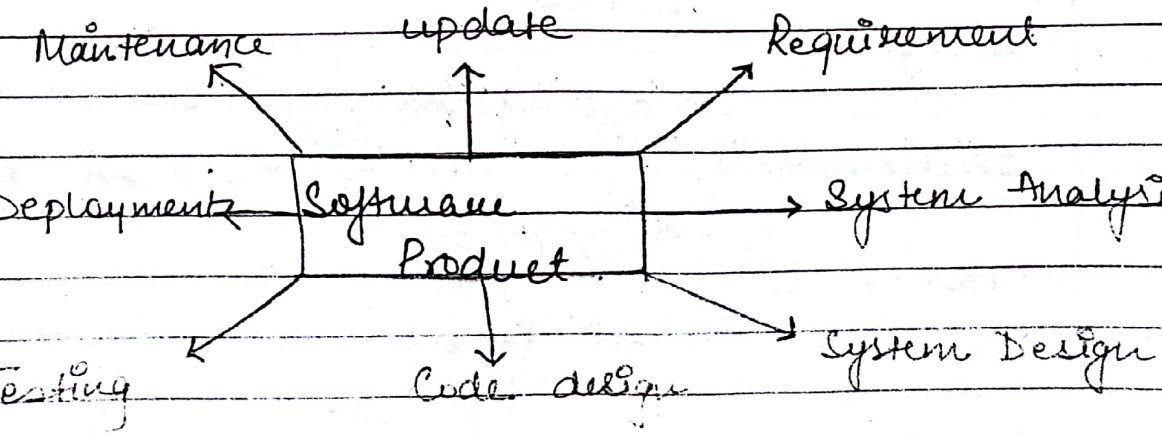
Software Engg

'Software' is more than just a program code.
It is a compination of programs*.
A program is an executable code which serve some computational purpose.
Software is considered to be collection of executable programming code, associated libraries and documentations. Software when made for specific req. is called software product.

'Engg' on the other hand is all about developing products using well defined scientific principles & methods.

'Software engg' is an engg. branch associated with development of software product using well defined scientific principals, methods & procedures. The outcome of software engg is an efficient & reliable software product.

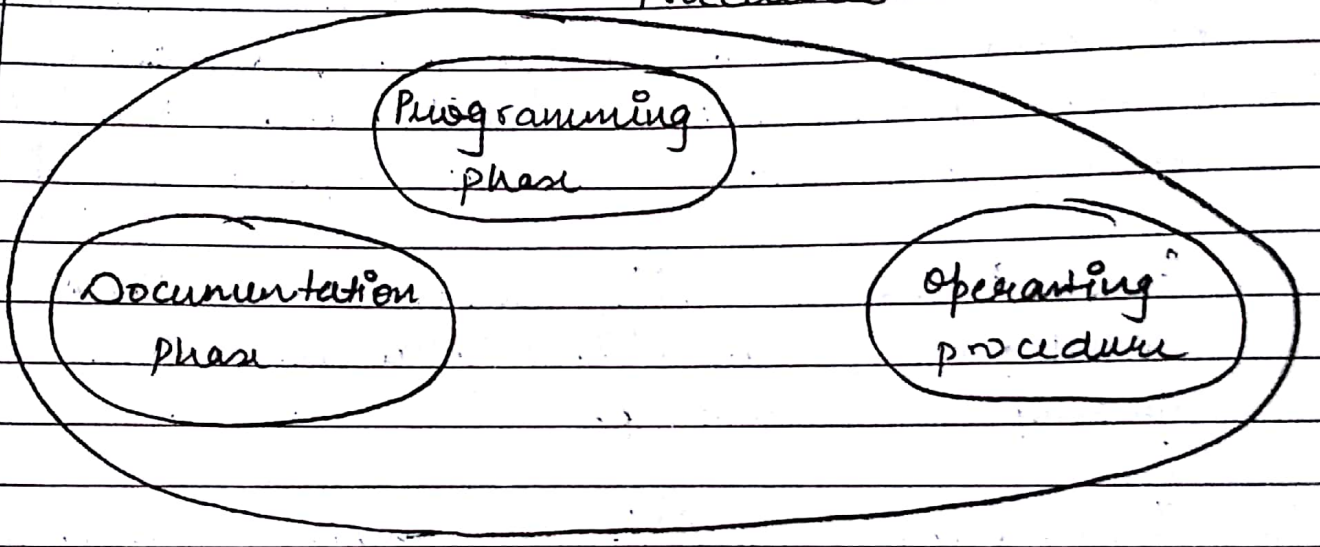


stated that.

* Fritz Bauer – Software engg. is the establishment & use of sound engg. principles in order to obtain economically software that is reliable & work efficiently on real machines.

Software Components:

S/W = Program + Documentation + Operating Procedures



- Analysis specification
  - Formal Definition
  - Context Diagram
  - Data flow diag.
- Design
  - flow chart
  - ER - diagram
- Implementation
  - source code
  - Cross Reference
- Testing
  - Test Data
  - Test Results

fig - List of Document Manual

E-R → Entity Relation ship

Page No. _____
Date _____

Operating procedure – It consist of instruction to set up a user software system & instructions on how to react to system failure. List of operating procedures manual is given below.

- operating procedure
  - User manual
    - System overview
    - Programers guide / Tutorial
    - Reference Guide
  - operational manual
    - Installation guide
    - System Administrational guide

Components of SE

- Programs
- Documentation
- Operating procedure

Software Paradigms

Software development

(Programming)

Software Design

Software design Paradigm

Design
Maintenance
Programming.

Need of Soft. Engg.

- Large software
- Scalability
- Cost
- Dynamic Nature
- Quality Management

Characteristics of good software

1. Operational
2. Transitional
3. Maintenance

1. Operational –
   - Budget
   - Usability
   - Efficiency
   - Correctness
   - Functionality
   - Dependability
   - Security
   - Safety

2. Maintenance -
- Modularity
- Maintainability
- Flexibility
- Scalability ~ backup

3. Transitional -
- Portability
- Interoperability - Ex exchange & info. make use of info.
- Reusability - components.
- Adaptability

Software Process

A software process is a set of activities whose goal is the development or evolution of software.

Fundamental activities in all software processes:-

- Specification - What the system should do & its development constants.
- Development - production of the software system (design & implementation)
- Validation - Checking that the software is what the customer wants.
- Evolution - Changing the software in response to changing demands.

Software Process Model

A software Process Model is a simplified representation of a software process, presented from a specific perspective.

- Examples of process perspectives:

Workflow ~~presentation~~ perspective - Represents input, output & dependencies.

Data flow ~~person~~ perspective - Represents data transformation activities.

Role/Action perspective - Represents the roles/activities of the people involved in the software process.

- Generic process models

→ Waterfall
→ Evolutionary development
→ Formal transformation
→ Integration from reusable components.

The software process is the way in which we produce software. This differs from organisation to organisation

Problem arises in software process,

1. Not enough time
2. Lack of knowledge
3. Wrong motivation
4. Insufficient commitments

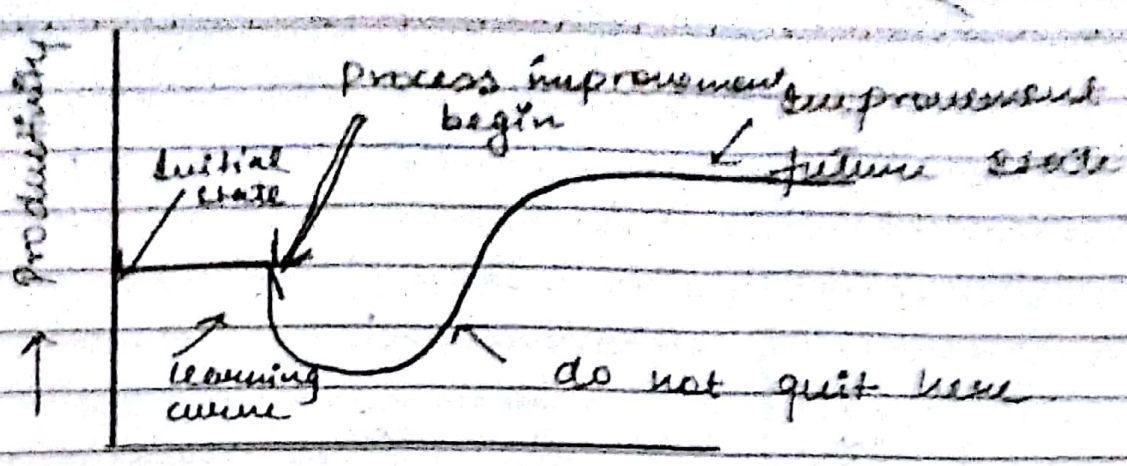
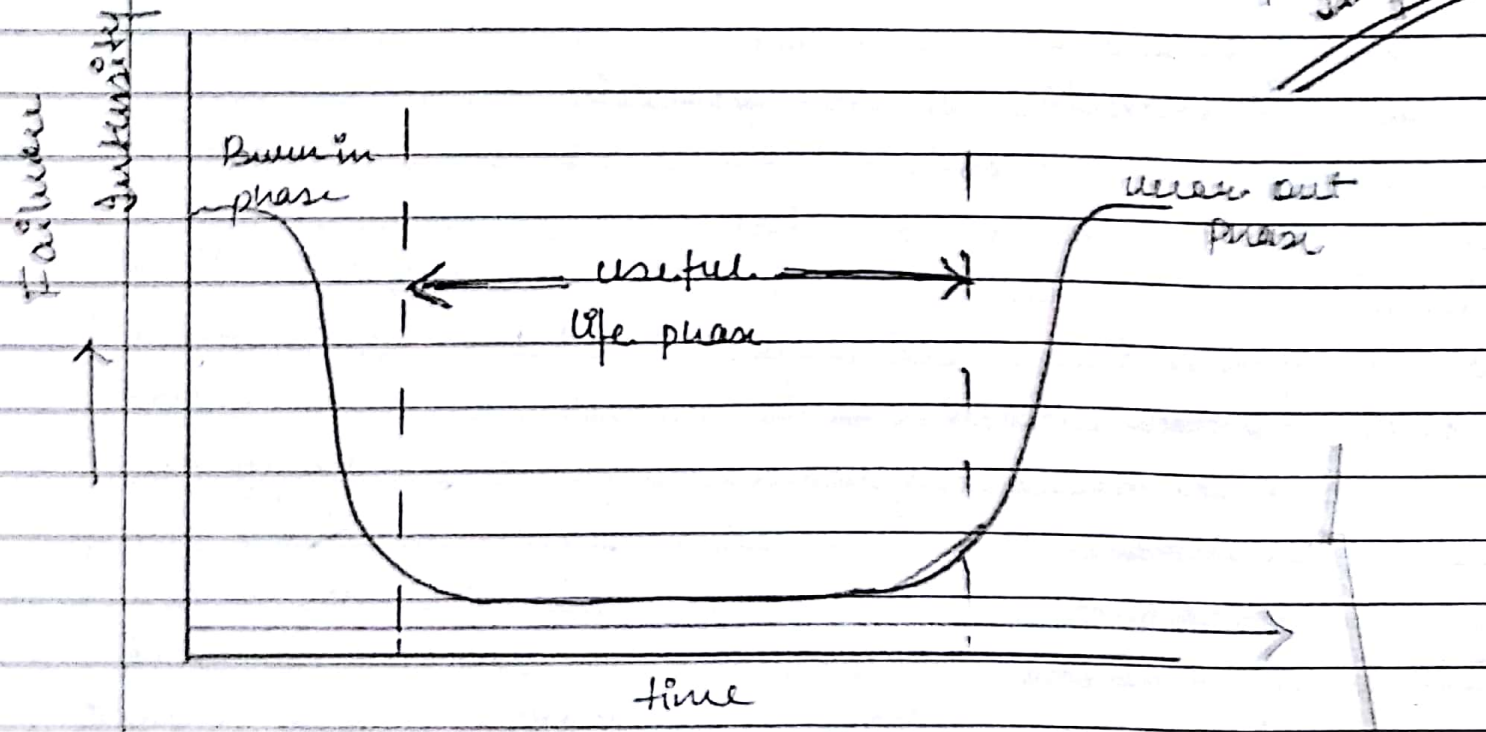


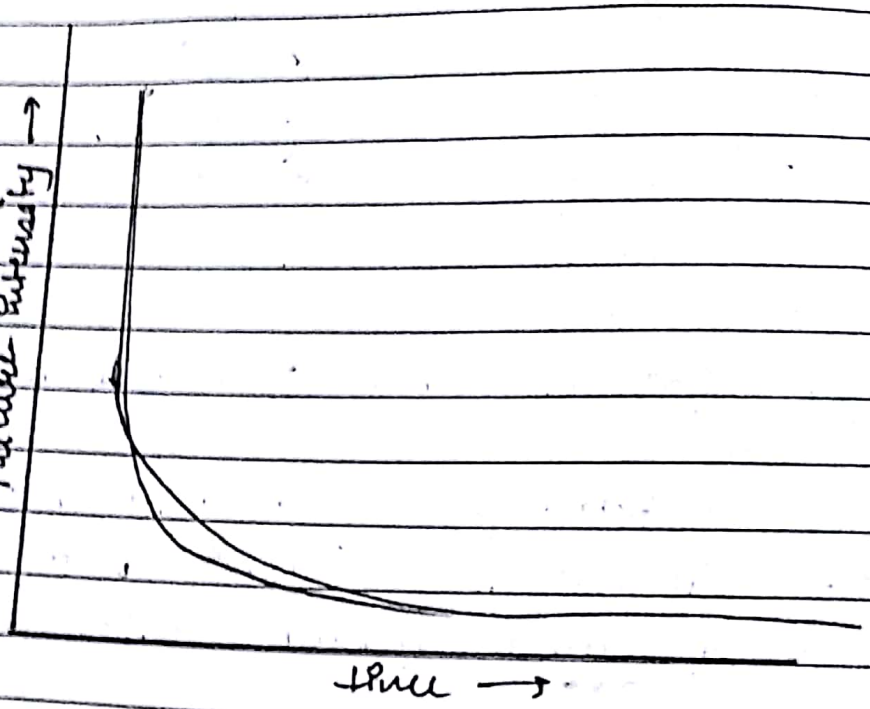
fig:- process improvement learning curve

Software characteristics

Special characteristic - "It is not wear out"
There is a well known Bath tub curve



Software Curve.



Software is not manufactured.

Reusablity of components

Software is flexible.

Software Application.

1. System Software
   Real time software
   Embidded software
   Business software
   P. Computer software
   Artificial intelligence software
   Web based software
   Engg. & Scientific software

Page No. _____
Date _____

Costs of Software Engg.

Roughly 60% of costs are development costs, 40% are testing costs. For custom software, evolution costs often exceed development cost.

Cost vary depending on the type of system being developed & the requirements of system attributes such as performance & system reliability.

Distribution of costs depends on the development model that is used.

Cost of SEC %) = Development cost + Testing cost(%)
60 % + 40 %

CASE

Computer Aided Software Engineering.

Software systems which are intended to provide automated support for software process activities, such as requirements analysis, system modelling, debugging & testing.

Upper Case	Lower Case
Tools to support the early process activities of requirements & design.	Tools to support later activities such as programming, debugging & testing.

Attributes of Good Software.

The software should deliver the required. functionality & performance to the user & should be maintainable, dependable, efficient & usable

- Maintainability
  Software must (easily) evolvable to meet changing needs.

Dependability
Software must be trustworthy & work with all data

Efficiency
Software should not make use of wasteful use of system resource.

Usability –
S/w must be usable by the users for which it was designed.

Key Challenges in Modern Software Engg.

(1) Legal systems – Old, valuable systems must be maintained & updated.

(2) Heterogeneity – Systems are distributed & include a mix of hardware & software.

(3) Delivery – There is (↑)ing pressure for fastest delivery of softwar

Page No.
Date

Trust - Developing techniques that demonstrate that software can be trusted by its user.

Professional & Ethical Responsibility.

Software engg. involves wider responsibility than simply the application of technical skills.

Software engrs must behave in an honest & ethically responsible way. if they are to be respected as professionals

Ethical behaviour is more than simply upholding the law

Issue of Professional Responsibility

Confidentiality - Engineers should normally respect the confidentiality of their employers or clients even without a formal confidentiality agreement

Competence - Engineers should not misrepresent their level of competence. They should not knowingly accept work which is beyond their competence.

Intellectual property rights - Engineers should be careful to ensure that the intellectual property of employers & clients is protected & know the local laws governing IP.

Computer misuse - Software engineers should not use their technical skills to misuse other

P.C

Software Design

Next step is to print out whole knowledge of the requirement & analyse on the disk & assign the software products. The i/p from user and information gathered in requirement gathering phase are the i/p of this step. The o/p of this step comes in the form of two designs: logical design & physical design. Engg. produce meta-data & data dictionaries, logical diagrams, data-flow diagrams & in some cases pseudo codes.

Coding - This step is also known as programming phase. The implementation of software design starts in terms of writing program code in the suitable programming language & developing error-free executable program efficiently.

Testing - An estimate says that 50% of whole software development process should be tested. Errors may ruin the software from critical level to its own removal. Software testing is done while coding by the developers exact & thorough testing is conducted by testing experts at various level of code such as module testing, program testing, product testing, in house testing & testing the product at user's end.

(3) Integration - Software may need to be integrated with the libraries, databases & other program(s). This stage of SDLC is involved in the integration of software with outside entities.

(4) Implementation - Software may need to be This means installing the software on user machine. At times, software needs post-installation configurations at user end. Software is tested for portability & adaptability & integration related issues are solved during implementation.

(5) Operation & Maintenance -
This phase confirms the software operations in terms of more efficiency & less error. If required, the user are trained on, or aided with the documentation on how to operate the the software & how to keep the software operational. The software is maintained timely by updating the code according to the changes taking place in user end environment or technology.

Same para done

Operation & Maintenance.

This phase confirms the software operation in terms of more efficiency & less error.

If required the users are trained on, or aided with the documentation on how to operate the software & how to keep the software operational.

The software is maintained timely by updating the code according to the changes taking place in user end environment or technology.

This phase may face challenges from hidden bugs & real world unidentified problems.

Disposition.

As time elapses, the software may decline on the performance front.

It may go completely obsolete or may need intense upgradation.

Hence a pressing need to eliminate a major portion of the system arises.

This phase includes archiving data & required software components, closing down the system, planning disposition activity & terminating system at appropriate end of system time.

IEEE Standard – The software life cycle

The period of time that starts when a software product is conceived & ends with when the product is no longer available for use. The software life cycle typically inclu

SDLC Model

1) Build & fix Model
2) Waterfall Model
3) Iterative Enhancement Model
4) Spiral Model
5) V-Model
6) Big Bang Model
7) Evolutionary development Model
8) Prototyping Model
9) RAD Model

1) Build & Fix Model

Sometimes a product is constructed without specification or any attempts at design. Instead the developed simply builds a product that is modified as many times as necessary to satisfy the client.

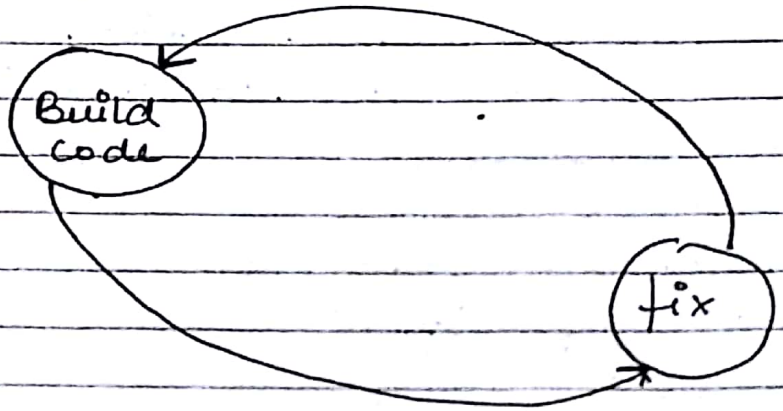
This an adhoc approach & not well defin

Basically it is two phase model

Write code

Fix:- fixing in the context may be error correction or addition of further functionality.

In addition, Maintence of product can be extremely difficult without specification or designed documents.



* Product is constructed without specifications or any attempt at design.
Maintenance is practically not possible
Adhoc approach is not well defined
No room for structure design
Simple two phase model
Suitable for small programming exercise of 100 or 200 line of code soon becomes unfixable & unenhanceable.
Unsatisfactory for software for any reasonable life

2) Waterfall Model

- Waterfall model is the simplest model of software development paradigm.
- It says the all the phases of SDLC will function one after another in linear manner. This is

where the first phase

It is named waterfall because its diagrammatic representation resembles a cascade of waterfall. This model is easy to understand & reinforce

Design
Implementation
&
Requirement gathering
System Analysis
Coading
Testing
Implementation
Operations &
Maintenance

This model is easy to understand & reinforce the notion of "define before design" & "design before code."
The model expects complete & accurate requirements early in the process, which is unrealistic.

Requirement gathering & specification - the goal of this phase is to understand the exact requirements of the customer & to document them properly. This phase produced a large document, written in natural language, contains a description of what the system will do without description how it will be done. The resultant document is known as software requirement specification (SRS)

System Analysis & Design Phase
The SRS is produced in previous phase which contains the exact requirement of the customer. This work is documented & known as software design description (SDD) document

Coding : The goal of this phase is to transform the requirements into a structure that is suitable that is suitable for implementation in some programming language

Implementation & unit testing, integration & System testing - During this phase, design is implemented. If (software design description) SDD is complete. The implementation or coding proceeds smoothly.

System testing involves the testing of the entire system, where as software is the part of the system.

This is very important phase - Effective testing will contribute to the delivery of higher

quality Software product, more satisfied users, lower maintenance cost, & more accurate & reliable results.

Problems of waterfall model

1. It is diff. to define all requirement at the beginning of a project.
2. This model is not suitable for accommodating any change.
3. A working version of the system is not seen until late in the project's life.
   It does not scale up well to large projects

ADVANTAGE

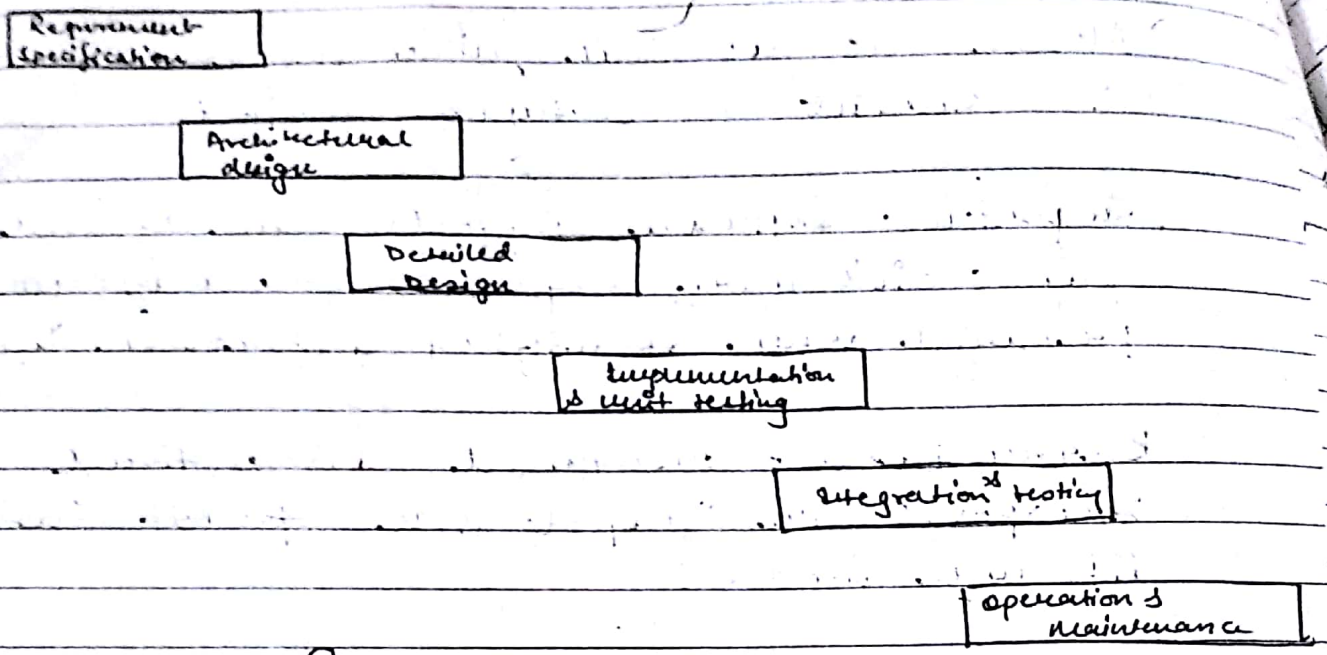
1. Natural approach for problem solving.
2. Conceptually simple, clearly divides the problem into distinct independent phase
3. Easy to administer in a contractual setup - each phase is
4. May fix hardware & other technologies too early.
5. Assumes that requirements can be specified & frozen early.
6. Very document oriented, requiring docs at the end of each phase

Iterative Enhancement model

This model has the same phases as the water model, but with fewer restrictions.

Date
Page

Iterative Enhancement Model



Spiral Model

Spiral Model can be seen as if you choose one SDLC model & combine it with cyclic process.

This model considers risk, which often goes un-noticed by most other models.

The model starts with determining objectives & constaints of the software at the start of one iteration.

Next phase is of prototyping the software. This includes risk analysis. Then one standard SDLC model is used to build the software.

Date / /
Page

In the fourth phase of the plan of next iteration is prepared.

Model do not deal with uncertainty which is inherent to software projects.

Important software projects have failed because project risk were neglected & nobody was prepared when something unforeseen happened.

Barry Boehm recognized this & tried to incorporate the "project risk" factor into a life cycle model.

The result is the spiral model, which was presented in 1986.

~~Spiral Model~~

The radial dimension of the model represents the cumulative costs. Each path around the spiral is indicative of (increased costs. The angular dimension represents the progress made in completing each cycle. Each loop of the spiral from X-axis clockwise through 360° represents one phase. One phase is split roughly into four roughly into four sections of majority.

Sectors Major Activities

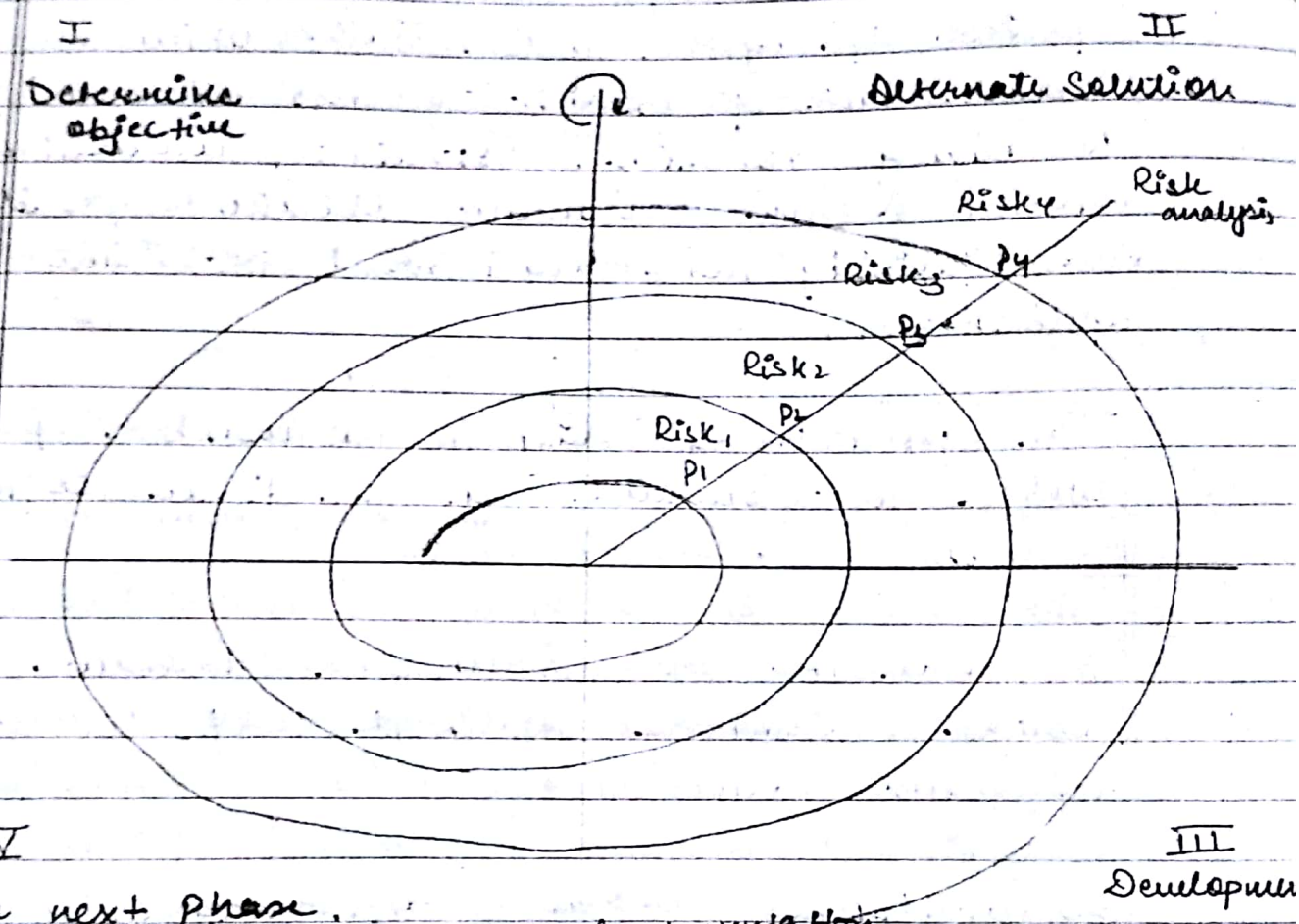
Planning: Determination of objectives, alternati & constraints.

Risk Analysis: Analyze alternatives & attempts to

identify & resolve the risks involves.

Development : Product development & testing product.

Assessment : customer evaluation.



(when customer is not satisfied, again the cycle goes on)

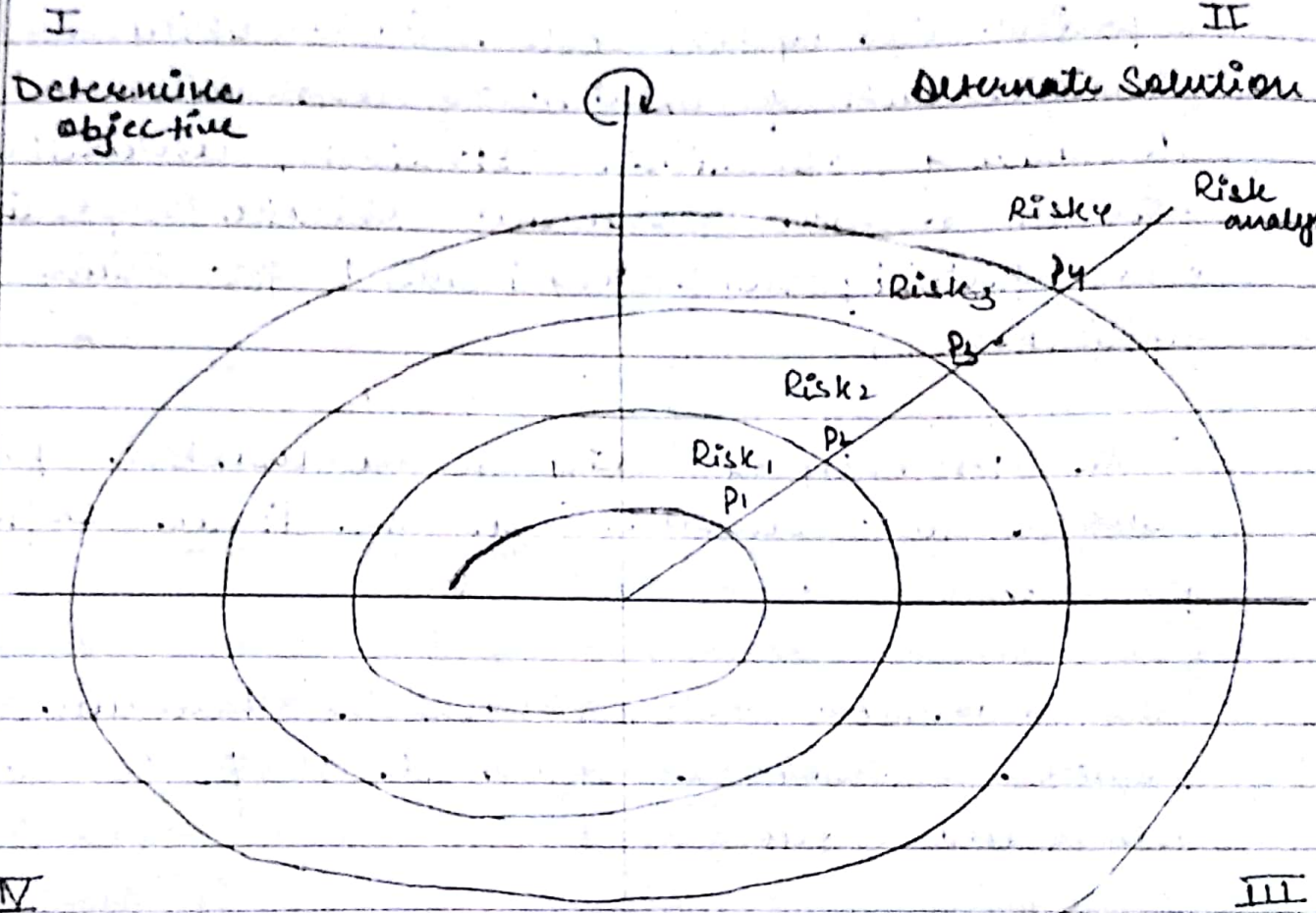
An important feature of the spiral model is that each phase is completed with a review by the people concerned with the project (designers & programmers)

The advantage of this model is the wide range of options to accommodate the good

identify & resolve the risks involves.

Development : Product development & testing product.

Assessment : customer evaluation.



(when customer is not satisfied, again the cycle goes on)

An important feature of the spiral model is that each phase is completed with a review by the people concerned with the project (designers & programmers)

The advantage of this model is the wide range of options to accommodate the good

Jump

It contains alternative waterfall prototype model

Spiral Model * * *

Developed by :- Barry Bochm (1986)
Main features :- Handling Uncertainty
Phases :- ① Planning
- Determine objectives & Alternatives
- Constraints.

② Risk analysis
- Identify risks, classify into different levels
- Alternative solution & Plan ahead

③ Development & testing
④ Assessment :- Customer Evolution (Feedback)

Phase 1 : Planning → Analysis Risks → Prototype is built → Customer Evaluations & feedback.
Phase 2 : Prototype refined. → Requirement in documental ~~validation~~ validation by customer → Final prototype
Phase 3 :- Risk are now known + Traditional approach for document.
Focus Area :- Plan for the next cycle before ahead
- Identify Problems → Classify into different levels → Alternative them before they affect the software.
- Continuous :- Validation (review) by concerned people.

This (spiral) model of development combines the features of the prototyping model & the waterfall & alternative model.
This spiral model is intended for large

expensive & complicates, aids risk managment at regular state of risk.

Spiral.

Selection of life cycle Model

Requirement
Development team
Users
Project type & associated risks

Based on Characteristic of Requirement

Are requirements easily understandable & defined

Do we change requirements quite often

	WFM	PM	ITM	EVM	SPM	RAD
Easily understandable & defined requirement	Y	N	N	N	N	Y
Do we change requirements quite often	N	Y	N	N	Y	N
Can we defined requirem easily in the cycle	Y	N	N	Y	Y	N
Requirements are indicating a complex system to be built.	N	Y	Y	Y	Y	N

Page No.
Date

Development team	WFM	PTM	ITM	EVM	SPM	RI
less experience on similar projects	N	Y	N	N	Y	I
less domain knowledge	Y	N	Y	Y	Y	A
less experience on tools to be used	Y	N	N	N	Y	A
Availability of training if req.	N	N	Y	Y	M	L