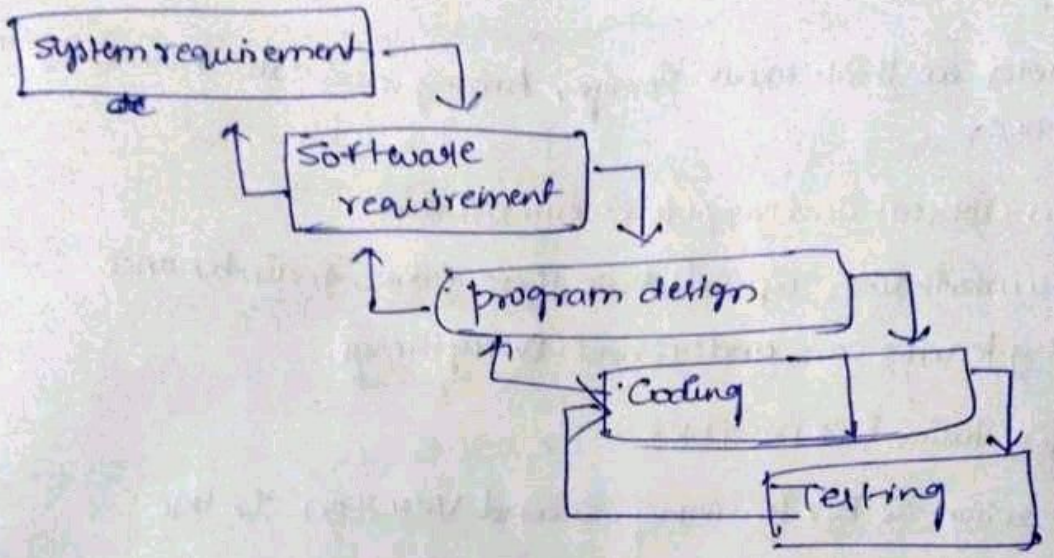
Sppm:- (6)2 (7) 2 (8) 2 (9) 3 (10) 2 (11) 2 5

(6) sketch waterfall model. What are five necessary improvements in waterfall model?

A:- Waterfall model:-

The waterfall model. This is one of the oldest and simplest software development approaches.



1. System requirement:-

* understand what the overall system should do.
* This is from the customer's and business's point of view.

2. Software requirement:-

* Translate system needs into clear, detailed software requirements
* This includes functions, features, and constraints.

3. program design:-

* plan the structure of the software
* Decide how different modules will work together.

4. Coding:-

* write the actual source code based on the design.

5. Testing:-

* Check if the developed software works correctly
* Find and fix errors before delivery.

⇒ Five necessary improvements:-

1. Complete program design before coding begins:

* Start with a clear design before jumping into analysis or coding.
* This ensures we think about storage, timing, and data handling in advance.

2. Maintain current and complete documentation:

* Keep documentation updated so that future maintenance and testing teams can understand everything.

3. Do the job twice (if possible).

* First version is for learning, second version is the real one.
* This "do it twice" idea is the origin of today's iterative development

4. plan, control, and monitor testing carefully.

* use a separate testing team.
* Inspect logic, check for small errors, and test on the target system.

5. Involve the customer Early :-

* Include customer reviews at key points
  1. preliminary review during design
  2. Critical review before coding
  3. final acceptance review after testing.

(7). Discuss in brief about the software economics and the necessity for software cost estimation?

Ans: Software Economics:-

↳ When we say software Economics, we are really talking about money, time, and resources in software development.

↳ It's like running a business, but here the product is software

↳ The goal is to build good-quality software while keeping costs low, time short, and customer satisfaction high.

↳ Size matters:-

* Bigger projects need more people, more time, and more money.
* If we can reduce the amount of code, we save cost and time.

↳ Cost is not just coding:-

* Coding is only about 15% of the total cost.
* The rest is for planning, designing, testing, bug fixing, and maintenance.

↳ Ways to save money:-

* Reuse existing software components
* Use ready-made commercial tools
* Choose better programming language or methods like object oriented design.

↳ Better process = better economics:-

* A good development process avoids rework and reduce costs.

↳ People are the biggest asset

* Skilled and experienced people can do the job faster and better.

*. Necessity for software cost estimation:-

Software cost Estimation means predicting how much money, time, and effort will be needed to build a software project.

1. Budget planning:-

* without a cost estimate, we don't know how much money to ask for or allocate.

2. Time scheduling:-

* It helps us decide how long the project will take and when each part will be ready.

3. Resource allocation:-

* we can plan how many people, tools, and machines we need.

4. Customer confidence:-

* when we give customers a clear cost and time estimate, they can trust us more.

5. Risk reduction:-

* If we know costs early, we can spot if the project might go over budget and take action.

6. Better decision-making:-

* Helps choose between different approaches

Ex:- whether to build from scratch or reuse components.

(11g) Life Cycle Evolution of the Artifact Sets.

Any ~ Artifacts = All the important project stuff like plans, designs, Code and tests.

They are grouped into Sets to Make work easy.

* In the Software life Cycle, these artifacts are arranged into five main sets :-

1) Management Set :-

* This set Contains things related to planning and running the project.
* It has notes, schedules, and documents used by Managers to track work.

Ex :- A weekly project progress report.

2) Engineering set :-

* This set is about technical work
* It's where we Move ideas and designs from One Stage to another.

Ex :- Better version of a design.

3) Requirement Set :-

* This set is about what the Software should do.
* It is used to check and evaluate other sets.

Ex :- "App must have Secure payment".

4) Design set :-

* This is the detailed design of the Software.
* Uses UML diagrams and has Many levels in detail.
* Includes all structural and behavioral details.

5) Implementation set :-

* This set Contains actual Coding work.
* Also includes things like debuggers, Compilers and Code analysis tools.

Ex :- Login Code.

6) Development Set :-

* Used for n/w Management tools, test Coverage & test

automation tools etc.

Ex :- Automated testing tool.

Life Cycle phases

These artifacts sets are used during the two main project phases :-

A) Engineering phase :-

- Goal :- Plan and prepare the project.
- Done by a small team and is less predictable.
- Has two steps :-
  - a) Inception phase :- Decide the goals & gather requirements.
  - b) Elaboration phase :- Study in detail, Make architecture, Check risks.

B) Production phase :-

- Goal :- Build and deliver software
- Has two steps :-
  - a) Construction phase :- Code and integrate features while minimizing risk.
  - b) Transition phase :- Do testing, fix issues, and Make the system user-friendly before final release.

(9.) Discuss about work flows for the process?

Sol:-

Workflows:-

↳ The workflow of the process describes how different activities in software development are arranged and connected so that project run smoothly

↳ It shows the order of steps, how teams coordinate, and how information moves between tasks.

Main workflows in software Development:-

1. Requirement workflow:
- Collect and understand what the customer wants
- Create requirement documents and get them approved.

2. Analysis workflow:
- Study the requirements deeply to identify what the system must do.
- Create models and diagrams to represent the system.

3. Design workflow:
- plan how the system will be built.
- Decide architecture, modules, and data structures.

4. Implementation workflow:
- Write the code based on the design
- Integrate different modules

5. Test workflow:
- Test the software to find and fix defects
- Ensure it meets requirements.

6. Deployment Workflow:-

- Deliver the product to the customer
- provide installation and training if needed.

7. Maintenance Workflow:-

- Keep improving and fixing the software after delivery.

Briefly explain pragmatic software cost estimation.

Pragmatic software cost estimation is a practical, experience-based approach to predicting the effort, time, and cost required to develop a software system.

Unlike purely theoretical or model-driven methods, pragmatic estimation relies on:

- Expert judgment – drawing on the experience of developers and project managers.
- Historical data – using information from similar past projects.
- Simple models or rules of thumb – such as cost per function point, lines of code, or story points.

The goal is not perfect accuracy but useful, realistic estimates that support planning, budgeting, and decision-making, while being adaptable as new information becomes available.