August 26, 2019 Lecture

GMU Fall 2019 CS 321

Connor Baker

Compiled on August 26, 2019 at 4:55pm

What is Software Engineering

According to the IEEE (IEEE93a):

> Software Engineering:
>
> 1. The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
> 2. The study of approaches as in 1.

You must be both disciplined and agile.

So why do we need Software Engineering?

- It provides for a systematic and cultured approach to the process of building software
  - This creates an approach which has the ability to scale.

The overarching goal is

> To produce a quality product, on time, within budget, and of value to the customers and users.

Software is written *for* people. As such, the end user and customer should be taken into consideration during development, not just marketing for release.

The customer often has tacit knowledge – that's the "I'll know it when I see it" that you might hear later in the workplace. Part of our job as Software Engineers is to extract that tacit knowledge.

Software Engineering Rationale

For any software project, a set of Software Engineering (SE) *objectives* should be specified.

Achievement of those objectives is governed by the adoption of SE *principles* in the *process*.

Use of such SE principles will induce desirable *attributes* in the *product*.

Four P's of Software Engineering

1. People
2. Process
3. Project
   - This typically involves managing the process – acquiring tools, money, Subject Matter Experts, etc.
4. Product

A Generic Software Engineering Process

The order of these events depends largely on the development process used.

1. Communication
2. Planning
3. Modeling
4. Construction
5. Deployment
6. Evolution

Umbrella Activities

These activities occur constantly through the development and software lifecycle.

- Project tracking and control
- Risk management
- Software Quality Assurance (QA)
- Technical reviews
- Measurement
  - Without some measure (a metric) we cannot hope to improve something
- Software Configuration Management (SCM)
- Documentation and other artifacts

The Essence of Software Engineering Practice

- Understand the problem (communication and analysis)
- Plan a solution (modeling and software design)
- Carry out the plan (code generation)
- Examine the result for accuracy (testing and QA)

Principles of Software Engineering Practice

According to David Hooker:

1. Deliver value
2. Keep it simple
3. Maintain the vision
4. Remember that others will consume what you produce

5. Be open the future
6. Plan ahead for reuse
7. Think

Characteristics of Software

- Software is engineered
  - It is *not* manufactured
- Software does not wear out
  - It *rots*
- Software is complex

Some Myths about Software

- If we fall behind schedule, we can add more programmers
- A general statement of objectives is sufficient to begin writing programs – we can fill in the details later
- Software requirements continually change, but can be easily accommodated because software is flexible
- The sooner we get to coding, the sooner we will get done

Remember: focus on reality as you navigate your way through software engineering decisions.