UNIT-2 Part-2

1) life-cycle phases:-

→ the most discriminating characteristic of a successful s/w development process is well defined separation b/w "research & development" activities and "production" activities.

→ this is true for conventional as well as iterative process.

→ most unsuccessful projects exhibit one of the following characteristics.

* An overemphasis on research & development
* An overemphasis on production.

→ A modern s/w development process must be defined to support the following:-

1) Evaluation of the plans, req & architecture, together with well-defined synchronization points.
2) Risk management & objective measures of progress and quality.
3) Evalution of system capabilities through demonstrations of increasing functionality.

→ there are different phases in life cycle they are

1) Engineering & production stages
2) inception
3) Elaboration
4) construction
5) Transition

Engineering & production stages :-

→ At first order are the following two stages of the life cycle :-

1) The engineering stage, driven by less predictable but smaller teams doing design and synthesis activities.

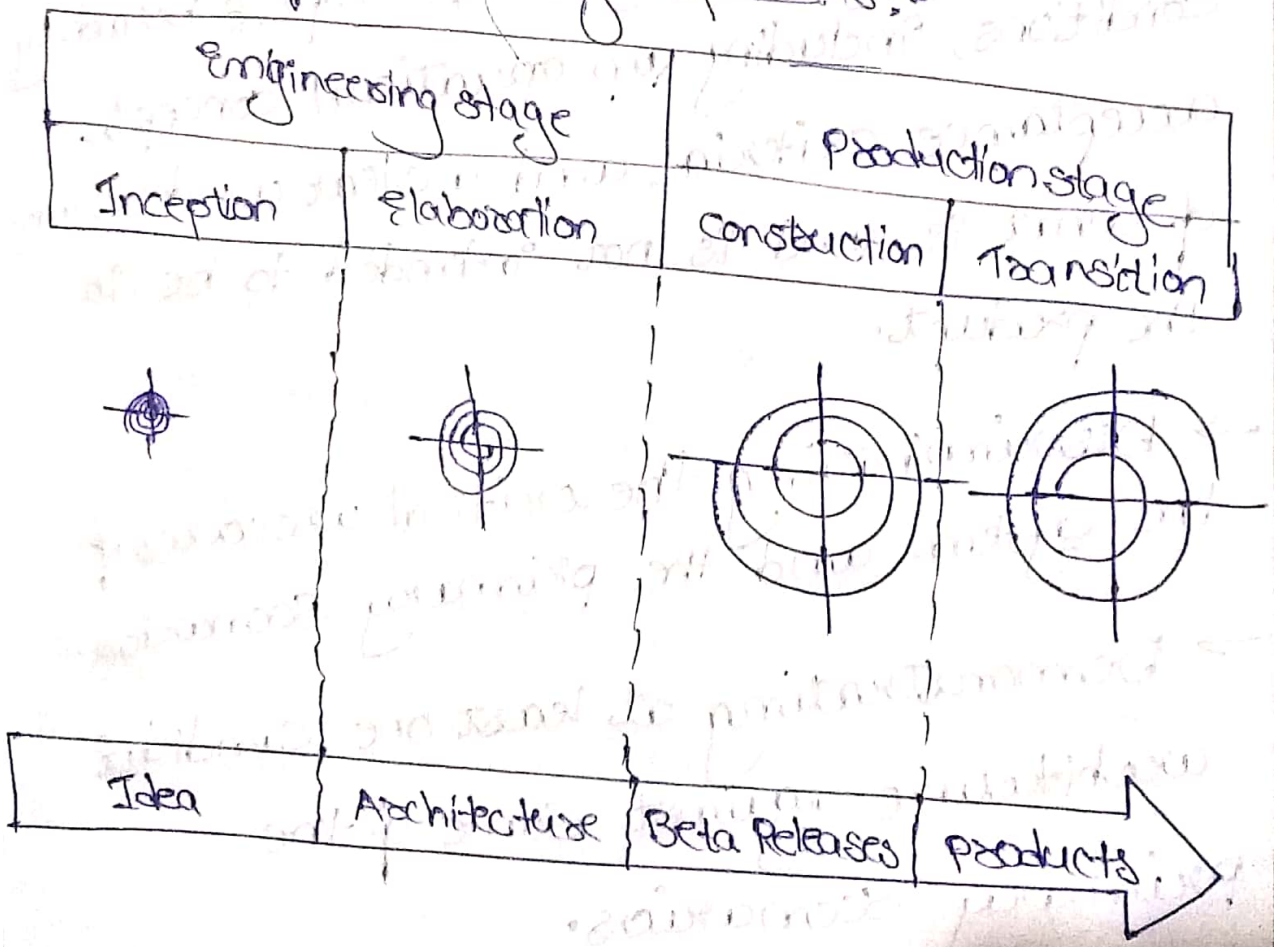
2) The production stage, driven by more predictable but larger teams doing construction, test, and deployment activities.

→ Summarizes the differences in emphasis b/w these two stages.

life-cycle Aspect	Engin stage	Production Stage
Risk reduction	schedule, technical feasibilty	Cost
Products	Architecture baseline	Product release baselines
Activities	Analysis, design, planning	Implementation, testing

Assessment	Demonstration, inspection, analysis	Testing
Economics	Resolving diseconomies of scale	Exploiting economies of scale
management	planning	operations

→ Phases of the life cycle process:-



2) Inception phase :-

→ the overriding goal of the inception phase is to achieve concurrence among stakeholders on the life-cycle objectives for the project.

Primary objectives :-

→ Establishing the project's s/w scope & boundary conditions, including an operational concept, acceptance criteria, and a clear understanding of what is and is not intended to be in the product.

→ Discriminating the critical use cases of the system and the primary scenarios.

→ Demonstrating at least one candidate architecture against some of the primary scenarios.

→ Estimating the cost & schedule for the entire project.

→ Estimating potential risks.

Essential activities :-

→ formulating the scope of the project.
→ Synthesizing the architecture.
→ planning and preparing a business case.

Primary Evaluation criteria :-

→ Do all stakeholders concur on the scope definition and cost & schedule estimates?
→ Are requirements understood, as evidenced by the fidelity of the critical use cases?
→ Are the cost and schedule estimates, and development processes credible?
→ Do the depth & breadth of an architecture prototype demonstrate the preceding criteria?
→ Are actual resource expenditures versus planned expenditures acceptable?

3) Elaboration phase :-

→ During the Elaboration phase, an Executable architecture prototype is built in one or more iterations, depending on the scope, size, risk and novelty of the project.

Primary objectives :-

→ Baselining the architecture as rapidly as practical.

→ Baselining the vision

→ Baselining the high-fidelity plan for the construction phase.

→ Demonstrating that the baseline architecture will support the vision at a reasonable cost in a reasonable time.

Essential activities :-

→ Elaborating the vision

→ Elaborating the process infrastructure.

→ Elaborating the architecture and selecting components.

Primary evaluation criteria:-

→ Is the vision stable?

→ Is the architecture stable?

→ Does the executable demonstration show that the major risk elements have been addressed and credibly resolved?

→ Is the construction phase plan of sufficient fidelity, and is it backed up with a credible basis of estimate?

→ Are actual resource expenditures versus planned expenditures acceptable?

4) construction phase :-

→ During the construction phase, all remaining components and application features are integrated into the application, and all features are thoroughly tested.

→ Newly developed s/w is integrated where required.

→ the construction phase represents a production process, in which emphasis is placed on managing resources & controlling operations to optimize costs, schedules and quality.

Primary objectives :-

→ minimizing development costs by optimizing resources and avoiding unnecessary scrap & rework.

→ Achieving adequate quality as rapidly as practical.

→ Achieving useful versions as rapidly as practical.

Essential activities :-

→ Resource management, control and process optimization.

→ Complete component development and testing against Evaluation criteria.

→ Assessment of product releases against acceptance criteria of the Vision.

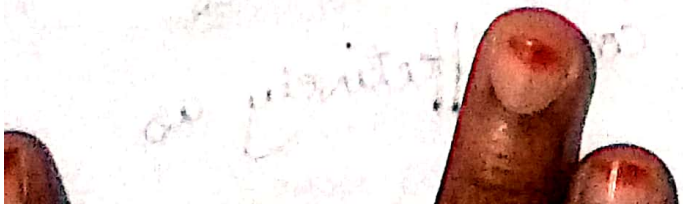
Primary Evaluation criteria :-

→ Is this product baseline mature enough to be deployed in the user community?

→ Is this product baseline stable enough to be deployed in the user community?

→ Are the stake holders ready for transition to the user community?

→ Are actual resource expenditures versus planned expenditures acceptable?



5) Transition phase :-

→ the transition phase is entered when a baseline is mature enough to be deployed in the end-user domain.

→ this phase could include any of the following activities :-

1) Beta testing to validate the new system against user expectations.
2) conversion of operational data bases.
3) Training of users and maintainers.

→ transition phase concludes when the deployment baselines has achieved the complete vision.

Primary objectives :-

→ Achieving user-self-supportability.
→ Achieving stake holder concurrence
→ Achieving final product baselines as rapidly and cost-effectively as Practical.

essential activities :-

→ Synchronization & integration of concurrent construction increments into consistent deployment baselines.

→ Deployment-specific engineering

→ Assessment of deployment baselines against the complete vision & acceptance criteria in the requirements set.

Primary Evaluation criteria :-

→ is the user satisfied?

→ Are actual resource expenditures versus planned expenditures acceptable?

6) model-based s/w architecture :-

→ An architecture is the s/w system design

→ the ultimate goal of the engineering stage is to converge on a stable architecture baseline.

→ An architecture baseline is not a paper document, it is a collection of information across all the Engineering sets.

→ architectures are described by extracting the essential information from the design models.

→ A view is a subset of a model that abstracts a specific, relevant perspective.

1. Architecture: A management perspective:

→ the most critical technical product of a s/w project is its architecture.

→ from a management perspective, there are three diff aspects of an

architecture.

i) An architecture is the design of a s/w system, as opposed to the design of a component.

ii) An architecture baseline is a slice of information across the engineering artifacts.

iii) An architecture description is an organized subset of information extracted from the design set model.

→ the importance of s/w architecture and its close linkage with modern s/w development process can be summarized as follows:-

* Achieving a stable s/w architecture.
* Architecture representations provide a basis for balancing the trade-offs b/w problem space & the solution space.
* Poor architectures and immature processes are often given as reasons for project failures.
* Architecture development & process definition are the intellectual steps.

2. Architecture : A Technical perspective :-

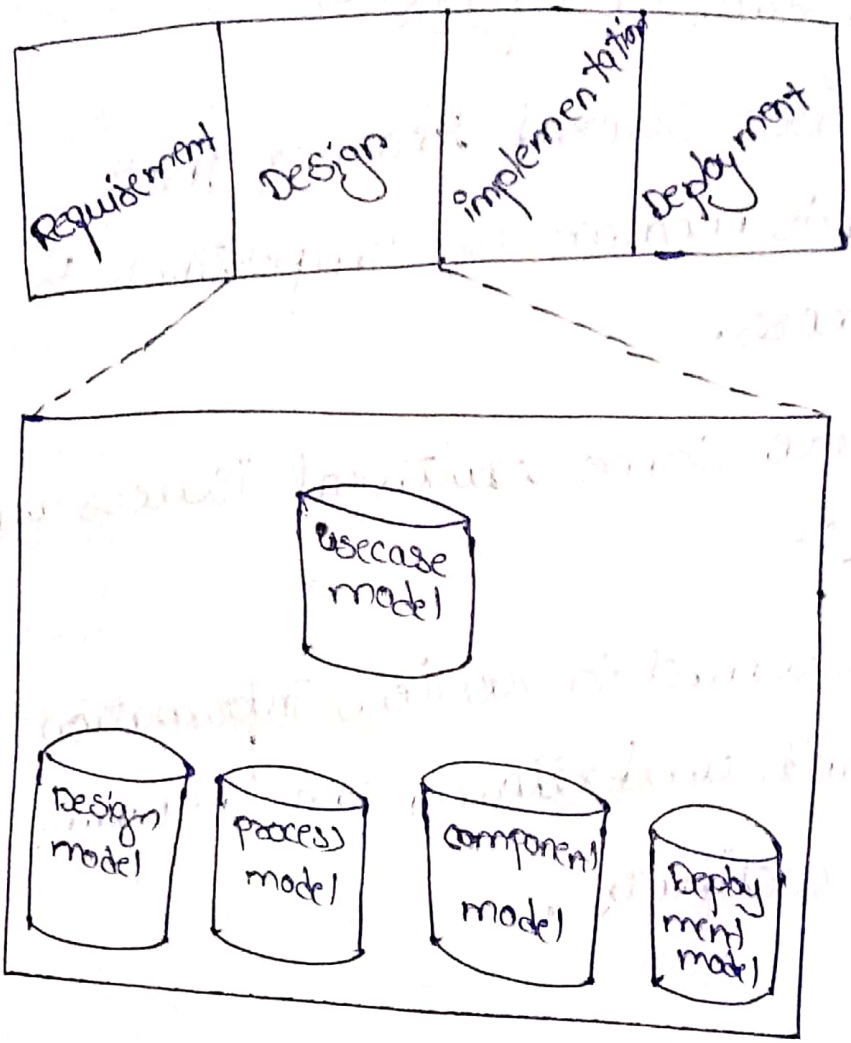
→ Although the architecture has been discussed at length over the past decade, convergence on definitions, terminology, and principles has been lacking.

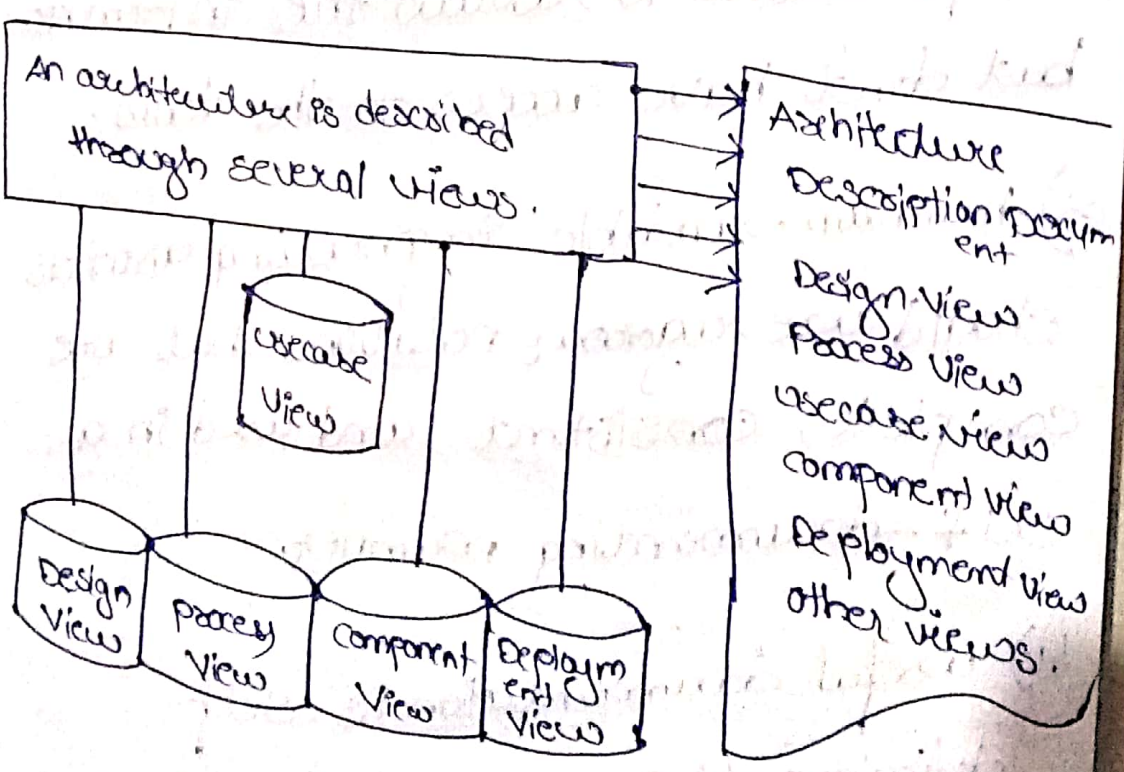
→ An architecture framework is defined in terms of views that are abstractions of the UML models in the design set.

→ most real-world systems requires four views they are as follows:-

i) Design
ii) process
iii) component
iv) Deployment.

→ this views is modeled statically using use case and class diagrams, & dynamically using sequence, collaboration, statechart and activity diagram.





7) Pragmatic artifacts :-

→ there are several reasons that documents became so important to the process.

→ there are some cultural issues are raises :-

1) People want to reviews information but don't understand the language of the artifacts.

2) People want to reviews the information but don't have access to the tools.

3) human-readable engineering artifacts should use rigorous notations that are complete, consistent, and used in a self-documenting manner.

4) useful documentation is 'self-defining'. It is documentation that gets used.

5) Paper is tangible; Electronic artifacts are too easy to change.

8) Engineering artifacts :-

→ most of the engineering artifacts are captured in rigorous engineering notations such as UML, programming languages, & executable machine codes.

→ however, three engineering artifacts are explicity intended for more general review, and they deserve further elaboration.

Vision Document :-

I. features set description
   A. precedence & priority
II. Quality attributes & ranges
III. Required constraints
   A. External interfaces
IV. Evolutionary appendixes
   A. Use cases
      1. primary scenarios
      2. Acceptance criteria
   B. Desired freedoms.

Typical Vision document outline.

Architecture Description :-

1. arch overviews
   i. objectives
   ii. constraints
   iii) freedoms
2. arch views
   i) Design view
   ii) process View
   iii) component "
   iv) Deployment "
3. arch interactions
4. Arch performance
5. Rationale, trade-offs & other substantiation.

Typical architecture description outline.

S/w user manual :-

→ the s/w user manual provides the user with the reference documentation necessary to support the delivered software.

9) Management artifacts :-

→ the management set includes several artifacts that capture intermediate results and ancillary information necessary to document the product legacy, maintain the product, improve the product and improve the process.

Business case :- the business case artifacts provides all the information necessary to determine whether the project is worth investing in.

→ the main purpose is to transform the vision into economic terms so that an organization can make an accurate ROI assessment.

S/w Development plan :-

→ the s/w development plan (SDP) elaborates the process framework into fully detailed Plan.

→ two indications of a useful SDP are Periodic updating & understanding and acceptance by managers.

Work Breakdown Structure :-

S/w change order Database :-

→ managing change is one of the fundamental primitives of an iterative development process.

→ with greater change freedom, a project can iterate more productively.

Release Specifications :-

→ Release description documents describes the results of each release, including performance against each of the Evaluation criteria in the corresponding release specification.

→ Release baselines should be accompained by a release description document that describes the Evaluation criteria.

Status Assessments :-

→ Status assessments provide periodic snapshots of project health and status, including the s/w project manager's risk

assessment, quality indicators, and management indicators.

Environment:-

→ An important emphasis of a modern approach is to define the development and maintenance environment as a first-class artifact of the process.

→ A robust, integrated development environment must support automation of the development process.

Deployment:-

→ A deployment document can take many forms. Depending on the project, it could include several document subsets for transitioning the product into operational status.

→ this environment should include requirments management, visual modeling, document automation, host & target programming tools, automated regression testing, and continuous & integrated change management & feature & defect tracking.

management Artifact sequences :-

→ In each phase of the life cycle, new artifacts are produced and previously developed artifacts are updated to incorporate lessons learned and to capture further depth and breadth of the solution.

10) Artifacts sets :-

→ to make the development of a complete s/w system manageable, distinct collections of information are organized into artifacts.

→ artifact represents cohesive information that typically is developed and reviewed as a single entity.

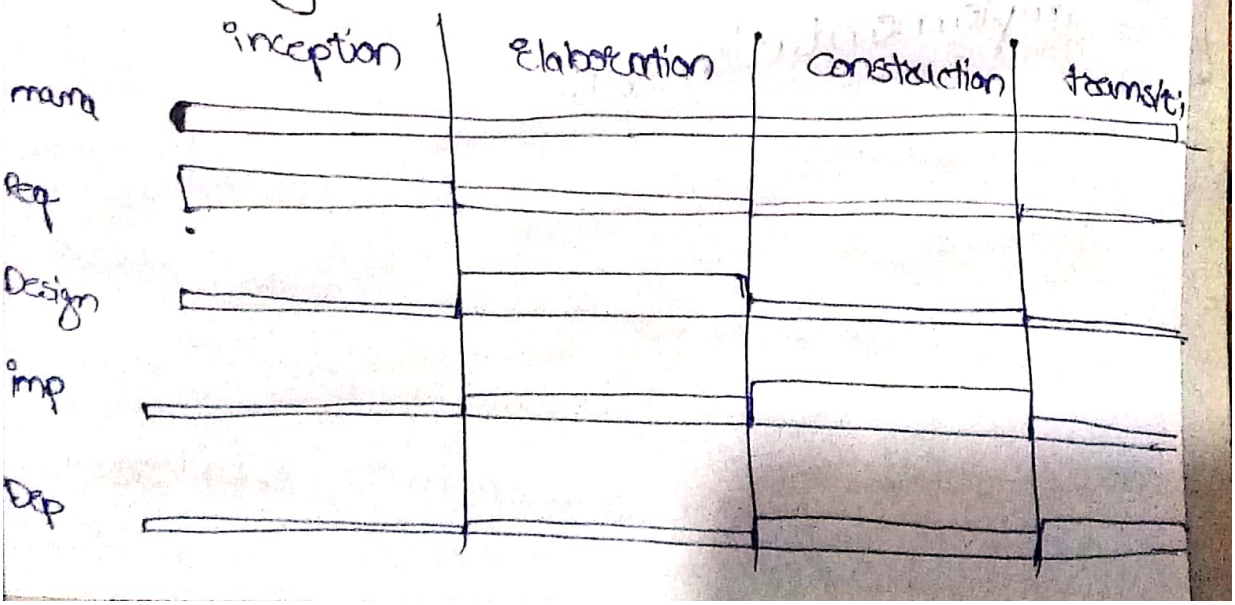
→ life-cycle s/w artifacts are organized into five distinct sets that are roughly partitioned by underlying language of the set:

1) management
2) requirements
3) design

4) implementation
5) deployment.

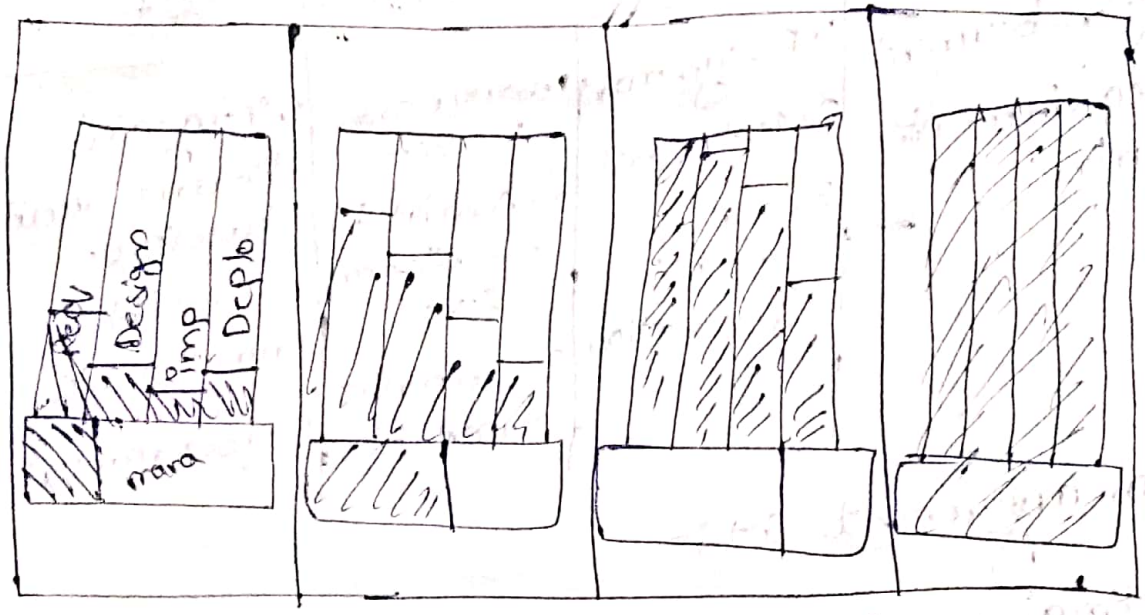
Requirement set	Design Set	Implement ation set	Deployment Set
1. Vision document 2. Requirements model	1. Design model 2. Test model 3. S/w archit description	1. source code baselines 2. Associated compile-time files 3. component executables	1. integrated product executable baselines. 2. Associated run-time files 3. User manual

1. management set:-
2. engineering set
   i) Requirements set
   ii) Design set
   iii) implementation set
   iv) Deployment set.



3. artifact Evaluation over the lifecycle :-

Engineering stage		Production stage	
inception	Elaboration	construction	Transition



4. test artifacts :-

* management set
* Requirements set
* Design set
* implementation set.

Part-1 UNIT-2

1. Conventional s/w management:-

→ the best thing about s/w is its flexibility. It can be programmed to do almost anything.

→ conventional s/w management practices are mostly old sound in theory, but practice is still tied to archaic technology and techniques.

→ conventional s/w economics provides a benchmark of performance for conventional s/w management principles.

→ they can be summarized as follows:-

1. s/w development is still highly unpredictable.
2. management discipline is more of a discriminator in success & failure than are technology advances.
3. the level of s/w scrap & rework is indicative of an immature process.

→ the three analyses provide a good introduction to the magnitude of s/w problem and the current norms for conventional s/w management performance.

→ there is much room for improvement.

I the waterfall model :-

→ most s/w engineering texts present the waterfall model as the source of the "conventional" s/w process.

→ this section examines and critiques the waterfall model theory, then looks at how most of the industry has practiced the conventional s/w process.

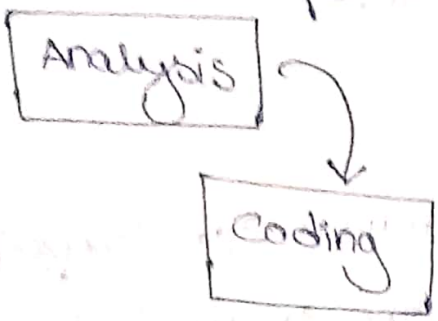
In theory :-

→ the paper made three primary points :-

1. there are 2 essential steps common to the development of computer programs :
   → analysis & coding.
2. In order to manage and control all of the intellectual freedom associated with s/w development.
3. the basic framework described in the waterfall model is risky and invites failure.

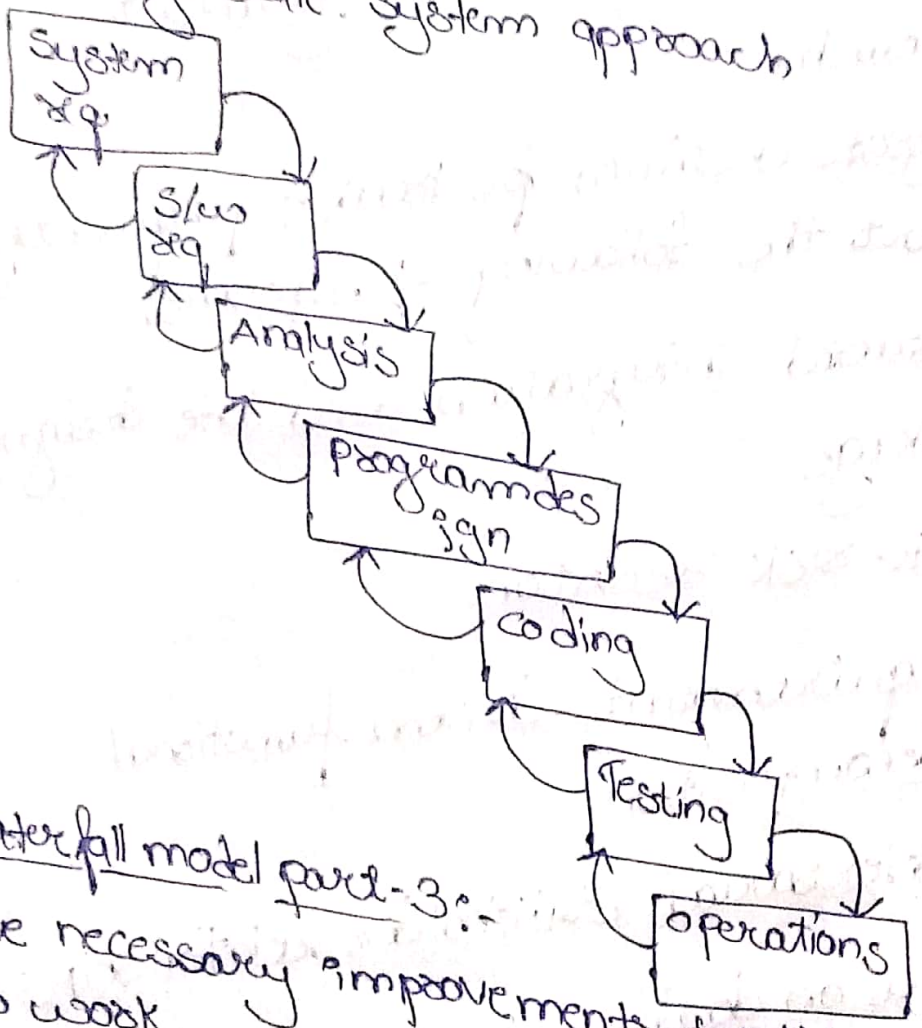
waterfall model part-1:-

the two basic steps to building a program



waterfall model part-2:-

the large scale system approach



waterfall model part-3:-

five necessary improvements for this approach to work.

1. complete program design before analysis & coding begin
2. maintain current & complete documentation
3. Do the job twice, if possible

4. plan, control and monitor testing.

5. involve the customer.

In practice :-

→ Despite the advice of many s/w experts and the theory behind the waterfall model, some s/w projects still practice the conventional s/w management approach.

→ Projects destined for trouble frequently exhibit the following symptoms:

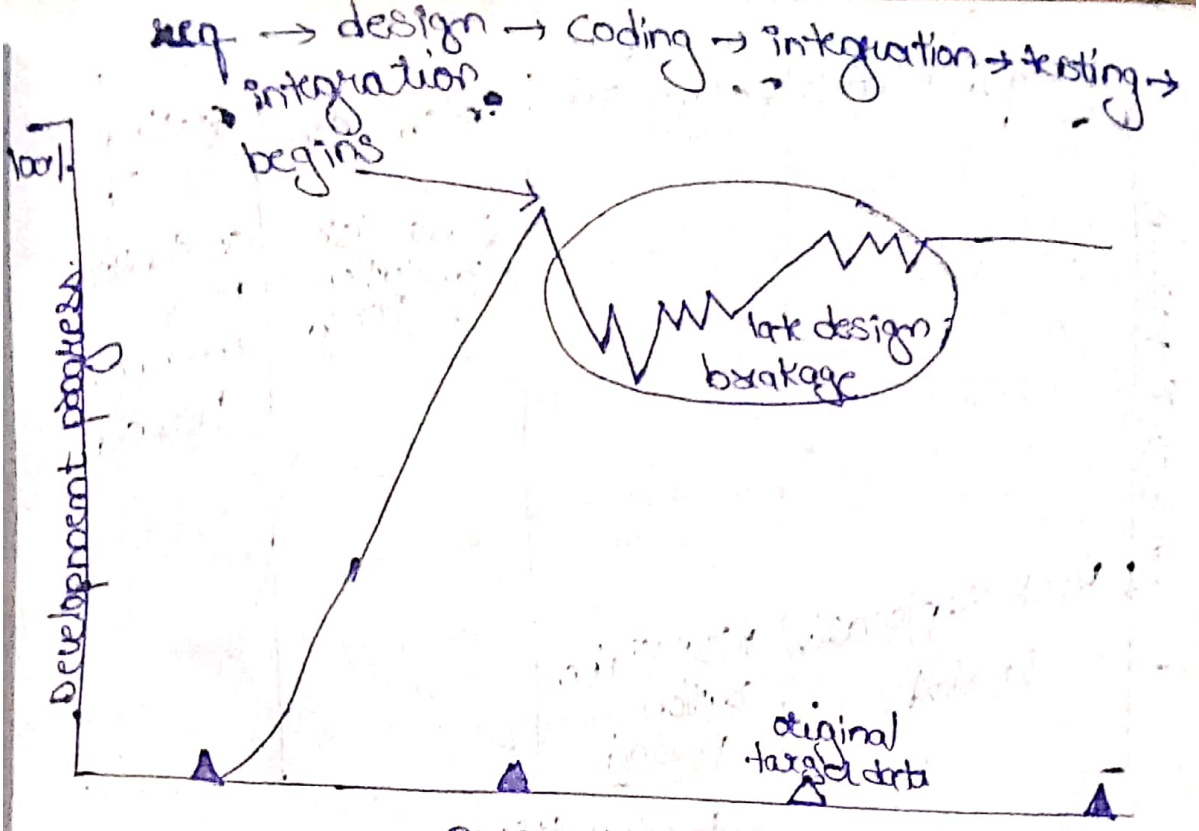
i) protracted integration and late design breakage

ii) Late-risk resolution

iii) Requirements-driven functional decomposition

iv) Adversarial stakeholder relationships

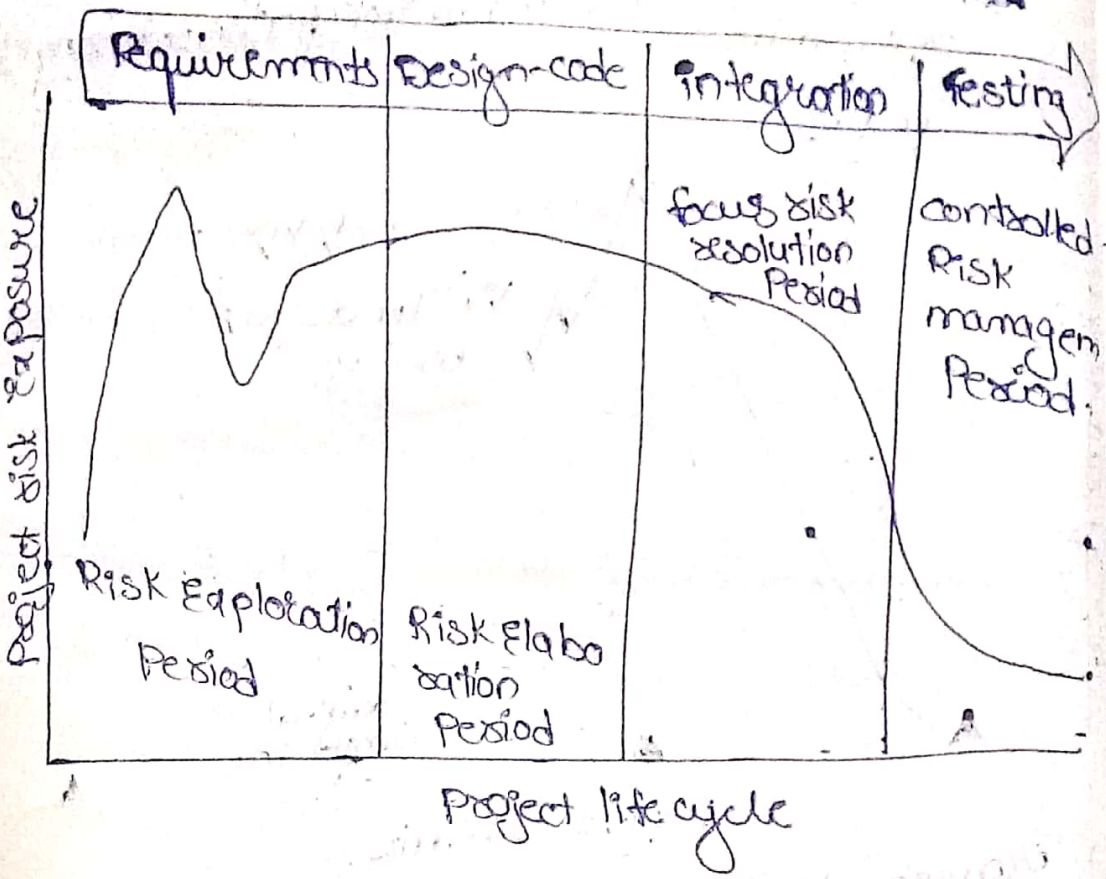
v) focus on documents & review meetings.

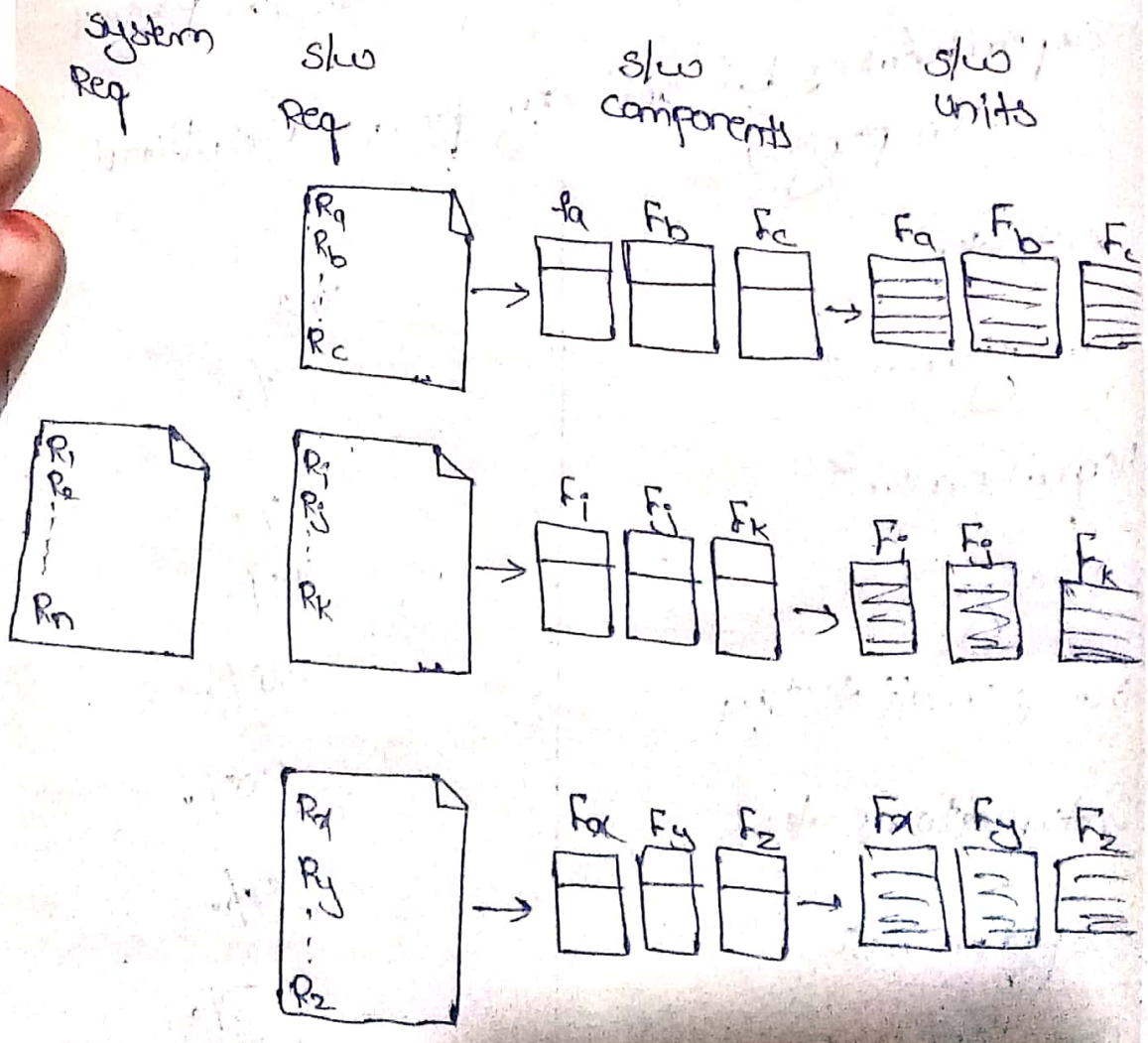


→ progress profile of a conventional s/w project.

→ Expenditures by activity for a conventional s/w project.

Activity	Cost
management	5%.
Requirement	5%.
Design	10%.
Code & unit testing	30%.
Integration & test	40%.
Deployment	5%.
Environment	5%.
Total	100%.





II. Conventional s/w management performance:-

→ the following paragraphs, quotations from Boehm's top 10 list are:-

1) finding and fixing the s/w problem after delivery costs 100 times more than finding & fixing the problem in early design phase.

2) you can compress s/w development schedules 25% of nominal, but no more.

3) for every $1 you spend on development, you will spend $2 on maintenance.

4) s/w development & maintenance costs are primarily a function of the number of source lines of code.

5) Variations among people account for the biggest diff in s/w productivity.

6) the overall ratio of s/w to h/w costs is still growing

7) only about 15% of s/w development effort is devoted to programming.

8) SLOC systems and products typically cost 3 times as much per SLOC as individual SLOC programs. SLOC system products cost 9 times as much.

9) walkthroughs catch 60% of the errors.

10) 80% of the contribution comes from 20% of the contributors.