e) Incremental model

With this approach, a model is designed, implemented and tested incrementally until the software is completed.

CHAPTER 6

SYSTEM DEVELOPMENT LIFE CYCLE STAGES

- ✓ PROBLEM DEFINITION/PRELIMINARY INVESTIGATION

It is the activity of identifying the problem and understanding any constraints that may limit the solution.

Preliminary system study/investigation involves the preparation of a system proposal which lists the problem definition, objectives of the study, and terms of reference for study, constraints and expected benefits of the new system in the light of the user requirements.

The system proposal is prepared by the system analyst and places it before the management. The management may accept the proposal and the cycle proceeds to the next stage or may reject the proposal or request some modifications in the proposal.

TERM OF REFERENCE (TOR)

It is the document that describes an assignment for an individual or a team.

It helps one to know who initiated the project, objectives of the project and benefits of the project.

TOR is a document that acts as a reference throughout system development stages.

Contents of TOR

- Personnel involved
- Subject involved
- Available resources
- Estimated duration for developing the system.
- Estimated cost of development and implementation
- Departments that are affected during system implementation.

PROBLEM STATEMENT

A problem statement defines a description of issues and problems that require to be solved.

Example

The college student registration system is unable to cope with the high volume of telephone calls received at registration time.

Signals and long distance charges are the problem of the telephone registration system.

An online registration system needs to be developed.

Students on campus, off campus and out the country can easily and in expensively take advantage of the many services provided by the college.

At the end of this phase, a preliminary study report or system proposal is produced and presented to the sponsors for approval.

Reasons for introducing a new system

- The current system may no longer be suitable for it purpose.
  Changes in work process expansion of the business, changes in business requirement or the environment in which the organization operates may all lead to reassessment of information system requirements.
- Current system may be too inflexible or expensive to maintain.
- Technology development may have made the current system outdated.

✓ FEASIBILITY STUDY

It is the measure of how beneficial the development of an information system will be to an organization.

It assesses the operational, technical and economic merit of the proposed system.

The aim of feasibility study is to understand the problem and determine whether it is worth proceeding.

Types of feasibility study

- **Economic feasibility**

It is a measure of the cost-effectiveness of a project.

It is an evaluation of all incremental cost and benefits.

- **Technical feasibility**

It means investigating whether the technology exists to implement the proposed system or whether there is a practical solution.

It is a measure of the practicality of a specific technical solution and the availability of technical resources and expertise.

- **Operational feasibility**

It is a measure of how well people feel about a system or a project.

It is a measure of how well a system will work in an organization.

Operational feasibility is concerned with whether the current work practices and procedures are adequate to support the new system.

- **Schedule feasibility/time**

It is a measure of how long it will take the new system to become operational. It looks at how long it will take to develop a system or whether it can be done in a desired time frame.

- **Legal feasibility**

Determines whether there is any conflict between the proposed system and the legal requirements.

Example: legal feasibility study, will determine whether the system violates data protection act.

- **Social/behavioral feasibility**

It is a measure of how the system will affect the behavior of people.

The completion of this phase is marked by the production of a feasibility study report produced by the systems analyst.

If the report concludes that the project should go ahead and this is agreed by the senior managers, detailed requirements analysis will proceed.

- ✓ **SYSTEM ANALYSIS**

It is the process of gathering and interpreting facts and using the information to recommend improvements to the system.

It is also a process of examining business situation in order to develop a system to solve existing problems.

Fact finding methods/techniques/data collection methods

1. Review of Written Documents: The first step to gather information of a system is to Search and review the various forms of written documents such as professional references, Procedure manuals, textbooks, company studies, government publications, consultant Studies, etc. of that system.
2. On-site observations: The major objective of on-site observation is to get close to the real system being studied. The methods used may be natural or contrived, obtrusive or Unobtrusive, direct or indirect, and structured or unstructured. The main limitation of observation is the difficulty of observing attitudes and motivation and the many unproductive hours that often are spent in observing one-time activities.
3. Interviews: It is a face-to-face interpersonal meeting designed to identify relations and Capture information as it exists. It is flexible tool, offering a better opportunity than the Questionnaire to evaluate the validity of the information gathered. The major drawback is Preparation time. Interviewing is an art that requires experience in arranging the Interview, setting the stage, establishing rapport, phrasing questions clearly, avoiding arguments, and evaluating the outcome.
4. Questionnaires: It is a self-administered tool that is more economical and requires less skill to administer than the interview. It examines a large number of respondents at the same time, provides standardized wording and instructions, and places less pressure on Subjects for immediate response. The main drawback is the low percentage of returns.

- ✓ **SYSTEM DESIGN**

It is the process of defining the architecture, components, modules, interfaces and data for assisting to satisfy specified requirements.

It is also the process of designing and developing systems to satisfy specified requirements of the user.

System design is based on the user requirements and the detailed analysis of the existing system.

Types of design

1. Logical design
2. Physical design

Logical design

It is a graphical representation of a system showing the system processes and the flow of data into and out of the processes.

To represent the logical design of a system, the system analyst can use different diagrams like: flowcharts, data flow diagrams (DFD), entity relation diagrams(ER) etc.

Physical design

It is a graphical representation of a system showing the system's internal and external entities and the flow of data into and out of the entities.

An internal entity is an entity within the system that transforms data.

An internal entity can be a person, place or thing.

The physical design part can be broken down into 3 sub-tasks

- User interface design
- Data design
- Process design

User interface design

Concerned with how users add information to the system and with how the system presents information back to them.

Data design

Concerned with how data is represented and stored within the system.

Process design

Concerned with how data move within the system and with how/where it is validated, secured or transformed as it flows into, through and out of the system.

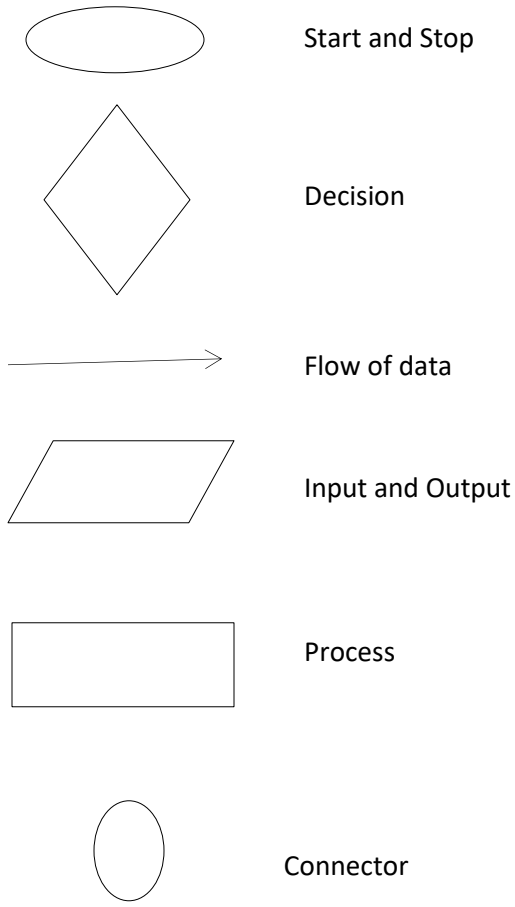
SYSTEM DESIGN TOOLS

- Flowcharts

It is a graphical representation of a system or algorithm.

It is a diagrammatic representation of the various steps involved in designing a system.

Notation/symbols used in flow charts



TYPES OF FLOW CHARTS

- System flow charts

It describes the data flow for a data processing system.

Provides logical diagram of how the system operates

Represents the flow of documents and operations performed in data processing system.

Example:

Algorithm

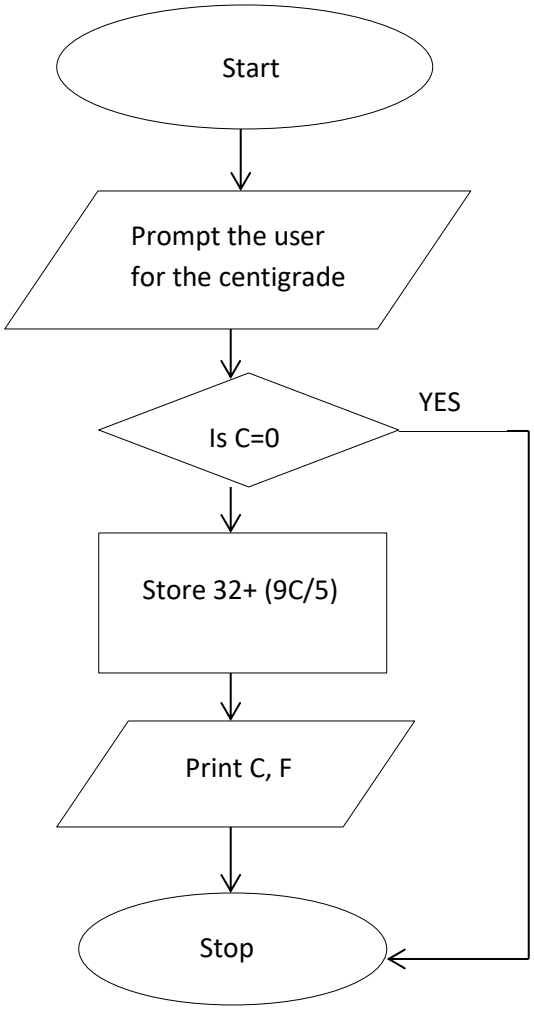
Prompt the user for the centigrade temperature

Store the value in the centigrade(C)

Set (F) to 32+ (9C/5)

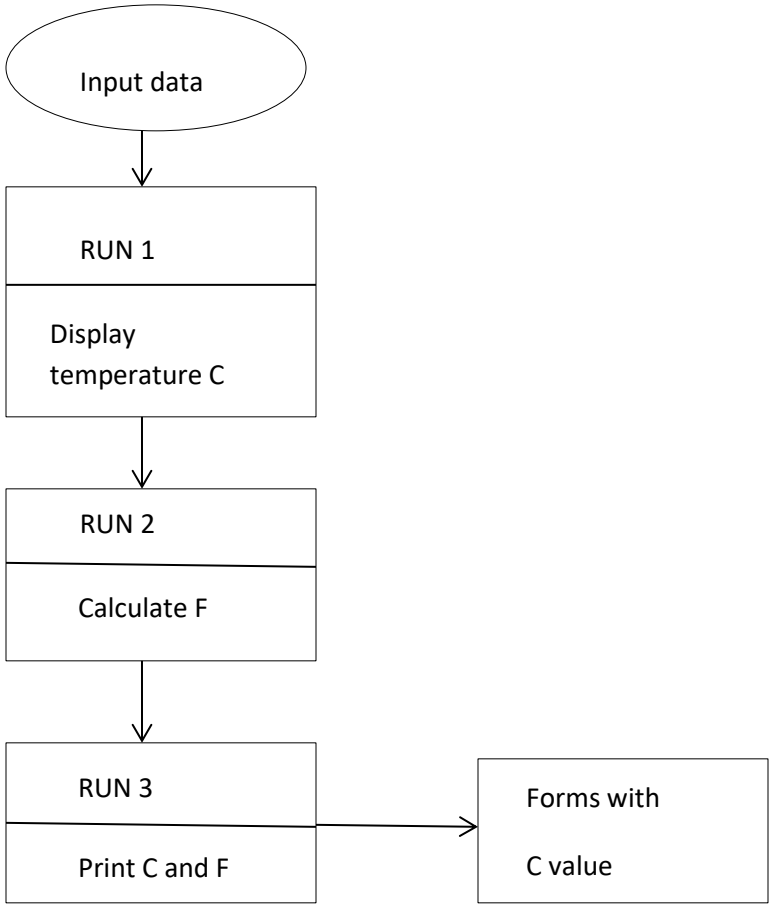
Print the value of C, F

Stop



- Run flow charts

Used to represent the logical relationship of computer routines along with inputs, master files, transaction files and outputs.



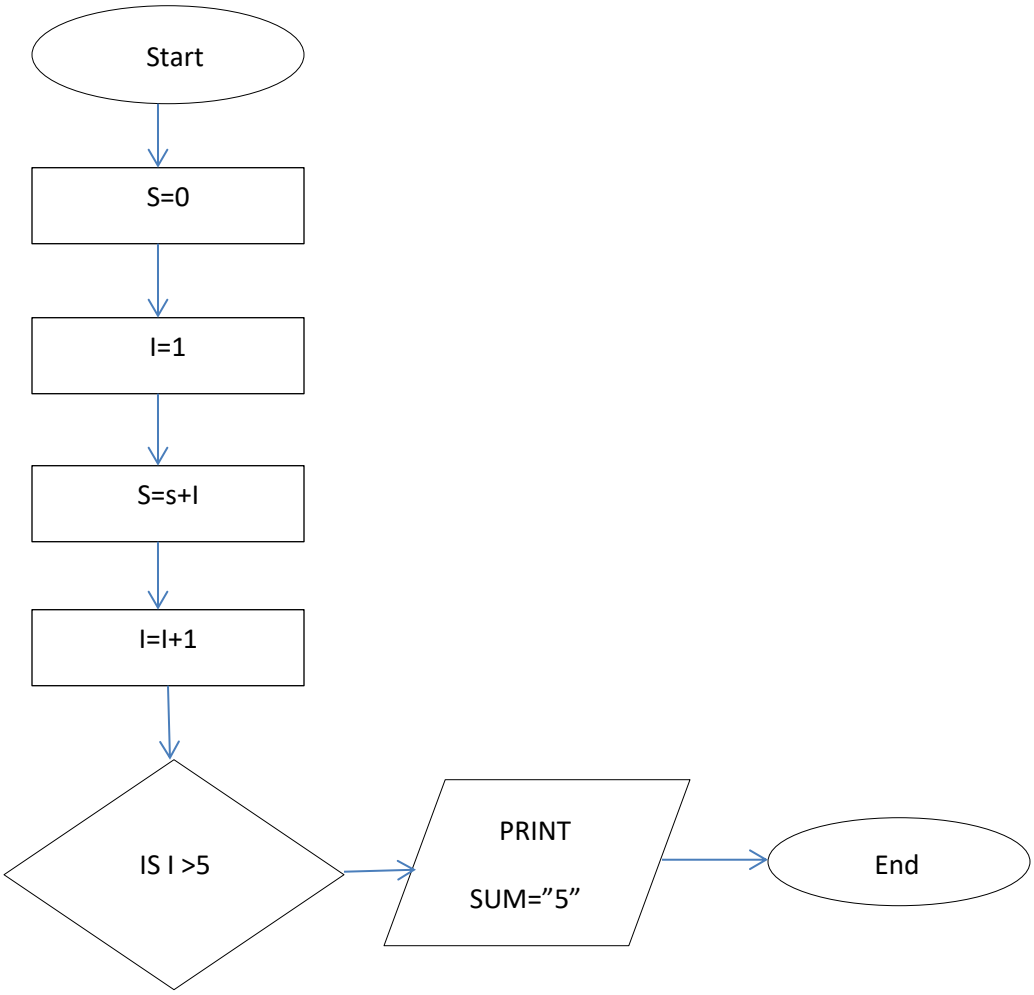
- Program flow charts

Represents in detail the various steps to be performed within the system for transforming the input into output.

The various steps are logical or arithmetic operations and algorithms.

Serves as the basis for discussion and communication between the system analyst and the programmers.

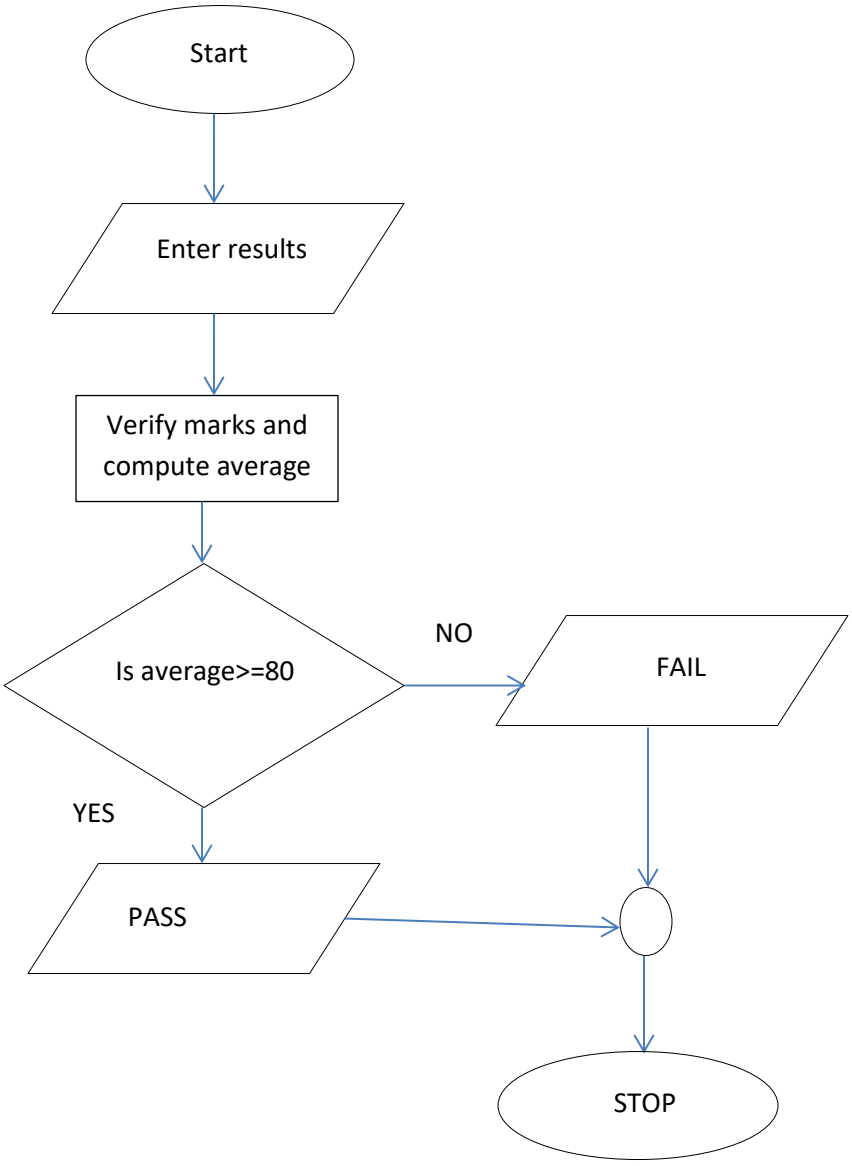
Program flow charts are helpful to programmers in organizing their programming efforts.



The registrar of wananchi technical institute intends to develop a system that will be used during the interview of new students.

During the interview, it is intended that the students undertake four written examinations. The proposed system would input the result. The system would then verify entry of marks before computing the average for each student. The system would then print “Pass” if the average is >80 else print “Fail”.

Draw a system flow chart to represent the logic of the program.



Advantages of using flow charts

- Communication: flow charts are better way of communicating the logic of a system to all stakeholders
- Effective analysis: problem can be analyzed in more effective way.
- Efficient coding: they act as a guide or blue print during the system analysis and program development phase.
- Proper documentation: program flow charts serve as a good program documentation which is needed for various purposes.

- Proper debugging: They help in debugging processes.

Limitations of using flow charts

- Complex logic: when the program logic is completed, the flow chart will become complex.
- Alterations and modifications: If alterations are required, the flow charts may require redrawing completely.

➢ CONTEXT DIAGRAMS

Also known as level 0 data flow diagrams (DFD)

Provides an overview of the data entering and leaving the system.

Also shows the entities that are providing or receiving that data.

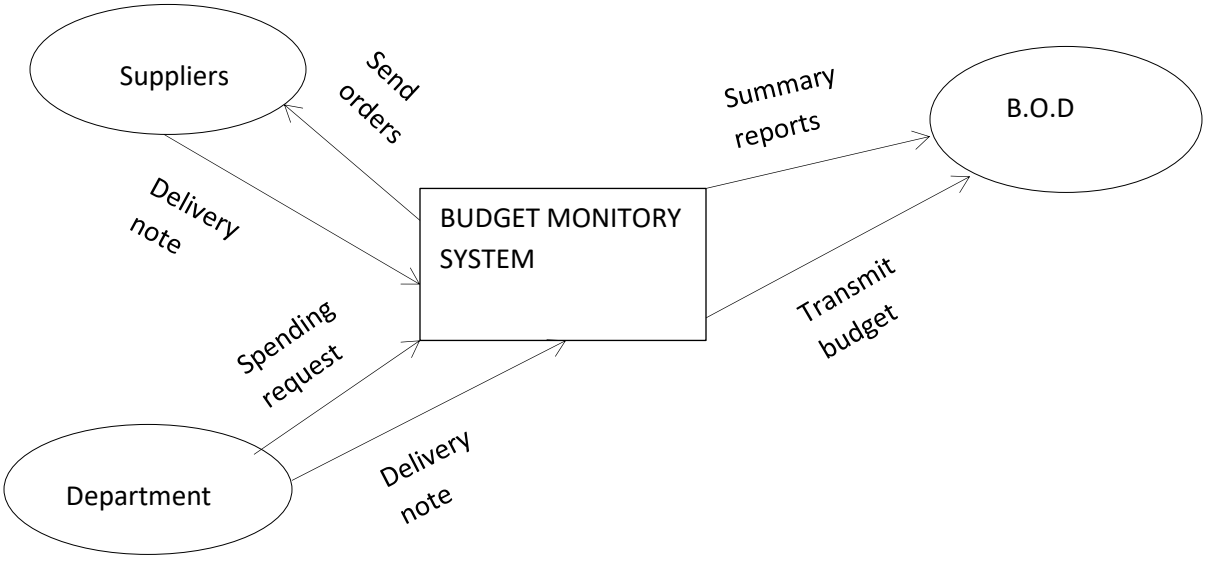
This usually corresponds to people who will use the system.

The context diagram helps to define system boundary to show what is included and what is excluded from the system.

Illustration of context diagram.

A certain company has a budget monitoring system (BMS) for its external entities namely department, Board of Directors (BOD) and suppliers. Each department channels its spending requests to the budgetary monitoring system, which ensures that the departmental budgetary allocation is not exceeded. The system then transmits the budgets to the BOD for approval. On approval, it is the function of the BMS to send orders to the suppliers. On receipt of the orders, the supplier channels the delivery note to the respective departments. In certain cases, the requests are rejected when they exceed the budgetary allocation. The BMS generates summary reports.

Draw a context diagram to represent this information.



- **DATA FLOW DIAGRAMS (DFD)**

DFD is a graphical representation of a systems data and how the processes transform data.

DFD is a graphical tool that allows system analysts and system users to depict the flow of data in an information system.

DFD is intended to serve as a communication tool among systems analyst, end users, DB designers, system programmers and other members of the project team.

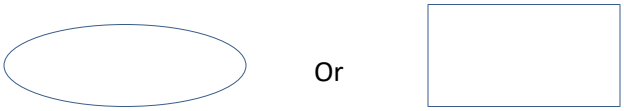
Components of DFD/Notations/Symbols

External entities

Represents the source of data as input to the system.

Are also the destination of system data.

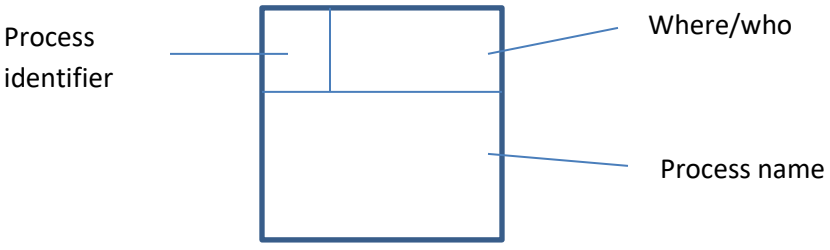
May represent job titles or other systems that interact with the system to be built.



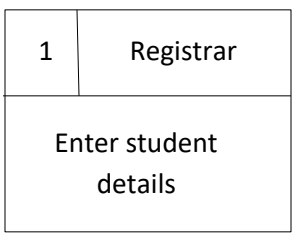
Process

These are actions carried out with the data that flows around the system.

A process accepts input data needed for the process to be carried out and produces data that it passes on to another part of the DFD.



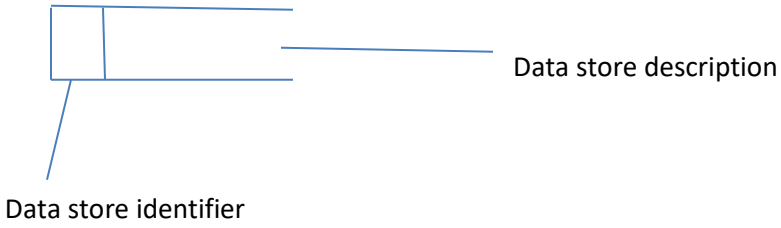
Example of a process



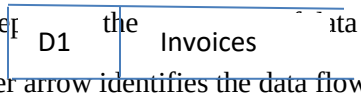
Data store

Data store are places where data may be stored.

This information may be stored either temporarily or permanently by the user.



Example of data store.



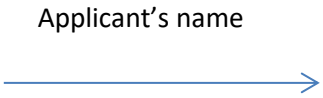
Data flows

They represent the movement of data from one point to another.

Another arrow identifies the data flow/data in motion

Data flows between external entities are shown as dotted line.

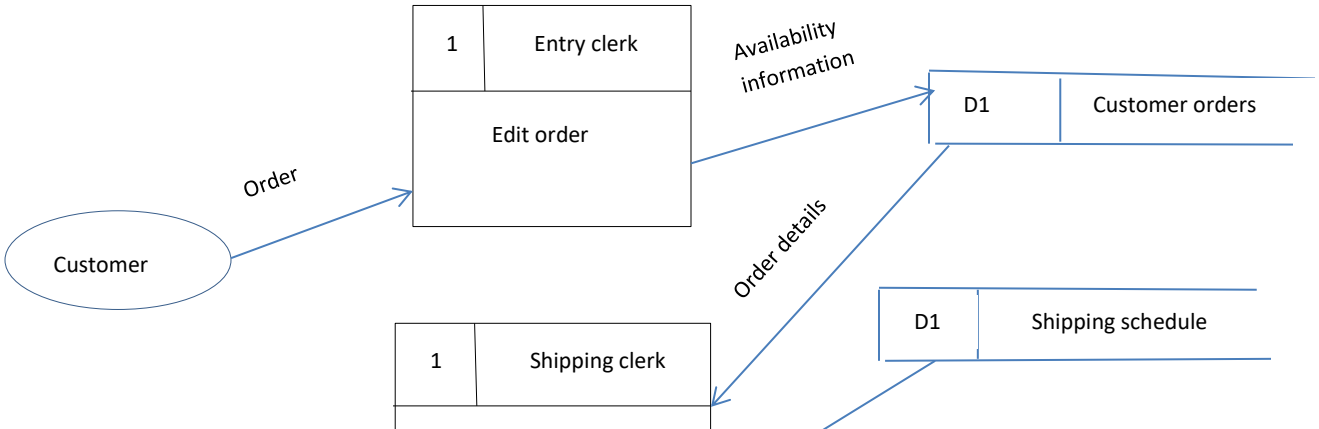
Example of data flow



Rules for drawing data flow diagrams

1. Any data flow leaving a process must be based on data input to the process
2. All data flows are named. The name reflects the data flowing between processes, data stores and sources.
3. Only data needed to perform the process should be an input to the process.
4. All processes in DFD must be linked to either another process or a data store.
5. Each data store must have at least one data flow going into and one data flow leaving it.
6. Data must be moved by a process from a one data store to another.

Example of a level1 DFD diagram.



Exercise

- An external entity “supplier” sending an invoice to a process “deals with payment”. Assume the process takes place in the accounts division.
- A process called “update membership” writing customer details to a data store called customers. Assume the process is performed by the membership secretary.
- A process called “order raw materials” is performed by the purchasing clerk. To do this, the clerk retrieves details from the supplier data store and sends the purchase order to the external entity supplier. The clerk also stores the purchase order details in a data store “purchase orders”.

➢ ENTITY LIFE HISTORY DIAGRAM

It is a map that attempts to map the possible life of each occurrence of an entity from creation to deletion.

ELH is part of the event model and shows the effects of time on an entity or the dynamic behavior of the system data.

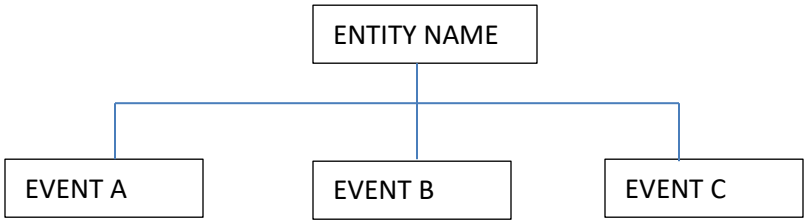
Entities have lives in that over time things happen to them which change their state.

Entity life history symbols

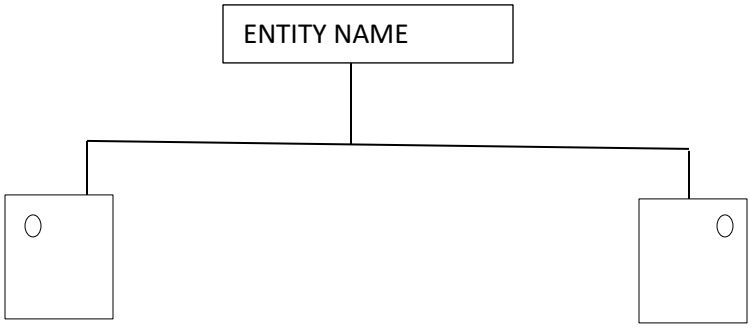
Sequence construct

Shows things that are expected to happen one after the other

The thing that is expected to happen first is positioned to the left and at the same height as the thing that is expected to follow it.

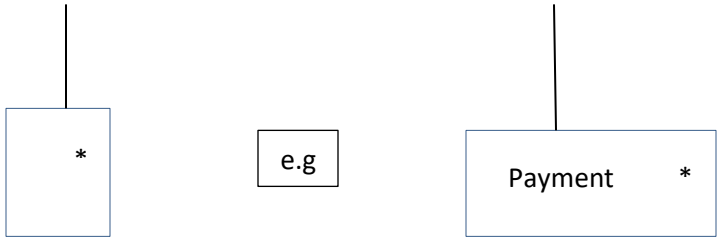


Selection construct



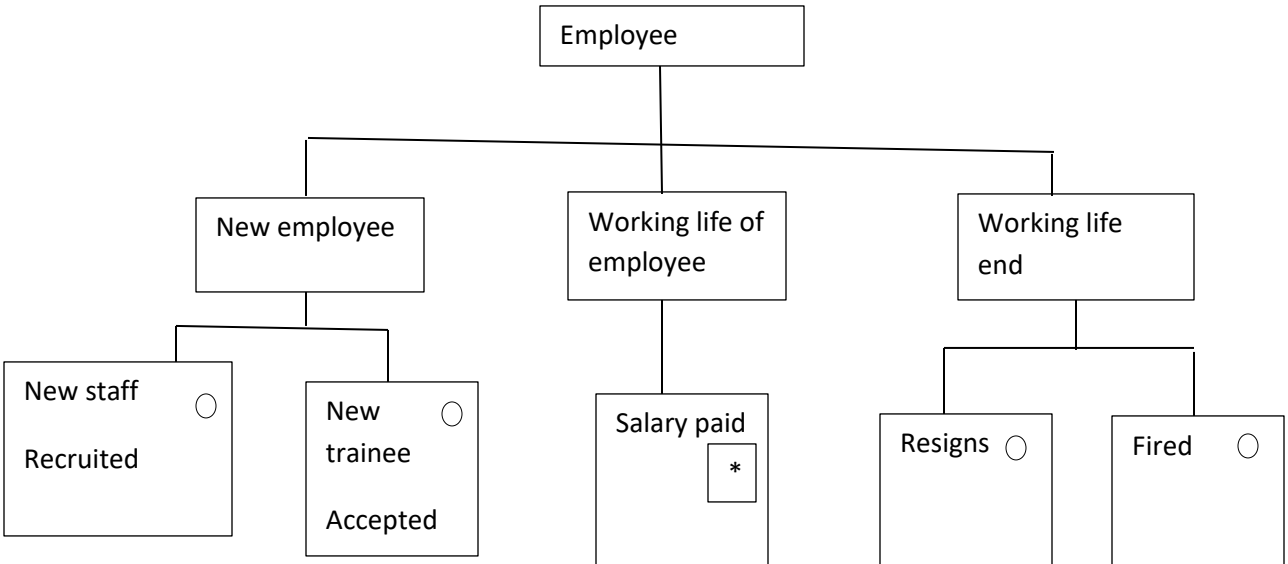
Iteration construct

These are entities that occur many times for each particular occurrence of an entity. It is represented using (*)



Parallel construct

Example of entity life history diagram



- **Data dictionary**

Describes the properties of the fields used in a database.

Data dictionary: It is a file containing descriptions of all data items in a database.

The data dictionary holds data about data and can be made to appear as a table.

The description might include: data type, element size, attribute entity relationships, entity descriptions and validation range of values.

Example of a data dictionary which stores employee information

Field name	description	Data type	form	Field size	key	validation
Pay roll number	A unique no given to an employee	number	-	double	primary	N
First name	First name of the person	text	-	30		N
Last name	Last name of the person	text	-	30		N
Date of birth		date	DD/MM/YY	8		Must be a date

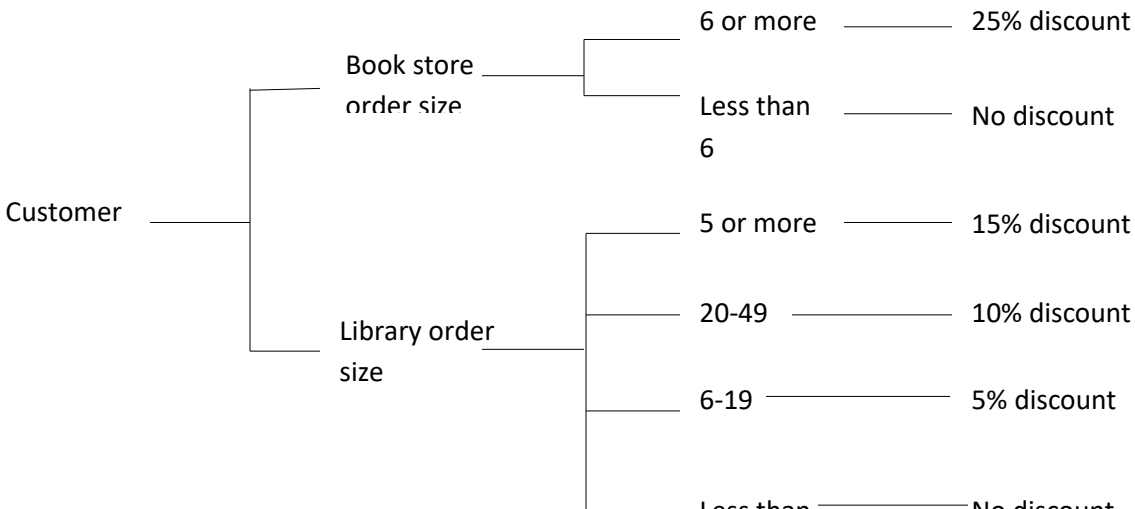
- **DECISION TREEE**

It defines the conditions as sequence of left to right test.

A decision tree helps to show the paths that are possible in a design following an action or decision by the user.

Illustration

1. If order is from book store and if order is for 6 copies, then discount is 25%.else if order is for less than 6 copies no discount is allowed. Else if order is from library and if order is for 50 copies or more, then discount is 15%.else if order is for 20-49 copies, then discount is 10%.else if order is for 6-15 copies, then discount is 5%.else if order is for less than 6 copies no discount is allowed.



Amua Sacco limited issues loans to its members using the following criteria.

A member would qualify for a loan if he/she has operational FOSA account for the last three years and he/she has never defaulted loan. In addition, a member should have 3 guarantors and provide proof of reliable monthly income.

Represent this using a decision tree.

- **STRUCTURED ENGLISH**

It is a narrative form of English written as a series of blocks that use indentation and capitalization to represent hierarchical structure of logical specifications.

It is based on structured logic.

It is used when process logic involves formulas or iteration or when structured decisions are not too complex.

A good structured English statement reads like a short sentence. By convention, only key words such as IF, THEN, REPEAT, DO, UNTILL, WHILE are capitalized. Data names and the general English needed to complete a sentence or a phrase are lower case.

Format/syntax:

IF (some condition)

THEN (what to do if true)

ELSE (what to do if false)

END IF

DO WHILE <condition>

(Do while condition is true)

END DO WHILE

e.g.

IF customer balance is>60 days past due

THEN hold the customer-order

Send reminder letter

ELSE process the customer-order

END IF

DO WHILE there are overtime pay records

Read a record

Add overtime hours to overtime hours total.

Add overtime pay to overtime pay total

END DO WHILE

Example

A car insurance company process claims as follows:

First it determines whether claims are valid. If a claim is invalid, the process terminates with regrets. Otherwise the claim will be processed depending on the insurance policy. This process will be repeated until all the claims for the day are complete.

Model this process using structured English

IF (Claim is valid)

THEN process claim according to the insurance policy

ELSE terminate policy

END IF

Alternatively

DO process claim depending on the insurance policy

WHILE claim is valid

END DO WHILE

Advantages of structured English

- Useful for planning/designing program routines, modules and manual procedures.

It resembles a programming language and hence programmers find it easy to understand.

- It is excellent for describing English and algorithm particularly when user communication is essential.

- **ERD(ENTITY RELATION DIAGRAM)**

An ERD is a data modeling technique that graphically illustrates an information system database entities and the relationship between these entities.

ERD's show entities in a database and relationships between tables within the database.

ERD's illustrate the logical structure of databases.

Elements in an ERD

Entity

It is an object in the system that we want to model and store information about. Can be person, place, event etc.

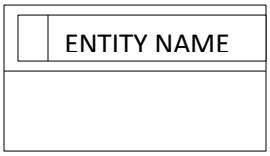
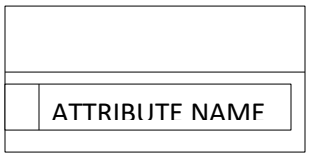
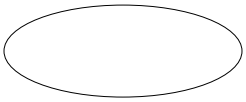
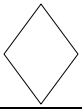
Attribute

It is an item of information which is stored about an entity e.g. student would have attributes such as: first name, last name, date of birth etc.

Relationship

It is an association of entities where the association includes one entity from each participating entity type.

Entity relation diagram notations/symbols

	SSADM	CHEN NOTATION
ENTITY		
ATTRIBUTE		
RELATIONSHIP		

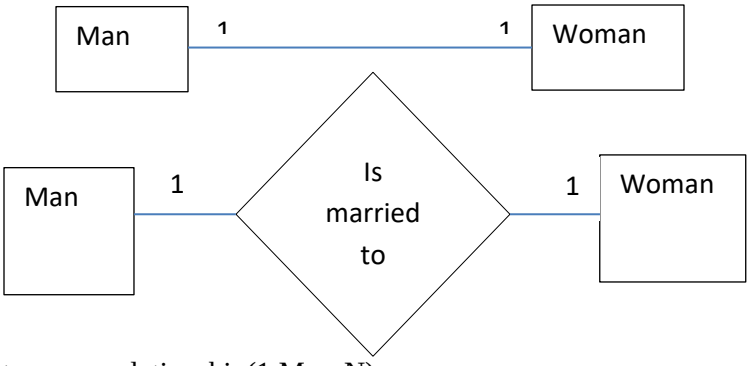
Degree/types of relationships/cardinality

- One-to-one relationship(1:1)

It is where one occurrence of an entity relates to only one occurrence of another entity.

Example:

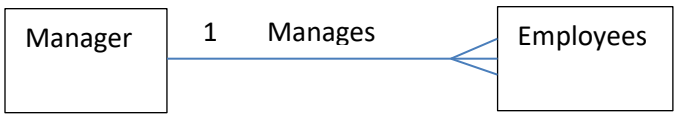
If a man marries one woman and a woman is married by only one man, it is a one to one relationship.



- One-to-many relationship(1:M or N)

It is where one occurrence in an entity relates to many occurrences in another entity.

Example: one manager manages many employees but each employee has one manager only.



- Many-to-many relationships(M:N,N:M)

It is where many occurrences in an entity relate to many occurrences in another entity.

Example:

One teacher teaches many students and a student is taught by many lecturers.

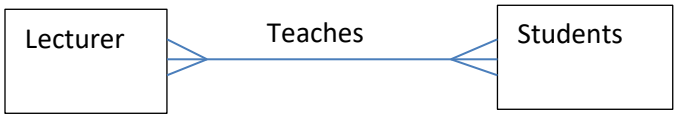
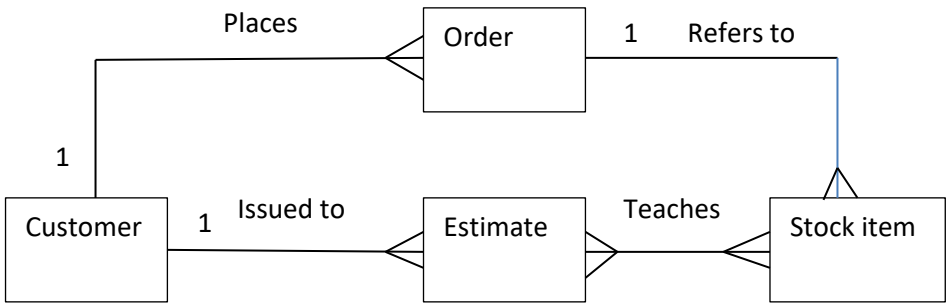
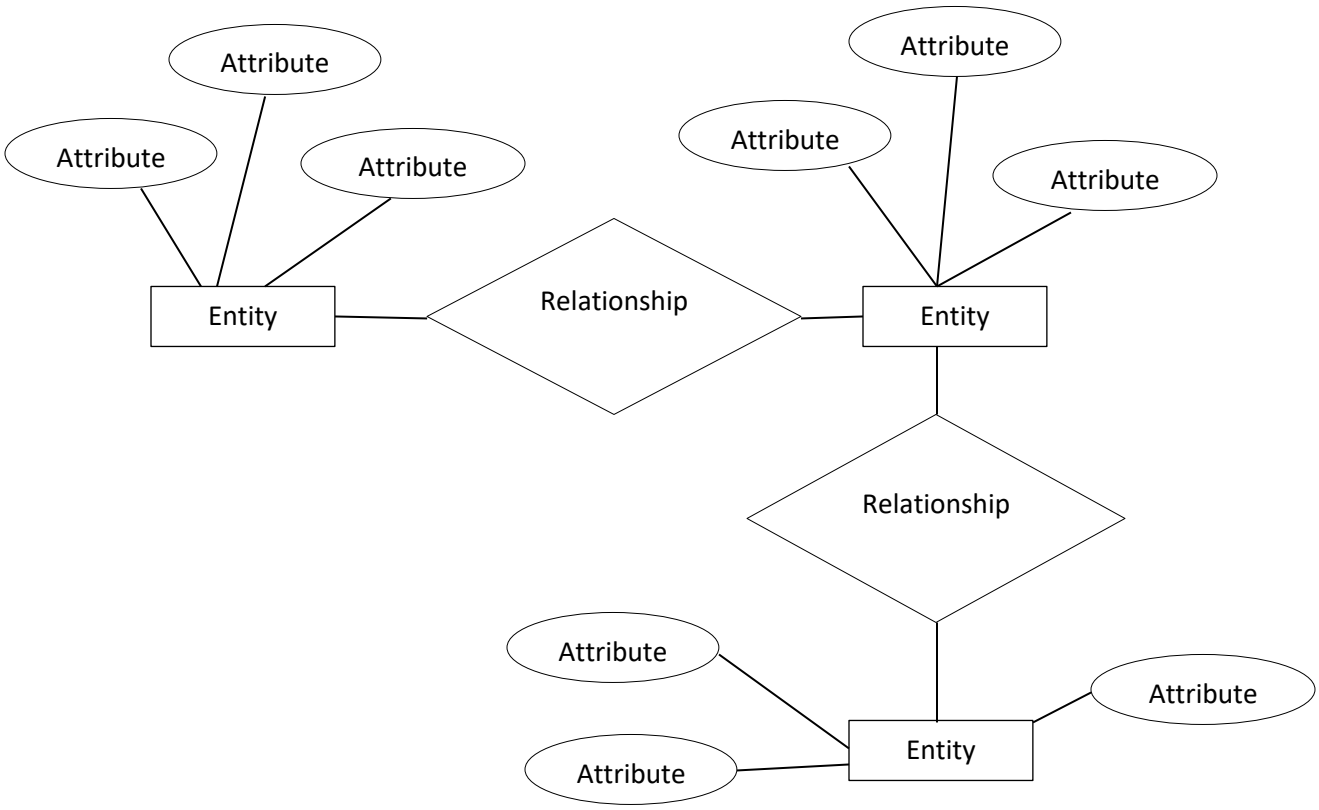


Illustration of ERD





- **STRUCTURE CHARTS**

It is a chart which shows the breakdown of a system to its lowest manageable levels.

Structured charts represent hierarchical structure of modules.

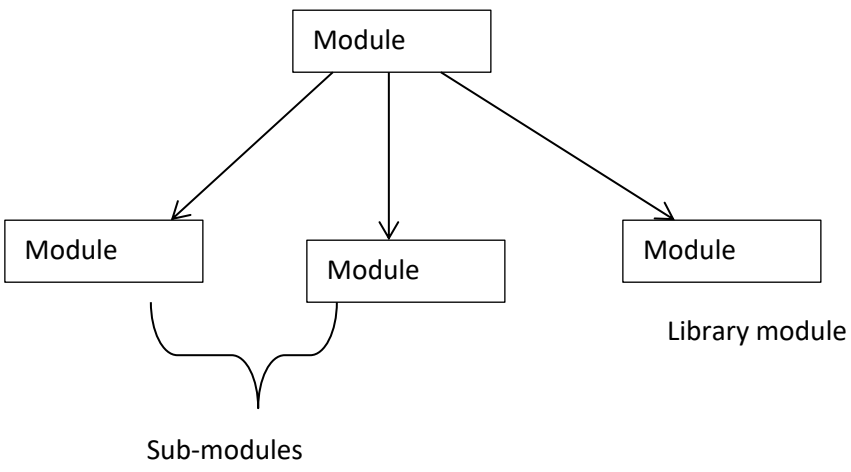
They describe functions and sub functions of each part of the system.

Modules are self-contained system components.

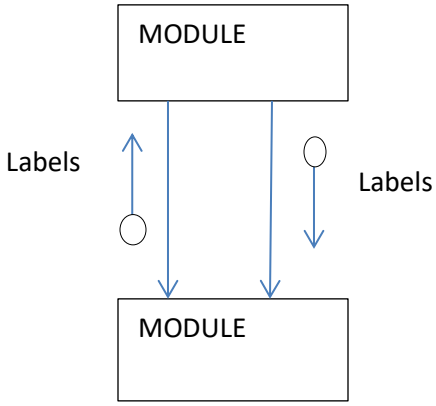
Symbols used in the construction of structure charts

a. Module

Represents process or subroutine or task.



b. Data flow

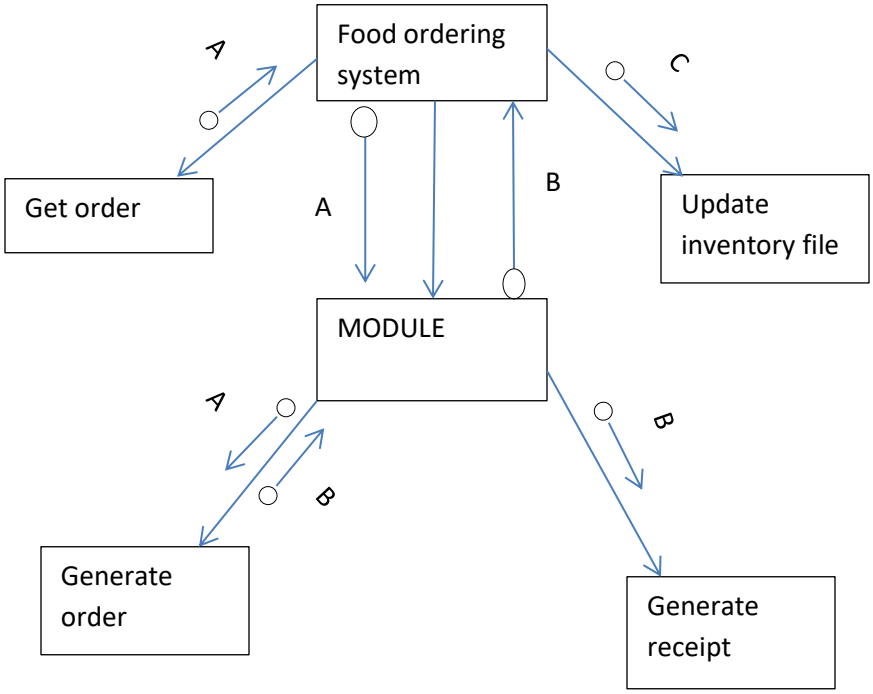


Example of structure charts

A: customer order

B: order details

C: inventory data



CHARACTERISTICS/QUALITIES OF A GOOD DESIGN

- Correctness: a good design should correctly implement all the functionalities identified in the system requirements specification document(SRS document)
- Minimal complexity/understandability: A good design is easily understandable. The design must be a readable and understandable guide for the developers who will write the code for the system based on it as well as for the developers who will later test and maintain it.
- Maintainability/ease of maintenance: it should be easily changed
- Re-usability: design of the system should allow part of it to be used in other systems.
- Extensibility/flexibility: should be easy to change part of a system without affecting other parts.
- Loose coupled: it means designing so that connections among different parts of a program are minimal. It should not be tightly coupled (components depend on each other so much)

The deliverable of system design phase is a blue print/model for the design with the necessary specification for the hardware, software, people and data resources.

- ✓ **SYSTEM CODING/CONSTRUCTION/PROGRAMMING**

The objective of system coding is to convert the system's specification into a functioning system.

The system design needs to be implemented to make it a workable system.

Programming languages commonly used are as follows:

Object oriented languages e.g. C++, VB, Java, C#(C Sharp)

Scripting languages e.g. Java Script, VBScript

Expert system languages e.g. PROLOG

Factors to consider when selecting a programming language

- Knowledge of software development staff: software developers should select a language which they are familiar with.
- Capabilities of software development staff: the staff within the organization (IS specialists) should be able to understand the language used in order to perform maintenance to the software.
- Application area: environment in which the software is to be executed.

- ✓ **SYSTEM TESTING**

Testing is the process used to assess the correctness, completeness, reliability and quality of developed computer software.

It helps to uncover different classes of errors in minimal time.

Levels of testing

- Unit testing

- Integration testing
- System testing
- Final acceptance testing

Unit testing

It is a software verification and validation method in which the programmer tests if individual units of source code are fit for use.

A unit is the smallest testable part of an application and it may be an individual module, function or procedures.

The five categories of test that a programmer performs on program unit are

- Parallel test: some data is used in the new and the old system and output results are then compared.
- Structural test: examining the internal processing logic of software.
- Performance test: to verify response time, execution and memory initialization
- Functional test: it is conducted to verify whether the system does what is expected.
- Stress test: performed to determine the stability of the system

Types of unit testing

- Static analysis testing: performs desk checks, structured walkthrough on source code inspection.

Desk check: programmer checks for logical and syntax errors and deviation from coding standards.

Structured walkthrough: the programmer leads other programmers through the source code of the software giving explanations.

- Dynamic analysis testing: consists of three types of testing
  - Black box testing: takes an external perspective of the test object to derive test cases. No knowledge of internal structure is required.it is applicable to all levels of software testing.(a testing that is concerned with the external working of a component)
  - White box testing: uses an internal perspective of the system to design test cases.
  - Grey box testing: combination of black box and white box testing.

Integration testing

It is a software testing activity in which individual software modules are combined and tested as a group.

System testing

It is a process in which software and other software management tools are tested as one system/unit.

The purpose is to ensure that the new system/modified function properly.

Final acceptance testing

It is conducted when the system is ready for implementation.

It ensures that the new system satisfies quality standards and user needs.

It has two major parts

Quality assurance: ensures the new system satisfies quality standards

User acceptance: users test the system to ensure that functional aspects expected by them have been well addressed in the new system.

Types of user acceptance testing

- Alpha testing: this often performed by users within the organization and it is the first stage.
- Beta testing: it is generally performed by external users.it involves sending the system outside the development environment for real world exposure/usage (performed by users outside the organization).

✓ SYSTEM IMPLEMENTATION

It is the process of ensuring that the information system is operational and then allowing users to take over its operation for use and evaluation.

Implementation includes all those activities that take place to convert from the old to the new system.

The new system may be totally new replacing an existing manual or automatic system or may be a major modification in an existing system.

The major steps/activities involved in this phase are

- Acquisition and installation of new information system.
- Conversion of data to the new system.
- User training
- Documentation(completion of user documentation)
- System changeover

System implementation techniques/changeover/conversion strategies

Conversion or changeover is the process of changing from old system to the new system

- Direct implementation changeover: it is the complete replacement of the old system by the new system.it is done in one operation completely replacing the old system at once.
- Phased implementation: the new system is put into use in stages or phases. Some files maybe converted and used by employees in the new system while other files continue to be used in the old system. If each phase is successful, the next phase is started eventually to the final phase where the new information system fully replaces the old system.(install some modules/phases with time until the whole system is completed)
- Pilot implementation: in this strategy, the new system replaces the old system in one operation but only in a small scale.

Example: it might be tried out in one branch in a company or in one department and if successful, the pilot is then extended until it eventually replaces the old system completely.

- Parallel implementation: the old and the new systems are both used alongside each other both being able to operate independently.

If all goes well, the old system is stopped and the new system carries on as the only system.

INFORMATION SYSTEM MAINTENANCE

Maintenance is the process of making needed changes to the structure of some information system.

System maintenance is the ongoing maintenance of a system after it has been placed into operation.

Types of information system maintenance

- ✓ **Corrective maintenance**

It implies removing errors in a program which might have crept into the system due to faulty design or wrong assumptions.

Thus, in corrective, it is the process where performance failures are repaired.

- ✓ **Adaptive maintenance**

Program functions are changed to enable the information system to satisfy the information needs of the user.

This type of maintenance may become necessary because of the organizational changes which may include change in the organizational procedures, change in forms, change in information needs of managers, change in system controls and security needs, change in organizational objectives and policies, change in operating system.

- ✓ **Perfective/enhancement maintenance**

Perfective maintenance means adding new features or modifying the existing programs to enhance the performance of the current system.

Perfective maintenance is undertaken to respond to users additional needs which may be due to changes within or outside the organization.

An example of this type of maintenance is the conversion of text based systems to graphical user interface design (GUI)

- ✓ **Preventive maintenance**

It deals with activities aimed at increasing system maintainability, such as updating documentation, adding comments and improving the modular structure of the system.

Reasons for information system maintenance

- ✓ Changes in business processes

Systems should be modified or updated to enable them address emerging or new business processes.

- ✓ New requests from stakeholders ,users and managers
- ✓ Bugs or errors in the system
  Maintenance is necessary to fix errors.
- ✓ Change in operating system or hardware on which the system runs.
- ✓ Corporate mergers and acquisitions
- ✓ Government policies

CHAPTER 7

SYSTEM DOCUMENTATION

It is a document that gives a comprehensive description of a system.

It shows how the software has been developed and the features or functionalities.

There are several types of documentation that should be produced when creating a new system.

Types of documentation

- ✓ Technical documentation

It is a document of source code, algorithms and interfaces. It is intended to assist skilled programmers/technical people know exactly how the system works.it includes:

- Details of Hardware and software for the system
- Details of expected inputs
- Details of validation checks
- Details of how data is processed
- Diagrams showing how data moves through the system
- Flow charts describing how the system works.

- ✓ User documentation

It is a complete description of the system from user's perspective.it gives in detail how to use the system or operate the system.

These are manuals prepared for the end user, system administrator and support staff.it is intended to help users of the system. Users are usually non-technical people who don't need to know how the software works. They just need to know how to use it.

User documentation includes:

- List of minimum hardware and software required to use the system.
- How to install the system.
- How to start/stop the system.
- Screen shots showing the system in typical use.

- Explanation of any error messages that might be shown.
- Example inputs and outputs.
- Troubleshooting guide.

✓ Marketing documentation

It describes how to market the product and analysis of the market demand.

INFORMATION SYSTEMS ACQUISITION

Factors affecting the choice of information system acquisition method

✓ Cost of acquisition

Small organizations can prefer to purchase commercial off-the-shelf software rather than developing in-house programs.

✓ Capability of in-house ICT team

The number of ICT personnel and the level of their knowledge and skills can determine if the organization has enough manpower or expertise to develop the system.

✓ System complexity

If in-house ICT team is not able to manage a complex system, the organization can opt to outsource ICT services.

✓ Size of the organization
Small organizations may not be able to develop in-house software and therefore can adopt other methods like purchasing ready-made software or using open source software.

Information system acquisition methods

✓ Commercial off-the-shelf purchase
✓ System development/bespoke development/in-house development
✓ Outsourcing
✓ Open source software
✓ Renting
✓ Leasing

- Commercial off-the-shelf purchase

This is an acquisition method that involves direct purchase of a pre-written application or system used by more than one company.

Advantages

-readily available for purchase and use

-cheap

Disadvantages

-the system may lack all the requirements needed.

- System development

This is where an information system is developed from scratch by information system professionals to suit the business requirements of the organization.

Advantages

-ownership: The organization owns the system completely

-the system has the required features

Disadvantages

-expensive: As it requires both resources and time to develop.

- Outsourcing

It is the practice of subcontracting part or all of an organization's information system functions to an external service provider

Advantages

-cost reduction: Focus/concentrate on their core competencies

-knowledge: a way to gain access to new technology and outside expertise.

- Open source software

Software that has no copyright over the code and allows the public to modify the source code and develop it to their own content.

Software that is developed, tested or improved through public collaboration and distributed with the idea that it must be shared with others ensuring an open future collaboration.

- Renting

An acquisition method where an organization that requires the hardware, software or computer system gets them from another company after signing a rental contract.

The computer system or hardware system can only be used for the activities or functions that have been specified in the contract.

- Leasing

An information system is acquired from another company after signing a lease contract. The lease contract is longer than that of renting.

CHAPTER 8

INFORMATION SYSTEM PROJECT MANAGEMENT

Information system project management is the process of planning, monitoring, controlling people, processes and events that occur as software evolves from a preliminary concept to an operational implementation.

Project

A set of related tasks that is coordinated to achieve a specific objective within a given time limit and under a specified budget.

Deliverable

It is the end product of a software development life cycle phase.

It can be a report or a working system depending on the software development phase.

Importance of information system project management

- ✓ Meeting customer expectation

Project management techniques will enable developers to deliver a system that satisfies user requirements.

- ✓ Satisfying budget constraints

Effective project management will ensure that the system is delivered within budget.

- ✓ Satisfying time constraints

Project management will ensure that the system is delivered within scheduled time.

- ✓ Equal distribution of tasks and responsibilities to members of the development team.

Project identification techniques

- Top-down identification: focuses on global needs of the organization and is usually done by the top management or steering committees.
- Bottom-up: project is identified by a business unit or information system group.

Information system project management techniques

Cost estimation techniques

- ✓ Expert judgment

Several experts on software development techniques and the application domain are consulted. They each estimate the project cost.

These estimates are compared and discussed.

The estimation process iterates/repeats until an agreed estimate is reached.

- ✓ Estimation by analogy

This technique is applicable when other projects in the same application domain have been completed. The cost of a new project is estimated by analogy with these completed projects.

- ✓ Pricing to win

The software cost is estimated to be whatever the customer has available to commit to the project.

- ✓ Constructive cost model(COCOMO)

It is an approximation of effort needed based on experience of past projects

Project scheduling

It is the process of estimating the duration of activities in a project and presenting the estimation using tools that are universally accepted.

The two graphical tools that are used in project scheduling are:

- ✓ Gantt chart
- ✓ PERT chart/Network diagram

Gantt chart

It is a graphical representation of a project that shows each task/activity as horizontal bars whose length is proportional to its time of completion.

Different colors or shades can be used to highlight different activities.

Example:

Time (weeks)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
feasibility	■	■																		
analysis			■	■	■	■	■	■	■											
design										■	■	■	■	■	■					
coding																■	■	■		
testing																			■	■

Pert chart/network diagram PERT (project evaluation and review technique)

On the PERT chart, a project is viewed as a network of activities of which some must be completed before others catermn begin.

PERT assumptions

- Inter-relations of activities are depicted/shown on a network on directed arrows which denote sequence of activities.
- The nodes called events represent instance in time when certain activities have been completed and others can then be started.
- The origin node is the beginning of the project.

Types of network diagrams

- ✓ Activity on arrow(AOA)
- ✓ Activity on node(AON)

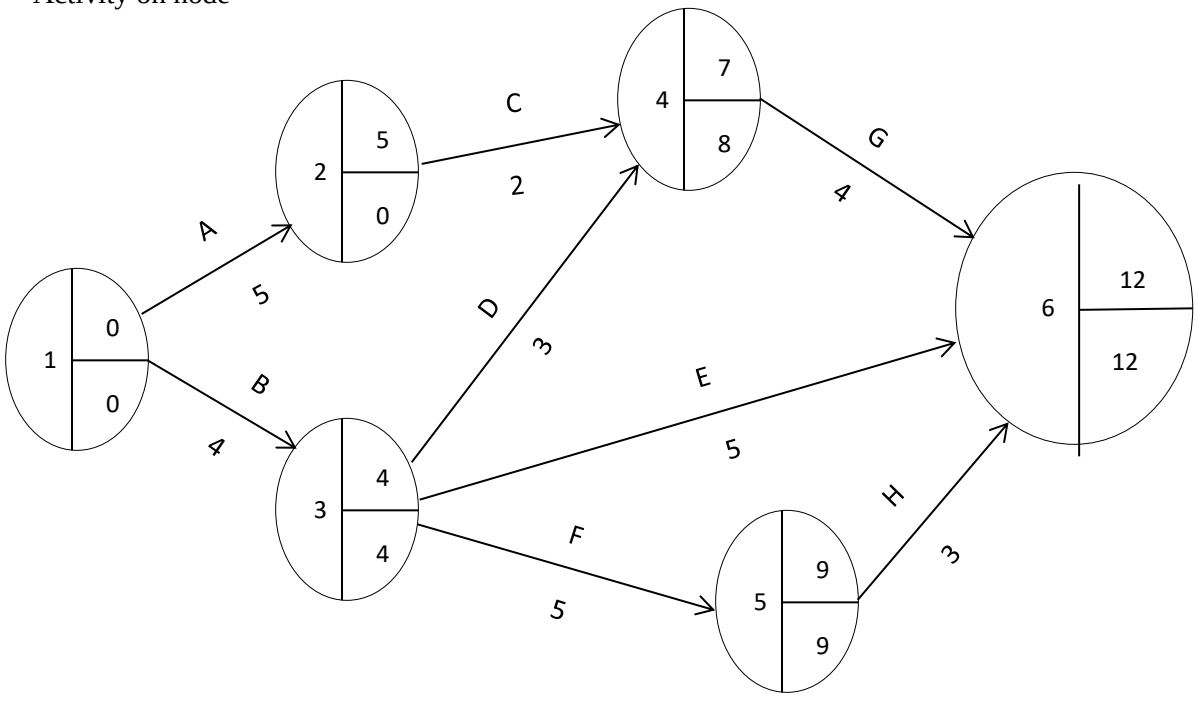
Activity on arrow

- Activities are shown on the arrow
- It is easier to draw and modify
- Non-experts are more likely to understand the network diagram
- Milestone events are readily visible.

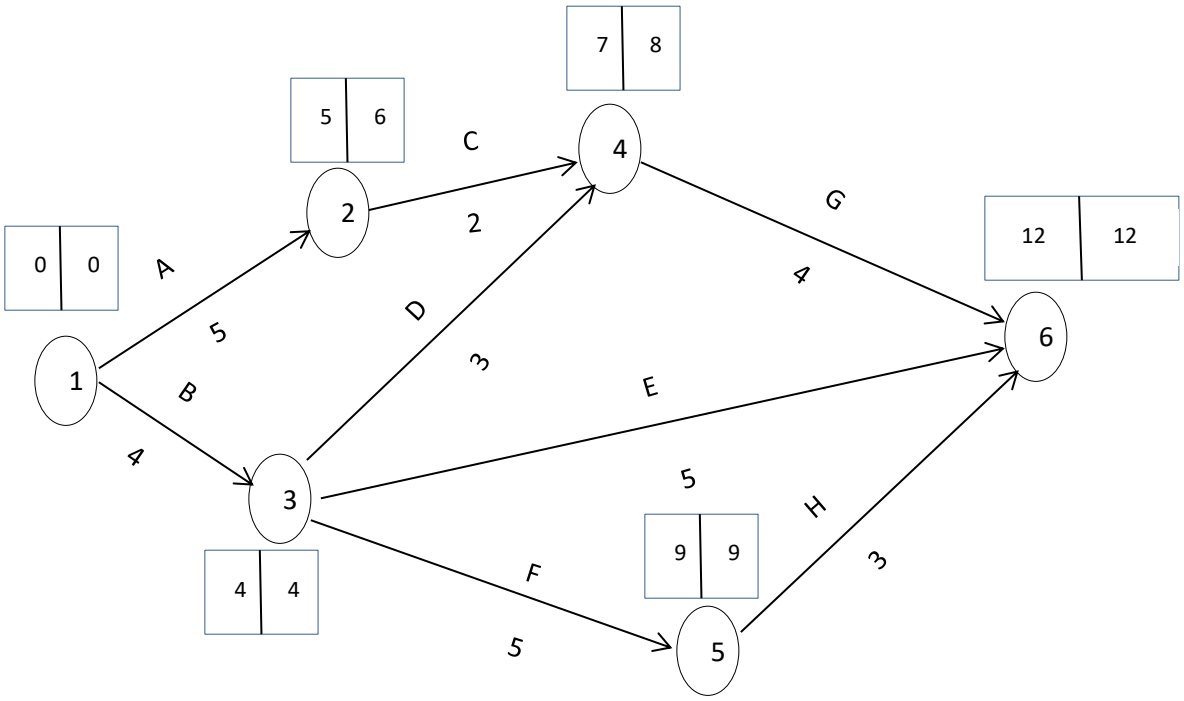
Illustration

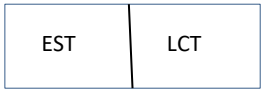
ACTIVITY	PRECEEDING ACTIVITY	DURATION(WEEKS)
A	-	5
B	-	4
C	A	2
D	B	3
E	B	5
F	B	5
G	C,D	4
H	F	3

Activity on node



Activity on arrow





EST: Earliest Start Time

LCT: Latest Completion Time

Critical path: B-F-H

Earliest Start Time

EST at an event is the earliest time activities ahead of that event can start, keeping in mind that all the activities before the event must be complete. It is calculated in the forward pass.

Activity durations on each path linking to an event are added and then the largest is taken.

The first event has EST value 0

The EST in the last event gives the project duration.

In the above example, the project duration is 12 weeks. Latest Completion Time

LCT at an event is the latest time that preceding activities can complete without delaying any of the succeeding activities.

It is calculated in backward pass, starting from the last event whose LCT is set to the project duration.

Critical path

It is the sequence of activities that takes the longest time to complete.

It is the sequence of activities that have the same EST and LCT values.

Any delay to an activity in the critical path will cause delay to overall project.

Slack time

It is free time associated with each activity as it represents unused resources that can be averted to the critical path.

Dummy activity

It is a hypothetical activity which requires zero time and zero resources for completion. A dummy activity has a completion time of zero.

Dummy arrow represents an activity with zero duration.

It is represented by a dotted line.

Estimation of activity times

Optimistic time

It is the estimate of the maximum time an activity will take

The most optimistic (O) case where everything goes right

Most likely time

The completion time having the highest probability.

The most likely (M) case given normal problems and opportunities

Pessimistic time

An estimate of the longest time that an activity might require.

The most likely (M) case given normal problems and opportunities

The resulting PERT estimate is calculated as (O + 4M + P)/6. This is called a "weighted average"

Signs of a failing information system project

- ✓ Poor communication: It is where no one understands what to do and there is no communication as to current progress.
- ✓ Poor planning and estimation: projects that are poorly estimated and planned tend to fail both in cost and schedule which eventually causes the overall project to fail.

- ✓ Poor documentation/minimal documentation: many failed projects reveal that there was too little documentation to adequately describe the project in its broader terms and serve as a clear communication channel.
- ✓ Poor user requirements: when the user requirements have not been adequately captured it may lead to misalignment between the project and business objectives.
- ✓ Budget overrun: projects that run over budget are likely to be cancelled.
- ✓ Poor project control: the project manager may not have the skills or experience required to manage the project.
- ✓ Time overrun: developers may run out of time that they had scheduled.

Causes of information system project failure

- ✓ Lack of senior management support and involvement in information system development.
- ✓ Lack of user participation

User involvement is necessary to reduce resistance to change and ensure adequate development.

- ✓ Shifting user needs

User requirements for ICT change constantly. Changes during an ongoing development process cause a challenge and may cause the project to fail.

- ✓ Poor estimation techniques

When project cost and time are not well estimated, developers may run out of funds and time.

- ✓ Inadequate testing and user training

New systems must be tested before installation

Users must be adequately trained on how to use the system.

- ✓ Undertrained development staff

Developers may lack the required skills and knowledge/expertise required.

- ✓ Lack of standard project and system development methodologies.
- ✓ Resistance to change

Users have a natural tendency to resist change.

Control measures and techniques of rescuing a failing information system project

- ✓ Pausing the project

Pausing the project creates an opportunity to restore integrity to the project.

- ✓ Auditing the project

The purpose of project audit is not to place blame but rather is to find out the root cause why the project is failing.

- ✓ Recognizing early warnings

It is always easier to get projects back on track if they have not drifted too far off the track.

- ✓ Assessing the effort to complete the project

The human effort required to complete the project should be reviewed or assessed.

Measures of project success

- ✓ The resulting information system is acceptable to the client or users
- ✓ The system was developed within the time scheduled.
- ✓ The system was delivered within budget.

Criteria for evaluating ICT projects/project appraisal methods

- Net present value(NPV)
- Return on investment(ROI)
- Payback time(pay back analysis)

Net present value

It is a method of calculating the expected monitory gain or loss from a project by discounting all future cash inflows and outflows to the present.

$$NPV=\sum t=1\ldots\ldots\ldots\ldots n^{A}\ (Hr)^{-t}$$

OR

$$NPV=\sum t=1\ldots\ldots\ldots\ldots n$$

Where, A= cash flow $\quad \frac{A}{(1+r)^{t}}$

r=required rate of return

t=year of cash flow

n=n^{th} year

It is normally assum diately and is indicate by year 0 and it is discounted.

Later cash flows are normally assumed to take place at the end of each year and are discounted by the appropriate amount.

NPV acceptance rules

- Accept the project if its NPV is positive
- Reject the project if its NPV value is negative
- A project with 0 NPV may be accepted or rejected
- NPV can be used to select between mutual exclusive projects. The one with the higher NPV should be accepted.

Example

Project A requires an initial investment of Ksh 363000 and it is expected to generate a cash inflow of ksh 50000 per month for 12 months. Assume that the salvage value of the project is 0 and the target rate of return is 12% p.a.Calculate NPV for the project.

$$\text{YEAR 0} = \frac{-363\,000}{(1+0.12)^0} = -363\,000$$

$$\text{YEAR 1} = \frac{600\,000}{(1+0.12)^1} = 535{,}714.28$$

NPV=-363,000+535,714.28=172,714.28.

The table below shows cash flow details of project A and B

YEAR	PROJECT A	PROJECT B
1	-5000	-100000
2	3500	65000
3	3500	65000

Using NPV at a discount rate of 15%, determine the most viable project to undertake assuming the starting capital was ksh 500.

Solution

PROJECT A

$$\text{YEAR 0} = \frac{-500}{(1+0.15)^0} = -500$$

$$\text{YEAR 1} = \frac{-5000}{(1+0.15)^1} = -4347.83$$

$$\text{YEAR 2} = \frac{3500}{(1+0.15)^2} = 2646.50$$

$$\text{YEAR 3} = \frac{3500}{(1+0.15)^3} = 2301.31$$

NPV Total=99.98

PROJECT B

$$\text{YEAR 0} = \frac{-500}{(1+0.15)^0} = -500$$

$$\text{YEAR 1} = \frac{-100000}{(1+0.15)^1} = -86056.52$$

$$\text{YEAR 2} = \frac{65000}{(1+0.15)^2} = 49148.34$$

$$\text{YEAR 3} = \frac{65000}{(1+0.15)^3} = 42738.56$$

NPV Total=4431.38

Return on investment (ROI)

It compares the lifetime profitability of alternative solution/project.

This is a percentage rate that measures the relationship between the amount invested and the amount the business gets back from investment.

$$ROI = \frac{\text{Life time benefits-life time costs}}{\text{Life time cost}} \times 100\%$$

OR

$$ROI = \frac{\text{Net profit}}{\text{Cost of investment}} \times 100\%$$

$$ROI = \frac{\text{Average annual profit}}{\text{Total investment}} \times 100\%$$

The solution offering highest ROI is the best alternative.

Example

The table below shows net profit for project A and B.The initial cost for the projects were Ksh 180,000 and 220,000 respectively.

Determine the cost worthwhile project using the ROI technique.

YEAR	PROJECT A NET PROFIT VALUE	PROJECT B NET PROFIT VALUE
1	30000	40000
2	35000	46000
3	40000	46000
4	45000	46000
5	50000	46000

TOTAL	200000	224000

Project A

$$ROI = \frac{\frac{200000}{5}}{180000} \times 100\%$$

$$= 22.22\%$$

Project B

$$ROI = \frac{\frac{224000}{5}}{220000} \times 100\%$$

$$= 20.36\%$$

Payback analysis (time/period)

It is Project A is the best alternative as it has a higher ROI . This perio be selected.

The table below shows net profit for project A and B.The initial cost for the projects were Ksh 180,000 and 220,000 respectively.

Calculate the payback time for each project.

YEAR	PROJECT A NET PROFIT VALUE	PROJECT B NET PROFIT VALUE
1	30000	40000
2	35000	46000
3	40000	46000
4	45000	46000
5	50000	46000

PROJECT A	PROJECT B
YEAR 1=-180,000+30,000=-150,000	YEAR 1=-220,000+40,000=-180,000
YEAR 2=-150,000+35,000=-115,000	YEAR 2=-180,000+46,000=-134,000
YEAR 3=-115,000+40,000=-75,000	YEAR 3=-134,000+46,000=-88,000
YEAR 4=-75,000+45,000=-30,000	YEAR 4=-88,000+46,000=-42,000
YEAR 5=-30,000+50,000=20,000	YEAR 5=-42,000+46,000=4,000
PROJECT A payback time=$4\frac{3}{5}$ years	PROJECT B payback time=$4\frac{21}{23}$ years

Project A should be chosen as it has shorter payback time

COPUTER AII

CASE: Refers to the use of software tools to assist in the development and maintenance of software.

CASE tool: It is a computer based product aimed at supporting one or more software engineering activities within a software development process.

CASE tools: Are those software which are used in one or all phases of developing an Information System including analysis, design and programming.

Components of CASE tools

- Upper CASE tools: tools that support software development activities from implementation
- Lower CASE tools: tools that support direct programming and integration tasks. Supports database schema generation, program generation, implementation testing and configuration management.
- Integration CASE tools: tools that integrate both upper and lower CASE tools.

Example: making it possible to design a form and build the database to support it at the same time.

It is an automated system development that provides numerous tools to create diagrams forms and reports.

It offers analysis reporting and code generation facilities.

Types of CASE tools

- Diagramming tools: enables system processes, data and control structures to be represented graphically.
- Report generators: helps to prototype how system looks like.
- Analysis tools: checks for inconsistency and incorrect specification in diagrams, forms and reports.
- Documentation generators: helps to produce technical and user documentation in standard format.
- Code generators: enable automatic generation of program and database definition code directly from design documents program forms and reports.

FUNCTIONS OF CASE TOOLS

- Analysis: CASE analysis tools automatically checks for incomplete, inconsistency or incorrect specification in diagrams and reports.
- Design: tools that enable the design of a blue print/model by designing the technical architecture.
- Code generation: tools that enable automatic generation of program code directly from the document forms, diagrams and reports.
- Documentation: tools that enable production of technical and user documentation in standard format.

Advantages of CASE tools

- Increased speed: provide automation and reduce the time to complete many tasks.
- Increased accuracy: enable error checking and debugging services.
- Better documentation: by using CASE tools, it produces vast amount of documentation.

Limitations of documentation

- Cost: the cost of fitting every system developer with appropriate CASE tool kit can be costly.
- Training: A lot of time is needed to train a person who has no knowledge of CASE tool concept.

CHAPTER 9

EMERGING TRENDS IN SYSTEM ANALYSIS AND DESIGN

- Connectivity and collaboration: software development teams do not occupy the same physical space when working on a project.
- Re-usable components: components can be used in other systems other than the original systems.
- Use of CASE tool: CASE tools have reduced system development time considerably
- OOAD methodology: development methodology that approaches Information system development from object point of view.

Supplementary notes.
Decision Tables

A matrix representation of the logic of a decision which specifies the possible conditions for the decision and the resulting actions.
It has three parts

- Condition stub - list the conditions relevant the decision

- Action stub – That part of the decision table that list the actions that result for a given set of conditions.

- Rules – Specify which actions are to be followed for a given set of conditions.

- Indifferent conditions - a condition whose value does not affect which actions taken for two or more roles.

Example
The Decision Table showing a generic payroll system

Conditions AND Course of action		Rules					
		1	2	3	4	5	6
Condition Stub	Employee type	Salaried	Hourly	Salaried	Hourly	Salaried	Hourly
	Hours worked	<40Hrs	<40Hrs	40Hrs	40Hrs	>40Hrs	>40Hrs
Action Stub	Pay Basic Salary	✓		✓		✓	
	Calculate Hourly		✓		✓		✓
	Calculate Overtime						✓
	Absence Report	✓	✓				

Basic procedures in constructing the decision table

1. Name the conditions and the values that each condition can assume, determine all the conditions that are relevant to your problem. The conditions may be of limited entry (Yes/No) or extended answers.

2. Name all possible actions that can occur
3. List all possible rules

4. Define the actions for each rule
5. Simplify the decision table.

Compiled by Samuel Theuri