# Unit-2
# Process Model

2.1 The waterfall model

2.2 Prototyping Model

2.3 RAD Model

2.4 Spiral Model

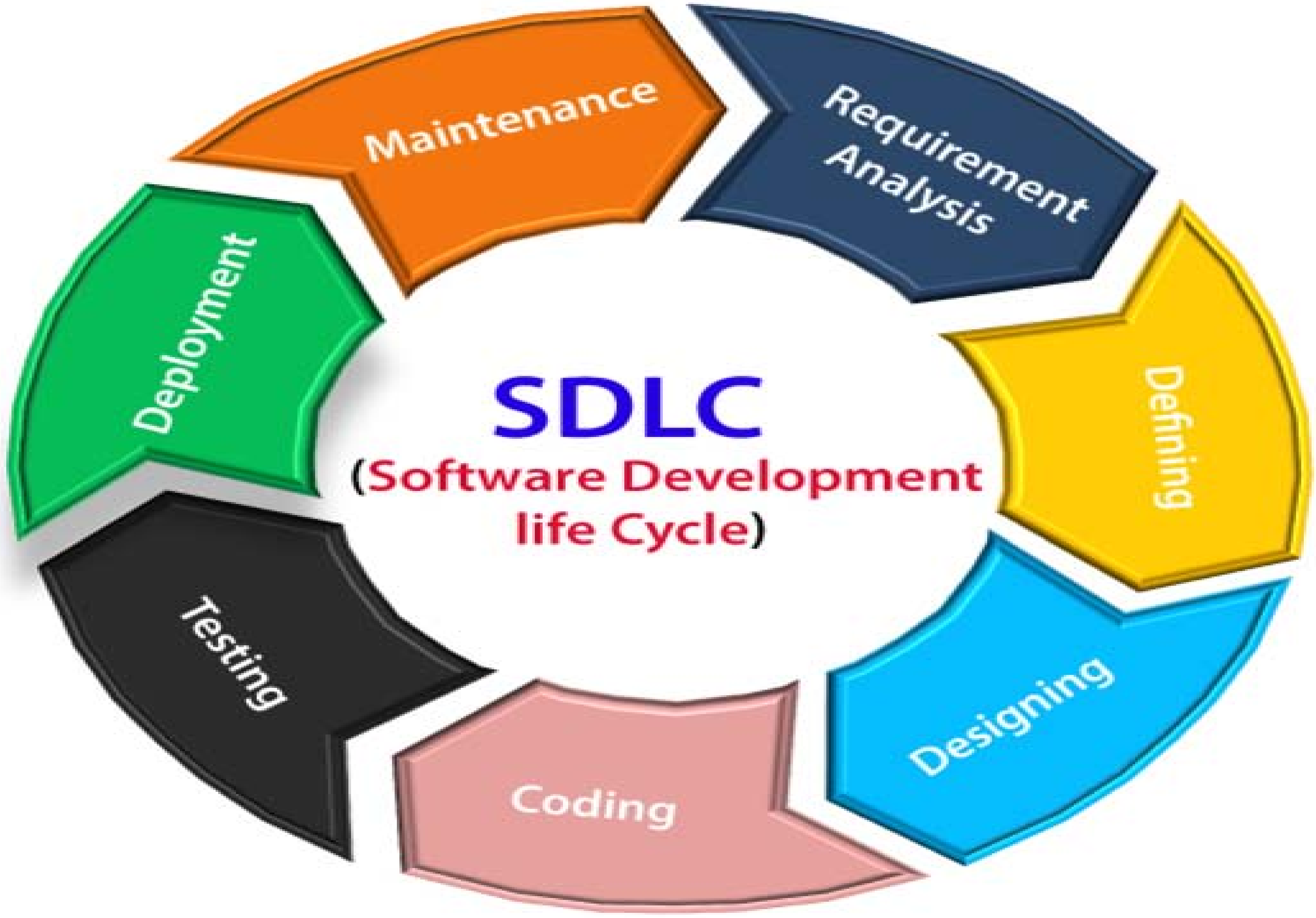
2.5 Agile Software Model

## Process Model

- In software engineering, a **process model** refers to a structured framework used to plan, develop, and manage the process of software development.
- It provides a systematic approach to organize and control the various stages of the software development life cycle (SDLC).

## Software Development Life Cycle (SDLC)

- A software life cycle model ( also termed process model) is a pictorial and diagrammatic representation of the software life cycle.
- SDLC is a process used by the software industry to design, develop and test high quality software.
- The SDLC aims to produce a high-quality software that meets or exceeds customers expectations, reaches completion within times and cost estimates.
- It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software.
- The life cycle defines a methodology for improving the quality of software and the overall development process.


SDLC
(Software Development life Cycle)
Requirement Analysis
Defining
Designing
Coding
Testing
Deployment
Maintenance

## The stages of SDLC are as follows

### Stage1: Planning and requirement analysis

- Requirement analysis is the most important and necessary stage in SDLC.
- The senior members of the team perform it with inputs from all the stakeholders and domain experts or SMEs(Subject Matter Experts) in the industry.
- Planning for the quality assurance requirements and identifications of the risks associated with the projects is also done at this stage.

### Stage2: Defining Requirements

- Once the requirement analysis is done, the next stage is to certainly represent and document the software requirements and get them accepted from the project stakeholders.
- This is accomplished through SRS( Software Requirement Specification) document which contains all the product requirement to be constructed and developed during the project life cycle.

**Stage3: Designing the Software**

- The next phase is about to bring down all the knowledge of requirements, analysis, and design of the software project.
- This phase is the product of the last two, like input from the customer and requirement gathering.

**Stage4: Developing the project**

- In this phase of SDLC, the actual development begins, and the programming is built. The implementation of design begins concerning writing code.
- Developers have to follow this coding guidelines described by their management and programming tools like compilers, interpreters, debuggers, etc. are used to develop and implement the code.

**Stage5: Testing**

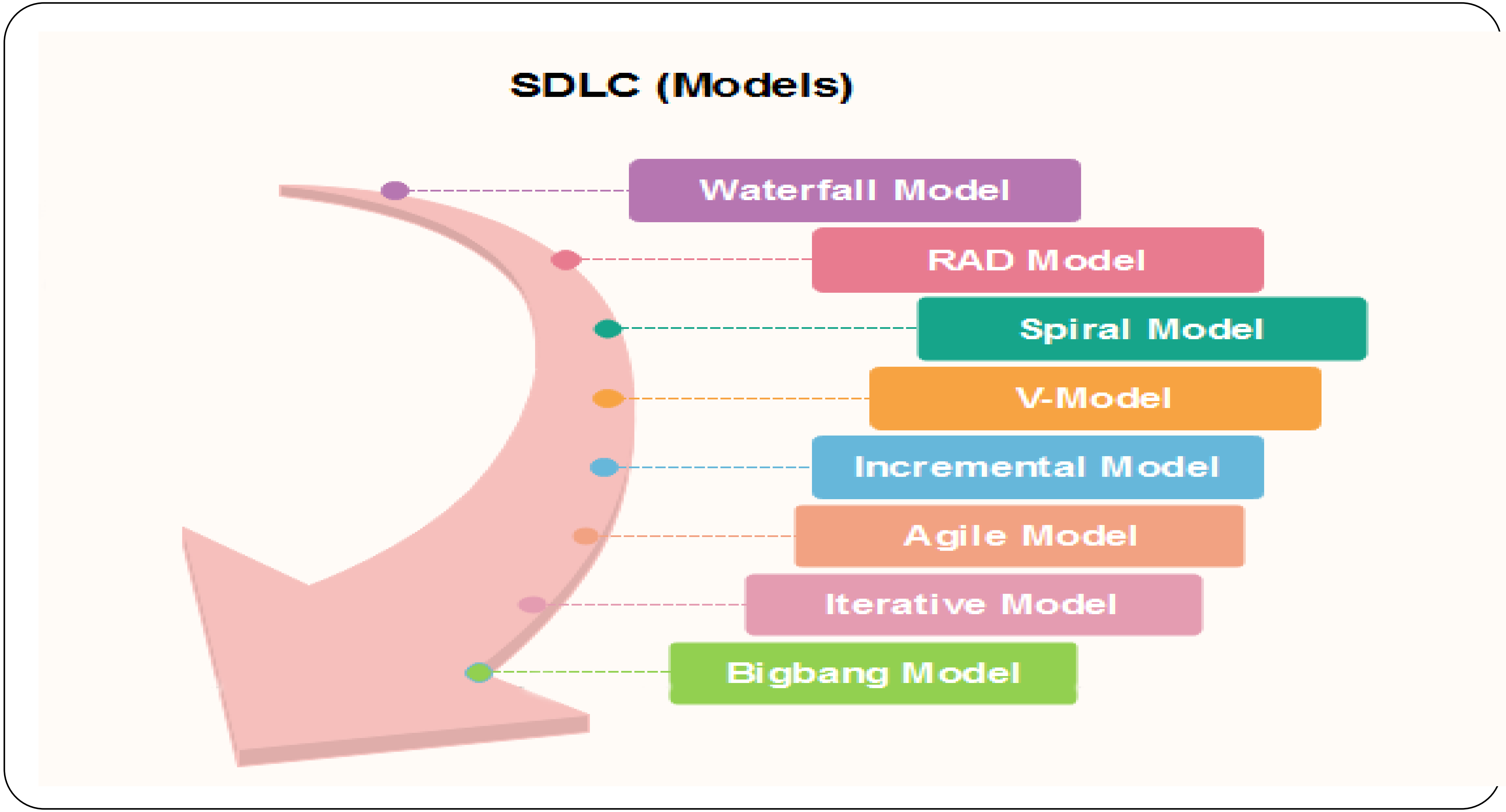
- After the code is generated, it is tested against the requirements to make sure that the products are solving the needs addressed and gathered during the requirements stage.
- During this stage, unit testing, integration testing, system testing, acceptance testing are done.

**Stage6: Deployment**

- Once the software is certified, and no bugs or errors are stated, then it is deployed.
- Then based on the assessment, the software may be released as it is or with suggested enhancement in the object segment.
- After the software is deployed, then its maintenance begins.

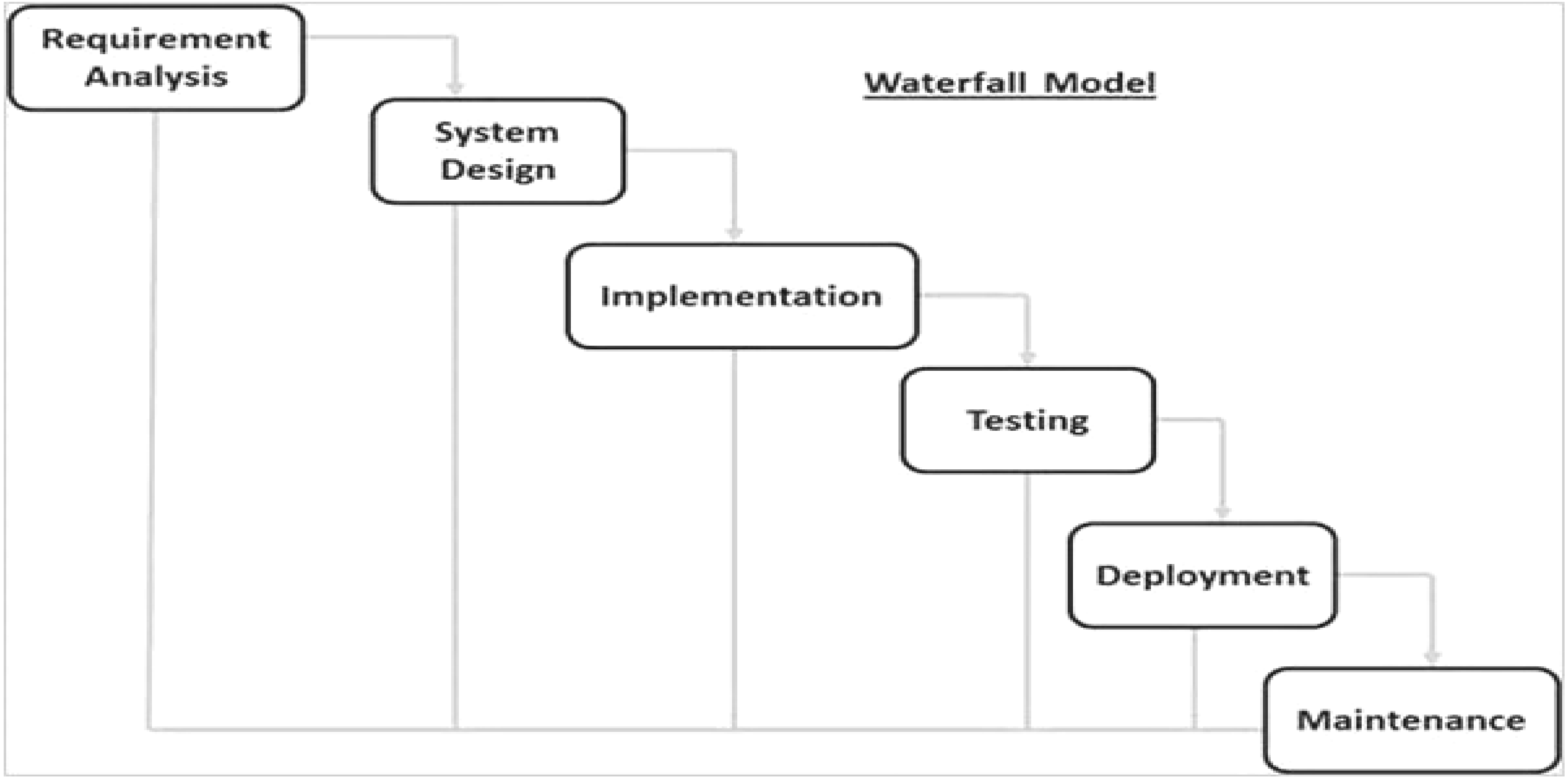
**Stage7: Maintenance**

- Once when the client starts using the developed systems, then the real issues come up and requirements to be solved from time to time.
- This procedure where the care is taken for the developed product is known as maintenance.


SDLC (Models)
Waterfall Model
RAD Model
Spiral Model
V-Model
Incremental Model
Agile Model
Iterative Model
Bigbang Model

## Waterfall Model

- The Waterfall Model was the first Process Model to be introduced. It is also referred to as a **linear-sequential life cycle model**.
- It is very simple to understand and use.
- In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.
- The Waterfall model is the earliest SDLC approach that was used for software development.
- The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.


Requirement Analysis
Waterfall Model
System Design
Implementation
Testing
Deployment
Maintenance

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

## Application of Waterfall Model

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

## Advantages of Waterfall Model

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

## Disadvantages of Waterfall Model

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.

## Iterative Model

- Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented.
- At each iteration, design modifications are made and new functional capabilities are added.
- The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).
- Iterative and Incremental development is a combination of both iterative design or iterative method and incremental build model for development. "During software development, more than one iteration of the software development cycle may be in progress at the same time." This process may be described as an "evolutionary acquisition" or "incremental build" approach."

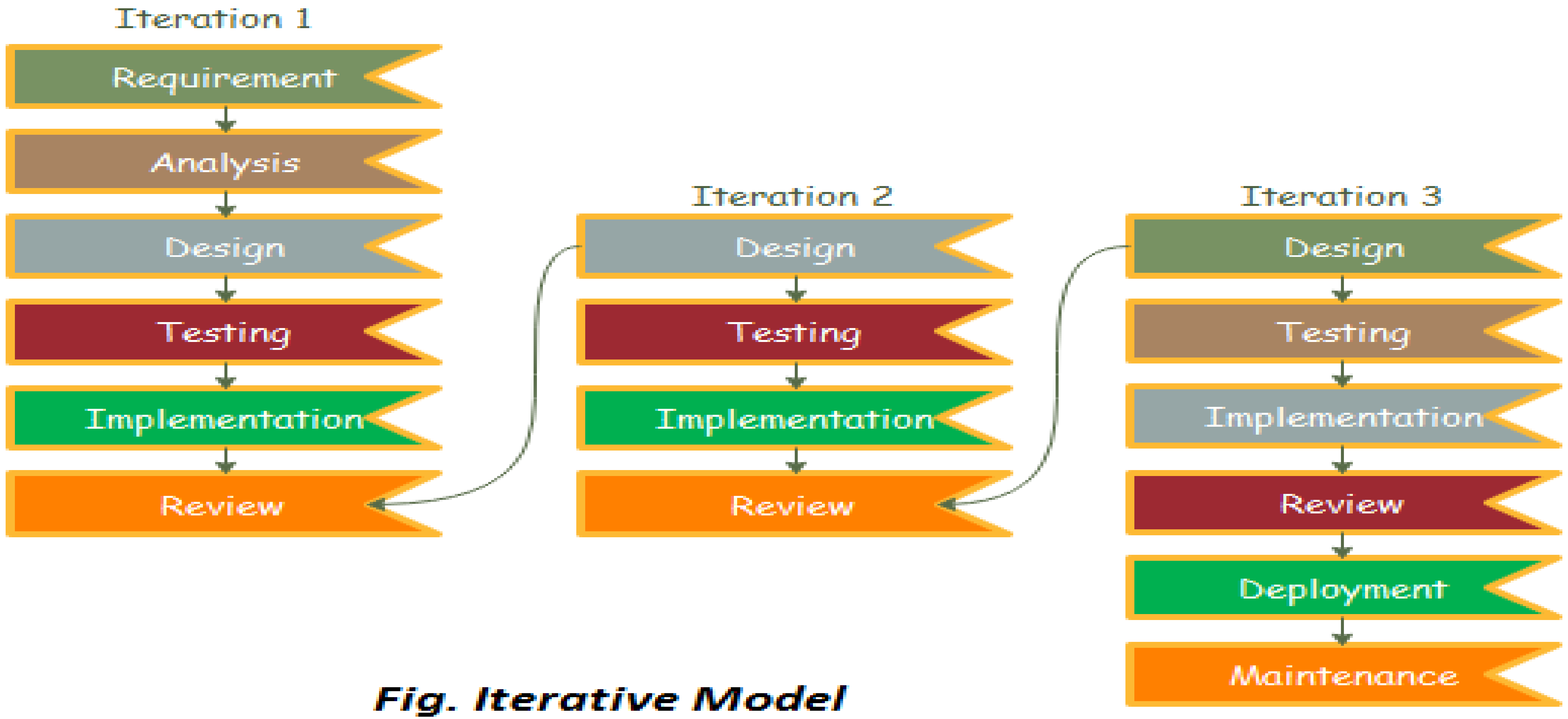


***Fig. Iterative Model***

## When to use Iterative Model

- When requirements are defined clearly and easy to understand.
- When the software application is large.
- When there is a requirement of changes in future.

## Advantages

- Testing and debugging during smaller iteration is easy.
- A Parallel development can plan.
- It is easily acceptable to ever-changing needs of the project.
- Risks are identified and resolved during iteration.
- Limited time spent on documentation and extra time on designing.

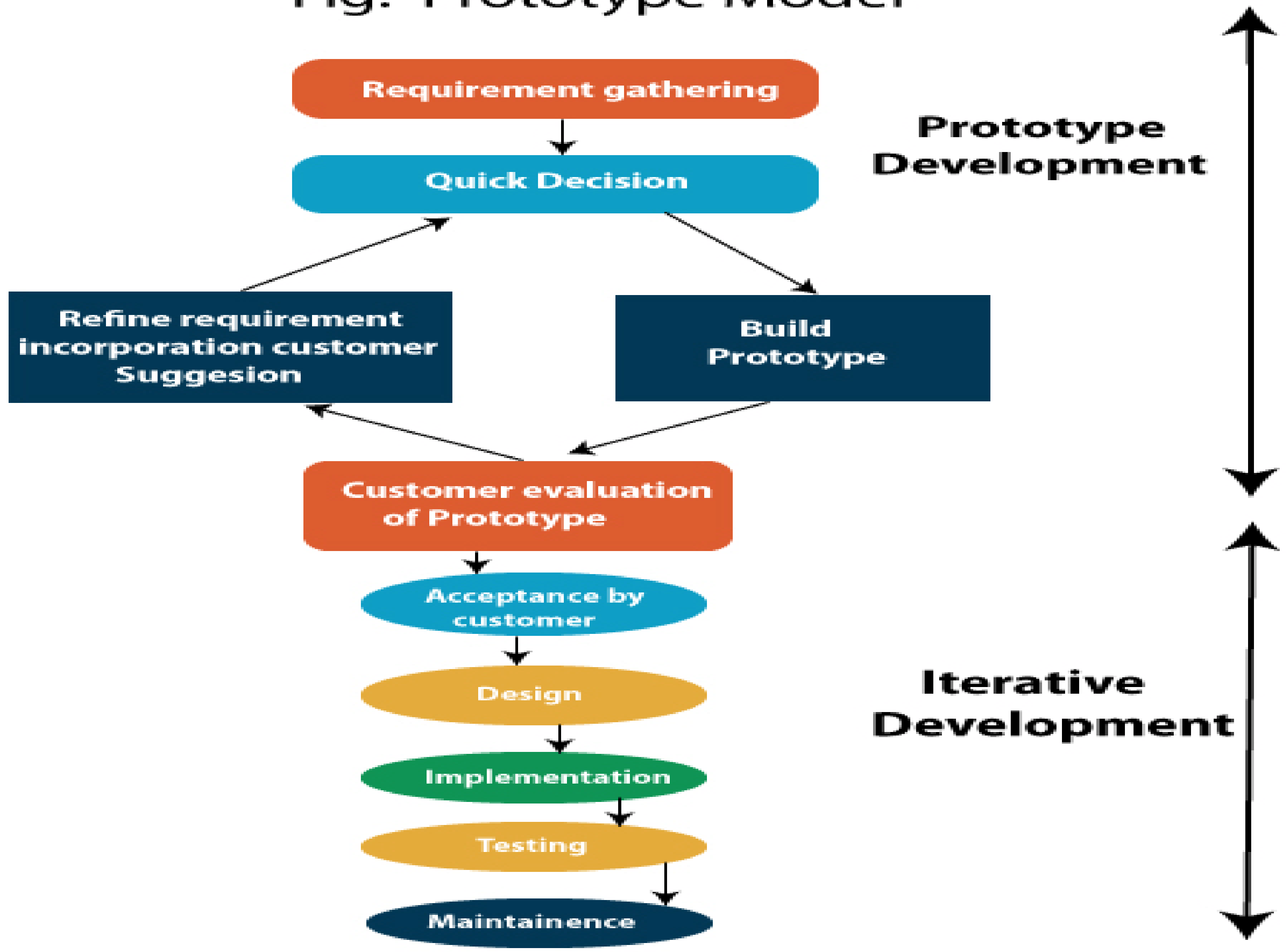
## Disadvantages

- It is not suitable for smaller projects.
- More Resources may be required.
- Design can be changed again and again because of imperfect requirements.
- Requirement changes can cause over budget.
- Project completion date not confirmed because of changing requirements.

## Prototype Model

- The prototype model requires that before carrying out the development of actual software, a working prototype of the system should be built.
- A prototype is a toy implementation of the system.
- A prototype usually turns out to be a very crude version of the actual system, possible exhibiting limited functional capabilities, low reliability, and inefficient performance as compared to actual software.
- In many instances, the client only has a general view of what is expected from the software product. In such a scenario where there is an absence of detailed information regarding the input to the system, the processing needs, and the output requirement, the prototyping model may be employed.

Fig: Prototype Model



## Steps of Prototype Model

- Requirement Gathering and Analyst
- Quick Decision
- Build a Prototype
- Assessment or User Evaluation
- Prototype Refinement
- Engineer Product

## Advantage of Prototype Model

- Reduce the risk of incorrect user requirement
- Good where requirement are changing/uncommitted
- Regular visible process aids management
- Support early product marketing
- Reduce Maintenance cost.
- Errors can be detected much earlier as the system is made side by side.

## Disadvantage of Prototype Model

- An unstable/badly implemented prototype often becomes the final product.
- Require extensive customer collaboration
  - Costs customer money
  - Needs committed customer
  - Difficult to finish if customer withdraw
  - May be too customer specific, no broad market
- Difficult to know how long the project will last.
- Easy to fall back into the code and fix without proper requirement analysis, design, customer evaluation, and feedback.
- Prototyping tools are expensive.
- Special tools & techniques are required to build a prototype.
- It is a time-consuming process.

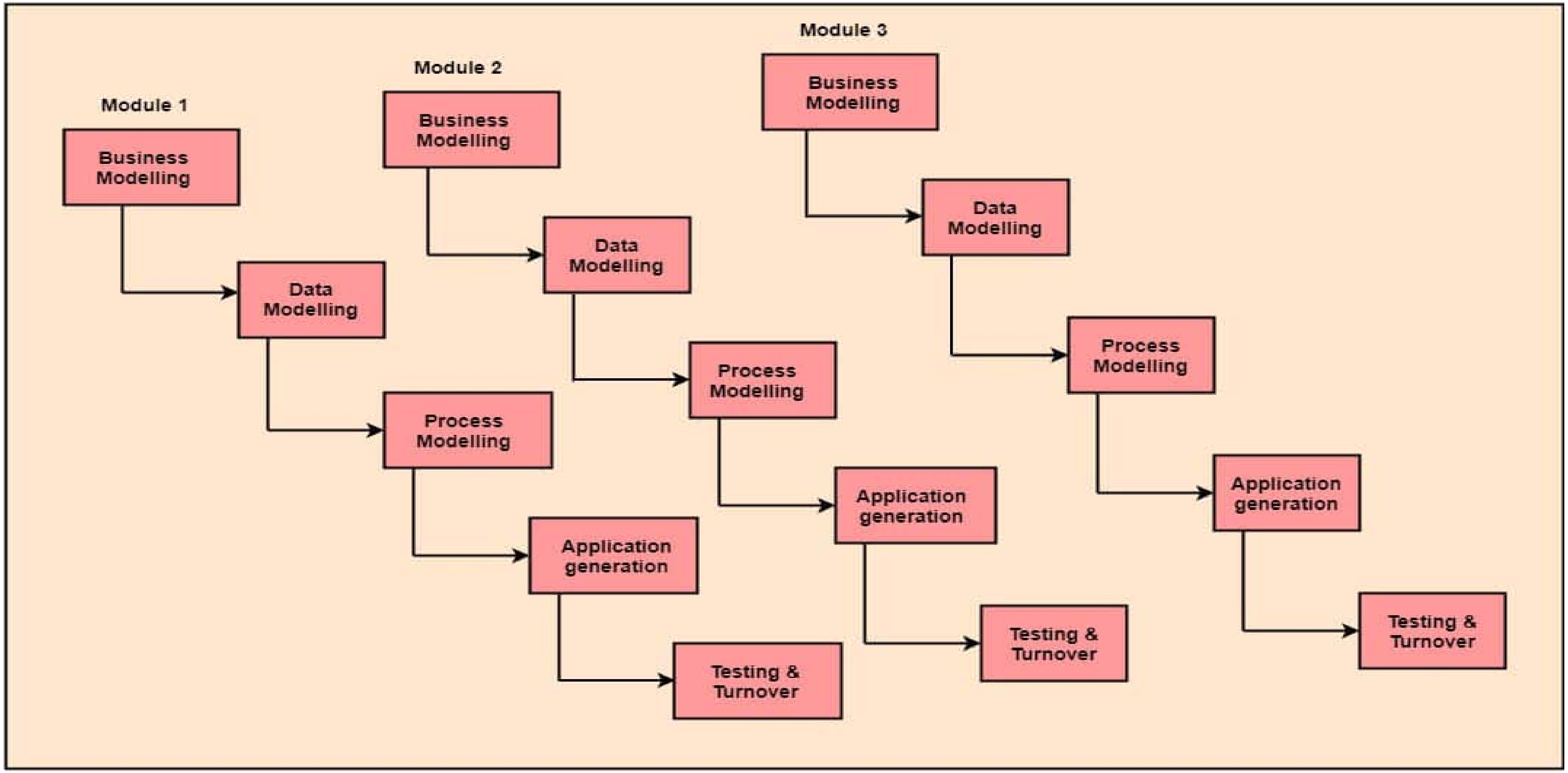
## RAD(Rapid Application Development)

- The RAD model or Rapid Application Development model is a type of software development methodology that emphasizes quick and iterative release cycles, primarily focusing on delivering working software in shorter timelines.
- Unlike traditional models such as the Waterfall model, RAD is designed to be more flexible and responsive to user feedback and changing requirements throughout the development process.
- The **Rapid Application Development (RAD) model** is a type of software development methodology that prioritizes rapid prototyping and quick feedback over long, drawn-out development and testing cycles. It is designed to deliver high-quality systems quickly by focusing on user feedback and iterative improvements.

RAD (Rapid Application Development) is a concept that products can be developed faster and of higher quality through:

- Gathering requirements using workshops or focus groups
- Prototyping and early, reiterative user testing of designs
- The re-use of software components
- A rigidly paced schedule that refers design improvements to the next product version
- Less formality in reviews and other team communication

Fig: RAD Model



## Why we use RAD?

- Quick protoyping
- Iterative  Development
- Incremental Releases
- User Involvement
- Time Boxing
- Parallel  Development

1. **Business Modelling:** The information flow among business functions is defined by answering questions like what data drives the business process, what data is generated, who generates it, where does the information go, who process it and so on.
2. **Data Modelling:** The data collected from business modeling is refined into a set of data objects (entities) that are needed to support the business. The attributes (character of each entity) are identified, and the relation between these data objects (entities) is defined.
3. **Process Modelling:** The information object defined in the data modeling phase are transformed to achieve the data flow necessary to implement a business function. Processing descriptions are created for adding, modifying, deleting, or retrieving a data object.
4. **Application Generation:** Automated tools are used to facilitate construction of the software; even they use the 4th GL techniques.
5. **Testing & Turnover:** Many of the programming components have already been tested since RAD emphasis reuse. This reduces the overall testing time. But the new part must be tested, and all interfaces must be fully exercised.

## When to use RAD Model?

- When the system should need to create the project that modularizes in a short span time (2-3 months).
- When the requirements are well-known.
- When the technical risk is limited.
- When there's a necessity to make a system, which modularized in 2-3 months of period.
- It should be used only if the budget allows the use of automatic code generating tools.

## Advantage of RAD Model

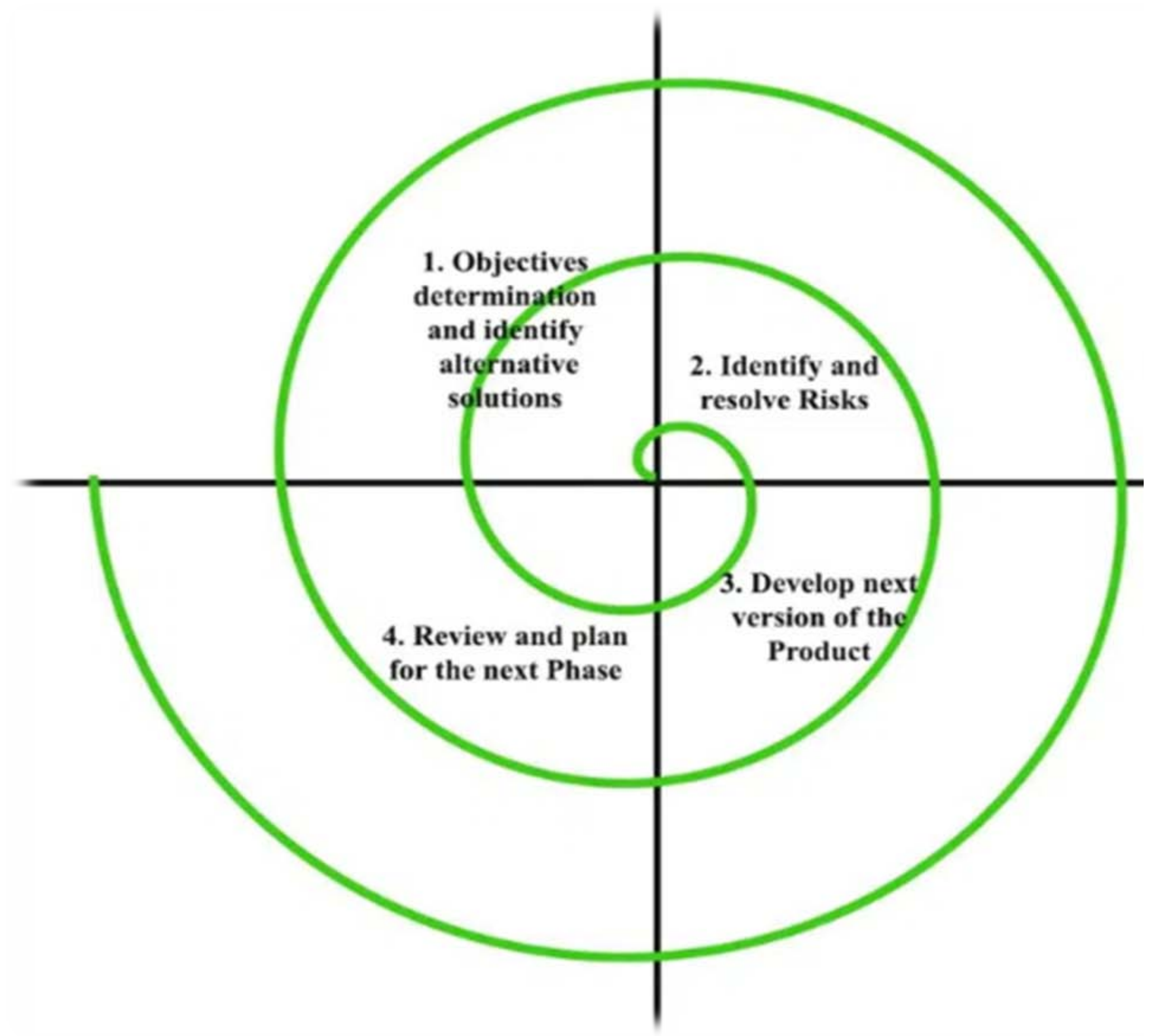
- This model is flexible for change.
- In this model, changes are adoptable.
- Each phase in RAD brings highest priority functionality to the customer.
- It reduced development time.
- It increases the reusability of features.

## Disadvantage of RAD Model

- It required highly skilled designers.
- All application is not compatible with RAD.
- For smaller projects, we cannot use the RAD model.
- On the high technical risk, it's not suitable.
- Required user involvement.

## Spiral Model

- The Spiral Model is a Software Development Life Cycle (SDLC) model that provides a systematic and iterative approach to software development.
- In its diagrammatic representation, looks like a spiral with many loops.
- The exact number of loops of the spiral is unknown and can vary from project to project.
- Each loop of the spiral is called a phase of the software development process.
- It is the combination of waterfall model and iterative model.
- Each phase in spiral model begin with design goal and ends with client reviewing.
- Software is developed in a series of incremental releases.


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

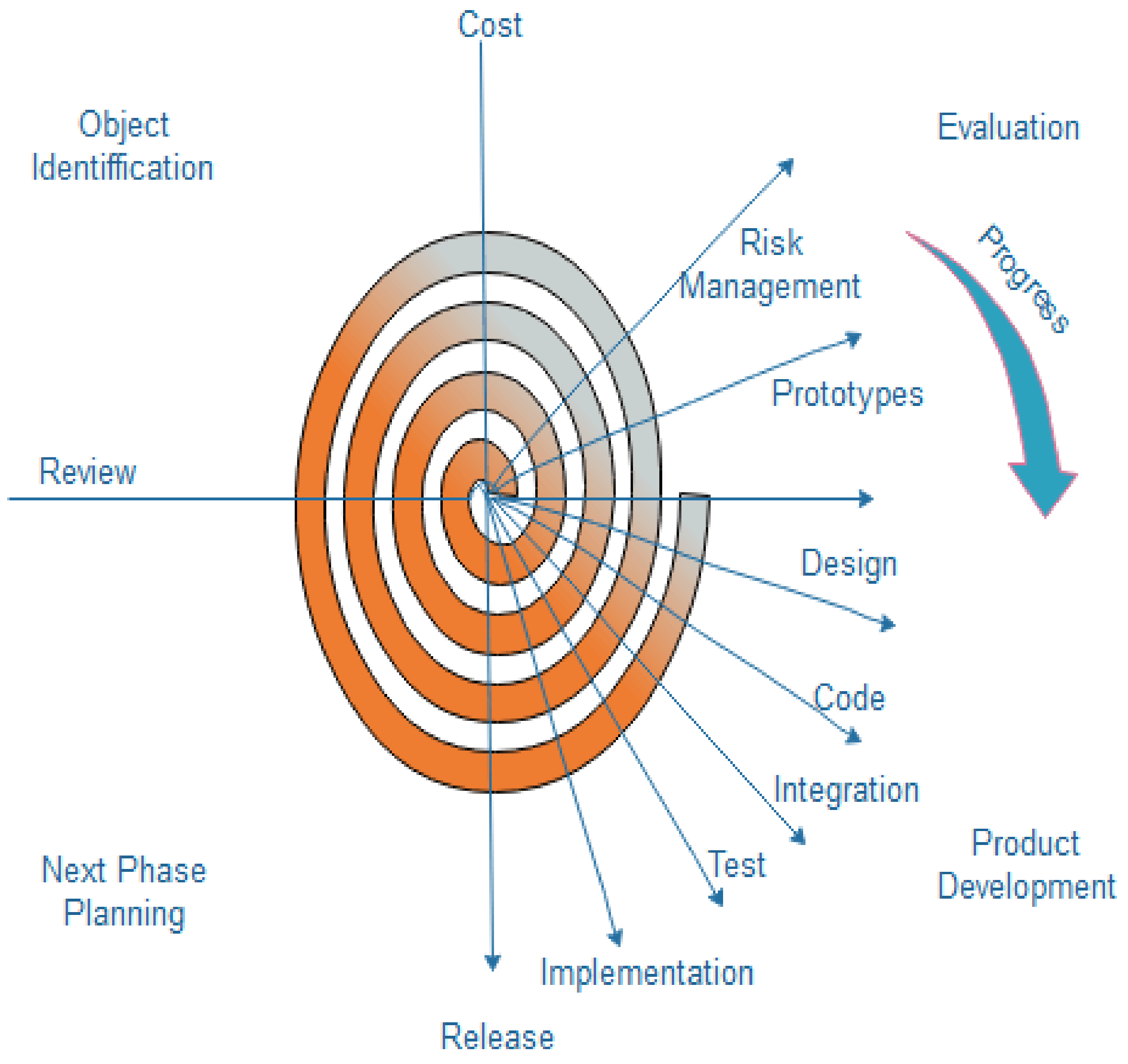


*Fig. Spiral Model*

- **Objectives determination and identify alternative solutions**
  - Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase.
  - Then alternative solutions possible for the phase are proposed in this quadrant.
- **Identify and resolve Risks**
  - During the second quadrant, all the possible solutions are evaluated to select the best possible solution.
  - Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy.
  - At the end of this quadrant, the Prototype is built for the best possible solution.
- **Develop the next version of the Product:**
  - During the third quadrant, the identified features are developed and verified through testing.
  - At the end of the third quadrant, the next version of the software is available.
- **Review and plan for the next Phase**
  - In the fourth quadrant, the Customers evaluate the so-far developed version of the software.
  - In the end, planning for the next phase is started.

## When to use spiral model?

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

**Advantages:**

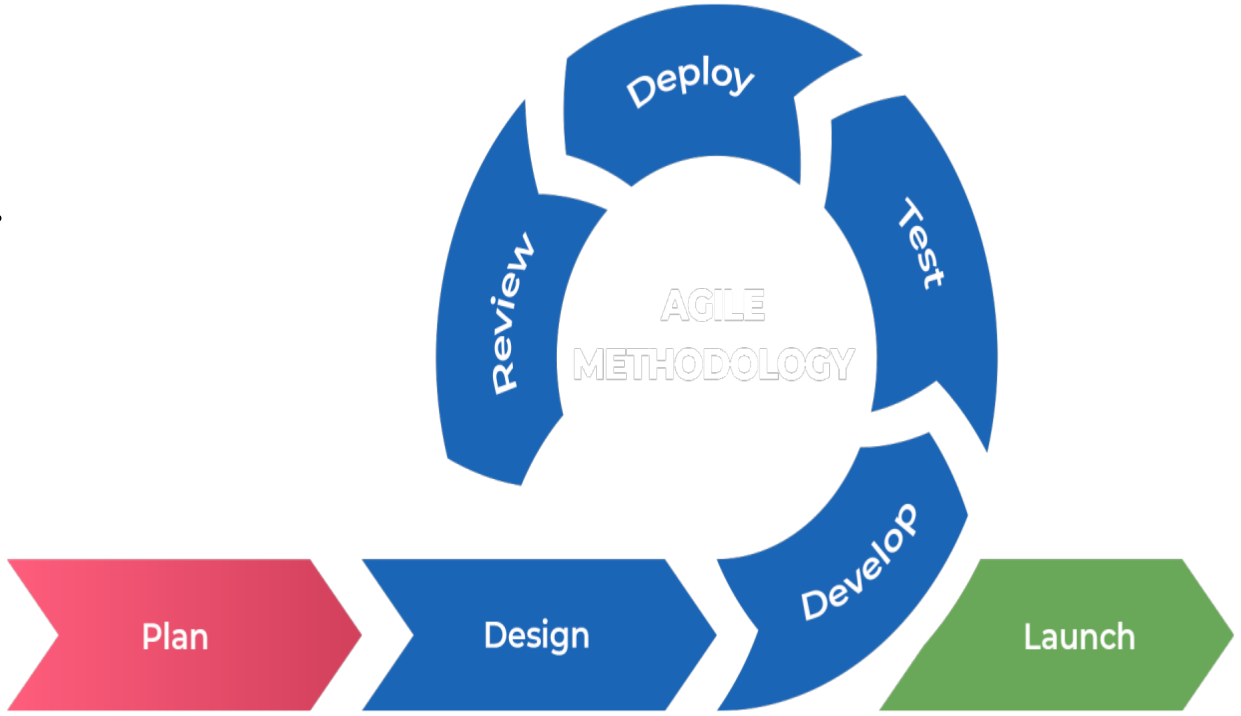
- Additional functionality or changes can be done at later stage.
- Cost estimation becomes easy.
- Development is fast and features are added in a systematic way
- There is always space for customer feedback.
- High amount of risk analysis.
- Useful for large and mission-critical projects.

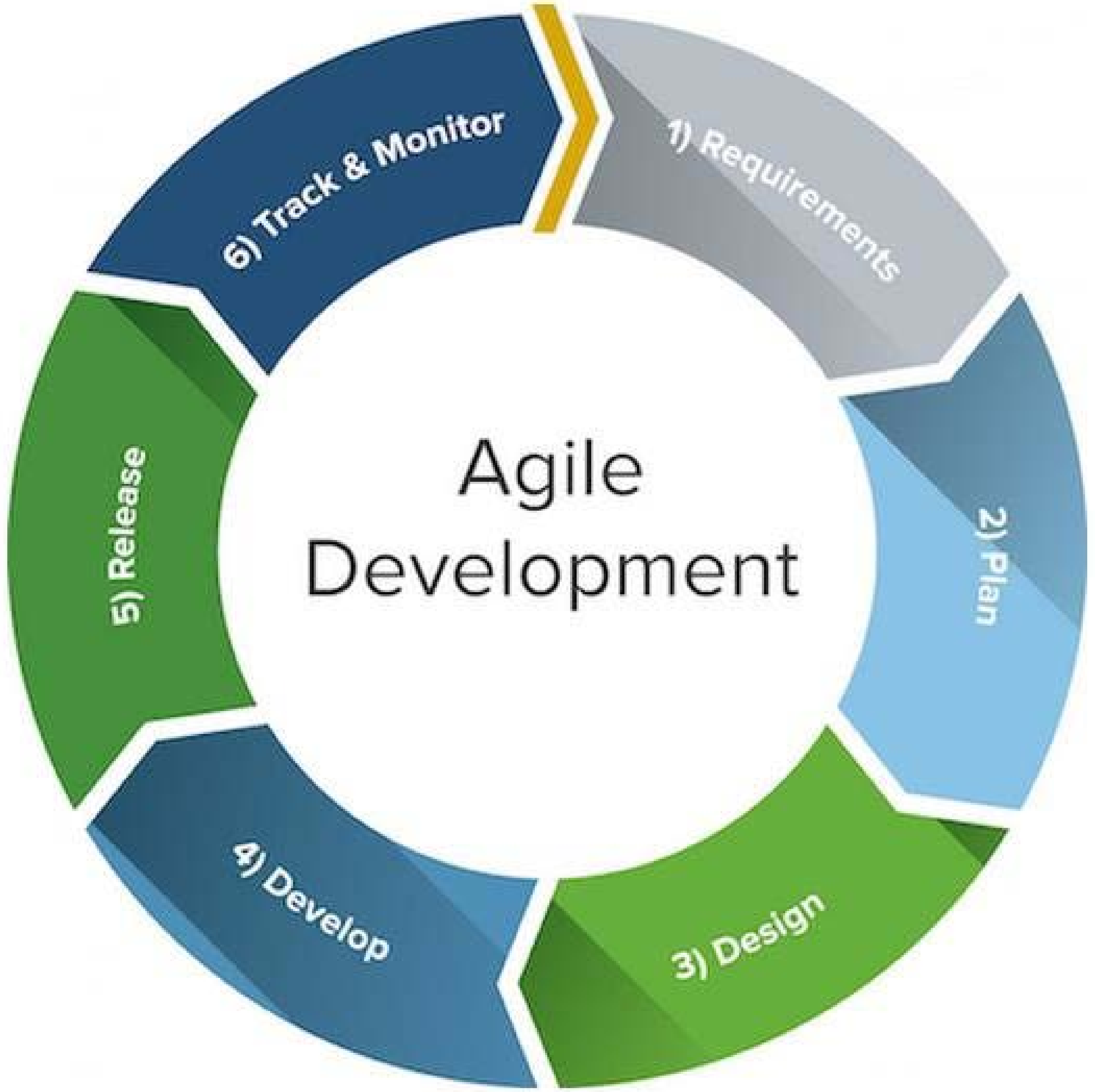
## Disadvantages

- Can be a costly model to use.
- Risk analysis needed highly particular expertise.
- Doesn’t work well for smaller projects.
- Risk is not meeting the schedule as budget.

## Agile Model

- The meaning of Agile is swift or versatile.
- In earlier days, the **Iterative Waterfall Model** was very popular for completing a project.
- But nowadays, developers face various problems while using it to develop software.
- The main difficulties included handling customer change requests during project development and the high cost and time required to incorporate these changes.
- To overcome these drawbacks of the Waterfall Model, in the mid-1990s the **Agile Software Development** model was proposed.





Agile Development
1) Requirements
2) Plan
3) Design
4) Develop
5) Release
6) Track & Monitor

- "**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.
- **The Agile Model** was primarily designed to help a project adapt quickly to change requests. So, the main aim of the Agile model is to facilitate quick project completion.
- To accomplish this task, agility is required. Agility is achieved by fitting the process to the project and removing activities that may not be essential for a specific project.

- 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.

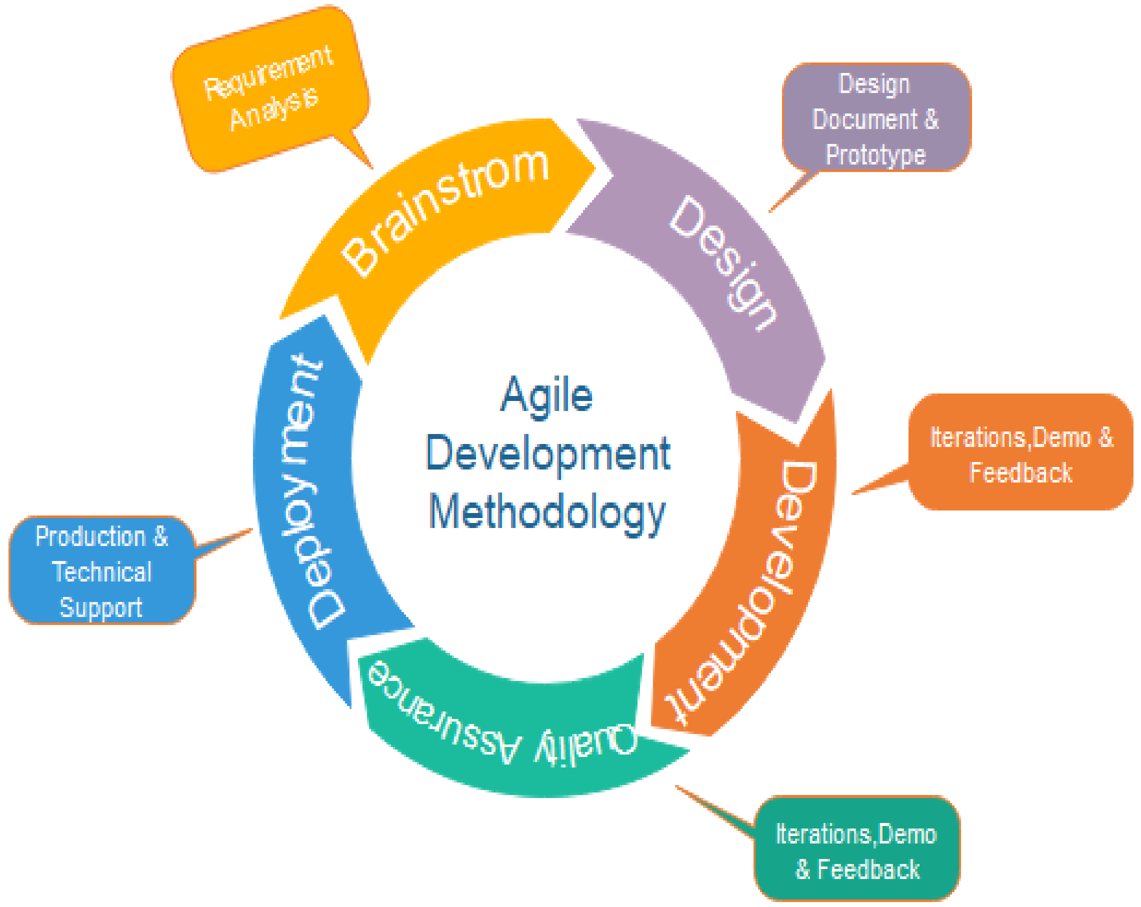

Requirement Analysis
Design Document & Prototype
Brainstrom
Design
Agile Development Methodology
Iterations,Demo & Feedback
Development
Production & Technical Support
Deployment
Quality Assurance
Iterations,Demo & Feedback

*Fig. Agile Model*

- **Requirement Gathering:-** In this step, the development team must gather the requirements, by interaction with the customer. development team should plan the time and effort needed to build the project. Based on this information you can evaluate technical and economical feasibility.
- **Design the Requirements:-** In this step, the development team will use user-flow-diagram or high-level UML(Unified Modeling Language) diagrams to show the working of the new features and show how they will apply to the existing software. Wireframing and designing user interfaces are done in this phase.
- **Construction / Iteration:-** In this step, development team members start working on their project, which aims to deploy a working product.

- **Testing / Quality Assurance:-** Testing involves Unit Testing, Integration Testing, and System Testing. A brief introduction of these three tests is as follows:

  **Unit Testing:-** Unit testing is the process of checking small pieces of code to ensure that the individual parts of a program work properly on their own. Unit testing is used to test individual blocks (units) of code.

  **Integration Testing:-** Integration testing is used to identify and resolve any issues that may arise when different units of the software are combined.

  **System Testing:-** Goal is to ensure that the software meets the requirements of the users and that it works correctly in all possible scenarios.

- **Deployment:-** In this step, the development team will deploy the working project to end users.
- **Feedback:-** This is the last step of the Agile Model**.** In this, the team receives feedback about the product and works on correcting bugs based on feedback provided by the customer.

## Principles of the Agile Model

- To establish close contact with the customer during development and to gain a clear understanding of various requirements, each Agile project usually includes a customer representative on the team. At the end of each iteration stakeholders and the customer representative review, the progress made and re-evaluate the requirements.
- The agile model relies on working software deployment rather than comprehensive documentation.
- Frequent delivery of incremental versions of the software to the customer representative in intervals of a few weeks.
- Requirement change requests from the customer are encouraged and efficiently incorporated.
- It emphasizes having efficient team members and enhancing communications among them is given more importance. It is realized that improved communication among the development team members can be achieved through face-to-face communication rather than through the exchange of formal documents.
- It is recommended that the development team size should be kept small (5 to 9 people) to help the team members meaningfully engage in face-to-face communication and have a collaborative work environment.
- The agile development process usually deploys Pair Programming. In Pair programming, two programmers work together at one workstation. One does coding while the other reviews the code as it is typed in. The two programmers switch their roles every hour or so.

## When To Use the Agile Model?

- When frequent modifications need to be made, this method is implemented.
- When a highly qualified and experienced team is available.
- When a customer is ready to have a meeting with the team all the time.
- When the project needs to be delivered quickly.
- Projects with few regulatory requirements or not certain requirements.
- Those undertakings where the product proprietor is easily reachable
- Flexible project schedules and budgets.

## Advantages of the Agile Model

- Working through Pair programming produces well-written compact programs which have fewer errors as compared to programmers working alone.
- It reduces the total development time of the whole project.
- Agile development emphasizes face-to-face communication among team members, leading to better collaboration and understanding of project goals.
- Customer representatives get the idea of updated software products after each iteration. So, it is easy for him to change any requirement if needed.
- Agile development puts the customer at the center of the development process, ensuring that the end product meets their needs.

## Disadvantages of the Agile Model

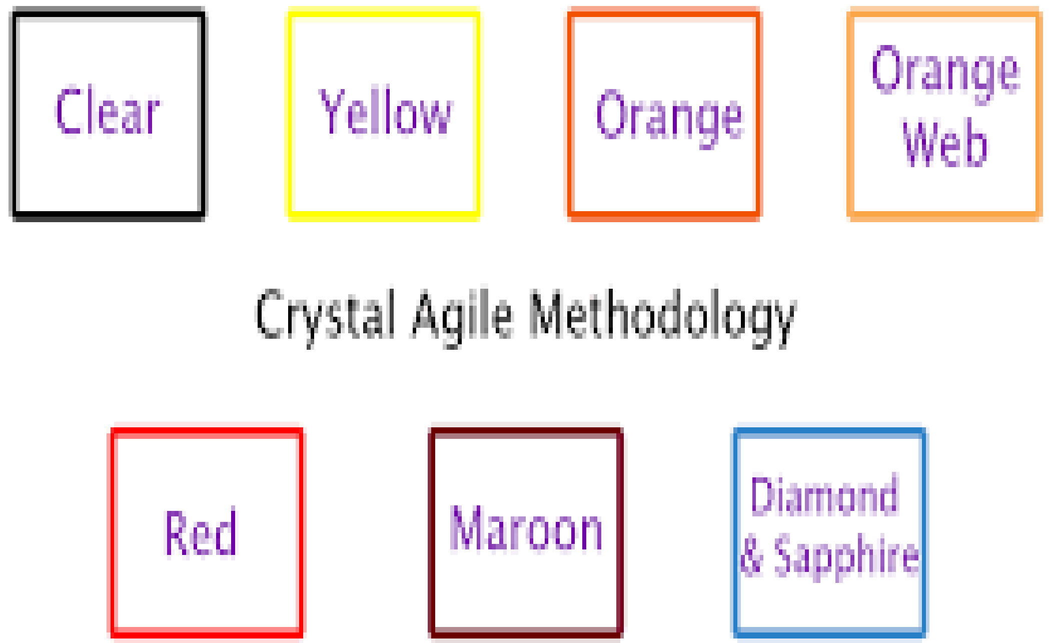
- The lack of formal documents creates confusion and important decisions taken during different phases can be misinterpreted at any time by different team members.
- It is not suitable for handling complex dependencies.
- The agile model depends highly on customer interactions so if the customer is not clear, then the development team can be driven in the wrong direction.
- Agile development models often involve working in short sprints, which can make it difficult to plan and forecast project timelines and deliverables.
- Agile development models require a high degree of expertise from team members, as they need to be able to adapt to changing requirements and work in an iterative environment.
- Due to the absence of proper documentation, when the project completes and the developers are assigned to another project, maintenance of the developed project can become a problem.

## Some Agile SDLC Models

- Crystal Agile methodology
- Dynamic Systems Development Method (DSDM)
- Feature-driven development (FDD)
- Scrum
- Extreme Programming (XP)
- Lean Development
- Unified Process

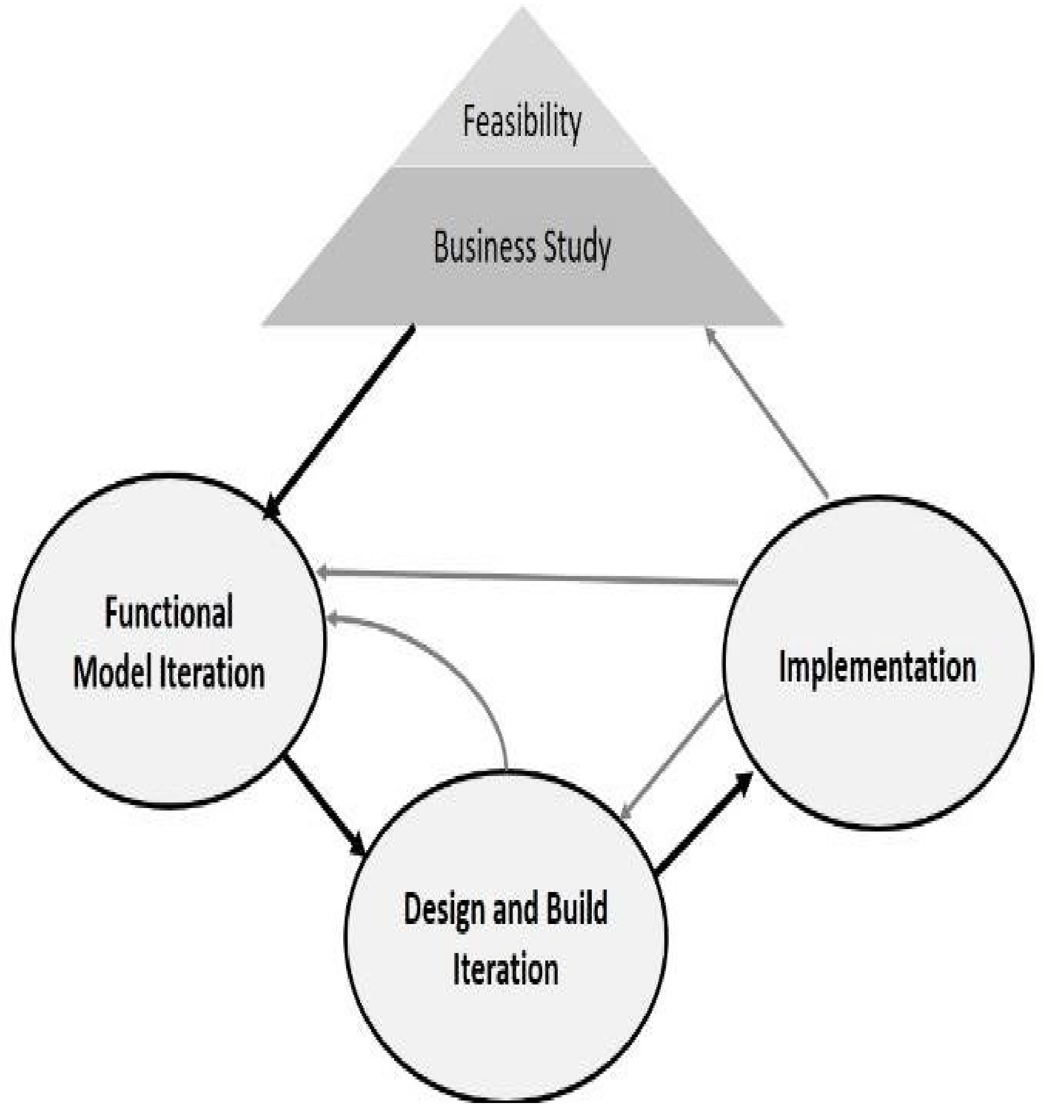
## ❑Crystal Agile methodology

- The crystal method is an agile framework that is considered a lightweight or agile methodology that focuses on individuals and their interactions.
- The methods are color-coded to significant risk to human life.
- It is mainly for short-term projects by a team of developers working out of a single workspace.



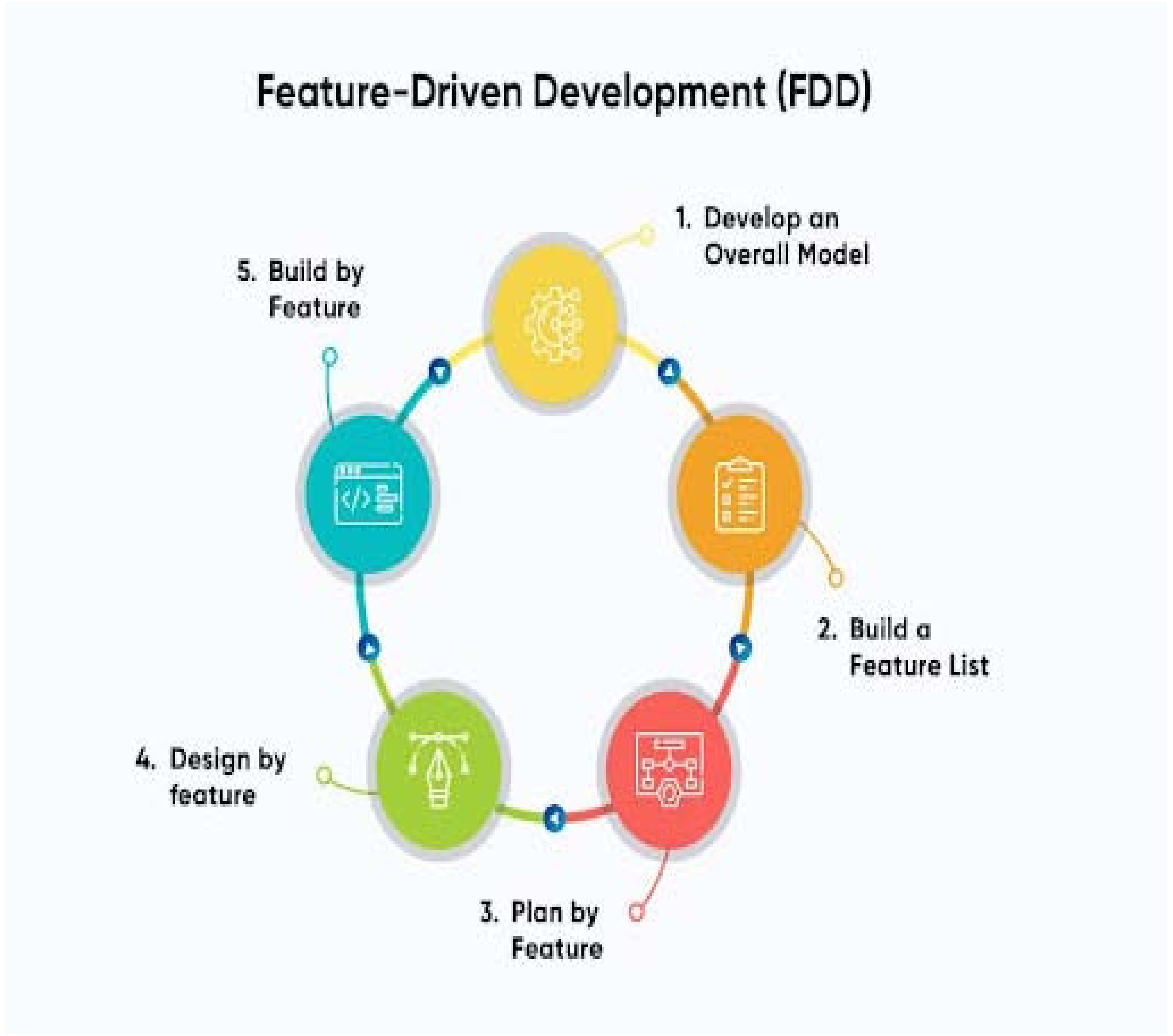
## ❑Dynamic Systems Development Method (DSDM)

- DSDM methodology is tailored for projects with moderate to high uncertainty where requirements are prone to change frequently.
- Its clear-cut roles and responsibilities focus on delivering working software in short time frames.
- Governance practices set it apart and make it an effective approach for teams and projects.



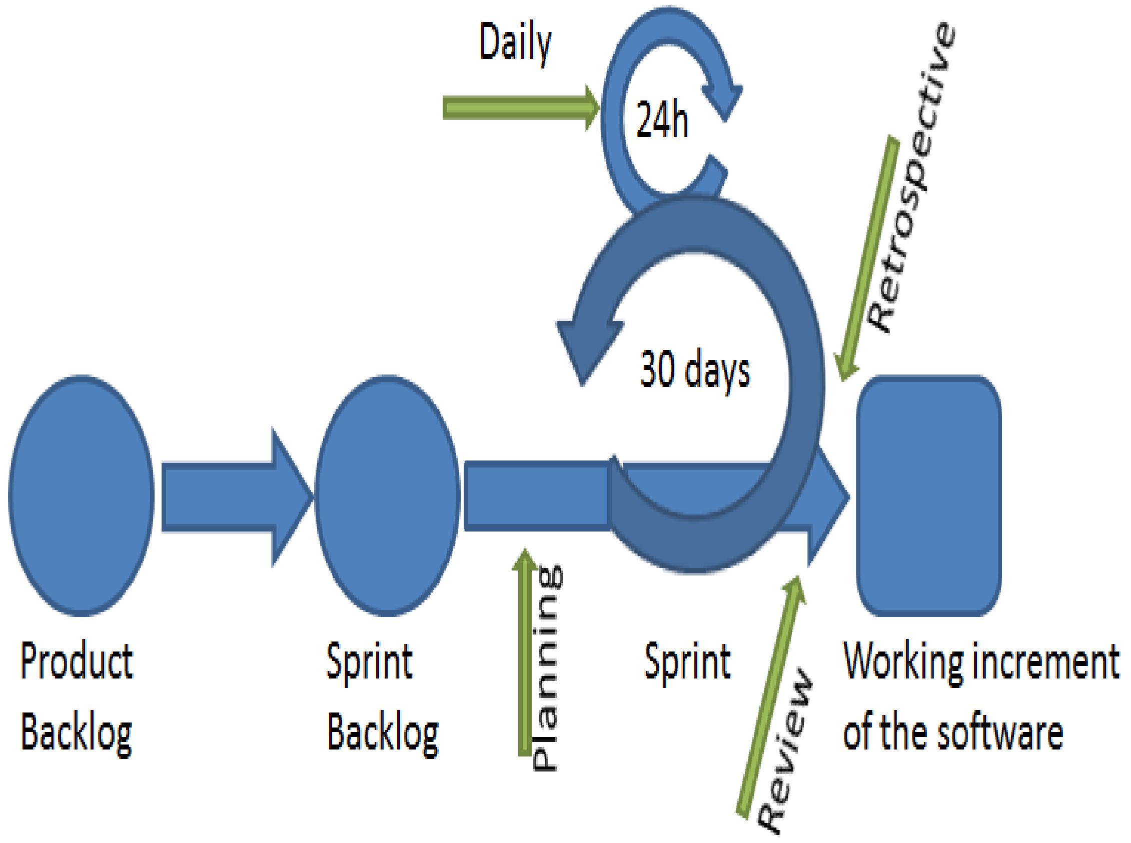
## ❑ Feature-driven development (FDD)

- ➢ FDD approach is implemented by utilizing a series of techniques, like creating feature lists, conducting model evaluations, and implementing a design-by-feature method, to meet its goal.
- ➢ This methodology is particularly effective in ensuring that the end product is delivered on time and that it aligns with the requirements of the customer.



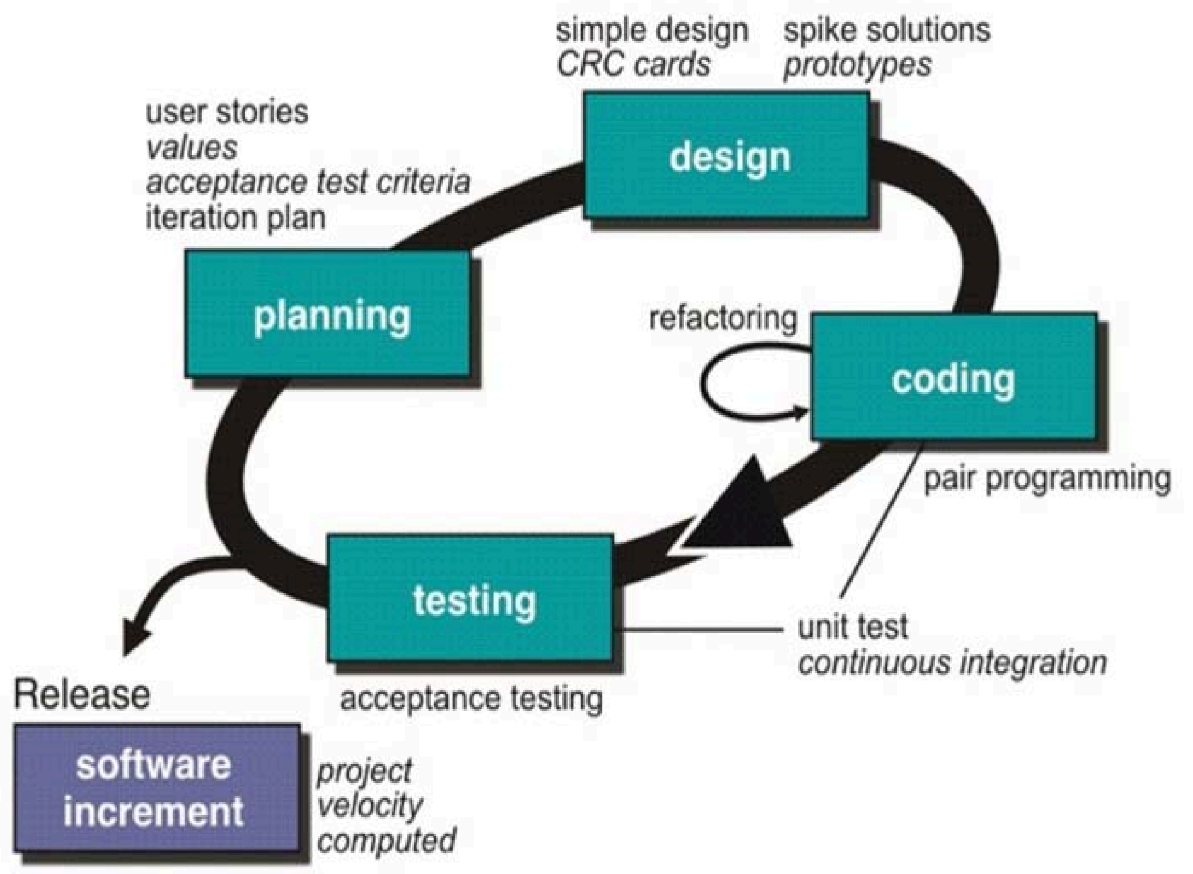
## ❑Scrum

- ➢ Scrum is a management framework that teams use to self-organize tasks and work towards a common goal.
- ➢ It is a framework within which people can address complex adaptive problems while the productivity and creativity of delivering products are at the highest possible value.
- ➢ Scrum is a management framework that teams use to self-organize and work towards a common goal.
- ➢ Scrum allows us to develop products of the highest value while making sure that we maintain creativity and productivity.
- ➢ The iterative and incremental approach used in scrum allows the teams to adapt to the changing requirements.



## Extreme Programming (XP)

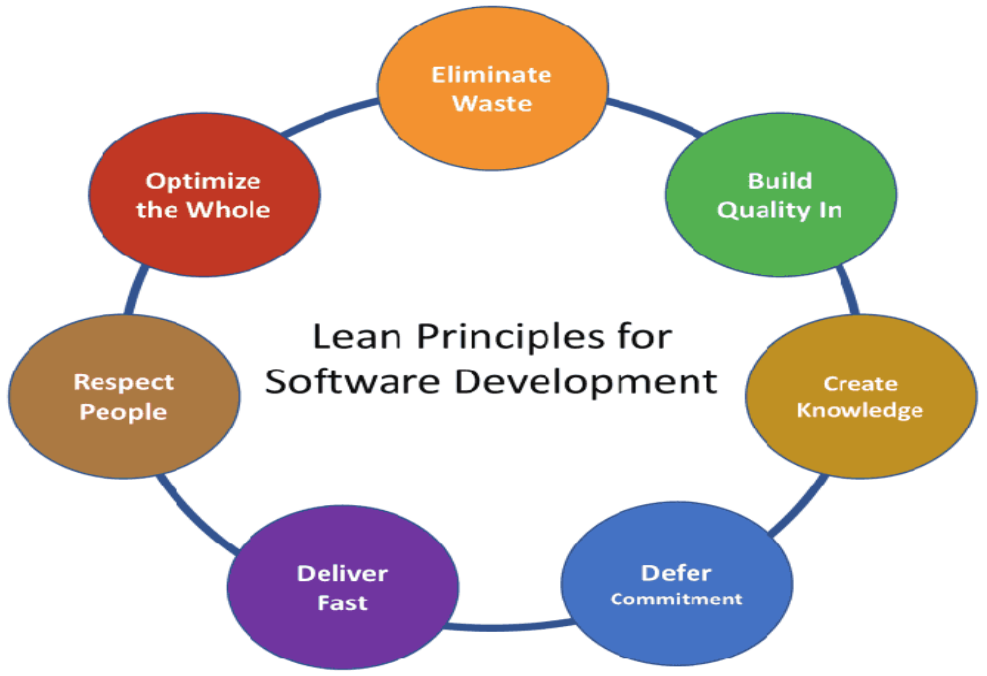
- Extreme Programming (XP) is an Agile software development methodology that focuses on delivering high-quality software through frequent and continuous feedback, collaboration, and adaptation.
- XP emphasizes a close working relationship between the development team, the customer, and stakeholders, with an emphasis on rapid, iterative development and deployment.



## ❑ Lean Development

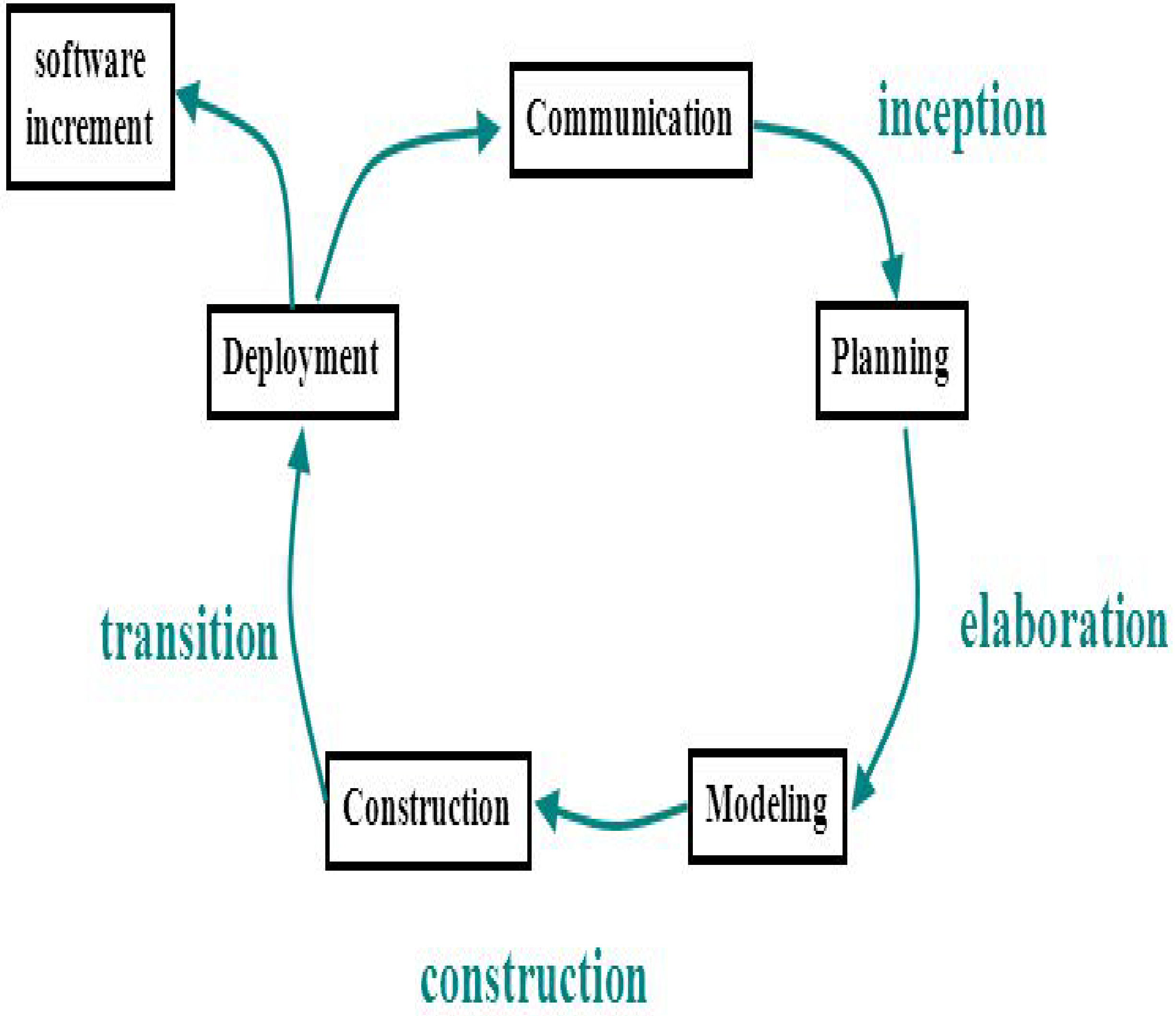
- Lean Software Development (LSD) is an agile framework used to streamline and optimize the software development process.
- It may also be referred to as the Minimum Viable Product (MVP) strategy as these ways of thinking are very similar since both intend to speed up development by focusing on new deliverables.
- Lean Software Development (LSD) is an approach derived from lean manufacturing principles aimed at optimizing efficiency and minimizing waste in the software development process.

- **Prevent Defects:** It integrates quality assurance throughout the development process to prevent defects.
- **Eliminate Waste:** It focuses on activities that add value to the customer and eliminates those activities that do not add value.
- **Fast Delivery:** Reduces cycle time to deliver software quickly and respond to feedback and changing requirements rapidly.
- **Delay Decisions:** Delay decisions until they can be made based on facts.



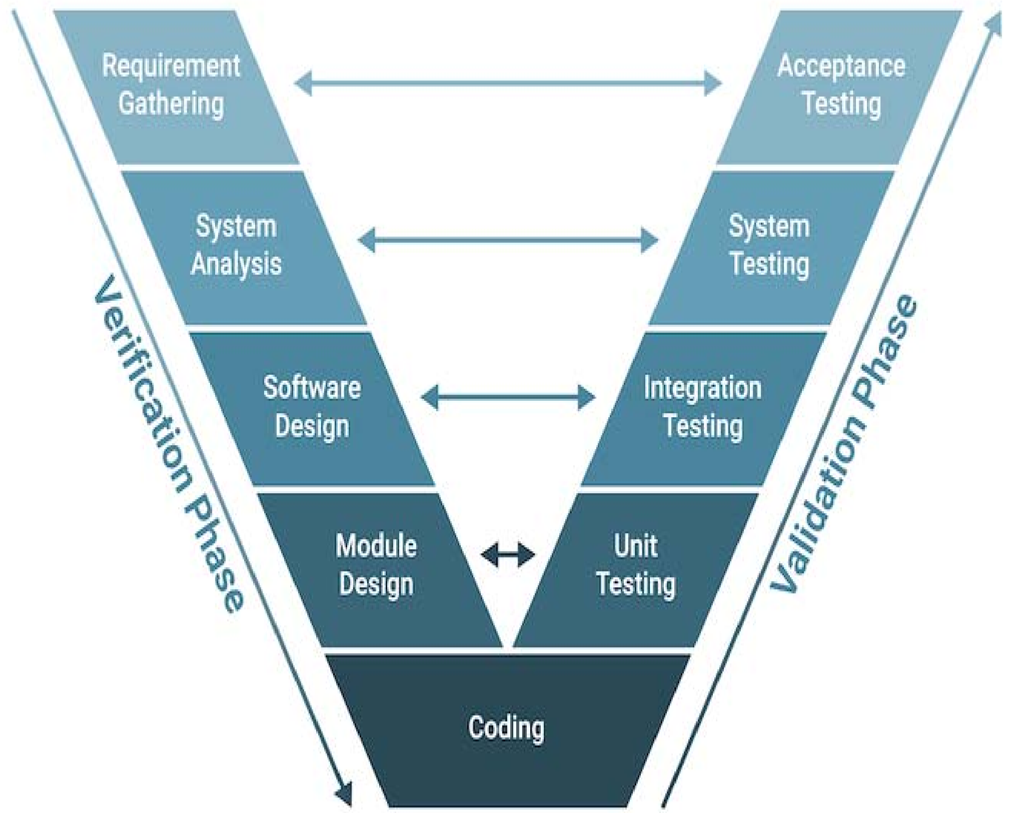
## Unified Process:

- Unified Process is a methodology that can be tailored to the specific needs of any given project.
- It combines elements of both waterfall and Agile methodologies, allowing for an iterative and incremental approach to development.
- This means that the UP is characterized by a series of iterations, each of which results in a working product increment, allowing for continuous improvement and the delivery of value to the customer.



## V-Model

- The V-model is a type of SDLC model where the process executes sequentially in a V-shape.
- It is also known as the Verification and Validation model.
- It is based on the association of a testing phase for each corresponding development stage.
- The development of each step is directly associated with the testing phase.
- The next phase starts only after completion of the previous phase i.e., for each development activity, there is a testing activity corresponding to it.



## Big Bang Model

- In this model, developers do not follow any specific process.
- Development begins with the necessary funds and efforts in the form of inputs. And the result may or may not be as per the customer's requirement, because in this model, even the customer requirements are not defined.
- This model is ideal for small projects like academic projects or practical projects.
- One or two developers can work together on this model.

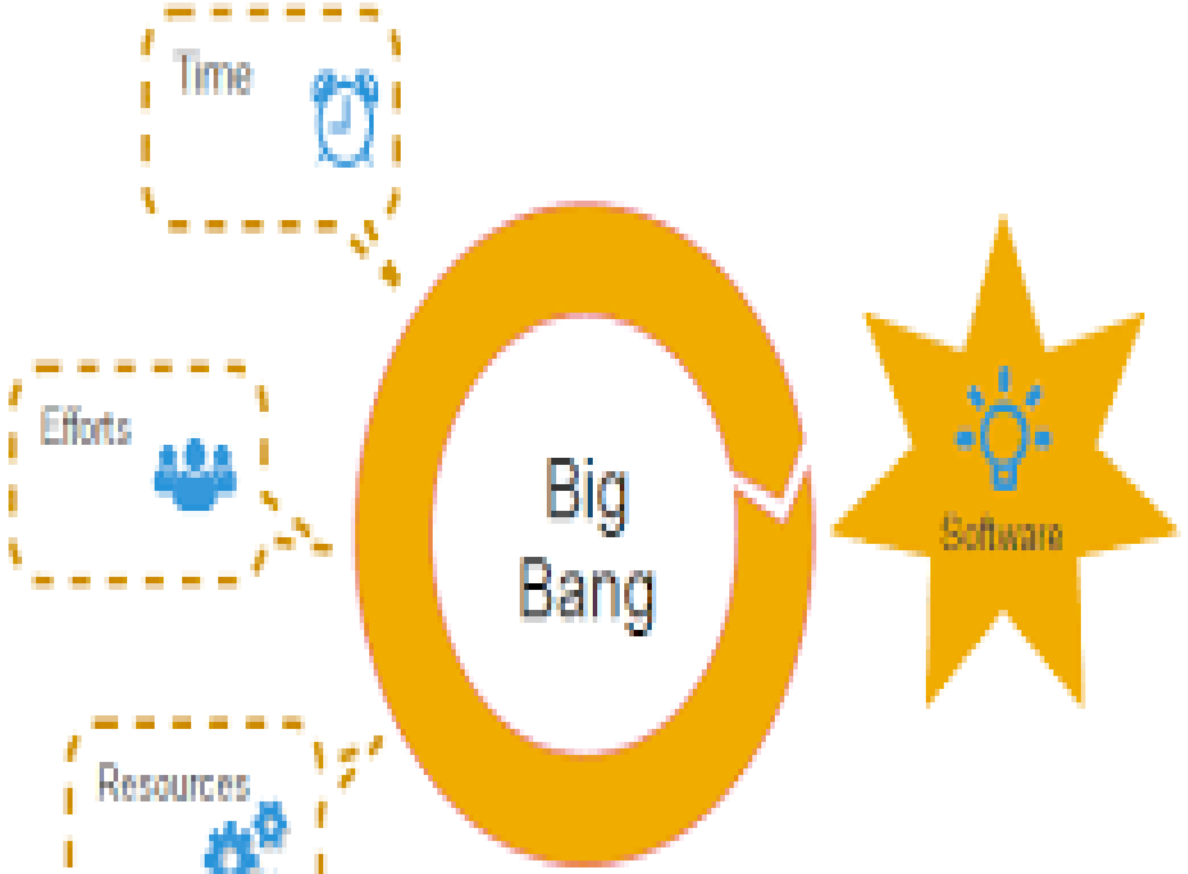
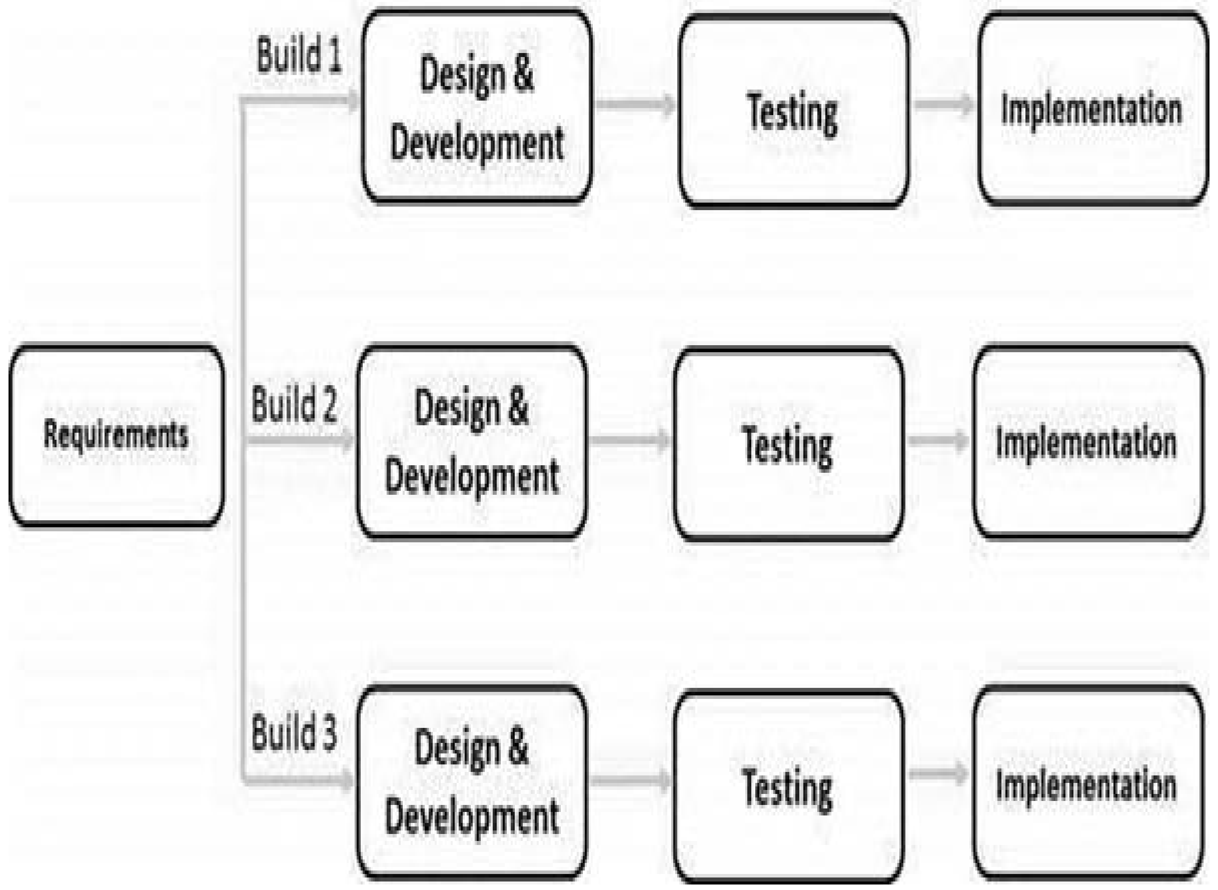


*Fig. Big Bang Model*

## Incremental Model

- Incremental Model is a process of software development where requirements divided into multiple standalone modules of the software development cycle.
- In this model, each module goes through the requirements, design, implementation and testing phases.
- Every subsequent release of the module adds function to the previous release.
- The process continues until the complete system achieved.


Requirements
Build 1
Design & Development
Testing
Implementation
Build 2
Design & Development
Testing
Implementation
Build 3
Design & Development
Testing
Implementation