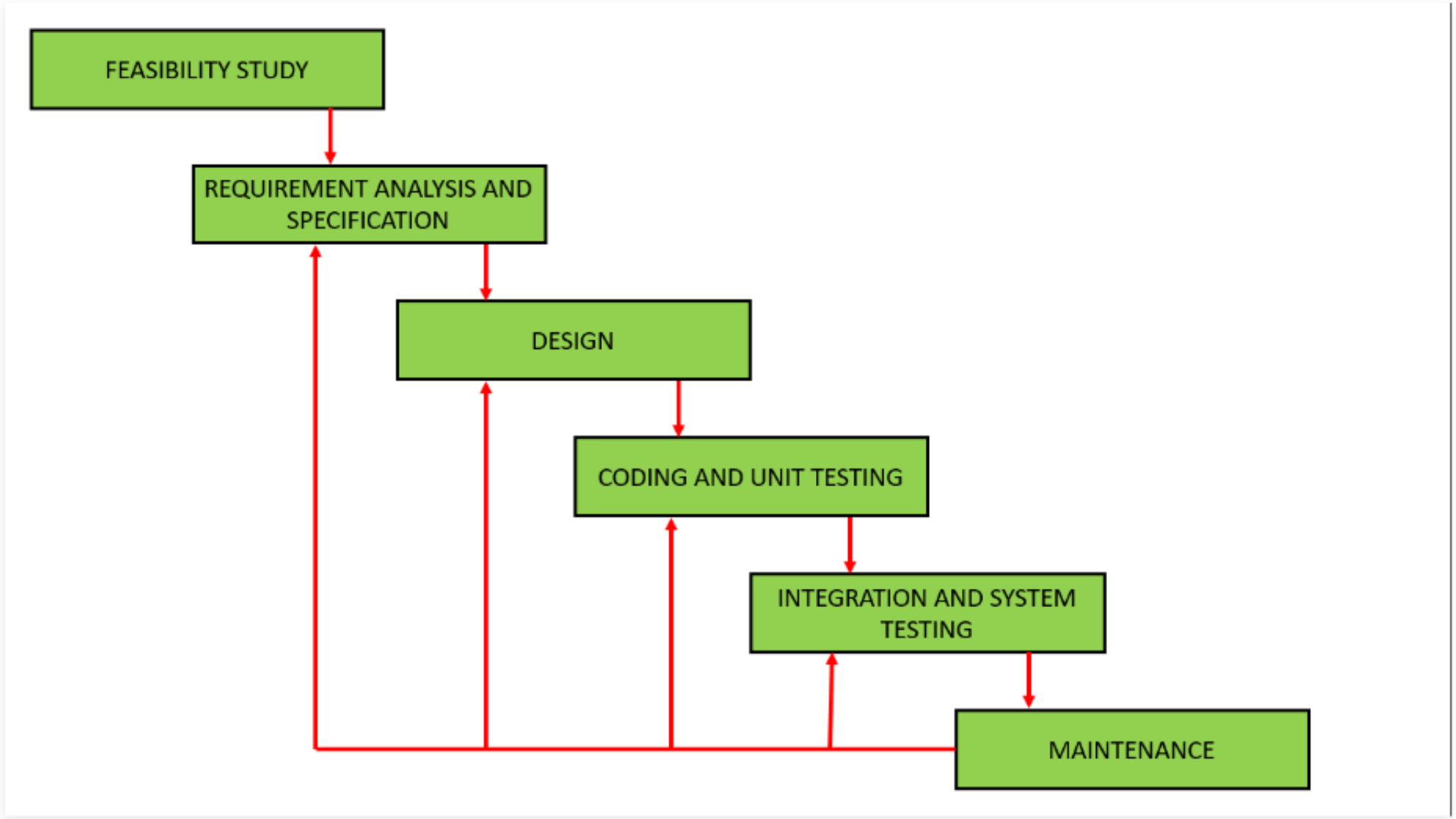
ITERATIVE WATERFALL MODEL

- In a practical software development project, the classical waterfall model is hard to use. So, the Iterative waterfall model can be thought of as incorporating the necessary changes to the classical waterfall model to make it usable in practical software development projects. It is almost the same as the classical waterfall model except some changes are made to increase the efficiency of the software development.

- The iterative waterfall model provides feedback paths from every phase to its preceding phases, which is the main difference from the classical waterfall model.

- Feedback paths introduced by the iterative waterfall model are shown in the figure below.


FEASIBILITY STUDY
REQUIREMENT ANALYSIS AND SPECIFICATION
DESIGN
CODING AND UNIT TESTING
INTEGRATION AND SYSTEM TESTING
MAINTENANCE

- When errors are detected at some later phase, these feedback paths allow correcting errors committed by programmers during some phase.

- The feedback paths allow the phase to be reworked in which errors are committed and these changes are reflected in the later phases.
-
- But, there is no feedback path to the stage – feasibility study, because once a project has been taken, does not give up the project easily.

- It is good to detect errors in the same phase in which they are committed. It reduces the effort and time required to correct the errors.

Advantages of Iterative Waterfall Model :

- **Feedback Path –**
  In the classical waterfall model, there are no feedback paths, so there is no mechanism for error correction. But in the iterative waterfall model feedback path from one phase to its preceding phase allows correcting the errors that are committed and these changes are reflected in the later phases.
- **Simple –**
  Iterative waterfall model is very simple to understand and use. That's why it is one of the most widely used software development models.
- **Cost-Effective –**
  It is highly cost-effective to change the plan or requirements in the model. Moreover, it is best suited for agile organizations.
- **Well-organized –**
  In this model, less time is consumed on documenting and the team can spend more time on development and designing.

- Drawbacks of Iterative Waterfall Model :

- Difficult to incorporate change requests –
  The major drawback of the iterative waterfall model is that all the requirements must be clearly stated before starting the development phase. Customers may change requirements after some time but the iterative waterfall model does not leave any scope to incorporate change requests that are made after the development phase starts.

- Incremental delivery not supported –
  In the iterative waterfall model, the full software is completely developed and tested before delivery to the customer. There is no scope for any intermediate delivery. So, customers have to wait a long for getting the software.

- Overlapping of phases not supported –
  Iterative waterfall model assumes that one phase can start after completion of the previous phase, But in real projects, phases may overlap to reduce the effort and time needed to complete the project.

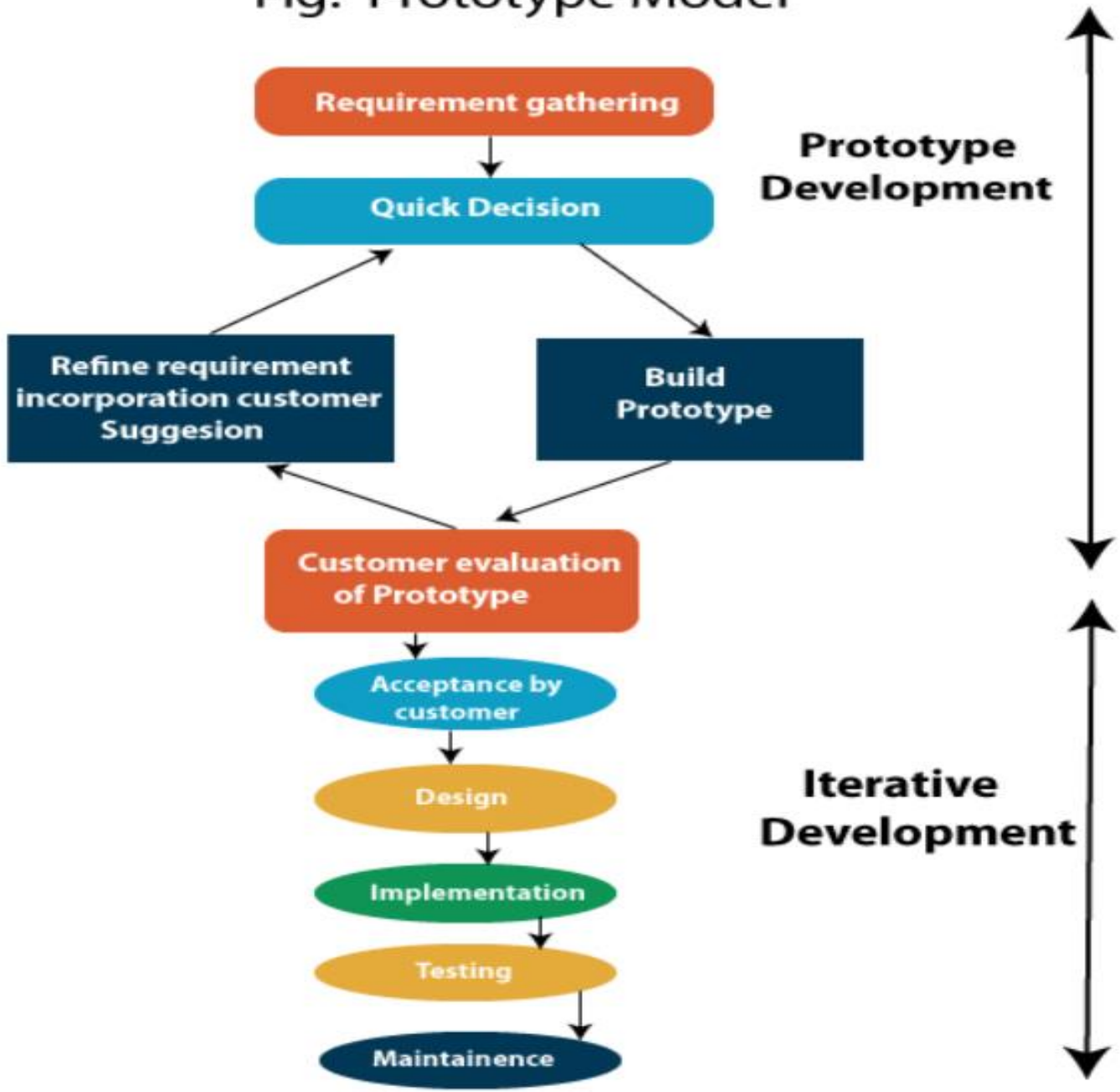
- Risk handling not supported –
  Projects may suffer from various types of risks. But, the Iterative waterfall model has no mechanism for risk handling.

- Limited customer interactions –
  Customer interaction occurs at the start of the project at the time of requirement gathering and at project completion at the time of software delivery. These fewer interactions with the customers may lead to many problems as the finally developed software may differ from the customers' actual requirements.

PROTYPE 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

1. Requirement Gathering and Analyst
2. Quick Decision
3. Build a Prototype
4. Assessment or User Evaluation
5. Prototype Refinement
6. Engineer Product

- Advantage of Prototype Model

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

- Disadvantage of Prototype Model

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