Software: The Product

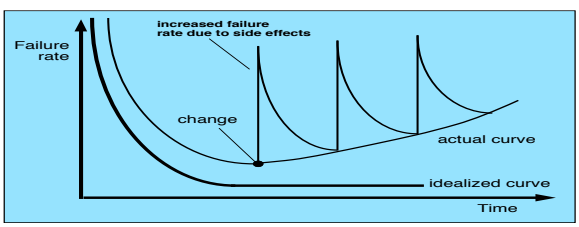
- Software is a product designed and built by software engineers.
- Software is important because it is everywhere.
- Software engineers have a moral and ethical responsibility.
- Software engineers care about whether product
  - embodies the technical elegance worthy of their effort
- Customers care about whether product:
  - meets their needs
  - is easy to use.

The Role of Modern Software

- Primary purpose of software is that of information transformer.
- Most modern software is developed by teams of software specialists.
- Software developer's concerns are the same as they were decades ago:
  - Why does software take so long to complete?
  - Why does software cost so much to produce?
  - Why can't all software errors be found and removed before software is delivered to the customer?

Software Construction

- Unlike artifacts in other engineering disciplines, software is developed, not manufactured
- Building a software product is more like constructing a design prototype.
- Opportunities for replication without customization are not very common.
- Software may deteriorate, but does not wear out.



Software Challenges and Myths

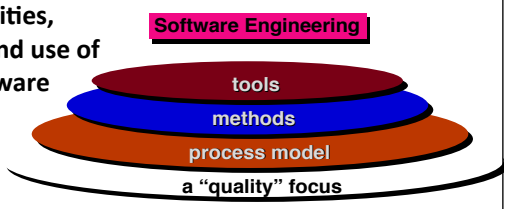
- Despite some spectacular software failures, there are also a large number of successes
- We need to produce more software to meet growing demand, while maintaining a (growing) body of existing software.
- Managers think that buying hardware and adding people will spontaneously solve all software development problems.
- Customers think that deploying new computer software makes all their other problems go away.
- Software engineers think that they can add quality and correctness later if they have to.

Software: The Process

- There is no one process model that works well for every project.
- Developers need to follow some systematic process.
- Software processes organize the work products needed during a software development project.
- Best indicator of how well a software process has worked is the quality of the deliverables produced.
- A well-managed software development process will produce high quality products on time and under budget.

Software Engineering - Process, Phases

- Software engineering encompasses a process, the management of activities, technical methods, and use of tools to develop software products.



- Software is engineered by applying three distinct goals in sequence:
  - Definition
  - Development
  - Support

7 Principles of Software Engineering Practice

- 1. Software system exists to provide value to its users
- 2. KISS (Keep It Simple, Stupid!)
- 3. Maintain the vision
- 4. What you produce, others will consume
- 5. Be open to the future
- 6. Plan ahead for re-use
- 7. Think! Place clear thought before action

The Software Process

- Hypothesis:
  - A quality process usually results in a quality product
- Common process framework
  - Framework activities (defined for each process)
  - Umbrella activities
    - Apply to all tasks, products, etc.
- Communication
  - Understand stakeholders' objectives
- Planning
- Modeling
- Construction
- Deployment

"Umbrella" Activities

- *Performed throughout any software development process*
- Software project management
- Formal technical reviews
- Software quality assurance
- Software configuration management
- Work product preparation and production
- Reusability management
- Measurement
- Risk management

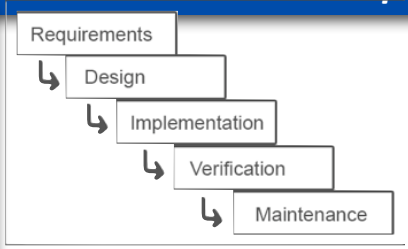
Adaptability of a Process Model

- Framework activities are always intended to be applied on every project
- Tasks (and degree of rigor) for each activity will vary based on:
  - The type of project (an "entry point" to the model)
  - Characteristics of the project
  - Common sense judgment
  - Concurrence of the project team

Software Process Models

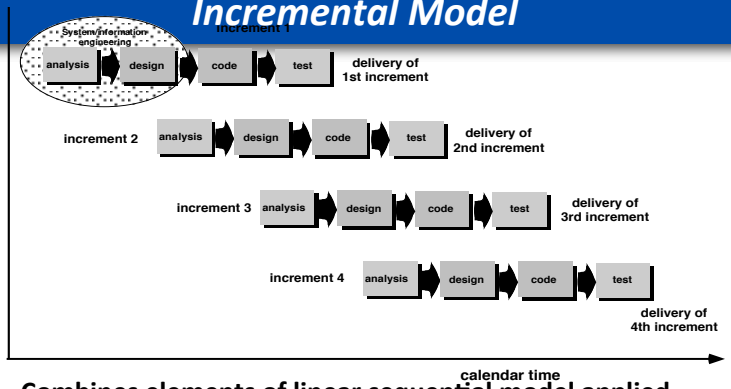
- Common features
  - Possess an overall strategy
  - Usually divide development into phases
  - Phases are not independent of each other
- Waterfall (linear sequential) model
- Incremental process models
  - Incremental
  - Rapid Application Deployment Model (RAD)
- Evolutionary models
  - Prototyping Model
  - Spiral model
  - Concurrent development model
- Specialized models

Linear Sequential Model



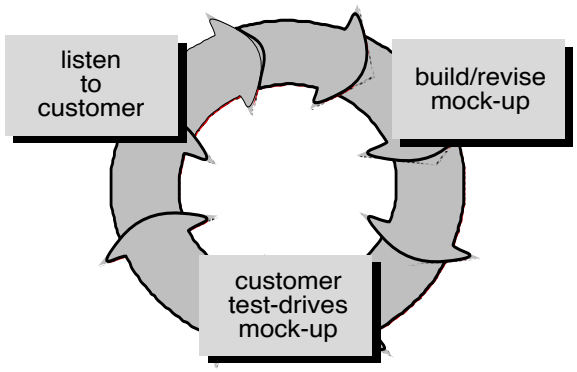
- Also known as the classic life cycle or waterfall model.
- System development proceeds though phases.
- Re-visiting a phase (i.e., going back) means a process failure is indicated
  - e.g., re-visiting requirements during design means that requirements analysis should have been more thorough
- Good model to use when requirements are very well understood and risk is low.

Incremental Model



- Combines elements of linear sequential model applied repetitively with the iterative philosophy of the prototyping model.
- Each increment produces a working version of a software product with increasing functionality.
- There is no throwaway code.

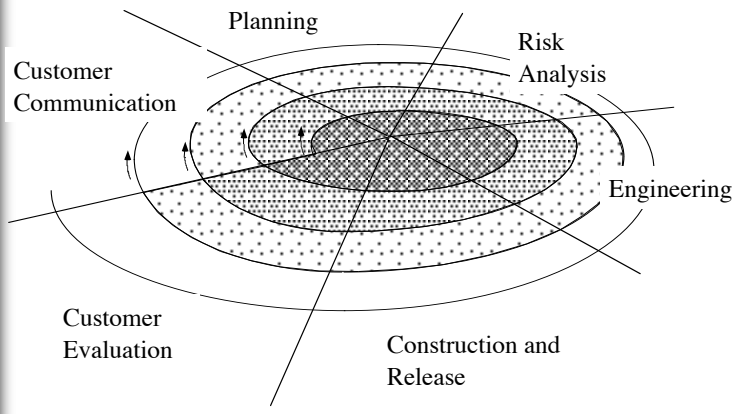
Prototyping Model



Prototyping Model

- Use when customer has legitimate needs, but can't articulate the details.
- Small mock-up of a working system is developed and presented to the customer.
- First prototype may be discarded or extended based on the customer's feedback.
- Particularly useful for interactive interfaces

Spiral Model



Evolutionary Model - Spiral

- **The spiral model**
  - **Combines iterative nature of prototyping with systematic control of the linear sequential model.**
  - **Assessment of both management and technical risks is performed as each incremental release is completed.**
  - **Each iteration results in a collection of features that operates correctly and meets some customer needs**
  - **Each iteration's products are "expendable" but should never throw out more than the most recent iteration**

Questions to consider...

- **Identify a software system that you consider to be of good quality? Why do you think so?**
- **Identify a software system that you consider to be of poor quality? Why do you think so?**
- **Do you have experience with any of these models? Was using the model appropriate to the kind of development?**
- **What are the main differences between an evolutionary model and a prototyping model?**