Software Engineering

- Designing, building and maintaining large software systems

Objectives

- To define software engineering and explain its importance
- To discuss the concepts of software products and software processes
- To explain the importance of process visibility
- To introduce the notion of professional responsibility

Topics covered

- Software products
- The software process
- Boehm’s spiral model
- Process visibility
- Professional responsibility

Software engineering

- The economies of ALL developed nations are dependent on software
- More and more systems are software controlled
- Software engineering is concerned with theories, methods and tools for professional software development
- Software engineering expenditure represents a significant fraction of GNP in all developed countries

Software costs

- Software costs often dominate system costs. The costs of software on a PC are often greater than the hardware cost
- Software costs more to maintain than it does to develop. For systems with a long life, maintenance costs may be several times development costs
- Software engineering is concerned with cost-effective software development

Software products

- Generic products
  - Stand-alone systems which are produced by a development organization and sold on the open market to any customer
- Custom products
  - Systems which are commissioned by a specific customer and developed specially by some contractor. Sometimes involves hardware/software co-design.
- Each type of product has different goals and constraints, and, therefore, uses different development processes.
Software product attributes

- Maintainability
  - can evolve to meet changing requirements
- Dependability
  - doesn’t cause damage in the event of failure
- Efficiency
  - uses system resources wisely
- Usability
  - appropriate user interface and documentation
- What are some other attributes?

Importance of product characteristics

- The relative importance of these characteristics depends on the product and the environment in which it is to be used
- In some cases, some attributes may dominate
  - In safety-critical real-time systems, key attributes may be dependability and efficiency
- Costs tend to rise exponentially if very high levels of any one attribute are required

The software process

- Structured set of activities required to develop a software system
  - Specification
  - Design
  - Validation
  - Evolution
- Activities vary depending on the organization and the type of system being developed
- Must be explicitly modeled if it is to be managed

Process characteristics

- Understandability
  - Is the process defined and understandable?
- Visibility
  - Is the process progress externally visible?
- Supportability
  - Can the process be supported by tools?
- Acceptability
  - Is the process acceptable to those involved in it?

Generic software process models

- The waterfall model
  - separate, distinct, and sequential development phases
- Evolutionary development
  - development phases are interleaved
- Formal transformation
  - a mathematical system model is formally transformed to an implementation
- Reuse-based development
  - The system is assembled from existing components
Waterfall model

Waterfall model phases
- Requirements analysis and definition
- System and software design
- Implementation and unit testing
- Integration and system testing
- Operation and maintenance
- The drawback of the waterfall model is the difficulty of accommodating change after the process is underway

Evolutionary development

Evolutionary development
- Throw-away prototyping
  - Objective is to understand the system requirements.
  - Should start with poorly understood requirements
- Exploratory prototyping
  - Objective is to work with customers and to evolve a final system from an initial outline specification.
  - Should start with better-understood requirements

Evolutionary development
- Problems
  - Lack of process visibility
  - Systems are often poorly structured
  - Special skills (e.g. in languages for rapid prototyping) may be required
- Applicability
  - For small or medium-size interactive systems
  - For parts of large systems (e.g. the user interface)
  - For short-lifetime systems

Risk management
- Perhaps the principal task of a manager is to minimize risk
- The ‘risk’ inherent in an activity is a measure of the uncertainty of the outcome of that activity
- High-risk activities cause schedule and cost overruns
- Risk is related to the amount and quality of available information. The less information, the higher the risk
Process model risk problems

- **Waterfall**
  - High risk for new systems because of specification and design problems
  - Low risk for well-understood developments using familiar technology
- **Prototyping**
  - Low risk for new applications because specification and program stay in step
  - High risk because of lack of process visibility
- **Transformational**
  - High risk because of need for advanced technology and staff skills

Hybrid process models

- Large systems are usually made up of several sub-systems
- The same process model need not be used for all subsystems
- Prototyping for high-risk specifications
- Waterfall model for well-understood developments

Spiral model of the software process

Phases of the spiral model

- **Objective setting**
  - Specific objectives for the project phase are identified
- **Risk assessment and reduction**
  - Key risks are identified, analyzed and information is sought to reduce these risks
- **Development and validation**
  - An appropriate model is chosen for the next phase of development.
- **Planning**
  - The project is reviewed and plans drawn up for the next round of the spiral

Spiral model flexibility

- Well-understood systems (low technical risk) - Waterfall model. Risk analysis phase is relatively cheap
- Stable requirements and formal specification. Safety criticality - Formal transformation model
- High UI risk, incomplete specification - prototyping model
- Hybrid models accommodated for different parts of the project

Spiral model advantages

- Focuses attention on early error elimination
- Puts quality objectives up front
- Integrates development and maintenance
- Provides a framework for hardware/software development (i.e., not software-specific)
- Focuses attention on reuse options
### Spiral model problems

- Contractual development often specifies process model and deliverables in advance
- Requires risk assessment expertise
- Needs refinement for general use

### Process visibility

- Software systems are intangible so managers need documents to assess progress
- However, this may cause problems
  - Timing of progress deliverables may not match the time needed to complete an activity
  - The need to produce documents constrains process iteration
  - The time taken to review and approve documents is significant
- Waterfall model is still the most widely used deliverable-based model

### Key points

- Software engineering is concerned with the theories, methods and tools for developing, managing and evolving software products
- Software products consist of programs and documentation. Product attributes are maintainability, dependability, efficiency and usability
- The software process consists of those activities involved in software development

- The waterfall model considers each process activity as a discrete phase
- Evolutionary development considers process activities as concurrent
- The spiral process model is risk-driven
- Process visibility involves the creation of deliverables from activities