig. 5: Phase 4: Programming and formative evaluation

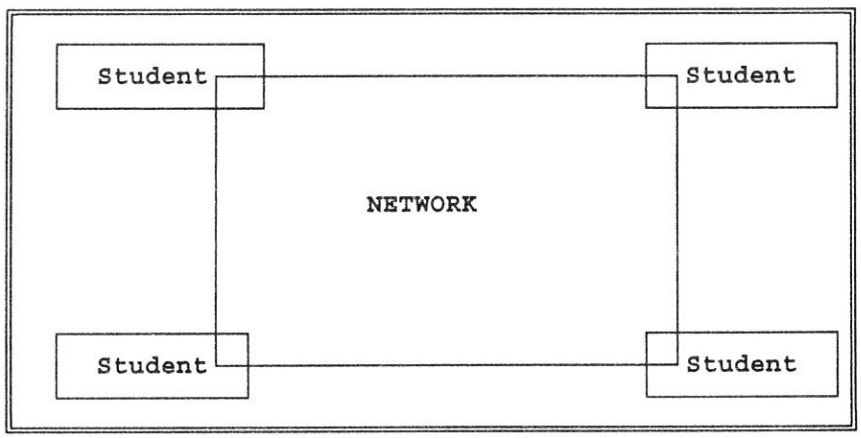


Fig. 6: Phase 5: Summative evaluation by means of CMI

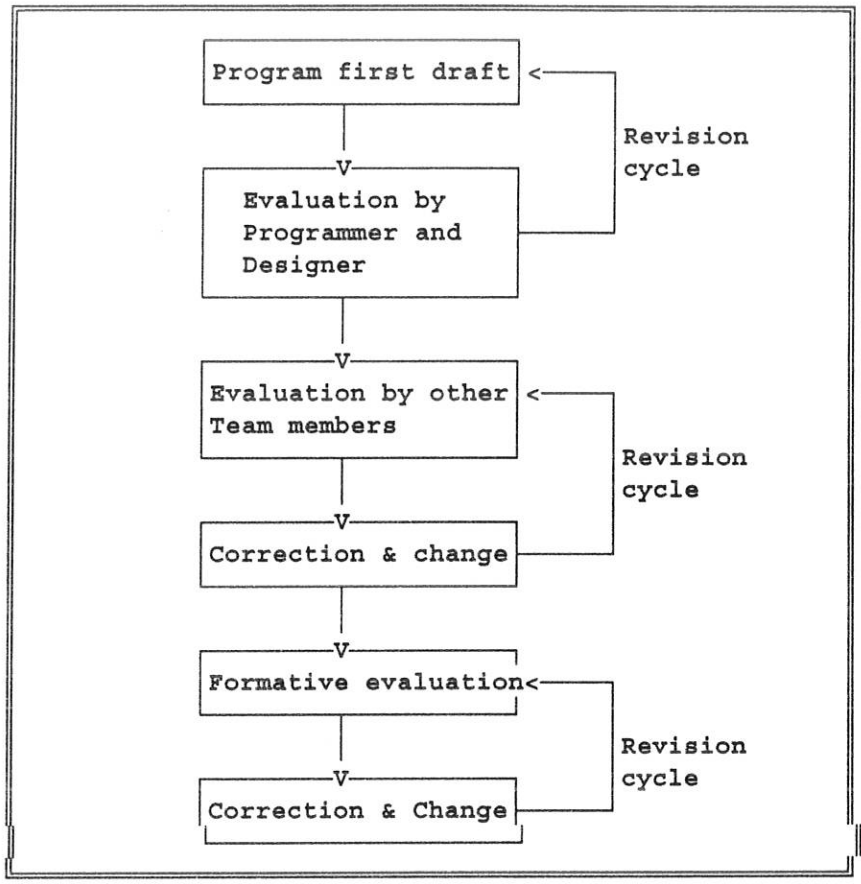


Fig. 5: Phase 4: Programming and formative evaluation

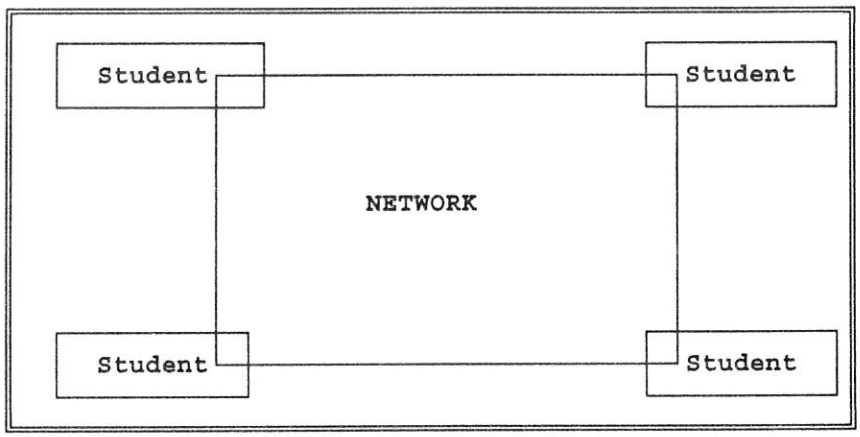


Fig. 6: Phase 5: Summative evaluation by means of CMI

According to the model advocated by Barker and Manji (1992), a typical courseware engineering exercise include: NEED ANALYSIS; REQUIREMENT SPECIFICATION; INSTRUCTIONAL DESIGN; PROTOTYPING COURSEWARE; IMPLEMENTATION; EVALUATION AND TESTING; BULK PRODUCTION; DISTRIBUTION AND MARKETING.

Peter Chandra (Tagg and Lovis 1988,) suggests the design team approach to the development of computer based learning (CBL) courseware design. This model (fig. 7) depicts the different expertise provided by team members for the various design stages. The design team approach relies heavily on the effective communication between different members of the team including up-to-date paper work and documentation.

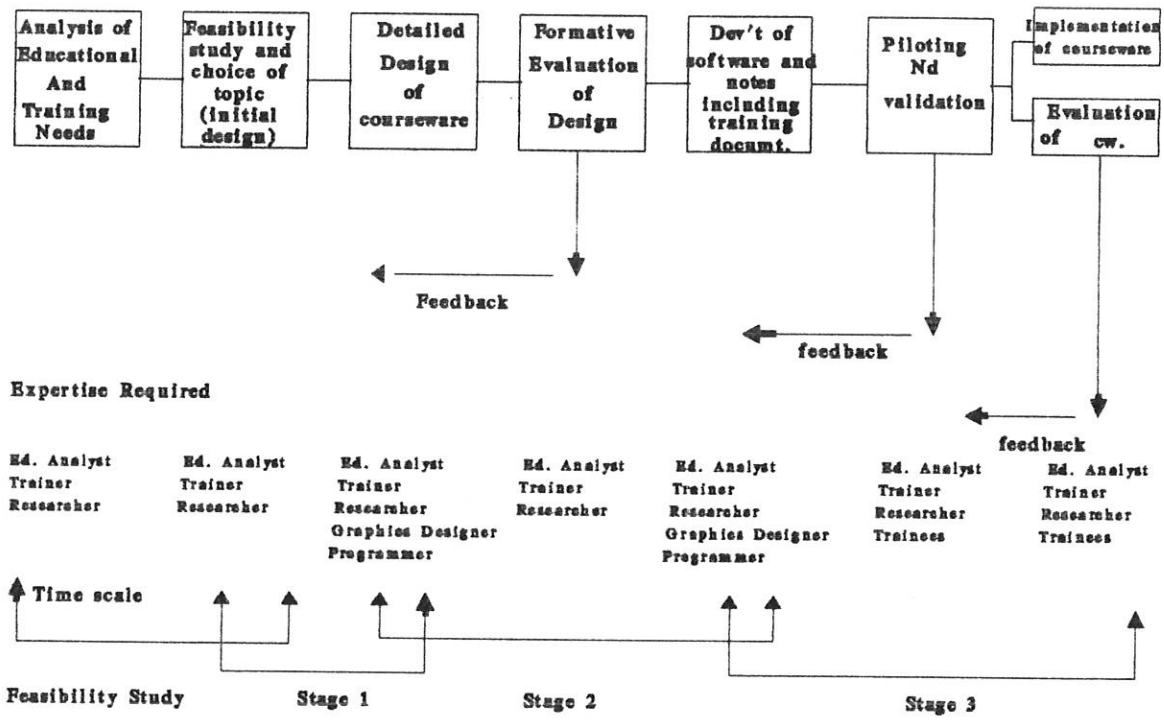


Fig. 7. The team approach to CBL courseware design

Another more sophisticated model suggested by Ahmed-Oumer (1993) is the Computer Assisted Courseware Engineering (CACE) for industrial application called AGEDI (Atelier de GEnie DIdacticiel).

This Model (described in Fig. 8) takes advantage of advances in information technology, on computer based learning and instruction. The main components of AGEDI are the Intelligent Computer Assisted (ICAI) module (application Layer), the hypermedia system (presentation layer) and the interaction supervisor (control) which chooses the best interaction mode in a learning session: free, guided or mixed one.

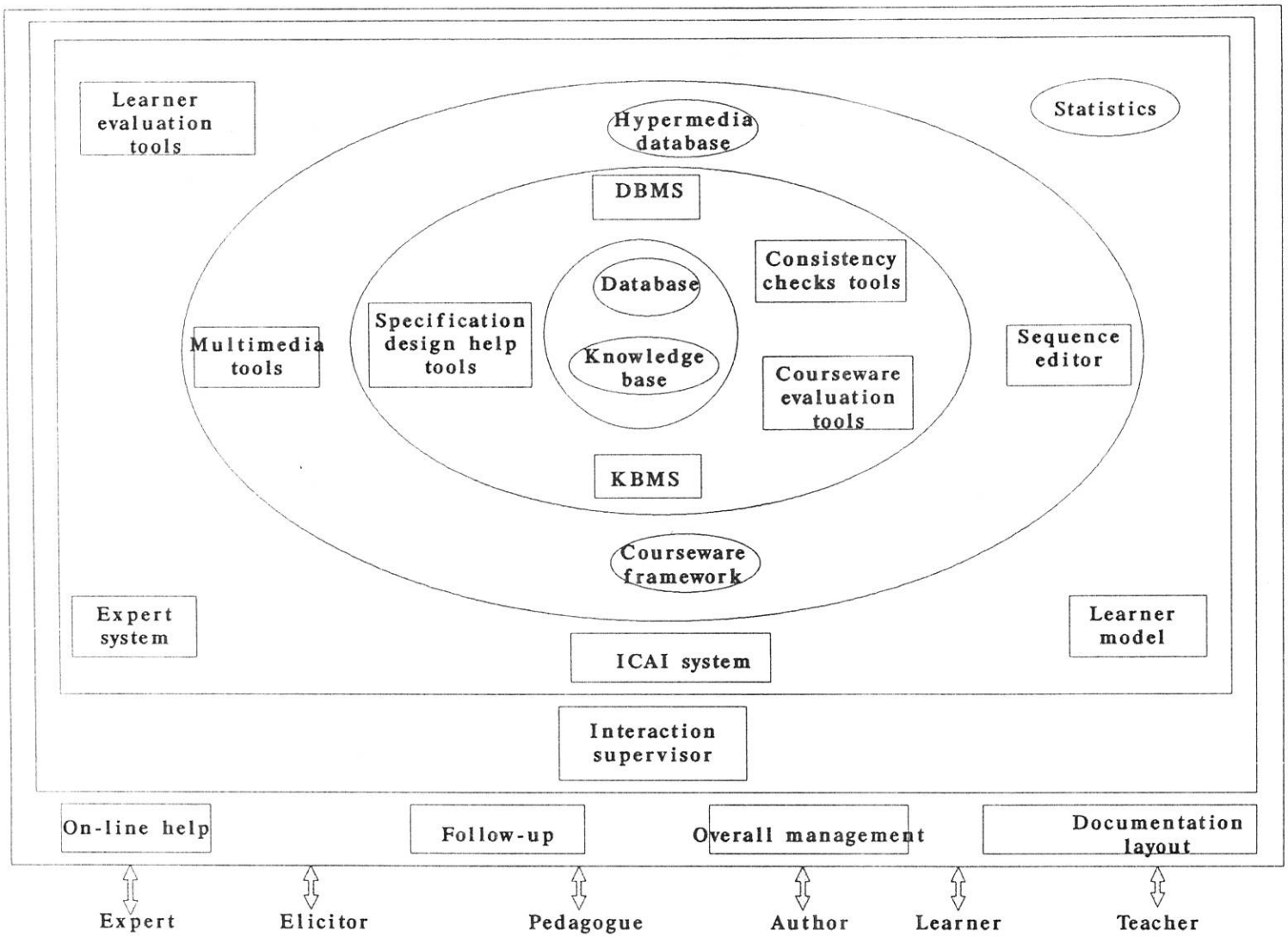


Fig. 8. Overall architecture of the CACE

Another useful model recommended by many workers in the field is the Instructional System Design (**ISD**). Various studies have shown that, for effective courseware design, the systems approach is the best approach.

The general term that refers to the systematic approach to the design and development of instructional materials regardless of the medium is Instructional Systems Design (O'Neil 1981 and Rosenberg, 1982). This approach gets universal acceptance (Gagne and Briggs 1979, Gagne, Wager and Rojas 1981, Rosenberg 1982, O'Neil 1981) because its application results in designing relevant, complete and interactive courseware. It also provides a procedure for systematically identifying and manipulating significant components which make up the instructional process, the goals of which are increased learning and improved performance

.

Some of the weakness as reported in the literatures about the different models discussed above, for example, include:

1. he approach suggested by Peter Chandra is reported to has been hampered by the lack of authors who have the technical appreciation of how their ideas and teaching strategies can be implemented on a computer (Tagg and Lovis 1988).
2. AGEDI practicality is reported to have been limited due to the unavailability of the sophisticated technological facilities/resources required by the model in many of courseware development projects (Ahmed- Oumer (1993)

For these and other reasons such as the workers exposure/familiarity with the model, the ISD approach, although is criticized for specifying only what is to be done and not how to do it (O'Niel 1981), is selected for use in the current work. To this end the remainder of this section attempts to describe this model in detail.

3.2 THE ISD MODEL

As indicated in the preceding section, ISD refers to the systematic approach to the design and development of instructional materials. Although there are many variations on ISD methodology, most are based on five phase model ,i.e., ANALYSIS, DESIGN, DEVELOPMENT, IMPLEMENTATION, AND EVALUATION as indicated in the following figure 9.

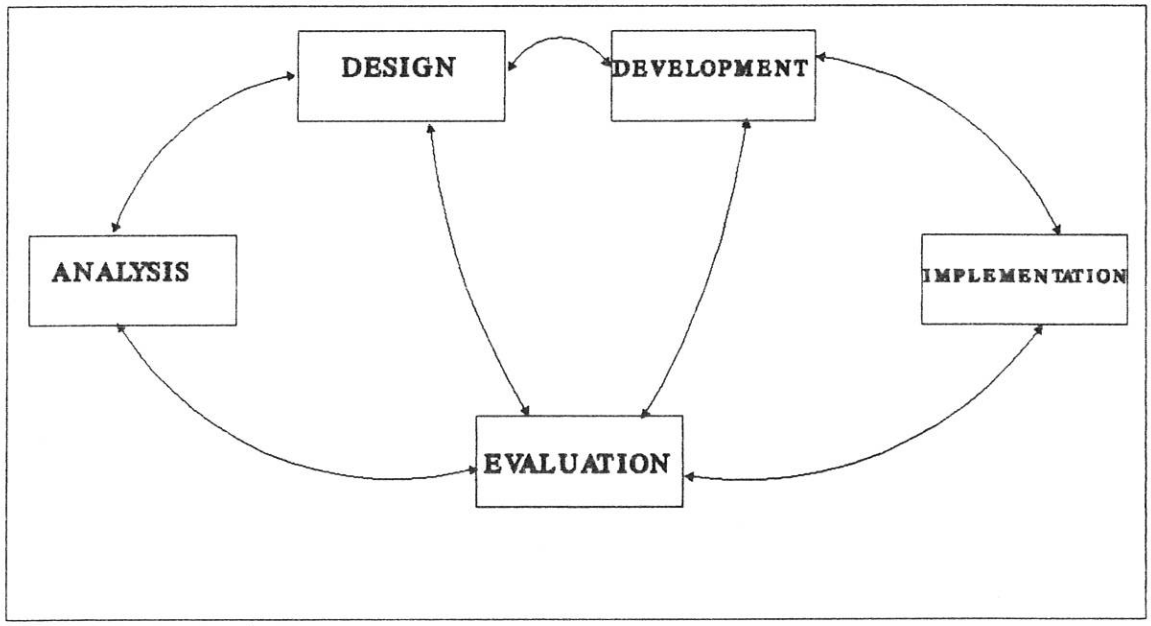


Fig. 9: Major Phases of ISD

3.2.1 Analysis Phase

This phase involves **analyzing learner** and **course needs.** In analyzing the learner needs the main issues are identification of difficulties of students in the class room and the specific knowledge and skills that they must have prior to beginning instructions. It also involves identification of the general characteristics of the student such as intellectual and academic background, previous experience and training, current technological knowledge and abilities.

The course need analysis is to determine the precise content of the course for which courseware is contemplated. This process may result in the identification of concepts, rules and information which must be included in the course. The basic information for this is obtained from the analysis of the tasks/jobs students are expected to perform upon completion of the course.

Composition of the courseware development team and selection of authoring software to be used in programming the system are matters for consideration at this phase. Usually the courseware development team is composed of: project manager, the subject matter expert, instructional design expert and programmers. Fig. 10 illustrates the components/activities of the analysis phase.

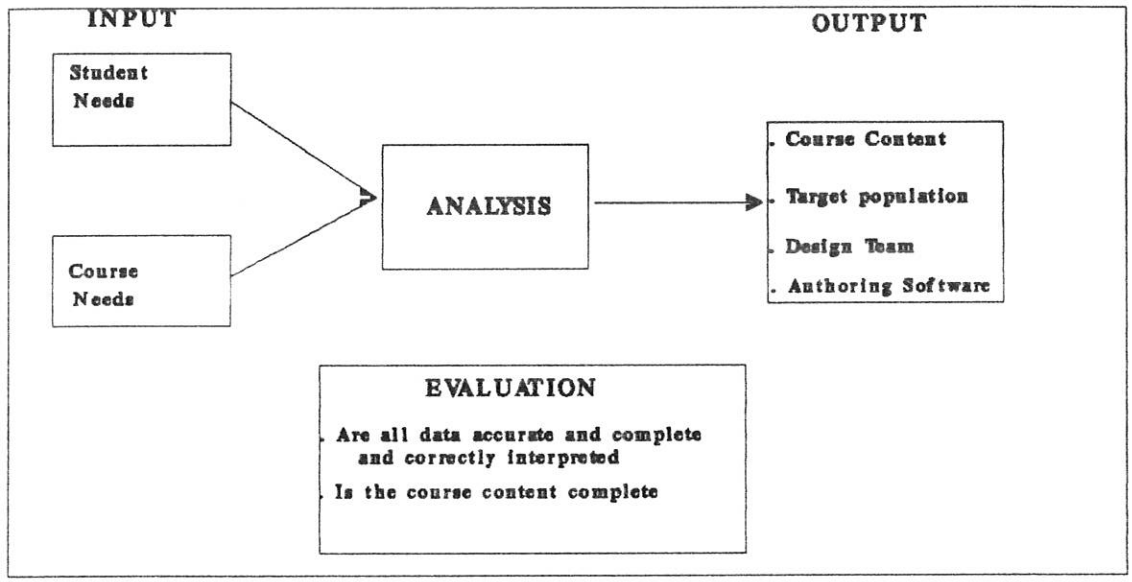


Fig.10: Analysis Phase

3.2.2 Design Phase

Specifying the different design considerations to be implemented in developing a courseware is the major activity to be carried out in this phase of the ISD model. These design considerations include

a. modular break down of the course content

b. end of module objective determination

c. test specifications

d. courseware type selection

e. display (screen) specifications

f. deciding on the branching rules and

g. formulating the response analysis

Most courseware design could be easily handled if they are structured in a segmented or **modular formats** (Dean and Whitlock 1983). The modular approach enables the designer to present clearly defined areas of activity to the students and to take account of differences in students experience and competence. In developing a modular structure, the designer should identify the title of each modules, an objective and the expected learning outcome (Dean and Whitlock 1983).

In line with the above, the content of the course identified in the analysis phase have to be **broken down in to modules and sub-modules** whereby each sub module should represent one concept, rule, principle, etc.

In determining the behavioral **objectives**, in order to reflect the expected learning outcomes, the objectives, must be specified in observable and/or measurable terms. Gagne (1977) categorize learning outcomes into five:

1. Information (knowledge)

2. Intellectual capability

2.1. Discrimination

2.2. Concrete concept

2.3. Defined concept

2.4. Rules

2.5. Problem Solving

3. Cognitive Strategy

4. Motor Skill

5. Attitude

According to ITU/TDG (1979), on the basis of these learning outcomes, one could have the following types of behavioral objectives (tests)

1.0 Verbal behavioral objectives

1.1 Recall a name

1.2 Explain How to do a task

1.3 Define a concept

1.4 Solve a theoretical problem

1.5 Solve a practical problem

2.0 Physical Behavioral objectives

2.1 Identify Things

2.2 Perform simple physical acts (with instruction)

2.3 Perform physical acts (without instruction)

2.4 Perform physical skilled actions (i.e., motor skill)

2.5 Perform physical actions in problem solving situation

2.6 Determine quality of a product or evaluate a performance

3.0 Attitudinal Behavioral Objectives

Either a verbal or a physical behavioral objective or a combination of both

In addition to determining the types of behavioral objectives to be considered, proper specification/description of them is also important in courseware design.

The following table extracted and adapted from ITU/TDG (1979) gives key verbs that will help in stating the objectives.

Table 1: OBJECTIVE SPECIFICATION VERB TABLE

LEARNING OUTCOME	KEY VERB	MAJOR ACTIVITY	OTHER VERBS TO BE USED
Information	State	Verbal chain (produce a sequence of words)	cite, copy, enumerate, list, mention, record, repeat, transcribe, reproduce
Discrimination	Discriminate	Identifying two or more stimuli	choose,compare,contrast, decide, distinguish, detect,differentiate, isolate,judge, recognize select
Concept	Identify Classify	Classifying using concept	allocate,arrange, assign catalogue, categorize, characterize, classify, collect, compile, define,demonstrate, divide, file, group, index, order, rank, sort, specify, tabulate
Simple rule	Determine	Rule using	anticipate, calculate, check, compute, convert, correct, design, determine, examine, explain, figure, illustrate, interpret, organize, plan, predict, solve, translate, verify
Problem solving Cognitive Strategy	Generate Resolve	Solve a problem by combining two or more rules or principles	accommodate, adapt, analyze, compare,create, conclude, construct, coordinate, correlate develop,devise, estimate, evaluate,invent,program, realize, reason,resolve, study, troubleshoot
Motor Skill	Executes	Motor chain producing a sequence of motions	activate, adjust, close, copy, (dis)assemble, (dis)connect, draw, duplicate, insert, load, manipulate, measure, open, operate , remove, replace, trace, turn off & on
Attitude	chooses		

A test in courseware serves different purposes: such as evaluating the aptitude a student already posses; evaluating the aptitude he acquires during training and evaluating the result of his training.

There are different classification of tests such as: **mastery test** (find out whether or not a trainee has attained a performance observed based on what he must be able to do, know, and apply); **progress test** (to measure if a specific performance objective or a series of such objectives are achieved; **diagnostic test** (similar to that of progress test, the diagnostic test usually samples the whole range of capabilities required for a specific course); **discriminatory test** (asses the achievement of one students in relation to others) **retention test** (find out how much of the skills and knowledge acquired during learning are retained by student after a certain time has elapsed) **pre test** (administered before assigning specific modules to students, enables to determine either the modules that are needed for a student or find out whether or not students have the required entry level) **post test** (administered after completion of a module and enables to measure the extent to which a student comprehend the modules)

Tests can also be classified on the basis of the **test items**. The most common test items include: real on the job performance, real performance in simulated situation, simulated performance in simulated situation, alternate response, multiple choice, matching, completion, short answer and essay.

The selection of test categories and test items (commonly known as **Test specification**) generally depends on the objective, i.e, the tests have to reflect the described behavioral

objectives and cover all the aspects there of. In designing tests a number of factors has to be considered which include suitability for keyboard entry, range and number of questions and nature of responses (Dean and Whitlock 1984).

Once the objectives are determined and the corresponding tests are designed, as part of the design phase the next activity is to determine what **type of courseware** (i.e., drill and practice, tutorial, demonstration, simulation and/or games) are needed. The selection of the courseware type(s) depends on the expected learning outcome as each learning outcome is suitable for a selected courseware type.

Screen display refers to the determination of what students or users see on their screen. This will be either text or graphics of some kind.

There are many dimensions to be taken in to account in **creating text** for a screen display. These include **Topography, Readability (vocabulary and sentence length), Enhancement (Highlighting, underlining) and so on**. The **graphics consideration** on the other hand includes presenting concepts that are dynamic or simply to increase motivation or attention.

Beyond the text and graphics consideration, another display variable to be taken in to account is the window size - how much information is presented in a single display. There seems to be a general consensus that a window should present one major idea (concept, example, rule, test, question e.t.c.) (keresely 1983 and Dean and Whitlock 1984).

The nature of instruction, comment, and prompt displayed on the screen and given to the student constitutes another area of concern. As a general principle authors need to grasp how to create interactive display that make student an active participant in the instructional process.

Branching rules refers to the rules which specifies how students get from one lesson, component or display to another. In courseware design there are different branching models which include: *Mastery, Adaptive, Learner control and Aptitude/ treatment interaction.*

In the ***mastery model*** pre and post tests are used to determine if the student has achieved the desired mastery level. If the student hasn't achieved mastery, remedial instructions and/or additional practice is given until mastery is demonstrated. In the ***adaptive model***, the student is continuously branched to varying levels of difficulties based on immediately preceding performance. In the ***learner control model***, students are given the responsibility for choosing their own sequencing and components with the help of a system advisor. In the ***aptitude treatment interaction model***, branching decisions are based on the personality attributes of the learner.

Response analysis: one of the major attractions of courseware as a technique is its ability to react intelligibly to students answers. To do this, the authoring language must be able to take the students reply and analyze it according to rules specified by the course writer.

The most common techniques in **response analysis** are (Dean and Whitlock 1984) **EXACT MATCHES** (where it is necessary to match exactly the students reply with the anticipated

response. Usually applies for numeric responses); **EXTRA WORDS IN THE REPLY** (allowing additional words by looking for a keyword or keywords in a response); **WORDS IN ANY ORDER** (eliciting more than one word regardless of the order); **PERMITTING EXTRA CHARACTERS IN WORDS** (if little importance is given to students spelling the answer correctly); **UPPER AND LOWER CASE** (where it is of little importance weather the reply to a question is entered in capital or small letters or a mixture).

Figure 11 illustrate the activities/components of the design phase

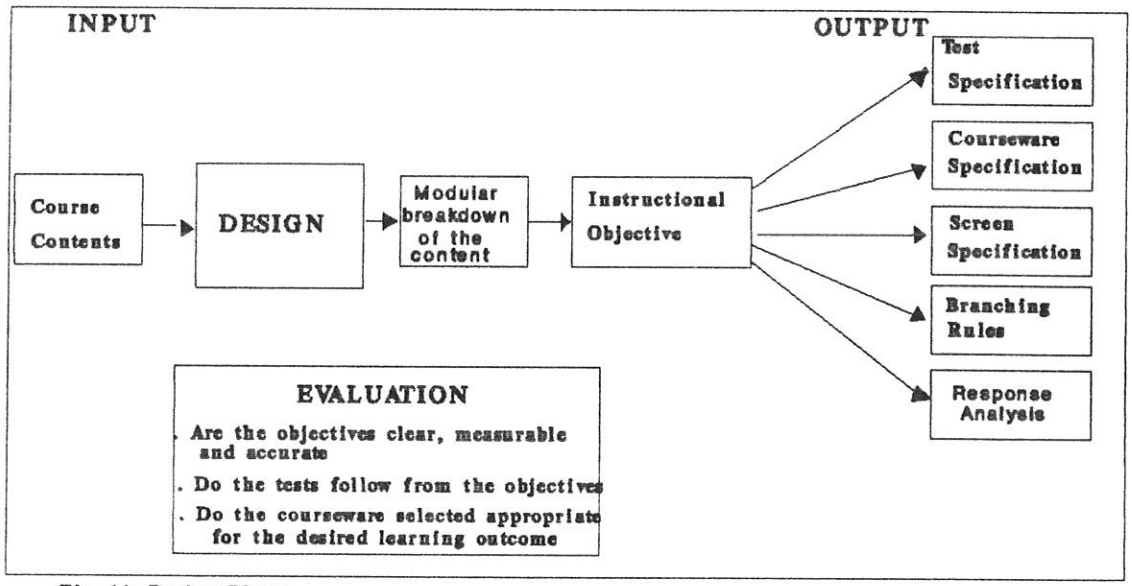


Fig. 11: Design Phase

3.2.3 Development Phase

This phase relates to the authoring process which involves actual design of the lesson, the different screens, interactions and branching, i.e., the creation of instructional programs.

There are four basic steps associated with authoring:

- the representation of the content and logic of the learning activities in some way (eg., hand-written note, flow charts, or diagrams);
- the programming of the content and logic;
- the debugging and testing of the program and
- the development, production and integration of any offline materials involved such as video, micro fiche, work books, slide tapes etc.

In undertaking each of the authoring process steps, effort has to be exerted to map the steps to the different stages of learning. This will be ensured by following the events of instruction. That is gaining attention, informing lesson objective(s), stimulating recall of prior learning, presenting stimuli, guiding learning, eliciting performance, providing informative feedback, assessing performance, enhancing retention and learning transfer (Dean and Whitlock 1984 and Gagne, Wager and Rojas 1981).

The following figure describes the events of instruction.

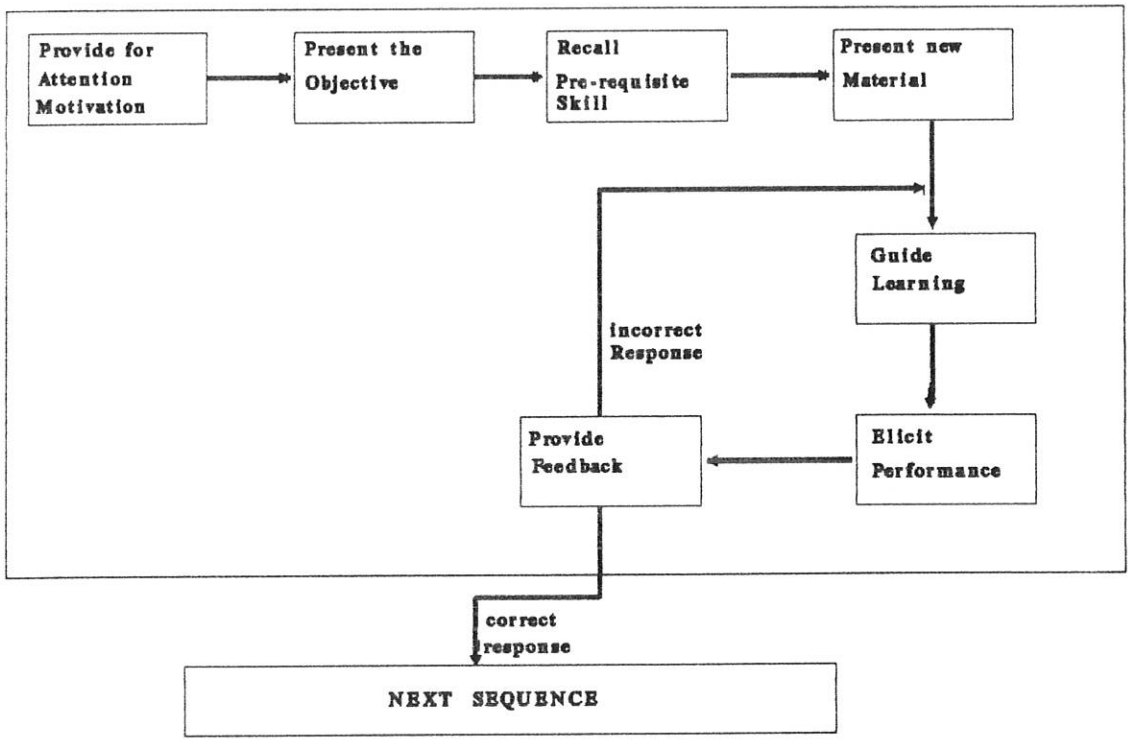


Fig 12: EVENTS OF INSTRUCTION

Generally, the development phase activities are represented by the following diagram

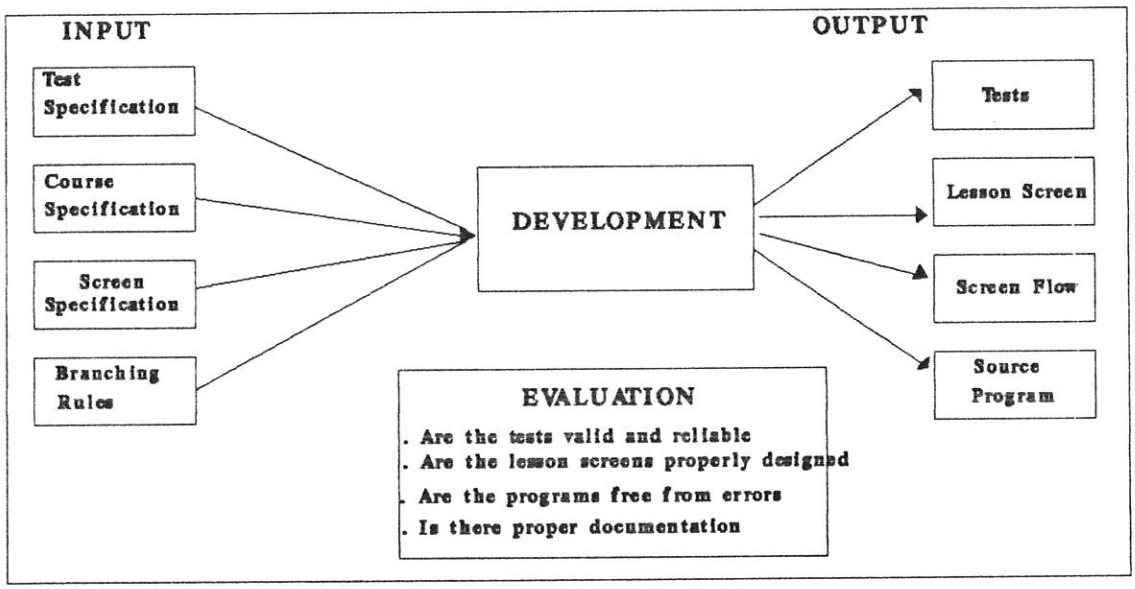


Fig. 13: The Development Phase

3.2.4 Implementation Phase

The implementation phase involves installation and start up of the system, initial and ongoing training of all system users such as students, instructors, and operators. It also involves on going revision of courseware. Figure 14 represents the implementation phase activities/components

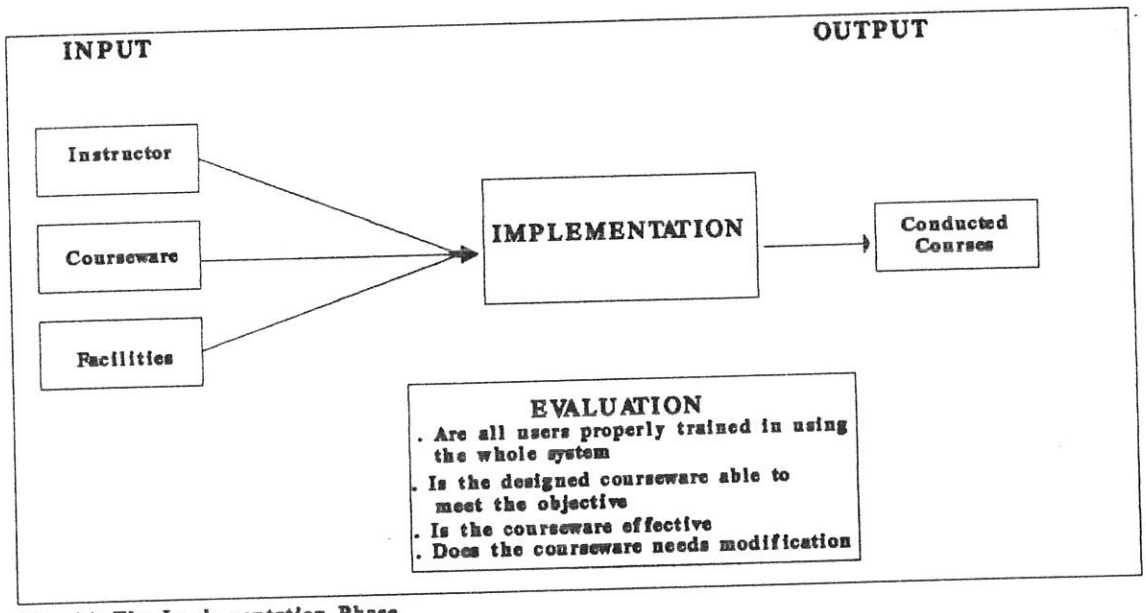


Fig. 14: The Implementation Phase

3.2.5 Evaluation Phase

The evaluation phase is not an activity to be considered at last as seems to be from its relative position in the description above, but it is an activity that goes on during the preceding phases. It is to ensure whether the analysis, design and development undertaken are correct.

Revisions are also made on the formulations of objectives and lesson specifications as deemed necessary. As a whole this evaluation phase aims at evaluating the effectiveness of the courseware and testing the usefulness of the instructional material capability.

3.3 LIMITATIONS

Usually courseware development is a team work which involves a variety of specialists. Some of the expertise needed are Subject Matter Experts(SME), Instructional System Design (ISD) experts, Computer Assisted Instruction (CAI) design experts, etc. In this study, though effort will be exerted to form a role playing team to represent the team composition requirements, the effort in this regard, however, couldn't be taken as such a success due to lack of resource persons. Therefore the study may be limited by the lack of such expert opinion.

In courseware design the programmer needs languages that will permit him to present questions to the student, analyze responses, give feedback and remedial help if necessary, etc. This implies that the language should be fairly easy to learn and use and not require an indepth knowledge of computing (Dean and Whitlock 1984). The type of features mentioned above, however, could not be found in most of the general purpose languages, but they are common to author languages.

In the absence of such authoring languages, it is reported in the literatures (Dean and Whitlock 1984), that it is possible to write courseware programs in general purpose programming language. Dean and Whitlock (1984) reported a variety of such coursewares written in general purpose language

Due to an unavailability of authoring software, the language used for writing the program of the prototype inventory management courseware is Turbo Vision. Turbo vision, being a general purpose language, lacks those mentioned features of authoring languages.

In addition as it is true with the conventional ways of instruction, the developed courseware has a built in rigidity. The subject content is restricted to a defined scope and can not address topics beyond that scope regardless of the desires of the students. This rigidity to some degree, also, applies for the students response. That is all correct student input and some(predefined) incorrect input to a given question can be responded appropriately. However the program can't deal with unpredictable responses of students such as those who want rephrasing of the question and ask for clarification.

CHAPTER FOUR

THE COURSEWARE DEVELOPMENT PROCESS

4.1 REQUIREMENT ANALYSIS

The attempt in this and subsequent sections of this chapter is to apply the design method (ISD) selected for the course work and as outlined in the preceding chapter to the course area selected.

4.1.1 Student Requirement

As indicated in chapter three of this study, the case area considered in the design of the courseware is Inventory Management. This section attempts to define/determine the requirements of students to be considered in the design of the courseware. This is important particularly to determine the entry level for the courseware- to identify the intended type of audience.

In the selected case area, usually, students majoring in purchasing and supplies management take the inventory management course in their second year - second semester and third year second semester for the regular and extension students respectively. In order to register for this course, beside the common courses like English, Basic Mathematics, etc students have to complete the prerequisite courses and obtain a minimum grade of "C" in each one of them. These prerequisite courses are Purchasing Management part I and II, and Stores Management.

CHAPTER FOUR

THE COURSEWARE DEVELOPMENT PROCESS

4.1 REQUIREMENT ANALYSIS

The attempt in this and subsequent sections of this chapter is to apply the design method (ISD) selected for the course work and as outlined in the preceding chapter to the course area selected.

4.1.1 Student Requirement

As indicated in chapter three of this study, the case area considered in the design of the courseware is Inventory Management. This section attempts to define/determine the requirements of students to be considered in the design of the courseware. This is important particularly to determine the entry level for the courseware- to identify the intended type of audience.

In the selected case area, usually, students majoring in purchasing and supplies management take the inventory management course in their second year - second semester and third year second semester for the regular and extension students respectively. In order to register for this course, beside the common courses like English, Basic Mathematics, etc students have to complete the prerequisite courses and obtain a minimum grade of "C" in each one of them. These prerequisite courses are Purchasing Management part I and II, and Stores Management.

From the analysis of the prerequisite courses, a student, in order to make effective use of this inventory management courseware and get the best benefit out of it, has to fulfil the following entry level requirements

- Average fluency in English
- Basic knowledge of operating microcomputer system
- Familiar with business terms and languages
- Ability to explain the principles and functions of management
- Familiarity with elementary statistics tools and techniques
- Understanding of the principles and practice of purchasing
- Familiarity with the storage and material handling activities

4.1.2 Course Requirement

The Department of Purchasing and Supplies Management of AACC has objectives in training students for a diploma in Purchasing and Supplies Management. A student, after completing the two years-(for a regular students) and three years-(for extension students) program has to be able to:

- secure materials needed in any organization at a reliable and competitive price
- efficiently manage and control the supplies acquired, and
- ensure maximum usage and minimum wastage

So students upon completing all the necessary requirements according to the college standards are expected to achieve the above basic goals of the department. To this end, the department, has included in the curriculum all the necessary and relevant courses with their respective details-topics to be covered.

Accordingly, Inventory Management, being one of the major requirements for the fulfilment of the diploma in Purchasing and Supplies Management at AACC has the following standard course content (outline).

1. INTRODUCTION

1.1 Types of Inventories

1.2 Functions of Inventories

1.3 Objectives of Inventory Control

2. MATERIALS PLANNING AND BUDGETING

2.1 Forecasting

2.2 Approaches to Forecasting

2.3 Methods of Forecasting

2.4 Materials Budget

3. INVENTORY ANALYSIS

3.1 Selective Inventory Management

3.2 ABC Analysis

4. INVENTORY CONTROL SYSTEMS

4.1 Fixed Order Quantity System

4.2 Cyclical Ordering System (P system)

4.3 Material Requirement Planning

4.4 Japanese Just In Time (JIT) system

4.5 Other Systems

5. STOCK CONTROL

5.1 Scope and Purpose

5.2 Determination of Order Quantity

5.3 Economic Order Quantity Concept and Computation

5.4 Uncertainty in Inventory Management

5.5 Approaches to Handling Uncertainties

5.6 Order Points, Safety Stocks and Service Levels

6. STOCK RECORDS, STOCK TAKING AND MATERIALS PRICING

6.1 Stock Records

6.2 Stock Taking

6.3 Stock Obsolescence and Redundancy

6.4 Pricing Material Issues

7. STORES MANAGEMENT AND ELECTRONIC DATA PROCESSING

4.1.3 Other Requirements

According to the design model selected for the current work, Besides the course and student requirements, the courseware development team composition and the selection of the authoring language for writing the program are two additional areas which need consideration in courseware development.

THE TEAM

Courseware development is usually a team work which requires the expertise of different personnel. As indicated in chapter three, some of the basic expertise needed are Subject Matter Expert, Instructional System Design expert and Programming expert.

For the purpose of this study and in due consideration of the constraints with in which the study operates, a role playing team was formed. In particular, Dr. Taye Tadesse who has a good background in economics and management was actively involved in revising and modifying the lessons prepared by assuming the role of the subject matter expert.

Ato Tesfaye Biru, who is the instructor of the information systems analysis, design and evaluation course (part 1 and 2) at The School Of Information Studies for Africa (SISA) and who has a consolidated experience in systems analysis and design was involved in the design of the courseware as the instructional system design expert.

The worker in view of his experience in teaching, course developments and instructional strategies as well as familiarity with a number of programming languages, plays the role of both programmer and subject expert.

AUTHORING LANGUAGE

Due, mainly, to the unavailability of a full-fledged authoring software on site, Turbo Vision version 2.0 (a high level language with object oriented features) is selected for use in the

development of the courseware.Turbo Vision is a complete object-oriented application framework including

- Multiple, resizable and overlapping windows
- Pull down menus
- Mouse support
- Dialog Boxes
- Data validation
- Built-in colour installation
- Buttons, Scroll bars, input boxes, check boxes and radio buttons
- Standard handling of keystrokes and mouse clicks

In addition, Turbo Vision is an event driven programming software. It enables to write flexible programs that give program users control over what part of the program they want to access. Beside, the use of turbo vision enables one to save an enormous amount of unnecessary, repetitive work and provides a proven application framework that can be trusted and applied to build new applications.

4.2 GENERAL DESIGN

According to the ISD model, the general design phase involves module breakdown, objective determination, and design specifications. This is the phase which provides the blue print of the courseware design.

4.2.1 Module Breakdown

Module Breakdown involves decomposing of the whole course in to modular and sub-modular parts. This is done in such a way that a single module has to emphasize more or less one concept or rule or principle. Such breakdown of the course in to modules and sub modules helps to determine the objectives and good design of the courseware.

Accordingly the content of inventory management course outlined in section 4.1.2 of this chapter is divided in to 7 main topics and 24 modules. Figure 14 illustrates the hierarchical structure of the courseware modules. Table 2 shows the title of the main topics and the modules that are to be included in each of the main topics.

TABLE 2: MAIN TOPICS AND MODULE TITLES

MAIN TO PIC	MODULE NUMBER	MODULE TITLES
INTRODUCTION	M1	Function and types of inventories
	M2	Objectives of Inventory control
MATERIAL PLANNING AND BUDGETING	M3	Forecasting: importance features and approach
	M4	Methods of forecasting
	M5	Materials Budget
INVENTORY ANALYSIS	M6	Selective Inventory mgt.
	M7	ABC analysis
	M8	VED analysis
	M9	Other Methods
INVENTORY CONTROL	M10	Cyclical Ordering System
	M11	Order Point System
	M12	Material Requirement Planning
	M13	Just In time System
	M14	Other system
STOCK CONTROL	M15	Purpose and scope of stock control
	M16	Order Quantities
	M17	Basic Economic Order Quantity computation
	M18	Modified Economic Order Quantity computation
	M19	Uncertainty in Inventory Control
	M20	Re-order Point
STOCK RECORDS AND STOCK TAKING	M21	Stock Records
	M22	Stock Taking
	M23	Obsolescence
COMPUTERS AND INVENTORY MANAGEMENT	M24	Computers and Inventory Management

4.2.2 Instructional Objectives

Objective determination, which follows the module breakdown, enables to know where one is going and at the same time to recognize the place when one get there. It also enables to know and measure whether a student reaches at the desired level of performance or not.

In addition, good determination of the objectives facilitates the design of the contents of the courseware by giving guidance. The following table provides the behavioral objectives of the inventory management courseware.

TABLE 3: MODULE OBJECTIVE SPECIFICATION OF THE COURSEWARE

COURSEWARE: INVENTORY MANAGEMENT		
	MAIN TOPIC:INTRODUCTION	
MODULE NUMBR.	**LEARNING OUTCOME**	**OBJECTIVE** **At the end of the module the student will be able to:**
M1	Concept and information	1. Recognize the definition of inventories 2. List and classify the different types of inventories 3. List functions of inventories
M2	Simple rule	1. Recognize the objectives of inventory control from the view point of different Departments
	MAIN TOPIC: MATERIAL PLANNING AND BUDGETING	
M3	Information and concept	1. Mention the importance and features of forecasting 2. List the steps involved in forecasting 3. Define the two approaches in forecasting
M4	Simple rule and information	1. List and explain forecasting methods 2. Develop a material forecast using time series analysis and moving average forecasting methods
M5	Concept and simple rule	1. Define budget and materials budget 2. Develop a materials budget
	MAIN TOPIC:INVENTORY ANALYSIS	
M6	Concept and Simple rule	1. Define and explain the concept of selective inventory management

COURSEWARE: INVENTORY MANAGEMENT		
M7	Information concept and simple rule	1. Define and explain the concept of ABC analysis 2. List the steps involved in the ABC analysis 3. Calculate usage value and cumulative usage value by applying the formula 4. Explain the guideline for policy implementation 5. Compute optimum no. of orders for ABC and no. of orders per each class 6. Undertake ABC analysis
M8	Simple rule and concept	1. Define VED analysis 2. Explain Items to be categorized under VED 3. Determine and categorize items in their V, E, D category
M9	Concept	1. Define SDE, HML, FNSD and XYZ methods of inventory analysis
	MAIN TOPIC:INVENTORY CONTROL	
M10	Concept, simple rule, information and discrimination	1. Define the method of cyclical ordering system 2. list and explain the different systems to be categorized under this category 3. Recognize the disadvantages of cyclical ordering system
M11	Information discrimination and concept	1. Define the order point system 2. Identify the requirements for applying each of these system 3. Recognize the advantages & disadvantages associated with order point system
M12	Information Discrimination, Concept, simple rule	1. Define the following terminologies: Allocated available & on hand inventory, Bucket, Dependent & lumpy demand, End item, Gross and net requirement, planned order receipt & release, Product structure levels, safety stock, service parts, MRP 2. List the objective of MRP 3. List and explain the elements of MRP 4. List and recognize the steps to calculate MRP 5. Calculate and develop MRP using the steps 6. Explain the requirement for applying MRP
M13	Simple rule	1. Explain the objective of JIT system 2. Explain how the JIT system works
M14	Concept and simple rule	1. Define and explain Dollar Time Limits, Inventory Turnover, Visual control (imprest, open access bin, one-for-one, operational replenishment) systems of inventory control
	MAIN TOPIC: STOCK CONTROL	

COURSEWARE: INVENTORY MANAGEMENT		
M15	Information	1. List the purpose, scope and aims of stock control 2. Cite the sources of information for stock control 3. Enumerate qualities of good stock control system 4. List the methods to improve the control of stock 5. list features to be considered in determining stock levels
M16	Information Discrimination, concept	1. Recognize factors that determine the order quantities 2. Define EOQ 3. Recognize and explain the costs in EOQ 4. Enumerate the limitations of EOQ
M17	Information Discrimination, Simple rule	1. List the steps involved in EOQ computation 2. Recognize the relevant assumptions in EOQ computation 3. Following all the necessary steps and taking the relevant assumptions compute the EOQ
M18	Discrimination and simple rule	1. Recognize the steps necessary and compute modified EOQ with the gradual deliveries and quantity discounts
M19	Simple rule	1. Describe sources of uncertainty in inventory controlling 2. Explain the approaches for handling uncertainties
M20	Information Discrimination, Simple rule and concept	1. List the four determinants of reorder point quantity 2. Identify the four cases for which re-order point models are to be considered 3. Explain the concepts of safety stock and service level and recognize their application for re-order level computation 4. Compute safety stock 5. Determine the reorder level applying the relevant formula for the four different cases 6. Explain the concepts of Fixed order interval and single period model and determine the amount to be ordered by using these models 7. Recognize the advantages and disadvantages associated with each of these models
MAIN TOPIC: STOCK RECORDS AND STOCK TAKING		
M21	Information Discrimination, Simple rule, concept	1. Define Stock records 2. Recognize the purpose of stock records 3. List and explain the Methods of stock record 4. Identify the sources of information for stock records 5. Enumerate factors to be considered in formulating a stock record system

COURSEWARE: INVENTORY MANAGEMENT		
M22	Information Discrimination	1. List The importance, benefits and methods of stock taking 2. Decide the information to be included in stock taking 3. Cite causes for discrepancies and the classification
M23	Information and concept	1. Give the Definition of Obsolete stocks 2. List The causes for obsolescence and ways to reduce obsolete

COURSEWARE: INVENTORY MANAGEMENT		
	MAIN TOPIC: COMPUTERS AND INVENTORY MGT	
M24	Discrimination, Information and concept	1. Define computer 2. Describe the different components of a computer 3. Define Database 4. Recognize importance of computers in inventory management 5. Recognize how a computer is used in managing inventories

4.2.3 Courseware Design Specification

The courseware design specification as indicated in section 3.1.1 includes selection of the courseware type, test specification, screen specification and determination of the branching rules.

TYPE OF COURSEWARE

As indicated in section 2.3 of this study, the selection of the courseware type (drill and practice, tutorial, demonstration, simulation and instructional game) largely depends on the expected learning (instructional) outcome(s).

On the basis of the expected learning outcome of each modules of the inventory management courseware, as defined in table 3, three, namely, information, concept, and simple rule learning outcome could be clearly identified. To achieve these different learning outcomes involved in the inventory management courseware, the courseware type must as well be varied.

The type of courseware selected for inventory management courseware under consideration is the combination of drills and practice, tutorials and simulations. This choice is made on the basis of the expected learning outcome of most of the modules.

TEST SPECIFICATION

Among the various classification of tests discussed earlier in section 3.1.1; mastery test and progress test, are considered for implementation in the designed courseware.

In addition, simulated performance in simulated situation, multiple choice, completion and alternate response types of tests will also be used as deemed necessary in formulating the test items.

SCREEN SPECIFICATION

One of the screen specification considerations is the presentation of one major idea (concept, example, rule, test question...) within a window. The inventory management courseware under consideration consists of several windows with the following titles: Objective, lesson, highlight, test and information.

The window titles are all self explanatory. The objective windows provide the objectives of each of the different modules as stated in Table 3 of this chapter. While Windows with the title lesson, test and information contain the lesson, test and other related information on the courseware, respectively.

The content of the windows, with the title highlighted, is basically the same as that of the lesson screens. The difference however is that, this type of windows will be opened only to elaborate on a concept that is selected from the lesson screens by the users. Windows with the title help contains the general commands of the courseware and major points of the lessons. An experienced user, alternatively, can make use of the branching commands and/or the texsts in the help screen to jump through opening windows.

In addition, there is a sound effect with animation when the courseware program is activated, where the objective screens are opened and the user achieves acceptable performance in doing the tests.

The nature of instruction given to the user in using the courseware windows excepting some few cases, mostly includes the following which will be available online to the user by pressing F1.

ALT + X: TO EXIT FROM THE COURSEWARE TO THE DOS PROMPT

ALT +Y: TO SELECT A COURSEWARE MENU WHERE Y REPRESENTS THE MENU TITLES SUCH AS INTRODUCTION, PLANNING, ANALYSIS...

F10 +↓ : TO SELECT THE COURSEWARE MENU

QUIT : TO EXIT FROM A GIVEN WINDOW OR DIALOG BOX TO THE COURSEWARE MENU

CONTINUE: TO CONTINUE TO THE NEXT SEQUENCE

OK : TO ACCEPT THE ARGUMENT IN THE DIALOG BOX

BACK : TO GO BACK TO THE ORIGINAL CALLING WINDOW

F1 : TO ACTIVATE ON LINE HELP

The user will also be provided with the key words that are used to mainly open the contents of the highlighted screens. Moreover, there may be different prompt and instruction windows that elaborate on what is expected from a student in an attempt to make the courseware more interactive.

With regard to the screen topography, in most of the cases, the last row (the lower bottom) and first row (the upper top part) of the screen contains comments and/or instructions and the courseware menu items respectively. The remaining part, unless otherwise a menu is selected in which case the pull down menu bar will be displayed or a module is activated, will be empty. In the later case depending on the size of the text content, a window will be displayed with the one or more of the instructions given above on what to do next. The user will also be given an option to change the video mode if he/she wants to do so by activating the help menu which also contains the about message.

In addition to the above, the courseware screens also make use of animation and sound effect to emphasize the concept being discussed.

BRANCHING RULES

As indicated in chapter three of this study, there are different branching rules which guide a student in moving from one lesson to another. In this respect,

1. to make the courseware more user friendly (interactive)
2. to unnecessarily limit the choice of a second and third time user
3. in due consideration of the need for individualization
4. to provide the courseware a potential of serving as a reference material where a user can use it for quick reference

the learner control branching rule will be used in the inventory management courseware. As a result a student is given free and total control supported by an advise both on the selection of a main topic or a specific module.

The movement of the student within the content of a particular module, however, is determined based on the adaptive model. This module is chosen for the main reason that it enables to branch a student to a higher level of difficulty by starting from the lower level. But even so to add more flexibility on the courseware, the student is provided with a chance of reverting this sequence (model) by progressing using the help text.

RESPONSE ANALYSIS

The response analysis to be implemented in the current prototype courseware is mainly **exact matching and upper and lower case.**

The following table summarizes the different design specifications considered for the development of the inventory management courseware.

TABLE 4: SUMMARY OF THE INVENTORY MANAGEMENT COURSEWARE DESIGN SPECIFICATIONS

COURSEWARE TYPE	TEST SPECIFICATION	SCREEN SPECIFICATION	BRANCHING RULES	RESPONSE ANALYSIS
Drills and Practice, Tutorials and Simulation	TEST CLASSIFICATION Mastery test, Progress test, TEST ITEM TYPES Simulated Performance in simulated situation, Multiple choice, Completion and Alternate response	SCREEN TITLES Objective Screen Lesson screen Highlighted Screen Test Screen Help Screen INSTRUCTIONS ALT+ X: Exit ALT+I,C,A,:Menu F10 + ↓:Menu F1: Help QUIT CONTINUE: OK: Accept BACK	Learner Control Adaptive	Exact match Upper or lower case

The hierarchical structure diagram of the initial prototype courseware is provided below

4.3 COURSEWARE DEVELOPMENT

The development aspect of a courseware, as indicated in chapter 3, relates to the authoring process which involves actual design of tests, lessons, different screens, interactions, branching, and response analysis i.e., the creation of instructional program.

However, prior to actual programming of the courseware, representation of tests, lessons, questions, different screens and program structure in some form (eg., hand-written note, flow charts, or diagrams) on paper are essential.

The remainder of this section deals with such representation for the inventory management courseware prepared on the basis of the design specifications described in the preceding section.

In order to keep the size of this report manageable and taking in to account the purpose (which is demonstration of the development of a courseware), the tests and lesson screens presented under this section are only samples (for complete representation of tests for the lessons please refer to Appendix 1)

4.3.1 Test Construction

MODULE 1: TEST

INSTRUCTION: *ANSWER THE FOLLOWING QUESTIONS IN THE SPACE PROVIDED*

TS 1

1. Inventories usually refer to a high valued items held by an organization (TRUE/FALSE)____

2. The kind of business a firm undertakes usually determines the items to be carried out in inventory(TRUE/FALSE). _____

3. Inventories in Manufacturing enterprise can be classified into two as _____and _____

4. One of the following is not a reason in maintaining inventories

   a. Meeting anticipated demand

   b. Promoting stock out

   c. Smoothing production requirement

   d. Taking stock cycle advantage

   Your choice ____

<QUIT>

MODULE 4: TEST :

INSTRUCTION: *ANSWER THE FOLLOWING QUESTIONS*

TS 4.3

8. Dire dawa textile mills has kept records of usage for one of its items for the last four years as shown below

YEAR	QUARTER1	QUARTER2	QUARTER3	QUARTER4
1983	5500	6400	6800	5000
1984	6000	6500	7000	6100
1985	5800	6400	6700	6000
1986	6600	7200	8000	6300

Forecast the factory's use of the item for each quarters of 1987 using trend analysis.

Enter Your answer in the space provided.

a.n= ____ b. The summation of X (period)____

c. The summation of XY _____

d. The summation of X^2 ____

e. The summation of Y (demand)____

f. The Value of '**a**' = _____

g. The value of '**b**'= ____

h. The Regression line Equation = _____

i. Forecast for the 1st quarter = _____

j. Forecast for the 2nd quarter = _____

k. Forecast for the 3rd quarter = _____

l. Forecast for the 4th quarter = _____

<PREV> <NEXT> <QUIT>

4.3.2 Lesson Preparation

MAIN TOPIC: INTRODUCTION

MODULE NO. 1: FUNCTION AND TYPE OF INVENTORIES

LS 1

Any organization, regardless of its size or orientation, needs material resources. But due to the gap in time distance and place and uncertainty, organizations may not get these resources as they want them from the market. SO WHAT SHOULD BE DONE?_____
<CONTINUE> <QUIT>

LS 2

YES! They have to keep a stock of these resources- **INVENTORIES**
<CONTINUE> *<QUIT>* *<INVENTORY>*

S 1

Inventories are idle physical stock of goods ranging from small things like pencils, paper clips to machines, trucks possessing an economic value and Kept for the purpose of future use
<CONTINUE> *<QUIT>* *<BACK>*

HS 16.7a

The formula needed in calculating the trend are

$$YC = a + b_x$$

$$\sum y = na + \sum x(b)$$

$$\sum xy = \sum x(a) + \sum x^2(b)$$

and before actually starting computation you have to identify the x's and y's and determine

$$\sum x, \sum y, \sum xy, \sum x^2$$

<CONTINUE> *<QUIT>* *<BACK>*

HS 16.8a

Now watch very carefully how the computation is going to be done

Example: The monthly sales record of XYZ organization for the last three years show the following figure. XYZ wants to forecast its demand for the coming year (4^{th} year). How much would that demand be.

MONTH	YEAR 1		YEAR 2		YEAR 3	
	Period	DD	Period	DD	Period	DD
JANUARY	1	100	13	125	25	146
FEBRUARY	2	128	14	148	26	175
MARCH	3	138	15	155	27	173
APRIL	4	127	16	138	28	161
MAY	5	112	17	128	29	150
JUNE	6	115	18	142	30	162
JULY	7	128	19	152	31	171
AUGUST	8	134	20	156	32	173
SEPTEMBER	9	130	21	140	33	168
OCTOBER	10	116	22	122	34	157
NOVEMBER	11	100	23	108	35	130
DECEMBER	12	95	24	100	36	125

16.9a

SOLUTION:

From the above data the

x = the time period (the period number which ranges from 1 up to 36)

y = the demand.

n = Total time period which is 36 Therefore

$\sum y = 4928$ $\sum x = 666$

$\sum xy = 16206$ $\sum x^2 = 95902$

MONTH (X)	DEMAND (XY)	XY	X^2
1	100	100	1
2	128	256	4
3	138	414	9
....	....	....	..
36	125	4500	1296
666	4928	95902	16206

<CONTINUE> <QUIT> <BACK>

HS 16.10a

So the equation for finding the values 'a' and 'b',with these two variables unknown,looks like the following

$$\sum y = na + \sum x(b) \quad = 4928 = 36a + 666b \ldots\ldots\ldots 1$$

$$\sum xy = \sum x(a) + \sum x^2(b) \quad = 95902 = 666a + 6206b \ldots 2$$

To solve the two equations simultaneously and get the values of a and b, we will multiply equation 1 by -18.5 (666/36) and add it to equation 2. After the multiplication, equation 1 becomes

-91168 = -666a - 12321b

<CONTINUE> *<BACK>* *<QUIT>*

HS 16.11a

When we add equation 1 to equation 2 we get

$$-91168 = -666a - 12321b \quad \ldots\ldots 1$$

$$95902 = 666a + 6206b \quad \ldots\ldots 2$$

$$4734 = 3885b$$

$$b = \frac{4734}{3885}$$

$$b = 1.22$$

by substituting the values of '**b**' in equation 2 we can solve the value for '**a**' as follows

$$95902 = 666a + 16206b$$

$$95902 = 666a + 16206 * 1.22$$

$$666a = 95902 - 19771$$

$$a = \frac{76131}{666}$$

$$a = 119.31$$

By substituting the values of '**a**' and '**b**' in the

YC = a + bx equation, we will get the following regression line equation.

YC = 119.31 + 1.22x

<CONTINUE> *<QUIT>* *<BACK>*

In addition to the test and lesson screens, the courseware will have an objective and information screens.

The content of the objective screens is the same as the one presented in table 2. Sample of such objective screen is presented as under

OS1

At the end of this module you will be able to

1. Recognize the definition of inventories
2. List and classify the different types of inventories
3. list functions of inventories

<CONTINUE> *<CANCEL>*

The following are the samples of the different information screens

IS 1

WELCOME !!!

INVENTORY MANAGEMENT COURSEWARE

IS 2

IF YOU NEED HELP ON HOW TO USE

THE COURSEWARE PRESS F1

OTHERWISE PRESS ENTER

IS 3

SORRY! YOU CAN NOT HAVE THIS LESSON IT IS UNDER DEVELOPMENT *< OK >*

IS 4

EVENT HANDLING ERROR !!! PLEASE ENTER ONE OF THE OPTIONS INDICATED IN THE DIALOG BOX *< OK >*

IS 5

OUT OF MEMORY !!! IT IS NOT POSSIBLE TO COMPLETE THIS OPERATION *< OK >*

IS 6

INTERRUPTED !!! PRESS ENTER TO GO BACK TO THE COURSEWARE MENU *< OK >*

4.3.3 Screen Flow

The following diagrams represent the screen flows of the courseware.

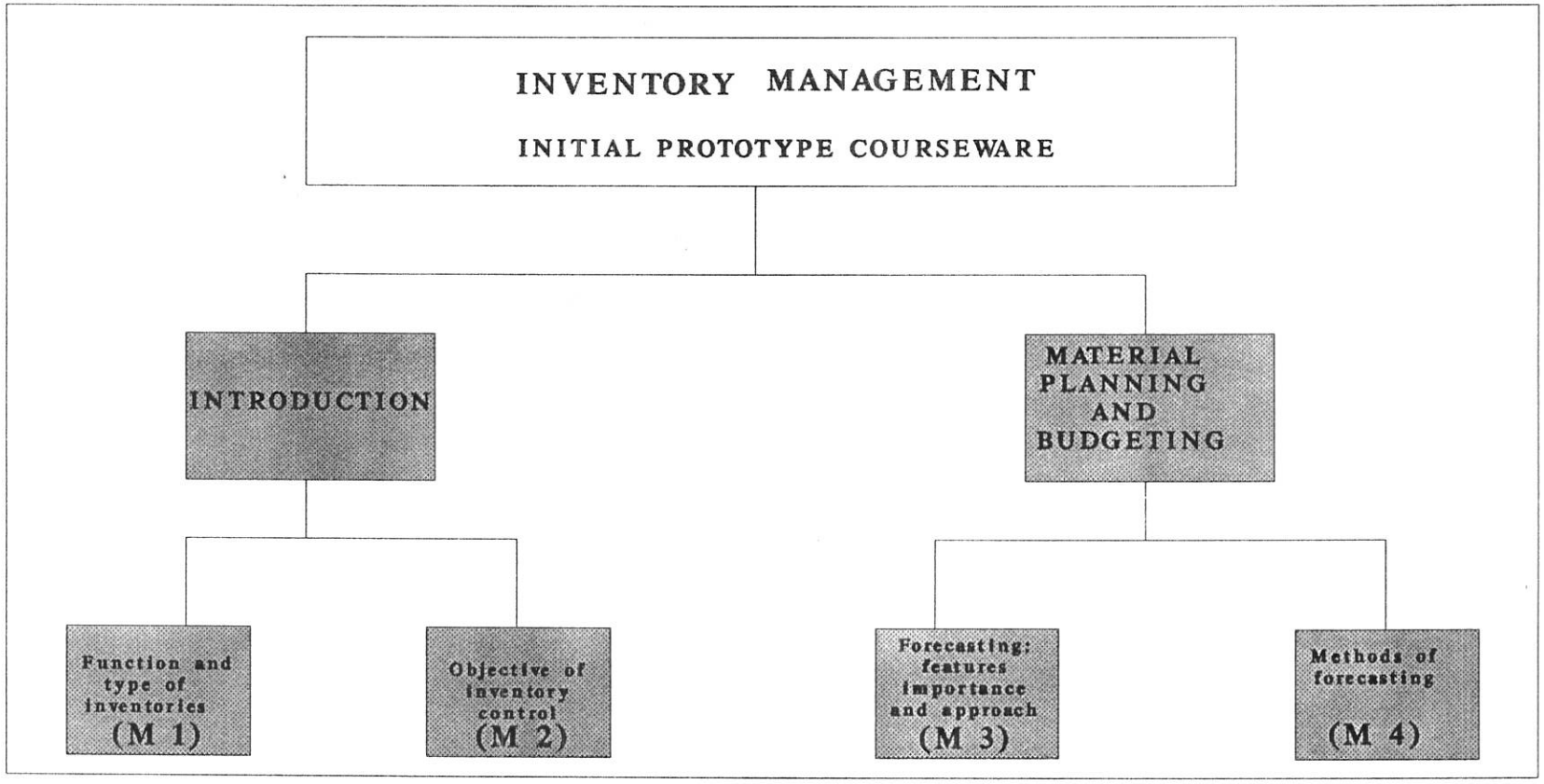


FIG. 16 : Hierarchical structure of the initial prototype courseware

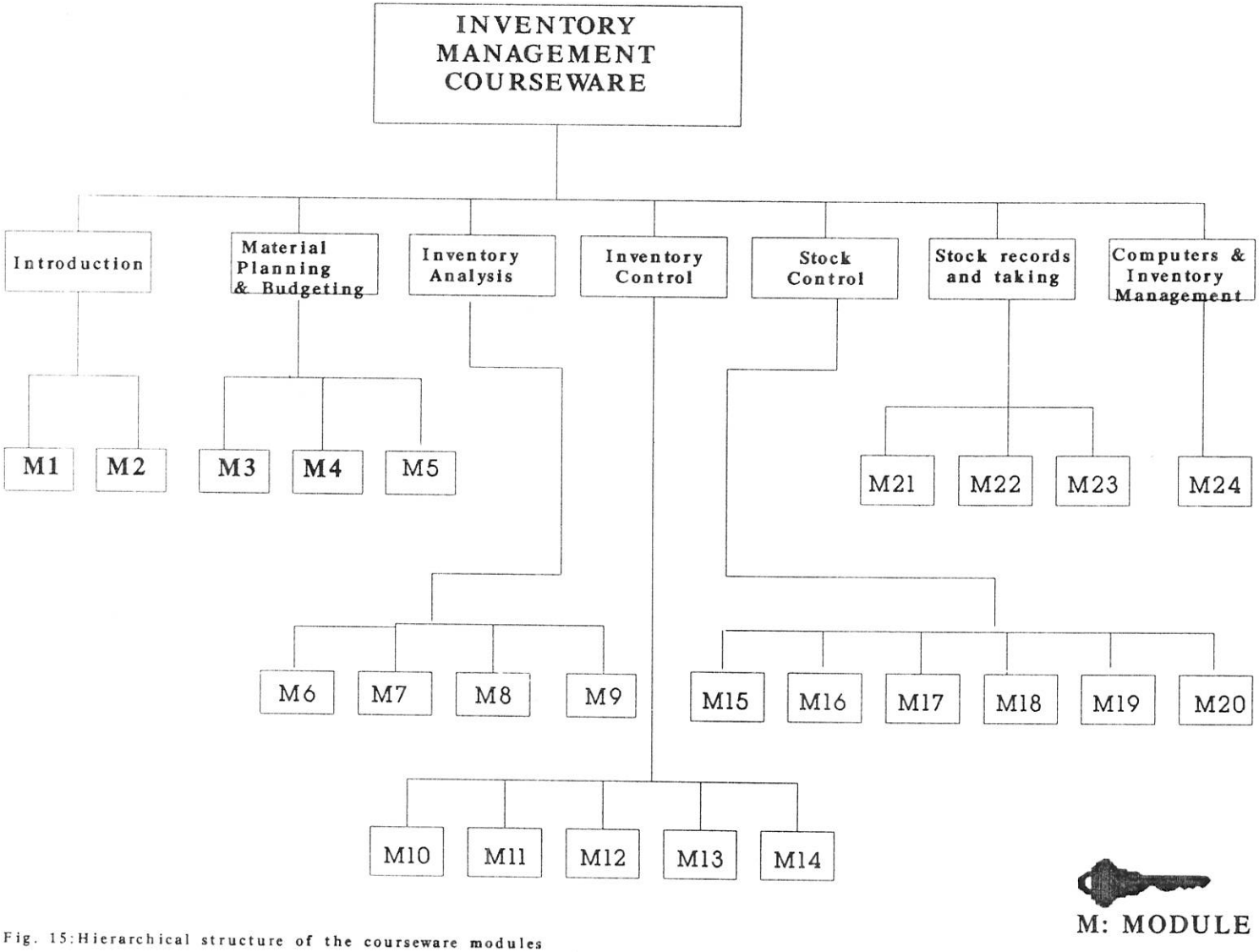
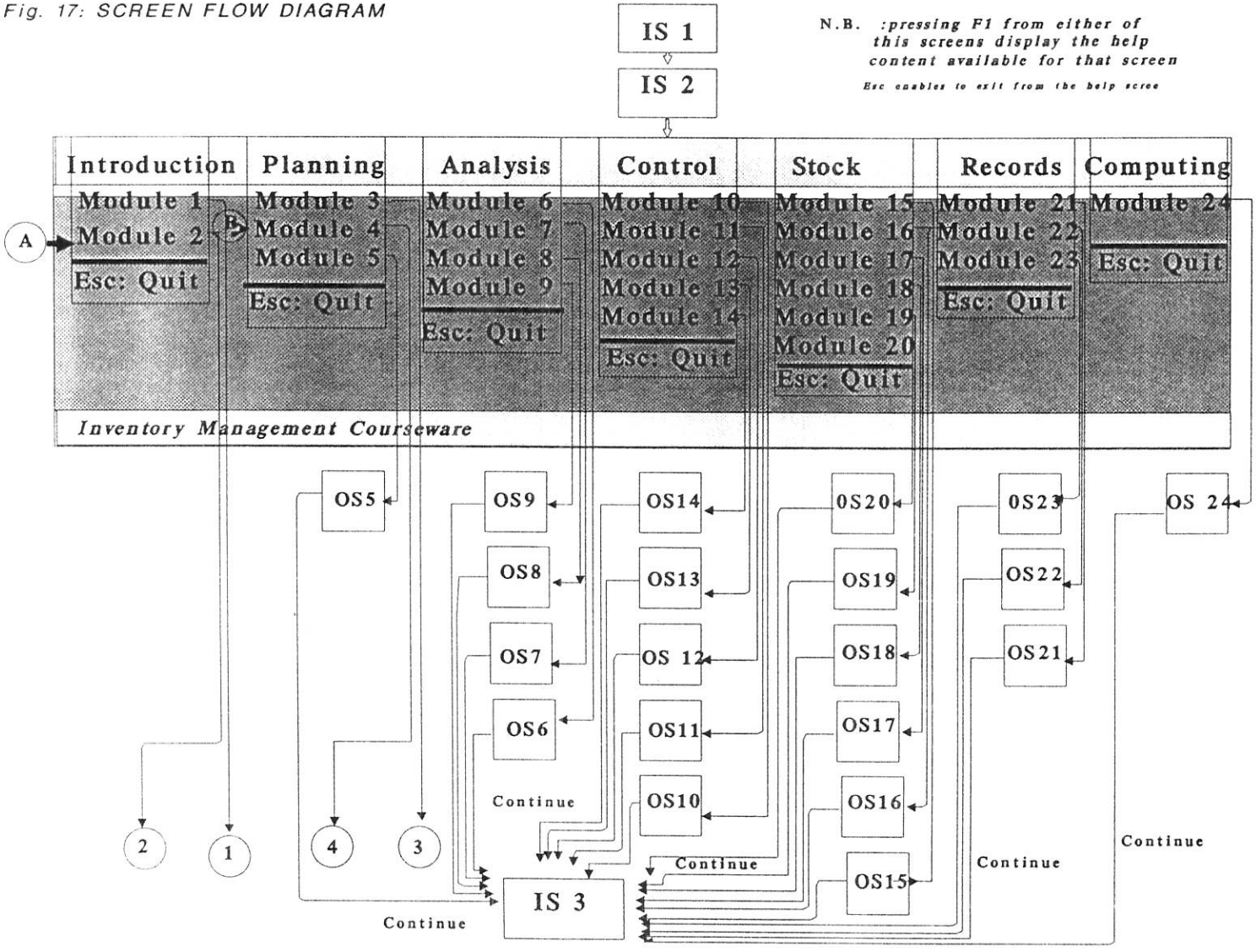


Fig. 15:Hierarchical structure of the courseware modules

Fig. 17: SCREEN FLOW DIAGRAM



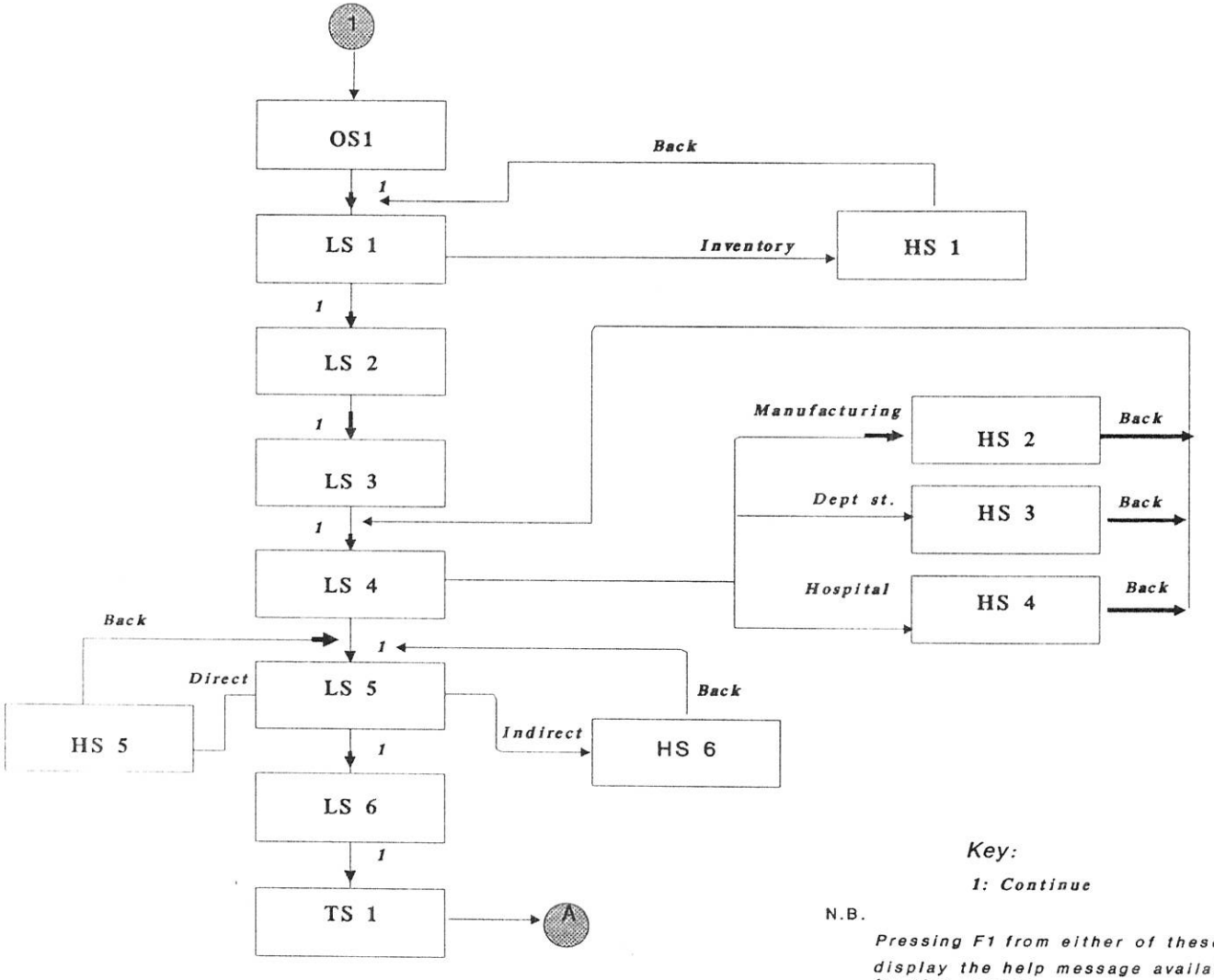


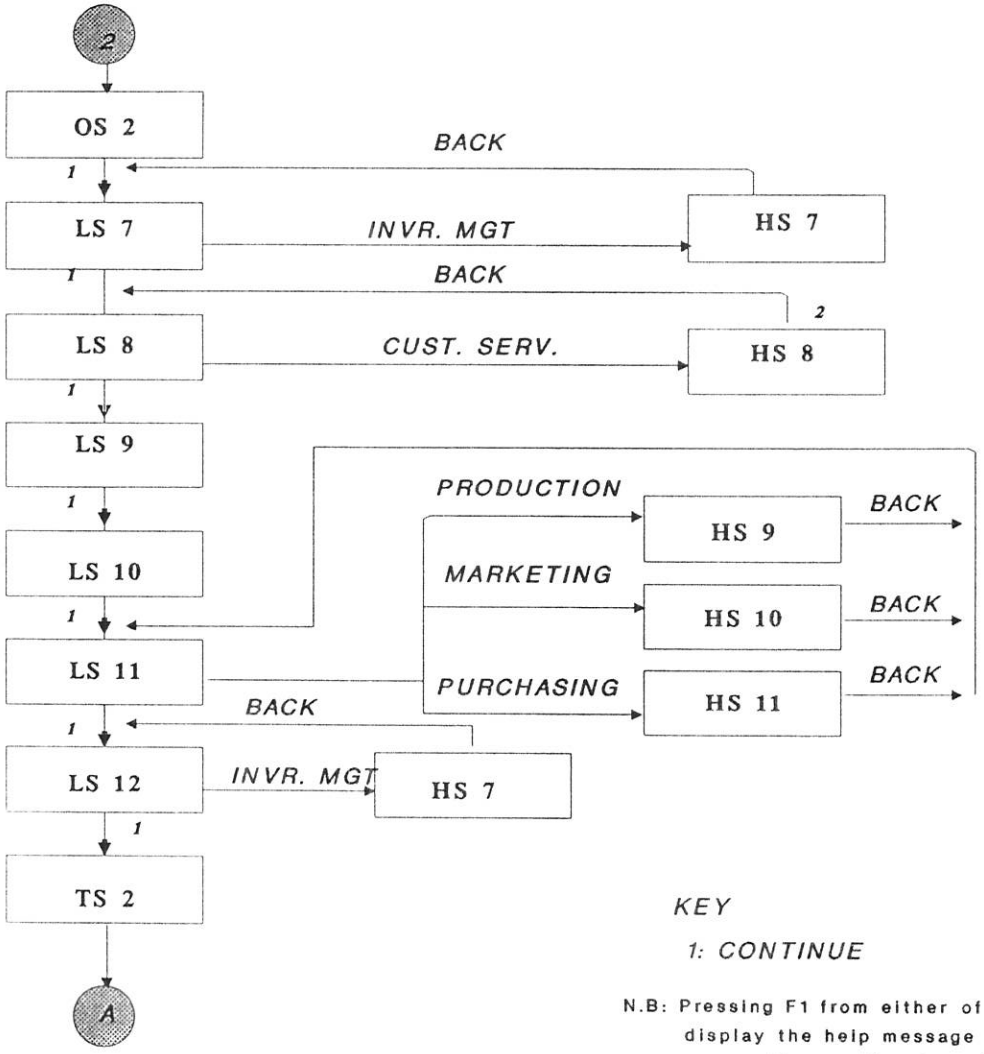
Fig.17: CONTINUED

Key:

1: Continue

N.B.

Pressing F1 from either of these screens display the help message available for that screen
Esc enables to quit from the help screen



KEY

1: CONTINUE

N.B: Pressing F1 from either of the screens display the help message available for that screen. Esc enables to quit from the help screen

Fig. 17 CONTINUED

4.3.4 Response Analysis

Analysis of the responses entered by the user has an influence on the general flow of the screens. Such analysis is generally implemented through the usage of a response analysis form.

A response analysis form is used to specify the response that one which to look for at any point and the actions that are going to be taken if a match (based on the model implemented) is found.

The following response analysis form describes the possible responses and the corresponding actions for the exercises and interaction points used in the courseware.

TABLE 5: **RESPONSE ANALYSIS FORM TABLE**

SCREEN NO.	IF RESPONSE	GOTO	GIVE THIS FEED BACK	THEN GOTO
LS 1	INCLUDES:INVENTORY/ STOCK ELSE	LS 2	 You could have entered as a correct answer inventory or stock	 LS 2
HS 16.4A	INCLUDES: REGRESSION LINE/REGRESSION ELSE		Yes! you are right It seems that you forget the purpose. Least square method was used for calculating the regression line equation	HS16.5A HS16.5A
HS16.14A	IS: YC = a + b_x , $\sum y = na + \sum x(b)$ $xy = \sum x(a) + \sum x^2($ ELSE		That is very good! May be you miss to enter the formulas correctly. You have always to keep this formulas in mind.	HS 16.15A HS16.7A
HS16.15A	IS: YES OR Y ELSE	16.13A 16.16A		
HS16.16A	IS:36,84500,377000, 204 ELSE	16.17A	 May be you miss/include some numbers in your computation. The correct answer is 36,84500,377000,204	 HS 16.17A
HS16.17A	IS= 84500= 8a + 36b 377000= 36a + 204b 4.5 ELSE	HS 16.18A	 The correct way to enter the equation is 84500= 8a + 36b 377000= 36a + 204b and the multiplier is 4.5	 HS 16.18A

HS16.19A	IS: -19.35, 10581.87 ELSE	HS 16.20A	The correct values are -19.35 and 10581.87	HS 16.20A
HS16.21A	IS: 9,10,11,12,13 ELSE	HS 16.22A	What are you doing ! This is just a simple counting starting from 9 to 12	HS 16.21A
HS16.22A	IS: YES OR Y ELSE	16.13A 16.1B		
HS16.10B	IS: YES OR Y ELSE	16.6B		HS 16.11B
HS16.12B	IS: 8 ELSE	HS 16.13B	Come on ! Think properly your base will be the largest month	HS 16.12B
HS16.13B	IS: 120, 135 150, 175, 160 AND 130,145,120, 135 150, 175, 160 ELSE		That is right do not forget to leave the oldest and take the nearest data for the subsequent months What is wrong with you! This doesn't require any computation just look the data very carefully	HS 16.14B HS 16,13B
HS16.14B	IS 148 7 145.00 10.00 155 15 148.57 21.43 162 23 152.14 32.86 169 26 161.00 33.57 173 2 170.00 5.00 176 19 173.57 21.43 ELSE		THAT IS VERY GREAT !!! Try to check your computation and rounding is to two decimal places	HS 16.15B HS 16.14B

HS16.15B	IS: 184, 176.43 ELSE	HS 16.17B	Just take 5 and 7 data sum them and divide the sum by 5 and 7	HS 16.15B
HS16.17B	IS : 15.33 AND 20.72 ELSE	HS 16.18B	Check your number of observation and absolute deviation	HS 16.17b
HS16.18B	IS: 5 OR 5 WEEKS OR WEEK 5 ELSE		YOU FINISHED THE MODULES GOOD LUCK IN THE TEST DON'T TELL ME	HS 16.18B

4.3.5 Program Structure

The prototype courseware uses units and objects of the units provided by the Turbo Vision library.Units are the basic elements of modular programming in Borland Pascal. They are used to create libraries and to divide large programs into logically related modules. A typical unit in Pascal consists of the following parts:

unit heading
interface part
implementation part
initialization part

The **unit heading** specifies the unit name which is used when referring to the unit in a **uses** clause with in the body of the program.**Interface part** declares constants, types, variables, procedures and functions that are available to the users of the unit. The procedures and

functions are listed in this interface part only as headings. **The implementation part** defines the bodies of all the public procedures and functions. In addition it declares constants, types, variables, procedures and functions that are private and not available to the users of the unit. **Initialization part** is the last part of a unit. it consists of either :

. the reserved word **end** (no initialization code),or

. a statement part to be executed in order to initialize the unit

The general structure of a unit (i.e. its definition) looks like the following:

```
Unit name;
interface
        uses
        const
        type
        var
        procedure
        function
implementation
        uses
        label
        const
        type
        var
        procedure
        begin
          statement;
          statement;
        end.
```

Table 6 describes the objects used in the current prototype development and the units which categorize these objects.

TABLE 6: **DESCRIPTION OF THE OBJECT AND UNIT TYPES USED IN THE INVENTORY MANAGEMENT PROTOTYPE COURSEWARE**

OBJECT TYPE	UNIT TYPE
TApplication	**App Unit**
TBackGround	**App Unit**
TDeskTop	**App Unit**
TDialog	**Dialogs Unit**
TInputLine	**Dialogs Unit**
TStaticText	**Dialogs Unit**
TStatusLine	**Menus Unit**
TMenuBar	**Menus Unit**
TResourceFile	**OBjects Unit**
TRect	**Objects Unit**

Source: Borland International. 1992. Turbo Vision: version 2.0 programming guide. Ireland: Borland International Inc:94.

Figure 18 and 19 give the hierarchical and inheritance diagram of the objects used by the prototype courseware from the turbo vision units and objects library.

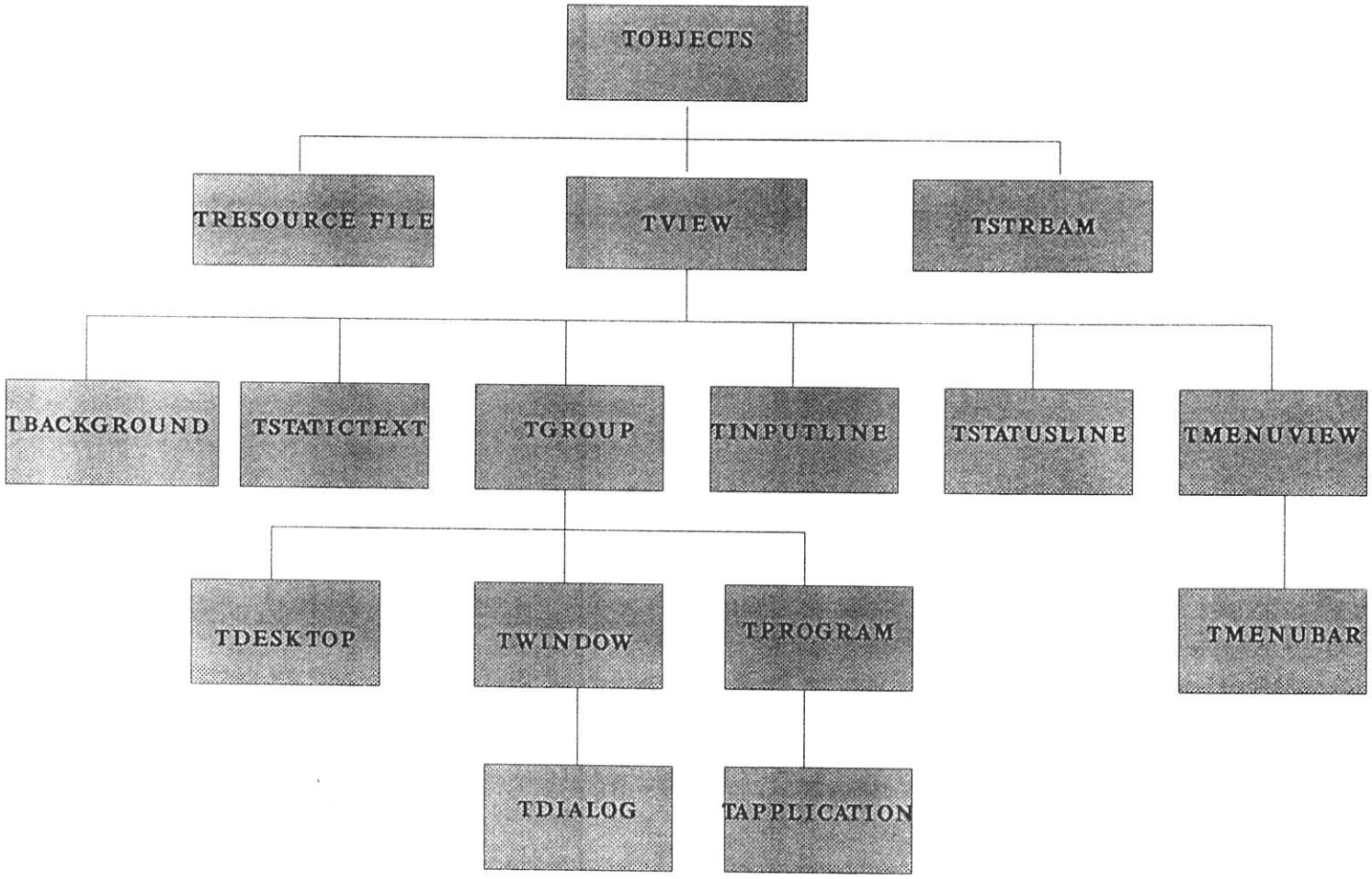


Fig.18:Hierarchical diagram of the objects used for the prototype courseware

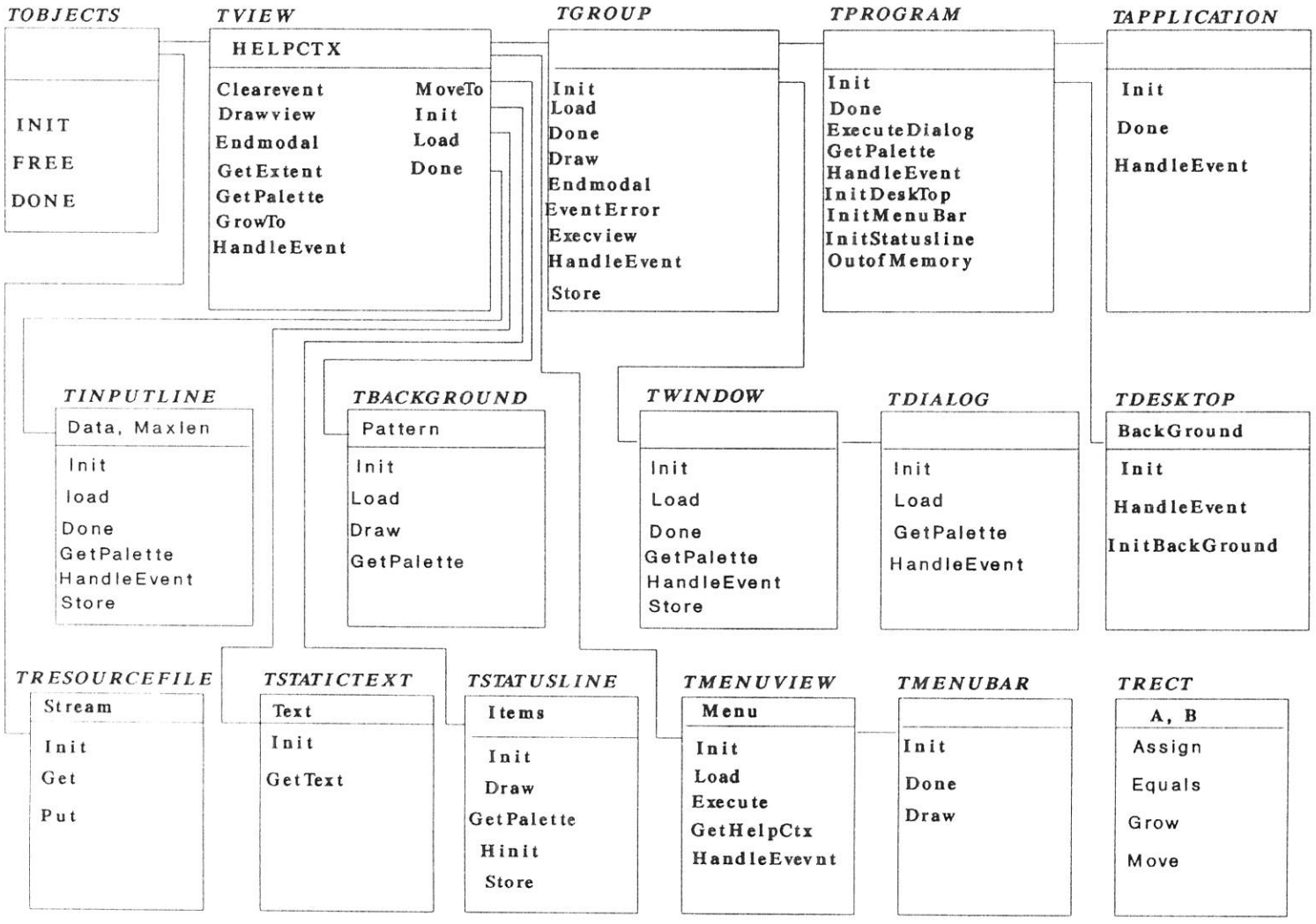


Fig.19: Inheritance diagram of the objects used

The flowchart in figure 20 to 28 describes the general program structure of the prototype courseware. A high level flowchart is presented showing the flow of control among the different testing session of the courseware. The procedures of the main objects used in the programme are presented in the pseudocode following the flowchart.

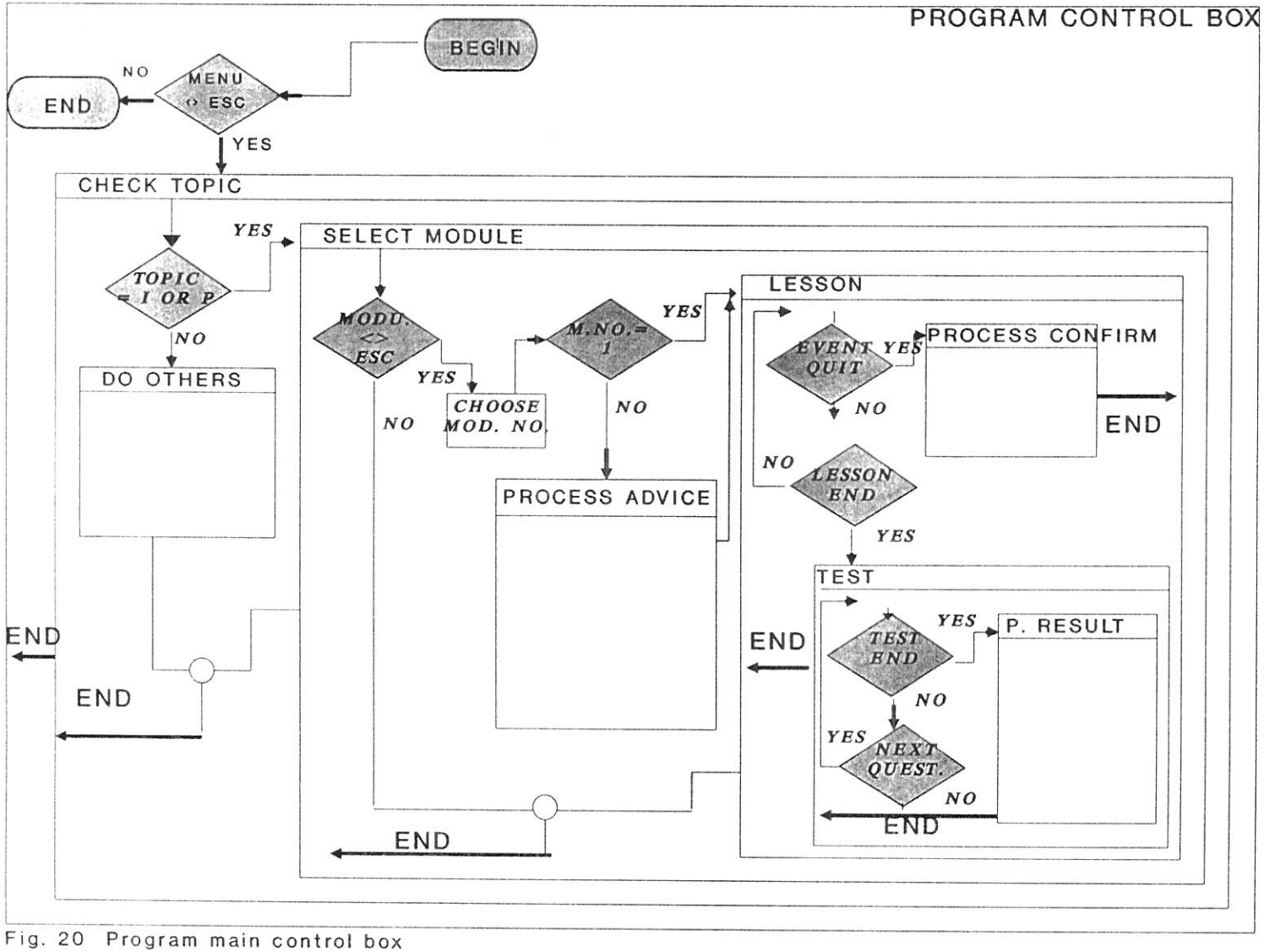


Fig. 20 Program main control box

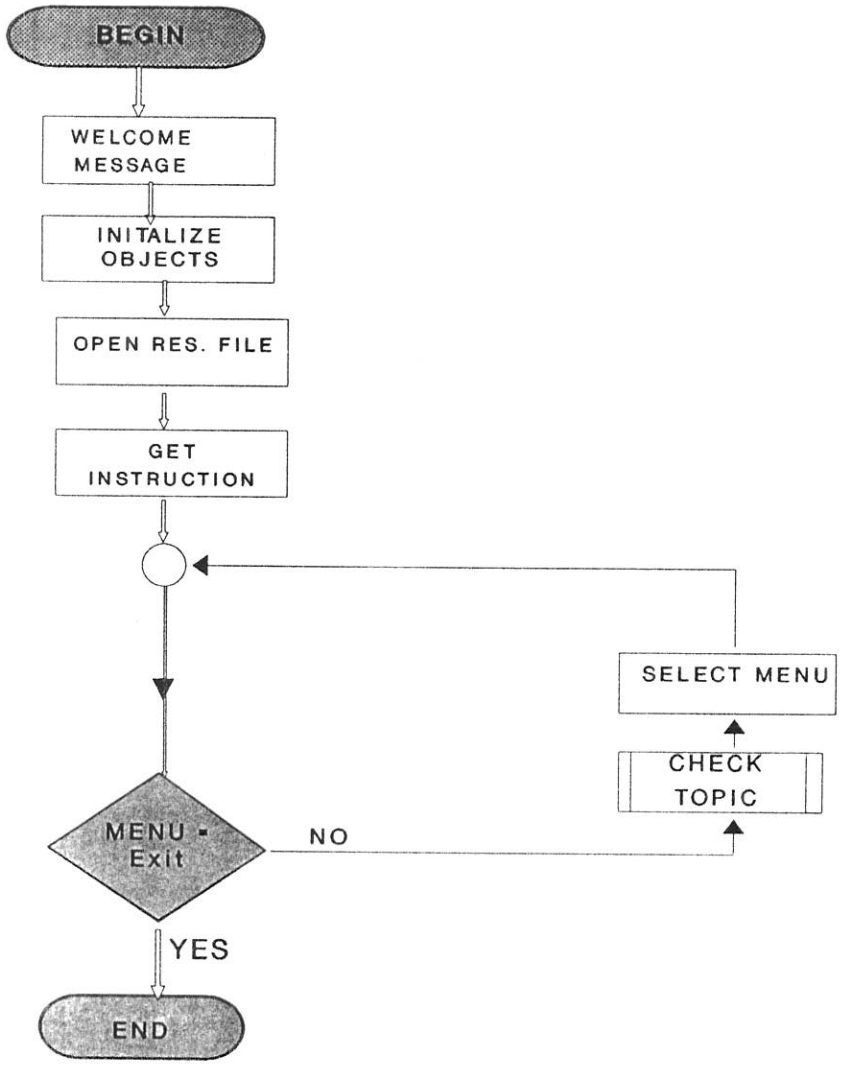


Fig.21: PROGRAM FLOWCHART- MAIN

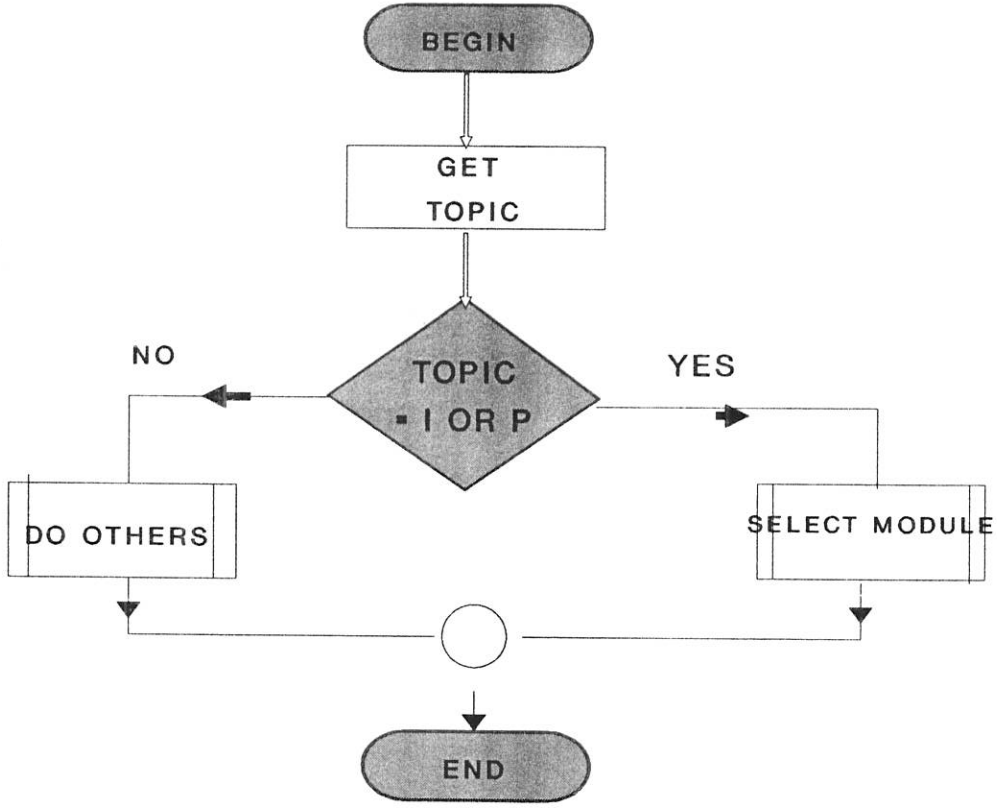


Fig.22: PROGRAM FLOWCHART -CHECK TOPIC PROCEDURE

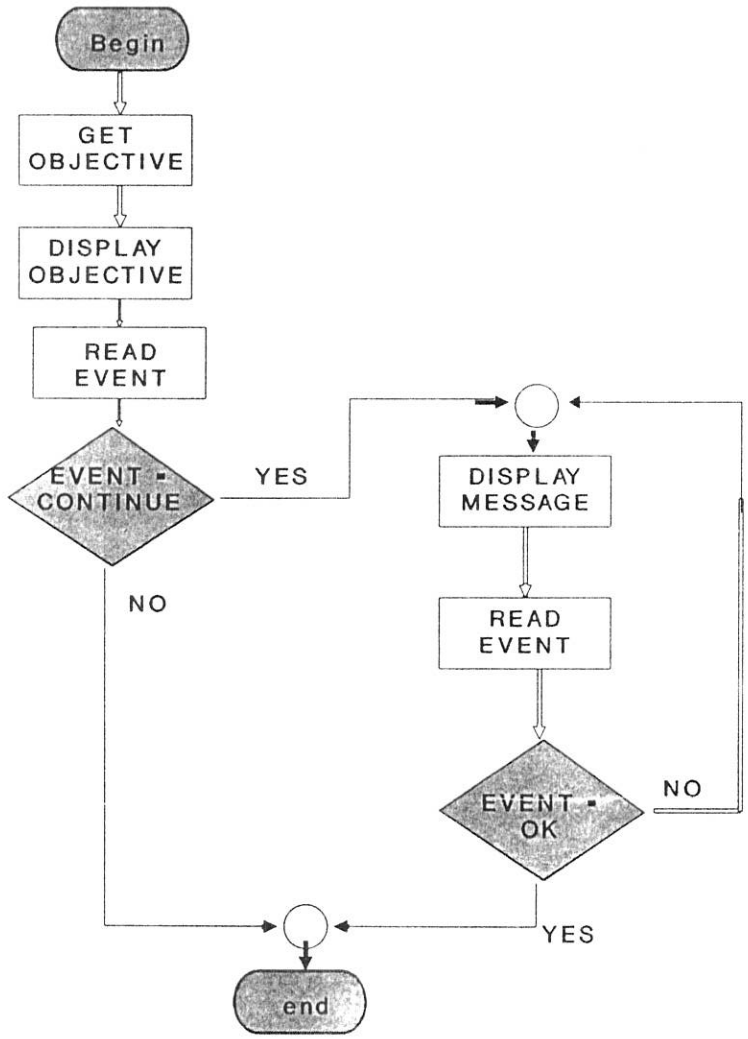


Fig.23: program flowchart- procedure do others

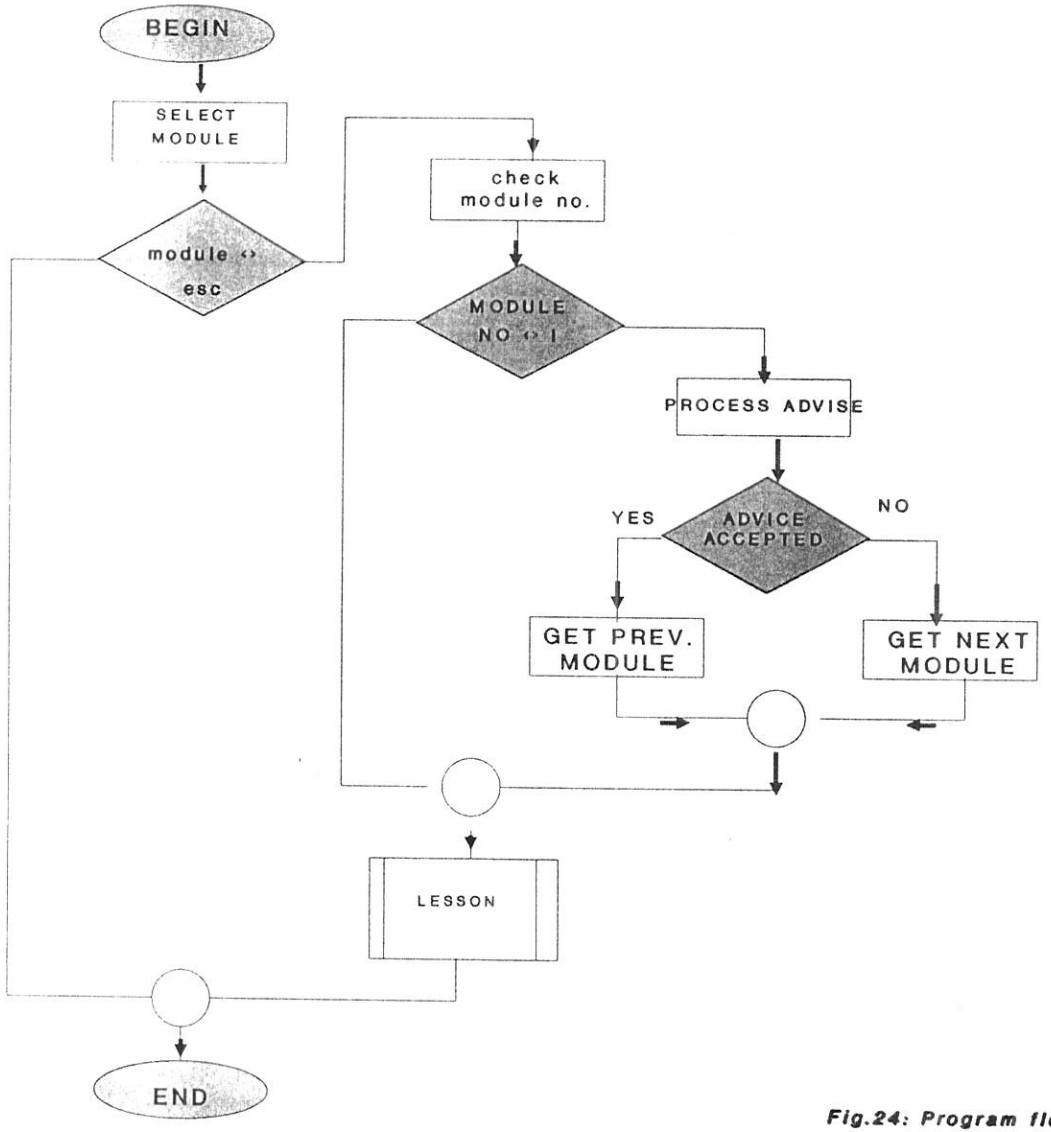


Fig.24: Program flowchart -Module Select

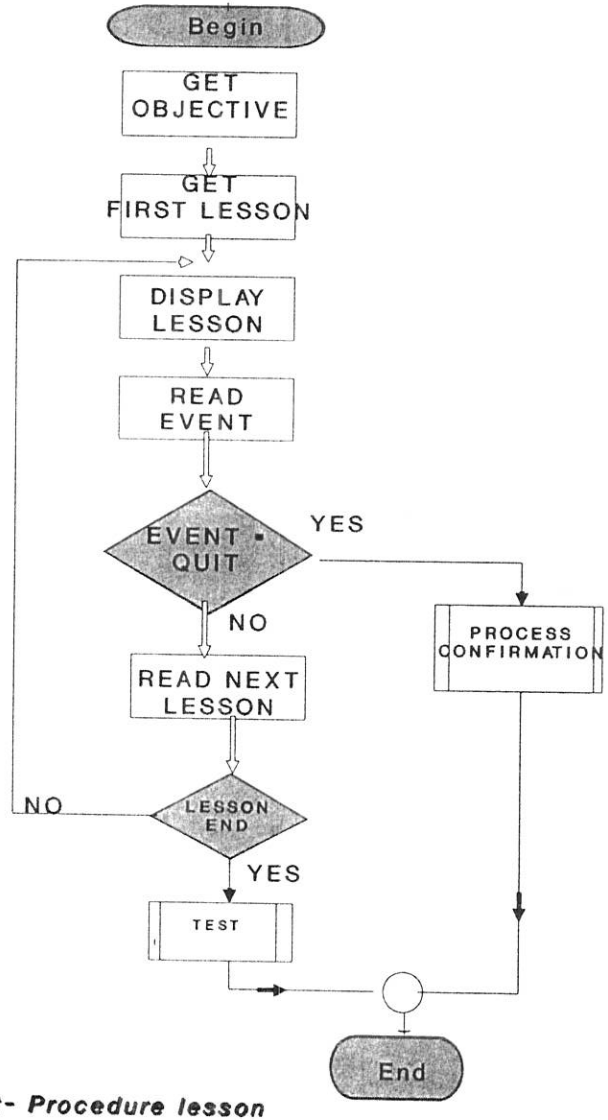


Fig.25: Program flowchart- Procedure lesson

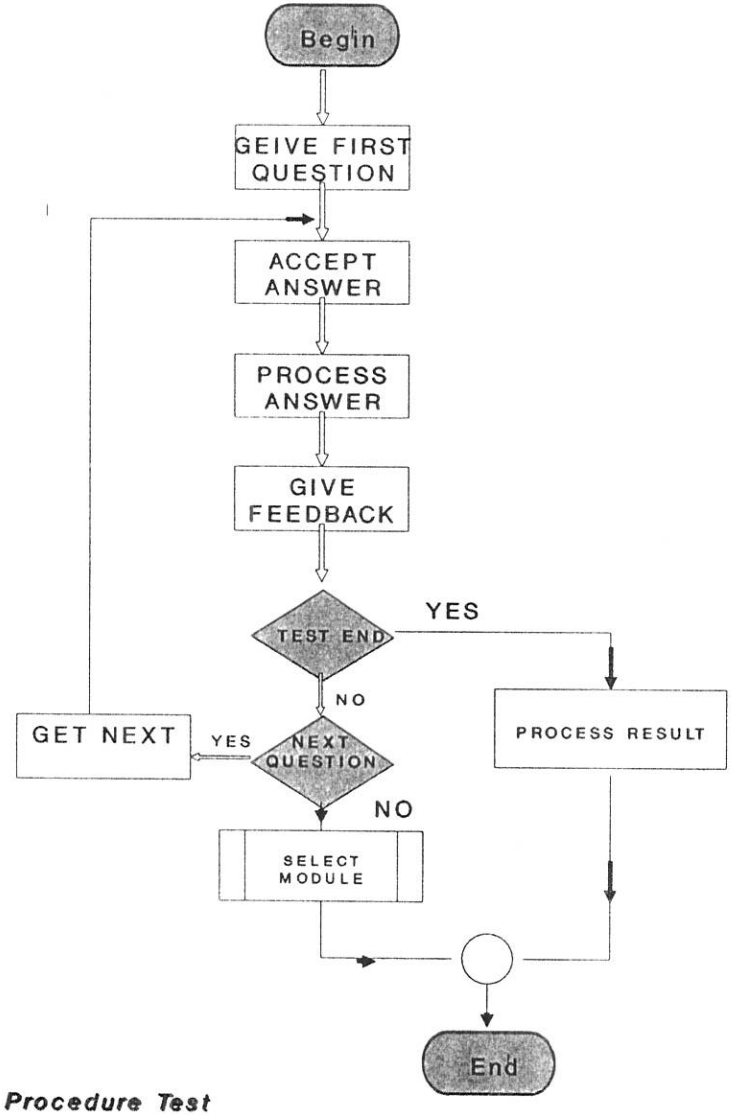


Fig.26: Program flowchart- Procedure Test

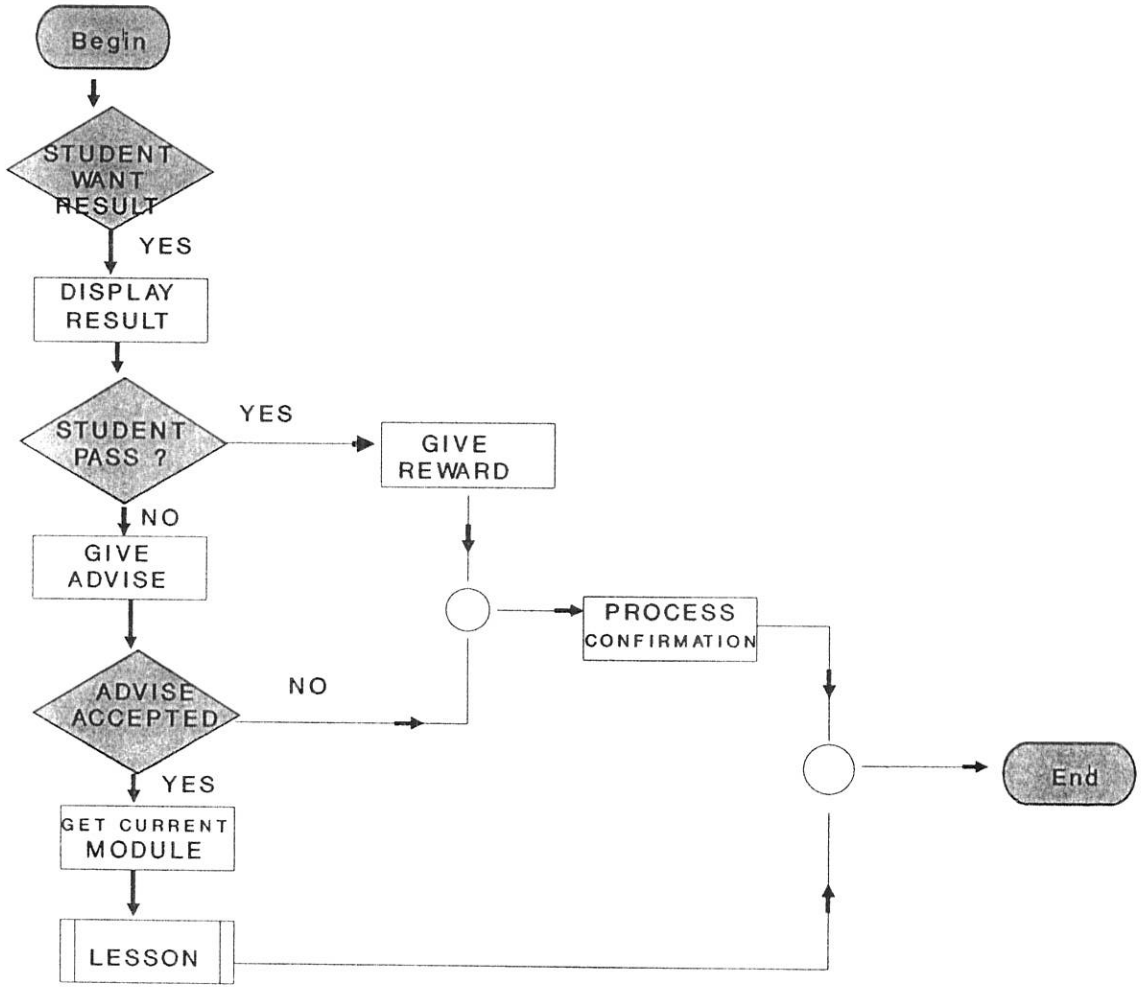


Fig.27: Program flowchart -Procedure process result

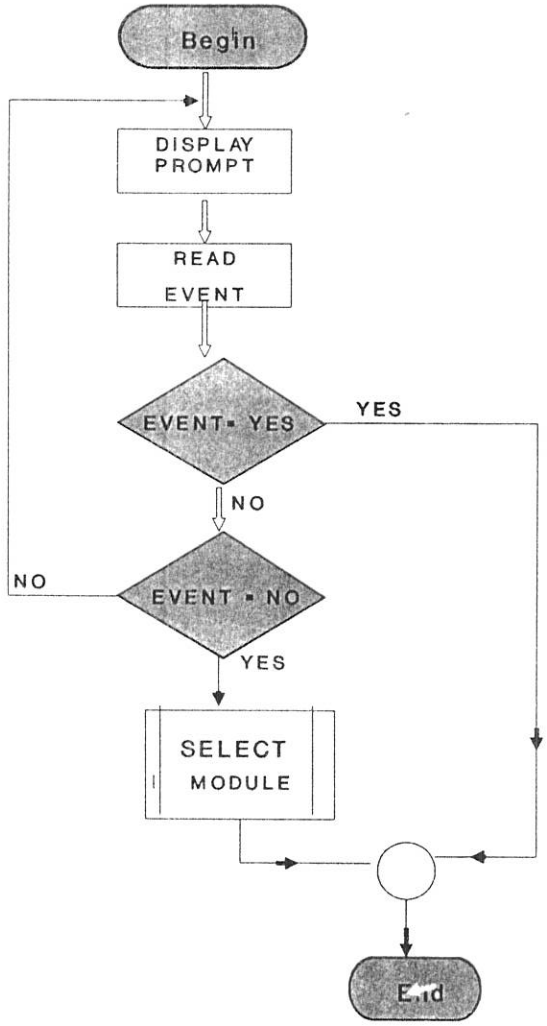


Fig.28: Program flowchart- procedure confirmation

PSEUDOCODE

Following is a description and pseudocode (where possible) of the main objects used for the current task

TAPPLICATION

The first step in writing an application program using Turbo Vision is to drive a new application object to which a programmer add his changes. As indicated in fig 19 , in the current work, the **INIT, DONE and HANDLEEVENT** methods of the **TAPPLICATION** object are in use. Following is the Pseudocode for Tapplication and the methods.

```
. With the type Declaration declare all the procedures and
 functions that inherit the application object and build
 the procedures and functions
 TYPE
        <POINTER VAR> =^<VAR>;
        <VAR> = Object(Tapplication);
        Constructor INIT;
        Procedure HandleEvent;
        Procedure InitMenuBar;
        Procedure InitStatusline;
        Procedure InitDeskTop;
       Procedure EventError;
        Destructor done;
        .....
END

Constructor <VAR>.INIT;
      define a constant variable and assign to it the resource file  declare other variables
       Register the objects
```

Initialize the resource file

if error encountered in the resource file Do halt

Else display the introductory message

Construct the Menu bar by calling Procedure <VAR>.InitMenuBar

Construct the status line by calling Procedure <VAR>.InitStatusLine

construct the desktop by calling Procedure <VAR>.InitDeskTop

interact with the user

Destructor <VAR>. done

Inherit the done method from Tprogram

Dispose the application object by using the inherited done method

Dispose the resource file

Procedure <VAR>. HandleEvent

Declare all the variables

inherit the handleevent from Tprogram (This will enable to handle all APP unit events)

Check Whether the generated event is a command

If it is a command then

DO the action(procedures, functions) initiated by that command

when through, clear the event by calling the TVIEW clear event

Else Generate event error by calling Procedure <var>.EventError

TBACKGROUND

Tbackground is a simple view consisting of a uniformly patterned rectangle. It is usually owned by a TDesktop. The current prototype courseware uses The **INIT, DRAW**, and **GETPALETTE** methods of this object. Following is the pseudocde

. With the type Declaration declare the draw procedure, init constructor and getpalette function as a decedents of the object Tbackground and build the procedure, constructor and function

```
TYPE
        <POINTER VAR> =^<VAR>;
        <VAR> = Object(Tbackground);
        Constructor Init
        Procedure Draw
        Function GetPalette
End

Constructor <VAR>.Init
    Declare variables
        Inherit from TVIEW the bounds
        Create (modify) the background object by calling the Procedure InitBackground
        Set the growmode and the pattern

Procedure <VAR>.Draw
        Declare variables
        Modify (optional) the colour
        Fill the background with the rectangle and the pattern

Function <VAR>.GetPalette
        Get the colour constant
        return the pointer to the default background Palette
```

Tdesktop

The desktop owns the background view upon which the application's windows and other views appear. It represents the desktop area of the screen between the top menu bar and bottom status line. The method used for the current work from this object is

INITBACKGROUND.

. With the type declaration, declare an InitBackground procedure within the Desktop

```
TYPE
        <POINTER VAR> =^<VAR>;
        <VAR> = Object(Tdesktop);
        Procedure InitBackGRound;
End

Procedure <VAR>.InitBackGround
        Declare variables
        Get the Trect of the background
        Assign a new view to the background
```

TDIALOG

Tdialog is a specialized descendant of Twindow. It is use is mainly intended for modal use and for holding controls. From this object Init , Load and Handleevent methods are used in the prototype inventory management courseware.

With the type declaration, declare constructor INIT, constructor LOAD and procedure HANDLEEVENT as a descendants of object TDIALOG.

```
TYPE
        <POINTER VAR> =^<VAR>;
        <VAR> = Object(Tdialog);
        Constructor Init
        Procedure HandleEvent;
        Constructor Load
End
```

Constructor <VAR>.Init

Define Variables

Inherit The init constructor from Twindow (this will create a dialog box with the given size and title)

Construct a dialog box and insert control buttons by calling procedure addbut

Constructor <VAR>.Load

Inherit the load constructor from Twindow (this reads a dialog box object from the stream (resource)

read the records in the dialog box

Procedure <VAR>. HandleEvent

Declare all the variables

inherit the handleevent from Twindow (This will enable to handle all Dialog unit events)

Check Whether the generated event is a command

If it is a command then

DO the action(procedures, functions) initiated by that command

when through,

End the modal state of the dialog box

clear the event by calling the Twindow clearevent method

Else Generate event error by calling Procedure <var>.EventError

TSTATUSLINE

This object is specialized view, usually displayed at the bottom of the screen. Typical status line displays are lists of available hot keys, displays of available memory time of day current edit modes and hints for users. The status line is normally owned by the Tapplication group. The prototype courseware uses the **HINT** method of the statusline object.

With the type declaration, declare Function HINT as a descendants of object TSTATUSLINE.

```
TYPE
        <POINTER VAR> =^<VAR>;
        <VAR> = Object(Tstatusline);
        Function hint
End;
Function <VAR>.Hint
        With case statement define the hint context like
        CASE Ahelpctx of
           hcnocontext:hint:= '<string>'
           hc.....:hint:= '<string>'
        end
```

TRESOURCEFILE

Tresourcefile implements a stream that can be indexed by key strings. When objects are stored in a resourcefile, using Tresourcefile.Put, a key string, which identifies the object, is also supplied. The methods used currently are **INIT, GET and PUT**.

INIT enables to initialize the resource file

PUT stores objects in to a resource file

GET calls objects from the resource file

TINPUTLINE

A Tinputline object provides a basic input line string editor. It handles Keyboard input and mouse clicks and drags for block markings and a variety of line editing functions. This object has the following fields: **Curpos** (which gives the current cursor position) **Firstpos** (index

to the first displayed character) **MaxLen** (Maximum Length allowed for the string **Validator** (points to the data validator object associated with the input line or nil if the input line has no validator). The Inheritance diagram (fig. 19) gives the methods used from this object.

CHAPTER FIVE

IMPLEMENTATION

Implementation, as used in the ISD model, refers to the installation and start-up of the system, initial and on going training of all system users and revision of the courseware. This account, however, is limited to the discussion of the result obtained in attempting (applying) the ISD model design specifications presented in the preceding chapter. In particular, the description/demonstration of the prototype Inventory Management Courseware (IMC) developed by applying the ISD model to the inventory management course under offering at AACC. The chapter concludes with a general remarks on the ISD model and its suitability for developing a courseware (on the basis of the experience gained in the current work) and recommendations for completing the prototype courseware to a fully operational system.

5.1. IMC : THE PROTOTYPE COURSEWARE

5.1.1 Naming

IMC is an acronym to the prototype Inventory Management Courseware developed in this study by applying the ISD model to the Inventory Management course Under offering at AACC.

5.1.2 IMC System Requirement/Environment

The program is developed using Turbo Vision 2.0 programming language - an object oriented extension of Turbo Pascal for developing window based applications- running the program, however, does not require the pascal compiler.

While the hardware environment used in developing IMC is: 386 processor, 4 MB RAM, DOS 6.0 and 20 MB Hard Disk, IMC has been tested and found to successfully run on a 286 processor, with 640 KB RAM and 20 MB hard disk. Better performance has been observed, however, with higher capacity systems.

5.1.3 IMC Program Structure

The general structure of IMC consists of one main body, two resource file programs (corres and corres1 whose executable version is Inventor.tvr and Invwind.tvr), and a number of units.

One of the IMC units implemented in the current work (**Crwrcons**), for example, consists of the definition/ declaration of the various objects described in chapter four, the **event handler** method of the decedents of **TDIALOG** (an object inherited for storing the courseware lessons), object identifier constants, etc. Figure 29 below shows an extract of the implementation of this unit.

```
{$O+,F+}
        unit Crwrcons;
        interface
        uses App, Objects, cwrgcons, Menus,Msgbox, Drivers, views,Dialogs,Crt;
        TYPE
        ...
        CrWrScrn = ^TCrWrScrn;
        Tcrwrscrn = Object(Tdialog)
      ....
          Procedure HandleEvent(var event: tevent);virtual;
      Procedure addbut;
      procedure store(var s: Tstream);
      constructor load(var s: Tstream);
        End;
        .....
        const
                  RCrWrScrn: TStreamRec = (
                  ObjType: 1160;
                  VmtLink: Ofs(TypeOf(Tcrwrscrn)^);
                  Load:  @Tcrwrscrn.load;
                  Store: @Tcrwrscrn.Store);
       ....
       var
        twin,twin1:pcrinvtext;
        win,win1: Pinvtext;
       ...
       Implementation
       ...
       Procedure Tcrwrscrn.HandleEvent(var event: tevent);
       var
       begin
                  inherited handleevent(event);
                  if Event.What=evcommand then
       begin
                  CASE Event.Command OF
       cmgo:begin
                endmodal(cmgo);
                clearevent(event);
           end;
    end;end;end;
       ....
       End.
```

Fig. 29: An example of the implementation of a unit in IMC

IMC Resource files contain the different lessons, comments, instructions, menu items and statusline definitions of the courseware which are stored using The PUT method and a unique identifier. Such files are implemented in IMC system as shown in Figure 30 below.

```
Program CoursewareRes;
uses app,Objects, Validate,Editors,TUTCONST,Menus,Msgbox, Drivers, Views,cwrgcons,
   {$M 65440, 0, 650000}
TYPE
.....
var
 Corres: TResourceFile;
Procedure Createscreens;
Type
          PCrWrstr = ^Crwrstr;
var
        butt: Crwrstr; CWtextstr:Tcrwrtextstr;  oScreen:Crwrscrn;
...
  Begin
  R.assign(4,4,76,16);
Cwtextstr[1].r.assign(5,1,65,10);
Cwtextstr[1].crwr:=#3'MODULE 1:'#13#13+
  'At the end of the module you will be able to'#13#13+
  '1. Recognize the definition of inventories'#13+
  '2. List and classify the different types of inventories'#13+
  '3. List functions of inventories';
  Oscreen := New(crwrscrn, Init(R, 'INFORMATION',Cwtextstr,4,butt,2));
  oscreen^.helpctx:=3000;
  corres.Put(Oscreen, 'OS1');
  Dispose(Oscreen, Done);
...
  end;
  begin
  Crwrstrm:=new(phaltstream,Init('Inventory.Tvr',stcreate,1024));
  ..........
  corres.init(Crwrstrm);
  registertype(Rcrwrscrn) ;
  ........
  Createscreens;
  corres.done;
  end.
```

Fig. 30: An example of a resource file structure within the courseware

The IMC main program consists of mainly the application object, an object which is used to drive the current application. An extract from the IMC main program is shown below.

```
Program Courseware;
uses Test,Jolly,Dos,Cinvhlp,Cinvts,App,Cwrfront,Objects, Menus, Msggbox, Drivers, Views, cwrgcons,
Var
        CORRES: Tresourcefile;
        corres1:Tresourcefile;
   ...
TYPE
 Pinvcowr= ^Tinvcowr;
 Tinvcowr = Object(TApplication)
  Constructor init;
  Procedure InitStatusLine; virtual;
  Procedure Initmenubar;virtual; Procedure InitDesktoP;virtual;
  Procedure Getevent (var Event: Tevent); virtual;
  Function  Getpalette:ppalette;virtual;
Procedure HandleEvent(var Event: Tevent); virtual;
      ...
Procedure Introduction;
Destructor Done;virtual;
End;
...
Procedure Tinvcowr.Introduction;
var
 screen1: crwrscrn;
begin
 screen1:= crwrscrn(corres.get('OS1'));
 Insertwindow(screen4);
 Dialogkey:= ExecuteDialog(screen1 , nil);
 if Dialogkey < > 102 then
.......
END;
var
 TInv: Tinvcowr;
begin
 Tinv.Init;
 Tinv.Run;
 Tinv.Done;
End.
```

FIG. 31: IMC MAIN PROGRAM STRUCTURE

When the program is activated the **TINV.INIT** method of the main program initializes all the objects and resourcefiles and construct the desktop, the menubar and the statusline and passes the control to the **TINV.RUN** method. The TINV.RUN method interacts with the user and waits until an event is generated from the keyboard, mouse,etc. When the event is generated, the **TINVCOWR.HANDLEEVENT** (Tapplication handle event) method handles the event and directs it to the required action. For example, if the event generated is a menu choice of Module 1, **PROCEDURE TINVCOWR.INTRODUCTION** will be activated. When the control gets the first instruction in this procedure which in the above case is to get a resource from the resourcefile **CORRES**, it reads the resource file and gets the required resource, assign it to a variable and draws it in the desktop using the **insertwindow** method.

The **TDIALOG** event handler now waits for the event to be generated and accordingly handles the event and passes control to the next instruction. When the program terminates, control passes to the **TINV.DONE** method which destructs/closes all the initialized variables, objects and resources.

5.1.4 Starting IMC

In order to run the prototype program the following files are required: **IMC.EXE, INVENTOR.TVR and INVWIND.TVR**. Once these files are in a disk (floppy or hard), to activate the program, type **IMC** at the DOS prompt and press **ENTER.** The following IMC front screen will appear. The major factor considered in the design of the IMC front screen is getting/catching the attention of the learner.

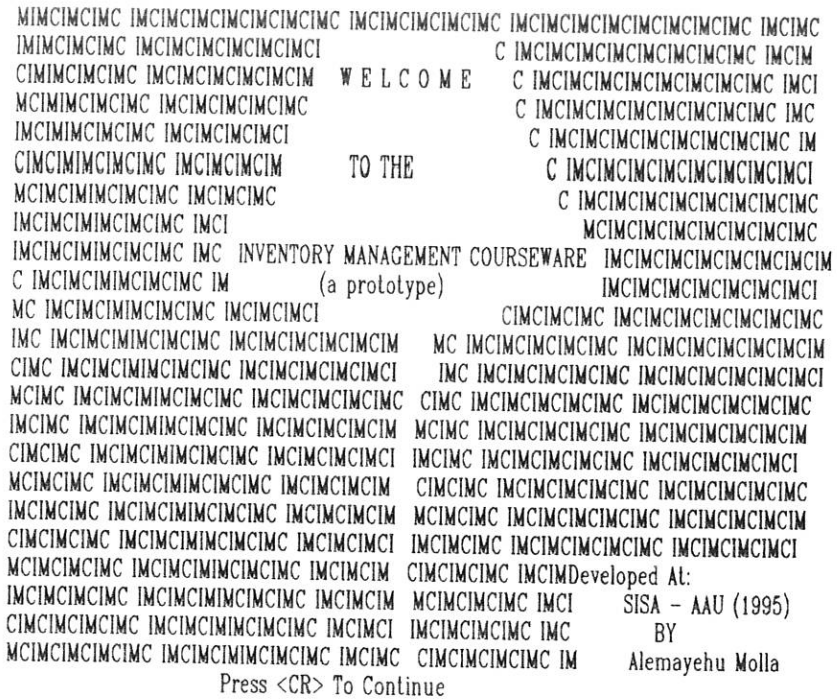


Fig. 32.1: IMC Front Screen

In the design of a courseware one of the issues to be considered is the nature of instruction. That is, instructions given to the learner during actual conduct of the course must be clear and enable the learner use the courseware at his own pace and without needing much help. Based on this consideration, in IMC, starting from the very beginning the user is provided with all the necessary instructions needed in using the courseware- at the bottom line of IMC front screen a prompt on what to do next,say to press enter. When **ENTER** key is pressed, the background object of the courseware with an instruction box in the middle appears.

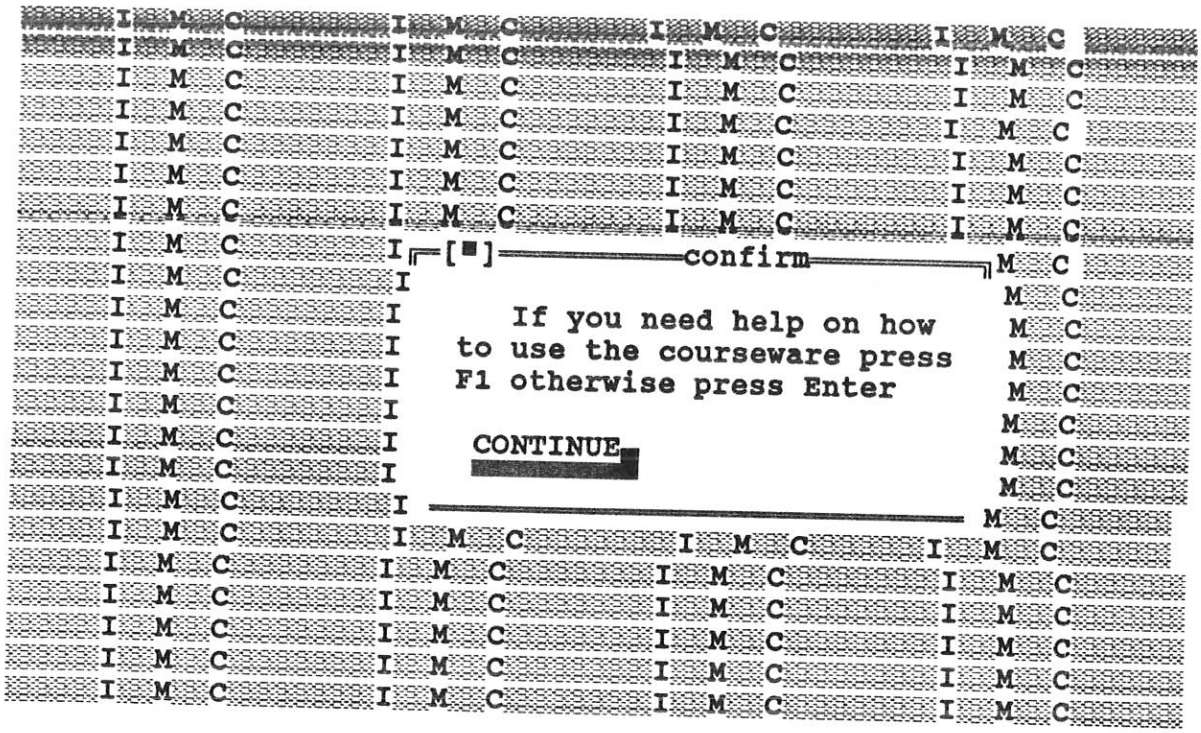


Fig. 32.2 : IMC User Information Screen

This instruction box is intended especially to help first time users by reducing and avoiding the problem in the use of the courseware. If the learner wants help on how to use the courseware and presses F1, the following information on how to use IMC appears.

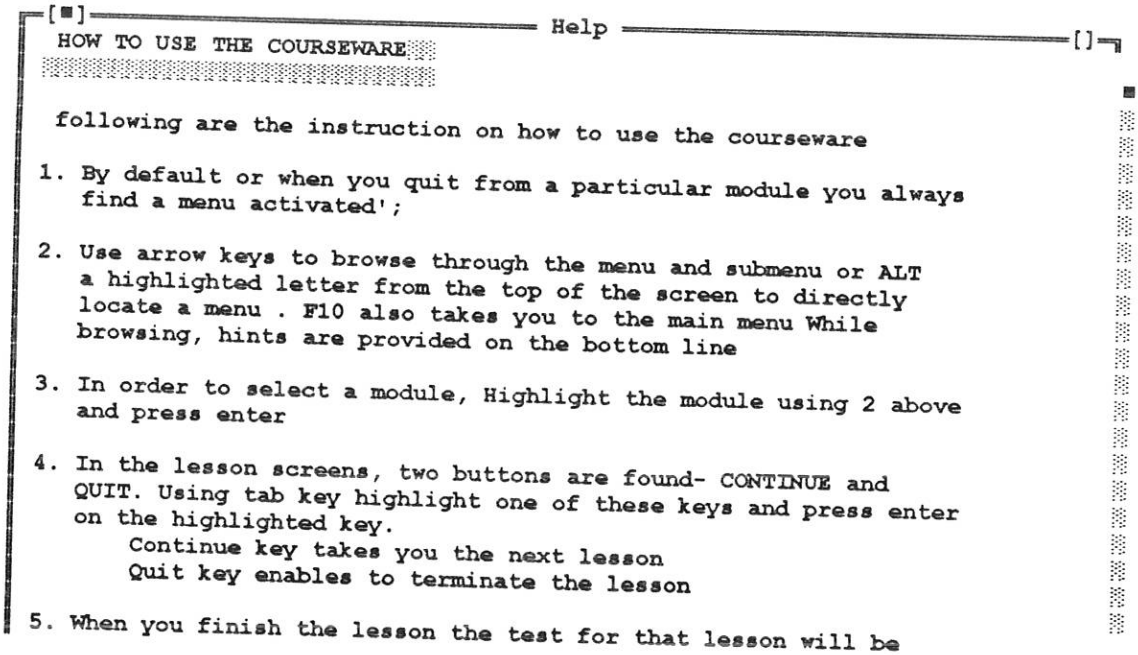


Fig. 32.3: IMC How to Use Information Screen

In addition, to make the necessary instructions available all the time to the user so that to avoid the need of memorizing, certain major IMC commands are made to appear on the screen throughout IMC session and are accessible to the user from any of the lesson by pressing F1 or their respective hot keys.

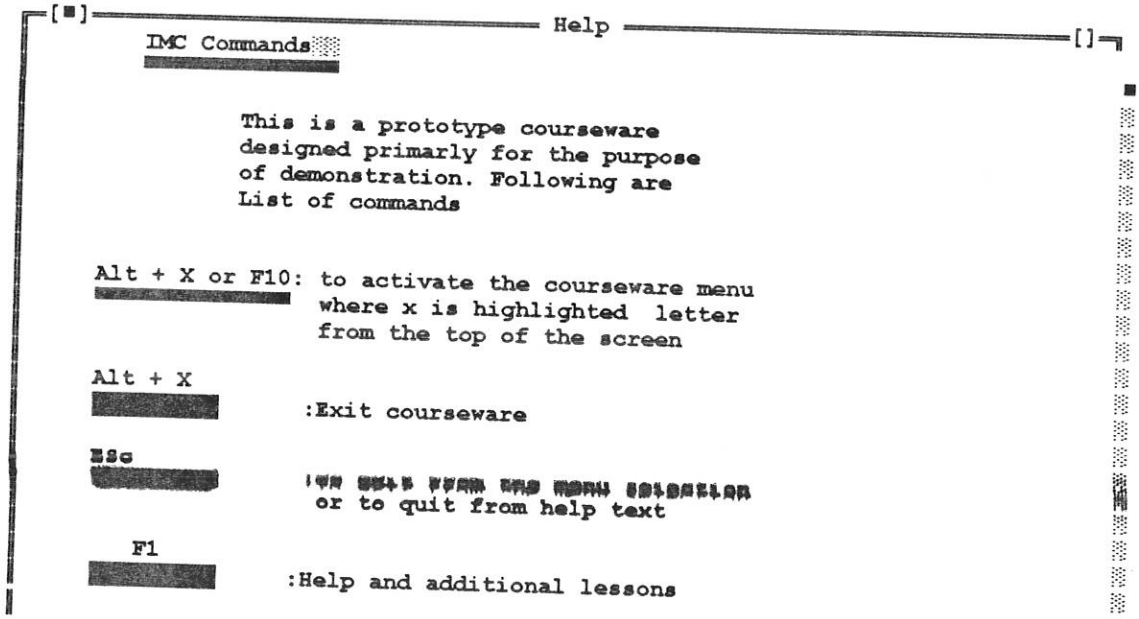


Fig. 32.4: IMC Major Commands

5.1.5 The IMC Menu

User friendliness (interactivity) is among the key/major characteristic features required in courseware. One of the methods adopted in the design of IMC to implement this feature is to make the software menu driven. Accordingly, IMC consists of a series of easy to understand structured menus of commands, options, lessons, tests and help.

In designing the IMC menus effort has also been made to maintain top down structure through the use of menu bars, dropdown menus, message/dialog boxes/ windows.etc., as appropriate to

the various modules and objectives. During IMC all these are made to run/play (or controlled) through one or a combination of the structured program control elements: sequence, selection, and repetition. Throughout, for instance, lessons could be repeated as desired by the user or suggested by IMC, selections are made from available options and sequences are followed as appropriate, for example, users need to pass through the explanation during IMC lesson before activating the test.

By default, the learner gets one of the options from the main menu activated when he/she

- finishes reading the how to use information or
- quit from a particular module

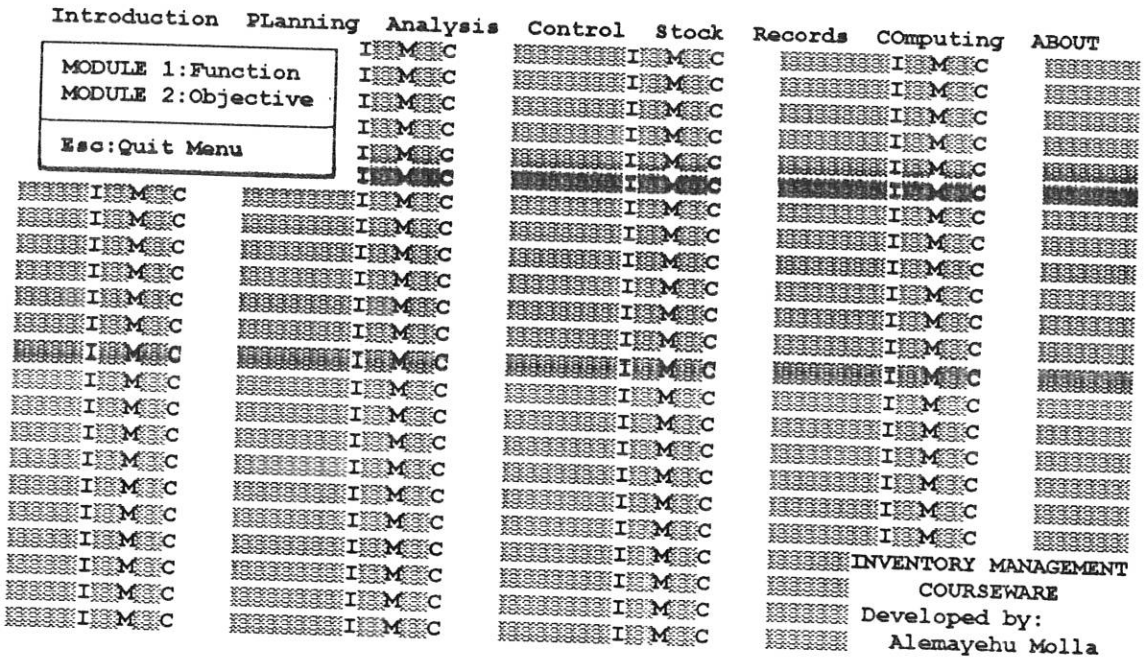


Fig **32.5: IMC Menu Structure**

To provide a learner additional options, the following commands are also used in activating the menu. IMC menu can be invoked by using either **ALT + highlighted letter** from the top of the screen or **F10** followed by ↓. What is more, using arrow keys, the user can browse through the

menu and sub menu of the courseware. A message that elaborates on the brief menu option description is always displayed on the status line as the option is highlighted. **Esc** key enables the user to quit from any module(sub menu) selection.

5.1.6 IMC lesson Session

One of the considerations stated in the design part of IMC was the branching rule. By considering the importance of user control taking into account experienced users that do not require complete guide and to make the courseware lessons serve the purpose of reference where by a student with a difficulty can consult a specific part of the courseware, the user, except provided with an advise on the selection of a lesson to be studied first, is given free and total control over the sequence of lessons he/she wants to study.

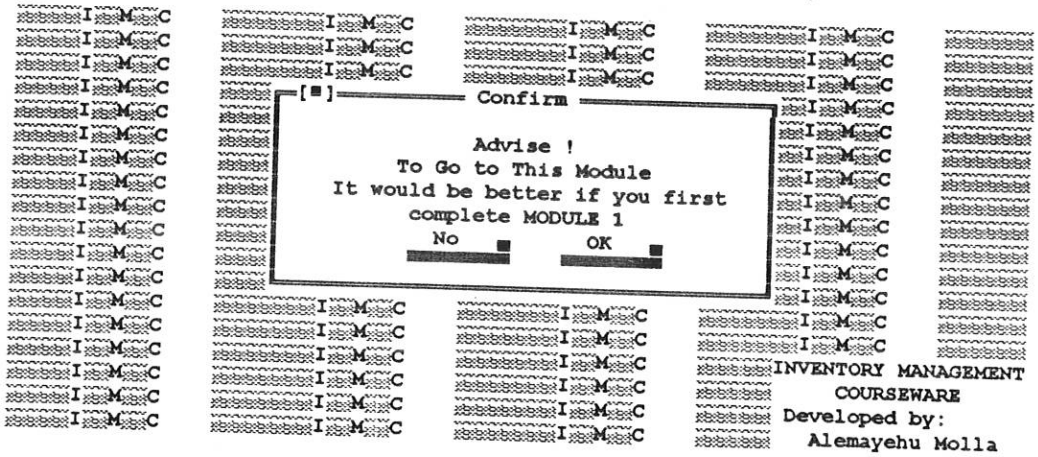


Fig 32.6: IMC lesson Selection Advise

With this feature, when a particular module is selected, the first screen to appear after such an advise, if any, is an objective screen. The objective screen describes what the student is expected to achieve at the end of the module.

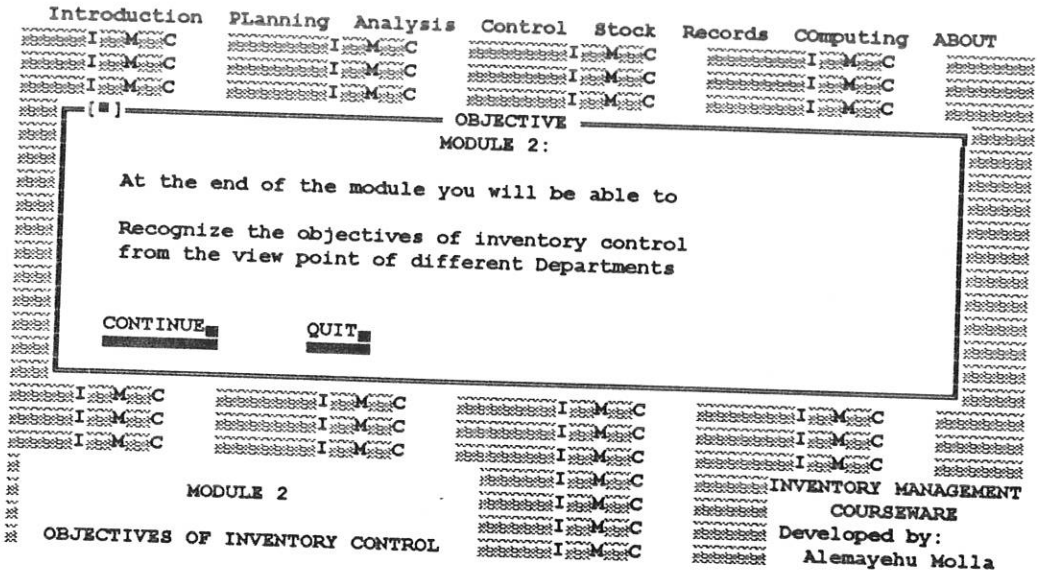


Fig 32.7: IMC Sample Objective Screen

Following this objective screen, the lesson screen appears. In designing the lesson and the objective screens, the screen design consideration, i.e., limiting the length of the text to be displayed to the learner to a manageable size is considered. Therefore, effort has been placed to limit the text displayed in the screen to the minimum possible- the screens are not loaded with information.

In addition, to provide the learner a mechanism of controlling his own pace (one of the advantages of using a courseware), all the IMC lesson and objective screens are attached with buttons (keys). Using the **<TAB>** key, for instance, one can highlight the desired option and pressing **<ENTER>** to execute the or activate same.

All the objective screens and most of the lesson screens are, usually, attached with two standard buttons **continue and quit** for going to the next lesson or quitting from the lesson respectively.

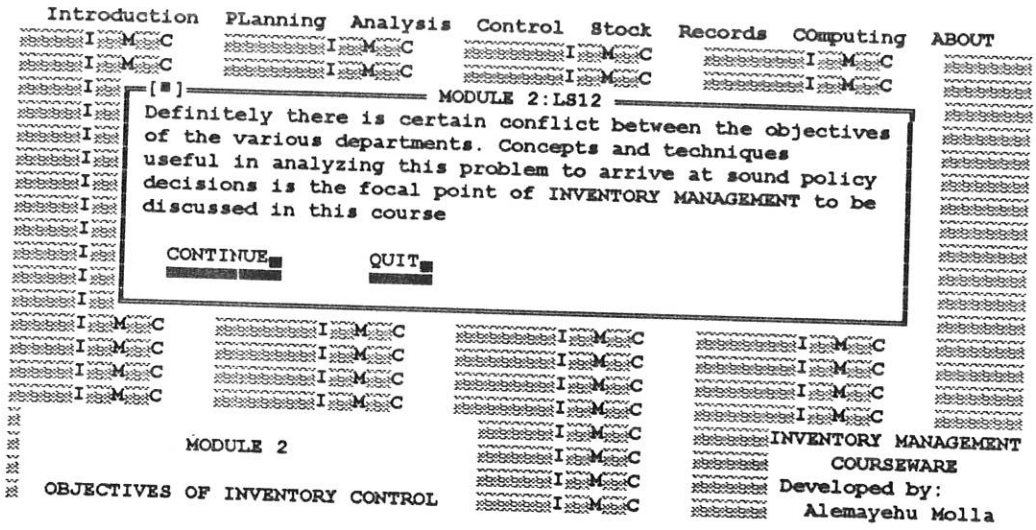


Fig 32.8: IMC Sample lesson Screen With Two Buttons

In addition to the two buttons, occasionally, some of the lesson screens have also a third and forth buttons for branching to a window(s) which contains an explanation of the concept/term selected at the button. Such design/structure is followed to give the learner a chance of skipping/branching those materials within a module which he/she already is familiar with.

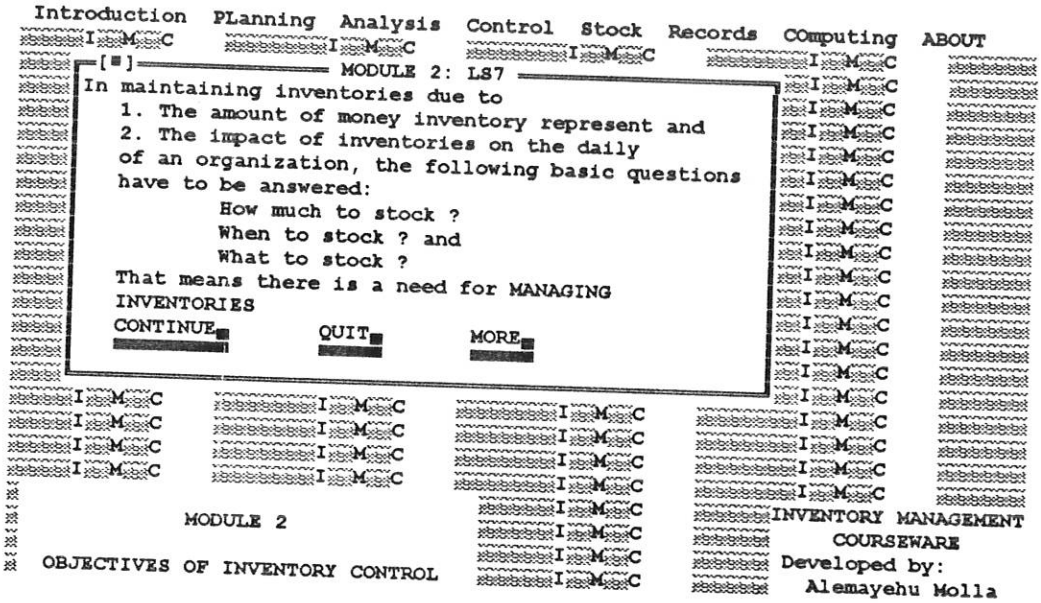


Fig 32.9: IMC Sample Lesson Screen With Three Buttons

If the learner selects the third or (forth) button, the resulting screen will look like the following.

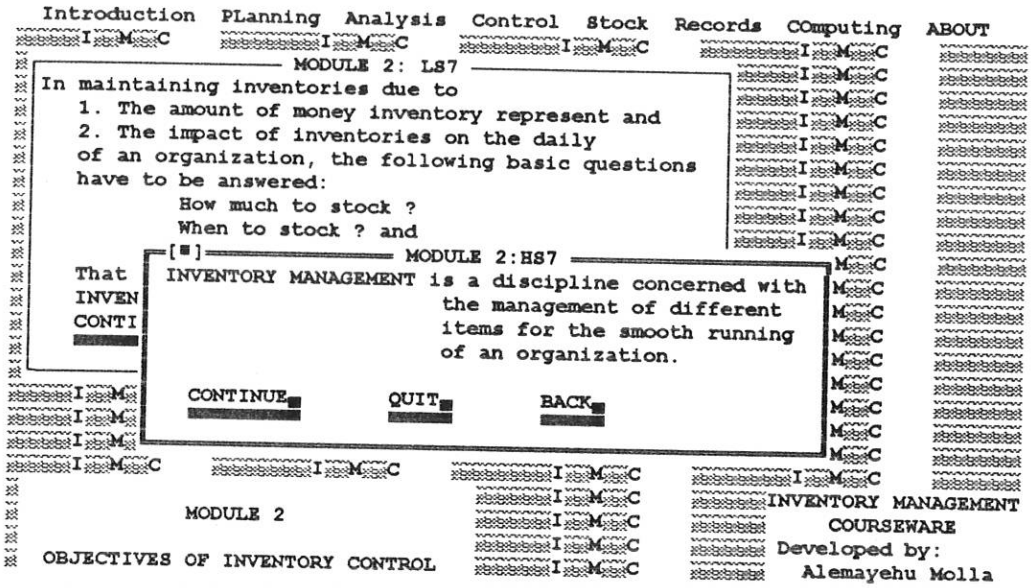


Fig 32.10: IMC Sample Highlighted Lesson Screen

The **back** button provided in the highlighted/open up windows enables the learner to go back to the calling window.

In addition, to add flexibility to the operation of the courseware, major points of the lessons are made accessible by pressing F1 from the lesson windows (in the form of help). This help text is structured in such a way that the first help screen the user gets when pressing F1 from the lesson windows links all the rest of the key words/concepts linked for that and preceding modules. The learner, therefore, can branch through the lessons by selecting the highlighted words in the help text. Inclusion of this help text provides the courseware a chance of being used as a reference material and also facilitates easy access to the courseware materials. For example, figure 33 shows the help text keywords link for module one.

And the following layout depicts a typical help text screen

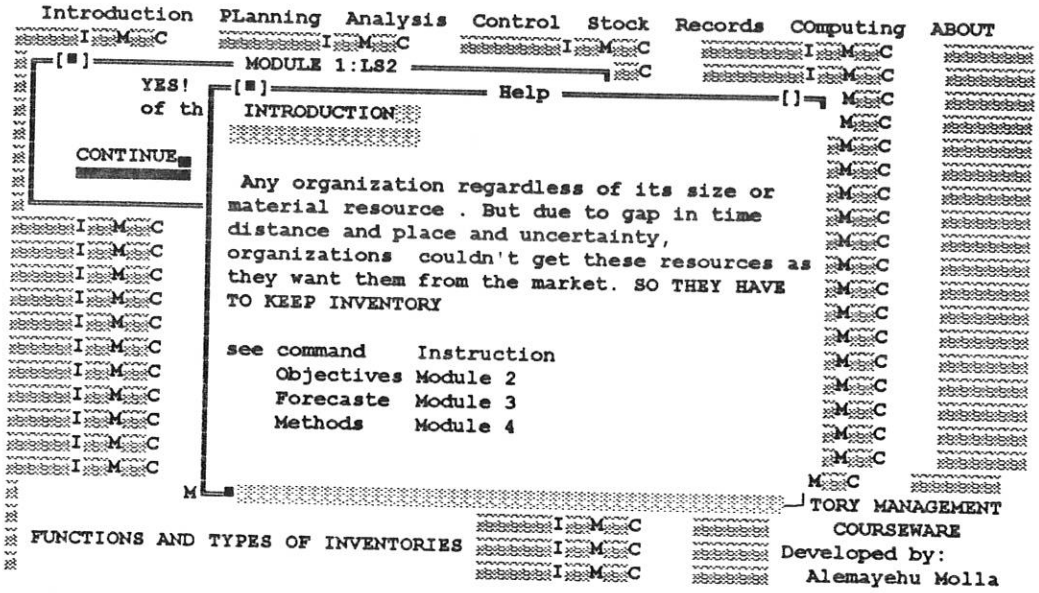


Fig 32.11: IMC Sample Help Text Lesson Screen

The **QUIT** button provided with the lesson window enables the user to quit from the courseware at any particular moment. Up on selecting this option, the learner will be prompted if he wants

to go to another module or not. This option provides the learner with a chance of selecting the modules from within the lesson without going to the menu.

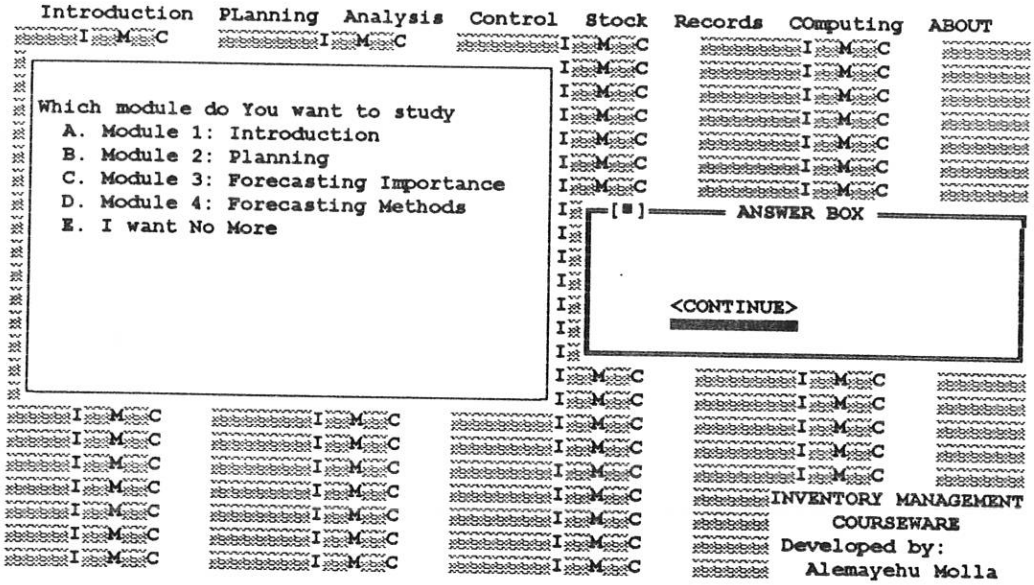


Fig 32.12: IMC Module Selection Option

5.1.7 IMC Testing Session

Upon completion of each module, the learner is given an option either to proceed to the test session or see the summary of the lesson by pressing F1.

While in the test session, instruction on how to complete the tests are available through F1. Based on the design specification, the test items adopted are alternate response, multiple choice, and open ended. The following is an example showing a multiple choice IMC test session

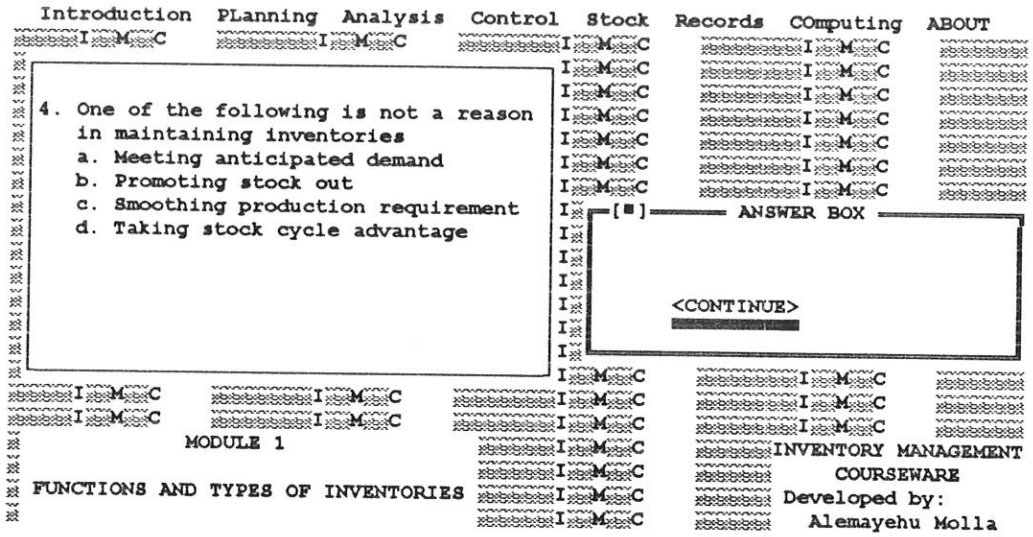


Fig 32.13: IMC Sample Test Screen

One of the advantages of courseware is the ability to provide immediate feedback to the user input. So each response of the user to the test items are evaluated and the user is given a feedback on his answer. A typical IMC test feedback for an alternate response question looks like the following.

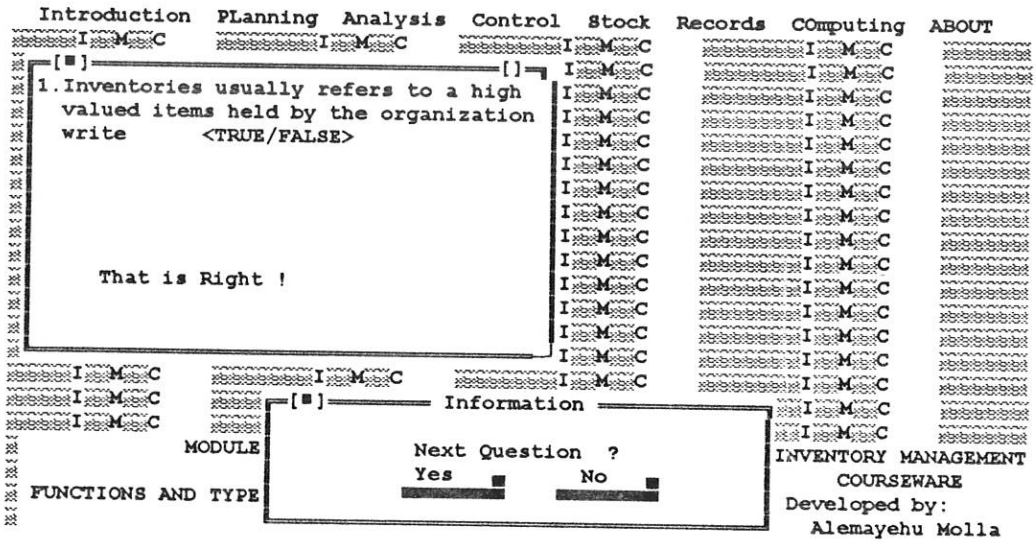


Fig 32.14: IMC sample test feedback screen

At the end of each question, the learner is given an option either to continue or quit from the test sessions. Up on completion of the test session, the total result of the test will be computed and displayed to the student. The criterion used in assessing student"s test performance as acceptable or not is simply a 50% rule. When the learner scores 50% or above on a test, he/she will be given a reward (a song which says SO SHE IS A JOLLY). Otherwise, he/she will be provided with an advise to repeat the lesson. Following is a screen which shows such test result information

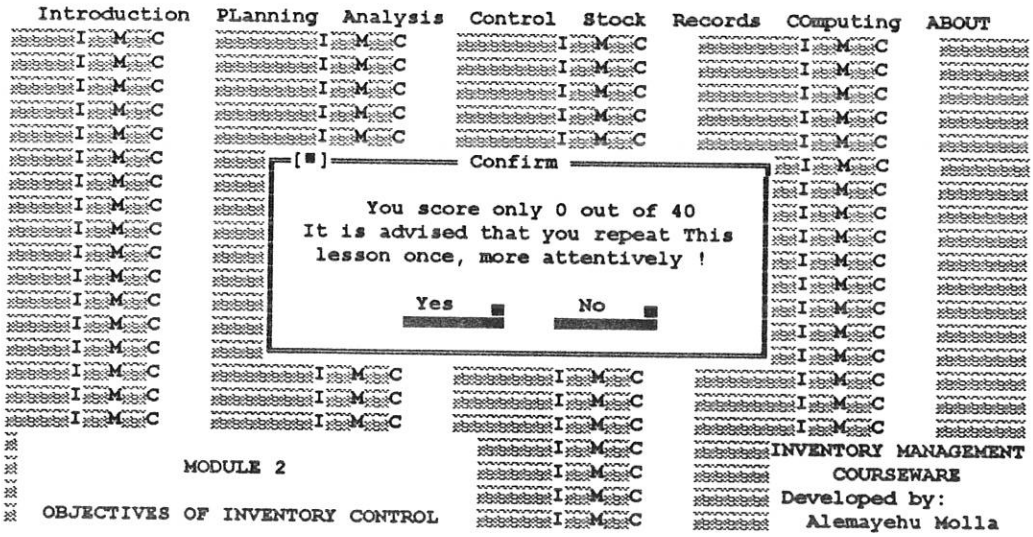


Fig 32.15: IMC Sample Test Result Screen

In addition, the courseware makes an extensive use of the colour animation and sound feature provided in the Turbo Vision Software.

5.1.8 Ending IMC Session

Exiting from IMC is possible only from the main menu. The User can exit from the system by pressing **ALT + x**.

5.2 CONCLUSION AND RECOMMENDATIONS

This study was an attempt to demonstrate the development of a courseware by applying one of the suggested models in courseware development. In particular, the study outlined, at the out set, the following questions for which it tried to find explanations. The questions are

1. **What are the factors that have to be considered in designing a courseware?**
2. **What characteristic features should a courseware posses ? and**
3. **How does one go about that task of designing a courseware?**

This section, therefore, attempts to provide some concluding remarks based on the experience gained in developing IMC and hence in answering the above questions; practical recommendations for converting the prototype courseware into a matured system and recommendations for further study.

5.2.1 Conclusion

From the experience gained in developing IMC using the ISD model and general purpose programming language-Turbo Vision, the following remarks could be drawn/stated.

1. The identification of the basic characteristic features of a courseware: **program operation, pedagogical, program content and documentation,** from the literatures prior to the development process has helped to have a checklist that can serve as a basis for directing and measuring all the activities of developing IMC.

2. In developing the courseware, ISD model was used. The model proposes a five-phase activity (analysis, design, development, implementation and evaluation) in developing a courseware. Experience in implementing this model in the current work (together with other tools, see below) indicates that, the model has not only guided the worker in fully implementing the objectives of the study and hence answering the above tested questions but also enabled to accomplish this task within the specified period of time, as planned.

As indicated in chapter three, however, the model is limited only in telling what to do and the order in which the activities should be done and not "how to do ". To complement such limitation of the model, different tools were used. In particular,

a). in order to define the instructional objectives in measurable and observable terms, the study makes use of key verbs adapted from ITU/TDG. The settingof the instructional objectives, based on these key verbs, facilitated the development task and enabled IMC to be related to the specific requirements outlined in the analysis phase.

b). to represent the responses to be used in the courseware, a response analysis form is implemented

c). in preparing the program specifications, structured programming and systems analysis tools such as screen flow diagrams, structured programming flowcharts and pseudocode are used.

In addition, in most of the cases, the specifications prepared in the design phase were implemented directly without much difficulty. Where this was not the case (for instance, in

converting the design specifications of response analysis and lesson flows), it was possible to work back and adjust/revise the specifications - as per the provision of the ISD model depicted in the relationship between its different events (analysis, design, development, implementation, and evaluation).

3. As the expertise of an individual is limited to a specific area and because courseware development requires different expertise, it is suggested that a team approach be followed in the design/development of courseware.The experience gained in developing IMC, engaging a role playing team, also ascertains the benefit of following the team approach.

4. It is possible to develop a courseware, even in the absence of a specialized authoring software, by making use of general purpose programming languages such as Turbo Vision. In particular, the object orientation/ event driven feature of the language which allows multiple, resizable and overlapping windows, pull down menus, dialog boxes, built in colour installations, buttons, etc., enabled in incorporating into IMC the characteristic features of a courseware (such as animation, colour, sound.)

5. Generally, the study, by way of attempting to design and develop a user-friendly prototype courseware has provided sufficient answer for the questions outlined at the outset and hence has met its objectives.

5.2.2 Recommendations

RECOMMENDATION FOR MATURING THE SYSTEM

Much of the design work has already been undertaken in developing the current prototype Inventory Management Courseware (IMC). In this regard, a framework design and program which could easily be converted into a mature system is established; the objective a student is expected to attain at the end of each module or lesson which is the basis for developing the lessons and the tests is defined; almost all of the objects needed in the program are initialized and constructed; the procedures, files, lesson menus, comments and instructions are also defined. In order to convert this prototype courseware in to a mature system and make it operational, the following implementation measures are identified for subsequent implementation.

1. Incorporating in IMC the tests and lessons for the remaining part of the courseware
2. Converting the lessons and tests in to a program source code
3. Once the programming aspect of the entire courseware is finished, to ensure the quality and interactivity of the courseware, different types of tests have to be undertaken. These tests have to at least include
    a. **Program Operation Testing** : to ensure its clarity from bugs and breaks, the courseware has to primarily be tested against bugs and breaks. This test must also make sure the clarity of the different instructions and the error handling mechanism implemented.

b. **Program Content Testing** : the courseware has also to be tested as to its content coverage. This involves testing the objectives of each modules, the validity and reliability of the tests and the significance of the lessons

c. **Interactivity testing**: Sample students have to be taken to use the courseware and the feedback of the students have to be collected to ensure the interactivity of the courseware

4. Experiment with different categories of learners has to be undertaken in order to determine the average time required to complete a particular lesson

5. A complete manual of how to use the courseware and other related documentations has to be prepared.

6. A training has to be provided on how to use the courseware for the different users

RECOMMENDATIONS FOR FURTHER STUDY

After the completion of the mature development of the courseware and its initial testing, to ascertain the effectiveness of the courseware over other methods and prove or disprove the claimed advantages of a courseware, undertaking an experiment is recommended as a potential area subject to further study. This experiment may be undertaken in the following manner

1. Forming two groups - one experiment group and one controlling group
2. The experiment group will take the course for a semester using the courseware in addition to the class lecture while the control group uses the conventional ways of learning ,i.e., class lectures, text books ...

3. Evaluate the performance of students in each group and make inference

In addition as the current trend in computer utilization is towards a networked system, developing the networkable version of the courseware (as required) is another potential area for further study.

BIBLIOGRAPHY

1. Ahmed-Ouamer, Richard. 1993. AGEDI a computer assisted courseware engineering for industrial applications. In Proceedings of the IEEE International Conference on Systems Man and Cybernetics Part 2 (15) V (2). IEEE service center, Pisactaway, New Jersey: 540-545.

2. Alberico, R., and Mico, M. 1990. Expert systems for reference and infornation retrieval. London: Meckler: 5-15.

3. Alessis, Stephen M., and Trollip,Stanley R. 1991.Computer Based Instruction: methods and development. NewJeresy: Prentice Hall Englewood Cliffs: 119-177.

4. Arano ,Takashi et al. 1993. An object oriented prototyping approach to system development. IEEE service center, Pisactaway, New Jersey: 20-19.

5. Asmerom, K. et al. 1989. Students dropout in institutions of higher learning in Ethiopia: magnitude, causes and cures. Ethiopian Journal of Education 11(21): 9-21.

6. Augenstine, Friedrich, T. Ottmann and J. Shehoning. 1993. How to incorporate expert knowldege in to an authoring system: computer meditated

education of information technology. In Proceedings of The IFIP TC3/WG3.1/Wg3.5 open Conference on Informatics and Changes in Learning IFIP transaction A: computer Science and Technology N A - 34: North-Holland: Elsevier Science Publishers B.V: 117-126.

7. Azeb, Desta. 1986. characteristics of effective teaching.In Proceedings of the National Workshop on Strengthening Educational Research Capabilities for Effective Teaching and Learning Process, edited by the IER, AAU. Addis Ababa:1-19.

8. Barker, Philp., and Karim Manji. 1992. Computer based training : an institutional approach. Education and Computing 8(3) (July): North-Holland: Elsevier Science Publishers B.V:229-237.

9. Bear, George G. 1984. Microcomputers and School Effectivness. Educational Technology (January): 11-15.

10. Cronin, Blaise (ed.) 1989. Training and education for on-line. London: Taylor Graham and Contributors.(The foundations of Information Science Series). 52-66. 11. Boufaida, Mohamoud,and Patrick Barril. 1993. Situtated planning and resources in the design of an intellegent tutorial. In Proceedings- ACM Computer Science Series Conference. New York: ACM: 189-194.

12. Boyad-Barreft, Oliver. 1990. Computers and learning. Great Britain: Addison Wesley Pub.

13. Bruce, Juilet. 1992. The world future society: taking the long view. Topic 193: 31-33.

14. Bunderson,Victor, C. 1981. Courseware: computer assisted instruction: a state of the art assessment.Orlando: Acadamic Press: 91-122.

15. Burns, Richard. Methods for Individualizing Instruction: 45-46.

16. Cheever, Daniel S. et al. 1985. School Adminstrators guide to computers in education. New York: Addison Wesley. 125-159.

17. Chorover, Stephan L. 1984. Cautions on computer in education: effects on the student teacher relation. BYTE:(June): 18-22.

18. Coburn, Peter et al. 1985. Practical guide to computers in education. 2nd ed.USA: New York: Addison Wesley.1-77, 159-189.

19. Dean, Christopher and Quentine Whitlock. 1983. A Handbook of computer based training. London: Kogan Page : 28-37, 173-196.

20. Dunn, Seamus, and Valerie Morgan. 1987. The impact of the computer on education: a course for teachers. London: Prentice-HAll International: 99-119.

21. Ford, Nigel. 1991. Expert system and AI: an Information Managers Guide. London: Libraray Association Publishing: 212-217.

22. Gagne, M. Robert, Walter Wager and Alicia Rojas. 1981. Planning and authoring computer assisted instruction. Educational Technology. (september): 17-26.

23. Gagne, R. M. 1977. The conditions of learning. 3rd ed. NewYork: Holt, Rinehart and Winston.

24. Gagne, R. M., Brigs L.S. 1979. Principles of instructional design. 2nd ed. New York: Rinehart and Winston.

25. IDRC. 1988. Paper resulting from the meeting to develop an information strategy for IDRC for Africa. Canada

26. Ihara, Hiroyoshi. 1993. The multi purpose telephone. Kenshu-In: periodical for Jica ex-participants 70: 5-7.

27. ITU. 1983. 4th world Telecommunications Forum Part -1.Geneva.

28. ITU/TDG. 1979. Training Development Guidelines 29.Kearsley, G. 1983. Computer based training. london: Addison-Wesley Publishing : 1-41, 103-122.

30. Longhorne, Manjo, et al. 1989. Teaching with computers a new menu for the 90's. 1-11, 32-48.

31. Lunin, Lois F. 1993. Prespectives on knowlwdge utilization. JASIS 44(4): 217-220.

32. Mandell, Steven l. 1985. Computers and data processing: concepts and applications.3rd ed. USA: West Publishing : 419-447, 474-496.

33. Marsh, Herbert W., Bailey Michael. 1993. Mulitdimensional students evaluation of teaching effectivness: a profile analysis. The Journal of Higher Education 64(1): 1-13

34. Mcknight, cliff, Andrew Dillon, and Jhon Richardson.1991. Hypertext in context. Cambridge: Cambridge University Press: 105-120

35. Meera, B. M., and Rahel Bekele. 1994. Computer Assisted Instruction for Classification (CAIC). Paper presented at a DRTC annual seminar-28 on

Teaching and research in classification and Indexing languages- Banglore.

36. Merrill, Paul F., Marvin N. Tolman, Lary christensen, Kathy Hammons, Bret R. Vincent, Peter L.Reynolds.1986. Computers in Education. New Jersey:Prentice-Hall: 1-52.

37. Moonen, Jef. 1989. Courseware development at the crossroads?. Journal of Education and Computing 5: 103-109.

38. NSTDIC. 1991. Proceedings of the national policy seminar on information systems and services. 27-29 November. Addis Ababa.

39. Offir, Baruch, Mitka R. Golub, and Sarah Cohern Fridel.1993. Attiudes towards courseware as a function of high school students' creativity level. In Proceedings of The IFIP TC3/WG3.1/Wg3.5 Open Conference on Informatics and Changes in Learning IFIP transaction A: computer Science and Technology N A - 34: North-Holland: Elsevier Science Publishers B.V: 211-216.

40. O'Neil, Harold F.(ed.) 1981. Computer based Instruction:a state of the art assessment. Orlando: Acadamic Press:1- 64.

41. PADIS. 1992. Proceedings of PADIS workshop on computer networking in

Africa. 13th September: Addis Ababa.

42. Parodi, G., and D. Ponta. 1993. Multimedia environment for computer based training in electronics and information technology. In Proceedings of The IFIP WG3.4 Working Conference on Computer meditated education of Information Technology Proffessionals and advanced end users (A 35): IFIP transaction A: computer Science and Technology N A-35: North-Holland: Elsevier Science Publishers B.V: 173-180.

43. Parsaye, Kamran et al. 1989. Intellegent Databases Object-Oriented, Deductive hypermedia technologies. New York: John Wiley and Sons: 97-104, 152-167, 223-240, 280-291.

44. Pistorius, M. C. et al. 1992. Computer assisted instruction development model for a computer concepts course. Computing and Control Engineering Journal 3(1) (January): 13-18.

45. Rahel, Bekele. 1994. Computer Assisted Instruction for Bibiliometrics (CAIB). paper presented at a seminar in DRTC, Indian Statistical Institute, Banglore.

46. Rahel, Bekele. 1992. The application of information technologies in Higher

learning institutions in Ethiopia with special emphasis to computer technology. MSC thesis. SISA- Addis Ababa University, Addis Ababa.

47. Rosenberg, Marc J. 1982. The ABC's of ISD. Training and Development Journal. (September): 44-50.

48. Salisbury David F. 1984. How to decide when and where to use microcomputer for Instruction. Journal of Educational Technology. (March) : 22-24.

49. Sanders, Donald H. 1985. Computers Today. 2nd ed. USA: McGraw-Hill Inc.: 423-473.

50. Scholtz Jean et al. 1993. Object oriented programming: the promise and the reality. North-Holland: Elsevier Science Publishers B.V: 199-204.

51. Stern, Robert, and Nancy, Stern. 1982. An Introduction to computers and information processing. Canada: John Wiley and Sons : 423-473.

52. Tagg E. D and Frank Lovis. ed. 1988. Computers In Education. Proceedings of the IFIP TC 3 1st European Conference on Computers in Education- ECCE 88. 24-29 July. North Holland: Elsevier science publishing company.

53. Watts, Norman. 1981. A dozen uses for the computer in education. Journal of Educational Technology.(April): 18-22.

54. Wenger, Etienre. 1987. Artificial Intellegence and Tutoring Systems: Computational and cognitive approach to the communication of knowldege. California: Morgan Kaufman Publishers: 1-23.

55. Williams, Fredrick. 1989. The new communication.New York: Wadsworth Inc.: 300-305.

APPENDIX

1A :TESTS USED IN THE COURSEWARE

MODULE 1: TEST
INSTRUCTION: *ANSWER THE FOLLOWING QUESTIONS*
TS 1

1. Inventories usually refer to a high valued items held by the organization <**TRUE/FALSE**>____
2. The kind of business a firm undertakes usually determines the items to be carried out in inventory <**TRUE/FALSE**>. _____
3. Inventories in Manufacturing enterprise can be classified in to two:
   _____ and _____
4. One of the following is not a reason in maintaining inventories
   a. Meeting anticipated demand
   b. Promoting stock out
   c. Smoothing production requirement
   d. Taking stock cycle advantage
   Enter the letter of Your choice ____

MODULE 2: TEST
INSTRUCTION: *ANSWER THE FOLLOWING QUESTIONS*
TS 2

1. Maintaining higher customer level is not the objective of inventory management <TRUE/FALSE> _____
2. Why do you think marketing favours high inventory level ?
   a. To get flexibility
   b. To take advantage of consolidation of requirements
   c. To assure rapid assembly and delivery of a wide range of product models
   d. To help vendors
   Enter the letter of Your Choice ____
3. Production manager favours high inventory, why ?
   a. to get flexibility in daily planning
   b. to prevent production shut down
   c. To protect unforseen problems due to lack of materials
   d. All of the above
   Enter the letter of Your Choice ____
4. Purchasing management also favours higher inventory levels
   <**TRUE/ FALSE**> _____

MODULE 3: TEST

INSTRUCTION: *ANSWER THE FOLLOWING QUESTIONS*

TS 3

1. Identify the incorrect feature of forecasting from the following
   a. System variation in the past and feature
   b. Possible variation between actual and predicted result
   c. Group forecast is more accurate than individual forecast
   d. Inverse relationship between forecast and time horizon

   Enter letter of Your Choice ____

2. Which of the following doesn't belong to the steps of forecasting
   a. Determiming when, where and why is the forecast needed
   b. Defining the techniques used for forecasting
   c. Validating the techniques
   d. Data collection
   e. None of the above

   Enter letter of Your Choice ____

3. What are the two approaches in forecasting?
   1. _____
   2. _____

MODULE 4: TEST

INSTRUCTION: *ANSWER THE FOLLOWING QUESTIONS*

TS 4

1. list the 3 methods of quantitative forecasting
   1. _____
   2. _____
   3. _____
2. Which of the following statement is true about TOP DOWN approach
   a. Top-down approach doesn't make inference from general business forecast
   b. In top down approach planning starts from top management and goes down to lower management
   c. Top down approach makes inference only from the forecasts of government
   d. None of the above

   Enter letter of Your Choice ____

TS 4.1

3. In bottom up approach the basis for preparing aggregate demands are individual government and business forecasts **<TRUE/FALSE>** ____
4. List the components of time series
   1. _____
   2. _____
   3. _____
   4. _____

TS 4.2

5. Identify the false statement
   a. Trend represents long term savings about the trend line
   b. Seasonal effects are patterns of similar kind which occurs during corresponding months
   c. Random components are sporadic effects due to changes and unusual occurrences
   d. None of the above
      Enter letter of your choice ____

6. Identify the statement which doesn't belong to the Monk's forecasting procedure
   a. Projecting the past trend
   b. getting historical data
   c. developing a trend equation
   d. all of the above
      Enter letter of your choice ____

TS 4.3

7. Dire dawa textiles mills has kept records of usage for one of its items for the last four years as shown below

YEAR	QUARTER 1	QUARTER 2	QUARTER 3	QUARTER 4
1983	5500	6400	6800	5000
1984	6000	6500	7000	6100
1985	5800	6400	6700	6000
1986	6600	7200	8000	6300

Forecast the factory's use of the item for each quarters of 1987 using trend analysis.
Enter Your answer in the space provided. Round the numbers to two decimal places

a. n= _____
b. The summation of X (period)____
c. The summation of XY ____
d. The summation of X^2 ____
e. The summation of Y (demand)____
f. The Value of 'a' = _____
g. The value of 'b' = ____
h. The Regression line Equation = _____
i. 1st quarter Forecast = _____
k. 3rd quarter Forecast = _____
j. 2nd quarter Forecast = _____
l. 4th quarter Forecast = _____

TS 4.4

8. Akaki spare parts has the following sales data for the last 10 weeks

Week	1	2	3	4	5	6	7	8	9	10
Demand	120	110	80	60	140	135	95	130	115	124

a. Using 3 weeks, 5 weeks and 6 weeks moving average forecast the demand for the 11th week.
b. Recommend a forecast figure for the company.

Fill your answer in the space provided (round of to one decimal place)

i. From Which week do you start computing the moving average _____

ii.

WEEK	FORECAST		
	3 WEEKS	5 WEEKS	6 WEEKS
7			
8			
9			
10			
11			
MEAN			
ABS.DEVIATION			

iv. Which forecasting method do you propose _____

1B: VALID ANSWERS FOR THE TESTS

MODULE NO.	SCREEN NO.	QUESTION NO.	VALID VALUES
1	TS1	1 2 3 4	FALSE, false, F, f true, TRUE, t, T DIRECT, direct, INDIRECT, INDIRECT B, b
2	TS2	1 2 3 4	FALSE, false, F, f c, C D, d TRUE, true, t, T
3	TS3	1 2 3	A, a C, c QUANTITATIVE, QUALITATIVE, quantitative, qualitative
4	TS 4.0	1 2	TOP DOWN, TOPDOWN, topdown, top down; BOTTOM UP, BOTTOMUP, bottom up, bottom up; STATISTICAL, statistical ** *the order in which this responses are entered doesn't matter* D, d
	TS 4.1	3 4	FALSE, false, F, f TREND, CYCLICAL, SEASONAL, RANDOM; trend, cyclical, seasonal, random ** *the order in which this responses are entered doesn't matter*
	TS 4.2	5 6	A, a, A, a,
	TS 4.3	7 a. b. c. d e. f. g. h. i. j. k. l.	16 136 1175600 1496 102300 -1257.53 900.15 YC= -1257.53 + 900.15b 14045.02 14945.17 15845.32 16745.47
	TS4.4	i. ii. III. IV.	7, WEEK 7, FOR WEEK 7, 7 WEEK ; 7, week 7, for week 7, 7 week 111.7 105 107.5 123.3 102 103.3 120.0 112 106.7 113.3 123 112.5 9.78 10.5 14.75 3 WEEKS, WEEK 3 ; 3 weeks, week 3

2: COURSEWARE LESSONS

NAIN TOPIC: INTRODUCTION

MODULE NO. 1: FUNCTION AND TYPE OF INVENTORIES

LS 1

Any organization regardless of its size or orientation needs material resources. But due to gap in time, distance, place and uncertainty, organizations couldn't get these resources as they want them from the market.

SO WHAT SHOULD BE DONE?

LS 2

YES! They have to keep a stock of these resources- **INVENTORIES**

HS 1

Inventories are idle physical stock of goods rangig from small things like pencils, paper clips to machines, trucks possessing an economic value and Kept for the purpose of future use

LS 3

At this point you may be wondering how a firm decide what inventory to carry out

LS 4

Normally many of the items a firm carries in inventory relate to the kind of business it engages, i.e ***MANUFACTURING***, ***DEPARTMENTAL STORE***, ***HOSPITAL*** and etc.

HS 2

MANUFACTURING firms carry supplies of raw materials purchased parts, partially completed items and finished goods.

HS 3

DEPARTMENTAL STORES carry clothing, furniture carpeting , stationary appliances...

HS 4

HOSPITALS stock drugs, surgical supplies sheets, etc.

LS 5

Though organizations, depending on their activity keep inventory items, inventories for manufacturing enterprises can, generally, be classified into two categories ***DIRECT*** and ***INDIRECT***. The basic reasons behind maintaining a direct or indirect inventories are ....

LS 6

1. To meat anticipated demand
2. To smooth production requirements
3. To protect against stock outs
4. To take advantages of stock cycles
5. To hedge against price increases
6. To permit operations

HS 5

DIRECT inventories are those items which form a component of a finished good. They can be raw materials, work in progress and finished products.

HS 6

INDIRECT inventories are inventory of raw materials which do not form part of finished products. They may include such items as lubricants, grease,oil ...

MODULE NO. 2: **OBJECTIVES OF INVENTORY CONTROL**

LS 7

In maintaining inventories due to

1. The amount of money inventory represents
2. The impact of inventories on the daily operation of an organization the following basic questions have to be answered:
   - How much to stock ?
   - When to stock ?, and
   - What to stock ?.

That means there is a need for ***MANAGING INVENTORIES***

HS 7

INVENTORY MANAGEMENT is a discipline concerned with the management of different items for the smooth running of an organization.

LS 8

The two broader objectives of inventory controlling are

1. Maximize level of ***CUSTOMER SERVICE*** and
2. Minimize cost of providing high level of customer service

But are these objectives (the above objectives) not contradictory?

HS 8

Customer service is the totality of the service to be given to a customer

LS 9

YES ! generally they are in opposition, that is high level of customer service lead to high cost, and low costs are usually accompanied by low level of customer service.

So how can a firm ensure high level of customer service and at the some time minimize the cost of providing such high level customer services ?

LS 10

As you probably would have imagined it may not be possible to ensure both of them with maximum combination. Consequently most of the inventory decisions are trade-offs involving a compromise between cost and customer service level.

LS 11

This compromise is made by management's selection of a certain desired service level where the goal of inventory control becomes attaining that service level at the lowest possible cost or conversely. So one has to make a balance between these decisions in order to avoid both over and under stocking. And this problem has different implications for various departments such as: ***MARKETING, PRODUCTION,*** **and** ***PURCHASING***

HS 9

To assure rapid assembly and delivery of a wide range of product models marketing favours a relatively large inventory stocks

HS 10

The production manager like marketing favours high inventory levels. The reason being for getting flexibility in daily planning, protecting from production shut down due to stock outs and protecting unforeseen problems in producing a given component.

HS 11

The purchasing manager whose primary concern is the size and frequency of individual orders favours placing of fewer and larger orders which may increase total inventory.

LS 12

Definitely there is a certain conflict between the objectives of the various departments. The concepts and techniques useful in analyzing this problems to arrive at sound policy decisions are the focal points of ***INVENTORY MANAGEMENT*** which will be discussed throughout this courseware

MAIN TOPIC: **MATERIAL PLANNING AND BUDGETING**

MODULE NO. 3: **FORECASTING: Importance, Features and Approaches**

LS 13

You may have heard from the radio, or read from newspaper ... the daily weather forecastes. You may have wondered sometimes by the closeness and sometimes by the variation of the forecast from the actual. Don't be taken away by that. That is the nature of forecasting, i.e., there is no such a thing as a sure bet. Predictions usually turn out to be in the ball mark, but occasionally they can miss the mark completely.

LS 14

Forecasting demand is a lot like forecasting the weather. In business forecasts are the basis for any sort of plans such as capacity, budgets, sales, production, inventory, purchasing, etc., They are also used to predict profits, revenues, costs, productivity, prices, etc. All of these forecasts have some common ***FEATURES*** and needs certain ***STEPS.***

HS 12

The **FEATURES** of forecasts are

1. Uniformity of the system in the past and feature
2. Possible variation in the actual and predicted result
3. Accuracy of group item forecasts than individual item forecasts
4. Time horizon sensitivity, i.e., decrease in forecasts accuracy as the time period increases.

HS 13

The five major steps in forecasting are

1. Determine purpose and time the forecast is needed
2. Establish the time period the forecast covers.
3. State the techniques
4. Collect data and prepare the forecast
5. Evaluate the forecast to see practicability

MODULE 4: METHODS OF FORECASTING

LS 15

There are different methods for undertaking quantitative forecasts. Accordingly forecasting is sometimes done by a ***TOP-DOWN*** method or a ***BOTTOM-UP*** method or in some other cases by using ***STATISTICAL AND MATHEMATICAL PROCEDURES*** and extrapolating past experience in to the future.

LS 16

Most companies, however use both Top-down and bottom-up methods at the same time and combine the resulting projections in to a single forecast. This forecast, before put in to action will be adjusted up or down according to what the organization's top people think about the future.

HS 14

The ***TOP DOWN*** approach of forecasting basically is making inference from forecasts of general business and economic conditions made by the government, large companies, higher learning institutions or organizations which specialize in economic forecasting.

HS 14.1

Example: An economic forecast from one of the above sources may predict the gross national product sayfor the coming year, to be 11 billion dollars. The question for a particular organization, therefore, is to evaluate and translate the effect of this forecast on the company's operation and then prepare a product or sale forecast for that organization

HS 15

The ***BOTTOM UP*** approach is the reverse of the top down approach. In this approach, an aggregate forecast is prepared from the individual estimates of expected end product sale, and required service hours. These information are obtained from sales people, dealers and customers. In addition, the forecaster evaluates the past sales patterns to make the aggregate forecast.

HS 16

STATISTICAL /MATHEMATICAL forecasting method involves time series analysis, moving averages, exponential smoothing and regression analysis. In this courseware we will discuss the ***TIME SERIES*** and *MOVING AVERAGE* methods only.

HS 16.1a

A ***TIME SERIES*** is a set of observations of some variables over time. The series is usually tabulated or graphed in a manner that readily conveys the behaviour of the subject variable. Your main task as a forecaster in applying the time series method is to determine how the series is dependent on time and develop a means of predicting future levels with some degree of reliability.

HS 16.2a

The nature of the time dependence of the series is mostly analyzed by the components of the time series. These are

(T)REND
(C)YCLICAL
(S)EASONAL
(R)ANDOM OR IRREGULAR

FORMULA:

In the classical model of time series analysis
The forecast (Y) is given by

Y=T*C*S*R

HS 16.2a.1

TREND REPRESENTS a long term secular movement, and is the characteristics of many economic series.
CYCLICAL FACTORS are long term savings about the trend line and are usually associated with business cycles
SEASONAL EFFECTS are similar patterns occurring during corresponding months of successive years
RANDOM or irregular components are sporadic effects due to chance and unusual occurrences.

HS 16.3a

Though there are different forecasting procedures, we will use the forecasting procedure developed by Monks. It is important that you have to pay attention to these procedures.

. Obtain historical data
. Develop a trend equation to develop the data
. Develop seasonal index (if desired)
. Project the trend in to the future
. Multiply the monthly trend values by the seasonal index
. Modify the projected values by a knowledge of
    a. Cyclical business conditions (C)
    b. Anticipated irregular effects (R)

HS 16.4a

The common approach of trend calculation is to draw or fit a straight line through the plot of demand figures so that the points above and below the line are more or less equal in number and distance from the line. Now try to recall for what purpose you were using the least square method

HS 16.5a

The formula for the regression line equation is

$YC = a + bx$

Where: YC = The trend value at time period X

a = Trend value where x = 0

b = slope of the trend

x = time period with in the range 1,2,3...n

In calculating the trend value and the slope we need another Formula.

HS 16.6a

The formula for calculating the trend value ('a') can be represented by

$$\sum y = na + \sum x(b)$$

While the slope of the trend (b) will be calculated using the formula

$$\sum xy = \sum x(a) + \sum x^2(b)$$

HS 16.7a

Generally the formulas needed in calculating the trend are

$YC = a + b_x$

$$\sum y = na + \sum x(b)$$

$$\sum xy = \sum x(a) + \sum x^2(b)$$

and before actually starting computation you have to identify the x's and y's and find the

$$\sum x, \sum y, \sum xy, \sum x^2$$

HS 16.8a

Now watch very carefully how the computation is going to be made

Example: The monthly sales record of XYZ organization for thelast three years show the following figure. XYZ wants to forecast its demand for the coming year (4^{th} year). How much would that demand be.

MONTH	YEAR 1	YEAR 2	YEAR 3

	Period	DD	Period	DD	Period	DD
JANUARY	1	100	13	125	25	146
FEBRUARY	2	128	14	148	26	175
MARCH	3	138	15	155	27	173
APRIL	4	127	16	138	28	161
MAY	5	112	17	128	29	150
JUNE	6	115	18	142	30	162
JULY	7	128	19	152	31	171
AUGUST	8	134	20	156	32	173
SEPTEMBER	9	130	21	140	33	168
OCTOBER	10	116	22	122	34	157
NOVEMBER	11	100	23	108	35	130
DECEMBER	12	95	24	100	36	125

S 16.9a

SOLUTION:

From the above data the

x = the time period (the period number which ranges from 1 up to 36)

y = the demand. Therefore

n = Total time period which is 36

$\sum y$ **= 4928** $\sum x$ **= 666**

$\sum xy$ **= 16206** $\sum x^2$ **= 95902**

MONTH (X)	DEMAND (XY)	XY	X^2
1	**100**	**100**	**1**
2	**128**	**256**	**4**
3	**138**	**414**	**9**
....	....	....	..
36	**125**	**4500**	**1296**
666	**4928**	**95902**	**16206**

HS 16.10a

So the equation for finding the values 'a' and 'b' with these two variables unknown looks like the following

$$\sum y = na + \sum x(b) \quad = 4928 = 36a + 666b \ldots\ldots\ldots 1$$

$$\sum xy = \sum x(a) + \sum x^2(b) \quad = 95902 = 666a + 6206b \ldots 2$$

To solve the two equations simultaneously and get the values of a and b, we will multiply equation 1 by -18.5 (666/36) and add it to equation 2. After the multiplication equation 1 becomes

-91168 = -666a - 12321b

HS 16.11a

When we add equation 1 to equation 2 we will get

-91168 = -666a - 12321b 1

95902 = 666a + 6206b 2

4734 = 3885b

$$b = \frac{4734}{3885}$$

b = 1.22

Now by substituting the values of '**b**' in equation 2 we can solve the value for '**a**' in the following way

95902 = 666a + 16206b

95902 = 666a + 16206 * 1.22

666a = 95902 - 19771

$$a = \frac{76131}{666}$$

a = 119.31

By substituting the values of '**a**' and '**b**' in the

YC = a + bx equation we will get the following regression line equation.

YC = 119.31 + 1.22x

HS 16.12a

Using the equation **YC = 119.31 + 1.22x** we will get the forecast for each months of the forth year by substituting the values of 'x' starting from 37 to 48. The result is the following.

Month (x)	FORECASTED DEMAND
37	159.45
38	160.67
39	161.89
40	163.11
41	164.33
42	165.55
43	166.81
44	167.99
45	169.21
46	170.43
47	171.65
48	172.87

HS 16.13a

Now it is time that you practice forecasting using trend analysis. Here is the problem.

Addis Ababa Textile Factory is considering the purchase of one of its raw materials for the coming year. The factory has usage data for the past two years recorded in terms of the seasons of the year as shown below

YEAR	1ST qurater	2ND quarter	3rdquarter	4th quarter
1	12000	10000	9000	10500
2	14000	9000	8000	12000

Using time series analysis determine forecasts for year 3

HS 16.14a

Before actually going to the computation, you have to identify the formulas involved. Remember there are three formulas

What is the formula of the regression line equation? _____

What is the formula for calculating the values of '**a**'?_____

What is the formula for calculating the values of '**b**'?_____

HS 16.15a

From the given data identify the values of x, y and n.

Do you need to see the data? _____

HS 16.16a

OK! Now enter the values for the following

$$\sum x$$
$$\sum y$$
$$\sum xy$$
$$\sum x^2$$

HS 16.17a

OK ! this is the time to formulate the two equations and find the multiplier. From the formulas of '**a**' **and** '**b**' and using the above values

Equation to find '**a**' :_____

Equation to find '**b**' :_____

Multiplier of the first equation: ___

HS 16.18a

When the first equation is multiplied by the multiplier you will get the following result:

380250 = 36a + 36b

And subtrating this from the second equation gives

377000 = 36a + 204b

- 380250 =- 36a - 36b

-3250 = 168b

HS 16.19a

From the above result calculate values of '**b**' **and** '**a**'

'**b**' :_____

'**a**' :_____

HS 16.20a

Substitute the values of '**a**' and '**b**' in the equation

YC = a + bx.

The trend equatiom becomes

YC= 10581.87 - 19.34X

HS 16.21a

That is good ! So far you are doing right. The remaining step is to prepare the forecast for each of the quarters of the third year by substituting the values of '**x**' in to the equation.

What are these values of '**X**'**(ENTER THE VALUES BY SEPARATING WITH COMA)**

_____

HS 16.22a

Based on your computation the forecast looks like the following

Year	1st Quarter	2nd Quarter	3rd Quarter	4th Quarter
3	10407.72	10388.37	10369.02	10349.67

That is a good progress ! Do you like to repeat the exercise. **<YES/NO>**_____

HS 16.1b

The moving average method averages the data from a few past recent periods and this averages becomes the forecast for the next period. In doing so, based on the period considered for the average , it involves adding the most recent data and dropping the oldest data.

HS 16.2b

The key question in forecasting using the moving average method is to determine how many periods of data are to be included in computing the average. Therefore, depending on the period considered,i.e., 2 years, 3 years...; or 6 months, 7 months...; or 5 weeks, 6 week,etc., one can have different forecasts. From these different forecasting periods the one which gives the smallest mean absolute deviation will be selected.

HS 16.3b

The steps in calculating the moving average are the Following:
In order to calculate a moving average based on a three week period say for week 5
STEP 1: Take week 2,3 and 4's demand
STEP 2: Sum it and divide it to three. The result is forecast for week 5
STEP 3: Subtract the forecast you get in step 2 from the actual demand of week 5 and take the absolute value of the result. This becomes the absolute deviation.
STEP 4: To forecast the demand for week six, drop week 2's demand and take week 5's actual demand so your data become week 3rd's, 4th's and 5th's. Repeat step 2 and 3
STEP 4: Likewise for all given demand data repeat steps 2 through 4

HS 16.3b

NOTE: The process you have just seen will be the same in calculating the forecast based on 5 weeks 7weeks,etc., except that the data to be included are 5 week, 7 week,etc., and the divider is 5,7,etc.,

HS 16.4b

STEP 5: When you complete step 1 through 4 for all the time period given, you will get forecasts and absolute deviations. Here you sum up the absolute deviations for each of the time periods and divide it by the number of observations. this gives you the mean absolute deviation.

NOTE: the number of observations in calculating the mean absolute deviation should be the same for all time periods. Therefore the counting should start from the longer period involved in the forecast

HS 16.5b

STEP 6: Once the mean absolute deviation is selected for each time period, select a time period whose mean absolute deviation is small.

STEP 7: Using the selected time period prepare the forecast for the required period.

The following example indicates how to using the moving average method in forecasting

HS 16.6b

The following data is related to the weekly demand of toys. Compute a 3 week, 5 week and 7 week moving average forecast. Compute the result and finally forecast the demand for the 18th week using the most accurate one.

Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
DD	100	125	90	110	105	130	85	102	110	90	105	95	115	120	80	95	100

HS 16.7b

In the above example since the longest time period for which moving average is going to be calculated is 7 weeks, we start computing the moving average forecast starting from the 8th week. Therefore for week 8 the forecast will be

3 week moving average = (105+130+85) / 3 = 106.7

Absolute Deviation = 102 - 106.7= 4.7

5 week moving average = (90+110+105+130+85) /5= 104

Absolute Deviation = 102 - 104 = 2

7 week moving average= (100+125+90+110+105+130+85)/7= 106.4

Absolute Deviation = 102 - 106.4 = 4.4

Likewise for week 9 the forecast and the absolute deviation based on 3, 5 and 7 weeks moving average.

	Actual Demand	Forecast	Absolute Deviation
week 3	110	105.7	4.3
week 5	110	106.4	3.6
week 7	110	106.7	3.3

This same computation continues until week 17 and the resulting figure would be

HS 16.8b

WEEK	DEMAND	3 WEEK		5 WEEK		7 WEEK	
		FORECAST	ABS DEV.	FORECAST	ABS DEV.	FORECAST	ABSDEV.
1	100						
2	125						
3	90						
4	110						
5	105						
6	130						
7	85						
8	102	106.7	4.7	104	2.0	106.4	4.4
9	110	105.7	4.3	106.4	3.6	106.7	3.3
10	90	99.0	9.0	106.4	16.4	104.6	14.6
11	105	100.7	4.3	103.4	1.4	104.6	0.4
12	95	101.7	6.7	98.4	3.4	103.9	8.9
13	115	96.7	18.3	100.4	14.6	102.4	12.6
14	120	105.7	14.3	103.0	17.0	100.3	19.7
15	80	110.0	30.0	105.0	25.0	105.3	25.3
16	95	105.0	10.0	103.0	8.0	102.1	7.1
17	100	98.3	1.3	101.0	1.0	100.0	0.0

HS 16.9b

from the table result the number of observations (n) are 10 and the sum of the absolute deviation for

3 week = 104.0

5 week = 92.6

7 week = 96.3

Dividing the absolute deviation by 10 (n) gives MeanAbsolute Deviation (MAD)

Therefore MAD For 3 week = 10.4 (104/10)

For 5 week = 9.26 (92.6/10)

For 7 week = 9.63 (96.3/10)

From the above MAD's, the smallest is 9.26 (A period based on 5 weeks). Therefore this will be the base for forecasting the 18th week demand

HS 16.10b

The demand for week 18 using five weeks moving average will be

$$\frac{115+120+80+95+100}{5}=102$$

do you think that you are now ready to perform an exercise or you want to repeat the above processes

<YES/NO>_____

HS 16.11b

Exercise

Tana supermarket has the following sales data for the last 13 months for one of its products.

MONTH	1	2	3	4	5	6	7	8	9	10	11	12	13
DEMAND	130	145	120	135	150	175	160	155	170	185	195	175	195

1. using the 5 month and 7 month moving average forecast the demand for month 14
2. Which forecast figure would you recommend for the super market why

HS 16.12b

From which month are you going to start computing the moving average ___

HS 16.13b

What will be your data for computing the 5 month moving average for month 8____
What are the data for 8 months moving average for month 8_____

HS 16.14b

Now enter the forecast and the corresponding absolute deviations based on 5 months and 7 months moving average for months 8 to 13

MONTH	5 MONTH		7 MONTH	
	FORECAST	ABS DEV	FORECAST	ABS DEV
8				
9				
10				
11				
12				
13				

HS 16.15b

What is the forecast for month 14 based on
5 months moving average _____
8 months moving average _____

HS 16.16b

In order to answer the question which says 'what forecast figure would you ecommend for the supermarket ?' you need to compute the mean absolute deviation

HS 16.17b

Compute the absolute mean deviation (MAD)
MAD based on 5 month is: _____
MAD based on 7 month is: _____

HS 16.18b

So which time period do you choose to prepare the forecast for month 14 _____

ANNEX
PROGRAM SOURCE CODE

```
{*********************************************************}
{                                                         }
{  INVENTORY MANAGEMENT COURSEWARE - Prototype            }
{                                                         }
{            By:                                          }
{             Alemayehu Molla                             }
{                                                         }
{             SISA  1995                                  }
{*********************************************************}
Program Courseware;

uses Test,Jolly,Dos,Cinvhlp,Cinvts,App,Cwrfront,Objects, Menus,
     Msgbox, Drivers, Views,cwrgcons,Crwrcons,Dialogs,Crt;

Var
  CORRES: TResourceFile;
  corres1:TResourceFile;
  Dialogkey:word;
  event:tevent;
TYPE
    Pinvcowr= ^Tinvcowr;
    TInvCoWr = Object(TApplication)
    Constructor init;
    Procedure InitStatusLine; virtual;
    Procedure Initmenubar;virtual;
    Procedure InitDesktoP;virtual;
    Procedure Getevent (var Event: TEvent); virtual;
    Function  Getpalette:ppalette;virtual;
    Procedure HandleEvent(var Event: TEvent); virtual;
    Procedure outofmemory;virtual;
   { procedure EventError(var Event: TEvent);virtual;}
    Procedure Opentest1;
    Procedure Take;
    Procedure Bring;
    Procedure Inform;
    Procedure DoAboutBox;
    Procedure Proto(var s:string);
    Procedure Active;
    Procedure Inputr( k: byte; A1,A2,A3:string; VAR K3: INTEGER);
    Procedure Branch(var s1,s2,s3:string);
    procedure objectve(var s1: string);
    Procedure show1(s: string; k,m: word);
    PROCEDURE INITIAL(VAR S1:STRING);
    procedure right;
    procedure call;
    procedure wrong;
    procedure final(var s1:string);
    Procedure Introduction;
    procedure forecast;
    procedure planning;
    Procedure SELECT;
    Procedure Method;
    Destructor Done;virtual;

End;
```

```
Constructor Tinvcowr.Init;
const
  corresFileName: FNameStr = 'inventor.tvr';
  Var
  L,c,i:INTEGER;
  Screen1, Screen2 :CrWrScrn;
Begin
  Registerviews;
  Registerdialogs;
  RegisterType(RCrWrScrn);
  RegisterHelpFile;
  RegisterType(rStatusline);
  registertype(Rstatuslinecmnt);
  RegisterType(Rmenubar);
  registertype(Rcrwrexcr);
  registertype(Rcrinvtext);
  registertype(Rcrwsheet);
  corres.Init(New(Pbufstream, Init(corresFileName, stOpen, 1024)));
  if corres.Stream^.Status < > 0 then Halt(1);
  corres1.Init(New(PBufStream, Init('invwind.TVR', stOpen, 1024)));
  if corres1.Stream^.Status < > 0 then Halt(1);
  inherited init;
  Take;
  screen1:=crwrscrn(corres.get('IS20'));
  insertwindow(screen1);delay (1000);
  ExecuteDialog((SCREEN1), nil);

  twin:=pcrinvtext(corres1.get('WS1'));
  New(win);
  win^.n := twin^.tinv1.n;
  win^.cinvt := twin^.tinv1.cinvt;
  R.Assign(60,20,81,26);
  win3:= New(pcrwrwindow, init(R,'',wnnonumber,win));
  insertwindow(win3);
    active;
  take;

End;

{*********************************************************************}

Procedure Tinvcowr.InitStatusLine;
Begin
  StatusLine := PStatusLine(corres.Get('STAT'));
  desktop^.GetExtent(R);
  StatusLine^.MoveTo(0, R.B.Y - 1);
End;

{*********************************************************************}

Procedure Tinvcowr.InitMenuBar;
Begin
  desktop^.getextent(r);
  MenuBar := PMenuBar(corres.Get('MENU'));
End;

{*********************************************************************}

Procedure TInvCoWr.InitDesktop;
begin
  GetExtent(R);
```

```
  R.Grow(0, -1);
  Desktop := New(PcrwrDesktop, Init(R));
end;

{*************************************************************************}

Procedure TinvCoWr.GetEvent(var Event: TEvent);
var
  W: PWindow; HFile: PHelpFile; HelpStrm: PDosStream;
const
  HelpInUse: Boolean = False;
begin
  inherited GetEvent(Event);
  case Event.What of
    evCommand:
      if (Event.Command = cmHelp) and not HelpInUse then
      begin
        HelpInUse := True;
        HelpStrm := New(PDosStream, Init('HCINVT.HLP', stOpenRead));
        HFile := New(PHelpFile, Init(HelpStrm));
        if HelpStrm^.Status < > stOk then
        begin
          MessageBox('Could not open help file.', nil, mfError + mfOkButton);
          Dispose(HFile, Done);
        end
        else
        begin
          W := New(PHelpWindow,Init(HFile, GetHelpCtx));
          if ValidView(W) < > nil then
          begin
            ExecView(W);
            Dispose(W, Done);
          end;
          ClearEvent(Event);
        end;
        HelpInUse := False;
      end;
      {evMouseDown:
      if Event.Buttons < > 1 then Event.What := evNothing;}
  end;
End;

{*************************************************************************}

Function TINVCOwR.GetPalette: PPalette;
Const
  CNewColor = CAppColor + CHelpColor;
  CNewBlackWhite = CAppBlackWhite + CHelpBlackWhite;
  CNewMonochrome = CAppMonochrome + CHelpMonochrome;
  P: array[apColor..apMonochrome] of string[Length(CNewColor)] =
  (CNewColor, CNewBlackWhite, CNewMonochrome);
Begin
  GetPalette := @P[AppPalette];
End;


{*************************************************************************}

Procedure TInvCoWr.HandleEvent(var event: tevent);
var
s:string;
Begin
```

```
inherited handleevent(event);
If Event.What = evcommand then
begin
 CASE Event.Command OF
   cmMODULE1 :
         begin
         Introduction;
         ClearEvent(Event);
         end;

   cmMODULE2 :
       begin
         dialogkey:=Messagebox(#3'Advise ! '#13#3+
                'To Go to This Module'#13#3+
                'It would be better if you first'#13#3+
                'complete MODULE 1 ',nil,
                 mfConfirmation or mfOKButton +mfNoButton);
         if dialogkey = 10 then
            Introduction
         else
            planning;
          ClearEvent(Event);
       end;
   cmMODULE3 :
      begin
       dialogkey:=Messagebox(#3'Advise ! '#13#3+
             'To Go to This Module'#13#3+
             'It would be better if you first'#13#3+
             'complete MODULE 2 ',nil,
              mfConfirmation or mfOKButton +mfNoButton);
         if dialogkey = 10 then
            planning
         else
            forecast;
          ClearEvent(Event);
        end;
  cmMODULE4 :
      begin
       dialogkey:=Messagebox(#3'Advise ! '#13#3+
             'To Go to This Module'#13#3+
             'It would be better if you first'#13#3+
             'completed MODULE 3 ',nil,
              mfConfirmation or mfOKButton +mfNoButton);
         if dialogkey = 10 then
            forecast
         else
            method;
          ClearEvent(Event);
        end;
  cmMODULE5..cmModule24:
       BEGIN
        s:=obj[event.command];
        PROTO(s);
        clearevent(event);
       end;
  cmOptionsVideo:
     begin
       SetScreenMode(ScreenMode xor smFont8x8);
       ClearEvent(Event);
     end;
  cmAbout:
    begin
```

```
            DoAboutBox;
            ClearEvent(Event);
          end;
    end;
 end;
 bring;
End;

{*************************************************************************}
Procedure Tinvcowr.OutOfMemory;
var
 cr1:string;
begin
  str(memavail,Cr1);
  MessageBox(#3'Available memory is '#13#3+cr1+
  '#13#3 this is not enough to finish the task.',
    nil, mfError + mfOKButton);
End;
{*************************************************************************}

Procedure tinvcowr.active;
begin;
   event.what:=evkeyboard;
   event.charcode:=#13;
   event.scancode:=28;
   putevent(event);
   Event.what:=evcommand;
   event.command:=desktop^.execview(menubar);
   if event.command<>0 then
   putevent(event);
end;

{*************************************************************************}

Procedure Tinvcowr.Take;
Begin
  Desktop^.getextent(R);
  Inc(R.b.y,2);
  Desktop^.locate(R);
  r.assign(0,-1,80,-2);
  Menubar^.locate(r);
  r.assign(1,26,80,27);
  Statusline^.locate(r);
End;

{*************************************************************************}

Procedure Tinvcowr.Bring;
Var
r:trect;
Begin
 r.assign(0,0,80,1);
 Menubar^.locate(r);
 r.assign(0,24,80,25);
 Statusline^.locate(r);
End;

{*************************************************************************}

Procedure TInvCoWr.DoAboutBox;
var s1:string;
Begin
```

```
  show1('LS',44,48);
  active;
 End;


 {*************************************************************************}

 Procedure Tinvcowr.Inform;
 var
   Cr1,Cr2 : string;
 begin
  str(memavail,Cr1);
  str(Dialogkey,cr2);
  MessageBox(#3'I am here with memory of'#13#3' '+Cr1+'and Dialogkey of '+ Cr2,nil,
  mfInformation or mfOKButton);
 end;

 {*************************************************************************}

 Procedure Tinvcowr.initial(var s1:string);

 begin
  pwsht:=powsheet(corres1.get(S1));
  opentest1;
  r.assign(20,18,60,24);
  dialogkey:= MessageBoxRect(R,#3' Next Question  ? ',
          nil, mfinformation or mfyesbutton + mfNobutton );
  pwwsht^.close;
 end;

 {*************************************************************************}

 Procedure Tinvcowr.final(var s1:string);
 begin

  pwsht:=powsheet(corres1.get(s1));
  opentest1;
  r.assign(20,18,60,24);
  pwwsht^.close;
  dialogkey:= MessageBox(#3' This ends your test'#13+
                #3'Want to see your result ? ',
        nil, mfinformation or mfyesbutton + mfNobutton );

 end;

 {*************************************************************************}

 procedure Tinvcowr.call;
 begin

     New(Prwsht); Prwsht^.n := pwsht^.two.n;
     prwsht^.replay := pwsht^.two.replay;
     prwsht^.inp:= pwsht^.two.inp;
     prwsht^.right:=pwsht^.two.right;
     prwsht^.wrong:= pwsht^.two.wrong;
     R.Assign(1,2,43,17);
     pwwsht:= New(pwinwsheet, init(R,'',wnnonumber,prwsht));
     insertwindow(pwwsht);
 end;

 {*************************************************************************}
```

```
Procedure Tinvcowr.right;
begin
    dispose(pwsht,done);
    FOR I:=1 TO 6
    DO
    BEGIN
     sound(150+i*i);delay(40+i);nosound;
    {sound(650);delay(600);nosound;
  {   sound(450);delay(400);nosound;}
    END;
    r.assign(1,2,43,17);
    PWWSHT^.writestr(5,10, PRWSHT^.RIGHT,10);delay(3000);
end;

{************************************************************************}

Procedure Tinvcowr.Wrong;
begin
    dispose(pwsht,done);
     sound(100);delay(300);sound(150);delay(350);nosound;
    r.assign(1,2,43,17);
    PWWSHT^.writestr(1,10, PRWSHT^.wrong,2);delay(3000);
End;

{************************************************************************}

Procedure TInvcowr.Opentest1;
var
 ttest1:Ptest1; ttest2:ptest2;ttest3:ptest3;
Begin
 case tc of
  1: begin
     call;
     TTEST1:= new(Ptest1,init);
     InsertWindow(TTEST1);
     with test1info do  begin   Ans1:=' '; end;
     Ptest1info := @test1info;
     dialogkey:=Executedialog(ttest1,Ptest1info);
     if  dialogkey<> 102 then
     begin
      if (upercase(Ptest1info^.ans1)= prwsht^.inp[1])or
      (upercase(Ptest1info^.ans1)= prwsht^.INP[2])then
      BEGIN
         count:=count+1;right;  end
      else
      begin
         count:=count+0;wrong; End;
      END;
   End;

  2 :begin
     CALL;
     TTEST2:= new(Ptest2,init);
     InsertWindow(TTEST2);
     with test2info do  begin   Ans1:=' '; Ans2:=' '; end;
     Ptest2info := @test2info;
     dialogkey:=Executedialog(ttest2,Ptest2info);
     if  dialogkey<> 102 then
     begin
      if((upercase(Ptest2info^.ans1)= prwsht^.inp[1]) and
        (upercase(Ptest2info^.ans2)= prwsht^.inp[2])) or
        ((upercase(Ptest2info^.ans1)= prwsht^.inp[2])and
```

```
          (upercase(Ptest2info^.ans2)= prwsht^.inp[1])) then
         BEGIN
          count:=count+2;RIGHT; end
        else
        begin
          count:=count+0;WRONG; End;
     END;
    End;

  3 :begin
     call;
     TTEST3:= new(Ptest3,init);
     InsertWindow(TTEST3);
     with test3info do begin  Ans1:=' '; end;
     Ptest3info := @test3info;
     dialogkey:=Executedialog(ttest3,Ptest3info);
     if dialogkey< > 102 then
     begin
      if (upercase(Ptest3info^.ans1)= prwsht^.inp[1]) then
      BEGIN
        count:=count+1;RIGHT; end
      else
      begin
        count:=count+0;WRONG; End;
    END;
   End;
End;{CASE}
End;{PROCEDURE}


{*************************************************************************}

Procedure Tinvcowr.proto(var s:string);
var
  screen1 : crwrscrn; C,l,d: integer;
begin
   l:=26;
   c := -18;
   screen1 := crwrscrn(corres.get(s));
   screen1^.moveto(L,c);
   insertwindow(screen1);
   for d:= 1 to 10 do
   begin
    sound(100*d);
    delay(200);
    sound(400*d*2);
    delay(30);
    nosound;
   end;
   while (l > 2) and (c < 6)do
   begin
    screen1^.moveto(l,c);
    delay(3);
    c := c+1;
    l:=l -1;
   end;
   Dialogkey := ExecuteDialog(screen1,nil);
   if Dialogkey < > 102 then
      Messagebox(#3'Sorry ! You can not proceed beyond This'#13#3+
           'This Part Is Under Development',nil,mfInformation or mfOKButton);
select;
active;
```

```
End;

{*************************************************************************
procedure Tinvcowr.EventError(var Event: TEvent);
begin
 write(#7);
 inherited eventerror(event);
  MessageBox(#3'Event Handling Problem!!'#13#3+' You have to enter the '
  + #13#3'options provided in the dialog box only ',nil
  ,mfError or mfOKButton);
end;

{*************************************************************************}

procedure TINVCOWR.inputr( k: byte; A1,A2,A3:string; VAR K3: INTEGER);
var
  i:string;  r:trect;
begin
  K3:=0;
  repeat
   i:='        ';
   R.assign(10,15,65,23);
   InputBoxRect(r,'ANSWER', 'ENTER YOUR ANSWER', i,k);
   K3:= K3+1;
  until(upercase(i)=A1) or (upercase(i)=A2) OR (K3=2);
end;


{*************************************************************************}

procedure inputb(k:byte;A1:string);
begin
    InputBox('ANSWER', 'You could have entered either', A1,k);
end;

{*************************************************************************}

Procedure Tinvcowr.objectve(var s1: string);
var
l,c: integer;screen1:crwrscrn;
R:TRECT;
begin
  l:=26; c := -18; screen1 := crwrscrn(corres.get(s1));
  screen1^.moveto(L,c);
  insertwindow(screen1);
  while (l > 2) and (c < 6)do
  begin
   screen1^.moveto(l,c);
   delay(15); nosound; c := c+1; l:=l -1;
  end;
   Dialogkey := ExecuteDialog(screen1,nil);
end;


{*************************************************************************}

procedure Tinvcowr.branch(var s1, s2,s3 :string);
VAR
 x:string;
 screen1,screen2:crwrscrn;
 c,l:integer;
```

```
begin
 repeat
   Dialogkey := ExecuteDialog(PDialog(corres.Get(s1)), nil);
   if (Dialogkey = 103)or (Dialogkey=104) then
   begin
    if (Dialogkey=103) then  s2:=s2
    else
     if (Dialogkey = 104) then
      s2:=s3; screen1 := crwrscrn(corres.get(s1));
      screen1^.moveto(1,2);insertwindow(screen1);
      c:=-15;l:=-15;screen2 := crwrscrn(corres.get(s2));
      screen2^.moveto(c,l); insertwindow(screen2);
      while  (c <=  10) and (l <= 10) do
      begin
        screen2^.moveto(c,l); delay(1); c := c+1; l :=l+1;
      end;
      Dialogkey := ExecuteDialog(screen2,nil); screen1^.close;
  end;
 until (Dialogkey <>103);
End;
{*********************************************************************}

Procedure tinvcoWr.show1(s: string; k,m: word);
Var
  screen1: CRWRSCRN; i: word;  s1 : string[3];
 begin
  repeat
    str(k,s1);
    screen1 := crwrscrn(corres.get('LS'+s1));
     SCREEN1^.MOVETO(100,2);
     insertwindow(screen1);
    for i := 100 downto 2 do
     begin screen1^.moveto(I,2); delay(5); end;
        Dialogkey := ExecuteDialog(screen1,nil);
    k := k+1;
  Until (Dialogkey = 102) or (k = m+1);
end;

{*********************************************************************}

Procedure Tinvcowr.select;
var ttest3:ptest3;
begin
dialogkey:= MessageBox(#3' Do you like to proceed '#13#3+
            ' to other module ?',nil, mfinformation or mfYesButton+mfNoButton);
if dialogkey <> 13 then
begin
 pwsht:=powsheet(corres1.get('TS9'));
 call;
 r.assign(20,18,60,24);
 TTEST3:= new(Ptest3,init);InsertWindow(TTEST3);
 with test3info do begin  Ans1:=' '; end;
 Ptest3info := @test3info;
 dialogkey:=Executedialog(ttest3,Ptest3info);
 if  dialogkey<> 102 then
 BEGin
  pwwsht^.close;
  if (upercase(Ptest3info^.ans1)= prwsht^.inp[1]) THEN INTRODUCTION;
  IF (upercase(Ptest3info^.ans1)= prwsht^.INP[2]) then PLANNING;
  IF (upercase(Ptest3info^.ans1)= prwsht^.INP[3]) then FORECAST;
  if (upercase(Ptest3info^.ans1)= prwsht^.INP[4]) then Method;
 END;
```

```
end;
end;

Procedure TInvCoWr.Introduction;
var
 screen1,screen2,screen3,screen4 : crwrscrn;
 screen5:PCRWREXCR; l,c,D,K3: integer; k1:byte;
 I,S1,S2,s3,cr2:string;
begin
 take;
 screen4:= crwrscrn(corres.get('IS5'));
 screen4^.moveto(-10,-10);insertwindow(screen4);
 for c:= -10 to 20  do
 begin
   screen4^.moveto(1,c);delay(100);
 end;
 Take;
 s1:='OS1'; objectve(s1);
 if Dialogkey < > 102 then
 begin
   screen1 := crwrscrn(corres.get('LS1'));
   screen1^.moveto(1,2);insertwindow(screen1);
   inputr(9,'INVENTORY','STOCK',' ',K3);
   if k3 = 2 then
   inputb(100, 'Inventory or stock as a correct answer');
   screen1^.close;
 end;
 if Dialogkey < > 102 then
 begin
  s1:='LS2'; S2:='HS1';s3:=' '; BRANCH(S1,S2,s3);
 END;
 if Dialogkey < > 102 then Dialogkey:= ExecuteDialog(PDialog(corres.Get('LS3')), nil);
 if Dialogkey < > 102 then
 begin; s1:='LS4'; S2:='HS2'; s3:='HS3';
   branch(s1,s2,s3); end;
 if Dialogkey< >102 then Dialogkey:= ExecuteDialog(PDialog(corres.Get('LS5')), nil);
 if Dialogkey< > 102 then
 begin
  if Dialogkey = 101 then
  begin
    screen1 := crwrscrn(corres.get('LS5'));
    screen1^.moveto(1,2);
    insertwindow(screen1);
    screen2 := crwrscrn(corres.get('LS6'));
    screen2^.moveto(2,4);
    insertwindow(screen2);
    for c:= 2 to 44  do
      screen2^.moveto(c,4);
    Dialogkey:= ExecuteDialog(screen2, nil);screen1^.close;
  end
  ELSE
  begin
    screen1:= crwrscrn(corres.get('HS5'));
    insertwindow (screen1);
    dialogkey:= executedialog (crwrscrn(corres.get('HS6')),nil);
    screen1^.close;
  end;
 END;
 if Dialogkey< > 102 then dialogkey:= ExecuteDialog(crwrscrn(corres.Get('IS9')), nil);
 if Dialogkey < > 102 then
 begin
   tc:=1; count:=0; S1:='TS1';
```

```
      INITIAL(S1);
      if dialogkey = 12 then
       begin tc:=1; s1:='TS2'; initial (S1); end;
      if dialogkey = 12 then
       begin tc:=2;  S1:='TS3';Initial(s1);  end;
     if dialogkey = 12 then
      begin tc:=3;  s1:='TS4'; final(s1);   end;
     if dialogkey = 12 then
     begin
      if count > = 3 then
       begin
        count:=count*10; str(count,cr2);
        dialogkey:= MessageBox(#3' Congradulation ! '#13#3+
                  'You Scored '+cr2+ ' out of 50'#13#3+
                  'Want a reward ? '
                   ,nil, mfConfirmation or mfYesbutton + mfNobutton);
       if dialogkey = 12 then
       begin   play; select; end;
       if dialogkey < > 13 then
       begin  screen4^.close; forecast;  end;
      end
     else
       begin
        count:=count*10;write(#7); str(count,cr2);
        dialogkey:=MessageBox(#3' You score only '+cr2+ ' out of 50'#13#3+
                'It is advised that you repeat This'#13#3+
                'lesson once, more attentively ! '
                , nil, mfconfirmation or mfYesButton+mfNOButton);
        if dialogkey = 10 then Introduction
        else
          select;
       end;
     end;
    end;

SCREEN4^.CLOSE;
select;
active;
End;


{*****************************************************************************}

Procedure TInvCoWr.Planning;
var
  screen1,screen2,screen3,screen4,SCREEN5 : crwrscrn;
  l,c,D,K3:integer; k1:byte; I,S1,S2,s3,cr2:string;
  ttest3:ptest3;
begin

 take;
 screen4:= crwrscrn(corres.get('IS6'));
 screen4^.moveto(-10,-10);insertwindow(screen4);
 for c:= -10 to 20  do
 begin
   screen4^.moveto(1,c);delay(100);
 end;
 Take;
  s1:='OS2'; objectve(s1);if dialogkey < > 102 then
  begin s1:='LS7'; S2:='HS7'; s3:=' '; branch(s1,s2,s3); end;
  if Dialogkey < > 102 then
  begin
```

```
 s1:='LS8'; S2:='HS8'; s3:=' '; branch(s1,s2,s3);
end;
if Dialogkey<>102 then Dialogkey:= ExecuteDialog(PDialog(corres.Get('LS10')), nil);
if Dialogkey<>102 then
begin
 screen1:=crwrscrn(corres.get('LS10'));
 insertwindow(screen1); screen2:=crwrscrn(corres.get('LS11'));
 insertwindow(screen2);
 for l:=1 to 10 do
 begin
  screen2^.moveto(4,l); delay(5);
 end;
 Dialogkey:= ExecuteDialog(screen2, nil); screen1^.close;
end;
if Dialogkey<>102 then
BEGIN
   Dialogkey:= ExecuteDialog(crwrscrn(corres.get('HS9')),nil);
   if dialogkey <>102 then  screen1:=crwrscrn(corres.get('HS9'));
   insertwindow(screen1);
   Dialogkey:= ExecuteDialog(crwrscrn(corres.get('HS11')),nil);
   if dialogkey <>102 then screen2:=crwrscrn(corres.get('HS11'));
   insertwindow(screen2);
   Dialogkey:= ExecuteDialog(crwrscrn(corres.get('HS10')),nil) ;
   screen1^.close;screen2^.close;
END;
if Dialogkey<>102 then dialogkey:=ExecuteDialog(PDialog(corres.Get('LS12')), nil);
if Dialogkey<>102 then dialogkey:=ExecuteDialog(PDialog(corres.Get('IS10')), nil);
if Dialogkey<>102 then
begin
  tc:=1;
  count:=0; S1:='TS5';INITIAL(S1);
  if dialogkey = 12 then
  begin
    tc:=3; S1:='TS6';INITIAL(S1); end;
    if dialogkey = 12 then
     begin
     tc:=3;s1:='TS7';INITIAL(S1);  end;
   if dialogkey = 12 then
   begin
    tc:=1;S1:='TS8';final(s1);end;
    if dialogkey = 12 then
    begin
     if count >= 2 then
     begin
      count:=count*10; str(count,cr2);
      MessageBox(#3' Hey ! This is Good ! '#13#3+
              'You Score '+cr2+ ' out of 40'#13#3+
               'Want a reward ? '
               ,nil, mfConfirmation or mfYesbutton + mfNobutton);
    if dialogkey = 12 then begin     play;  select; end;

    if dialogkey <> 13 then
    begin  screen4^.close; forecast;  end;
   end
   else
    begin
      count:=count*10;write(#7); str(count,cr2);
      dialogkey:=MessageBox(#3' You score only '+cr2+ ' out of 40'#13#3+
             'It is advised that you repeat This'#13#3+
             'lesson once, more attentively ! '
             , nil, mfconfirmation or mfYesButton+mfNOButton);
      if dialogkey = 10 then planning
```

```
        else
          select;
        end;
      end;
    end;

SCREEN4^.CLOSE;
select;
active;
End;

{*****************************************************************************}

procedure TINVCOWR.forecast;
var
  screen1,screen2,screen3,screen4 : crwrscrn;
  l,c,D,K3 : integer; k1:byte; I,S1,S2,s3,cr2:string;
  r:trect;  replaykey: word;
begin
  take;
  screen4:= crwrscrn(corres.get('IS7'));
  screen4^.moveto(-10,-10);insertwindow(screen4);
  for c:= -10 to 20  do
  begin
   screen4^.moveto(1,c);delay(100);SOUND(C);delay(20);NOSOUND;
 end; DELAY(50);
  s1:='OS3';
  objectve(s1);
  if dialogkey < > 102 then

  Dialogkey:=ExecuteDialog(PDialog(corres.Get('LS13')), nil);
  if Dialogkey < > 102 then  Dialogkey:=ExecuteDialog(PDialog(corres.Get('LS14')), nil);
  if Dialogkey < > 102 then  Dialogkey:=ExecuteDialog(PDialog(corres.Get('HS12')), nil);
  if Dialogkey < > 102 then
  begin
     screen1:=crwrscrn(corres.get('HS12'));
     insertwindow(screen1);
     Dialogkey:=ExecuteDialog(PDialog(corres.Get('HS13')), nil);
     screen1^.close;
  end;
 if Dialogkey < > 102 then
  DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('HS13')), nil);
  if Dialogkey< > 102 then
  DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('IS11')), nil);
  if Dialogkey< > 102 then
  begin
    tc:=3;
    count:=0; S1:='TS10';INITIAL(S1);
    if dialogkey = 12 then
    begin
      tc:=3; S1:='TS11';INITIAL(S1); end;
      if dialogkey = 12 then
       begin
       tc:=2;s1:='TS12';Final(S1);  end;
     if dialogkey = 12 then
     begin
      if count > = 2 then
      begin
       count:=count*10; str(count,cr2);
       MessageBox(#3' Hey ! This is Good ! '#13#3+
              'You Score '+cr2+ ' out of 40'#13#3+
               'Want a reward ? '
```

```
                ,nil, mfConfirmation or mfYesbutton + mfNobutton);
      if dialogkey = 12 then begin    play;  select; end;

      if dialogkey < > 13 then
      begin  screen4^.close; method;  end;
     end
    else
     begin
      count:=count*10;write(#7); str(count,cr2);
      dialogkey:=MessageBox(#3' You score only '+cr2+ ' out of 40'#13#3+
            'It is advised that you repeat This'#13#3+
            'lesson once, more attentively ! '
            , nil, mfconfirmation or mfYesButton+mfNOButton);
      if dialogkey = 10 then Forecast
      else
       select;
     end;
   end;
  end;

SCREEN4^.CLOSE;
select;
active;
End;

{*****************************************************************************}

Procedure Tinvcowr.Method;
var
  screen1,screen2,screen3,screen4 : crwrscrn;
  l,c,D,K3 : integer; k1:byte; I,S1,S2,s3,cr2:string;
   replaykey: word;
   ttest4:ptest4; ttest5:ptest5;ttest6:ptest6;ttest7:ptest7;ttest8:ptest8;
begin
  take;
  screen4:= crwrscrn(corres.get('IS8'));
  screen4^.moveto(-10,-10);insertwindow(screen4);
  for c:= -10 to 20  do
  begin
   screen4^.moveto(1,c);
   delay(100);SOUND(C);
   delay(20);NOSOUND;
  end;
  DELAY(50);
  s1:='OS4';
  objectve(s1);
  if dialogkey < > 102 then
   show1('LS',15,28);
  if Dialogkey < > 102 then
  BEGIN
   twin:=pcrinvtext(corres1.get('WS2'));
   New(win); win^.n := twin^.tinv1.n;
   win^.cinvt := twin^.tinv1.cinvt;
   R.Assign(-2,0,81,26);
   win2:= New(pcrwrwindow, init(R,'',wnnonumber,win));
   insertwindow(win2);
   DIALOGKEY:= ExecuteDialog(PCRWREXCR(corres.Get('LS29')), nil);
  END;
  if Dialogkey < > 102 then DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS30')), nil);
  if dialogkey< > 102 then
 BEGIN
   dispose(tWIN,done);
```

```
    twin:=pcrinvtext(corres1.get('WS3'));
    New(win);
    win2^.tinv^.n := twin^.tinv1.n;
    win2^.tinv^.cinvt := twin^.tinv1.cinvt;
    win2^.draw; DELAY(100);
    DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS31')), nil);
  end;
  if dialogkey<>102 then
   DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS311')), nil);
  if dialogkey<>102 then
   DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS32')), nil);
  if dialogkey<>102 then
   begin
    DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS33')), nil);
    twin1:=pcrinvtext(corres1.get('WS4'));
    win2^.writestr(10,14,twin1^.tinv1.cinvt[1].phrase,6);
    win2^.writestr(10,15,twin1^.tinv1.cinvt[1].phrase,6);
    DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS34')), nil);
   end;
   if dialogkey<>102 then
   DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS35')), nil);
   if dialogkey<>102 then
   DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS36')), nil);
   if dialogkey<>102 then
    DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS37')), nil);
   if dialogkey<>102 then
    begin
     dispose(tWIN,done);
     dispose(twin1,done);
     twin:=pcrinvtext(corres1.get('WS5'));
     New(win);
     win2^.tinv^.n := twin^.tinv1.n;
     win2^.tinv^.cinvt := twin^.tinv1.cinvt;
     win2^.draw;
     DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS38')), nil);
    end;
  if dialogkey<>102 then
    begin
     twin1:=pcrinvtext(corres1.get('WS6'));
     New(win);
     win2^.tinv^.n := twin1^.tinv1.n;
     win2^.tinv^.cinvt := twin1^.tinv1.cinvt;
     win2^.draw;
     DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS39')), nil);
    end;
if dialogkey<>102 then DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS40')), nil);
if dialogkey<>102 then DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS41')), nil);
if dialogkey<>102 then
    begin
     dispose(tWIN,done);
     dispose(twin1,done);
     twin:=pcrinvtext(corres1.get('WS7'));
     New(win);
     win2^.tinv^.n := twin^.tinv1.n;
     win2^.tinv^.cinvt := twin^.tinv1.cinvt;
     win2^.draw; DELAY(100);
     DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS42')), nil);
    end;
 if dialogkey<>102 then
    {*BEGIN                                    }
    dialogkey:=ExecuteDialog(PDialog(corres.Get('LS43')), nil);
```

```
  if dialogkey < > 102 then

   DIALOGKEY:= ExecuteDialog(PDialog(corres.Get('LS44')), nil);

 if dialogkey < > 102 then
   begin
   win2^.close;
    pwsht:=powsheet(corres1.get('TS13'));
     r.assign(20,16,60,24);
     New(Prwsht);
    Prwsht^.n := pwsht^.two.n;
    prwsht^.replay := pwsht^.two.replay;
    prwsht^.inp:= pwsht^.two.inp;
    prwsht^.right:=pwsht^.two.right;
    prwsht^.wrong:= pwsht^.two.wrong;
    R.Assign(5,2,70,15);
    pwwsht:= New(pwinwsheet, init(R,'',wnnonumber,prwsht));
    insertwindow(pwwsht);
    TTEST4:= new(Ptest4,init);
    InsertWindow(TTEST4);
    with test4info do begin  Ans1:=' '; Ans2:=' '; ANS3:=' '; end;
    Ptest4info := @test4info;
    dialogkey:=Executedialog(ttest4,Ptest4info);
   end;
    if  dialogkey < > 102 then
    New(Prwsht);
    Prwsht^.n := pwsht^.two.n;
    prwsht^.replay := pwsht^.two.replay;
    prwsht^.inp:= pwsht^.two.inp;
    prwsht^.right:=pwsht^.two.right;
    prwsht^.wrong:= pwsht^.two.wrong;
    R.Assign(1,2,43,17);
    pwwsht:= New(pwinwsheet, init(R,'',wnnonumber,prwsht));
    insertwindow(pwwsht);
 if dialogkey  < >102 then
 begin

    pwwsht^.close;
    TTEST5:= new(Ptest5,init);
    InsertWindow(TTEST5);
    with test5info do begin  Ans1:=' '; Ans2:=' '; ANS3:=' '; Ans4:=' ';end;
    Ptest5info := @test5info;
    dialogkey:=Executedialog(ttest5,Ptest5info);
 end;
 if  dialogkey < > 102 then
begin
    screen1:= crwrscrn(corres.get('LS46'));
    insertwindow(screen1);
    TTEST6:= new(Ptest6,init);
    InsertWindow(TTEST6);
    with test6info do  begin  Ans1:=' '; Ans2:=' '; ANS3:=' ';end;
    Ptest6info := @test6info;
    dialogkey:=Executedialog(ttest6,Ptest6info);
    screen1^.close;
end;
if dialogkey  < > 102 then
    dialogkey:= executedialog(PDialog(corres.Get('LS47')), nil);
if dialogkey  < > 102 then
 begin
    screen1:= crwrscrn(corres.get('LS47'));
    insertwindow(screen1);
    TTEST7:= new(Ptest7,init);
```

```
    InsertWindow(TTEST7);
    with test7info do  begin   Ans1:=' ';  Ans2:=' ';end;
    Ptest7info := @test7info;
    dialogkey:=Executedialog(ttest7,Ptest7info);
    screen1^.close
  end;
if dialogkey <>102 then
 begin
     TTEST8:= new(Ptest8,init);
     InsertWindow(TTEST8);
     with test8info do  begin   Ans1:=' '; end;
     Ptest8info := @test8info;
     dialogkey:=Executedialog(ttest8,Ptest8info);
     if dialogkey <> 102 then
     begin
    dialogkey:=executedialog(PDialog(corres.Get('LS49')), nil);
   if dialogkey<> 102 then
    dialogkey:= ExecuteDialog(PDialog(corres.Get('IS12')), nil);
   if  dialogkey<> 102 then
    begin
   tc:=2;
   count:=0; S1:='TS14';INITIAL(S1);
   if dialogkey = 12 then
   begin
     tc:=3; S1:='TS15';INITIAL(S1); end;
     if dialogkey = 12 then
      begin
      tc:=1;s1:='TS16';initial(S1);  end;
    if dialogkey = 12 then
      begin
      tc:=3;s1:='TS17';initial(S1);  end;
    if dialogkey = 12 then
    begin
      tc:=3;s1:='TS18';final(S1);  end;
    if dialogkey = 12 then
    begin
      if count >= 3 then
      begin
       count:=count*10; str(count,cr2);
       MessageBox(#3' Hey ! This is Good ! '#13#3+
                'You Score '+cr2+ ' out of 60'#13#3+
                'Want a reward ? '
                ,nil, mfConfirmation or mfYesbutton + mfNobutton);
    if dialogkey = 12 then begin    play;  select; end;

    if dialogkey <> 13 then
     screen4^.close;
   end
   else
    begin
     count:=count*10;write(#7); str(count,cr2);
     dialogkey:=MessageBox(#3' You score only '+cr2+ ' out of 60'#13#3+
            'It is advised that you repeat This'#13#3+
            'lesson once, more attentively ! '
            , nil, mfconfirmation or mfYesButton+mfNOButton);
     if dialogkey = 10 then method
     else
       select;
    end;
  end;
 end;
```

```
SCREEN4^.CLOSE;
select;
active;
End;


Destructor Tinvcowr.done;
Begin
 inherited done;
 corres.done;
End;

{*****************************************************************************}

var
 TInv: TInvCoWr;
begin
 front;
 TInv.Init;
 TInv.Run;
 TInv.Done;
end.
```

DECLARATION

The thesis is my orginal work and has not been presented for a degree in any other university.

(Signed)

Alemayehu Molla Adankew

May 24, 1995

The thesis has been submitted for examination with our approval as university advisors.

(Signed)

Dr. Taye Tadesse

May 24, 1995

(Signed)

Ato Tesfaye Biru

May 24, 1995