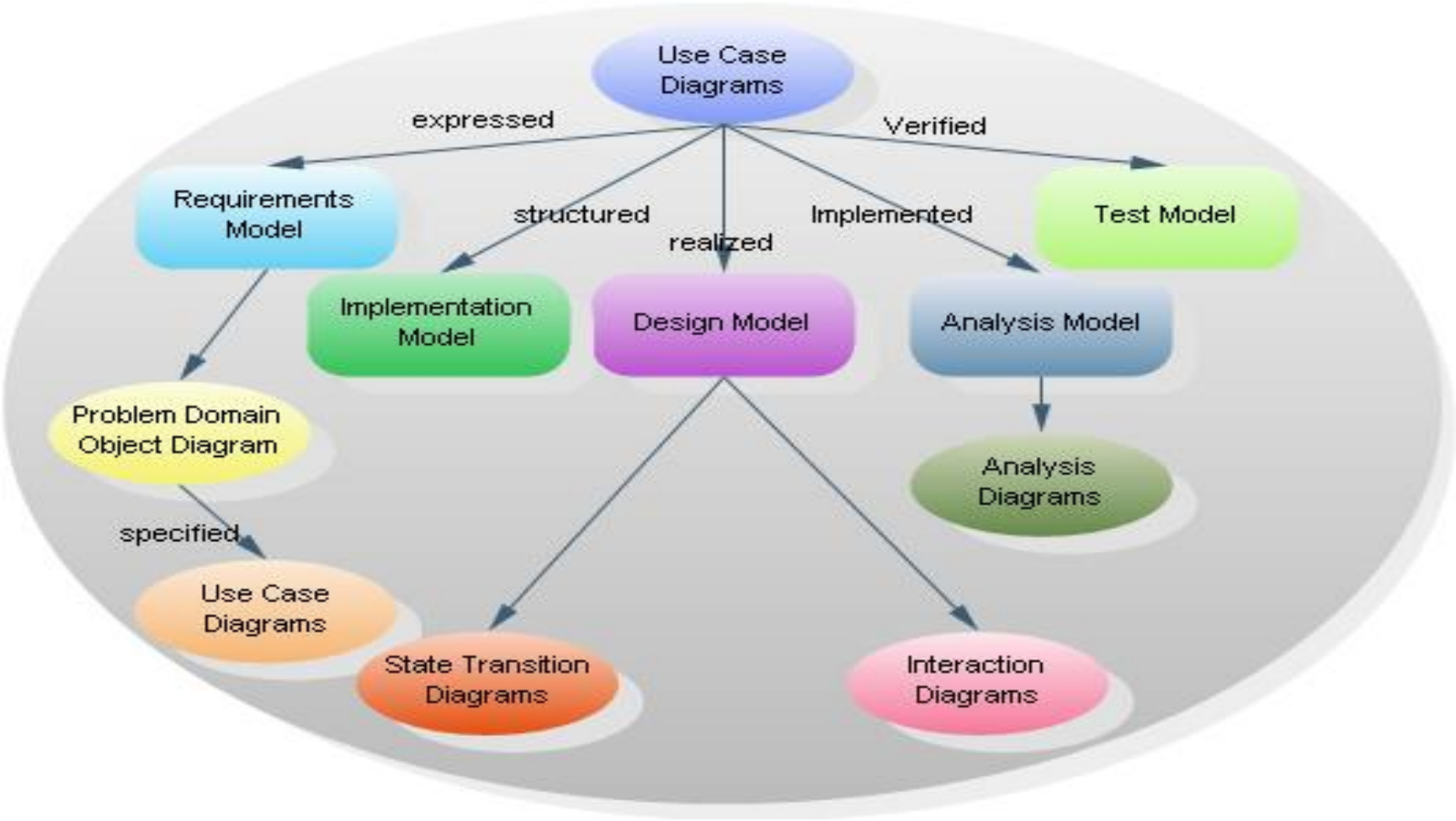
Introduction to Software Engineering

Session III

Kiran waghmare

The SDLC Model



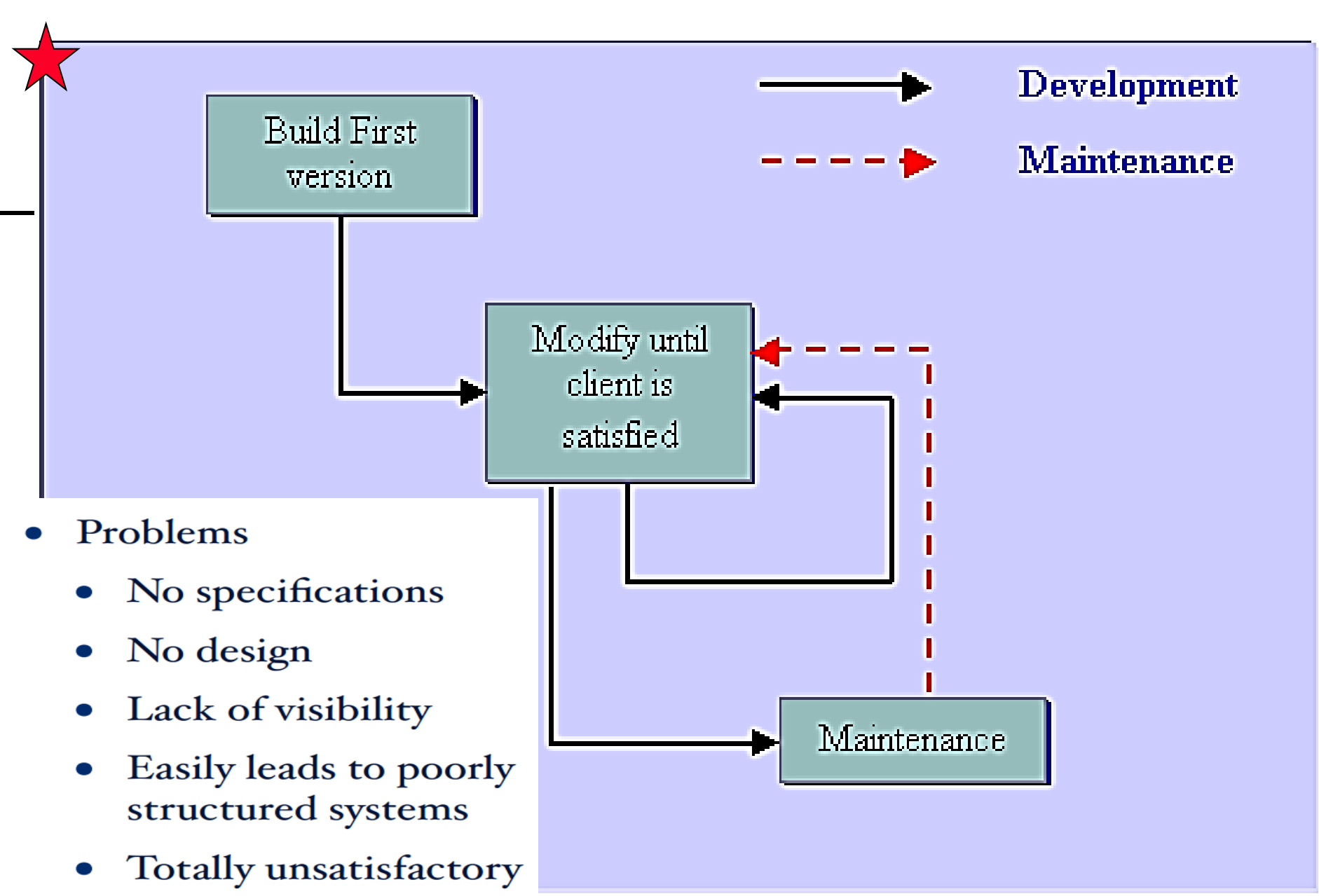
Software life cycle Models

1. Build and Fix Model
2. Waterfall Model
3. Rapid Prototyping Model
4. Incremental Model
5. Extreme Programming
6. Synchronize and Stabilize Model
7. Spiral Model
8. Object Oriented Life Cycle Model

Build and Fix Model

Build and Fix Model

- This is **short project** model.
- This is known as **Ad-hoc Model**.
- It does **not require specific requirements or design.**
- **Lack of** visibility.
- **Easily lead** to poorly structured systems.
- **Totally unsatisfactory** - The software is build and fix for errors until it satisfies the client's requirements.



- Problems
  - No specifications
  - No design
  - Lack of visibility
  - Easily leads to poorly structured systems
  - Totally unsatisfactory
  - Need life-cycle model

..1 - Build-And-Fix Model

Build and Fix Model

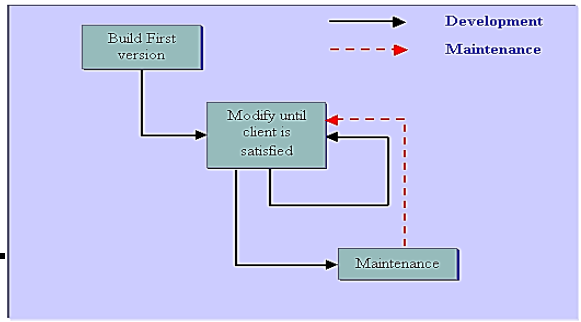


Figure 1.1 - Build-And-Fix Model

- This model includes the following two phases.
- **Build:**
  - In this phase, the **software code is developed and passed** on to the next phase.
- **Fix:**
  - In this phase, the code developed in the build phase is **made error free**.
  - Also, in addition to the corrections to the code, the code is **modified according to the user's requirements.**

Advantage:

1. Suitable for **small projects/software**.
2. **Requires less experience** to execute or manage other than the ability to program.
3. Requires **less project planning**

Disadvantage:

1. It is **not used for large projects**.
2. The **risk factors are not** considered.
3. **Cost is quiet high** as it requires rework until user's requirements are accomplished.
4. Informal design of the software as it **involves unplanned procedure.**
5. It is **difficult** to maintain.

Waterfall Model

Waterfall Model

- It is the **most common and classic** life cycle model, which is introduce by **Winston Royce** in 1970.

- Used for **large term projects**.
- Most **widely used** Software Development Process.
- Also called as **Linear Sequential Model** or **Classic Life Cycle** or **Iterative Model.**

- The model states that the phases are organized in a **Linear Order. i.e.,**
  - the output of one phase becomes the input for the next phase. Various phases have already been explained under a general model of system development.

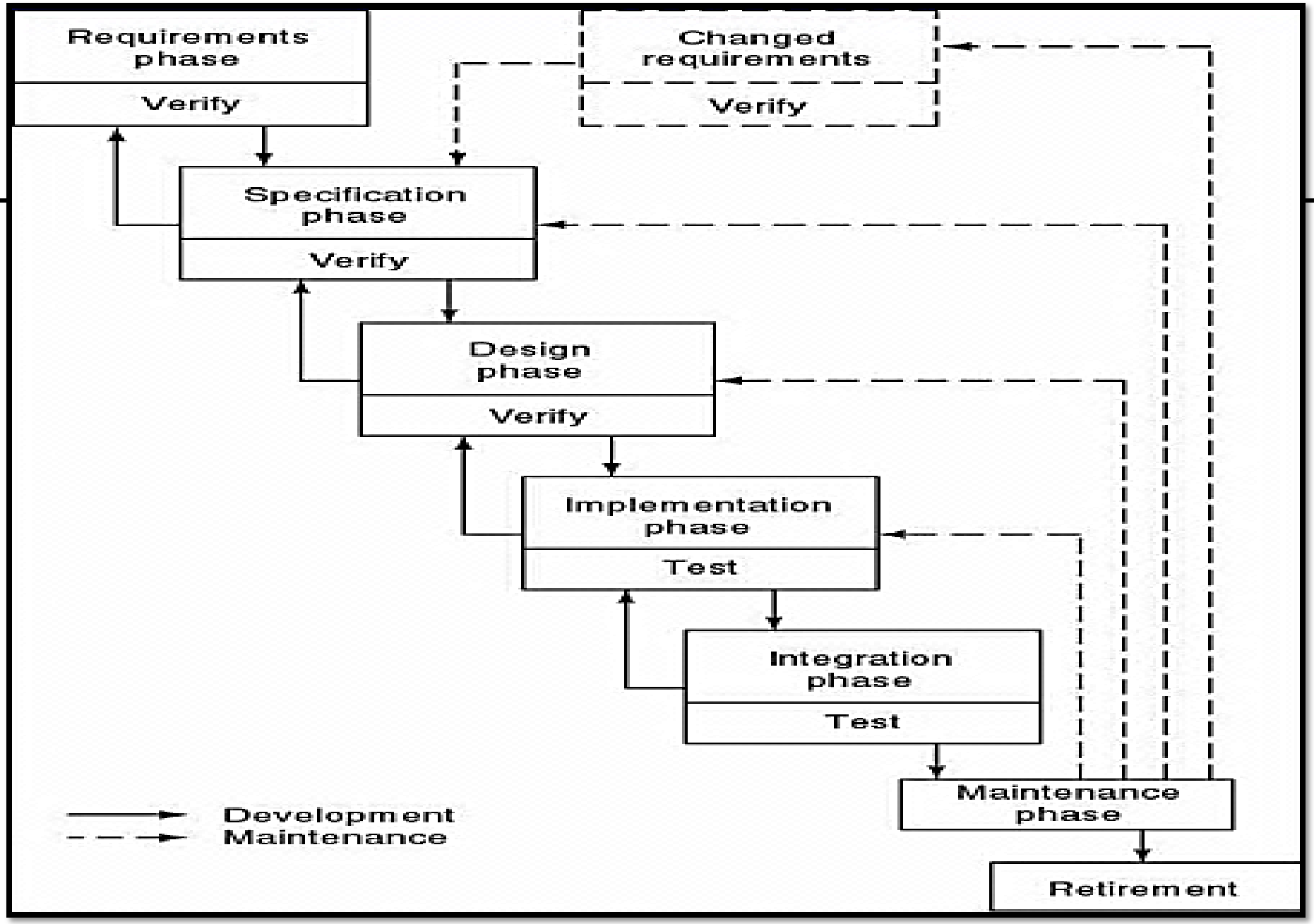


Fig. Waterfall Model

Diagram of Waterfall-model:

- Planning and control
- Documentation-driven
- "Doing the homework"
- Formal change management

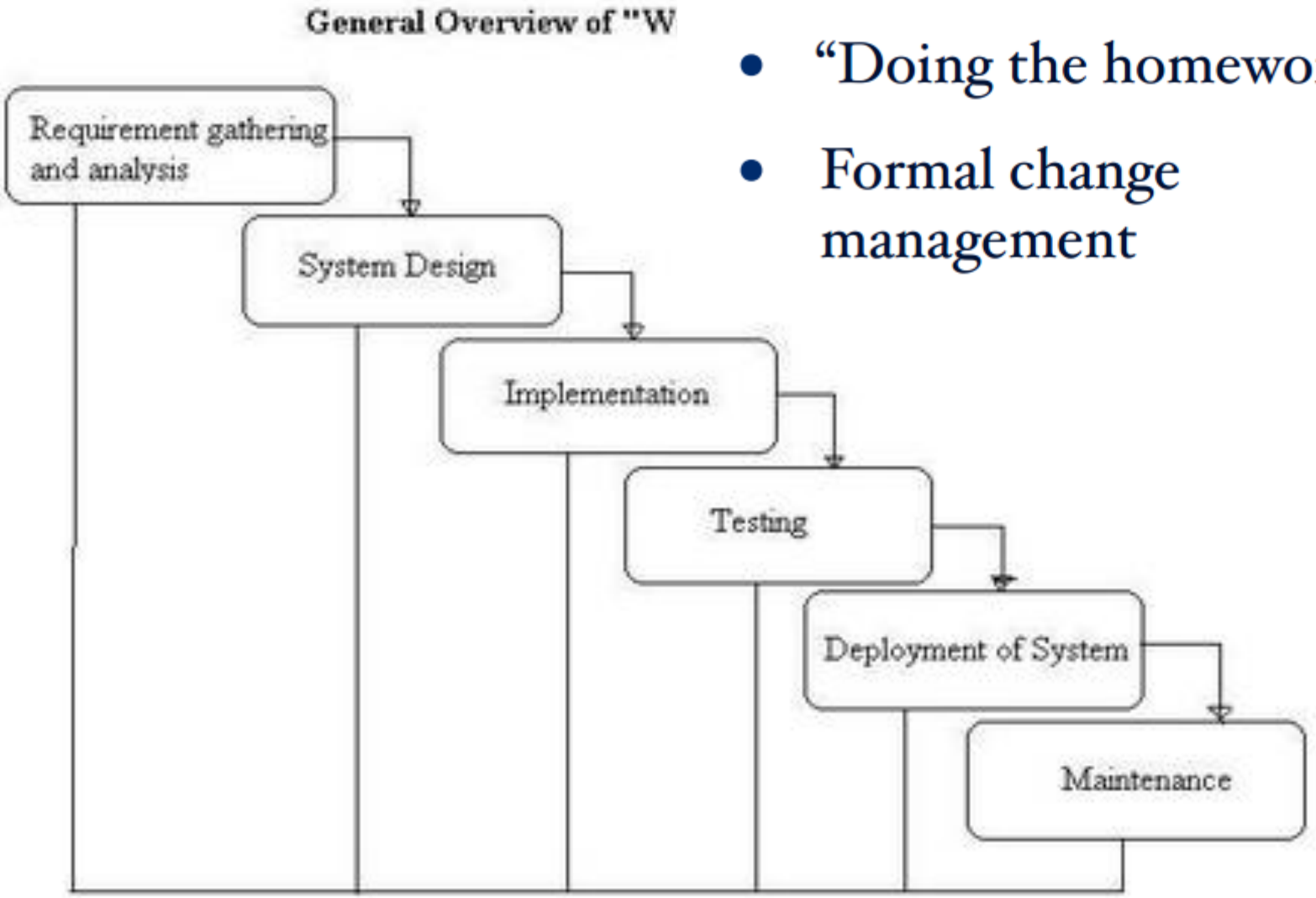


Fig. Waterfall Model

- Strengths
  - Easily manageable process (manager's favourite?)
  - Probably the most effective model, if you know the requirements
  - Extensive documentation
- Weaknesses
  - Inflexible partitioning of the project into distinct phases
  - Difficult to respond to changing customer requirements
  - Feedback on system performance available very late and changes can be very expensive
- Applicability
  - Appropriate when the requirements are well understood
  - Short, clearly definable projects (e.g. maintenance)
  - Very large, complex system development that requires extensive documentation. Safety critical systems.

Advantages of waterfall model:

1. **Easy** to understand, easy to use.
2. **Provides structure** to inexperienced staff.
3. Milestone are well **understood.**
4. Sets requirements **stability.**
5. **Good** for management control (plan, staff and track)
6. Works well when **quality is more important** than cost or schedule.

Disadvantages of waterfall model:

1. All requirements must be **known upfront**.
2. Deliverables created for each phase are considered **frozen – inhibits flexibility.**
3. Can give a **false impressions** of progress.
4. **Does not reflect problem-solving** nature of software development – iterations of phases.
5. Integration is a **big bang** at the end.
6. **Little opportunity** for customer to preview the system(until it may be too late).

When to use the waterfall model:

1. Requirements are very **well known, clear and fixed**.
2. Product definition **is stable**.
3. Technology is **understood**.
4. There are **no ambiguous** requirements
5. **Ample resources** with required expertise are available freely
6. The project is **short.**

Rapid Prototyping Model

Rapid Prototyping Model

- The rapid application development model emphasizes on **delivering projects in small pieces**.
- If the project is large, it is divided into a **series of smaller projects.**
- Each of these smaller projects is planned and delivered **individually.**
- Thus, with a series of smaller projects, the final project is **delivered quickly and in a less structured manner.**
- The major characteristic of the RAD model is that it focuses on the **reuse of code, processes, templates, and tools.**

- Linear
- "Rapid"
- Exploratory vs. throw-away prototypes

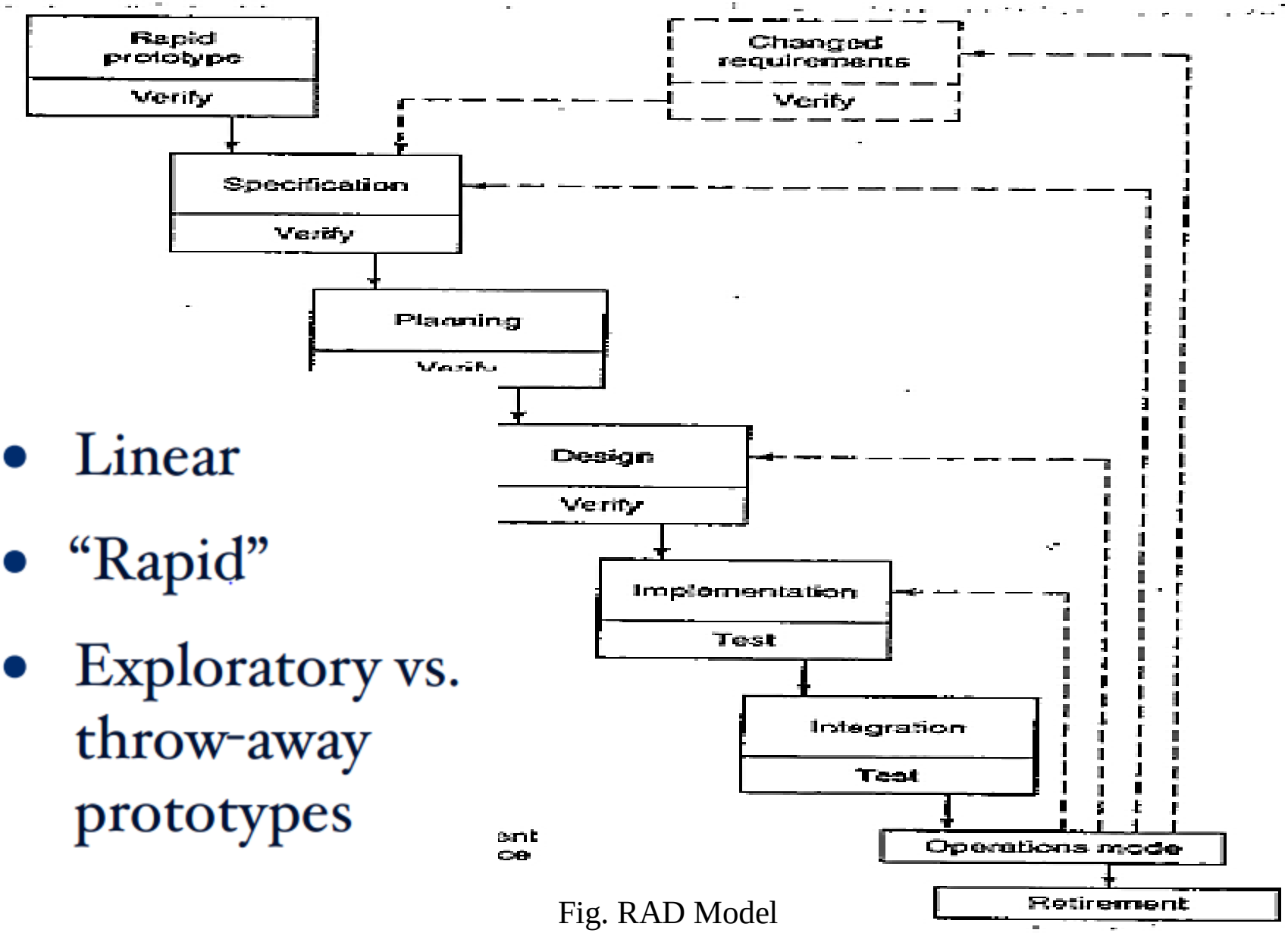


Fig. RAD Model

Advantages of Rapid Prototype

1. Reduce cycle time and improved productivity with fewer people means **lower costs.**

2. Time-box approach mitigates **cost and schedule risk.**

3. Customer involved throughout the complete cycle **minimizes risk of** not achieving customer satisfaction and business needs.

4. **Focus** moves from documentation to code (WYSIWYG)

5. **Uses modeling concept** to capture information about business, data and processes.

Disadvantages of Rapid Prototype

1. **Accelerated development** process must give quick responses to the user.
2. **Risk** of never achieving closure.
3. Had to **use with legacy** systems.
4. Requires a system that can be **modularized.**
5. Developers and customers must be committed **to rapid-fire activities** in an abbreviated time frame.

Incremental Model

Incremental Model

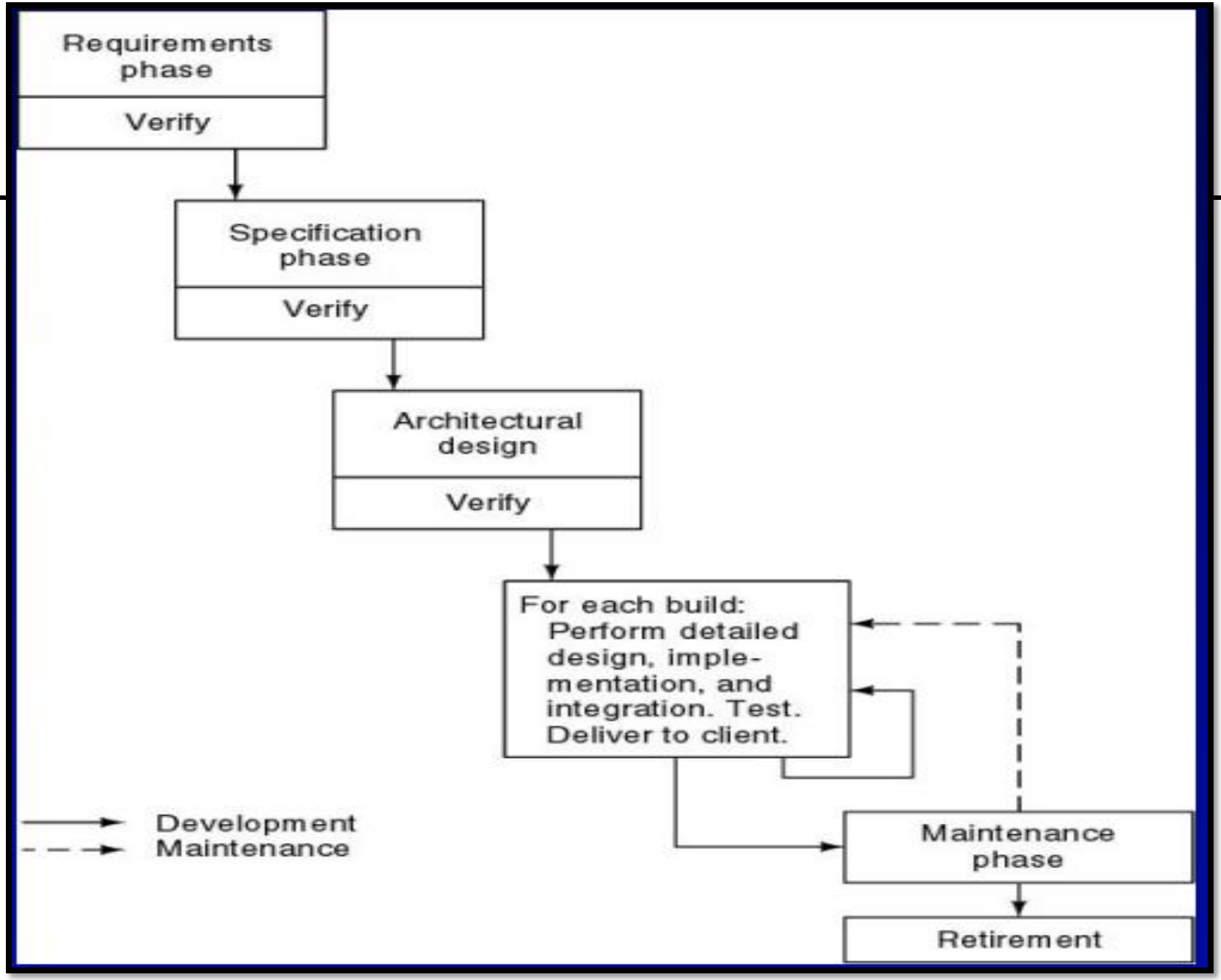
For example:

In the diagram above when we work **incrementally** we are adding piece by piece but expect that each piece is fully finished. Thus keep on adding the pieces until it's complete.

Incremental Model

- **Divide** project into builds.
- Each build **adds functionality**.
- **Prioritized** user requirements.
- Requirements **frozen** during each build !.

- also known as **iterative enhancement model** comprises the features of waterfall model in an iterative manner.


Requirements phase
Verify
Specification phase
Verify
Architectural design
Verify
For each build:
Perform detailed design, imple-mentation, and integration. Test. Deliver to client.
Development
Maintenance
Maintenance phase
Retirement

Incremental Model

- This model comprises **several phases** where each phase produces an increment.
- These increments are identified in the **beginning** of the development process and the entire process from requirements gathering to delivery of the product is carried out for each increment.
- The basic idea of this model **is to start the process with requirements and iteratively enhance the requirements until the final software is implemented**.
- This **approach is useful** as it **simplifies the software development** process as implementation of smaller increments is easier than implementing the entire system.

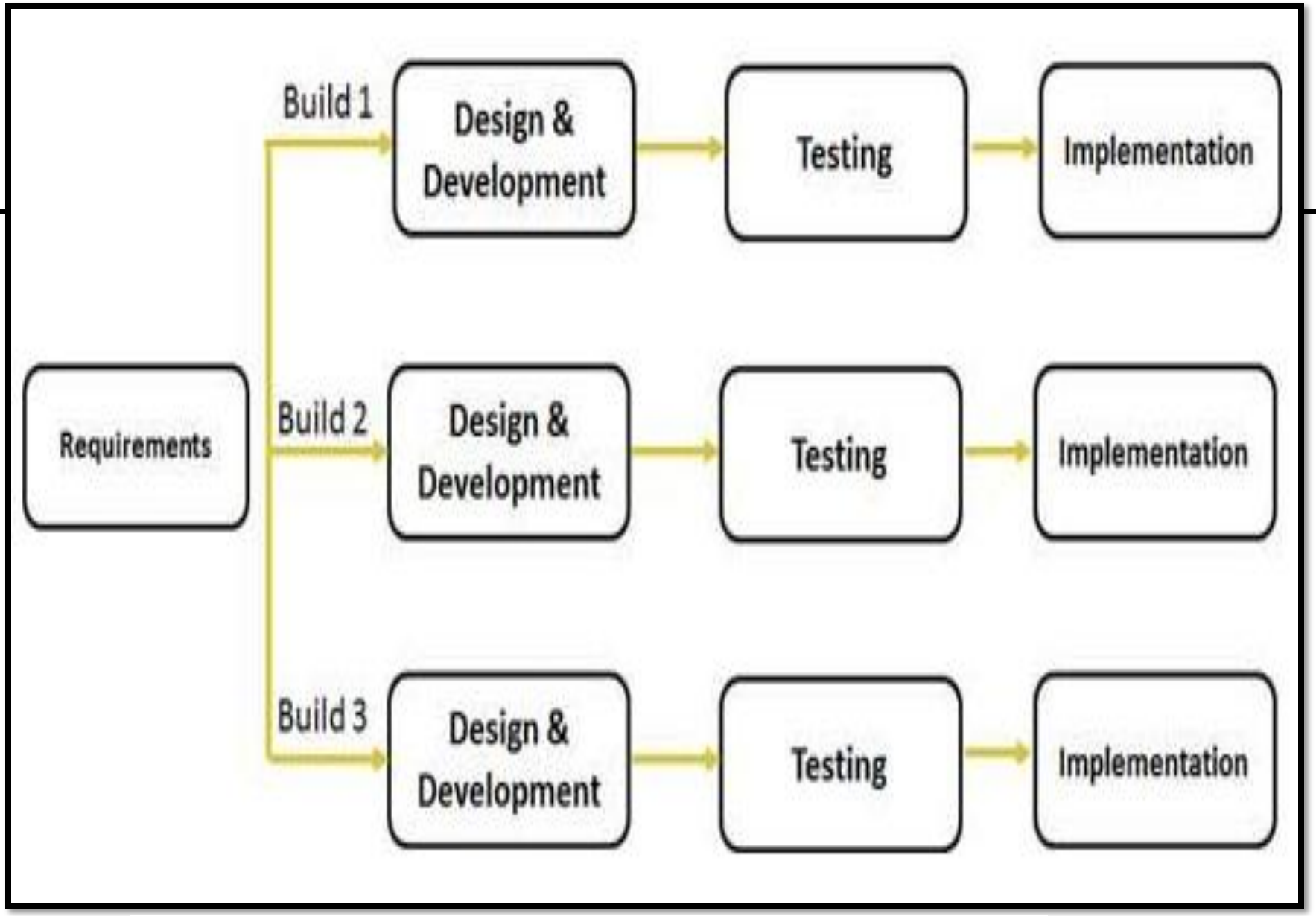


Fig. Incremental Model

- Each stage of incremental **model adds some functionality** to the product and passes it on to the next stage.
- The first increment is generally known as a **core product** and is used by the user for a detailed evaluation.

- This process **results in creation of a plan** for the next increment.
- This plan **determines the modifications** (features or functions) of the product in order to accomplish user requirements.

- The iteration process, which includes the **delivery of the increments** to the user, continues until the software is completely developed.

Advantages of Incremental model:

1. Avoids the problems resulting in **risk driven approach** in the software
2. **Understanding increases** through successive refinements.
3. Performs **cost-benefit analysis** before enhancing software with capabilities
4. Incrementally grows **in effective solution** after every iteration
5. Results are obtained **early and periodically.**
6. **Less costly** to change the scope / requirements.
7. Testing and debugging during smaller iteration is **easy.**
8. Easier to **manage risk -** High risk part is done first.
9. With every increment operational **product is delivered**.
10. Initial **Operating time is less.**

Disadvantages of Incremental model:

1. **Requires planning** at the management and technical level
2. **More resources** may be required.
3. Although **cost of change is lesser** but it is not very suitable for changing requirements.
4. More management **attention** is required.
5. **Not suitable for smaller** projects.
6. Management **complexity** is more.
7. End of project may not be known which is a **risk.**
8. Highly **skilled resources are required** for risk analysis.

When to use the Incremental model:

1. Requirements of the complete system are **clearly defined** and understood.
2. **Major requirements** must be defined; however, some details can evolve with time.
3. There is a **need to get a product to the market** early.
4. A **new technology** is being used
5. Resources with **needed skill set** are not available
6. There are some **high risk features** and goals.

Advantages of Incremental model:

1. Avoids the problems resulting in **risk driven approach** in the software
2. **Understanding increases** through successive refinements.
3. Performs **cost-benefit analysis** before enhancing software with capabilities
4. Incrementally grows **in effective solution** after every iteration
5. Results are obtained **early and periodically.**
6. **Less costly** to change the scope / requirements.
7. Testing and debugging during smaller iteration is **easy.**
8. Easier to **manage risk -** High risk part is done first.
9. With every increment operational **product is delivered**.
10. Initial **Operating time is less.**

Disadvantages of Incremental model:

1. **Requires planning** at the management and technical level
2. **More resources** may be required.
3. Although **cost of change is lesser** but it is not very suitable for changing requirements.
4. More management **attention** is required.
5. **Not suitable for smaller** projects.
6. Management **complexity** is more.
7. End of project may not be known which is a **risk.**
8. Highly **skilled resources are required** for risk analysis.

When to use the Incremental model:

1. Requirements of the complete system are **clearly defined** and understood.
2. **Major requirements** must be defined; however, some details can evolve with time.
3. There is a **need to get a product to the market** early.
4. A **new technology** is being used
5. Resources with **needed skill set** are not available
6. There are some **high risk features** and goals.