Chapter #1

Product

Software and Software Engineering

Dr. Rehan Ali Syed

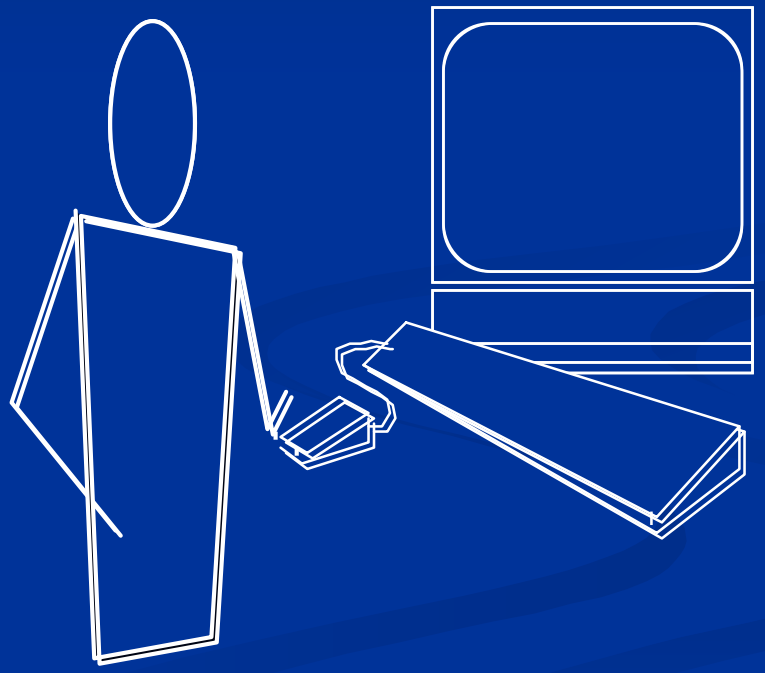
What is Software?

1. ***Instructions***
   - When executed provide desired features, function, and performance.
2. ***Data structures***
   - Enable the programs to adequately manipulate information.
3. ***Documentation***
   - Describes the operation and use of the programs.

What is Software?

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

- programs
- documents
- data ...



Software's Dual Role

- Software is a product
  - Delivers computing potential
  - Produces, manages, acquires, modifies, displays, or transmits information.
- Software is a vehicle for delivering a product
  - Supports or directly provides system functionality
  - Controls other programs (e.g., an operating system)
  - Effects communications (e.g., networking software)
  - Helps build other software (e.g., software tools)

Software! (Evaluations)

- The role of computer software has undergone significant change over a time span (since 50) years.
- 1970s and 1980s, "new industrial revolution.“
- During the 1990s , a Toffler said "power shift“ (Military, Government, Industrial and Economic domain.
- The lone programmer era has been replaced by a team of software specialists.

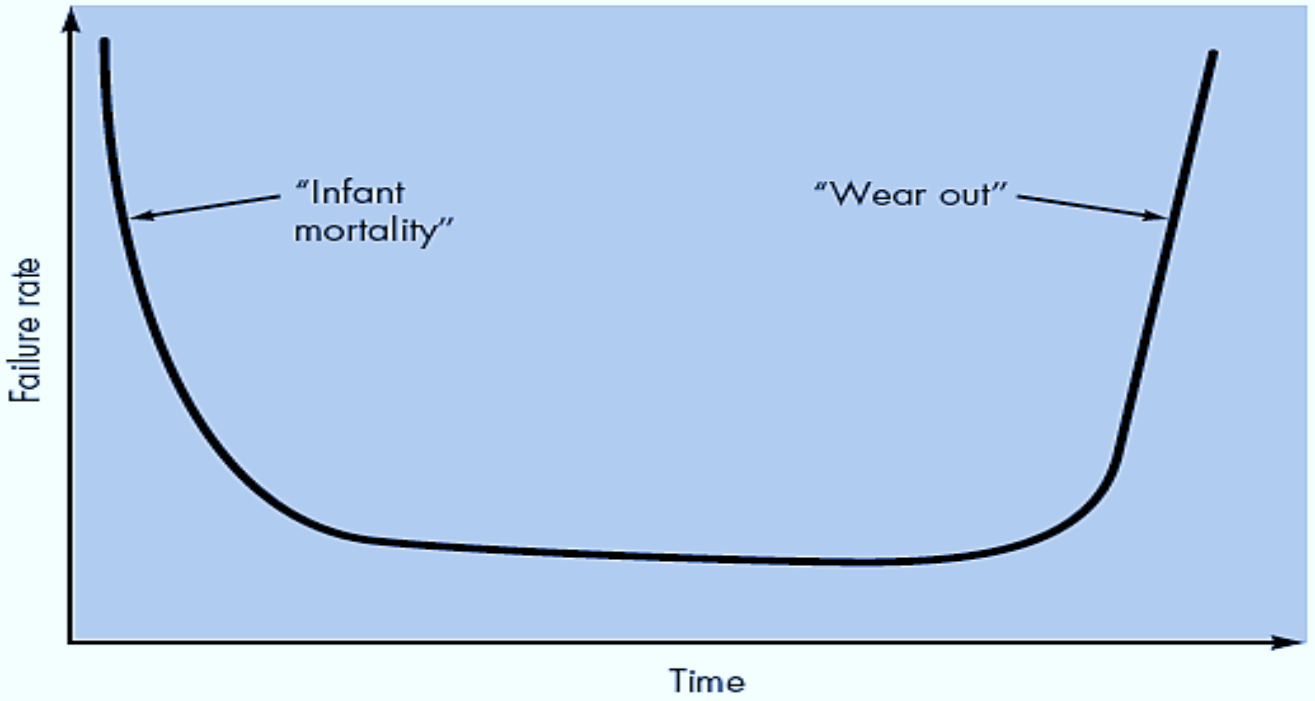
A Questions

- Why does it take so long to get software finished?
- Why are development costs so high?
- Why can't, we find all the errors before we give the software to customers?
- Why do we continue to have difficulty in measuring progress as software is being developed?

Software Characteristics

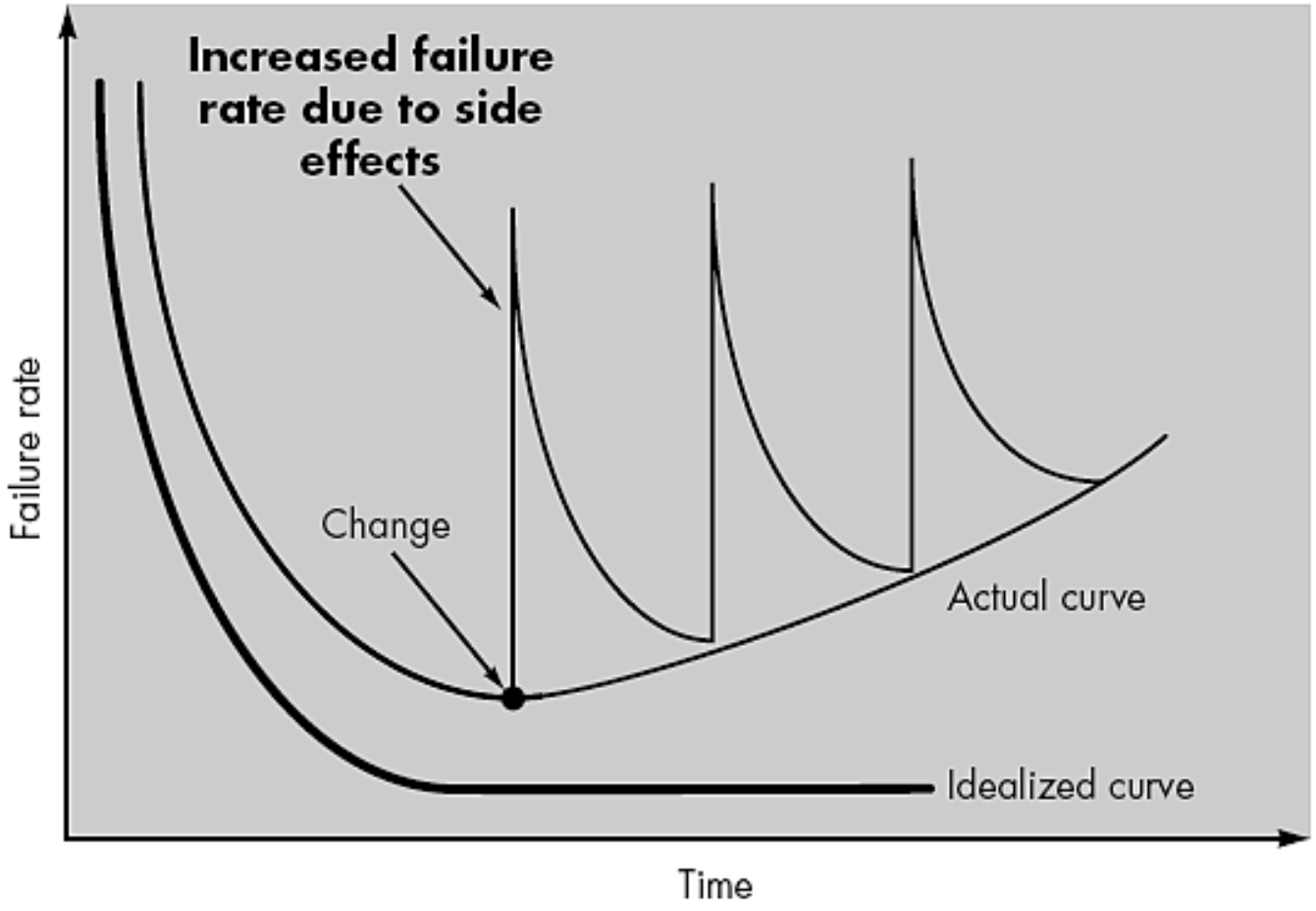
- Software is developed or engineered? it is not manufactured in the classical sense.
- Software doesn't "wear out."

FIGURE 1.1
Failure curve for hardware



Software Characteristics

FIGURE 1.2
Idealized and actual failure curves for software



Software Applications

- system software
- application software
- engineering/scientific software
- embedded software
- Web-Apps (Web applications)
- AI software

Software—New Categories

- Ubiquitous computing—wireless networks
- Netsourcing—the Web as a computing engine
- Open source—source code open to the computing community (a blessing, but also a potential curse!)
  - Data mining
  - Grid computing
  - Cognitive machines
  - Software for nanotechnologies

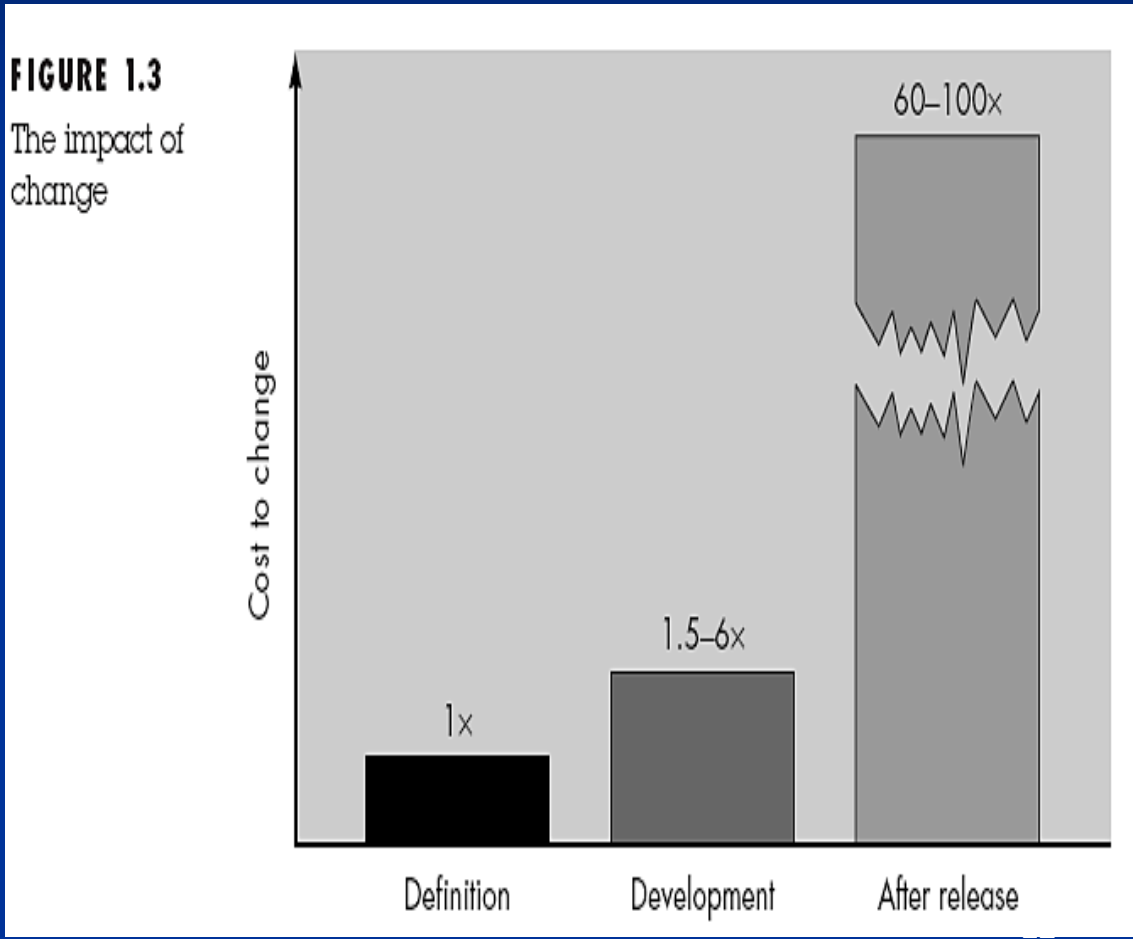
Legacy Software

Why must it change?

- software must be adapted to meet the needs of new computing environments or technology.
- software must be enhanced to implement new business requirements.
- software must be extended to make it interoperable with other more modern systems or databases.
- software must be re-architected to make it viable within a network environment.

Software Myths

1. **Management myths:**
2. **Customer myths**
3. **Practitioner's myths**



Chapter 2

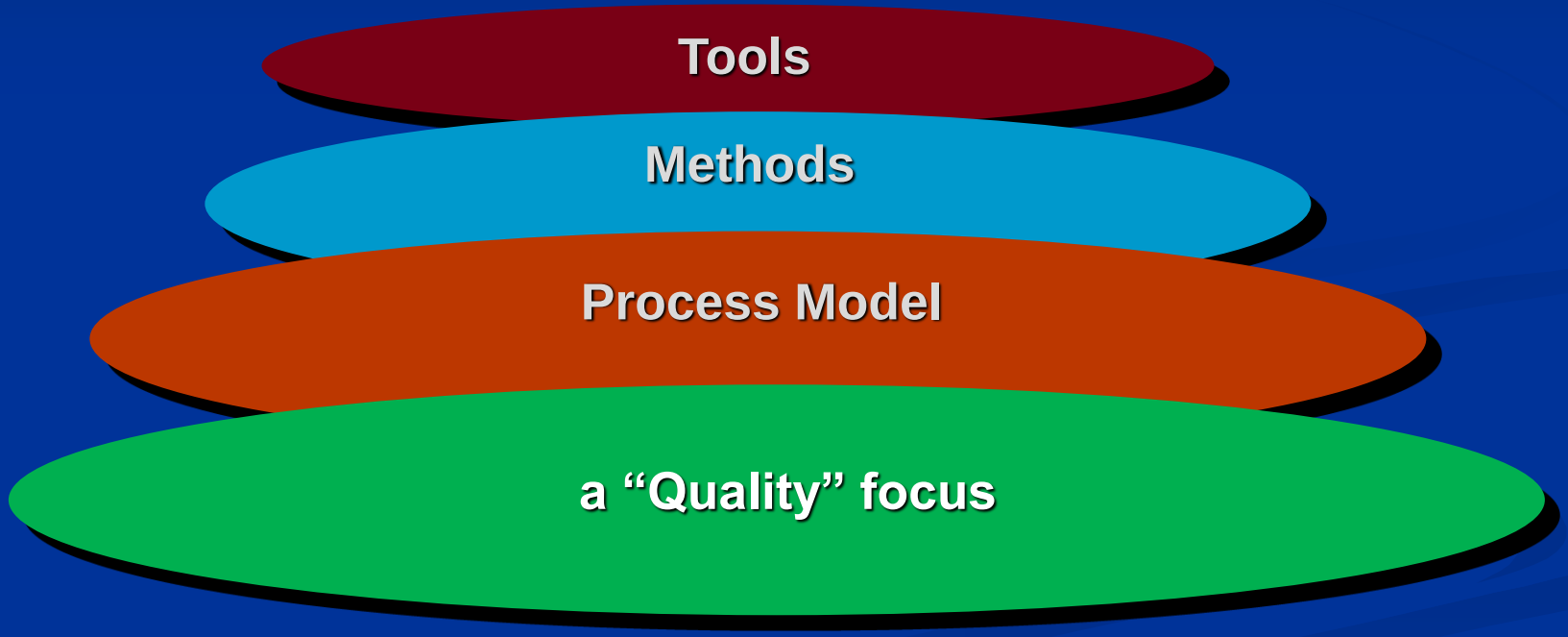
Process: A Generic View

A Generic View of Software Engineering

1. What is the problem to be solved?
2. What characteristics of the entity are used to solve the problem?
3. How will the entity (and the solution) be realized?
4. How will the entity be constructed?
5. What approach will be used to uncover errors that were made in the design and construction of the entity?
6. How will the entity be supported over the long term, when corrections, adaptations, and enhancements are requested by users of the entity.

A Layered Technology

Software Engineering



A Process Framework

- A small number of framework activities that are applicable to all software projects, regardless of their size or complexity,

Process framework

Framework activities

work tasks

work products

milestones & deliverables

QA checkpoints

- **Umbrella Activities**

A Process Framework

- Communication
- Planning
- Modeling
  - Analysis of requirements
  - Design
- Construction
  - Code generation
  - Testing
- Deployment

Umbrella Activities

- Software project management
- Formal technical reviews
- Software quality assurance
- Software configuration management
- Work product preparation and production
- Reusability management
- Measurement
- Risk management

The CMM

- The CMM defines each process area, in terms of “specific goals” and the “specific practices” required to achieve these goals.

- ***Specific goals*** establish the characteristics that must exist if the activities implied by a process area are to be effective.

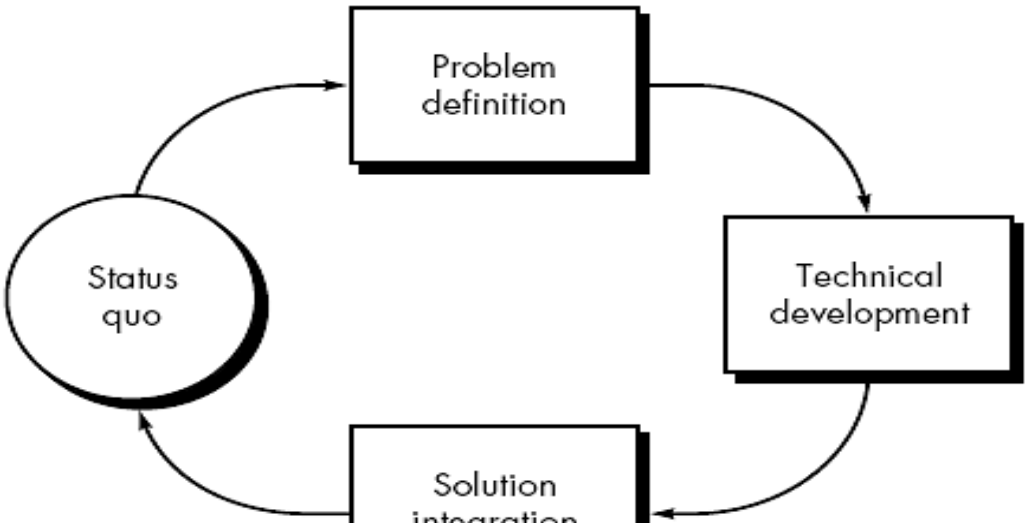
- ***Specific practices*** refine a goal into a set of process-related activities.

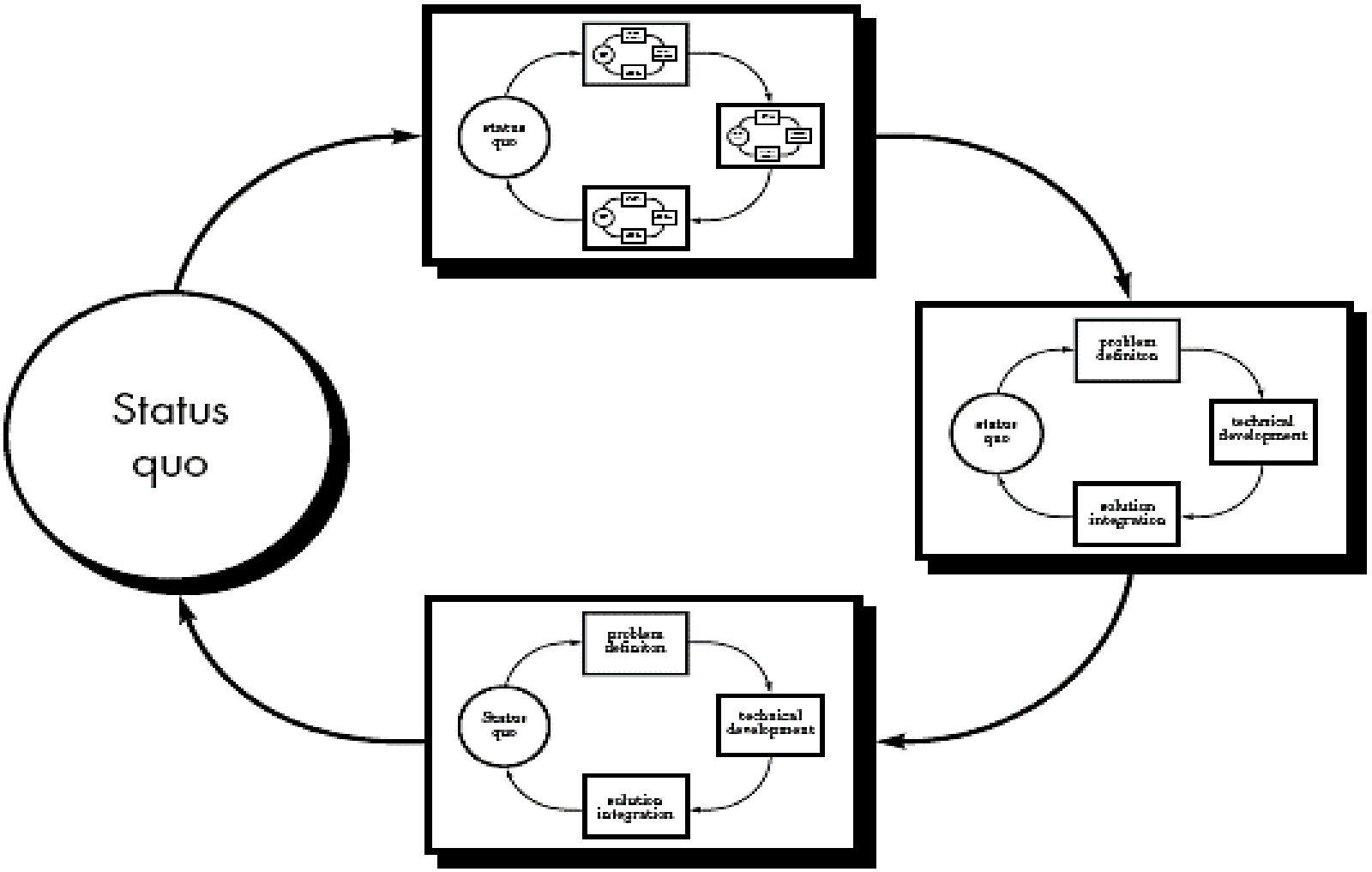
SOFTWARE PROCESS MODELS

- A process model for SE is chosen based on:
  - The nature of the project and application, the methods and tools to be used, and the controls and deliverables that are required.
  - L. B. S. Raccoon [RAC95] uses fractals as the basis for a discussion of the true nature of the software process.
  - Raccoon [RAC95] suggests a “Chaos model” coexist.

FIGURE 2.3

(a) The phases of a problem solving loop [RAC95]

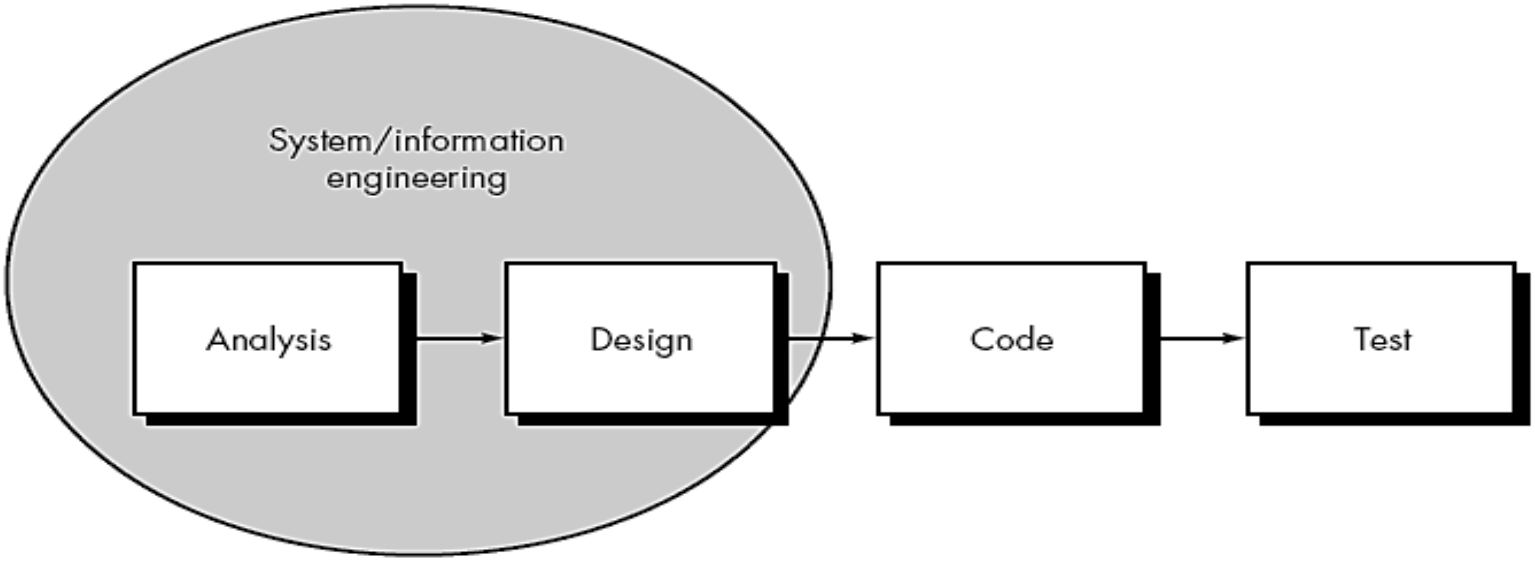



Status
quo
problem definition
technical development
solution integration
(b)

THE LINEAR SEQUENTIAL MODEL

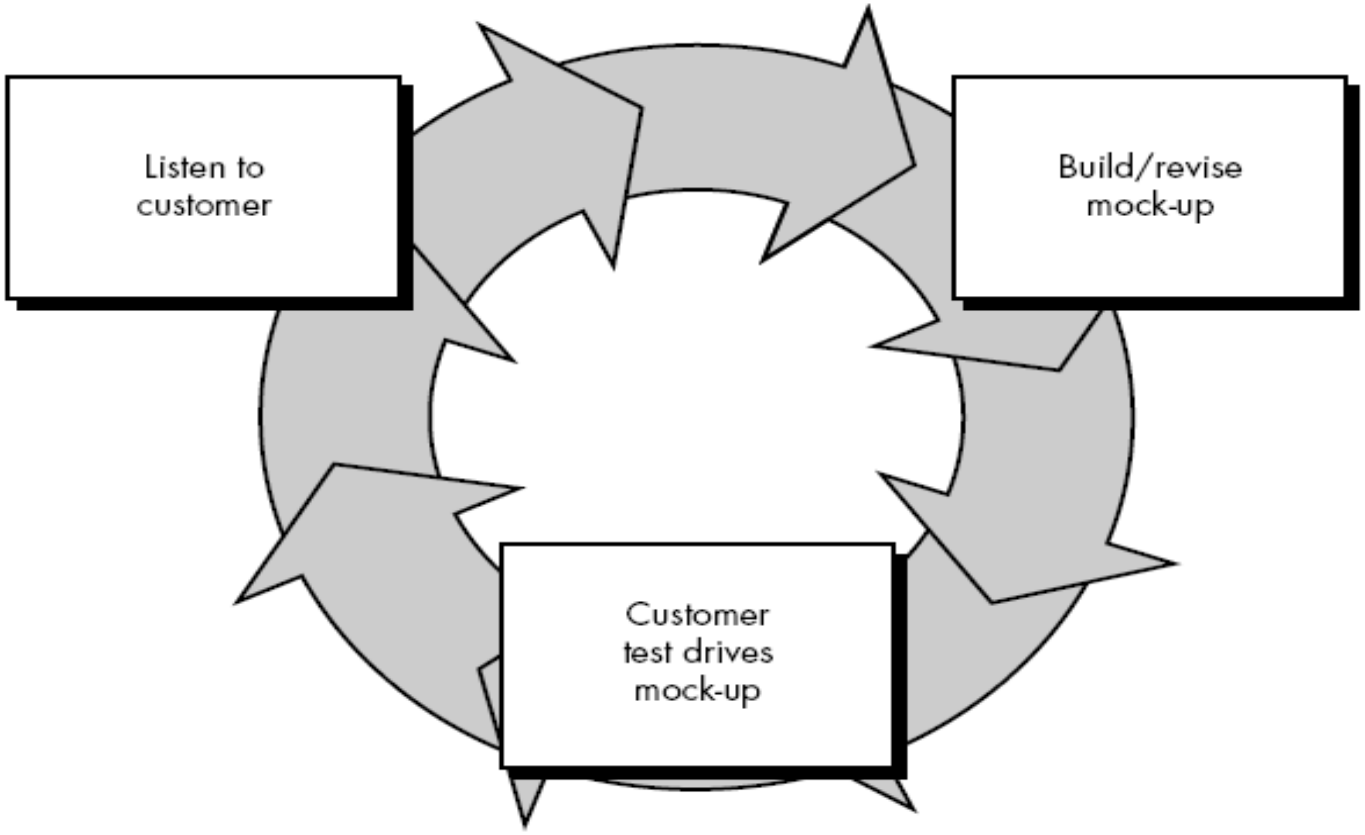
- Classic Life Cycle Model
- The Waterfall Model
- A model suggests a systematic, sequential approach to software development:
  - Analysis, Design, Coding, Testing, and Support.

FIGURE 2.4
The linear sequential model



THE PROTOTYPING MODEL

FIGURE 2.5
The prototyping paradigm



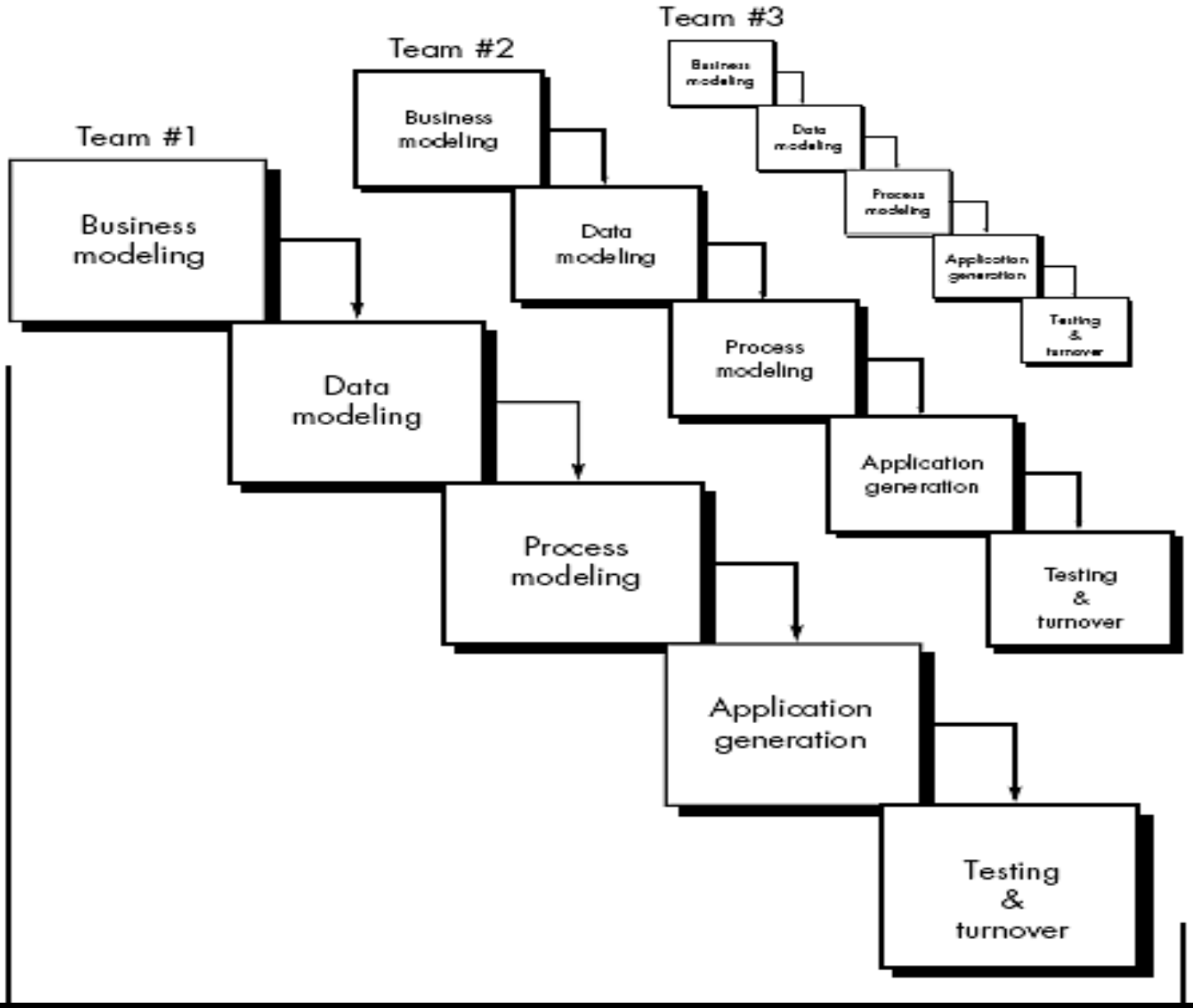
THE RAD MODEL

- RAD is an incremental software development process model that emphasizes an extremely short development cycle.
- High-speed" adaptation of the linear sequential model.
- In RAD (requirements are well understood and project scope is constrained)
  - The RAD process provides very short time periods (for the development).
- **Business modeling:**
  - The information flow among business functions is modeled
  - What information drives the business process?
  - What information is generated? Who generates it? Where does the information go? Who processes it?

THE RAD MODEL

- **Data modeling:**
  - Defined as part of the business modeling phase, a set of data objects that are needed to support the business. The characteristics (called attributes) of each object are identified and the relationships between these objects defined.
- **Application generation:**
  - RAD use of fourth generation techniques.
  - Not using conventional third generation programming languages
- **Testing and turnover**
  - The RAD approach has drawbacks.
    - RAD requires sufficient human resources to create the right number of RAD teams.
    - RAD requires developers and customers who are committed to the rapid-fire activities necessary to get a system complete in a much abbreviated time frame.

FIGURE 2.6
The RAD model



Evolutionary Software Process Models

- Business and product requirements often change as development proceeds,
  - Tight market deadlines make completion of a comprehensive software product impossible, but a limited version must be introduced to meet competitive or business pressure; a set of core product or system requirements is well understood, but the details of product or system extensions have yet to be defined.
- **The Incremental Model**
  - The incremental model combines elements of the linear sequential model (applied repetitively) with the iterative philosophy of prototyping.
- Each linear sequence produces a deliverable "increment" of the software [MDE93]. For example, **Word-processing Software.**

The Incremental Model

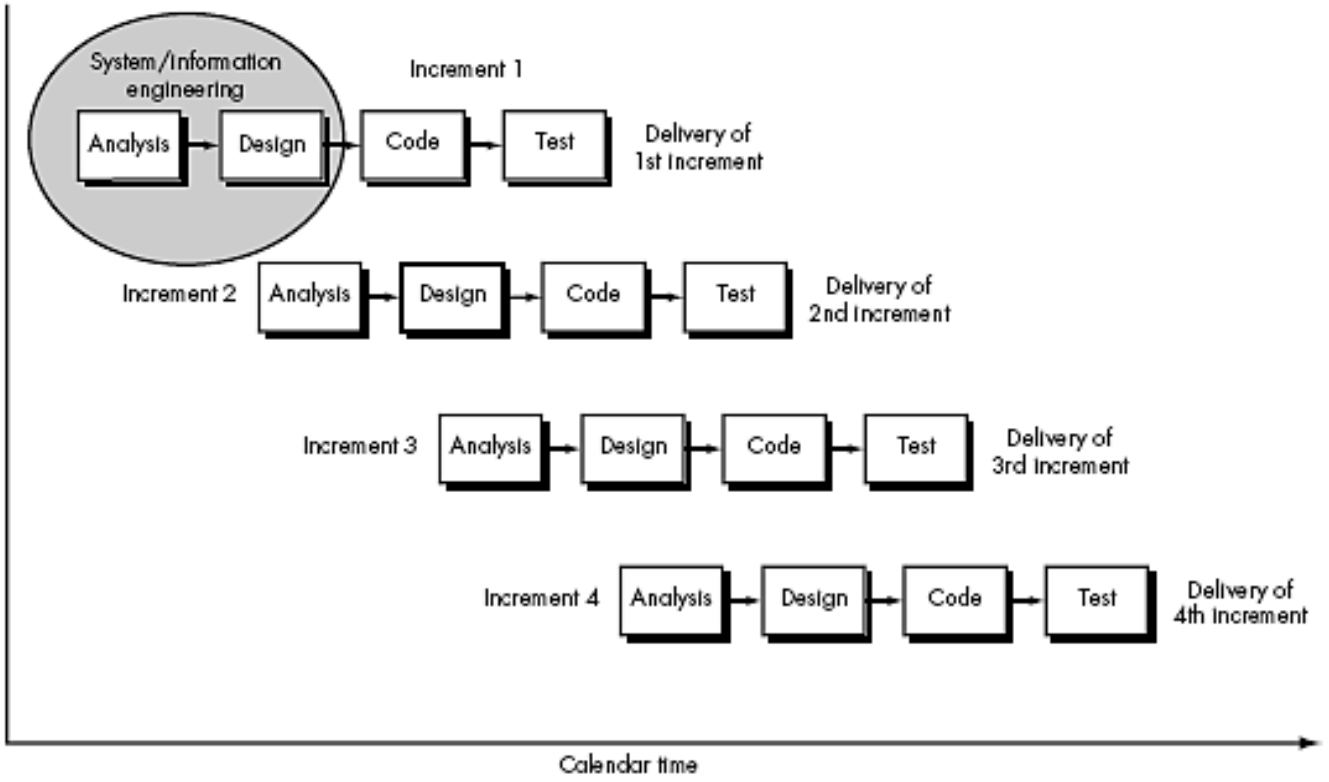


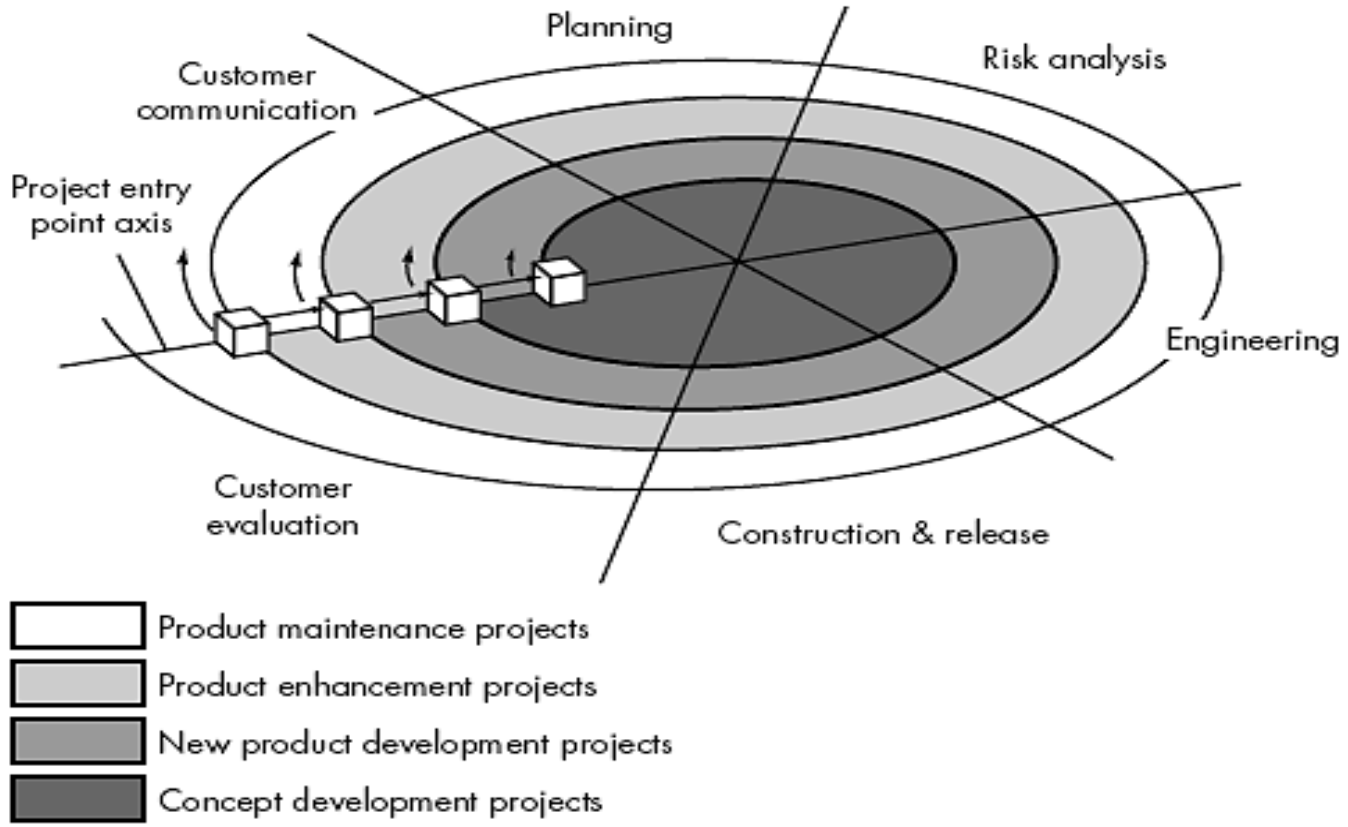
FIGURE 2.7
The incremental model

The Spiral Model

- The spiral model, originally proposed by Boehm [BOE88],
  - It is an evolutionary software process model.
  - Iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model.
- Employing the spiral model,
  - Software is developed in a series of incremental releases.
  - The incremental release might be a paper model or prototype.
- A spiral model is divided into a number of framework activities.

The Spiral Model

FIGURE 2.8
A typical spiral model



The Formal Methods Model

- A set of activities that leads to formal mathematical specification of computer software.
- Formal methods enable a software engineer to specify, develop, and verify a computer-based system by applying a rigorous, mathematical notation.
- The following concerns about its applicability in a business environment have been voiced:

1. The development of formal models is currently quite time consuming and expensive.
2. Few software developers have the necessary background to apply formal methods, extensive training is required.
3. It is difficult to use the models as a communication mechanism for technically unsophisticated customers.

Fourth Generation Techniques

- The term fourth generation techniques (4GT) encompasses a broad array of software tools that have one thing in common: each enables the software engineer to specify some characteristic of software at a high level.
- Tool then automatically generates source code based on the developer's specification.
- Nonprocedural languages for database query, report generation, data manipulation, screen interaction and definition, code generation; high-level graphics capability; spreadsheet capability, and automated generation of HTML and similar languages used for Web-site creation using advanced software tools.