UNIT-1

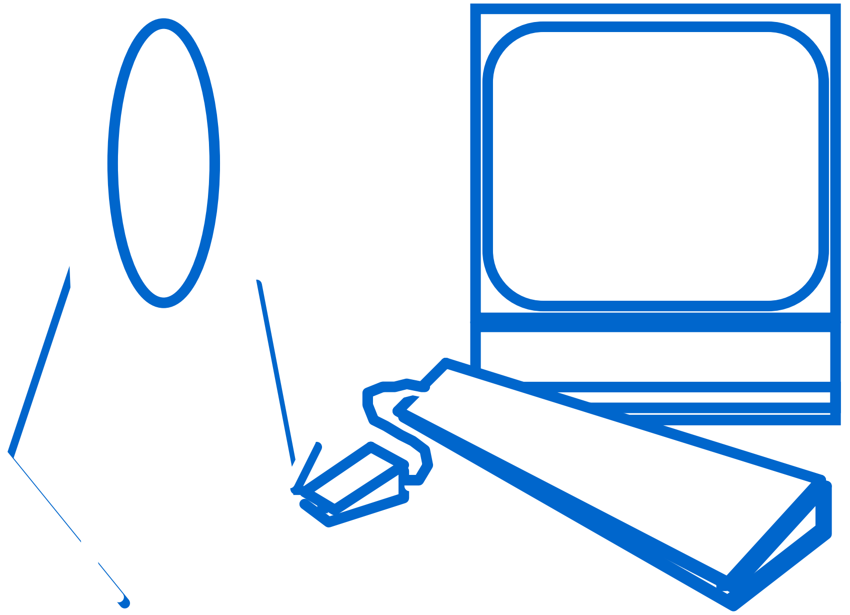
SOFTWARE PROCESS

Introduction –S/W Engineering Paradigm – life cycle models (water fall, incremental, spiral, WINWIN spiral, evolutionary, prototyping, object oriented)

What is Software?

Software is a set of items or objects that form a “configuration” that includes

- programs
- documents
- data



What is Software? Characteristics

The basic software have following characteristics:

- Software is engineered
- Software doesn’t wear out
- Software is complex

software is engineered

- **Software is developed or engineered, it is not manufactured in the classical sense.**

software doesn’t wear out

FIGURE 1.1
Failure curve for hardware

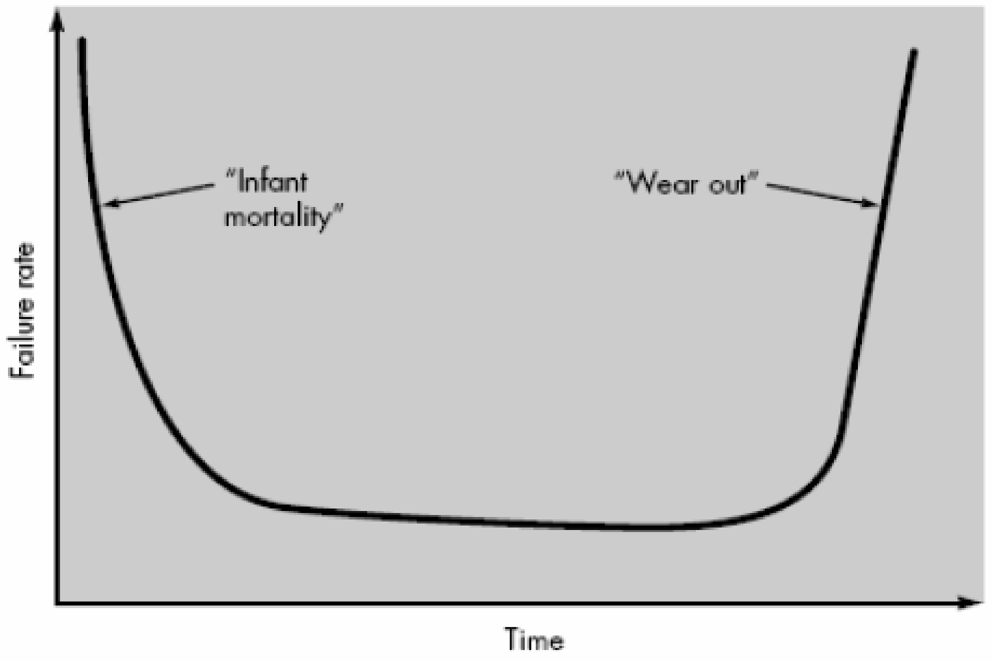
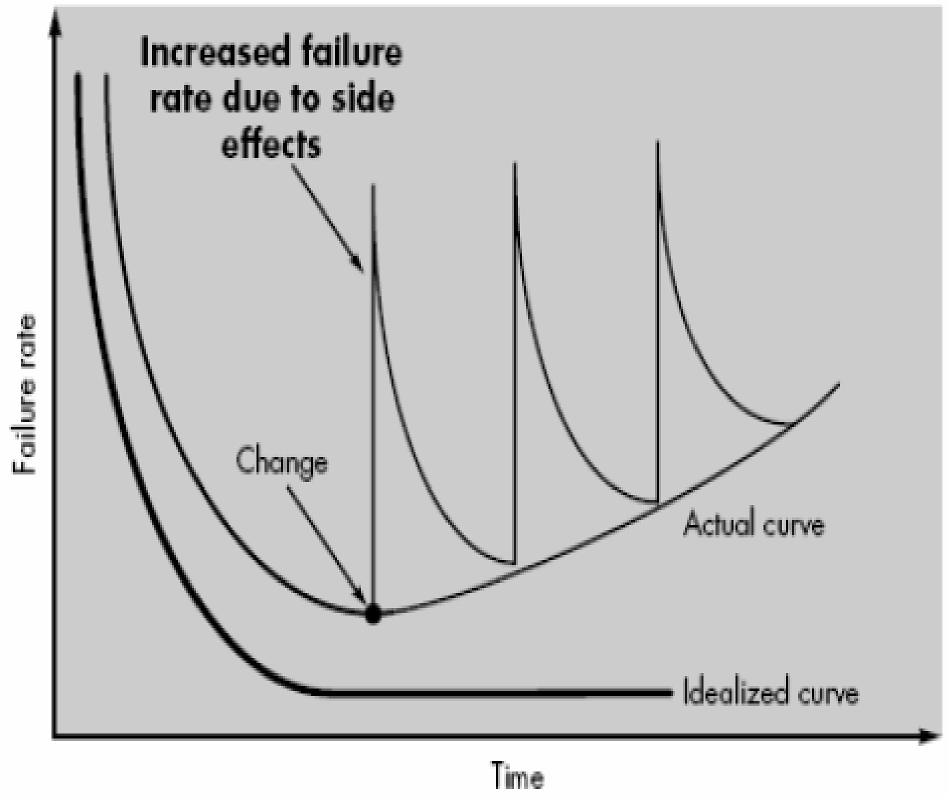


FIGURE 1.2
Idealized and actual failure curves for software



Software Applications

- System software
- real-time software(application sw)
- business software
- engineering/scientific software
- embedded software
- PC software
- AI software
- WebApps (Web applications)

Goals/ Objectives of Software Engineering

- **Satisfy user requirements**
- **High reliability**
- **Low Maintenance costs**
- **Delivery on time**
- **Low production costs**
- **High Performance**
- **Ease of use**

(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

Software engineering

- Software engineering (SE) is the application of a systematic, disciplined, quantifiable approach to the design, development, operation, and maintenance of software, and the study of these approaches; that is, the application of engineering to software.
- The computer science discipline concerned with developing large applications. Software engineering covers not only the technical aspects of building software systems, but also management issues, such as directing programming teams, scheduling, and budgeting.

(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

Source2: Wikipedia

Cont..

- A systematic approach to the analysis, design, implementation and maintenance of software.
- Software engineering is the engineering discipline through which software is developed. Commonly the process involves finding out what the client wants, composing this in a list of requirements, designing an architecture capable of supporting all of the requirements, designing, coding, testing and integrating the separate parts, testing the whole, deploying and maintaining the software. Programming is only a small part of software engineering.
- **Software engineering**: a discipline that uses computer and software technologies as a problem-solving tools

(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill) Source1 : Wikipedia

Cont..

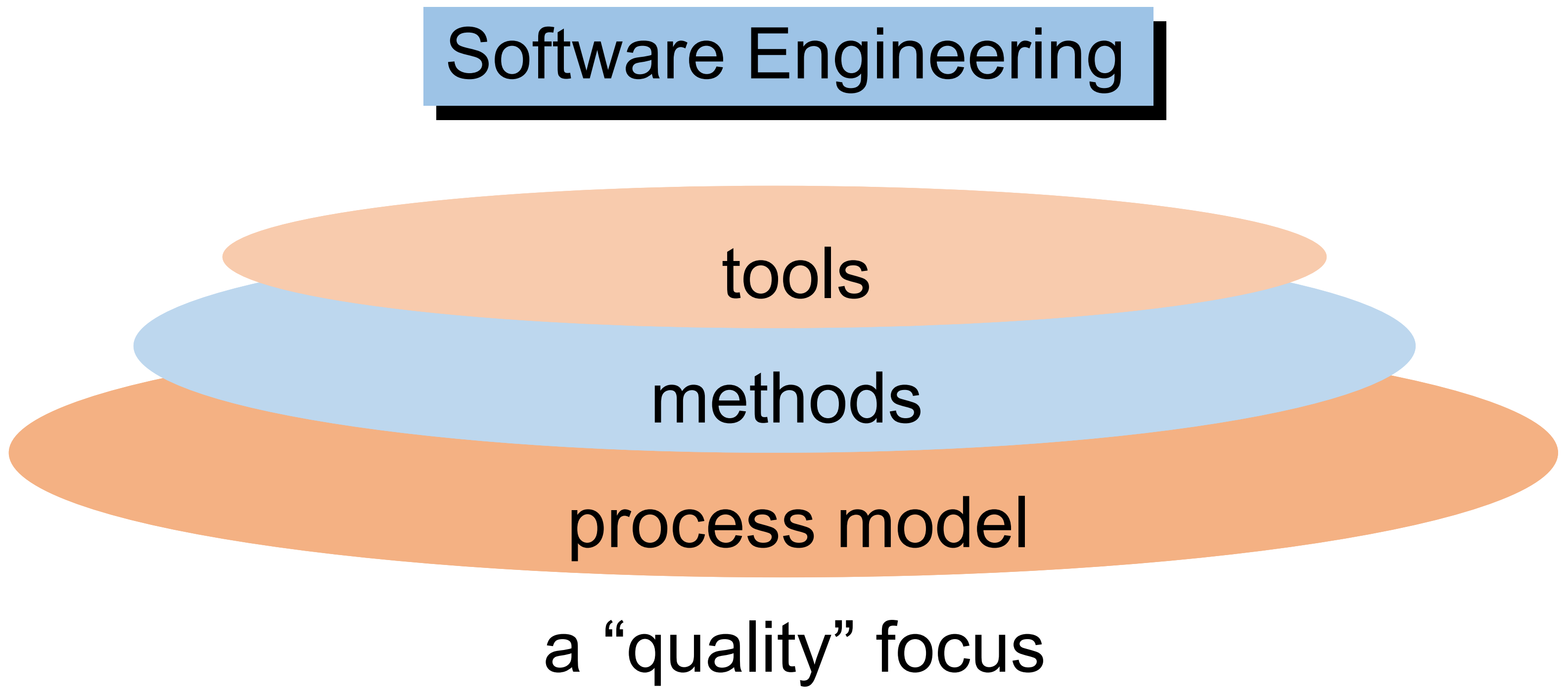
The Software Engineering Body of Knowledge (SWEBOK) divides software engineering into 10 knowledge areas:

- Software requirements
- Software design
- Software construction
- Software testing
- Software maintenance
- Software configuration management
- Software engineering management
- Software engineering process
- Software engineering tools and methods
- Software quality

(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

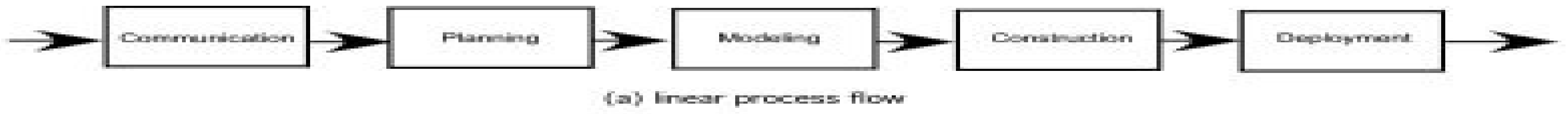
Source: Wikipedia

A Layered Technology

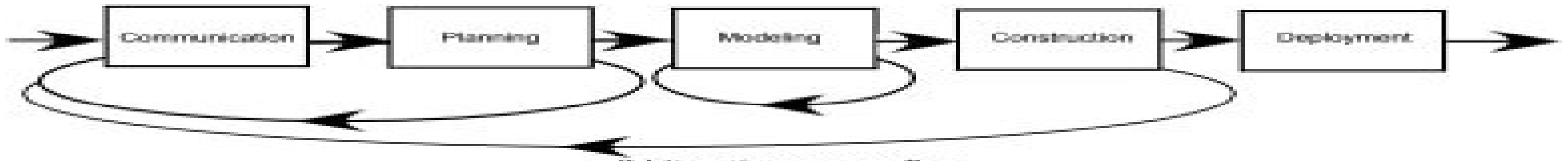


(Source: Pressman, R. Software Engineering: A Practitioner’s Approach. McGraw-Hill)

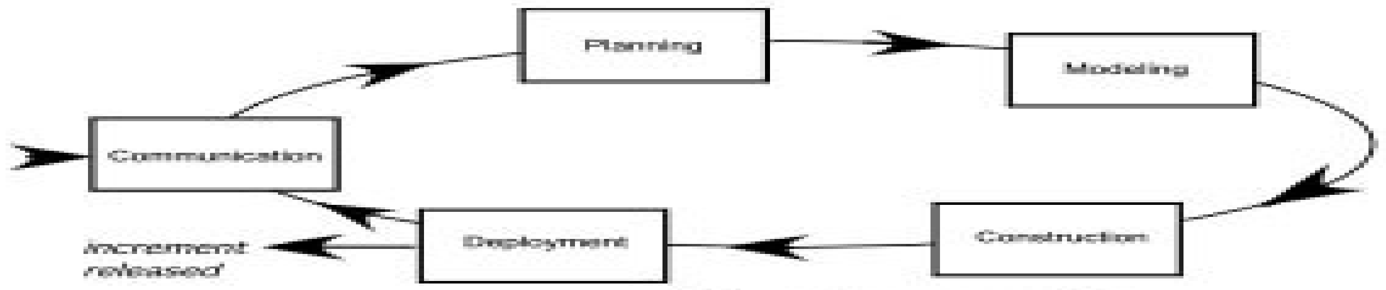
Process Flow



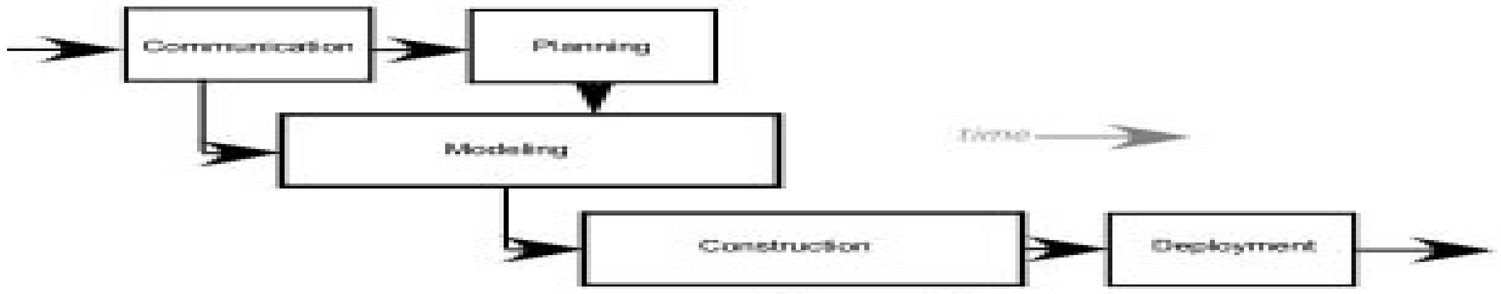
(a) linear process flow



(b) iterative process flow



(c) evolutionary process flow



(d) parallel process flow

Process Flow

- Linear process flow executes each of the five activities in sequence.
- An iterative process flow repeats one or more of the activities before proceeding to the next.
- An evolutionary process flow executes the activities in a circular manner. Each circuit leads to a more complete version of the software.
- A parallel process flow executes one or more activities in parallel with other activities (modeling for one aspect of the software in parallel with construction of another aspect of the software.)

S/W Engineering Paradigm or process

- The software development strategy is referred as Software Engineering Paradigm.
- The software development strategy consists of methods, tools, and procedures. There exist various software development strategies or process models.

Software Development Models (or) process models

Software Development Models

- Water Fall Model
- Incremental process model
  - Incremental Model
  - RAD (Rapid Application Development) Model
- Evolutionary Process models:
  - Spiral Model
  - WINWIN Spiral Model
- Prototyping model
- Object oriented Model

- Each model has its own specific steps for software development . A suitable development model is selected by considering several factors like requirement , application type, application software to be used for development etc

(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

The Waterfall Model (1)

- Sometimes called the *classic life cycle*
- Suggests a systematic, sequential (or linear) approach to s/w development
- The oldest paradigm for s/w engineering
- Works best when –
  - Requirements of a problem are reasonably well understood
  - Well-defined adaptations or enhancements to an existing system must be made
  - Requirements are well-defined and reasonably stable

(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

The Waterfall Model (2)

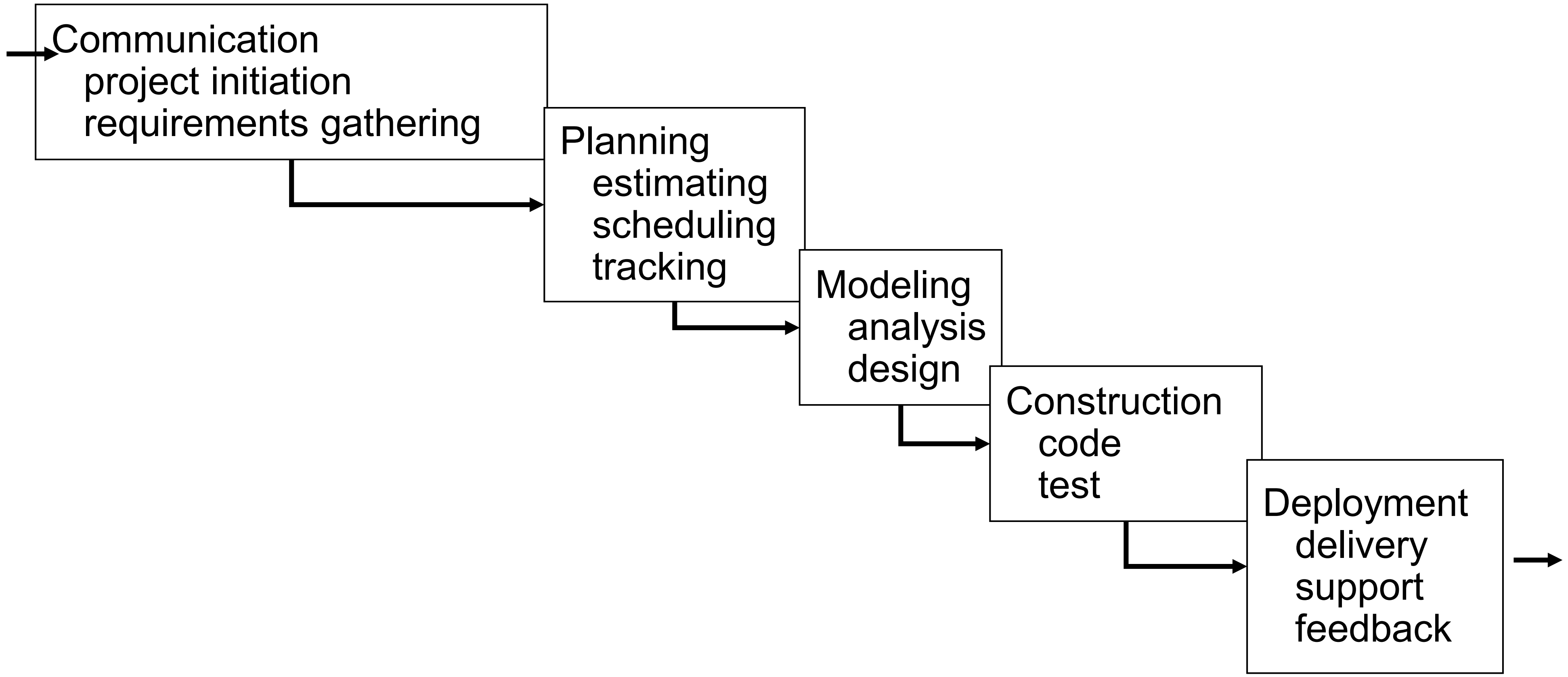


Fig The waterfall model

(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

Real time example

We can take real life examples for water fall model like automobile companies **make cars and bikes**. when they are building a car, the requirements are fixed, predefined. There will be no change in the requirements during the process of building a car. Once we complete a stage, we proceed to the next one. If there is any change in requirements come then we have to go back and make those changes. Then we have to change our complete system that requires extra time and extra money. This is a major disadvantage of water fall model.

Some of other examples are:

1. Online event management system.
2. Material resource planning in a manufacturing unit

The Waterfall Model - Problems

- Real projects rarely follow the sequential flow
  - Accommodates iteration indirectly
  - Changes can cause confusion
- It is often difficult for the customer to state all requirements explicitly
  - Has difficulty accommodating the natural uncertainty that exists at the beginning of many projects
- The customer must have patience
  - A working version of the program(s) will not be available until late in the project time-span
  - A major blunder, if undetected until the working program is reviewed, can be disastrous
- Leads to “blocking states”

Incremental Process Models (1)

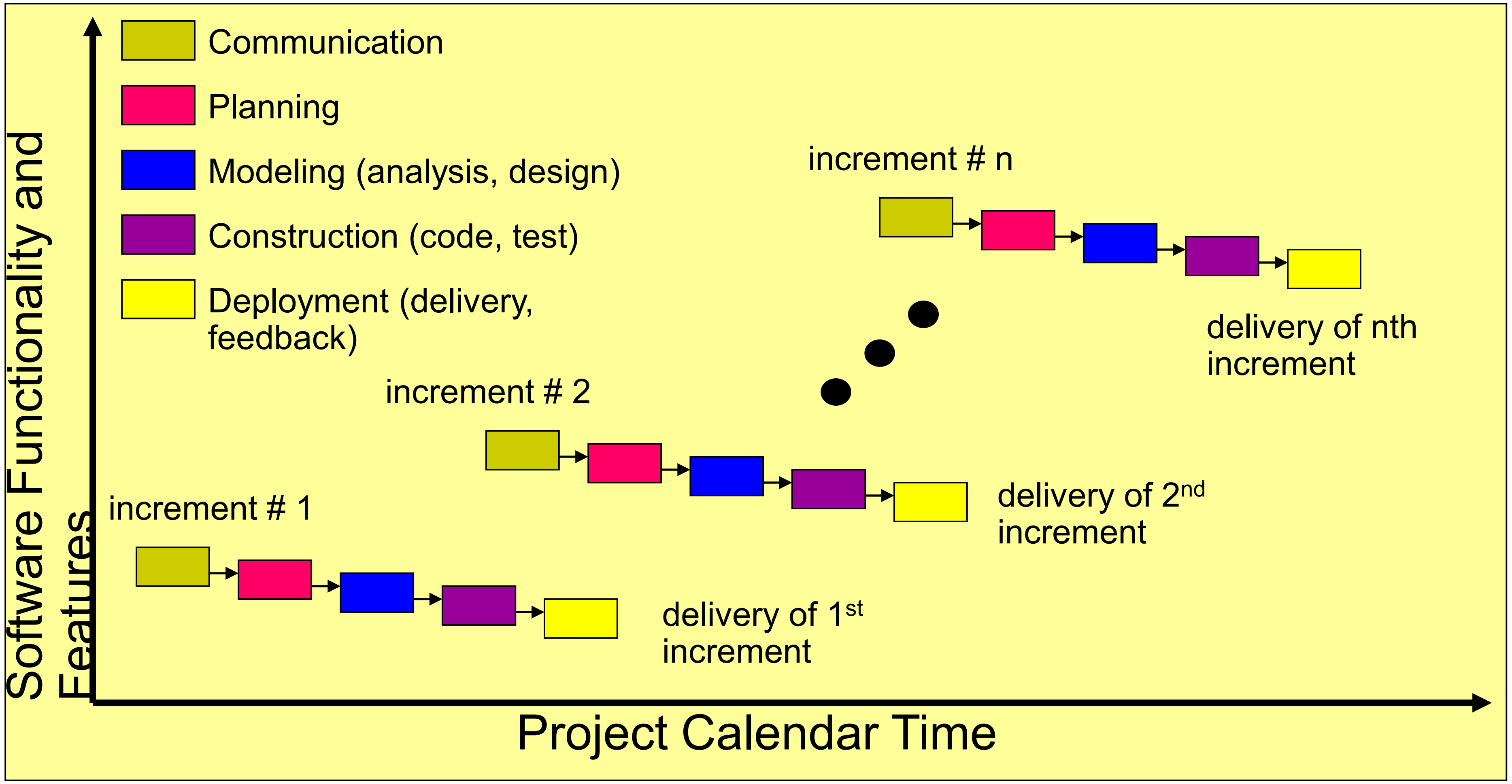

Communication
Planning
Modeling (analysis, design)
Construction (code, test)
Deployment (delivery, feedback)
increment # 1
delivery of 1st increment
increment # 2
delivery of 2nd increment
increment # n
delivery of nth increment
Software Functionality and Features
Project Calendar Time

Fig 3.2: The incremental model

(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

Incremental Models (2)

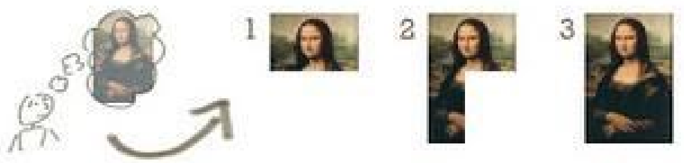
- Combines elements of the waterfall model applied in an iterative fashion
- Each linear sequence produces deliverable “increments” of the software
- The first increment is often a *core product*
- The core product is used by the customer (or undergoes detailed evaluation)
- Based on evaluation results, a plan is developed for the next increment

Incremental Process Models (3)

- The incremental process model, like prototyping and other evolutionary approaches, is iterative in nature
- Particularly useful when
  - **Staffing is unavailable**
- Increments can be planned to manage technical risks

Incremental model example

For Example:

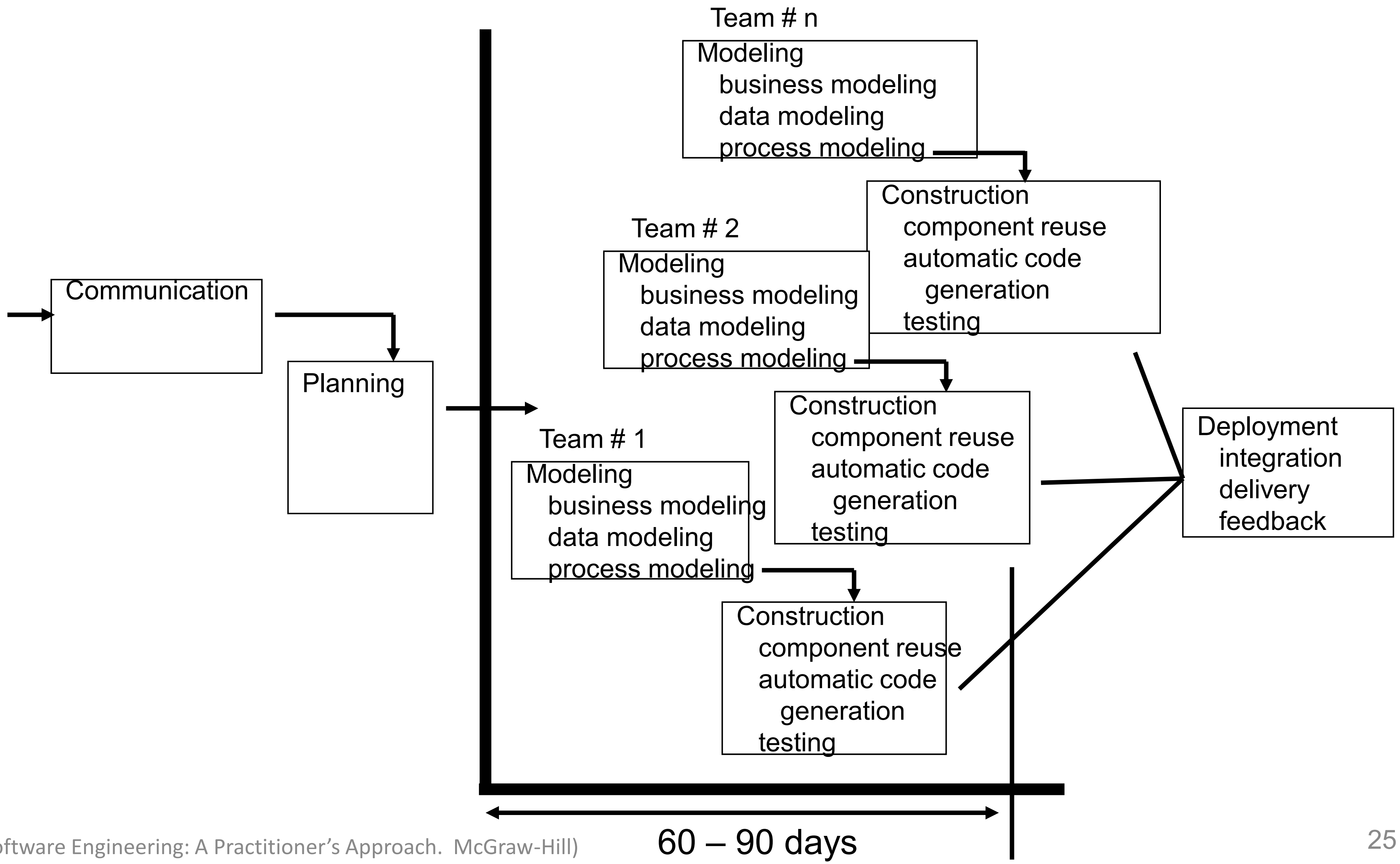


In the diagram above when we work **incrementally** we are adding piece by piece but expect that each piece is fully finished. Thus keep on adding the pieces until it's complete.

RAD(Rapid Application Development)

- Rapid Application Development
- Emphasizes a short development cycle
- A “high speed” adaptation of the waterfall model
- Uses a component-based construction approach
- components or functions are developed in parallel as if they were mini projects.
- May deliver software within a very short time period (e.g. , 60 to 90 days) if requirements are well understood and project scope is constrained

The RAD Model



(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

Advantages and disadvantages

Advantages of the RAD model:

- Reduced development time.
- Increases reusability of components
- Quick initial reviews occur
- Encourages customer feedback
- Integration from very beginning solves a lot of integration issues.

Disadvantages of RAD model:

- Depends on strong team and individual performances for identifying business requirements.
- Requires highly skilled developers/designers.
- High dependency on modeling skills
- Inapplicable to cheaper projects as cost of modeling and automated code generation is very high.

When to use RAD model:

- RAD should be used ***when there is a need to create a system that can be modularized in 2-3 months of time.***
- It should be used if ***there's high availability of designers for modeling and the budget is high enough*** to afford their cost along with the cost of automated code generating tools.

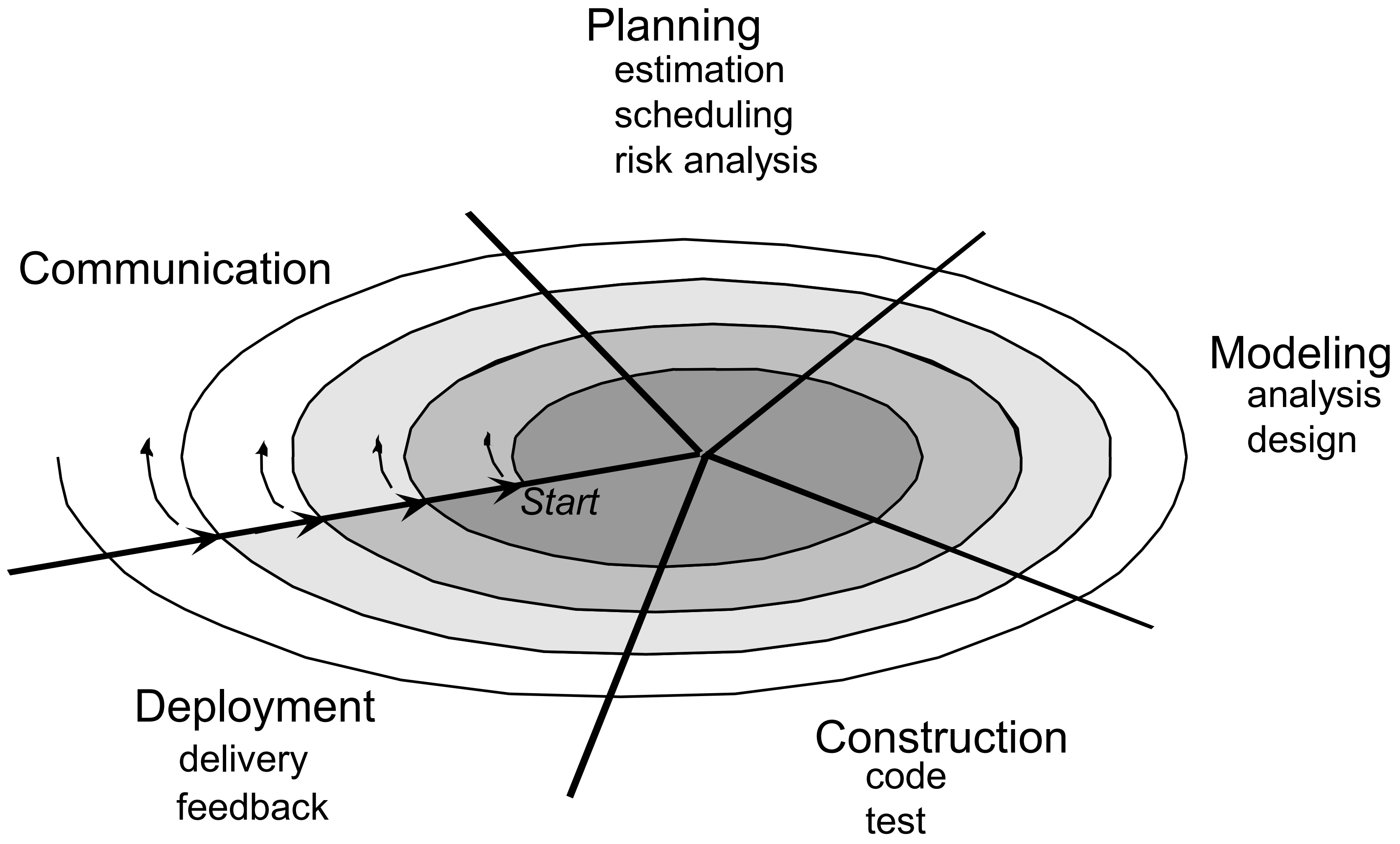
Evolutionary Process Models

- Evolutionary Models take the concept of “evolution” into the engineering paradigm.
- Therefore Evolutionary Models are **iterative.**
- They are built in a manner that enables software engineers to develop increasingly ***more complex versions of the software.***
- Business and product requirements often change as development proceeds, making a straight-line path to an end product is unrealistic.

The Spiral Model (1)

- Couples the iterative nature of prototyping with the controlled and systematic aspects of the waterfall model
- It provides the potential for rapid development of increasingly more complete versions of the software
- It is a *risk-driven process model* generator
- It has two main distinguishing features
  - Cyclic approach
    - Incrementally growing a system's degree of definition and implementation while decreasing its degree of risk
  - A set of *anchor point milestones*
    - A combination of work products and conditions that are attained along the path of the spiral

The Spiral Model (2)



(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

Fig : A typical spiral model

The Spiral Model (3)

- Unlike other process models that end when software is delivered, the spiral model can be adapted to apply throughout the life of the computer s/w
- The circuits around the spiral might represent
  - Concept development project
  - New Product development project
  - Product enhancement project
- The spiral model demands a direct consideration of technical risks at all stages of the project

Examples of Spiral model

- The evolution of Microsoft Operating System, Compilers and other operating systems.

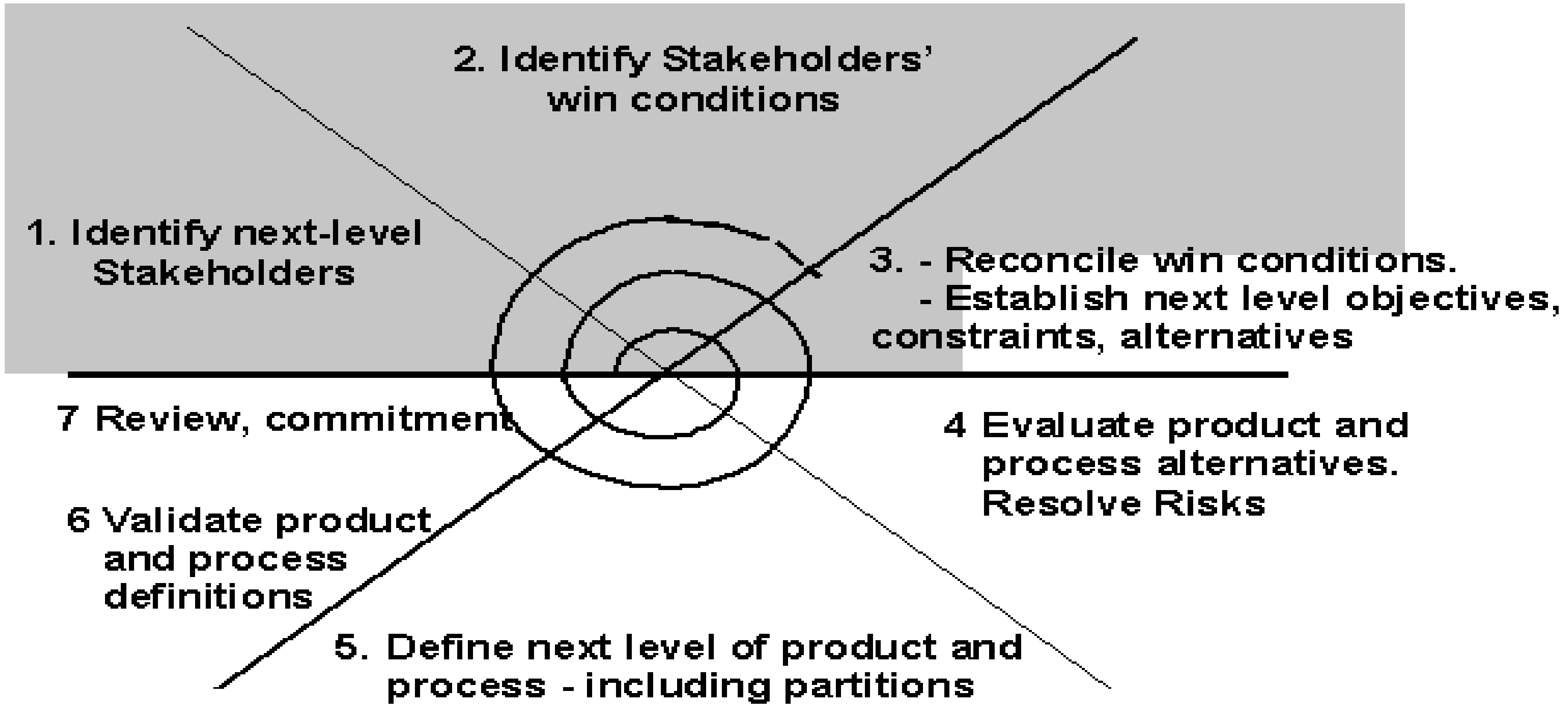
The Spiral Model - Drawbacks

- It may be difficult to convince customers (particularly in contract situations) that the evolutionary approach is controllable.
- It demands considerable risk assessment expertise and relies on this expertise for success. If a major risk is not uncovered and managed, problems will undoubtedly occur.

Win Win Spiral Model

- The Win-Win spiral approach is an extension of the spiral approach.
- The phase in this approach is same as the phase in the spiral approach.
- The only difference is that at the time of the identifying the requirements, the development team and the customer hold discussion and negotiate on the requirements that need to be included in the current iteration of the software.
- *<u>The approach is called Win-Win because it is a winning situation for the development team and also for the customer.</u> The customer wins by getting the product that fulfils most of the requirements while the development team wins by delivering software which is developed with all the requirements established after negotiations with the customer. The Win-Win approach is* ***generally used when you have time-bound releases.***

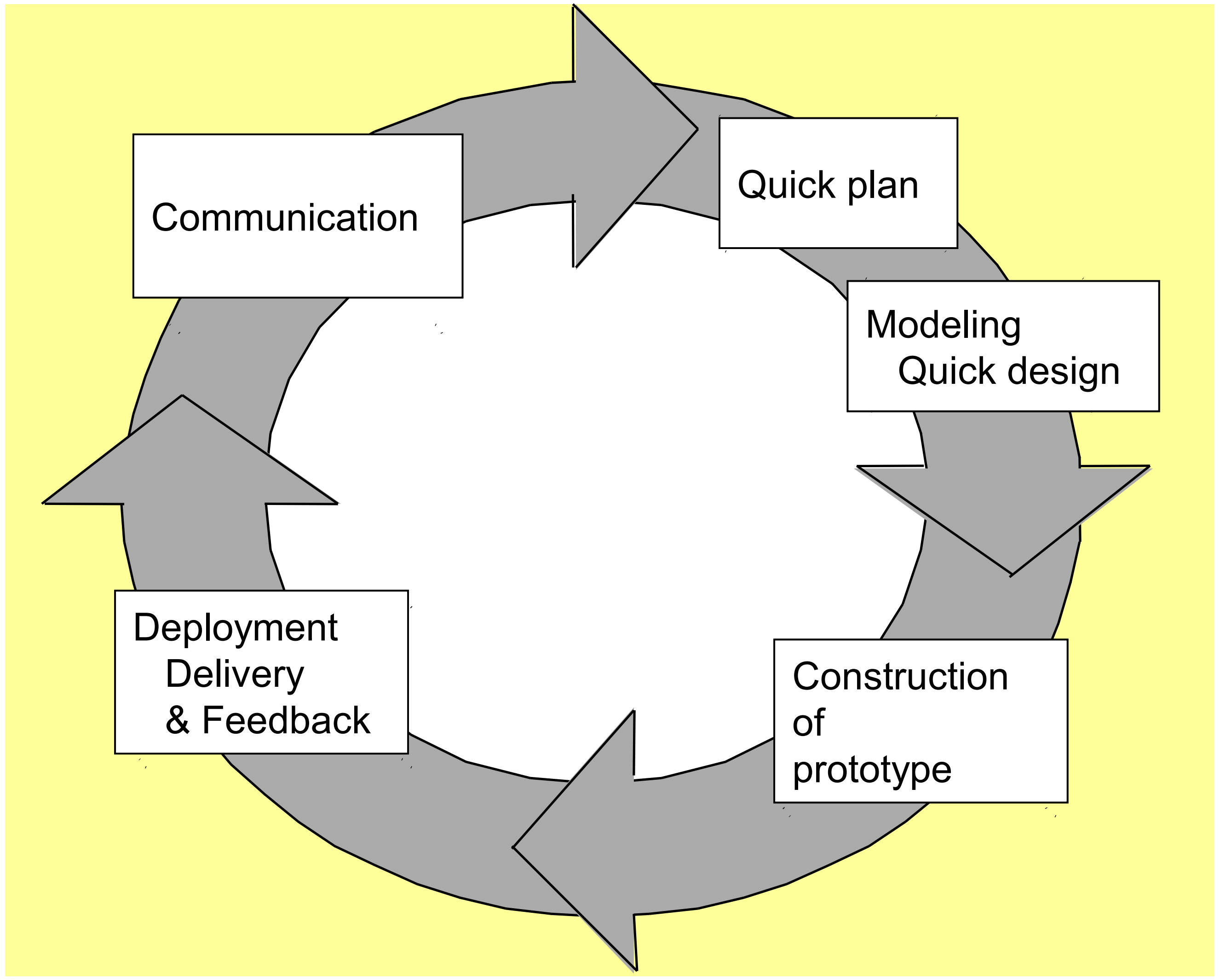
Win Win spiral model



2. Identify Stakeholders' win conditions
1. Identify next-level Stakeholders
3. - Reconcile win conditions. - Establish next level objectives, constraints, alternatives
7 Review, commitment
4 Evaluate product and process alternatives. Resolve Risks
6 Validate product and process definitions
5. Define next level of product and process - including partitions

Figure 1.8 WINWIN Spiral Model

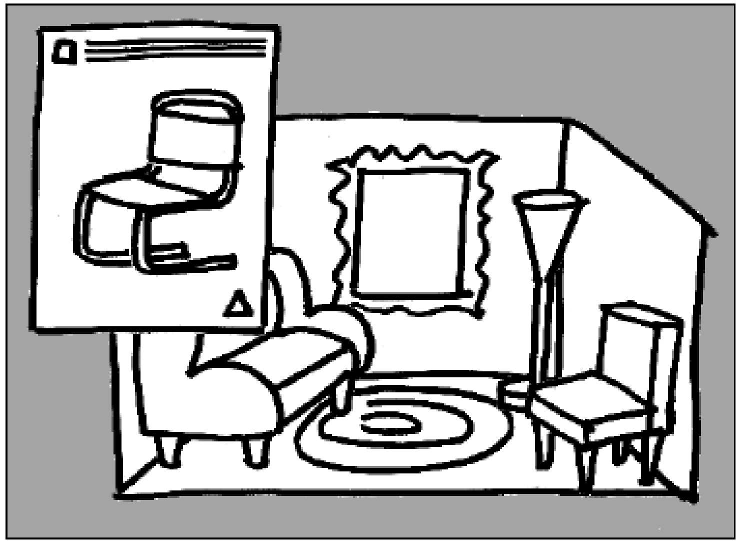
Prototyping

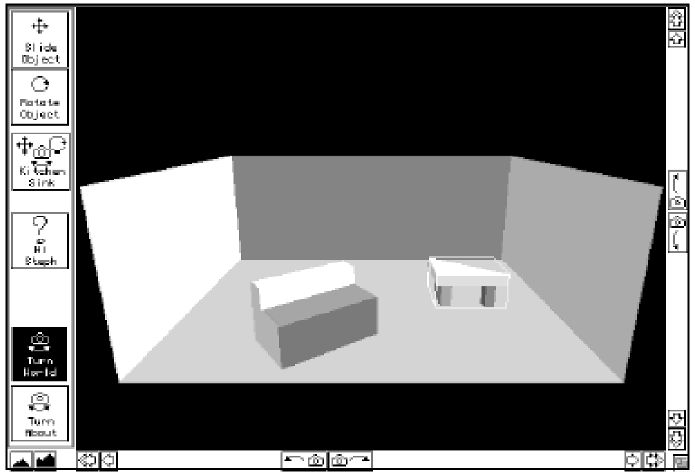


(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

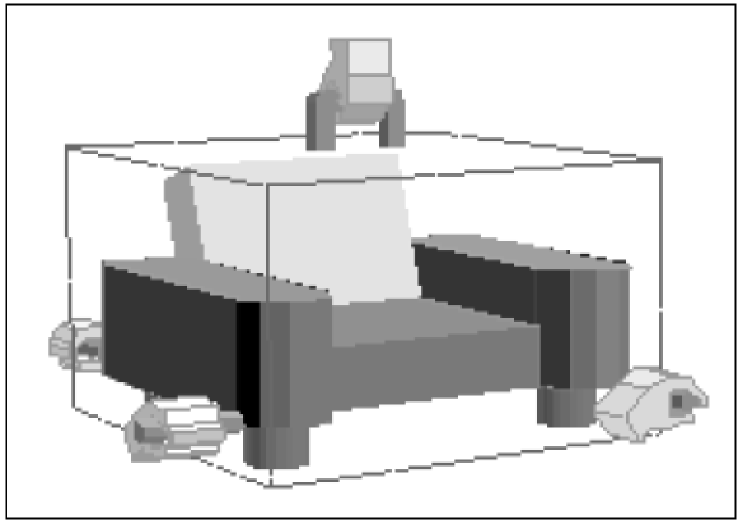
Fig : The prototyping model

Examples for prototype model


Example 1. Role prototype for 3D space-planning application [E1 Houde 1990].



Example 3. Implementation prototype for 3D space-planning application [E3 Chen 1990].


Example 2. Look-and-feel prototype for 3D space-planning application [E2 Houde 1990].

(Source: Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill)

Advantages

1. Iterative process facilitating enhancements.
2. Partial products can be viewed by the customer.
3. Flexibility of the product.
4. Customer satisfaction of the product.

Prototyping - Problems

1. Customers may press for immediate delivery of working but inefficient products
2. The developer often makes implementation compromises in order to get a prototype working quickly
3. No optimal solution.
4. Time consuming if the algorithm used is inefficient.
5. Iterative process continues forever which cannot be stopped at a particular stage.
6. Poorer documentation resulting in difficult maintenance.