

Software Process

What is Software Process?

Who does it?

Why is it done?

What is output?



Phases of Software Development

1. Project Startup - Concept, Proposal, Scope
2. Requirements and Analysis.
3. High level Design.
4. Low level Design.
5. Construction - Unit test, Code inspection.
6. Integration and System tests.
7. Replication, delivery, installation.
8. Acceptance Testing (Requirements mapped)
9. Training, Documentation
10. Project windup - Checklist, Report, Lessons learnt.
11. Maintenance (may involve all the above)

Software process model

- **Attempt to organize the software life cycle by**
 - defining activities involved in software production
 - order of activities and their relationships
- **Goals of a software process**
 - standardization, predictability, productivity, high product quality, ability to plan time and budget requirements




Software Life Cycle

- A **software life cycle** is the series of identifiable stages that a software product undergoes during its lifetime
- The first stage is called **scope**
- The subsequent stages are: **requirement Mgmt: analysis and specification, design, coding, testing, and maintenance**
- Each of these stages is called a *life cycle phase*



Benefits of SDLC

- Increasing quality
 - Reducing project cost and schedule
- 

Process Models

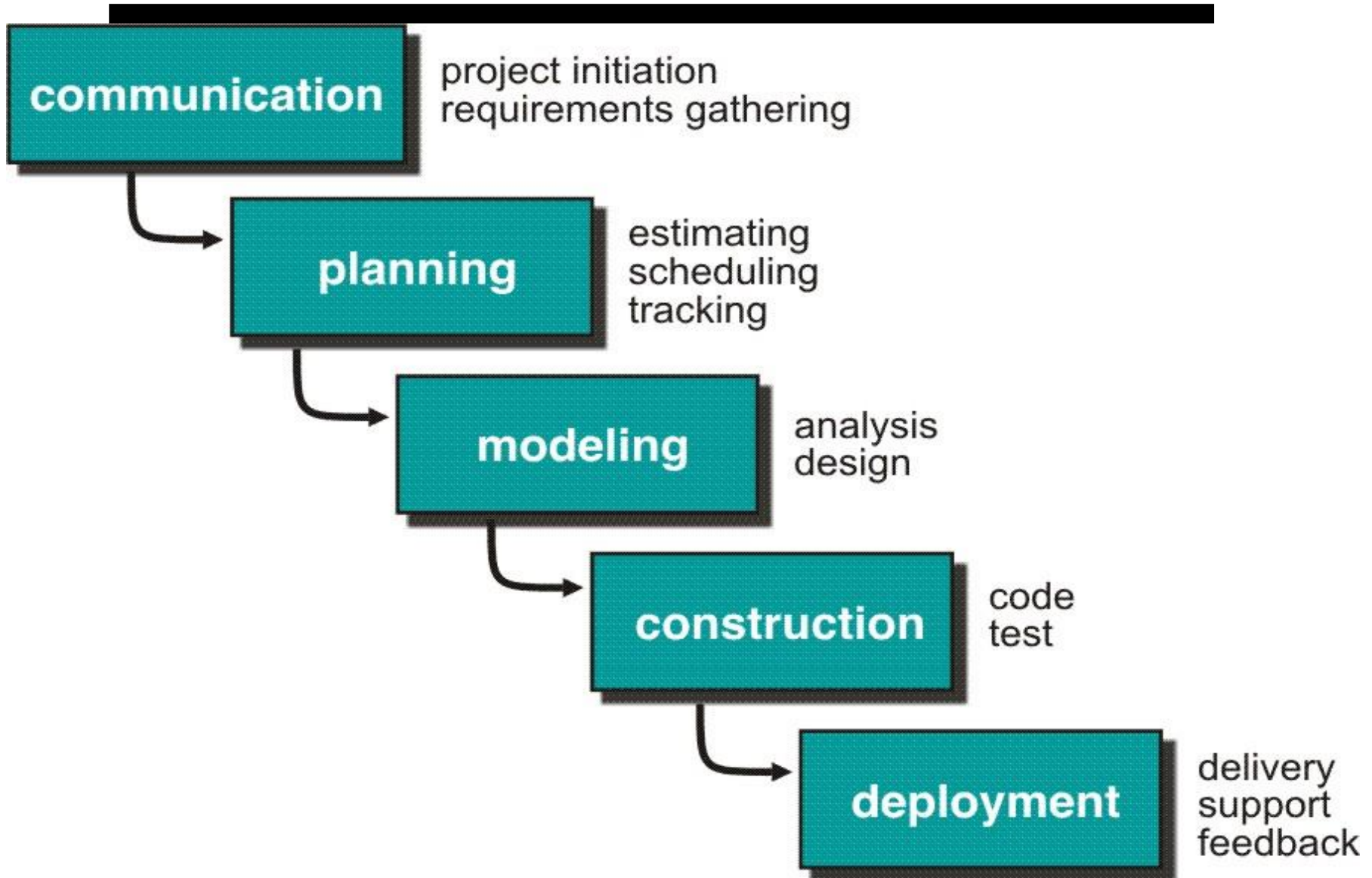
Prescriptive Process Models

- The Waterfall Model
- Incremental Model
- The RAD Model

Evolutionary Process Models

- The Prototyping Model
- The Spiral Model
- The Concurrent Development Model

The Waterfall Model





Linear Process Models

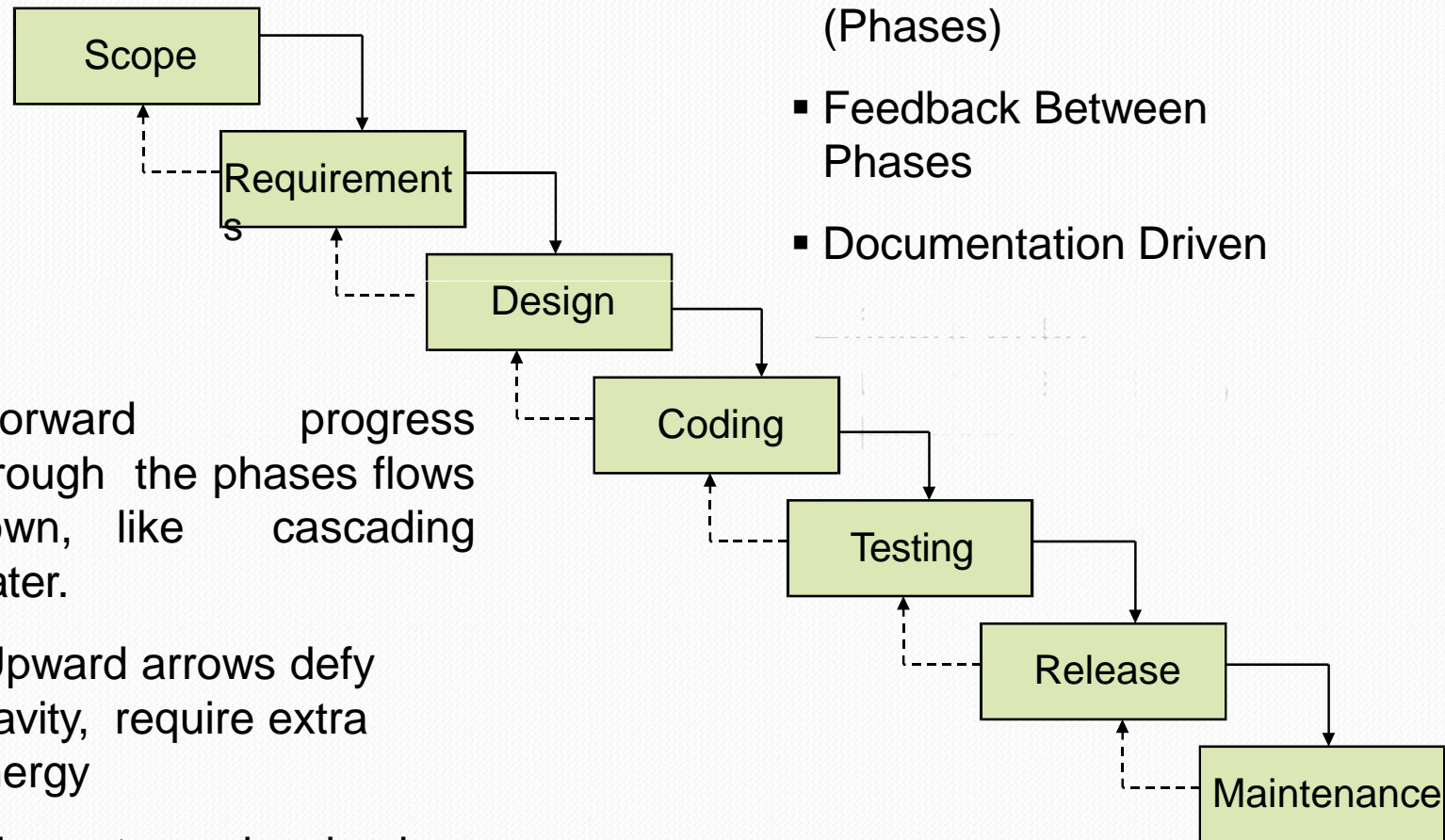
(Contd...
)

Waterfall Model

- **Pre-requisites/Pre-conditions**
 - **Requirements are all available**
 - **Requirements do not change**
 - **Relationship with customer is mature (means we understand their needs/requirements properly and they also have confidence in us)**

Linear Process Models

Waterfall Model



- Sequential Steps (Phases)
- Feedback Between Phases
- Documentation Driven

▪ Forward progress through the phases flows down, like cascading water.

▪ Upward arrows defy gravity, require extra energy

▪ No customer involved
From Design till
Deployment



Waterfall Strengths

- Easy to understand, easy to use
- Provides structure to inexperienced staff
- Milestones are well understood
- Sets requirements stability
- Good for management control (plan, staff, track)
- Works well when quality is more important than cost or schedule ##

Waterfall Deficiencies

- All requirements must be known upfront
 - Deliverables created for each phase are considered frozen – inhibits flexibility
 - Can give a false impression of progress
 - Integration is one big bang at the end
 - Little opportunity for customer to preview the system (High Risk of wrong product)
- ##



When to use the Waterfall Model

- Requirements are very **well known**
- Product definition is **stable**
- Technology is **understood**
- New **version of an existing product**
- **Porting an existing product** to a new platform.

##

The Waterfall Model: (Payroll System)

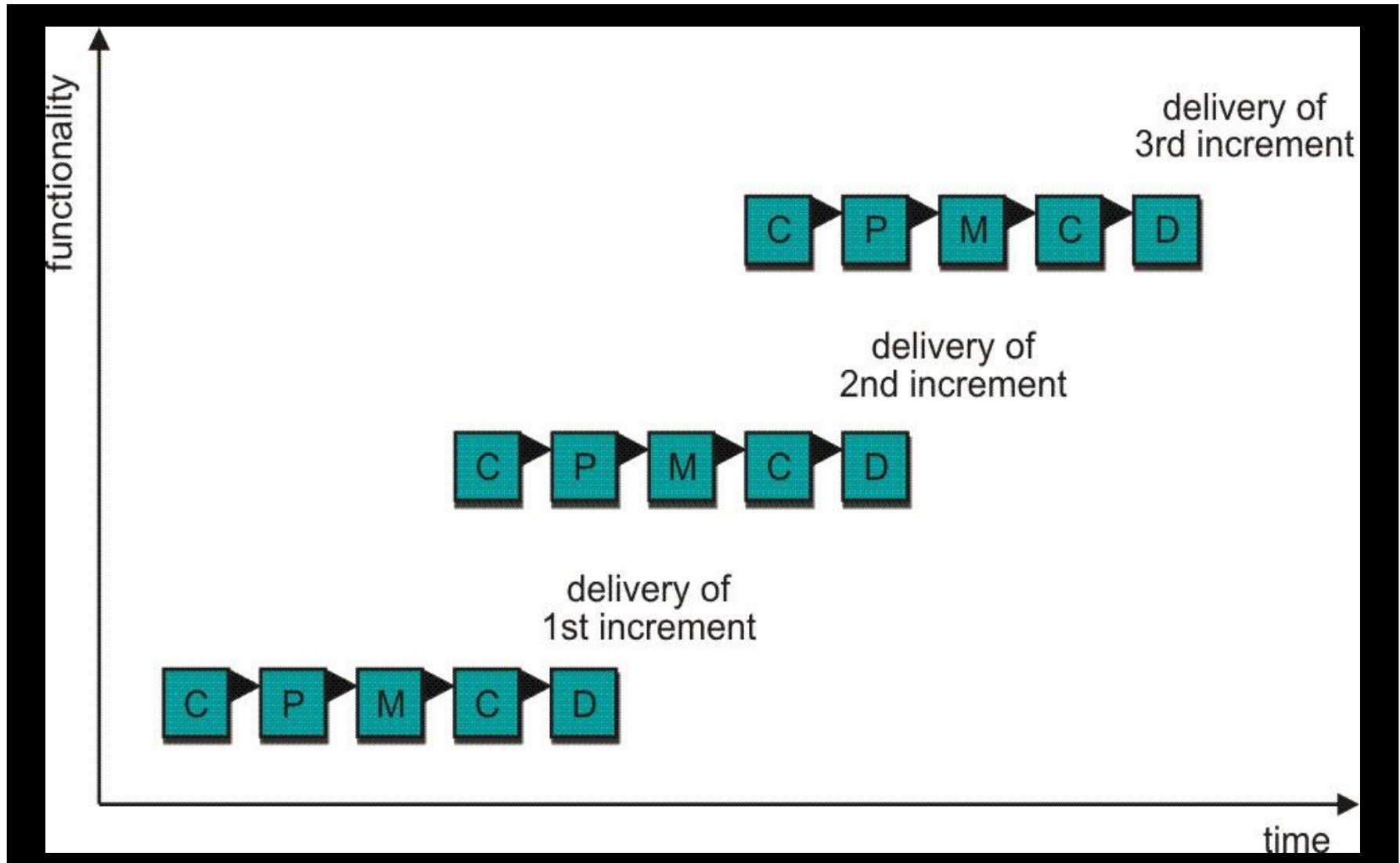
Merits :

- It is systematic sequential approach for Software Development

Demerits

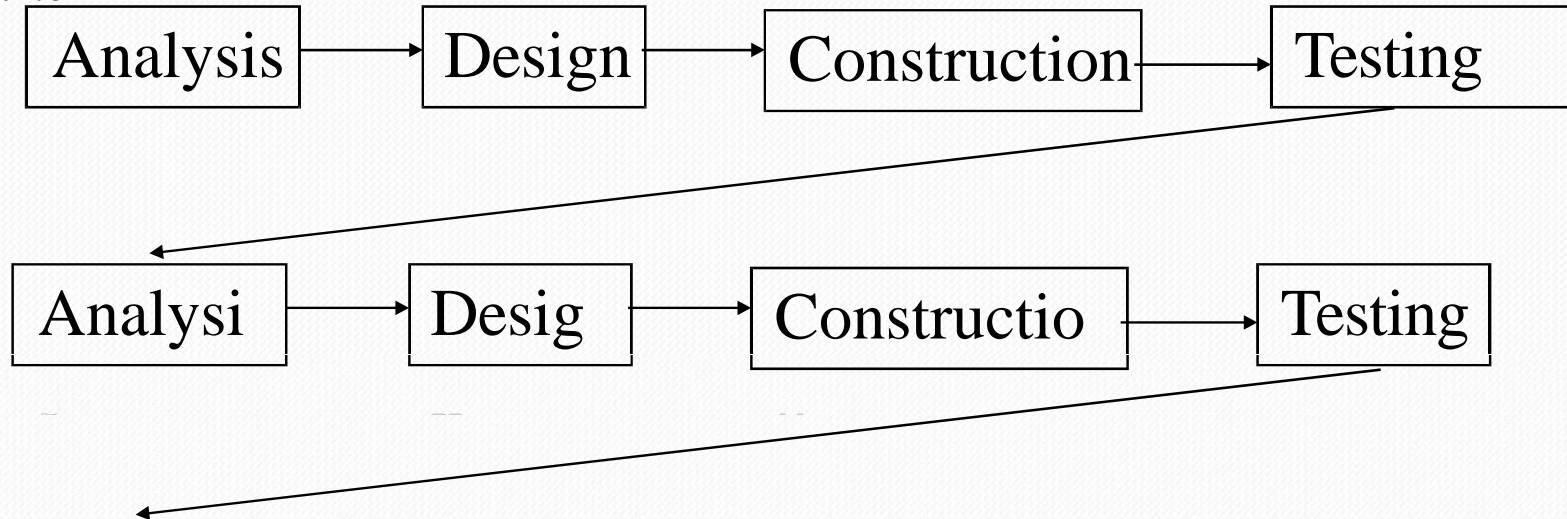
- All Customer Requirements at the start of project may be difficult
- Problems remain uncovered until testing phase
- Customer patience is needed, working version of the software is delivered too late.

Incremental Models: Incremental



Linear Process Models

Incremental Model



- Adding requirements one by one.
- When,
 - All the requirements are not available (Manufac. Comp.) and/or Part of the software is needed earlier (SBI-ICICI) and/or
 - When the relationship with customer is not mature and/or Not enough resources are available

Incremental Model: (Word Processor)

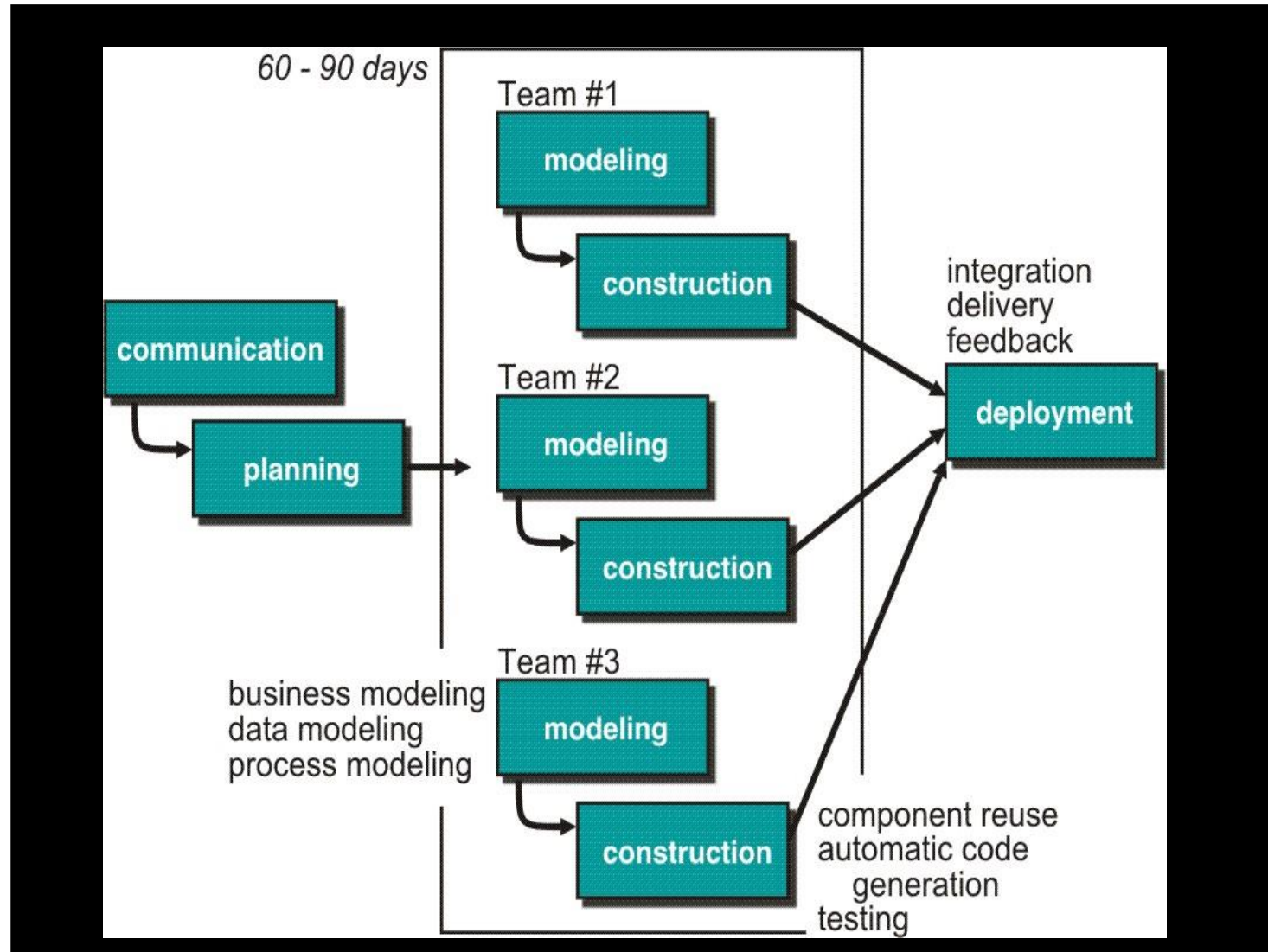
Merits :

- Less number of developers required
- All the requirements need not be known at the beginning of the project
- Technical risks can be managed

Demerits :

- Problems remain uncovered until testing phase
- Customer patience is needed, working version of the software is delivered too late.

Incremental Models: RAD Model



The RAD Model : (Very Large Projects)

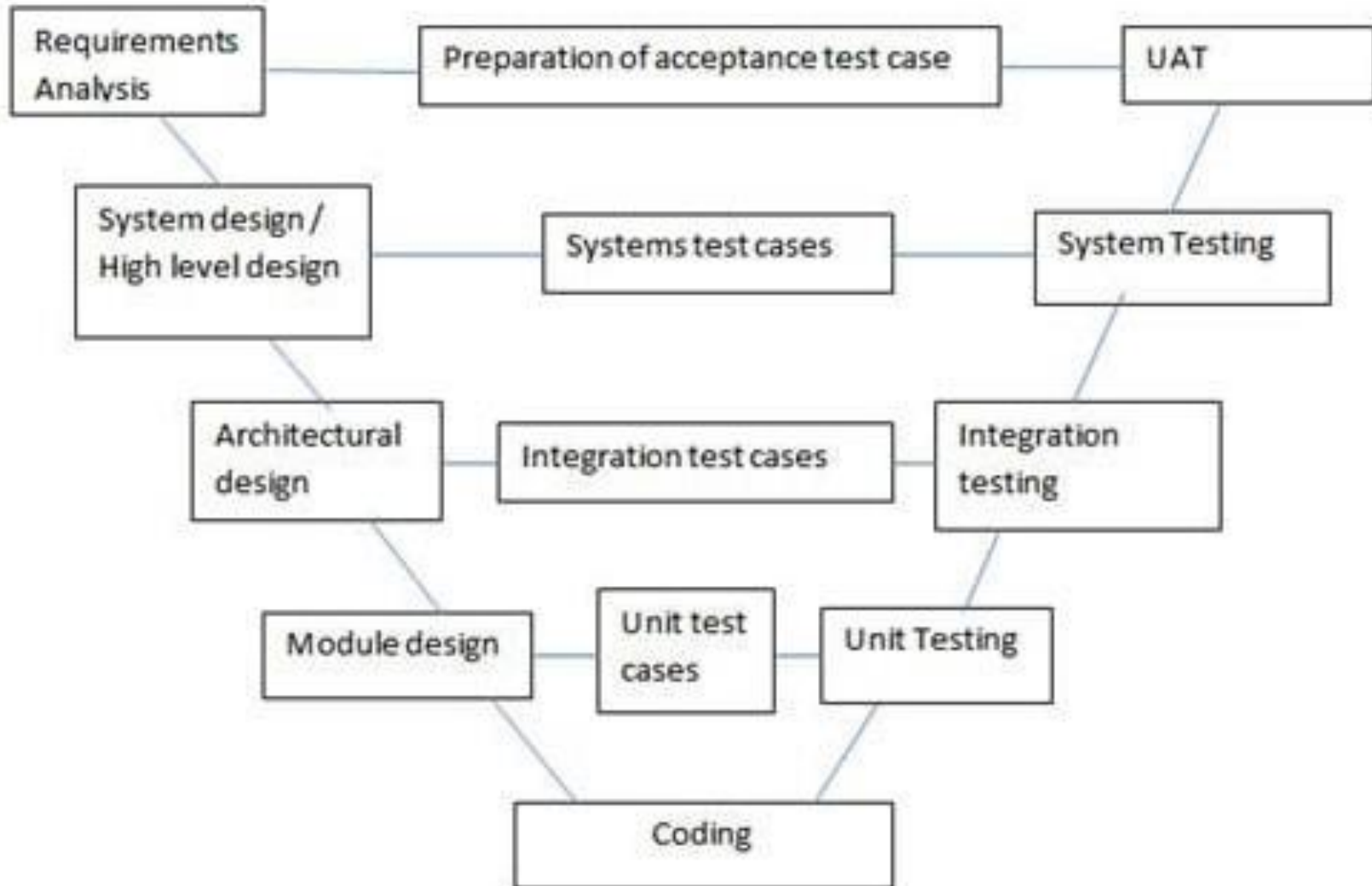
Merits :

- Project cycle time is reduced

Demerits :

- All Customer Requirements at the start of project may be difficult
- For large projects high human resources are required
- Risk of project failure if teams are not committed to rapid fire action
- Problems due to improper modularization of system
- RAD approach may not work if high is an issue
- RAD may not be appropriate if technical risks are very high

V Model –Verification, Validation Model



Linear Process Models

(Contd...
)

Prototyping Model

Listen to
custome
r

Build/revis
e mock-

Customer test
- drive mock
up

- Customer sticks to prototype for a working software
- Developer implements only min. necessary stuff
- Build an example system to help elicit requirements
- Perfection of Prototype can take too much time

Evolutionary Process Models: Need

- **Software like all complex systems evolves over a period of time.**
- **Target market deadlines make completion of a comprehensive software product impossible, but a limited version must be introduced to meet competitive or business pressure. Some of the core product or system requirements are well understood but the details of product or system extensions have yet to be defined.**
- **Solution is to adapt Evolutionary Model which is Iterative**

The Prototyping Model: (Need)

- Prototyping is used when customers requirements are fuzzy.
- OR the developer may not be sure of the efficiency of algorithm, the adaptability of an Operating System or the form that Human Computer interaction should take
- But we have to throw away the prototype once the customer requirements are clear & met for better quality. The product must be rebuilt using software engineering practices for long term quality.

Evolutionary Models: Prototyping

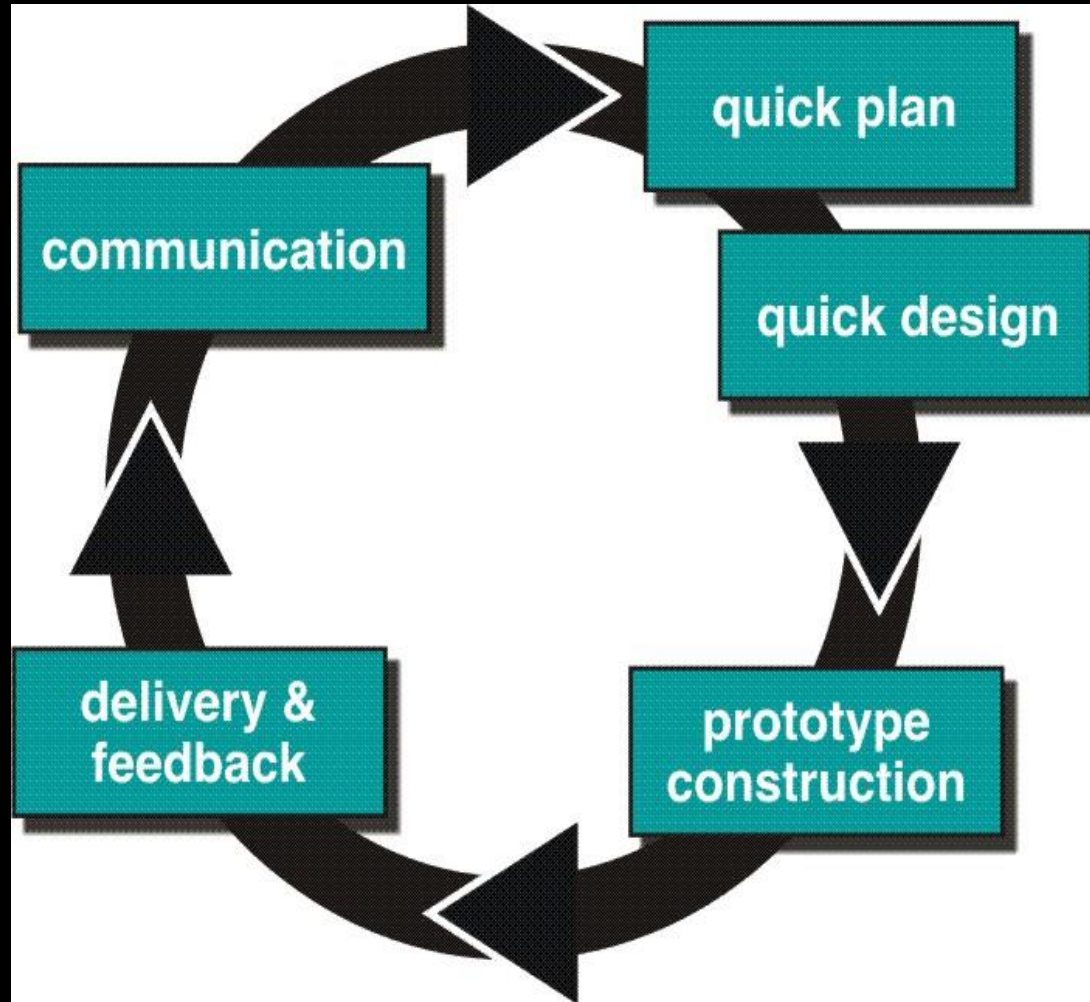
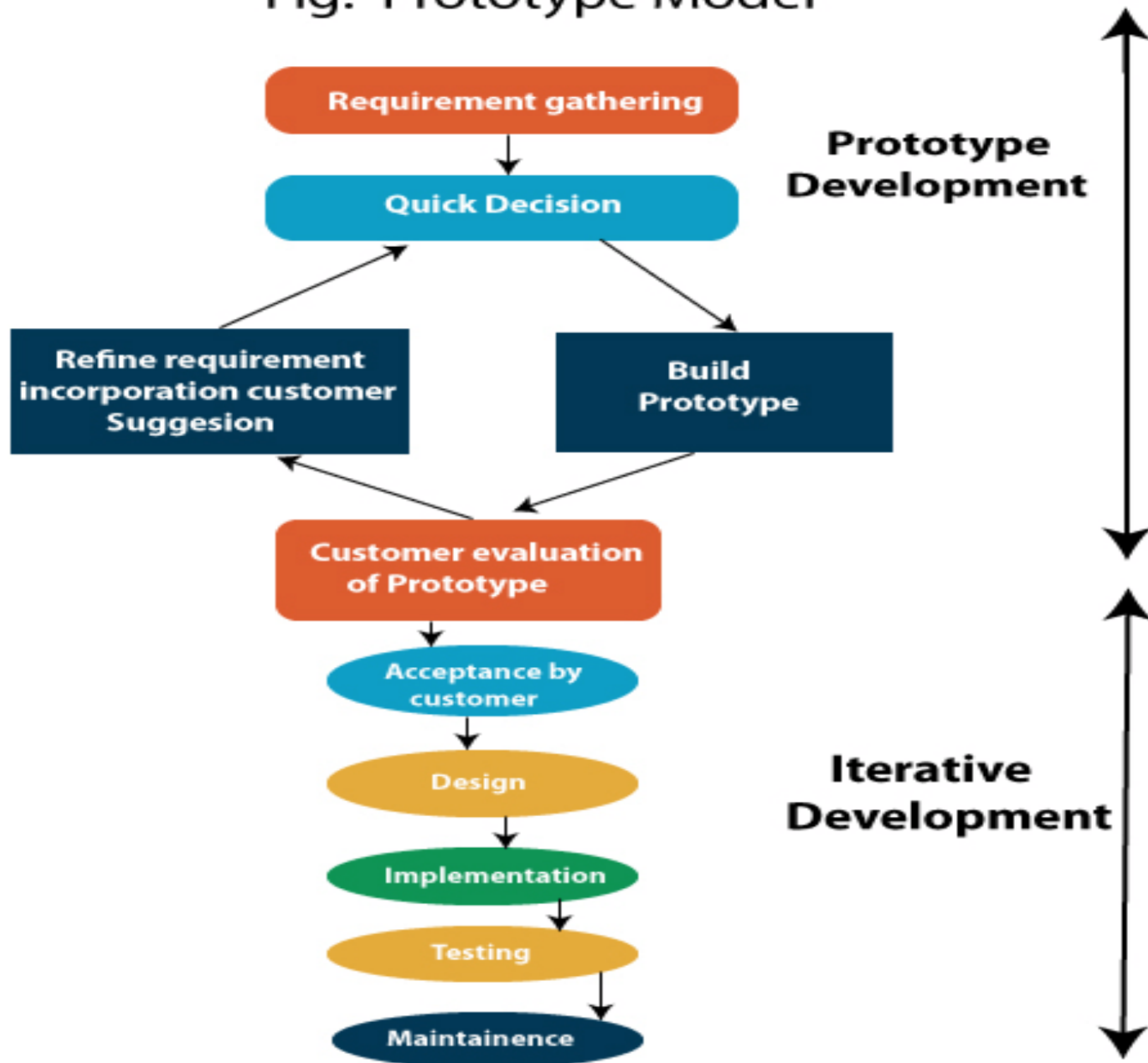


Fig: Prototype Model



The Prototyping Model:

Merits:

- Prototyping helps in requirement gathering & can be applied at any stage of the project.

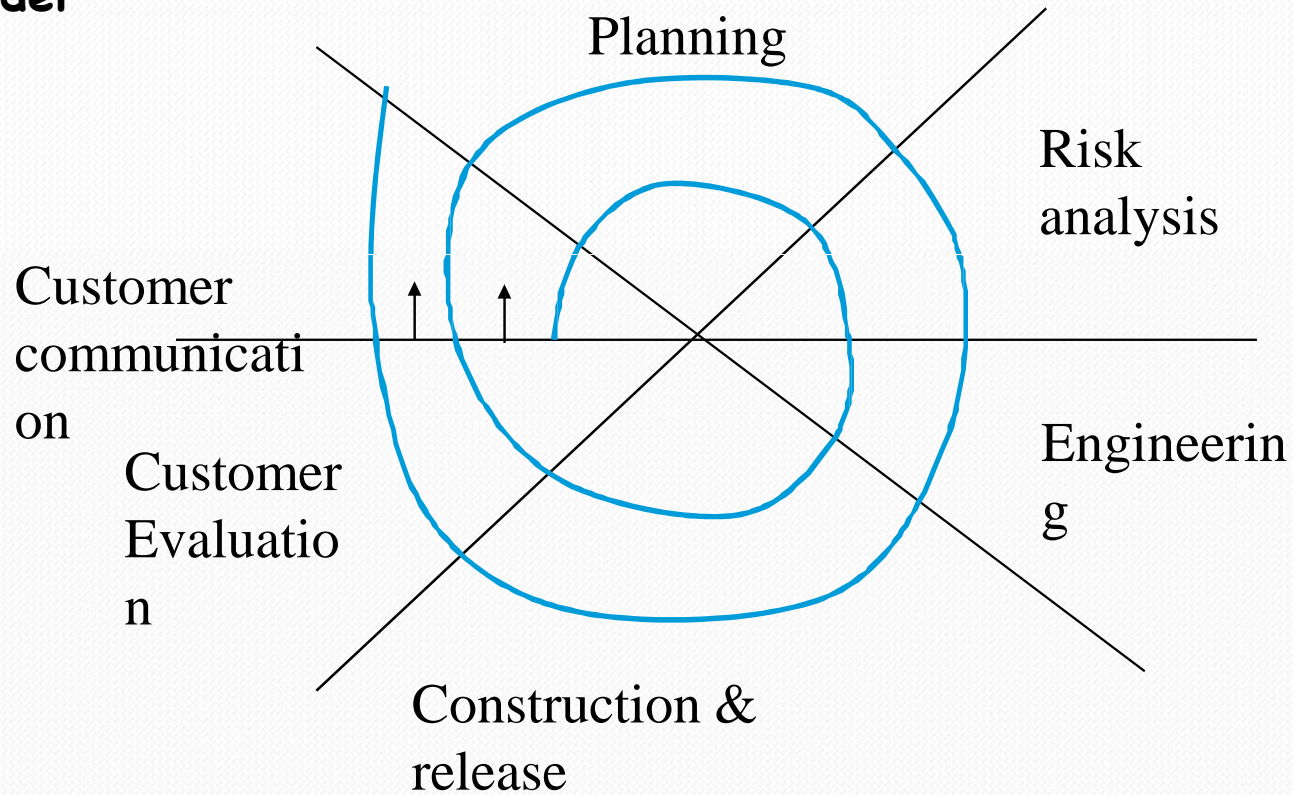
Demerits:

- Customer insists to convert prototype in working version by applying “few fixes”
- Developer may become comfortable with the compromises done. “The-less-than-ideal-choice” may become integral part of the system

Evolutionary Process Models

(Contd...
)

Spiral Model



Spiral Model

- Suitable for Software or Products that *evolve* over a period of time
- Uses prototyping as a risk reduction mechanism
- Relies on expertise for success (especially risk assessment)

Evolutionary Models: Spiral

Spiral Model is an evolutionary software process model that couples the iterative nature of Prototyping with controlled & systematic aspects of the Waterfall Model

Merits:

- Risk is considered as each iteration is made
- Spiral Model can be applied throughout the life of the computer software.

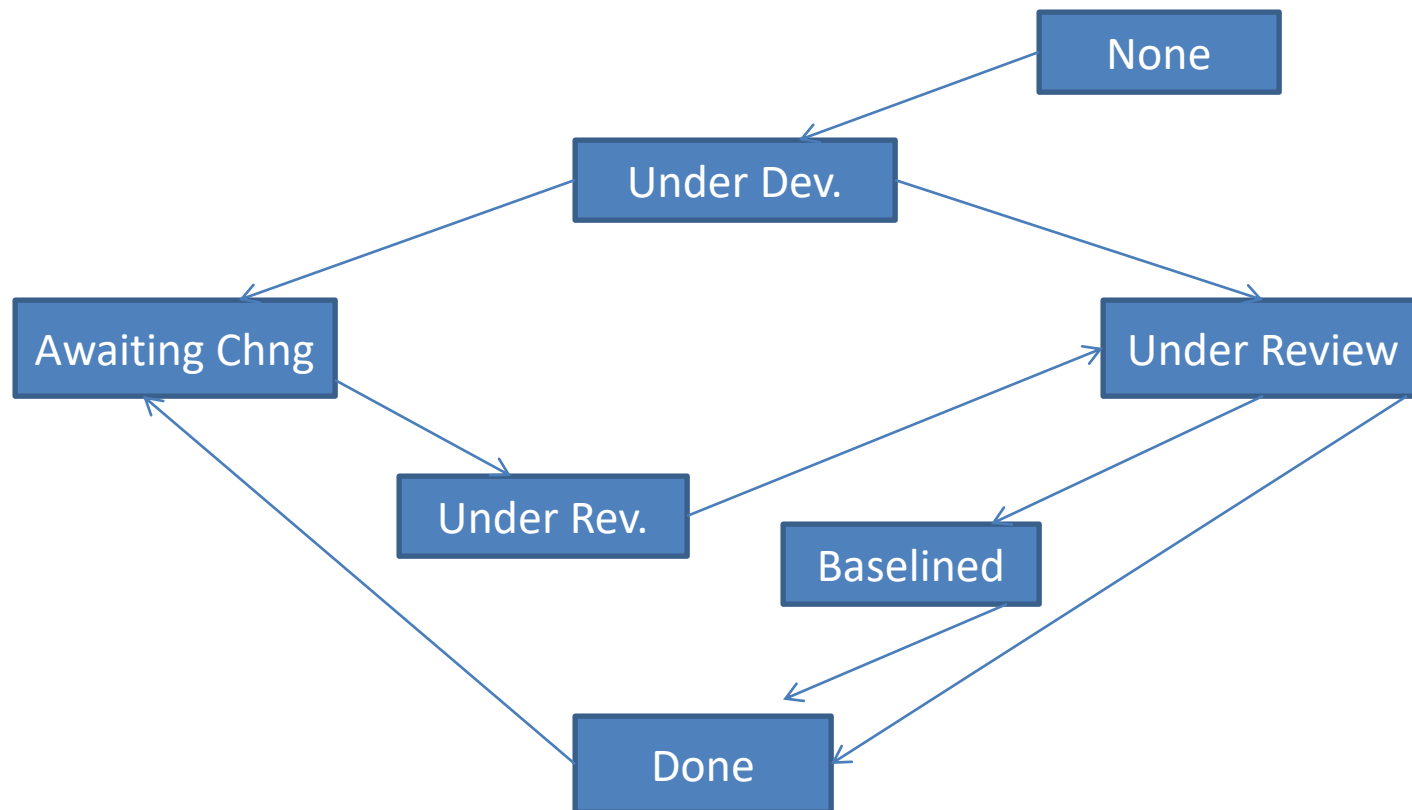
Demerits:

- It is difficult to convince customers that the evolutionary approach is controllable
- Considerable risk assessment expertise required
- If major risk is uncovered, problems will occur

Concurrent Development Model:

The concurrent Development Model, sometimes called concurrent engineering can be represented schematically as a series of framework activities, software engineering actions and tasks, & their associated states. All activities exist concurrently.

Modeling activity (Example) :



Concurrent Development Model: Contd...

Merits:

- Applicable to all types of S/W development & provides an accurate picture of the current state of the project.

Demerits:

- Problem to Project planning. How many No of iterations are to be planned? Uncertainty...
- Process may fall in chaos if the evolutions occurs too fast without a period of relaxation. On the other hand if the speed is too slow productivity could be affected.
- S/W processes are focussed on flexibility & extendability, rather than on high quality.