Basic Software Engineering

Unit – 5 System Analysis Models

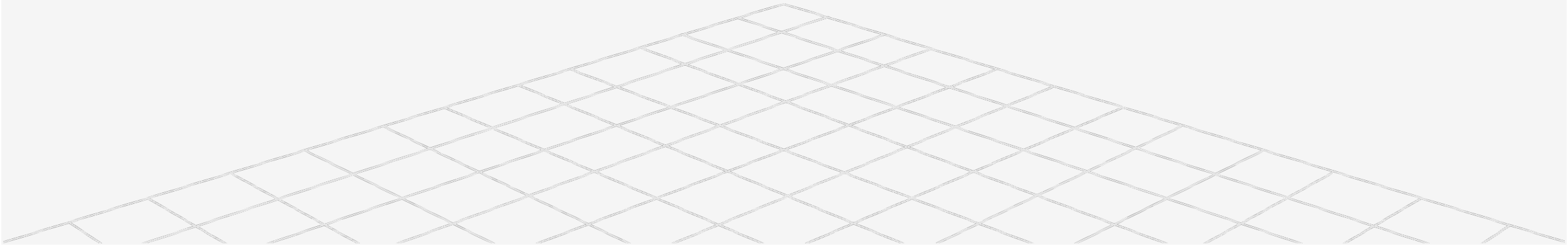
By: Vibhuti Patel

PICA – BCA

Generic Process Models

Software Process Frameworks

- Each software engineering action is defined by a ***task set that identifies the work tasks*** that are to be completed,
- ***the work products*** that will be produced,
- ***the quality assurance points*** that will be required,
- ***the milestones*** that will be used to indicate progress.

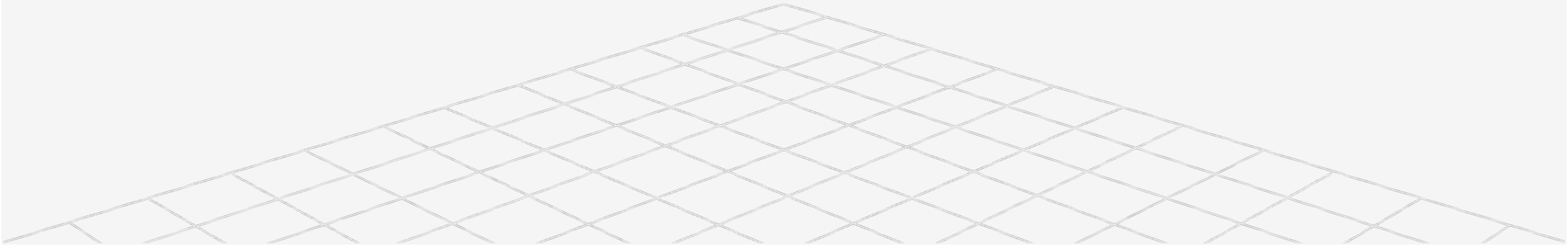


Perspective Model

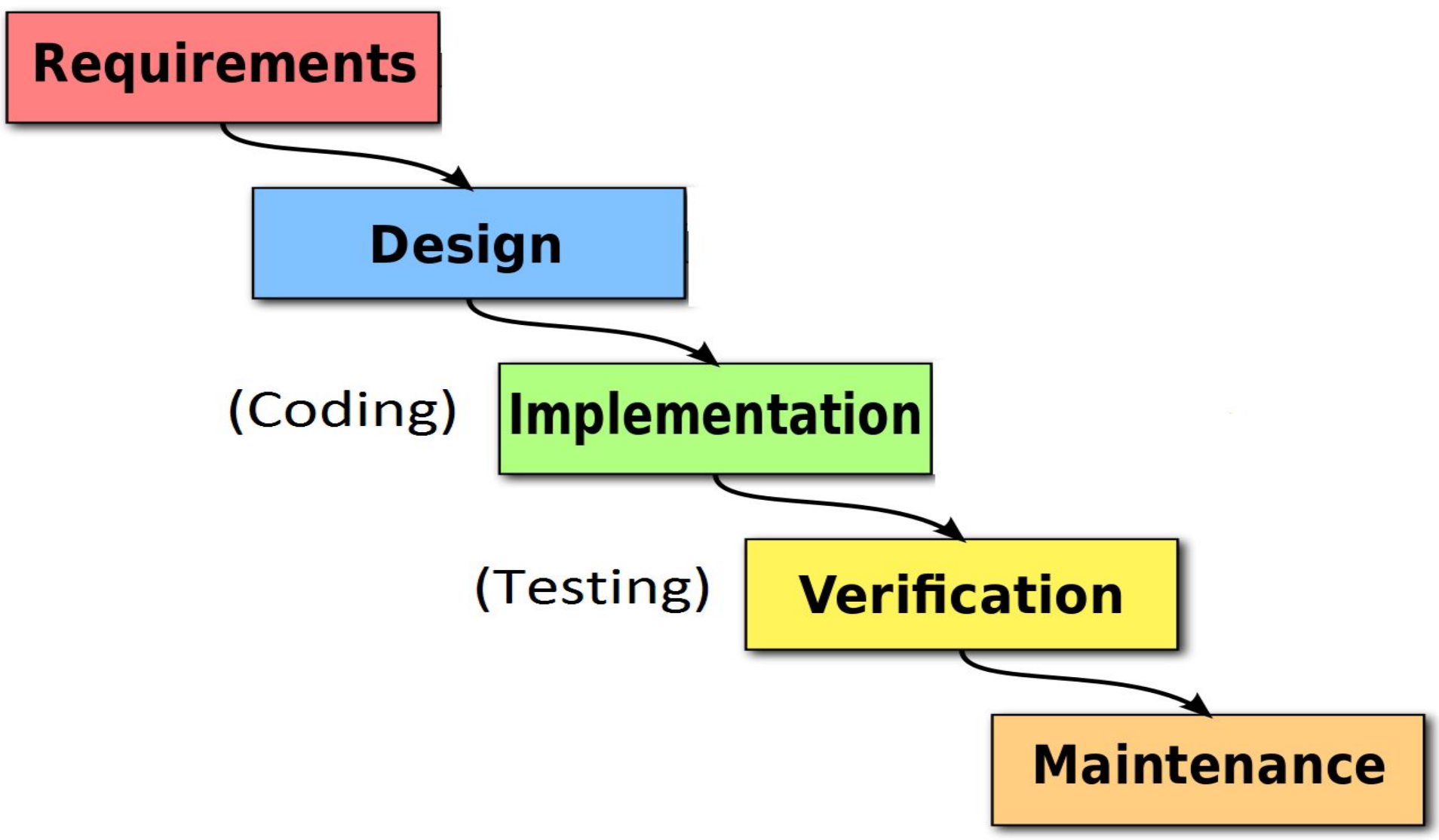
- A software **process model** is a simplified representation of a software **process**. Each **model** represents a **process** from a specific **perspective**.
- It prescribes
  - set of process elements
  - activities
  - actions
  - Tasks
  - work products
  - quality assurance

Water Fall Model

- The Waterfall Model was the first Process Model to be introduced.
- It is also known to as a **linear-sequential 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.



Water Fall Model



Water Fall Model

- **Requirement Gathering & analysis:** All possible requirements of the system to be developed are captured in this phase & documented in a requirement specification doc.

- **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

- **Implementation or Coding:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase.

- **Verification or Testing:** Each unit is developed and tested for its functionality which is referred to as Unit Testing.

- **Deployment & Maintenance:** Once testing done, software is deployed in customer environment or release in market. There are some issues which come up. To fix those issues patches are released. Maintenance is done to deliver these changes in customer environment or in market.

Water Fall Model

Advantages:

- The model suggests that software engineers should work in a series of stages
- Before completing each stage, they should perform quality assurance (testing)

Disadvantages:

- The disadvantage of waterfall development is that it does not allow for much reflection or revision
- Once a software is in the testing stage, it is very difficult to go back and change
- Clients cannot use software until when entire system is complete

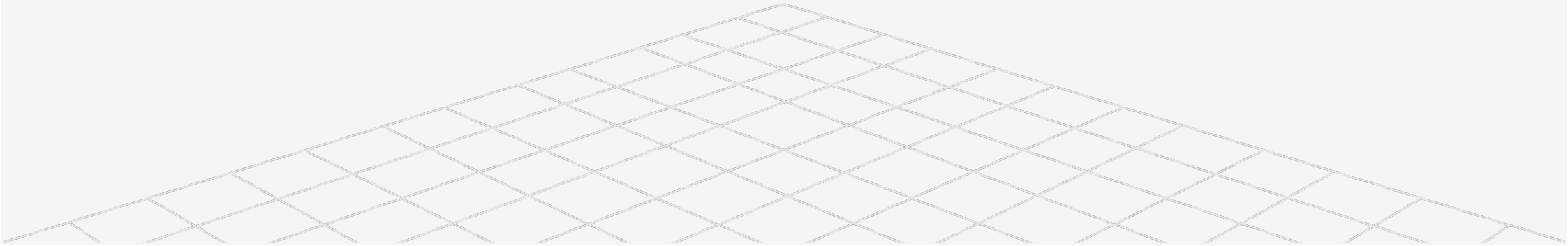
Water Fall Model

When to use?

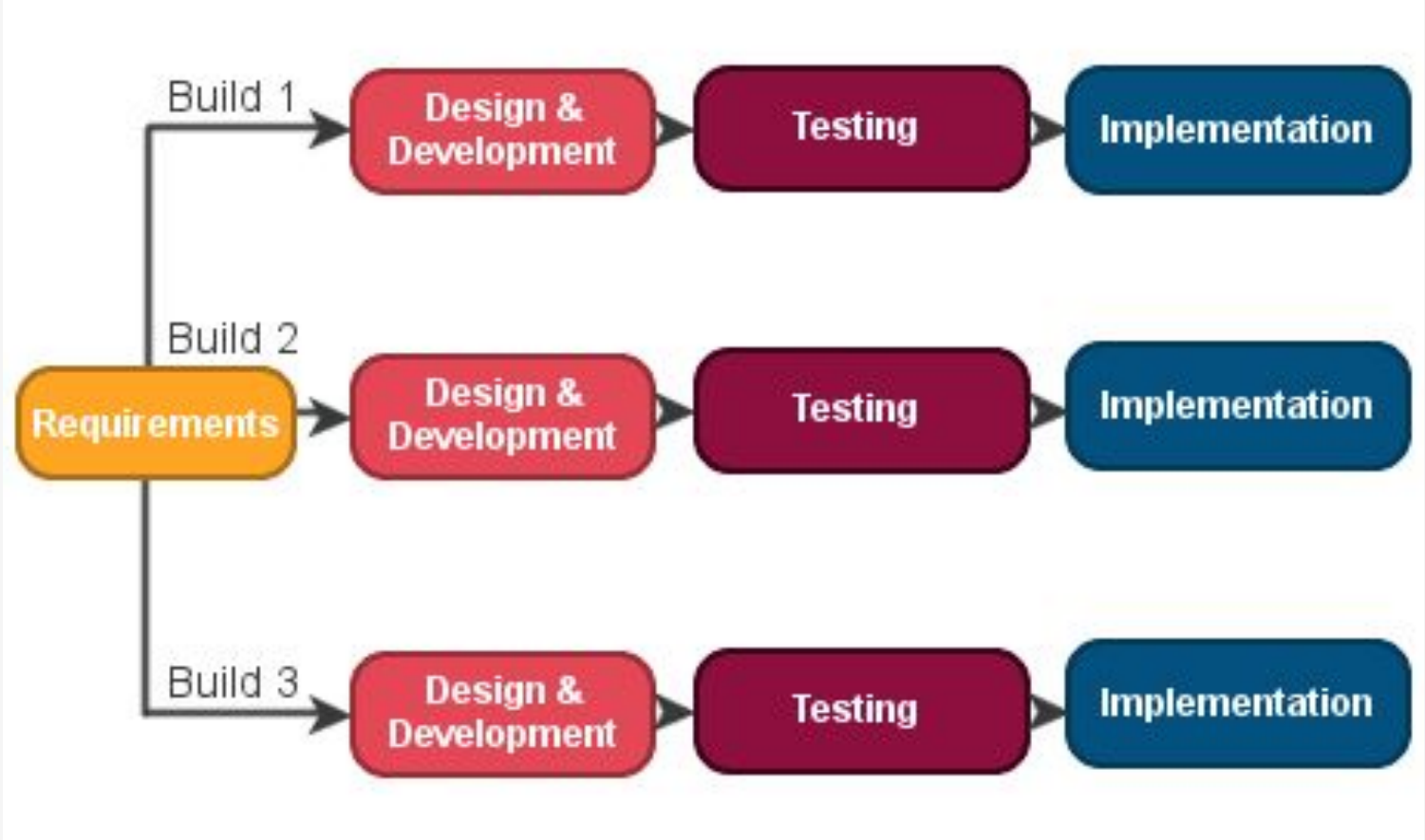
- Requirements of the complete system are clearly defined and understood
- Project definition is stable
- There is no need to get a product to the market early
- Technology is understood & is not dynamic
- Project is short
- Major design problems may not be detected till very late

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 Model



Iterative Model

- Iterative & Incremental development is a combination of both iterative design or method and incremental 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" or "incremental" approach.“
- In this model, the whole requirement is divided into various builds.
- During each iteration, the development module goes through the requirements, design, implementation & testing phases.
- Each subsequent release of the module adds function to the previous release.
- The process continues till the complete system is ready as per the requirement.

Iterative Model

- Successful use of an iterative software development lifecycle is rigorous validation of requirements, and verification & testing of each version of the software against those requirements within each cycle of the model.

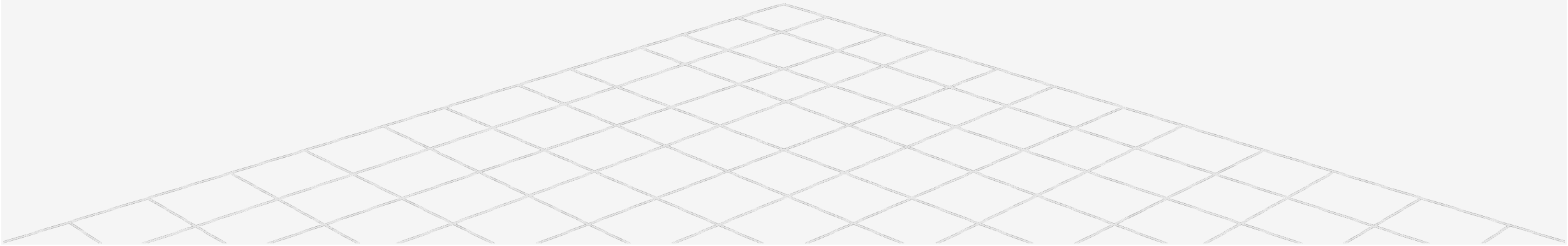
Advantages:

- Some working functionality can be developed quickly and early in the life cycle
- With every increment, operational product is delivered
- It supports changing requirements
- Best for large system
- Testing and debugging during smaller iteration is easy

Iterative Model

Disadvantages:

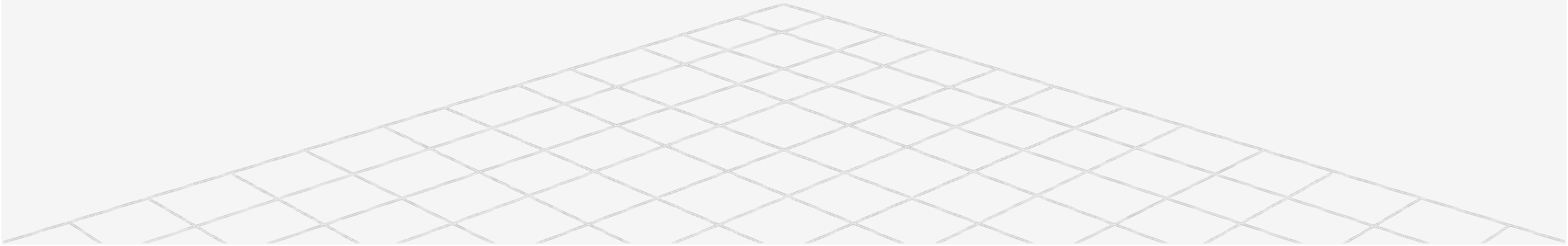
- It is applicable only to large software development projects
- Not suitable to small projects
- More management attention is requires



Iterative Model

When to use?

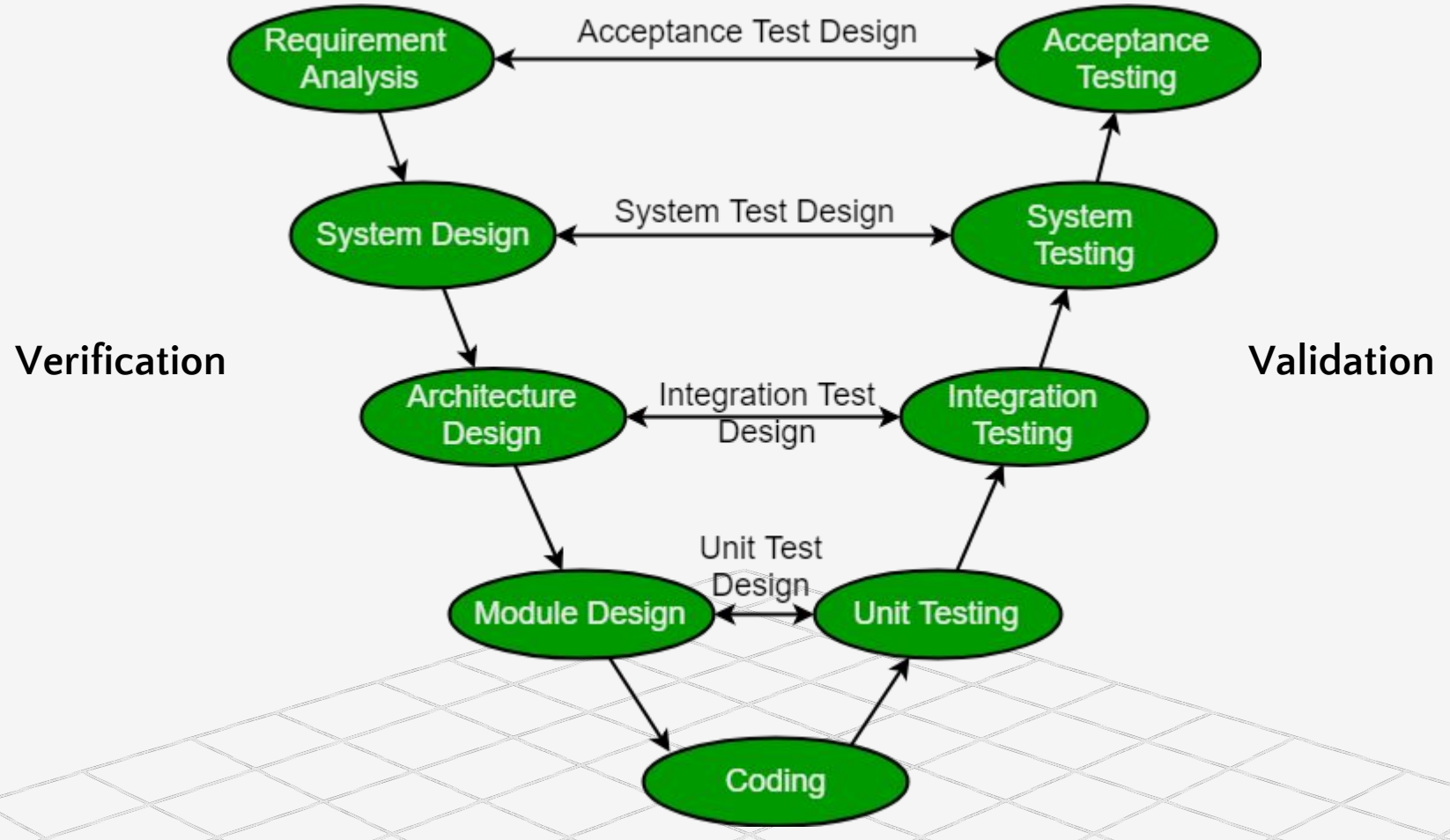
- Requirements of complete system are clearly defined & understood
- Major requirements must be defined
- A new technology is being used & is being learnt by the development team while working on project
- Resources with needed skills set are not available and are planned to be used for specific iterations.



V Model

- A variation in the representation of the waterfall model is called the **V-model.**
- Execution of processes happens in a sequential manner in a **V-shape**.
- It is also known as **Verification and Validation** model.
- The V-Model is based on the association of a testing phase for each corresponding development stage.
- This means that for every single phase in the development cycle, there is a directly associated testing phase.
- This is a highly-disciplined model and the next phase starts only after completion of the previous phase.

V Model


Requirement Analysis
Acceptance Test Design
Acceptance Testing
System Design
System Test Design
System Testing
Verification
Validation
Architecture Design
Integration Test Design
Integration Testing
Unit Test Design
Module Design
Unit Testing
Coding

V Model

Verification Phase

- **Requirement Analysis:** In this first phase software requirements are understood from client perspective. It involves requirements gathering. Acceptance Testing is done at this phase as requirements can be used as an input for testing.

- **System Design:** In this phase understanding and detailing the complete hardware and communication setup for the product under development. The system test plan is developed based on the system design.

- **Architecture Design:** Architectural specifications are understood and designed in this phase. The data transfer and communication between the internal modules and with the outside world (other systems) is clearly understood and defined in this stage. With this information, integration tests can be designed and documented during this stage.

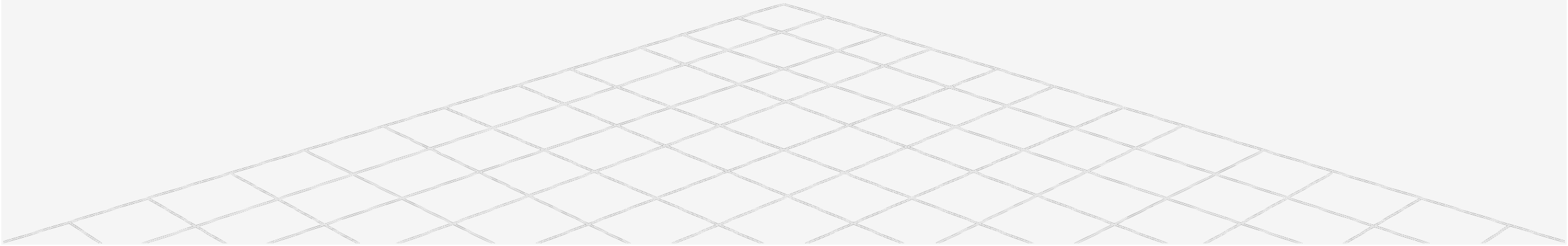
V Model

Verification Phase

- **Module Design:** In this phase, the detailed internal design for all the system modules is specified. It is important that the design is compatible with the other modules in the system architecture and the other external systems. The unit tests are an essential part of any development process and helps eliminate the maximum faults and errors at a very early stage. These unit tests can be designed at this stage based on the internal module designs.

V Model

Coding Phase: The actual coding of the system modules designed in the design phase is taken up in the Coding phase.



V Model

Validation Phase

- **Unit Testing:** It is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed.
- **Integration Testing:** It is the phase in software testing in which individual software modules are combined & tested as a group. It occurs after unit testing & before system testing.
- **System Testing:** System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements.
- **Acceptance Testing:** It is a test conducted to determine if the requirements of a specification or contract are met.

V Model

Advantages

- Phases are completed one at a time
- Good for smaller projects where requirements are very well understood
- Simple & easy
- Each phase has specific deliverables and a review process

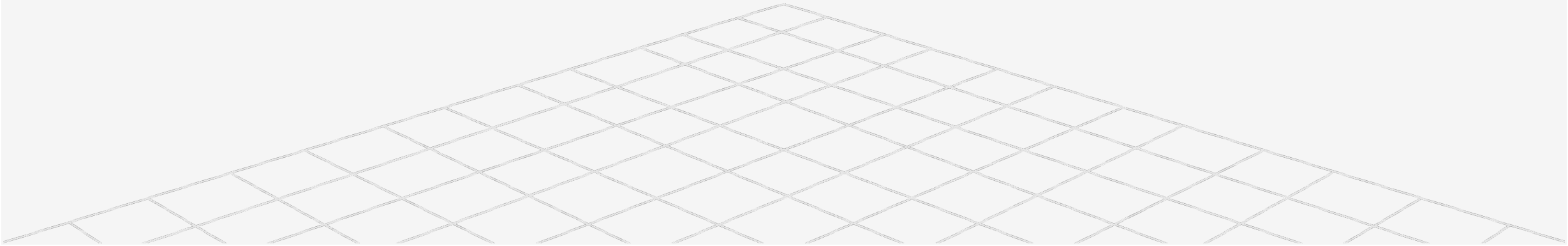
Disadvantages

- Not good for complex & object oriented projects
- Poor model for long & ongoing projects

V Model

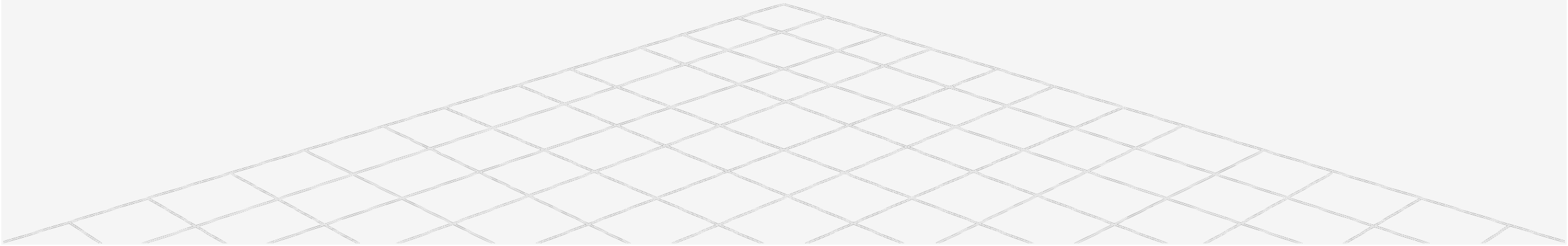
When to use?

- Requirements are well defined, clearly documented and fixed
- Project definition is stable
- Technology is stable and is well understood by the project team
- The project is short



Evolutionary Process Model

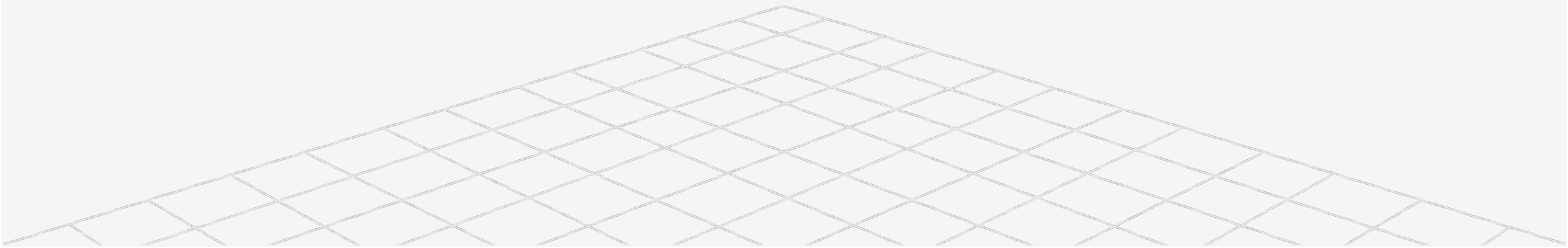
- It produce an increasingly more complete version of the software with each iteration.
- It is iterative.
- Enables software engineers to develop increasingly more complete version of the software.



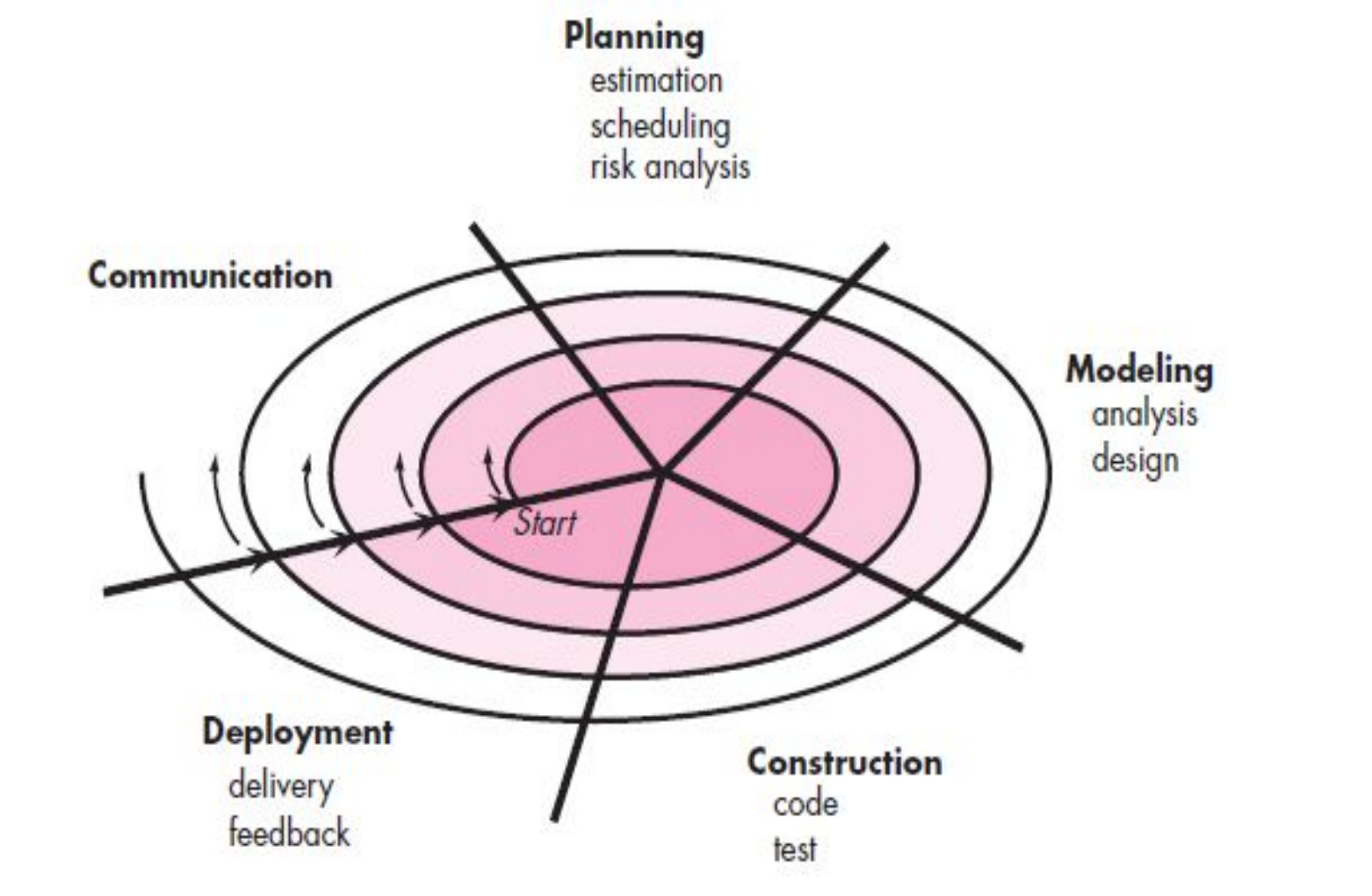
Spiral Model

- This Spiral model is a combination of iterative development process model and sequential linear development model i.e. the waterfall model with a very high emphasis on risk analysis.

- It allows incremental releases of the product or incremental refinement through each iteration.



Spiral Model



Spiral Model

It has two main features.

- One is a **cyclic approach** for incrementally growing a system's degree of definition & implementation while decreasing its degree of risk.
- The other is a set of **anchor point** (center position) milestones for ensuring stakeholder commitment to feasible & mutually satisfactory system solutions.

Spiral Model

Advantages

- Changing requirements can be accommodated
- More detailed processes for each development phase
- Clients see the system early

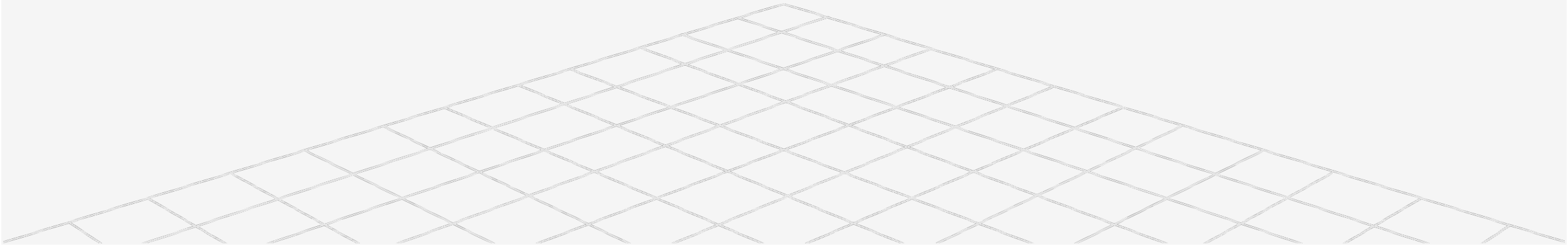
Disadvantages

- Cost is high
- Sometime difficult to implement or too time consuming
- Spiral may go on indefinitely

Spiral Model

When to use?

- Customer is not sure of their requirements which is usually the case
- Long term project commitment

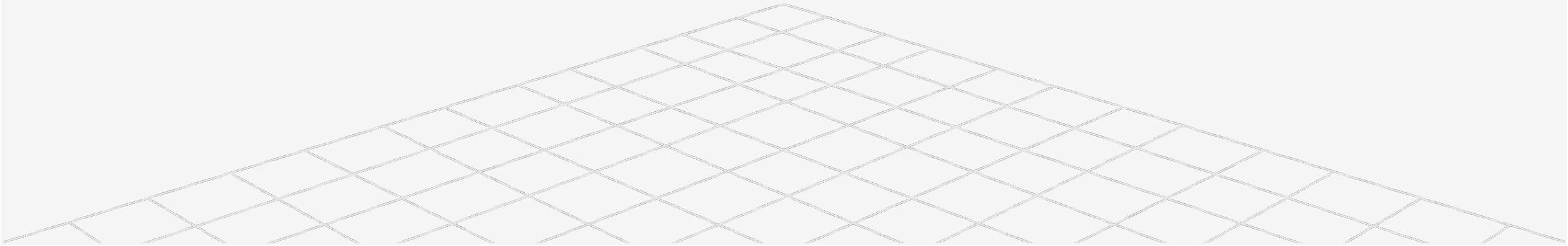


Prototyping Model

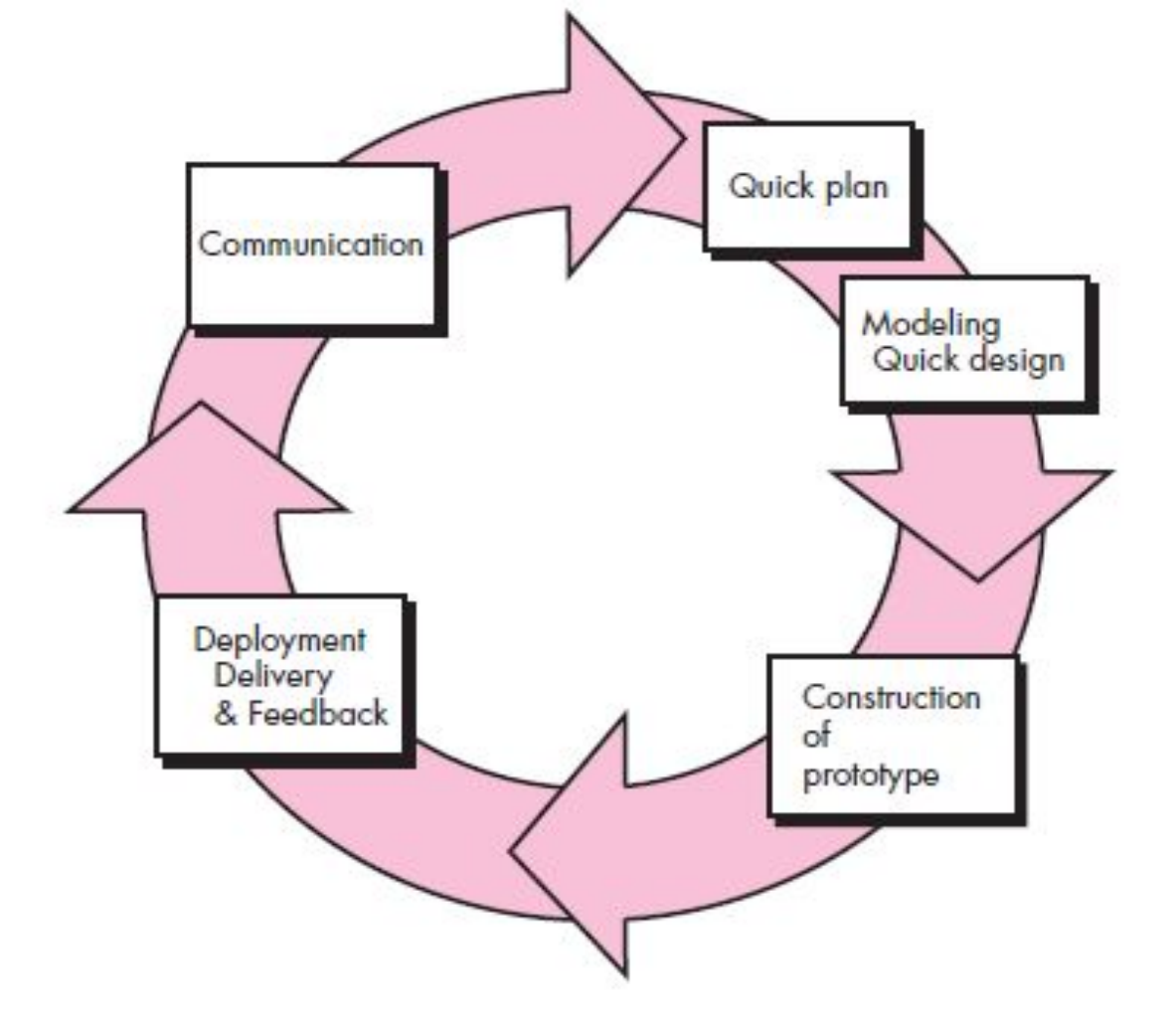
- It is a systems development method (SDM) in which a prototype (an early approximation of a final system) is built, tested, and then reworked as necessary until an acceptable prototype is finally achieved from which the complete system or product can now be developed.

- This model works best in scenarios where not all of the project requirements are known in detail ahead of time.

- It is an iterative, trial-and-error process that takes place between the developers and the users.



Prototyping Model



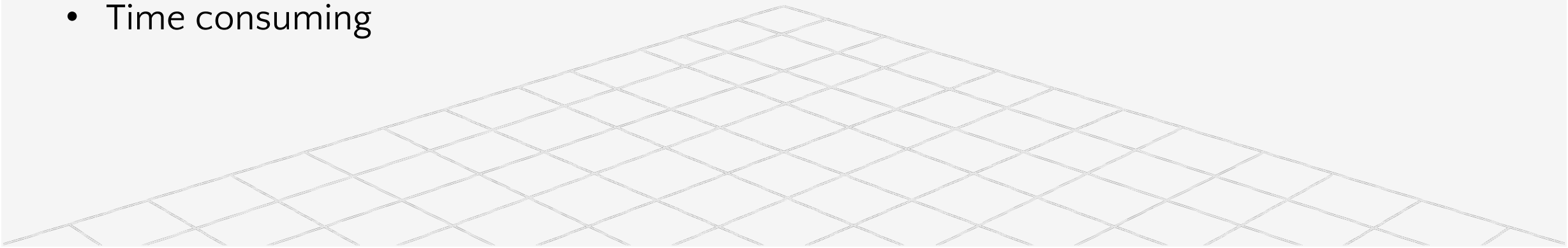
Prototyping Model

Advantages

- Increase user involvement in product even before its implementation
- Missing functionality can be identified
- Fast feedback is available

Disadvantages

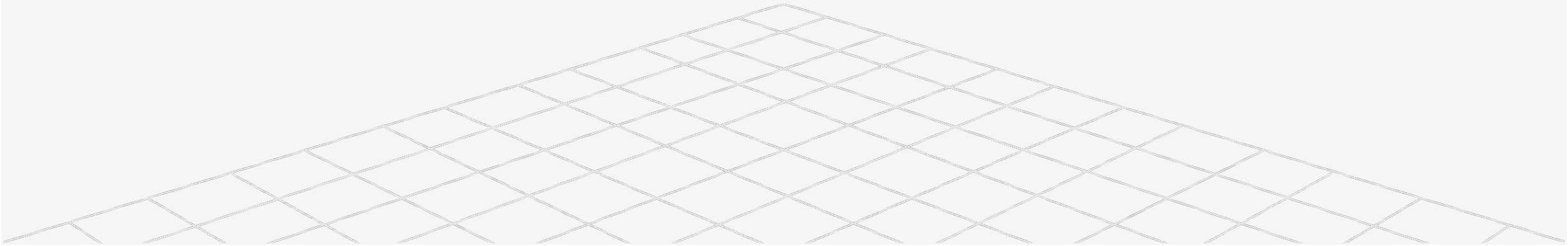
- Increase complexity
- Time consuming



Prototyping Model

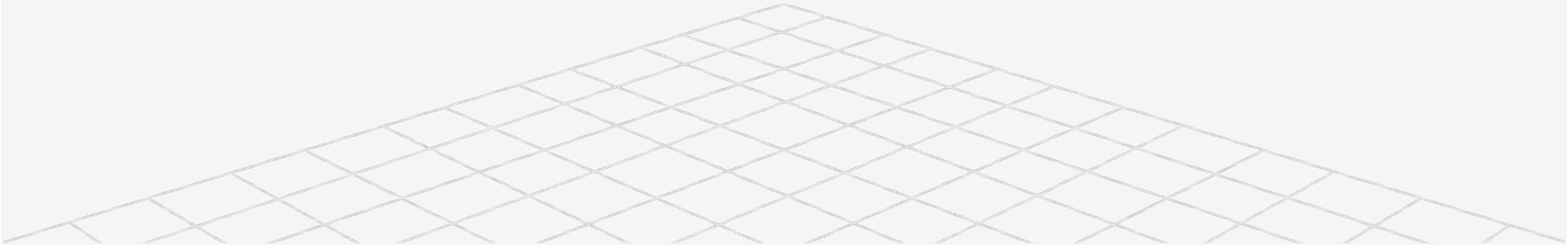
When to use?

- It is useful in development of systems having high level of user interactions such as online systems
- Systems which need users to fill out forms or go through various screens before data is processed can use prototyping very effectively to give the exact look and feel even before the actual software is developed



Big Bang Model

- It is an SDLC model where we do not follow any specific process.
- The development just starts with the required money and efforts as the input, and the output is the software developed which may or may not be as per customer requirement.
- This Big Bang Model does not follow a process/procedure and there is a very little planning required.
- Usually this model is followed for small projects where the development teams are very small.



Big Bang Model

Advantages

- Very simple model
- Easy to manage
- Less planning required
- Flexible to developers

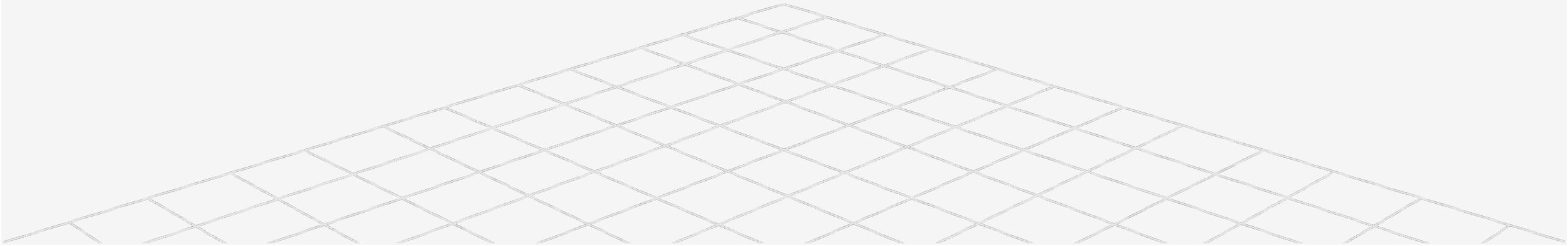
Disadvantages

- High risk & uncertainty
- Not good for complex project
- Poor model for long & ongoing projects

Big Bang Model

When to use?

- This model is good for small projects with one or two developers working together.
- It is good model for the product where requirements are not well understood and the final release date is not given.



Thank You...