CSC 308 Lecture Notes Weeks 1 and 2
Introduction to Software Engineering, Requirements Analysis, and Specification

I. Materials for weeks 1 and 2 of class:

- A. Syllabus.
- B. Projects descriptions.
- C. Milestone 1 writeup.
- D. Specification document outline.
- E. SVN basics.
- F. Standard operating procedures, Volume 1.
- G. These lecture notes.

II. Scheduling details for the first two weeks.

- A. First day’s activities (Monday):
  1. In lecture:
     - a. Tour of syllabus and other handouts.
     - b. Brief introduction to the software system life cycle and requirements analysis.
  2. In lab:
     - a. Choice of project teams and projects.
     - b. Preparation for initial customer interviews.
- B. Second day’s activities (Wednesday):
  1. Initial customer interviews with all teams, ***in both lecture and lab.***
  2. To provide ample interview time, there will be no normal lecture on this Wednesday.
  3. The precise meeting schedule will be determined on the first day of classes.
- C. Third day’s activities (Friday):
  1. Normal lecture.
  2. Lab introduction to project repository and SVN.
- D. Fourth day’s activities (Monday):
  1. Second round of customer interviews.
  2. As with preceding Wednesday, no normal lecture.
  3. Precise schedule TBA.
- E. Third week and beyond:
  1. "Normal" lectures.
  2. Lab meetings as described in syllabus, specific times TBA, in forthcoming handout.

III. What is software engineering?

- A. The *disciplined* creation of software.
- B. Well-known principles of scientific problem solving are applied, including:
  1. Defining a problem clearly before starting its solution.
  2. Using a "divide and conquer" strategy to manage the complexity of a problem and its solution.
- C. Well-known principles of engineering are applied, including:

1. Using formal mathematics to specify a system precisely.
2. Formally verifying that a problem solution meets its specification.

IV. The different types of software.

A. There are three broad categories of software, based on the application domain, i.e., the general area in which the software is applied.
   1. End-user software.
      a. Used by people to get work done.
      b. Has a human-computer interface (HCI).
   2. System software.
      a. Provides underlying support to end-user software.
      b. Has an application programmer interface (API), and perhaps limited HCI.
   3. Embedded software
      a. Used within hardware devices.
      b. Has no HCI; the interface is directly with the hardware.

B. There are two categories of software based on the clientele who purchase it.
   1. Off-the-shelf (or shrink-wrap) software is built by software developers who sell it on the open marker.
   2. Custom (or bespoke) software is built to satisfy the needs of specific customers, typically an organization of some kind.

C. In 308, we are building custom end-user software.

V. The people involved with software.

A. The following are software "stakeholders", i.e., people who have some interest in a software product and/or its development.
   1. ***end users*** -- people who will use the software or people who represent those who will use it
   2. ***customers*** -- people who purchase the software, which they may or may use themselves
   3. ***domain experts*** -- people who fully understand the application domain in which the software will run
   4. ***analysts*** -- members of the software development staff who specialize in requirements analysis and specification
   5. ***implementors*** -- members of the development staff who specialize in software design and implementation
   6. ***testers*** -- members of the development staff and user community who test the software to ensure that it meets the requirements specification
   7. ***managers*** -- those who manage the development process, as well as those who manage end users when the software is installed in an organization
   8. ***visionaries*** -- those who have the "big picture" for what the software is intended to do and how it will be developed
   9. ***maintainers and operators*** -- those who conduct post-development maintenance and operations, as necessary
   10. ***other interested parties*** -- anyone else interested in the software product, such as those with a financial investment

B. The first four groups work together as a team to develop the requirements, perhaps with some individuals in more than one group.

C. It may be the case that the members of the implementation team do not participate at all in the requirements specification, but rather accept the requirement specification document as input.

D. In CSC 308, you will primarily play the roles of analyst and tester, with a secondary roles as domain experts and end users as appropriate.

E. CSC 309 is concerned with project implementation.

VI. The software development process.

A. For software to be properly engineered, its development must be conducted in an orderly process.

B. The diagram in Figure 1 depicts the major stages of the software development process.

C. The Analyze step of the process addresses the requirements to be met by the software.

1. For software that is to be used directly by humans, the primary activity of the Analyze phase is to acquire and organize the functional requirements of the human users.
2. This part of the analysis involves a considerable amount of human-to-human communication.

D. The Specify step of the process involves the development of a formal model of the requirements.

1. The model represents the requirements in a form that can be mechanically analyzed.
2. Developing the formal specification allows the requirements to be analyzed for completeness and consistency.

E. The Design step of the process involves organizing the major components of the software system.

1. The initial design is derived from the components of the formal specification model.
2. The initial design is then refined into an efficient software architecture, consisting of packages, classes, and methods.

F. The Implement step fills in the operational details

1. Class data structure details are determined.
2. The code for methods is implemented.

G. The following are some noteworthy considerations about the process.

1. Ideally, each step of the process should be completed before the next step is begun.
   a. If we ignore the upward-pointing dashed arrows in Figure 1, we can view the process diagram as a

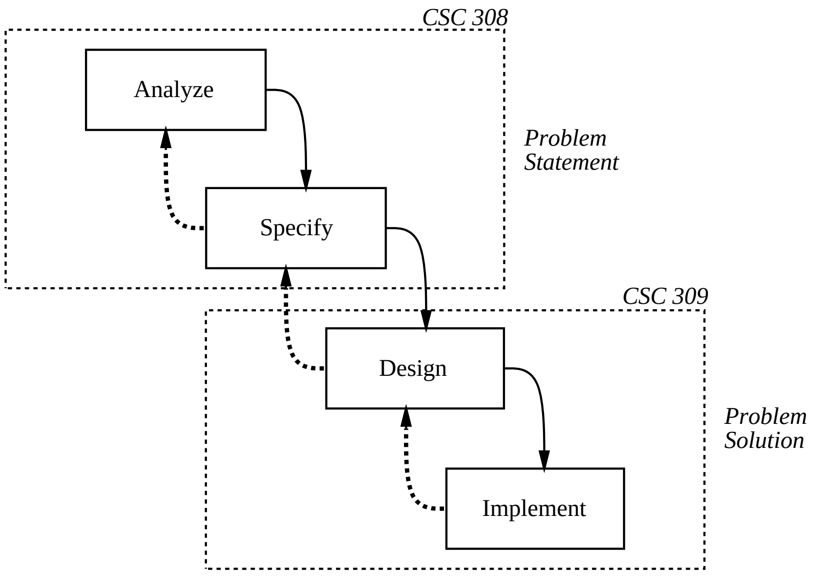


Figure 1: Major phases of the software development process.

"waterfall chart".

b. In this view, information only flows down from higher steps to lower steps.

2. In practice, the ideal waterfall view is rarely possible.
   a. In diagrammatic terms, water sometimes needs to flow up hill (the dashed lines).
   b. This view allows feed-back from a lower phase to a higher phase.

3. The process we follow in CSC 308 and 309 has the following properties:
   a. There is substantial feedback between the Analyze and Specify steps of the process.
   b. There is also substantial feedback between the Design and Implement steps.
   c. The feedback from the Design step back up to the Specify step is limited, and controlled by specification change orders (SCOs).
      i. The reason is that it is important to have a complete and sound specification before design and implementation begin.
      ii. Further, it is important to control any requirements specification changes during design and implementation.
      iii. A common source of problems in software system development is that of the "shifting requirements" -- requirements and/or specifications that change significantly once the design and implementation phases have begun.

H. Viewing the software process as a problem solving exercise, it is clear why changing requirements and specification are troublesome:

1. The requirements analysis and specification can be considered the "problem statement" phase.
2. The design and implementation are the "problem solution" phase.
3. When the system requirements or specification change after the design and implementation are in progress, it is like changing the problem to be solved while the solution is being developed.

VII. Additional "pervasive" steps of the software process.

A. Figure 1 shows the major steps of the process that are carried out in an ordered, step-by-step manner.

B. Even if there is some feedback among the steps, the overall order is first to analyze requirements, then specify, then design, and then implement.

C. In addition to the four ordered steps, there are four other steps that are carried out continuously, or "pervasively", throughout the development process.

D. The pervasive steps of the process are

1. Manage
2. Configure
3. Test
4. Document
5. Reuse

E. The Manage step entails the management of the people involved in the process.

1. Project meetings are scheduled at regular intervals.
2. Project supervisors oversee and evaluate the work of their subordinates.

F. The Configure step of the process involves the organization and management of the software artifacts.

1. The Configure step is supported by the use version control and configuration management tools.
2. These tools allow artifacts to be checked in to a software repository, with version changes controlled and documented throughout the ongoing process.

G. The Test step of the process ensures that software artifacts being produced meet certain measurable standards.

1. Testing requirements involves careful human inspection, and formal review to ensure that user needs are

being met and properly defined.

2. Testing the specification and design involves formal analysis for completeness, consistency, and other measurable properties.
3. Testing the implementation involves formal functional testing to ensure that execution results meet the specification.

H. The Document step produces documentation suitable for everyone involved in the process.

1. A requirements specification document is produced in a form suitable for both end users and system developers.
2. Software maintenance documentation is produced during the design and implementation steps.
3. Various forms of reports are produced for the administrative staff.
4. End user manuals and tutorials are produced for the final delivered software product.

I. The Reuse step evaluates existing artifacts to determine if they can be used in whole or in part in a new development project.

1. Reuse from libraries is a normal part of the development process.
2. Reuse of other artifacts involves refining and adapting them to meet current needs.

J. The important characteristics of a pervasive process step are the following.

1. Pervasive steps are carried out during each of the ordered steps, and in a manner specifically related to each step.
   a. For example, testing is carried out during each of the analyze, specify design, and implement steps of the process.
   b. Further, the form of testing carried out for each ordered step is specifically designed for the kind of software artifact that each ordered step produces.
2. Pervasive steps are typically performed at regularly scheduled intervals.
   a. For example in CSC 308, we will perform requirements testing on a weekly basis during the second half of the quarter.
   b. As part of the Manage step, we will have regularly scheduled project meetings and reviews.

VIII. Traditional process versus agile processes.

A. The process we follow in 308 and 309 can be considered *traditional* in the sense that it is based on a clear set of plans, carried out in a specific way.

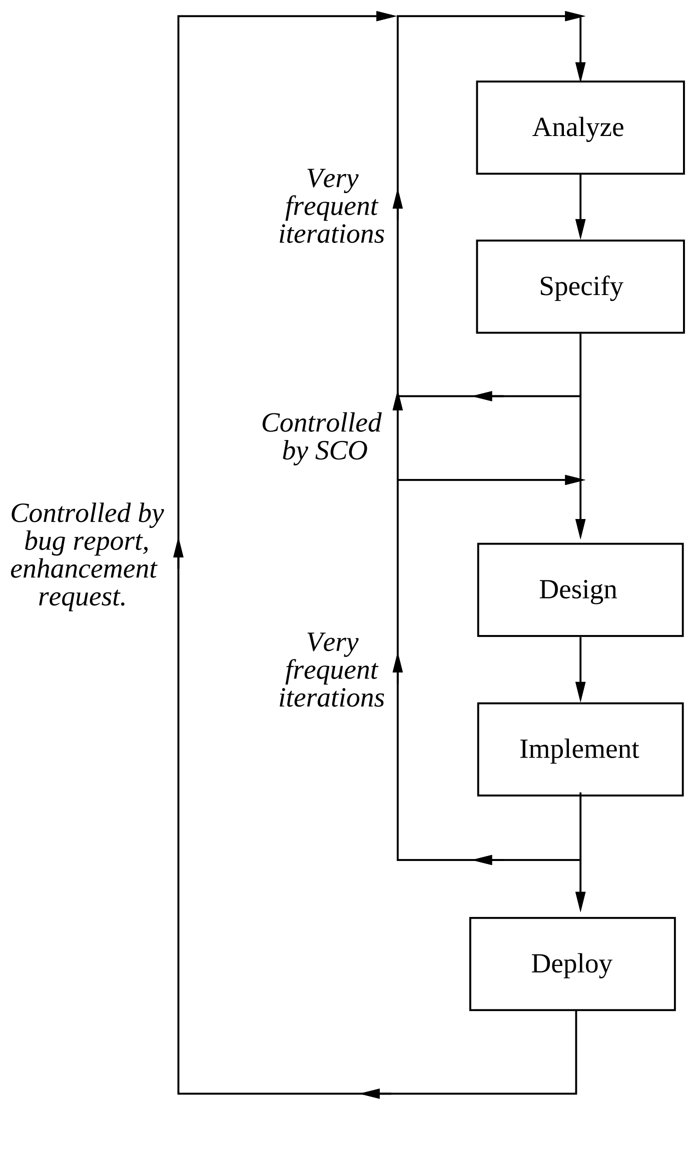
B. One of the particularly traditional aspects of the class process is the production of a substantial requirements document, which can be considered a plan for subsequent implementation.

C. A more incremental process approach, called *agile development*, does much less up-front requirements analysis, but instead develops requirements in small increments as the process proceeds (see Figure 2).

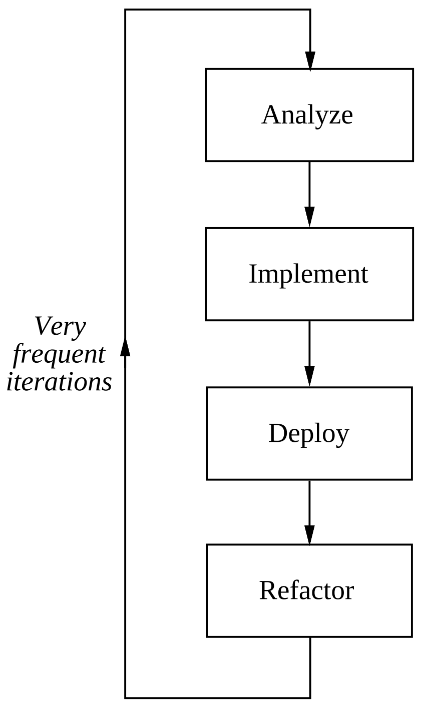
1. Customers and implementors work very closely together, and a product is deployed to the customers in very small increments.
2. The more traditional steps of specification and design are replaced by incremental "refactoring", which cleans up the incrementally developed code, that has a tendency to get messy along the way.

D. Agile development, and its underlying methodology of *extreme programming*, are relatively new in the world of software processes.

1. People have reported success using agile methods for small to medium-scale projects, with tight-knit teams of customers and developers.
2. Few solid studies have been done to determine how well agile methods work.
3. At present, there are questions about the efficacy of agile development for large-scale projects for which incremental development is not always feasible.
4. Without question, agile methods have proved successful in many cases.



a. Traditional process



b. Agile process

Figure 2: Comparing traditional and agile software processes.

IX. What is involved in requirements analysis and specification?

A. These steps involve precisely specifying the need for a proposed software system.

B. In order for a requirements specification to be complete and consistent, it is defined in a requirements specification document.

C. The informal sections of the document must be understandable to everyone who will be affected by the computing equipment, so that:

1. Everyone understands precisely how computing will affect their work.
2. Everyone can contribute to the formulation of the requirements.

D. The formal sections of the document must be precise enough for use as a contractual instrument for the subsequent development or acquisition of the software.

X. Importance of careful requirements analysis.

A. Before an organization procures a software system to meet its computing needs, or before a company builds a software system to meet marketplace needs, the people involved should have a precise understanding of exactly what the needs are.

B. This seemingly obvious idea is often overlooked in computer system acquisition and development.

C. The lure of technology and/or the skill of the technology vendor often lead to insufficient time being spent on requirements analysis and specification.

1. Vendors may over sell a system, with claims that the system can meet a wide range of needs
2. Marketers may overestimate or misunderstand the needs of a potential marketplace.

D. Many organizations have learned painfully that hastily-acquired computer systems can cause more problems than they solve.

E. Companies who have built hastily-specified software products have often found insubstantial markets for their product.

F. There is nearly universal agreement among software engineers that thorough requirements analysis is essential for successful software development.

XI. Patience is required from all participants

A. It may sometimes seem that the requirements analysis process spends a long time specifying the obvious.

B. During the requirements, many people may think that they have a clear idea of the needs to be met and what the software is supposed to do.

C. Even if many do, it is often the case that not everyone has the *same* ideas.

D. Hence, a precise requirements analysis document serves to help everyone agree on what the needs are, in addition to stating the needs precisely.

XII. Major phases of requirements specification

A. Requirements analysis with end-user scenarios.

1. The language used is English and pictures.
2. The primary audience is proposed customers and end users.
3. Much user consultation is required to gather necessary data.

B. Formal model specification.

1. A formal specification language is used.
2. The primary audience is system designers and implementors; a secondary audience is domain experts.
3. The final version is a *very* formal document suitable for use as a contractual instrument for the procurement of a computer system.

XIII. Details of user consultations and data gathering

A. It is critically important to involve end-users in the requirements analysis from the outset and throughout the process.

B. With this involvement, the ultimate success of the computer system is far more likely than without it.

C. Many serious failures in software development have resulted when the needs of end users are neglected.

XIV. Activities of user consultation include:

A. User interviews

B. User interface scenarios

C. User questionnaires or surveys

D. Visits to other similar organizations and computer system installations

E. Rapid system prototypes

XV. Customer interview techniques.

A. For users who are not computer specialists, the analyst should not ask questions using computer jargon or terminology.

B. Questions should be specialized to what individual users know best.

C. Analysts should use other common sense interview skills, including being prepared, polite, succinct, non-threatening, diplomatic, and empathetic, as appropriate.

XVI. User interface scenarios.

A. The goal of this activity is to provide potential users with a concrete view of what the proposed system will be like to use.

B. Scenarios begin with the premise "Suppose the system existed already, what would it look like to the user?"

1. The scenarios describe precisely what a user sees when the system is used.
2. Scenarios define what screens look like, what the user commands are, the format of user-visible data, and all other aspects of the end-user interface.

XVII. Rapid system prototyping.

A. For some software projects, building an operational prototype can be helpful to capture user requirements.

B. A prototype is a version of the software product that has reduced functionality that can be rapidly developed prior to full design and implementation.

C. Figure 2 shows two views of how prototyping can be integrated into the software development process.

1. In Figure 2a, prototyping is a specific step of the process.
   a. In this type of process, the prototype is typically a "store front" style of system that allows a user to interact through an operational user interface.
   b. Behind the interface there is no actual functionality.
2. In Figure 2b, a prototype is constructed by taking one or more initial passes through the complete process.
   a. As each full-process pass is completed, increasingly more functionality is added.
   b. When a complete and stable result is produced, the prototype has been transformed to a production software system.
   c. This kind of process is often called "iterative development".

D. In CSC 308, we will take the view of prototyping shown in Figure 2b. By the end of 309, we could have a first-pass prototype of an operational system.

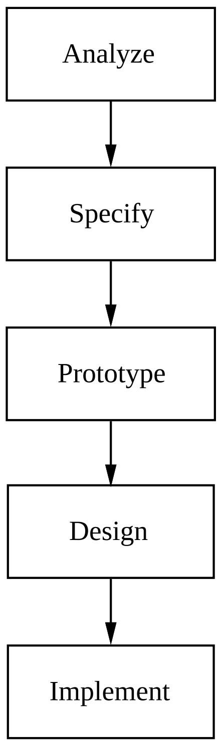
XVIII. Establishing genuine user needs.

A. The notion of establishing genuine need is quite critical in the analysis process.

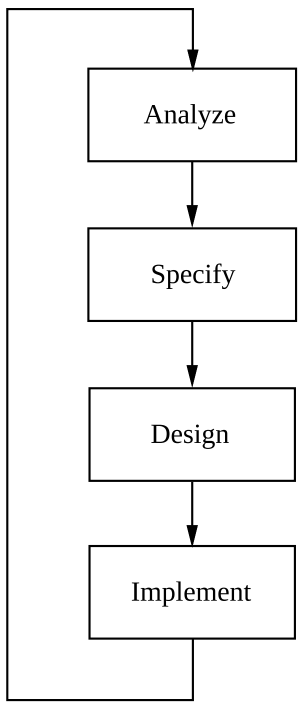
B. Many computer systems have been for which there is no substantial, demonstrated need.

C. When software is being analyzed for a particular organization, a forthright analyst should be prepared to say to customers at the end of the requirements phase: "Hey, you folks really don’t need any new software at all. What you already have works fine based on your current needs, or you could buy what you need off the shelf."

D. When requirements are being gathered for a general-market software product, marketing analysts must be prepared to conclude that there may not be a substantial market for a piece of software, once the



a. Prototyping as an explicit step of the process.



b. Prototyping as multiple passes of the process.

Figure 3: How prototyping fits into the software process.

requirements for it are fully understood.

XIX. Other important aspects of requirements analysis.

A. Identification of users and other operational personnel.
   1. Precisely who the end users will be.
   2. Who will be operate and maintain the system, as necessary.
   3. Others who will be affected by the system.

B. Overview of current and proposed operations.
   1. How things work before the proposed software system exists.
   2. How things will work after the proposed software system is installed.

C. Analysis of relevant existing systems.
   1. Are there systems with features similar to those required for the proposed system?
   2. What is good and bad about those systems?

D. Impact analysis.
   1. What positive impacts will the system have on the intended user community?
   2. What negative impacts will the system have?

XX. Examples of requirements specification

A. In 308, we will study a concrete example similar in size and scope to the projects you are working on; it is an electronic Calendar Tool.

B. The example will be presented in phases corresponding to the milestones of the 308 projects covered in the class syllabus.

C. The first concrete example covers roughly what Milestone 1 should look like.

D. We will go over this example in detail next.