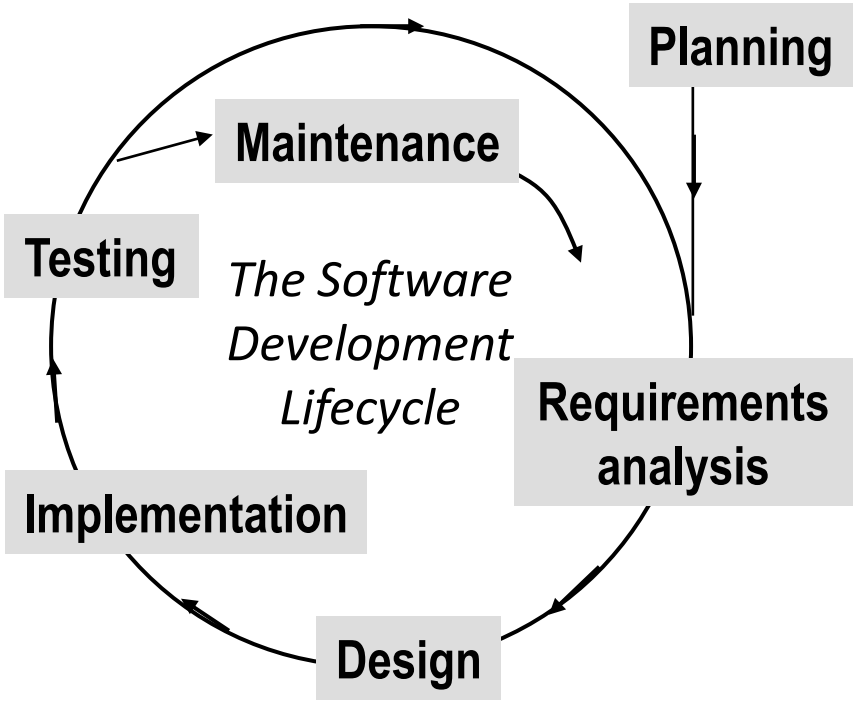
Part I

INTRODUCTION TO SOFTWARE ENGINEERING



- Why is software engineering important?
- Who and what does it consist of?
- What are its main activities?
- What are the principles of software engineering?
- What ethics are involved?
- What sorts of case studies will be used to illustrate the subject?

Goal of Software Engineering

- Creation of software systems that are
  - Reliable
  - Efficient
  - Maintainable
  - Meet the needs of customers
- Production of system meets
  - Schedule
  - Budget

What is Software Engineering?

- Engineering discipline
  - *the design, analysis and construction of an artifact for some practical purpose*

- IEEE definition:
  - *"the application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software; that is the application of engineering to software."*

NATO Study Group

- NATO Study Group on Computer Science (1968)
  - one of the first uses of the phrase *software engineering*

- "Programming management will continue to deserve its current poor reputation for cost and schedule effectiveness until such time as a more complete understanding of the program design process is achieved."

NATO Study Group (cont.)

- “Today we tend to go on for years, with tremendous investments to find that the system, which was not well understood to start with, does not work as anticipated. We build systems like the Wright brothers built airplanes — build the whole thing, push it off the cliff, let it crash, and start over again.”

Software Disasters

- Numerous examples of software disasters
  - Ariane Project
  - 1990 AT&T Disaster
  - Radiation Overdose

Software Failure

- What is it?
  - Failure to meet expectations
- What expectations are not achieved?
  - Over budget
  - Exceeds schedule and/or misses market window
  - Doesn’t meet stated customer requirements
  - Lower quality than expected
  - Performance doesn’t meet expectations
  - Too difficult to use

Software Failure (cont.)

- Reasons for failure:
  - Unrealistic or unarticulated project goals
  - Poor project management
  - Inaccurate estimates of needed resources
  - Badly defined system requirements
  - Poor reporting of the project's status
  - Unmanaged risks
  - Poor communication among customers, developers, and users
  - Inability to handle the project's complexity
  - Poor software design methodology
  - Wrong or inefficient set of development tools
  - Inadequate test coverage
  - Inappropriate (or lack of) software process

Software Engineering Activities

4 P's of Software Engineering

- **People**
  - *Project stakeholders*
- **Product**
  - *The software product plus associated documents*
- **Project**
  - *The activities carried out to produce the product*
- **Process**
  - *Framework within which the team carries out the activities necessary to build the product*

People

Stakeholders

- *Business Management*
- *Project Management*
- *Development Team*
- *Customers*
- *End-Users*

The Software Product Artifacts

- Project documentation

  Documents produced during software definition and development

- Code

  Source and object

- Test documents

  Plans, cases, and results

- Customer documents

  Documents explaining how to use and operate product

- Productivity measurements

  Analyze project productivity

Project

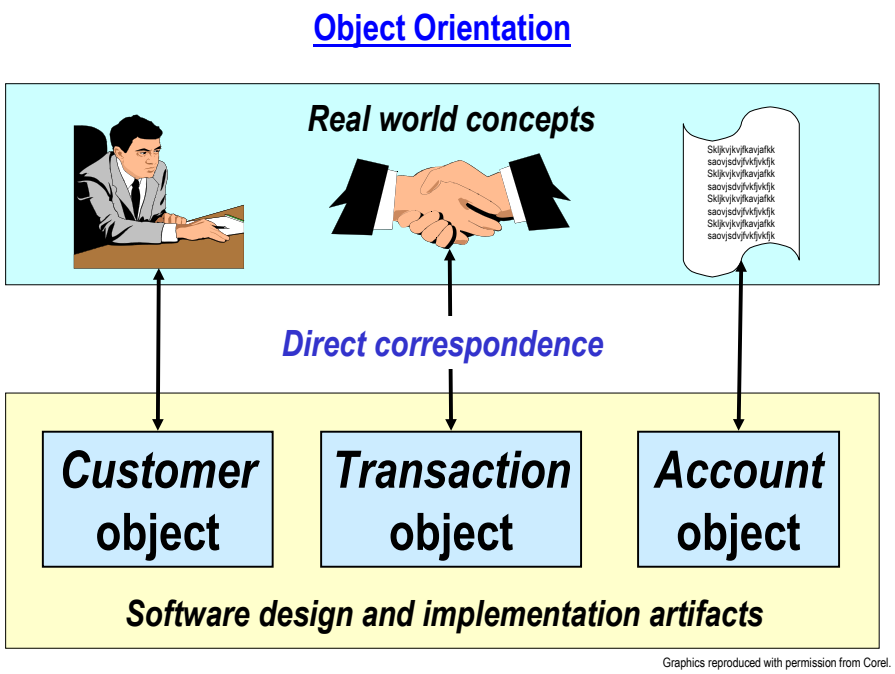
Software Project Activities

-- which produce a software product: Mainly…

- **Planning**
  - plan, monitor and control the software project
- **Requirements analysis**
  - define what to build
- **Design**
  - how to build the software
- **Implementation**
  - program the software
- **Testing**
  - validate software meets the requirements
- **Maintenance**
  - resolve problems; adapt software to meet new requirements;

Project (cont.)

- Development paradigm
  - e.g. object-oriented

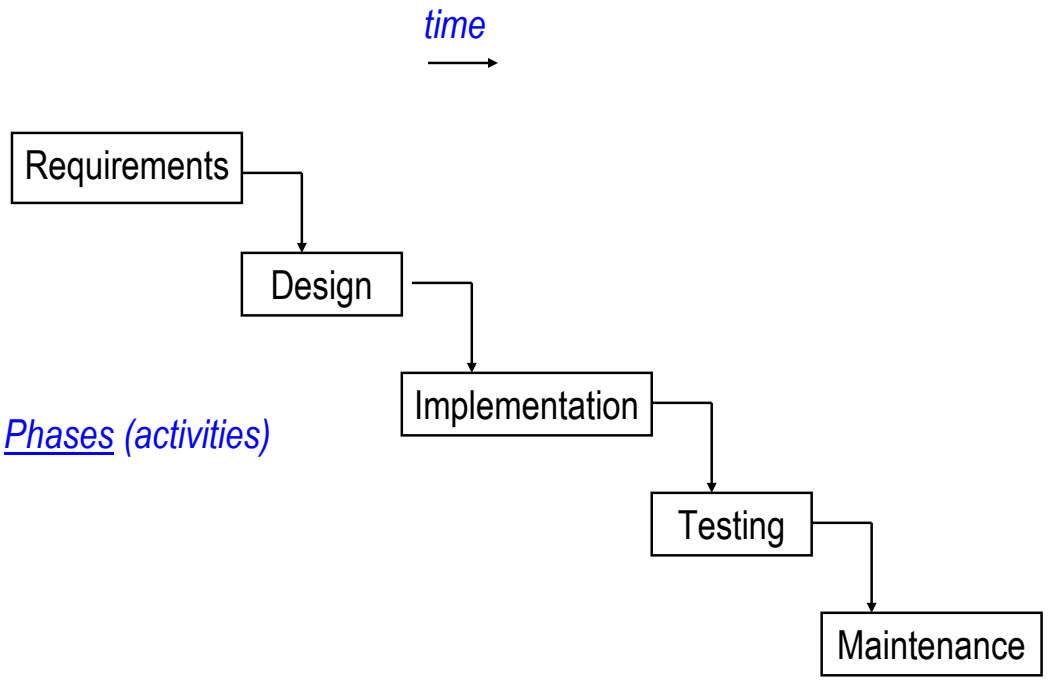


Process

- Framework for carrying out the activities of a project in an *organized* and *disciplined* manner.
- Imposes structure
- Waterfall or Iterative

Waterfall Process

The Waterfall Software Process



- Simplest process
- Sequential
- Basis for others

Iterative Process

- Software projects rarely follow ***strict*** waterfall
- Some ***iteration*** between specifications, design, implementation and test
- Requires discipline
  - e.g. update specifications when design changes

Software Engineering Principles

Software Engineering Principles

1. *Make Quality Number 1*
2. *High Quality Software is Possible*
3. *Give Products to Customers Early*
4. *Use an Appropriate Sofware Process*
5. *Minimize Intellectual Distance*
6. *Inspect Code*

Source: 201 Principles of Software Engineering, Alan Davis

7. *People are the Key to Success*

Software Engineering Ethics

- Most disciplines operate under a strict set of *ethical* standards

- The *Merriam-Webster* online dictionary defines *ethics* as:

  1: the discipline dealing with what is good and bad and with moral duty and obligation

  2: a set of moral principles

Software engineering ethics

- Software engineering involves wider responsibilities than simply the application of technical skills.
- Software engineers must behave in an honest and ethically responsible way if they are to be respected as professionals.
- Ethical behaviour is more than simply upholding the law but involves following a set of principles that are morally correct.

Issues of professional responsibility

- Confidentiality
  - Engineers should normally respect the confidentiality of their employers or clients irrespective of whether or not a formal confidentiality agreement has been signed.
- Competence
  - Engineers should not misrepresent their level of competence. They should not knowingly accept work which is without their competence.

Issues of professional responsibility

- Intellectual property rights
  - Engineers should be aware of local laws governing the use of intellectual property such as patents, copyright, etc. They should be careful to ensure that the intellectual property of employers and clients is protected.
- Computer misuse
  - Software engineers should not use their technical skills to misuse other people's computers. Computer misuse ranges from relatively trivial (game playing on an employer's machine, say) to extremely serious (dissemination of viruses).

The ACM/IEEE Code of Ethics

Software Engineering Code of Ethics and Professional Practice

ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices

PREAMBLE

The short version of the code summarizes aspirations at a high level of the abstraction; the clauses that are included in the full version give examples and details of how these aspirations change the way we act as software engineering professionals. Without the aspirations, the details can become legalistic and tedious; without the details, the aspirations can become high sounding but empty; together, the aspirations and the details form a cohesive code.

Software engineers shall commit themselves to making the analysis, specification, design, development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following Eight Principles:

Ethical principles

1. PUBLIC - Software engineers shall act consistently with the public interest.
2. CLIENT AND EMPLOYER - Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.
3. PRODUCT - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
4. JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.
5. MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
6. PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
7. COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.
8. SELF - Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.