UNIT - I

WHAT IS SOFTWARE ENGINEERING:=>

Software engineering discusses systematic and cost effective technique to software development.

We can view software engineering as the engineering approach to develop software.

WHY STUDY SOFTWARE ENGINEERING:=>

1) By using software engineering technique to develop systematic software.
2)To develop cost effective software.
3)To learn how to apply the principles of decomposition to handle complexity during software specification, design ,development and testing.
4) To manage the software complexity.

WHAT IS SOFTWARE

Software is collection of programs, functions, procedure and documentation that perform some tasks o an operating system.

OR

Software is collection of programs.

OR

TYPES OF SOFTWARE :

Following categories of computer software present continuing challenges for software engineers.

1) System software.
System software is collection of system programs written to service other programs.
Example of System software is as-
Compiler, editors, operating system, linker, loader etc.

2) Application software.
Application software consists of program that solves the business problem.

3) Scientific software or Engineering software:
Used to solve mathematical problem, statistical problem and engineering problem.

4) Embedded software:
Embedded software Resides within a product or system and is use to implement and control features and function for the end user and for the system itself.

- Digital function in a automobile such as fuel control.
Ex: Remote system, radar system.

5) Web application:

→Web application is a set of link hyper text files that present information using text and limited graphics.
→Web application used to getting the information from any where in the world by using internet.

Ex: ecommerce, on line banking system.

6) Artificial intelligence software:
→ Artificial intelligence software makes use of non-numerical algorithm to solve complex problem.

Ex: Expert system , pattern recognition.

7) Product line software:
This type of software used to solve business problem, provided used utility software for creating program and provide software for entertainment world.
Processing ,spreadsheet program, computer graphics, multi media entertainment ,database management, personal and business financial application.

QUE : EXPLAIN THE EVOLVING ROLE OF SOFTWARE

Today, software takes on a dual role. It is both a product and vehicle for delivering a product. As a product, it delivers the computing potential embodied by computer hardware or, more broadly, by a network of computers that are accessible by local hardware. Whether software resides within a cellular phone or operates inside a mainframe computer, it is an information transformer-producing, managing, acquiring, modifying, displaying, or transmitting information that can be as simple as a single bit or as complex as a multimedia presentation. As the vehicle for delivering the product, software acts as the basis for the control of the computer (operating systems), the communication of information (networks), and the creation and control of other programs (software tools and environments).

Software delivers the most important product of our time-information. It transforms personal data (e.g., an individual's financial transactions) so that the data can be more useful in a local context; it manages business information to enhance competitiveness; it provides a gateway to worldwide information networks (e.g., the Internet) and provides the means for acquiring information in all of its forms.

The role of computer software has undergone significant change over a span of little more than 50 years. Dramatic improvements in hardware performance, profound changes in computing architectures, vast increases in memory and storage capacity, and a wide variety of exotic input and output options have all precipitated more sophisticated and complex computer-based systems. Sophistication and complexity can produce dazzling results when a system succeeds, but they can also pose huge problems for those who must build complex systems.

The questions that were asked of the lone programmer are the same questions that are asked when modern computer-based systems are built:

- Why does it take so long to get software finished?
- Why are development costs so high?
- Why can't we find all errors before we give the software to our customers?
- Why do we spend so much time and effort maintaining existing programs?
- Why do we continue to have difficulty in measuring progress as software is being developed and maintained?

UE – CHARACTERISTICS OF SOFTWARE

Software has characteristics that are considerably different than those of hardware:

1) Software is developed or engineered; it is not manufactured in the classical sense.

Although some similarities exist between software development and hardware manufacturing, the two activities are fundamentally different.
In both activities, high quality is achieved through good design, but the manufacturing phase for hardware can introduce quality problems that are nonexistent (or easily corrected) for software.
Both activities are dependent on people.
Software costs are concentrated in engineering.

2) Software doesn't "wear out".

Another aspect of wear illustrates the difference between hardware and software. When a hardware component wears out, it is replaced by a spare part. There are no in the process through which design was translated into machine-executable code. Therefore, software maintenance involves considerably more complexity than hardware maintenance.

3) Although the industry is moving toward component-based construction, most software continues to be custom built.

Consider the manner in which the control hardware for a computer-based product is designed and built. The design engineer draws a simple schematic of the digital circuitry, does some fundamental analysis to ensure that proper function will be achieved, and then goes to the shelf where catalogs of digital components exits. Each integrated circuit has a part number, a defined and validated function, a well-defined interface, and a standard set of integration guidelines. After each component is selected, it can be ordered off the shelf.

A software component should be designed and implemented so that it can be reused in many different programs. Modern reusable components encapsulate both data and the processing that

is applied to the data, enabling the software engineer to create new applications from reusable parts.

QUE : EXPLAIN SOFTWARE EVOLUTION

when the software is adapted to a new environment, when the customer requests new features or function, and when the application is reengineered to provide benefit in a modern context.

Def : Evolution of Software

Evolution of Software means Compare new computerized system with old system and check out the performance of both system.

Old System:

- Identify, which is the font - end and back- end tool used for developing software. Ex: FoxPro, COBOL.
-More line of code is used. there are 30 or 50 lines of coding in each module.
-This system work on dos operating system.
-We can not make the attractive system.
-DBMS database used for storing data.
-In DBMS database we can not get data easily.
-In DBMS database increase the complexity.

New System:

- Now, we develop the new system. Ex: V.B. as font end and oracle as backend.
-There is a less line of code.
-This system is work on the window operating system.
-We make the attractive system by using variouse GUI components.
- RDBMS database used for storing data
- RDBMS database we can get data easily.
- In RDBMS database reduce the complexity.

Some Law of Software Evolution

1)The Law of Continuing Change (1974). E-type systems must be continually adapted, or else they become progressively less satisfactory.

2)The Law of Increasing Complexity (1974). As an E-type systems evolves its complexity increase unless work is done to maintain or reduce it.

3)The Law of Self-Regulation (1974). The E-type system evolution process is self-regulating with distribution of product and process measures close to normal.

4)The Law of Conservation of Organizational Stability (1980). The average effective global activity rate in an evolving E-type system is invariant over product lifetime.

5)The Law of Conservation of Familiarity (1980). As an E-type system evolves all associated with it, developers, sales personnel, and users, for example, must maintain mastery of its content and behavior to achieve satisfactory evolution. Excessive growth diminishes that mastery. Hence the average incremental growth remains as the system evolves.

6)The Law of Continuing Growth (1980). The functional content of E-type systems must be continually increased to maintain user satisfaction over the system's lifetime.

7)The Law of Declining Quality (1996). The quality of E-type systems will appear to be declining unless they are rigorously maintained and adapted to operational environment changes.

8)The Feedback System Law (1996). E-type evolution processes constitute multilevel, multiloop, multiagent feedback systems and must be treated as such to achieve significant improvement over any reasonable base.

QUE : GIVE THE DIFFERENT BETWEEN SOFTWARE ENGINEERING & COMPUTER SCIENCE

Software Engineering is the establishment & use of Sound Engineering principles in order to obtain Economically Software that is reliable & works efficiently on real machines.

Software Engineering is about the technical aspects of software quality. It doesn't directly address the need for customer satisfaction or timely product delivery. It omits mention of the importance of measurement and metries.

Software Engineering Is the application of the systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software that is the application of engineering to software.

Computer Science defines, investigate, and research the new principles and
Software engineering is about implementation of those principles which are created by Computer Science.

Computer Science is a broad term that covers various topics related to computers. Topics in computer science may include Networking encryption, Processor design, and several other fields.

DIFFERENCE BETWEEN SOFTWARE ENGINEERING AND SYSTEM ENGINEERING.

A software engineering is someone who analyzed designs, codes and/or tests software.

It is difficult to define the term "System Engineer" because the term is overloaded to mean many things some. System Engineering is an inter-disciplinary activity. A system engineer can come from on of the three sources
1) The System developer. 2) The Customer. 3)Some outside organizations

QUE : WHAT IS THE ROLE OF A SOFTWARE ENGINEER?

A Software engineer is a person who builds or designs Software. A software engineer must of course, be a good programmer, be well versed in data structures and algorithms.

To make programs to promote new technology in computer.

QUE : EXPLAIN VARIOUS TYPE OF COST EFFECTED TO DEVELOPING NEW COMPUTERSIZED SOFTWARE.

In developing cost estimates for a system, we need to consider several cost elements. Among them are hardware, personnel, facility, operating, and supply costs.

1) **Hardware costs** relate to the actual purchase or lease of the computer and peripherals (for example, printer, disk drive, tape unit).

2) **Personnel costs** include EDP staff salaries and benefits (health insurance, vacation time, sick pay, etc.) as well as pay for those involved in developing the system.

3) **Facility costs** are expenses incurred in the preparation of the physical site where the application or the computer will be in operation.

This includes wiring, flooring, acoustics, lighting, and air conditioning. These costs are one-time costs and incorporated into the overall cost estimate of the candidate system.

4) **Operating costs** include all associated with the day-to-day operation of the system.

QUE : GIVE THE USE OF SOFTWARE LIFE CYCLE MODELS.

A life cycle model prescribes the different activities that need to be carried out to develop a systematic and cost-effective software product.

A software life cycle model is a descriptive and diagrammatic representation of the software life cycle.

Explain phuse development process in detail.

QUE EXPLAIN PHASES OR ACTIVITIES OF CLASSICAL WATERFALL MODEL

-The classical waterfall model is the most obvious way to develop new software.
-We shall see that it is not a practical model in the sense that it cannot be used in actual software development projects.
-We can consider this model to be a theoretical way of developing software.
-This model used to developing software documentation.

The classical waterfall model breaks down the life cycle into the set of phases.
The different phases of this model are:
Feasibility study, requirement analysis and specification, design, coding and unit testing, integration and system testing, and maintenance.

Above six phases or activity provides help to develop software.

1) FEASIBILITY STUDY:
The main aim of the feasibility study activity os to determine whether it would be financially and technically feasible to develop the product.

There are three as pects in feasibility study.

1) Technical feasibility.
2) Economical feasibility.
3) Operational feasibility.

1) Technical feasibility. Here we check out that sufficient hardware and software technology is available for developing new software.
2) Economical feasibility : Here we considers about costing of new software, here we check out new software is financial beneficial or not on development side as well as client side.
3) Operational feasibility : will the system be used if it is developed and implemented ? also we find out that staff member's of customer are capable to operate and understand new software or not.

2) REQUIREMENTS ANALYSIS AND SPECIFICATION

This Phase consists of two distinct activities, namely requirements gathering and analysis, and requirements specification as follows:

1. Requirement gathering and analysis:

Here we apply certain fact finding technique to collect require information for developing systematic and cost effective software.

This activity consists of first gathering the requirement and then analyzing the gathered requirement.

2. Requirements specification : The customer requirements identified during the requirements gathering and analysis activity are organized into a software Requirements Specification (SRS) document.

The three most important contents of this document are the

1) functional requirements : this document contains details about the input data, the processing required on the input data, and the output data to produced by software.

2) The non-functional requirements : identify the performance requirements, the required standards to be followed, etc.

The SRS document is written using end-user terminology. This makes the SRS documents understandable by the customer .

3) DESIGN:-

The goal of the design phase is to transform the requirement specified in the SRS document into a structure that is suitable for implementation in some programming language.

Software design is the process of graphical or pictorial representation of system by using collecting information.

There are Two types of design approaches are used at present.

1) the traditional design approach and 2) the object-oriented design approach

1.TRADITIONAL DESIGN APPROACH:-

The traditional design approach is currently being used by many software development houses.

Here we perform a structure design activity.

Structured design involves preparing a E – R diagram , Contact level diagram, Data flow diagrams (DFD) and modules chart or structure chart.

Here also, we create data dictionary as well as related tables with of software .

To make input , output and dialog esign.

once the structured analysis activity is complete.
structured design consists of two main activities:
1) Architectural design (also called high-level design)
2) detailed design (also called low-level design).

1) Architectural design (also called high-level design)
High-level design involves decomposing the system into modules, and representing the interfaces and the invocation relationships among the modules.
A high-level software design is sometimes referred to as the software architecture.

2) detailed design (also called low-level design).
the data structures and algorithms of the modules are designed and documented.
Several well-known methodologies are available for working out the architectural and low-level designs.

2.OBJECT-ORIENTED DESIGN APPROACH:

Object-oriented design(OOD) is a relatively new technique.
In this technique, various objects that occur in the problem domain,
first identified and the different relationships that exist among these objects.
Here we create class and reusability of class.

4) CODING AND UNIT TESTING :-

-The purpose of the coding and unit testing phase of software development is to translate the software design into source code.
-The coding phase is also called the implementation phase since the design is implemented into a workable solution in this phase.
-Each component of the design is implemented as program module.
- Two enable the engineers to write good quality programs.
- every software development organization normally formulates it's own coding standards.
- A coding standard include program codes, variable and function naming conventions , maximum number of source lines permitted in each module , etc.

-After coding is complete, each module is unit tested.
-Unit testing involves testing each module in isolation from other module , than debugging , and documenting it.
-The main objective of unit testing is to determine the correct working of the individual modules during unit testing.
-During unit testing, each module is tested in isolation as this is the most efficient way to debug the errors identified add this stage.

5)INTEGRATION AND SYSTEM TESTING:-

-During the Integration and system testing phase, the different modules are integrated in a planned manner.

Finally, after all the modules have been successfully Integrated with main modules and then after whole system will be tested

System testing consists of three different kinds of testing activities as follows:

1) α-testing :a testing is the systein testing performed by the development team.

2) β-testing : This is the system testing performed by a customers.

3) Acceptance testing:-This is the system testing performed by the customer himself after the product deliver to determine whether to accept the delivered product or to reject it.

6) MAINTENANCE:-

Maintenance involves following three kinds of activities :

1) Corrective Maintenance:

This type of Maintenance involves correcting errors that were not discovered during the product development phase.

2) Perfective Maintenance:

This type of Maintenance involves improving the implementation of the system, and enhancing the functionalities of the system according to the customers requirements.

4) Adaptive Maintenance:

Adaptive Maintenance is usually required for porting the software to work in a new environment. For example, porting man be required to get the software to work on a new computer platform or with a new operating system.

QUE : EXPLAIN ABOUT SOFTWARE PROCESS :

Process is the collection of activity, action and task that are perform when some software is to be created.

Activity

- Activity server to achieve a broad objective or specify the problem definition.
- Activity applies the regardless of the application domain-size of the product, complexity of the product which is to be applied.

Action

To composite of the set of that product a measure work product.
Example: - Architecture design.

Task

The focus on small but well define objective.
Example: - Conducting the unit test that produces the successful output.
So software engineering is a process is an adaptable apron that enable to the people doing the work. To become and choose the appropriate action and task.

GENERIC PROCESS FRAME WORK

The framework activities that are used to all software projects.

It includes following five activities.

1) Communication
2) Planning
3) Modeling
4) Construction
5) Deployment

This activity Framework activity can be use for the development of small simple programs large web application or large computer base system.

1) Communication

Before doing any technical work it is important to communicate and collaborate with customer. It is intent to understand the objective of the project and gather the requirement of the project. Which helps to define the future and function of the software?
Activities – 1) define problem 2) request clarify 3) feasibility study 4) request approval
5) collect required data.

2) Planning

This activities establishes a plan for the software development.
It describes the technical tasks to be conducted, consider the risk factor.

The resources that will be require and manage a work schedule (Which helps to guide the team how it makes the journey).

3 Modeling

The software engineering creates the model, using this model developer and the customer to better understand the software requirement,
and to make the design of system by using collected requirements.

4 Construction

The activity combines the code generation and testing. The code generation is either manual or automated. The testing is required uncovering errors in the code.
And here make accurate system.

5 Software deployment

The software is delivering to the customer who evaluates the delivered product and provides the feedback base on the evaluation.
Here check out the software with new software provides required output compare to old system.

*** Definition of software increment :**

The framework activity is applied diffractively as project progress. This activity applies repeatedly true number of project interaction process called software increment each increment produces more and more complete software.

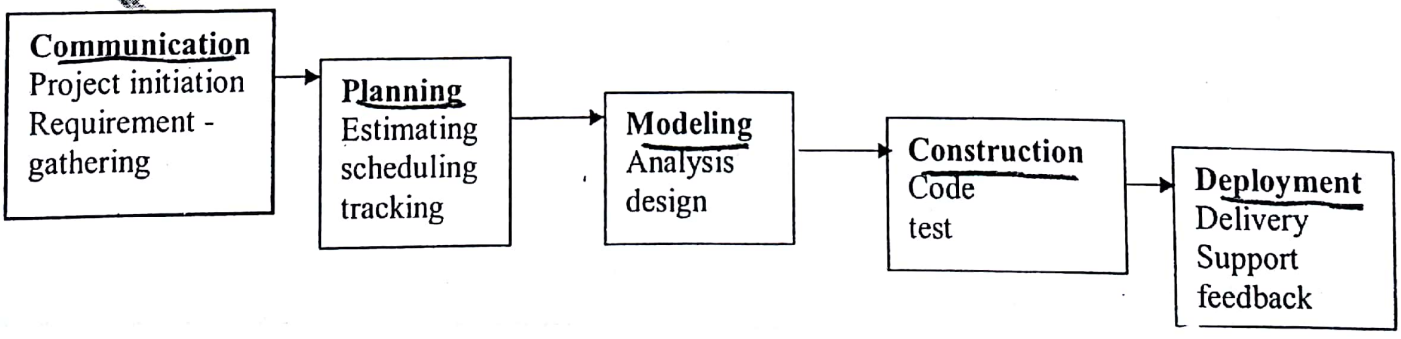
Software process model

QUE : EXPLAIN WATERFALL MODEL

It is also known as liner model. Also called sequential model.

The simplest model

-Which state that the phases (activity) are organized in liner fashion? It is use when the requirements of the problem are well understood.
-In the waterfall model the flow can be transfer from communication to the deployment in liner fashion.
-It is also known as clc (classic life cycle). Which is suggesting to the systematic sequential approach that begins with the customer requirement and progress through planning, modeling, construction, and deployment?



-This model is described the relationship quality insurance to action associated with communication, modeling and other activity.
-The project begins with the feasibility analysis and successfully demonstrating the feasibility of the project after complacent of feasibility analysis the requirement analysis and project planning begins.
-After complacent of the project planning design of the project, then after start the coding.
-Once the coding is finish the code integration and testing is done.
-After complacent of testing the system is installing and regular operation and maintenance of the system take place.

-Here all the activity can be perform in linear way; linear order of activity has some important consequence.

1) Clearly identify and of the way than begins the new phase.
   a)Certificate Mechanism (V & V validation & verification)
   b)Each phase has some define output.
   c) Intermediate products in the form of document or report.
2) Each phase complete and its output is certified. The output of this phase is the input of next phase and it can not be change or modify.

Relation of waterfall model

Any difference of ordering of the phase will generate less successful product for successful project listed in the waterfall model must be perform in linear way.

Limitation of waterfall model

1 Way of adopt (develop)
2 Freezing the requirement.
3 Not useful for developing generic product.

Fig : V - Model

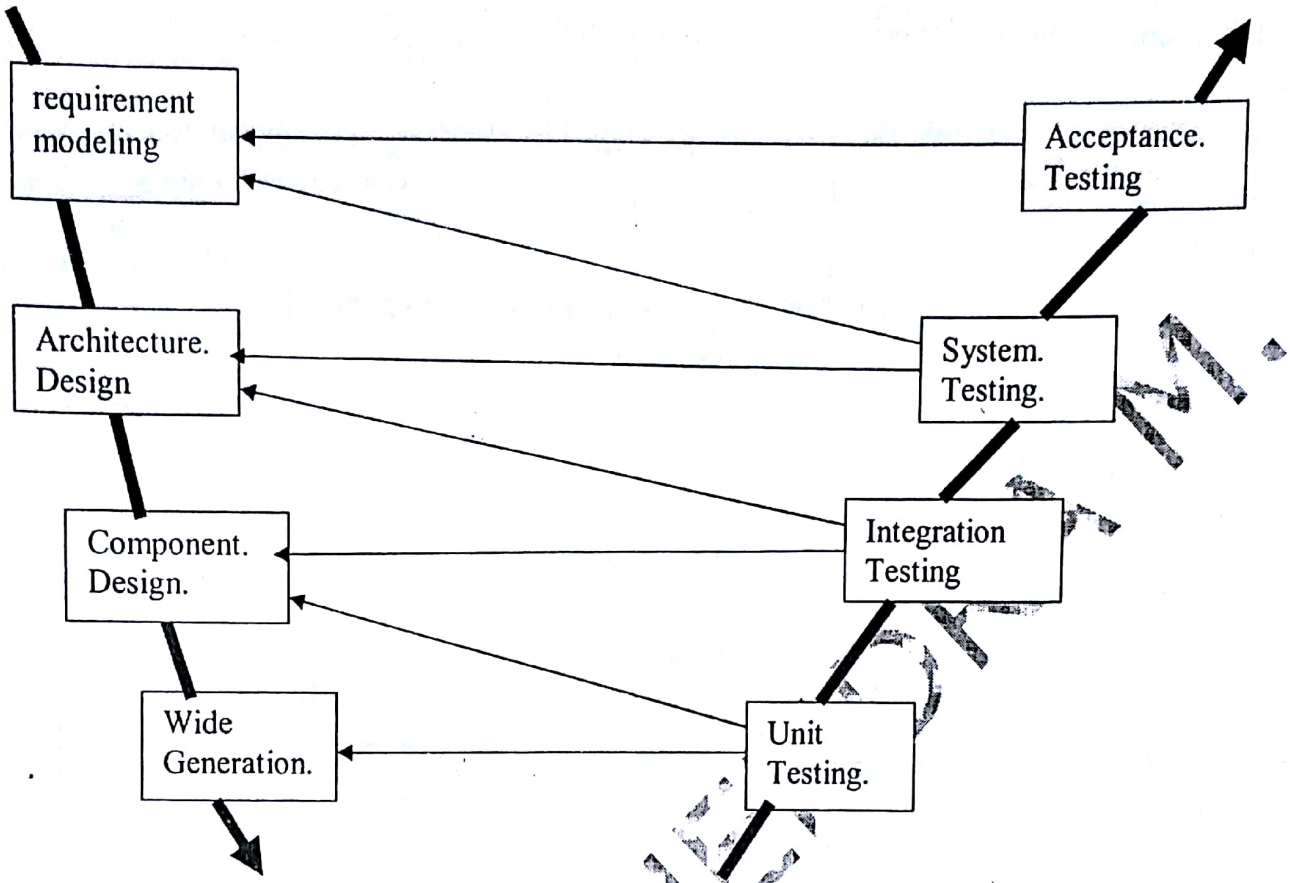
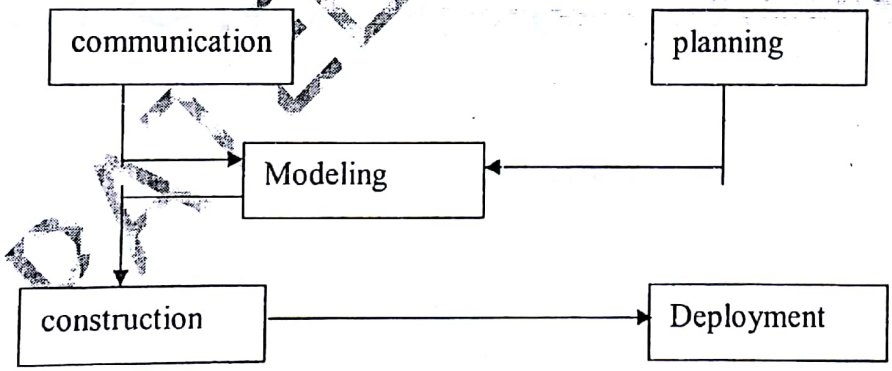
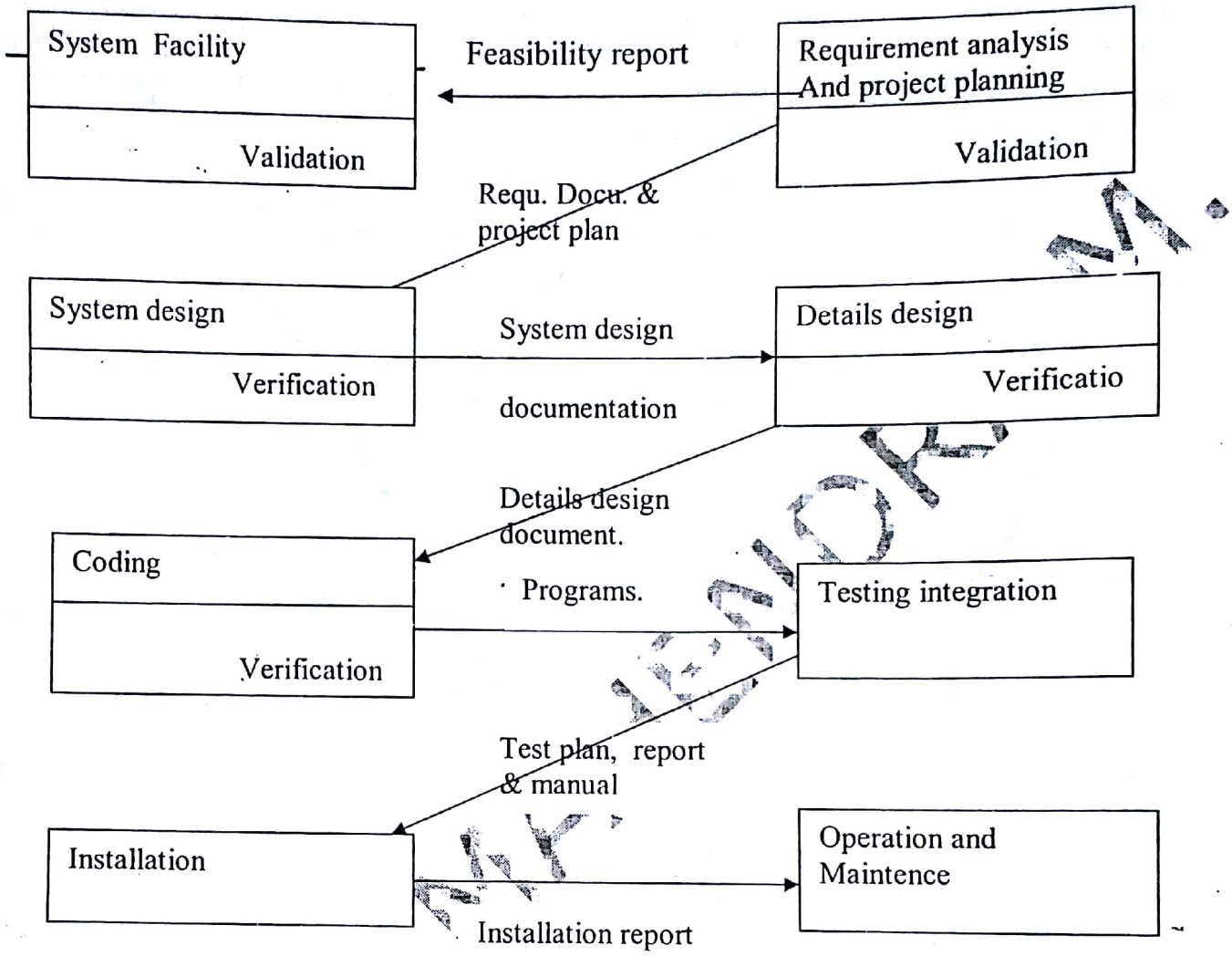


Fig : 1) Relation waterfall model



2) diagram of waterfall model



QUE : EXPLAIN INCREMENTAL OF MODEL

-Basic idea of increment of model is adding some functional capability into the system until the whole system is completed.
-User can quickly refine the function (expend or increment the functionality) in later software finish.
-Each step, extension and design modification can upgrade the system.

Advantage is:
-Result is better than existing system because each increment is physical then testing the entire system.
-The increment model combines element of linear phase process flow.
Each linear sequence produces renewable increment of the software.
Example:- The world processing software develop using increment model of deliver the basic file management, editing and document productions function in the first increment.
-In the second increment it delivers some new functionality spelling and grammar checking and more sophisticated editing and document production.
-In the third increment it at the advance page layout capability.
-In the forth increment it add the more user friendly environment and increase more functionality in it.
-In increment process model focus on the delivery of an product with each increment.
-Early increments are down version of the final product.
-It is plan for manage the technical risk.
-It is useful when unavailable for complete implementation by the business headline.
-The core product is well receive then the require staff can easily add new functionality in the next increment.
-Limitation is it is useful for existing system not for new system.

EVOLUTIONARY PROCESS MODELS

1) Prototyping process model
2) The spiral model

QUE : EXPLAIN PROTOTYPING MODEL :-

-The prototyping model requires that before carrying out the development of the actual software, a working prototype of the system should be built.
-A prototype is usually built using several short cuts.
- The short cuts might involve using inefficient, inaccurate, or dummy functions.

there are **several advantages** of building a prototype before the actual product is developed.
1) As important use of a prototype can be to illustrate the input data formats, massages, reports, and the interactive dialogues to the customer.

2) In this regard, the prototype model turns out to be especially useful in developing the Graphical User Interface (GUI) part of a system.

3) Another important situation where the prototyping model can be useful where the exact technical solutions to be adopted are unclear to the development team.
Here we have to find out technical capability of developer.

4) It is useful when we develop the new system.

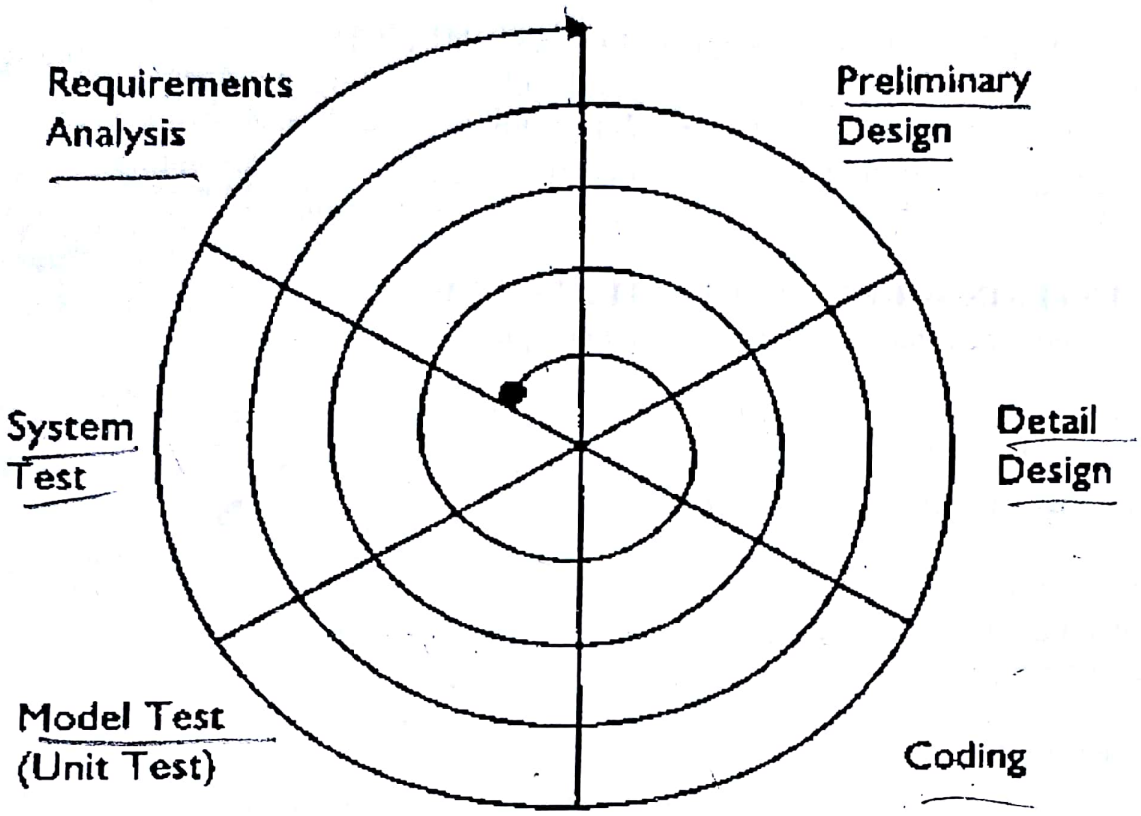
[here you refer figure from notebook of class room]
The prototyping model of software development is shown figure. As shown in figure, the phase is prototype development to control various risks. This is followed by an iterative development cycle. In this model, prototyping starts with an initial requirements gathering phase. A quick design is carried out and a prototype is built. The developed prototype is submitted to the customer for his evaluation. Based on the customer feedback, the requirements are refined and the prototype is suitably modified. This cycle of obtaining customer feedback and modifying the prototype continues till the customer approves the prototype. Once the customer approves the prototype, the actual system is developed using the iterative waterfall approach.

However, the experience gathered from developing the prototype help a great deal in developing the actual system.
Therefore, even though the construction of a working prototype might involve additional cost, for systems with unclear customer requirements and for systems with unresolved technical issues,
the overall development cost might turn out to be lower in the prototype model than that of an equivalent system developed using the iterative waterfall model.

Limitations: - It is not better for existing system.

QUE : EXPLAIN THE SPIRAL MODEL

The spiral model is originally proposed by Barry Boehm, the spiral model is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the waterfall model.
- The spiral development model is **a risk-driven process model** generate that is used to guide multi- stakeholders concurrent engineering of software intensive systems.
-It has two main distinguishing features.
1) One is a cyclic approach for incrementally growing a system's degree of definition and implementation while decreasing its degree of risk.
2) The other is a set of anchor point milestones for ensuring stakeholders commitment to feasible and mutually satisfactory system solutions.



-A spiral model is delivered into a set of framework activities defined by the software engineering team.
-Each of the framework activities represent one segment of the spiral path, the software team performs activities that are implied by a circuit around the spiral in a clock wise direction, beginning at the center.
-The first circuit around the spiral might result in the development of a products specification subsequent passes around the spiral might be used to develop a prototype and then progressively more sophisticated versions of the software.

Unlike other process model that end when software is delivered, the spiral model can be adapted to apply throughout the life of the computer software.

The spiral model is a realistic approach to the development of large-scale systems and software.

Because software evolves as the process progresses, the developer and customer better understand and react to risks at each evolutionary level.
The spiral model uses prototyping as a risk reduction mechanism but, more important, enables you to apply the prototyping approach at any stage in the evolution of the product.

The spiral model demands a direct consideration of technical risks at all stages of the project and, if property applied, should reduce risks before they become problematic.

QUE : GIVE THE SOFTWARE QUALITY ATTRIBUTES.

1) Safety 2) security 3) reliability 4) robustness
5) portability 6) adaptability 7) testability 8) modularity
9) complexity 10) user ability 11) efficiency 12) learn ability
13) reusability 14) resilience 15) under stability (simplicity)

QUE : EXPLAIN SOFTWARE QUALITY FECTOR

-There are no factor that determine the software quality.

The software quality can specify the three factors (dimension) of the product whose. quality is to be access?
Following factor affected to software quality.

1 product operation
2 product transaction
3 product Revision

1 product operation

It is define we have develop the product.
The following attribute Can increase the quality of product.

When we perform the development operation on product.then we consider the attributes are.

1 Correctness.
2 Efficiency.
3 Reliability.
4 Integrity.
5 Usability.

Correctness.
The correctness means software or program satisfies its specification.

Efficiency.
In this factor all the issue relating to the execution to the software.
Here we consider -Response time and Memory Requirement.

Reliability.
Reliability means software provides accurate requirement of customer.

Integrity

Integrity means to define accurate system module and combine all module with main module.

Usability:

Usability means new software is easy to learning and operating for end user.

2) Product Transaction.

1 Portability
2 Reusability
3 Interoperability.

Portability:

Portability manse it is easy to transfer the software from one hardware (or plate form) configuration to the another hardware configuration (or plate form).

Reusability

Reusability manse, the part of the software or the whole software can be reuse In another related software or application.

Intropability

It is the effort require to communicate The system with other system.

3 Product Revision

When The System can not be fulfill the specification of system that time the product Revision factor can be implemented that time we have consider the following attribute which Is increase the quality of product.

1 Maintainability
2 Flexibility
3 Testability

Maintainability

It is the effort require to locate the error in operating.

Flexibility

Flexibility manse It is easy to modify the program for increase the functionality of product.

Testability

Testability manse test the system and system provide require output.

QUE : EXPLAIN SOFTWARE PROBLAM.

1) Software is Expensive
2) Late, Costly and under liable
3) Problem of changing and reawake(Rework)

1) Software is Expensive

There must be require the hardware, software and employee for developing new software. Here spend the money , so software is expensive.

2) Late,Costly and under liable (error)

-The software an industry has Gained the reputation of not begin able to delivered the softwere on time within the budget.
-software is not complete the in time so increase the cost.
-This problem can arise because the budget and schedule are out of control.
-The control cost schedule and the problem of reliability are he major contribute to the software crises
-unreliable means the software failure due to some error during the software development processes.

3) Problem of changing and reawake(Rewolk)

-software is delivered and deployed and it enter an maintaining phase(All the system maintenance)

All The System need To maintain

- in some circumstances some component of system wear out and it need to be replace.
- Another problem is to maintain the software, when we not understand the requirement of software during the software development process.

QUE : EXPLAIN SOFTWARE ENGINEERING PROBLEM

1 The problem of skill
2 All schedule and quality (cost)
3 Problem of consistency

1 The problem of skil

-when we want to developing small system then there is not need skilled person.
-when we deal with the small system the technology requirement is low the project management requirement is low.
- but for the large system both are increase in a same direction The technology and project management requirement is high than small system.
-And highly qualified person must be required for developing large system.

2 Cost Schedule quality

-the cost of hardware, software, resources and developing is the major problem of software development.

the man power cost is considered to be the total no of person/month spend in the project.

schedule.

When the schedule can be out of control then the lifecycle should be increase.
So it affect to the cost of software and it decreases the quality of software.
The current time and quality is the main aim of the project.

3 Problem of Consistency: -

-The software development organization produces the software with consistent quality with consistent productivity.
-The system wants to be consistency manse software is accurate .
-The high quality low consistency and a small cycle time is the primary objective of any system.

QUE : EXPLAIN THE PRINCIPLE OF SOFTWARE ENGINEERING.

There are seven principles in the software engineering.
1 The reason at all exists.
2 Kiss(Keep it, smile and stupid)
3 Maintain the vision.
4 What you produce and other will consume.
5 Be open to the future.
6 Plan ahead for reuse.
7 Think.

1 The Reason at all exists.

- Before specifying system requirement.
- Before noting system functionality(Method and process)
- The software system exists from one reason "To provide value to its user"
- Before determine the hardware platform.
- Before determine Development Process.
- Ask the question yourself "Does this function add the real value to the system".
- If answer is known don't do it. If answer is do it.

2 Kiss(Keep it, smile, stupid)

- A manufactured is considered is design and development effort all the design should be simple as possible. But not simple.
- Using this system, Using this principle system is more easily understand and easily maintain.
- A system is simple is does not means quick and dirty.
- This principle often takes lot of through and multiple intersections to simply the system.

3 Maintain the vision.

- The clear vision is essential for the success of the software part.
- Using empowered architecture, who can hold the vision and it enforce the complain helps insure very successful project.
- Without vision the software project almost fail.

4 What you produce other will consume.

- A someone as will have to understand what you are doing like specify the objective, designing and implementing the system.
- The code is concern or those that must maintain and extend the system.
- Using this job user can easily add the new functionality or value to the system.
- Some way have also debugged the code, we write.(it is easy to understand for each and every one).

5 Be open to the Future.

- A system with the long life time as more value.
- The software lifetimes are typically measure in months instead of the years.
- Existing system must be ready to add the new functionality and easily of form other Changes in it.
- A system is prepared for all possible answer by creating system that solves the general problem. Not solve the all specific once.

6 The plan ahead for reuse

- A reuse the software is save the time and effort of the software product.
- Achieving the high level of reuse, Is the hardest goal to a Kong bless developing software system.
- The planning ahead for reuse reduces the cost and increase the value of reusable component.

7 Think

- The clear planning and complete through before action, in almost produce the better result.
- When you are think about something, you are more likely do, do it right.
- You also gain the knowledge, how to do write again.
- The side effect of thinking is learning to recognize when you does not know something at which point you can research the answer.

PROFESSIONAL AND ETHICAL RESPONSIBILITY.

- Software engineering involves wider responsibilities than simply the application of technical skills
- Software engineering must behave in an honest and ethically responsible way if they are to be respected professionals
- Ethical behavior is more than simply uploading the law.

QUE : EXPLAIN ISSUE OF PROFESSIONAL RESPONSIBILITY

1. **Confidentiality.**
   Engineering should normally respect the confidentiality of their employers or clients irrespective of whether or not a formal confidentiality agreement has been signed.

2. **Competence**
   Engineering should not misrepresent their level of competence. They should not knowingly accept work which is out with their components.

3. **Intellectual property rights**
   Engineers should be aware of local lows governing the use of intellectual property such as patients, copyright, etc. They should be careful to ensure that the intellectual property of employers and clients is protected.

4. **Computer misuse**
   Software engineers should not use their technical skills to misuse other people's computers. Computer misuse ranges from relatively trial (Game playing on an employer's machine say) to extremely serious (dissemination of viruses).

❖ **ACM/IEEE code of Ethics**

- The professional societies in the US have cooperated to produce a code of ethical practice.
- Members of these organizations sign up to the code of practice when they join.
- The code contains either principles related to the behavior of the decisions made by professional software engineers, including practitioners, educators, managers, supervisors and policy makers, as well as trainers and students of the profession.

❖ **Code of Ethics preamble.**

- The short version of the code summ arizes aspirations at a high level of the abstraction; the clauses that are included in the f ıll version give examples and details of how these aspirations change the way we act as software engineering professionals. Without the aspirations the details can be become legalistic and tedious. Without the details, the aspirations can become high sounding but empty; together, the aspiration and the details Form cohesive code.
- Software engineers shall commit themselves to making the analysis specification design development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public.

software engineers shall adhere to the following eight principles:

❖ **Code of Ethics Principles**

- **Public**
  Software engineers shall act consistently with the public interest.
- **Client and employer.**
  Software engineers shall act in a manner that is in the best interest of their client and employer consistent with the public interest.
- **Product**
  Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
- **Judgment**
  Software engineers shall maintain integrity and independence in their professional judgment.
- **Management**
  Software engineers managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
- **Profession**
  Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
- **Colleagues**
  Software engineers shall be fair to and supportive of their colleagues.
- **Self**
  Software engineers shall participate in lifelong learning regarding the p actice of their profession and shall promote an ethical approach to the practice of the profession.

QUE : EXPLAIN SOFTWARE MYTHS(MYTHOLOGY)

- The software myths propagated misinformation and confusion.
- The no of attributes that made them insider for instance.
- They appear to be reasonable statement of fact (Sometime containing elements of truth).
- The professional recognized means for what they are misleading attributes that caused serious problem for manager, technical people and the customers.

- There are three myths.
  1) Manager Myths.
  2) Practitioners Myths.
  3) Customer Myths.

1) Manager Myths

- Manager is the highest responsible person for software.
- Manager in most discipline are after under the pressure to maintain the budget.
- The keeps scheduled from sleeping and improve the quality.
- The software manager believes in the software mythology(software myths)

2) Partitioned Myths

- During the early days the software programming cause view as an art from the software mythology are still believe the software partitioned.

3) Customer Myths

- Customer who request the computer software. The customer believe mythology About software because manager and practitioner to little to correct misinformation. Customer myths lead to falls expectation and customer is dissatisfied with developer.