CSE 411
Software Engineering and System Analysis and Design

Topic 2: Software Development Model

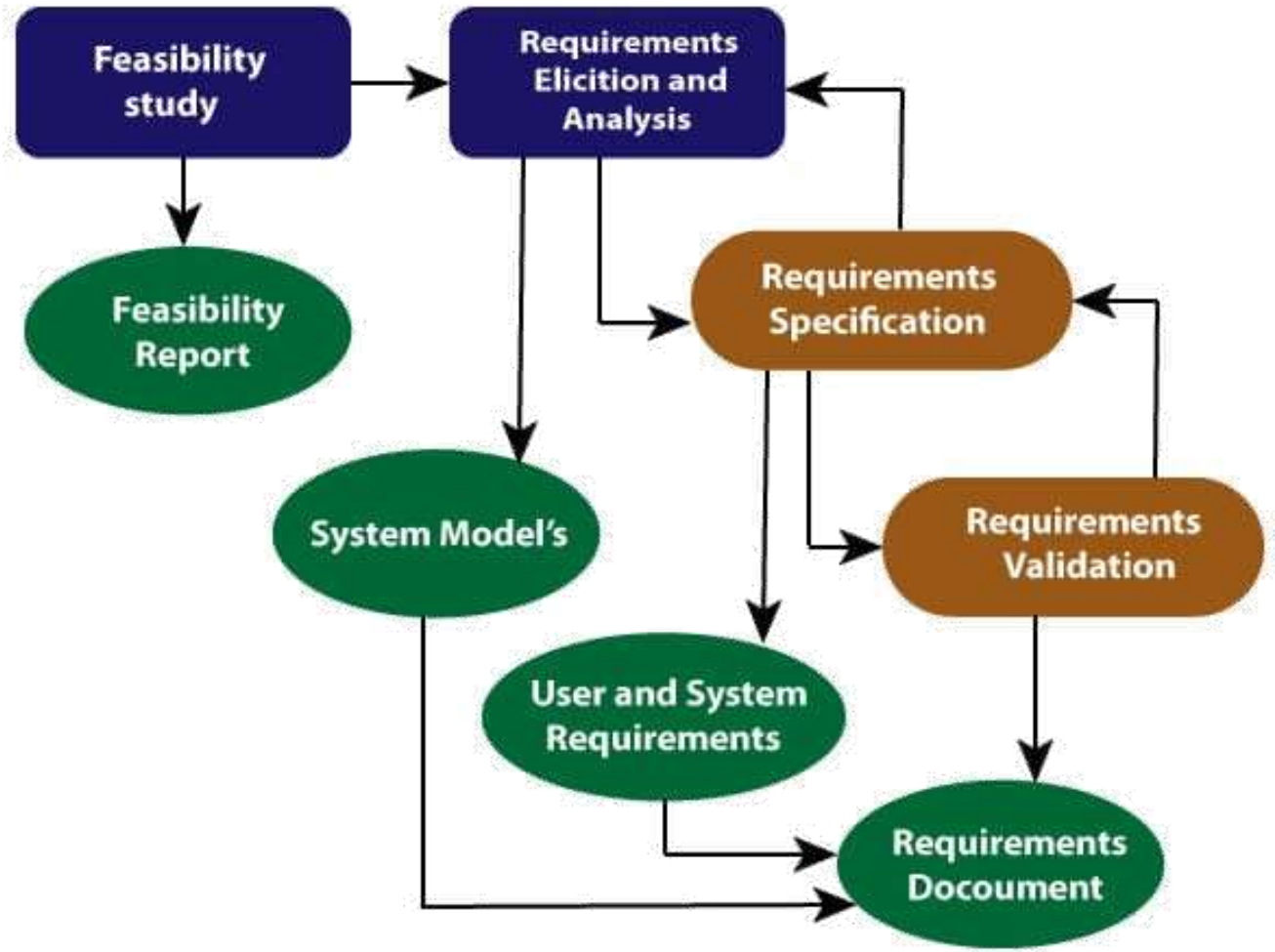
Requirements engineering (RE)

Requirements engineering (RE) refers to the process of defining, documenting, and maintaining requirements in the engineering design process.

Requirement engineering provides the appropriate mechanism to understand what the customer desires, analyzing the need, and assessing feasibility, negotiating a reasonable solution, specifying the solution clearly, validating the specifications and managing the requirements as they are transformed into a working system.

It is a four-step process -

- Feasibility Study
- Requirement Elicitation and Analysis
- Software Requirement Specification
- Software Requirement Validation
- Software Requirement Management



Requirement Engineering Process

1. Feasibility Study

The objective behind the feasibility study is to create the reasons for developing the software that is acceptable to users, flexible to change and conformable to established standards.

Types of Feasibility:

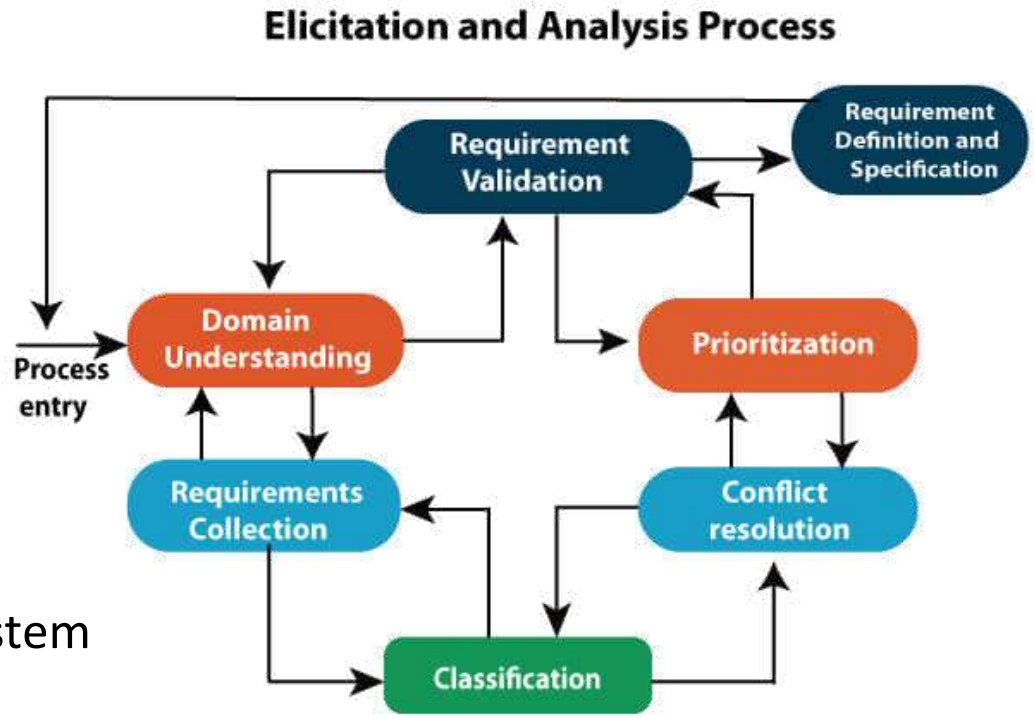
- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

2. Requirement Elicitation and Analysis

Gathering and analysis of requirements starts with requirement elicitation. The requirements are analyzed to identify inconsistencies, defects, omission, etc.

Problems of Elicitation and Analysis

- Getting all, and only, the right people involved.
- Stakeholders often don't know what they want
- Stakeholders express requirements in their terms.
- Stakeholders may have conflicting requirements.
- Requirement change during the analysis process.
- Organizational and political factors may influence system requirements.



3. Software Requirement Specification

Software requirement specification (SRS) is a kind of document which is created by a software analyst after the requirements collected from the various sources - the requirement received by the customer written in ordinary language. It is the job of the analyst to write the requirement in technical language so that they can be understood and beneficial by the development team.

3. Software Requirement Specification

The models used at this stage include ER diagrams, data flow diagrams (DFDs), function decomposition diagrams (FDDs), data dictionaries, etc.

Data Flow Diagrams: Data Flow Diagrams (DFDs) are used widely for modeling the requirements. DFD shows the flow of data through a system.

Data Dictionaries: Data Dictionaries are simply repositories to store information about all data items defined in DFDs.

Entity-Relationship Diagrams: Another tool for requirement specification is the entity-relationship diagram, often called an "***E-R diagram***." It is a detailed logical representation of the data for the organization and uses three main constructs i.e. data entities, relationships, and their associated attributes.

4. Software Requirement Validation:

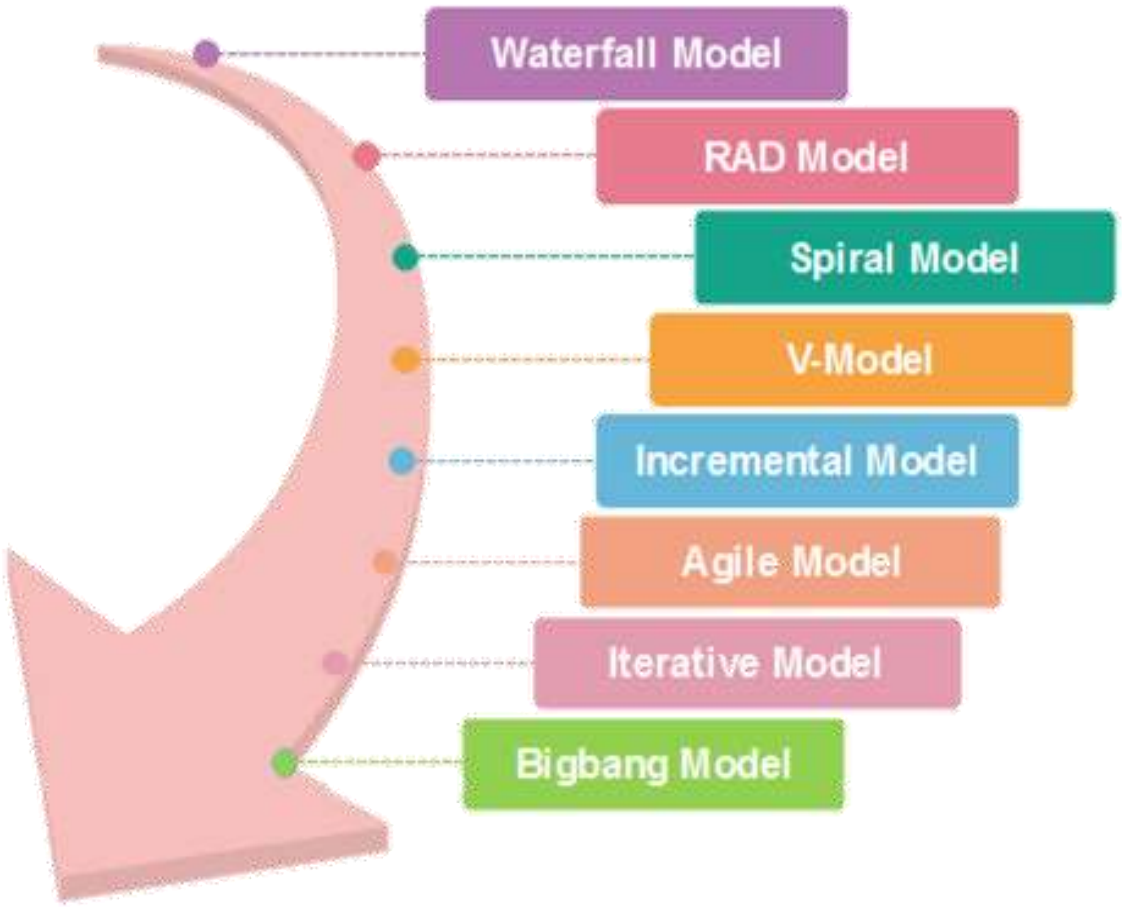
- After requirement specifications developed, the requirements discussed in this document are validated.
- The user might demand illegal, impossible solution or experts may misinterpret the needs.
- Requirements can be the check against the following conditions -
  - If they can practically implement
  - If they are correct and as per the functionality and specially of software
  - If there are any ambiguities
  - If they are full
  - If they can describe

5. Software Requirement Management:

Requirement management is the process of managing changing requirements during the requirements engineering process and system development.

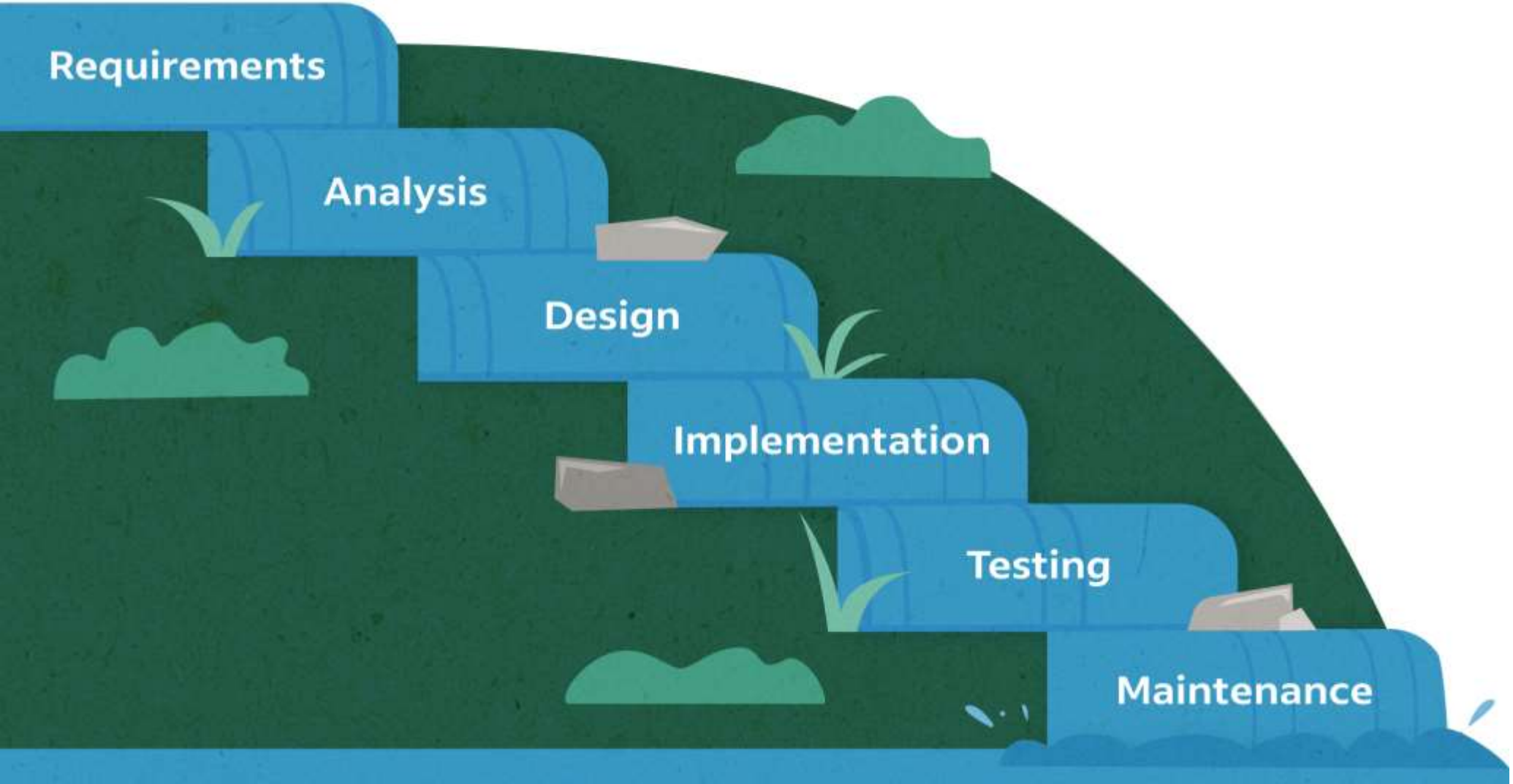
SDLC (Models)

There are different software development life cycle models specify and design, which are followed during the software development phase. These models are also called "Software Development Process Models."



Waterfall Model

Winston Royce introduced the Waterfall Model in 1970. The developer must complete every phase before the next phase begins.



1. Requirements analysis and specification phase:

- The aim of this phase is to understand the exact requirements of the customer and to document them properly.

- Both the customer and the software developer work together so as to document all the functions, performance, and interfacing requirement of the software.

- In this phase, a large document called Software Requirement Specification (SRS) document is created which contained a detailed description of what the system will do in the common language.

2. Design Phase:

- This phase aims to transform the requirements gathered in the SRS into a suitable form which permits further coding in a programming language.
- It defines the overall software architecture together with high level and detailed design.
- All this work is documented as a Software Design Document (SDD).

3. Implementation and unit testing:

- If the SDD is complete, the implementation or coding phase proceeds smoothly, because all the information needed by software developers is contained in the SDD.

- During testing, the code is thoroughly examined and modified. Small modules are tested in isolation initially. After that these modules are tested by writing some overhead code to check the interaction between these modules and the flow of intermediate output.

4. Integration and System Testing:

- This phase is highly crucial as the quality of the end product is determined by the effectiveness of the testing carried out. The better output will lead to satisfied customers, lower maintenance costs, and accurate results.

- Unit testing determines the efficiency of individual modules. However, in this phase, the modules are tested for their interactions with each other and with the system.

Waterfall model is most suited when:

- The requirements are constant and not changed regularly.
- A project is short.
- Where the tools and technology used is consistent and is not changing.
- When resources are well prepared and are available to use.

Disadvantages of Waterfall model:

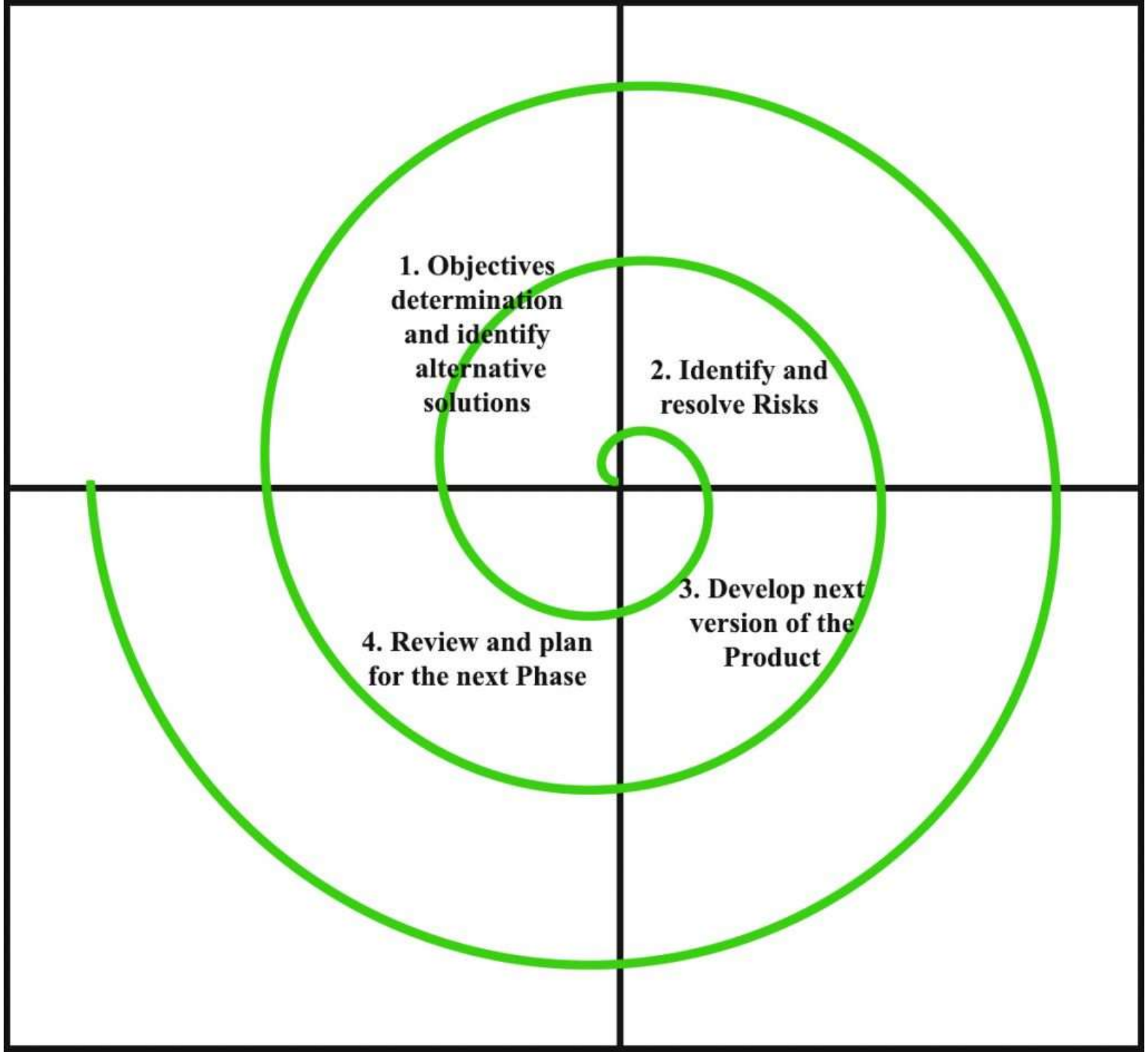
- In this model, the risk factor is higher, so this model is not suitable for more significant and complex projects.
- This model cannot accept the changes in requirements during development.
- It becomes tough to go back to the phase. For example, if the application has now shifted to the coding phase, and there is a change in requirement, It becomes tough to go back and change it.
- Since the testing done at a later stage, it does not allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare.

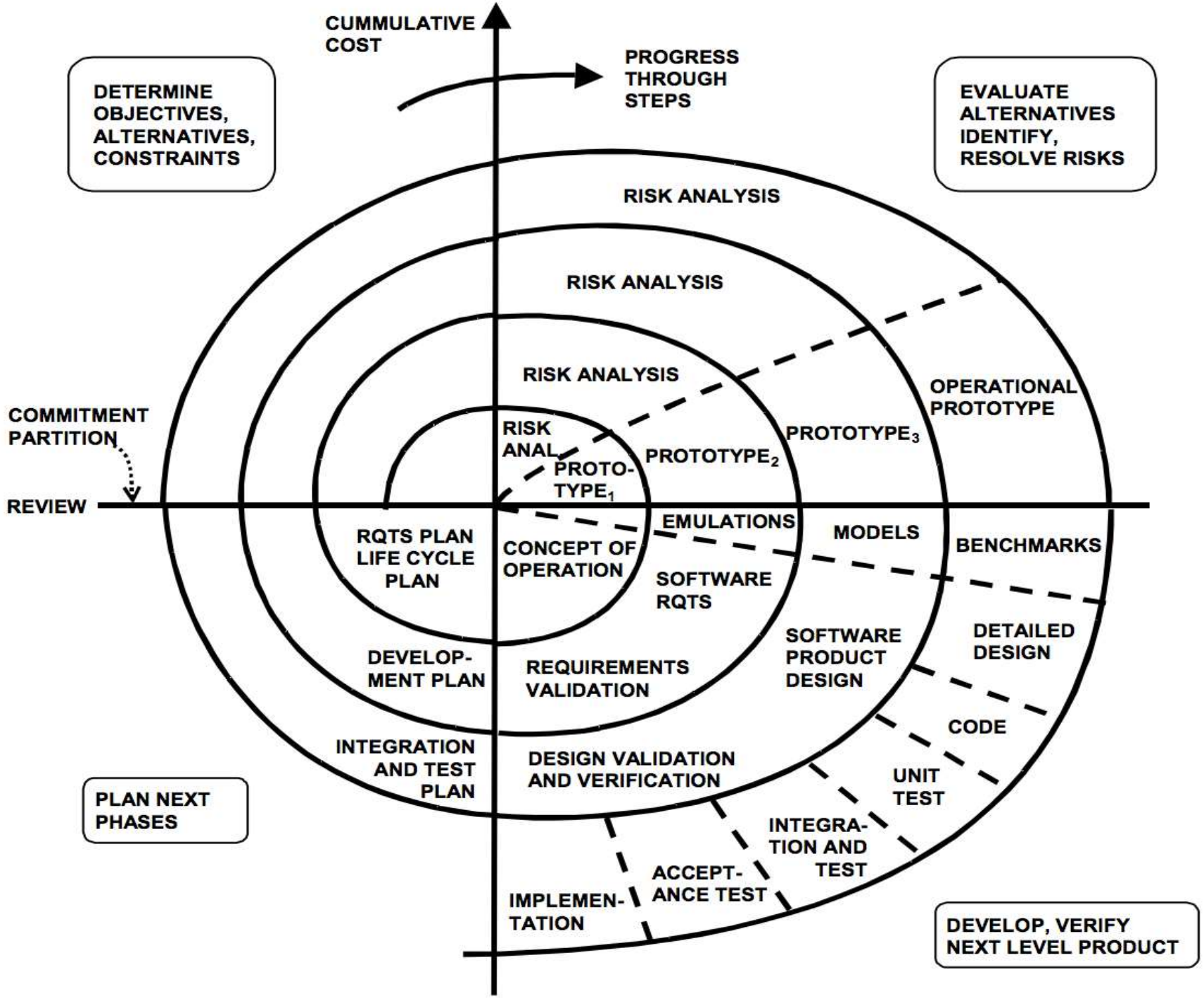
Disadvantages of Waterfall model:

- In this model, the risk factor is higher, so this model is not suitable for more significant and complex projects.
- This model cannot accept the changes in requirements during development.
- It becomes tough to go back to the phase. For example, if the application has now shifted to the coding phase, and there is a change in requirement, It becomes tough to go back and change it.
- Since the testing done at a later stage, it does not allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare.

Spiral Model

- The spiral model, initially proposed by Boehm, is an evolutionary software process model.
- Using the spiral model, the software is developed in a series of incremental releases.
- During the early iterations, the additional release may be a paper model or prototype.
- During later iterations, more and more complete versions of the engineered system are produced.


1. Objectives determination and identify alternative solutions
2. Identify and resolve Risks
3. Develop next version of the Product
4. Review and plan for the next Phase


CUMMULATIVE COST
PROGRESS THROUGH STEPS
DETERMINE OBJECTIVES, ALTERNATIVES, CONSTRAINTS
EVALUATE ALTERNATIVES IDENTIFY, RESOLVE RISKS
RISK ANALYSIS
RISK ANALYSIS
RISK ANALYSIS
COMMITMENT PARTITION
REVIEW
RISK ANAL
PROTO-TYPE1
PROTOTYPE2
PROTOTYPE3
OPERATIONAL PROTOTYPE
EMULATIONS
MODELS
BENCHMARKS
RQTS PLAN LIFE CYCLE PLAN
CONCEPT OF OPERATION
SOFTWARE RQTS
DETAILED DESIGN
SOFTWARE PRODUCT DESIGN
DEVELOP-MENT PLAN
REQUIREMENTS VALIDATION
CODE
INTEGRATION AND TEST PLAN
DESIGN VALIDATION AND VERIFICATION
UNIT TEST
INTEGRA-TION AND TEST
ACCEPT-ANCE TEST
IMPLEMEN-TATION
PLAN NEXT PHASES
DEVELOP, VERIFY NEXT LEVEL PRODUCT

Each cycle in the spiral is divided into four parts:

- **Objective setting:** Each cycle in the spiral starts with the identification of purpose for that cycle, the various alternatives that are possible for achieving the targets, and the constraints that exists.

- **Risk Assessment and reduction:** The next phase in the cycle is to calculate these various alternatives based on the goals and constraints. The focus of evaluation in this stage is located on the risk perception for the project.

- **Development and validation:** The next phase is to develop strategies that resolve uncertainties and risks. This process may include activities such as benchmarking, simulation, and prototyping.

- **Planning:** Finally, the next step is planned. The project is reviewed, and a choice made whether to continue with a further period of the spiral. If it is determined to keep, plans are drawn up for the next step of the project.

When to use Spiral Model?

- When deliverance is required to be frequent.
- When the project is large
- When requirements are unclear and complex
- When changes may require at any time
- Large and high budget projects

Advantages

- High amount of risk analysis
- Useful for large and mission-critical projects.

Disadvantages

- Can be a costly model to use.
- Risk analysis needed highly particular expertise
- Doesn't work well for smaller projects.

Agile Model

Phases of Agile Model:

- Requirements gathering
- Design the requirements
- Construction/ iteration
- Testing/ Quality assurance
- Deployment
- Feedback

Fig. Agile Model

- Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks.

- The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements.

- Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.

- The meaning of Agile is swift or versatile. **"Agile process model"** refers to a software development approach based on iterative development.

- Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning.

- The project scope and requirements are laid down at the beginning of the development process.

- Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.

1. Requirements gathering: In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.

2. Design the requirements: When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.

3. Construction/ iteration: When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.

4. Testing: In this phase, the Quality Assurance team examines the product's performance and looks for the bug.

5. Deployment: In this phase, the team issues a product for the user's work environment.

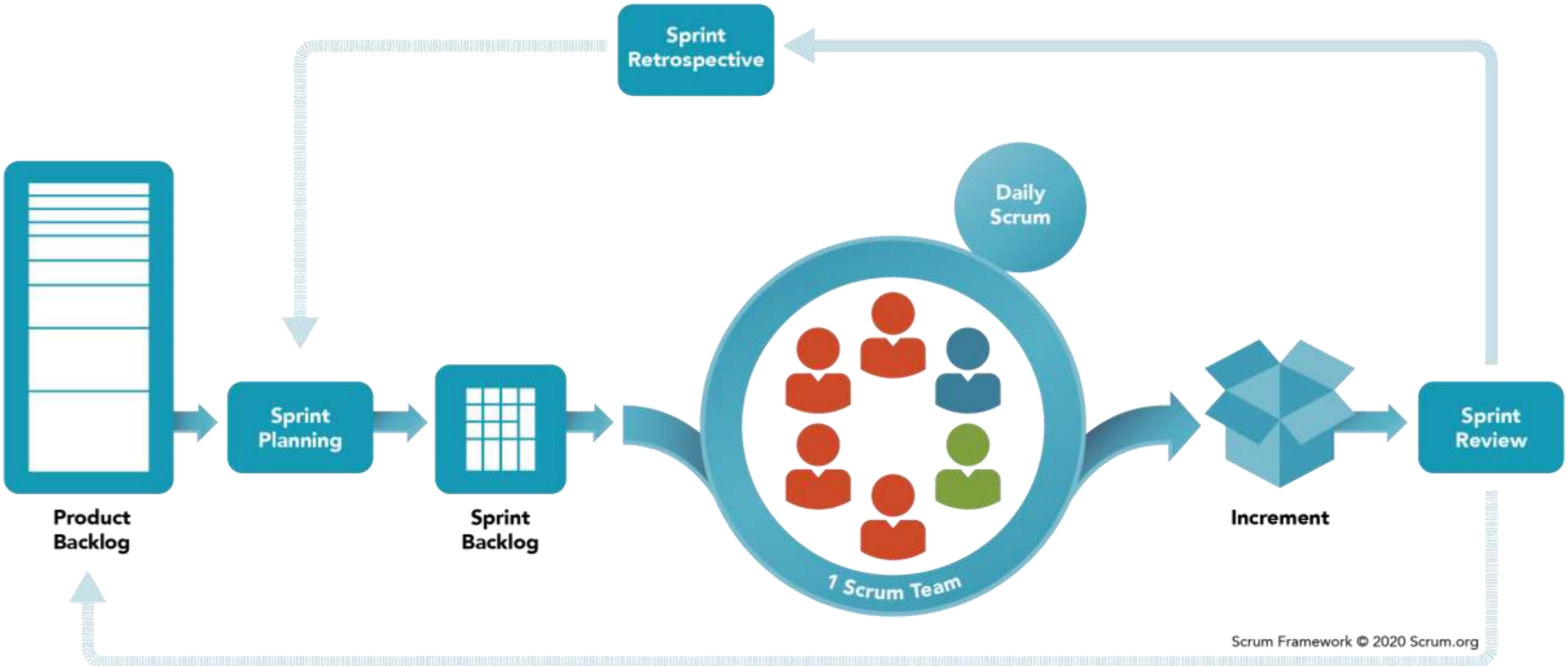
6. Feedback: After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

Agile Testing Methods:

- Scrum
- Crystal
- Dynamic Software Development Method(DSDM)
- Feature Driven Development(FDD)
- Lean Software Development
- eXtreme Programming(XP)

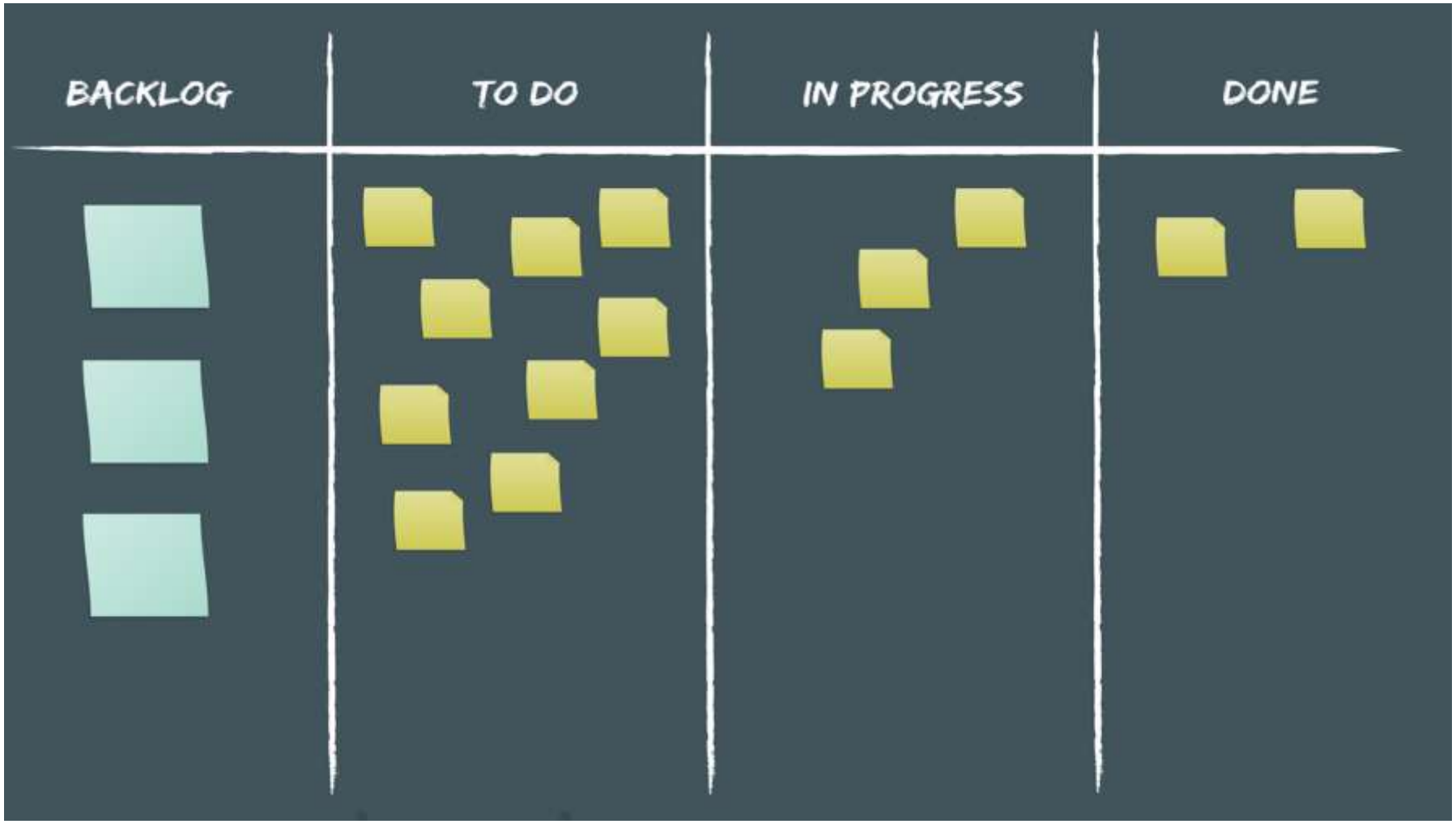
https://adevait.com/blog/agile-work/agile-testing-a-new-era-for-agile-teams

SCRUM is an agile development process focused primarily on ways to manage tasks in team-based development conditions.


Sprint Retrospective
Daily Scrum
Sprint Planning
Sprint Review
Product Backlog
Sprint Backlog
1 Scrum Team
Increment
Scrum Framework © 2020 Scrum.org

Daily Scrum

Scrum Board



There are three roles in it, and their responsibilities are:

- **Scrum Master:** The scrum can set up the master team, arrange the meeting and remove obstacles for the process
- **Product owner:** The product owner makes the product backlog, prioritizes the delay and is responsible for the distribution of functionality on each repetition.
- **Scrum Team:** The team manages its work and organizes the work to complete the sprint or cycle.

End of Topic 2