Cornell University
Computing and Information Science

CS 5150 Software Engineering

2. Steps in the software development process

William Y. Arms

Software Process

Fundamental Assumption:

Good processes lead to good software

Good processes reduce risk

Good processes enhance visibility

Good processes enable teamwork

Variety of Software Processes

Software products are very varied...

Therefore, there is no standard process for all software engineering projects

BUT successful software development projects all need to address similar issues.

This creates a number of **process steps** that should be part of all software projects.

Basic Steps in all Software Development

Feasibility and planning

Requirements

User interface design

System and program design

Implementation (coding)

Acceptance and release

Operation and maintenance

These steps may be repeated many times during the development cycle

It is essential to distinguish among these steps and to be clear which you are are doing at any given moment.

Note

- Considerations of testing, security and performance are part of many of these steps.

Feasibility

A **feasibility study** precedes the decision to begin a project.

- What is the scope of the proposed project?
- Is the project technically feasible?
- What are the projected benefits?
- What are the costs, timetable?
- Are the resources available?
- What are the risks and how can they be managed?

A feasibility study leads to a **decision**: go or no-go.

Requirements

Requirements define the function of the system **from the client's viewpoint**.

The requirements establish the system's functionality, constraints, and goals by consultation with the client, customers, and users.

They must be developed in a manner that is understandable by both the client and the development staff.

This step is sometimes divided into:

- Requirements analysis
- Requirements definition
- Requirements specification

The requirements may be developed in a self-contained study, or may emerge incrementally.

Failure to agree on the requirements and define them adequately is one of the biggest cause of software projects failing.

User Interface Design

Usability is of great importance in many modern applications and software systems. That requires good user interface design.

User interfaces need to be evaluated with users.

This requires iterative development:

- Design the user interface
- Test with users
- Revise the user interface
- *Repeat*

System and Program Design

Design describes the system **from the software developers' viewpoint**

System design:

Establish a system architecture, both hardware and software, that matches the requirements

Program design:

Represent the software functions in a form that can be transformed into one or more executable programs

Preliminary user testing is often carried out as part of the design step.

Models are used to represent the requirements, system architecture, and program design. This course teaches the basic concepts of the Unified Modeling Language (UML).

Implementation

Implementation (coding)

The software design is realized as a set of programs or program units.

These software components may be written by the development team, acquired from elsewhere, or modified from existing components.

Program testing

Program testing by the development staff is an integral part of implementation.

Individual components are tested against the design.

The components are integrated and tested against the design as a complete system.

Acceptance and Release

Acceptance testing

The system is tested against the requirements by the client, often with selected customers and users.

Delivery and release

After successful acceptance testing, the system is delivered to the client and released into production or marketed to customers.

Operation and Maintenance

Operation:

The system is put into practical use.

Maintenance:

Errors and problems are identified and fixed.

Evolution:

The system evolves over time as requirements change, to add new functions, or adapt to a changing technical environment.

Phase out:

The system is withdrawn from service.

This is sometimes called the **Software Life Cycle**

Quality Control in all Software Development

- Validating the **requirements**
- Validating the system and program **design**

- **Usability** testing
- **Program** testing
- **Acceptance** testing

- Bug fixing and **maintenance**

Some of these steps will be repeated many times during the life of the system

Categories of Testing

The term "**testing**" is used in several different contexts, which are easily confused:

User testing

Versions of the user interface are tested by **users**. Their experience may lead to changes in the requirements or the design.

Program testing

The **development team** tests components individually (unit testing) or in combination (system testing) against the design to find bugs, etc.

Acceptance testing

The **client** tests the final version of the system or parts of the system against the requirements.

Sequence of Processes

Every software project will include these basic steps, **in some shape or form**, but:

- The steps may be formal or informal
- The steps may be carried out in various sequences

Software Development Processes

Major alternatives

In this course, we will look at four categories of software development processes:

- **Iterative refinement:**

  Go quickly through all the steps to create a rough system, then repeat them to improve the system.

- **Spiral:**

  A variant of iterative refinement in which new and updated components are added to the developing system as they are completed.

- **Agile development:**

  Small increments of software are developed in a sequence of sprints, each of which creates deployable code.

- **Waterfall model:**

  Complete each process step before beginning the next.

Heavyweight and Lightweight Software Development

In a **heavyweight process**, the development team works through the steps slowly and systematically, with the aim of fully completing each step and delivering a complete software product that will need minimal changes and revision.

Example: Modified Waterfall Model

In a **lightweight process**, the development team releases working software in small increments, and develops the plans incrementally, based on experience. Each increment includes all the process steps. There is an expectation that changes will be made based on experience.

Example: Agile Software Development

Heavyweight and Lightweight Methodologies

Heavyweight	⟷	Lightweight
Processes and tools	⟷	Individuals & interactions
Specification	⟷	Working software
Client negotiation	⟷	Client collaboration
Following a plan	⟷	Responding to change

Based on the Manifesto for Agile Software Development:
http://agilemanifesto.org/

Deliverables

Deliverables

A deliverable is some work product that is delivered to the client.

- In a heavyweight process, each process step creates a deliverable, usually documentation, e.g., a requirements specification.
- In a lightweight process, the deliverables are incremental working code, with minimal supporting documentation.

For the course projects, the deliverables include three presentations and a report to the client and course team at each major milestone.

From an Old Exam Question

An online information system is being developed using a modified version of the Waterfall model. It is likely to be based on web technology.

(i) How much should the choice of technology be considered during the feasibility study?

(ii) In how much detail should the choice of technology be specified during the requirements phase of the project?

(iii) At what stage should the decision be made to use an Apache Web Server 2.0 with Tomcat 4.1?

From an Exam Question (Answer)

(i) How much should the choice of technology be considered during the feasibility study?

During the feasibility study it is important to know that the project is technically feasible.

This can be achieved by identifying one possible technical approach and analyzing it sufficiently to show that it is capable of fulfilling the requirements of the system. It can also be used to estimate costs of hardware, software, etc.

However, this is only a possible approach. When the system design is carried out in detail, totally different technology may be chosen (e.g., not web-based).

From an Exam Question (Answer)

(ii) In how much detail should the choice of technology be specified during the requirements phase of the project?

A requirement is a statement of need as expressed by a client.

The client's requirements are that the system collects certain data, saves it, and carries out specified processes, e.g., displaying it, performing calculations, etc.

The decision of how to store and manipulate the data (e.g., using specific web technology) is usually not a requirement of the client. It comes later, as part of the design.

From an Exam Question (Answer)

(iii) At what stage should the decision be made to use an Apache Web Server 2.0 with Tomcat 4.1?

This is part of the system design.