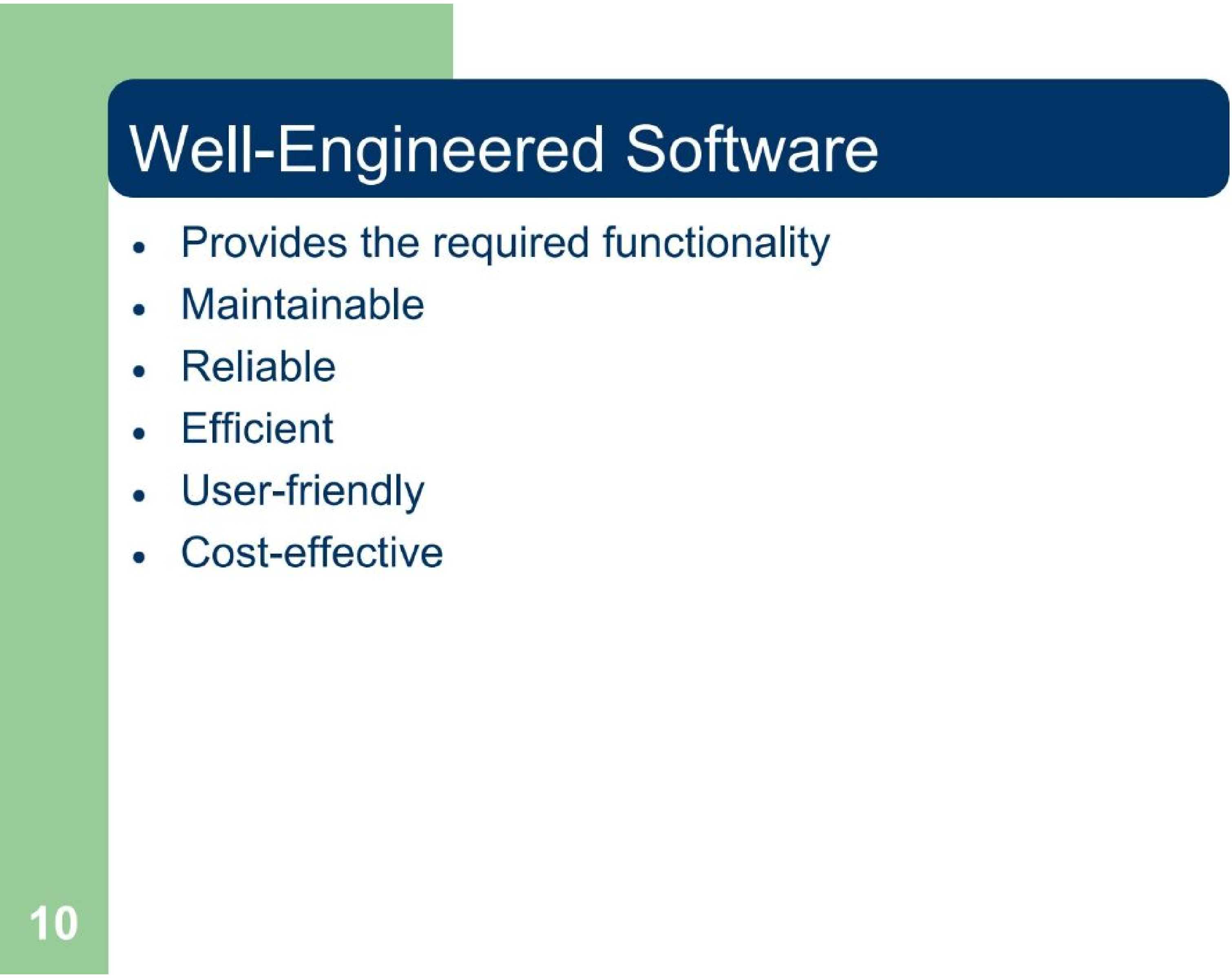
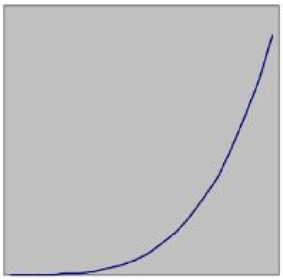


Well-Engineered Software

- Provides the required functionality
- Maintainable
- Reliable
- Efficient
- User-friendly
- Cost-effective

Well-Engineered Software - contd.

- These requirements may be conflicting:
  - Cost vs. Efficiency
  - Cost vs. Reliability
  - Efficiency vs. User-interface
- Law of diminishing returns.
- Challenge is to balance these requirements.



Software Life-Cycle Models

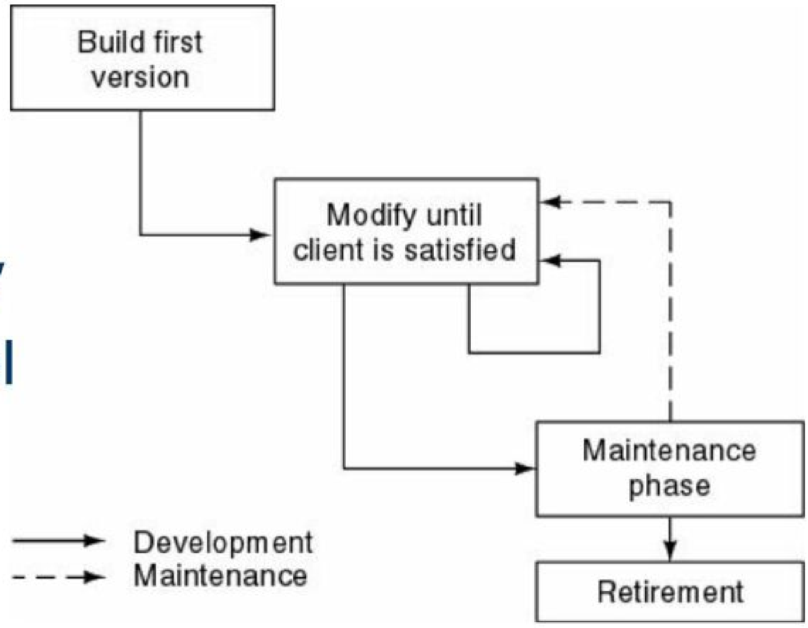
- The way you organize your activities
- The steps through which the product progresses
  - Requirements phase
  - Specification phase
  - Design phase
  - Implementation phase
  - Integration phase
  - Maintenance phase
  - Retirement

Software Processes

- Build-and-fix model
- Waterfall model
- Rapid prototyping model
- Incremental model
- Extreme programming
- Synchronize-and-stabilize model
- Spiral model
- Object-oriented life-cycle models
- Comparison of life-cycle models

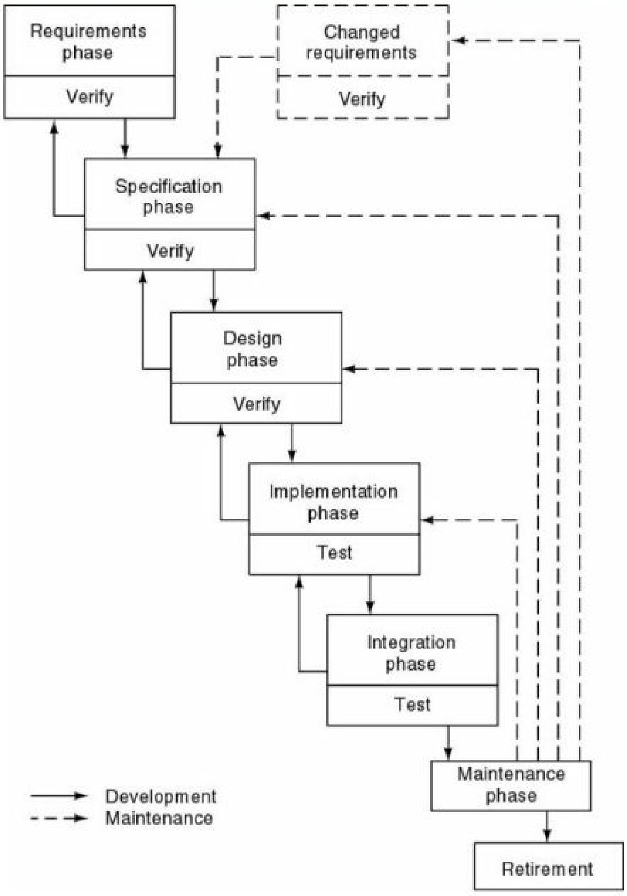
Build and Fix Model

- Problems
  - No specifications
  - No design
- Totally unsatisfactory
- Need life-cycle model
  - "Game plan"
  - Phases
  - Milestones



Waterfall Model (contd)

- Characterized by
  - With or without feedback loops
  - Documentation-driven
- Advantages
  - Documentation
  - Maintenance easier
- Disadvantages
  - Client feedback



Rapid Prototyping Model

- Linear model
- “Rapid”
- Horizontal versus vertical prototyping

Three Key Points

- Do not turn into product
- Rapid prototyping may replace specification phase—never the design phase
- Comparison:
  - Waterfall model—try to get it right first time
  - Rapid prototyping—frequent change, then discard