Introduction to Software Engineering

Tom Verhoeff
TU/e
Faculty of Mathematics & Computing Science

The Problem

Software products (often?) suffer(ed?) from

- bugs: low quality
- high cost: budget overrun
- late delivery: schedule overrun

History

- 1968 NATO Conference: Software Crisis
- Apply engineering to software development

Goal

Make quality software, on time, within budget

- large & complex systems
- built by teams
- exist in many versions & variants
- last for many years
- undergo frequent changes

IEEE Definition of SE

Application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software

- The IEEE develops and maintains numerous internationally-accepted standards for SE

Maintenance

Most software

- lives longer than planned
- undergoes more changes than planned

- Corrective maintenance
- Adaptive maintenance
- Perfective maintenance (e.g. enable reuse)

Nature of Software

- Intangible
- Malleable
- Intellecutally intensive
- Trivial replication

Planning Δ

Any two characteristics constrain the third:

- Size
- Cost (time, money)
- Quality

Metrics

- Measure size
- Measure cost
- Measure quality

Management Δ

- Plan: who does what, when, how; dependencies (use previous measurements)
- Execute
- Monitor: measure, adjust, handle risks

Human Factors

- Limited productivity: work in teams
- Limited oversight: divide and conquer
- Limited accuracy: verify work early and often
- Limited communication: write documentation

Product, Process, Documentation

- Product
- Product documentation, verification
- Process (awareness)
- Process documentation, verification

Life-Cycle vs Process

- Life-Cycle: various incarnations of product
- Process: tasks and disciplines to do work

Waterfall

- Requirements
- Design
- Production
- Transfer
- Operation & Maintenance

Alternatives

- Incremental
- Spiral
- Evolutionary
- 2D (Unified Process)

Management Issues

- Planning
- Configuration Management
- Quality Assurance

Drivers

- Documentation driven
- Risk driven
- Customer/requirements driven

Models & Prototypes

- Formal models
- Prototypes: from paper mock-up to executable

What Else?

- Software Qualities: Often “invisible”
- Software Engineering Principles

Keep in Mind

- … that you will be applying large-scale SE methods in a small-scale software project
- … that many software qualities focus on maintenance, and seem much less relevant when just getting something new to “work”
- … that it is important, but difficult, to measure and predict such aspects as size, cost, and quality of software