Software Engineering

Lecture 1: Introduction

Mahmoud El-Gayyar

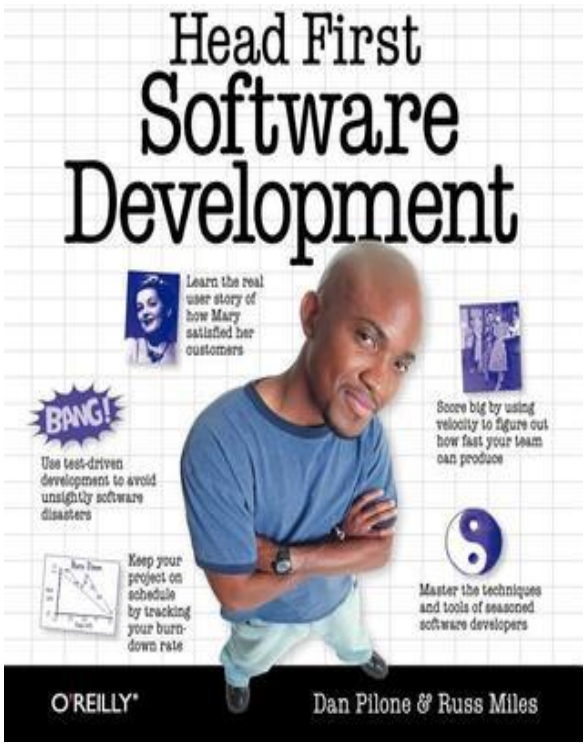
elgayyar@ci.suez.edu.eg

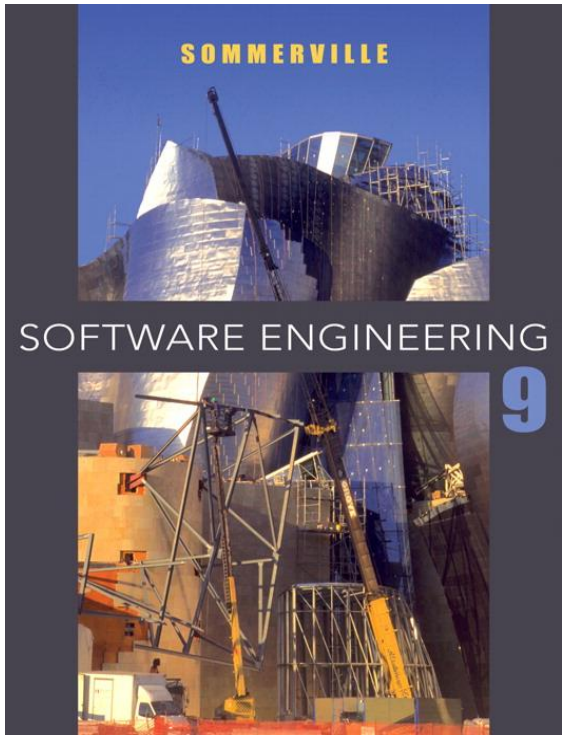
Slides are a modified version of the slides by
Prof. Kenneth M. Anderson and Prof. Ian Sommerville

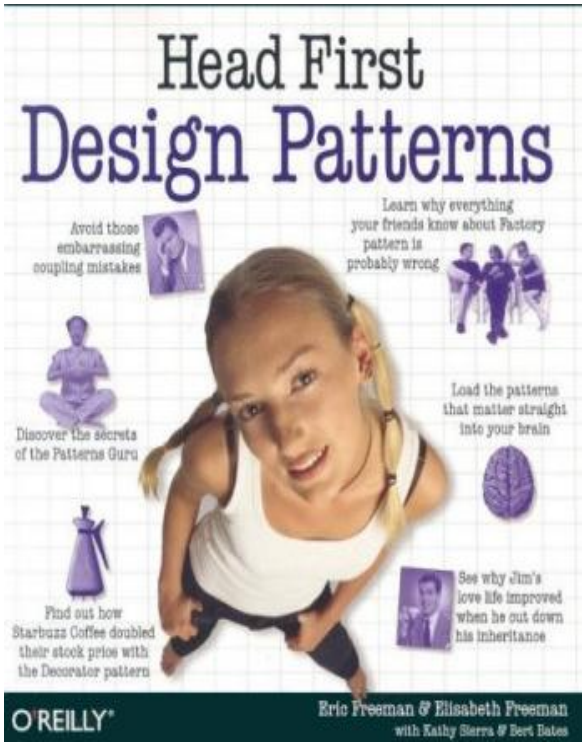
Course Objectives

- Survey software engineering concepts, terminology and techniques
  - Emphasis on Agile Design Methods
  - Will supplement with traditional/historical material as needed
- *Software modeling*
- *And some practical programming concepts (Design Patterns)*
- http://elgayyar.weebly.com/

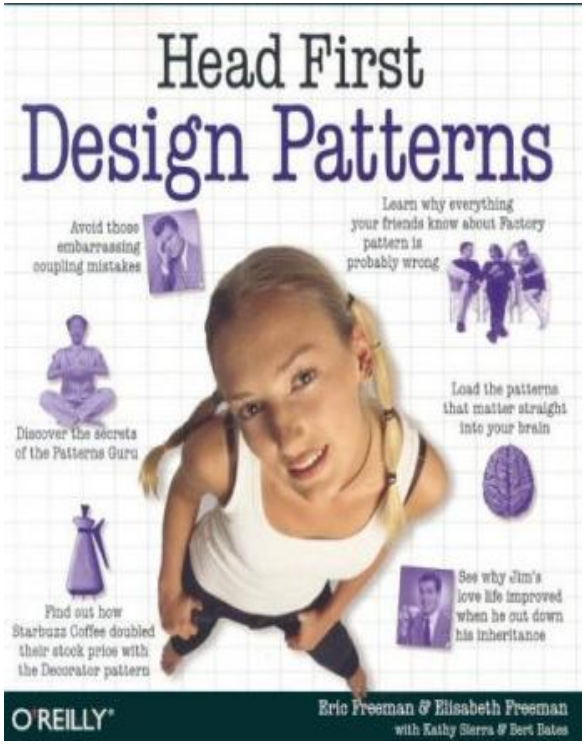
Course Books







In the LAB



Dr. Amira

- JAVA Design Patterns

Course Evaluation

- 50% Midterm Exam
- 25% 15-min Oral Presentations (Every week one group)
- 25% your participation in the LAB
- Practical and Oral Exams
  - Java Project
- *Divide yourself into groups*
  - 8 to 10 per group
  - ***Must next week !!!!!!!!!!***

Rules !!

Presentation Topics

1. *Software Process Models (at least 3 other than Agile)*
2. Kanban
3. Lean Software Development
4. Crystal Agile Family
5. Feature Driven Development
6. Agile development in the large and the Pragmatic Programmer
7. *Software Architectures*
8. *Cloud Computing*
9. *Web Services*
10. *Software Teams Structures*
11. *Git and Subversion: Distributed Configuration Management*

Outline (Ch 1. Sommerville)

- ***Professional software development***
  - What is meant by software engineering?
  - Software process activities?
  - Software Cost?
- *Software Engineering Methods*
  - Key challenges
  - Application types
  - CASE tools
- *Software engineering ethics*
  - A brief introduction to ethical issues that affect software engineering.
  - ACM/IEEE Code of Ethics

Why Software Engineering?

- *But …*
  - Where did the specification come from?
  - How do you know the specification corresponds to the user's needs?
  - How do you know the program actually meets the specification?
  - How do you know your program will always work correctly?
  - What do you do if the users' needs change?
  - How do you divide tasks up if you have more than a one-person team?

What is a software?

- *Software:*
  - Computer **programs** and
  - associated **documentation**.
- *Software products may be developed for*
  - a particular customer
  - a general market.

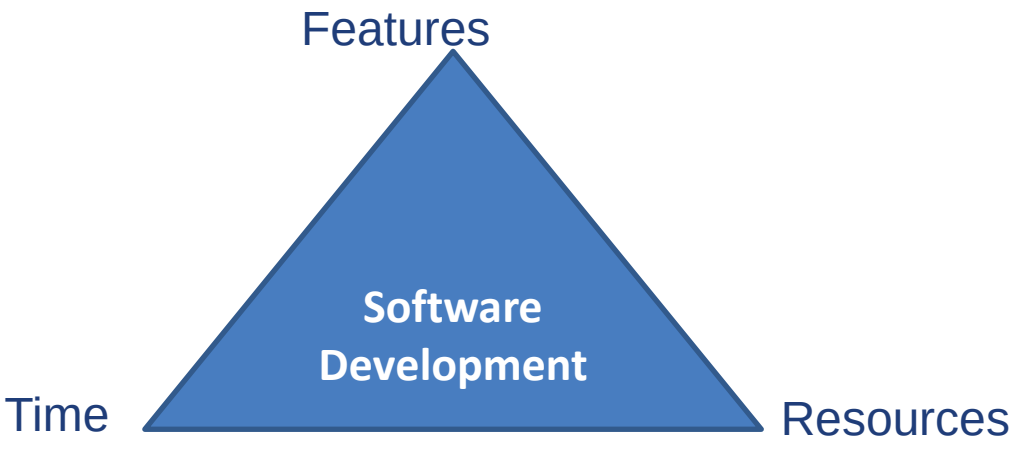
What is Software Engineering?

*Software engineering is concerned with **theories, methods and tools** for **professional** software development.*

-Sommerville

State of the art** of developing **quality** software **on time** and within **budget

-Anonymous



What is Software Engineering?

> ***Multi-person*** *construction of* ***multi-version*** *software.*
>
> *-Parnas*

- Team-work
  - Scale issue (“program well” is not enough) + Communication Issue
- Successful software systems must evolve
  - Change is the norm, not the exception

What is Software Engineering?

*Everything else, **besides programming**, that contributes to building **efficient** software systems.*

-Anonymous

- Programming is not enough
- Every one of you should be able to implement Facebook.
  - The reason why there is only one Facebook is everything else they did right besides programming.
  - Plus luck ☺

Software Engineering vs. Computer Science

- Computer science focuses on **theory and fundamentals**;

- Software engineering is concerned with the **practicalities** of developing and delivering useful software.

Software Engineering vs. System Engineering

- System engineering is concerned with all aspects of computer-based systems development including **hardware, software and process engineering**.
- Software engineering is **part of** this more general process

Software Products

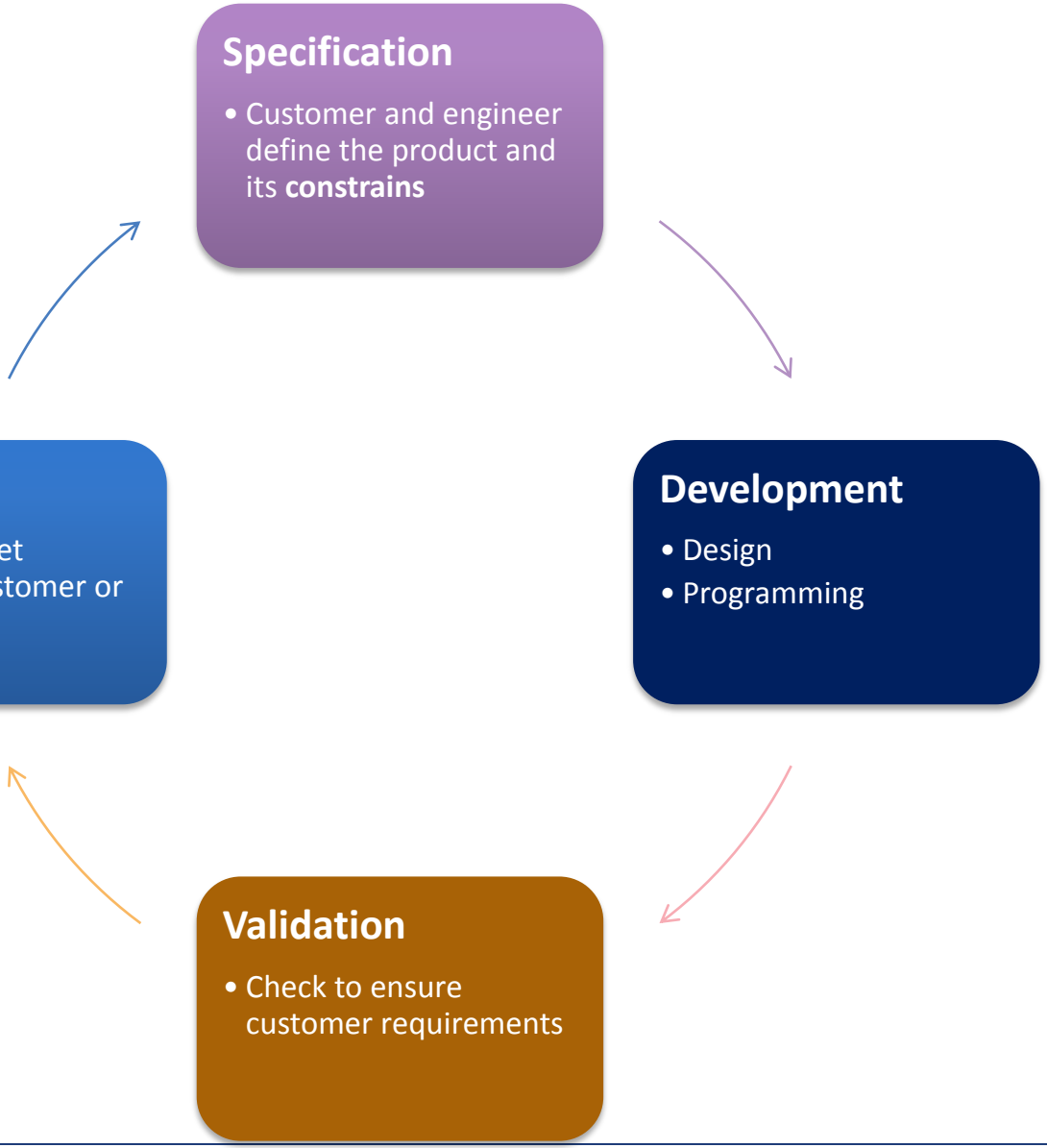
- *Generic products*
  - Stand-alone systems that are marketed and sold to any customer who wishes to buy them.
  - Examples – PC software such as graphics programs, project management tools; etc.
- *Customized products*
  - Software that is commissioned by a specific customer to meet their own needs.
  - Examples – embedded control systems, automation system for a certain company.

Product Specification

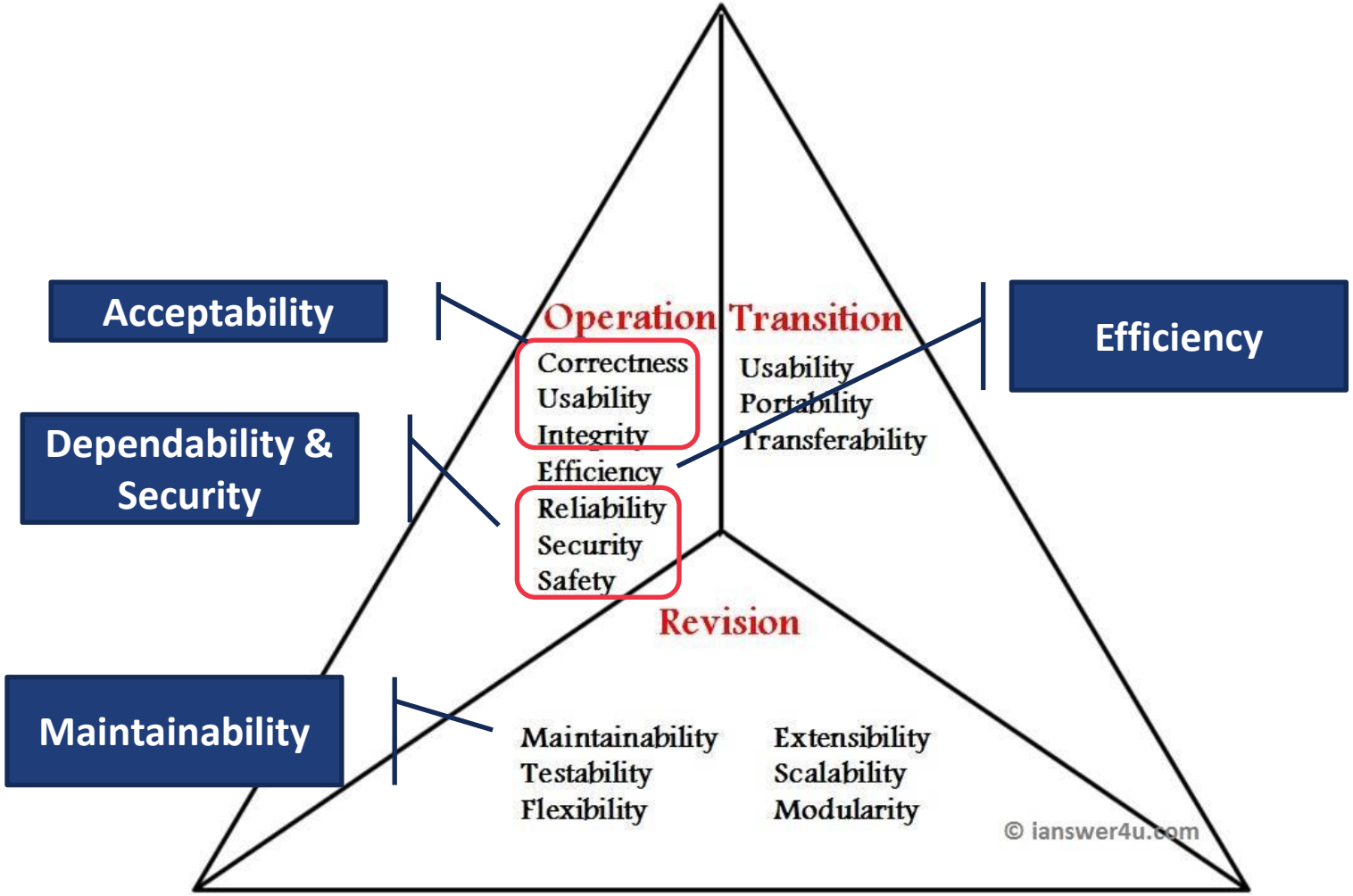
- *Generic products*
  - The specification of what the software should do is owned by the software developer and decisions on software change are made by the developer.
- *Customized products*
  - The specification of what the software should do is owned by the customer for the software and they make decisions on software changes that are required.

Software Process Activities

Software Process: A set of activities whose goal is the development or evolution of software.



What is a Good Software?



Software Quality Triangle

Software Costs

- *Software costs often dominate computer system costs.*
  - The costs of software on a PC are often greater than the hardware cost.
- *Software costs more to maintain than it does to develop.*
  - Roughly **60%** of software costs are **development** costs, **40%** are **testing costs**.
  - For systems with a long life, maintenance costs may be several times development costs.
- *Software engineering is concerned with **cost-effective** software development.*

Software Related Jobs

- *Software Developer*
- *Software Engineer*
- *SQA Engineer (Quality Assurance)*
- *Usability Engineer (UX)*
  - Strong HCI background
- *System Analyst*
  - Professional requirements gathering
  - Professional designer
- *DBA / System Admin*
- *Web Designer*
- *Build / Configuration Manager*

Outline

- *Professional software development*
  - What is meant by software engineering?
  - Software process activities?
  - Software Cost?
- ***Software Engineering Methods***
  - Key challenges
  - Application types
  - CASE tools
- *Software engineering ethics*
  - A brief introduction to ethical issues that affect software engineering.
  - ACM/IEEE Code of Ethics

SWE: Key Challenges

- *Coping with **increasing diversity**;*
  - Increasingly, systems are required to operate as distributed systems across networks that include different types of computer and mobile devices.
- *demands for **reduced delivery times**;*
  - Business and society are changing incredibly quickly as emerging economies develop and new technologies become available. They need to be able to change their existing software and to rapidly develop new software.
- *developing **trustworthy software**.*
  - As software is intertwined with all aspects of our lives, it is essential that we can trust that software. (internet-based payment)

SWE Methods is not Constant!

- *There are many different types of software system*
  - **no universal set of software techniques** that is applicable to all of these.
- *The software engineering methods and tools used depend on*
  - the **type** of application being developed,
  - the **requirements** of the customer and
  - the **background** of the development team.

Application Types

- ***Stand-alone** applications*
  - These are application systems that run on a local computer, such as a PC (no need for network connectivity) e.g.: Office, photo manipulation.
- ***Interactive transaction-based** applications*
  - Applications that execute on a remote computer and are accessed by users from their own PCs or terminals. These include web applications such as e-commerce applications. (security, large data)
- ***Embedded** control systems (most popular)*
  - These are software control systems that control and manage hardware devices. (Mobile phones, cars, microwave, ..)

Application Types

- ***Batch processing** systems*
  - These are business systems that are designed to process data in large batches. They process large numbers of individual inputs to create corresponding outputs. (e.g. phone billing system)
- ***Entertainment** systems*
  - These are systems that are primarily for personal use and which are intended to entertain the user. (e.g. Games → Interaction comes first)
- *Systems for **modelling and simulation***
  - These are systems that are developed by scientists and engineers to model physical processes or situations, which include many, separate, interacting objects. (high performance → parallel systems)

Application Types

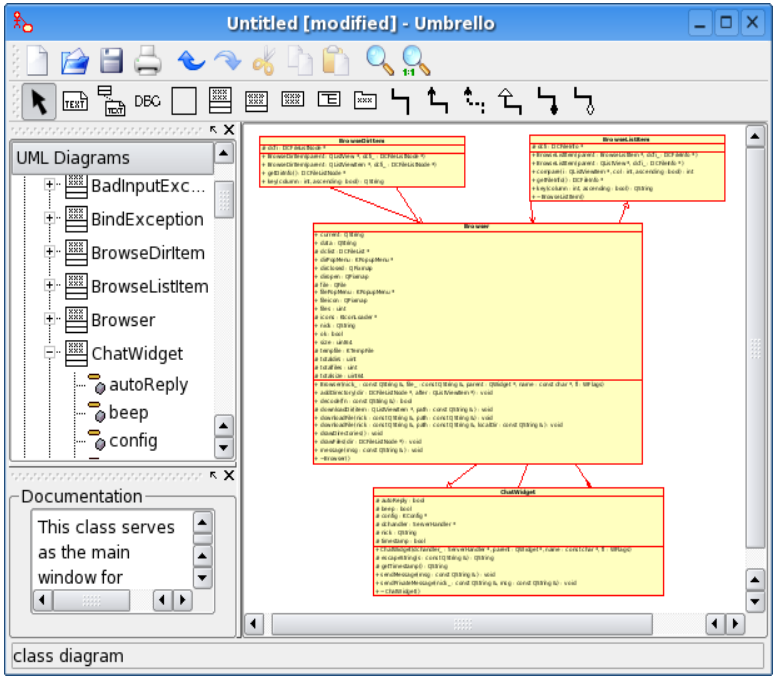
- ***Data collection** systems*
  - These are systems that collect data from their environment using a set of sensors and send that data to other systems for processing. (Hostile external environment)
- ***Systems of systems***
  - These are systems that are composed of a number of other software systems.

Software Engineering Fundamentals

- *Some fundamental principles apply to all types of software system, irrespective of the development techniques used:*
  - Systems should be developed using a **managed and understood development process**.
    - *Of course, different processes are used for different types of software.*
  - **Dependability** and **performance** are important for all types of system.
  - **Understanding** and managing the software specification and **requirements** (what the software should do) are important.
  - Where appropriate, you should **reuse** software that has already been developed rather than write new software. (**Fast development**)

CASE (Computer-Aided Software Engineering)

- *Software systems that are intended to provide automated support for software process activities.*
- *Upper-CASE*
  - Tools to support the early process activities of requirements and design;
- *Lower-CASE*
  - Tools to support later activities such as programming, debugging and testing (e.g. IDE like eclipse).



Key Points

- *Software engineering is an engineering discipline*
  - concerned with all aspects of software production.
- *Essential software attributes are*
  - maintainability, dependability (reliability, security), efficiency, and acceptability.
- *The high-level activities of specification, development, validation and evolution are part of all software processes.*
- *There are many different types of system and each requires appropriate software engineering tools and techniques for their development.*

Review Questions

1. *Why different application types require specialized software engineering techniques to support their design and development?*
2. *What are SWE challenges we have discussed? identify other problems and challenges that software engineering is likely to face in the 21st century?*
3. *What are the four important attributes that all professional software should have?*
4. *Explain the four dimensions of dependability?*
5. *It is accepted for what type of software engineering that it is impractical to specify all requirements for such a system in advance?*

Outline

- *Professional software development*
  - What is meant by software engineering?
  - Software process activities?
  - Software Cost?
- *Software Engineering Methods*
  - Key challenges
  - Application types
  - CASE tools
- ***Software engineering ethics***
  - A brief introduction to ethical issues that affect software engineering.
  - ACM/IEEE Code of Ethics

Professional and Ethical Responsibility

- *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**.*

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 outwith 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.

- ***Computer misuse***
  - Software engineers should not use their technical skills to misuse other people's computers. Computer misuse ranges from relatively trivial (game playing) to extremely serious (dissemination of viruses).

ACM/IEEE Code of Ethics

- *The professional societies in the US have cooperated to produce a code of ethical practice.*
  - IEEE (Institute of Electrical and Electronic Engineering)
  - ACM (Association for Computing Machinery)
- *The Code contains eight Principles related to the behaviour of and decisions made by **professional** software engineers.*

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:

CoE Principle #1: Public

- *Software engineers shall act consistently with the public interest.*
  - Accept full responsibility for their own work.
  - Approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy or harm the environment.

CoE Principle #2: *Client & Employer*

- *Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest.*
  - Provide service in their areas of competence, being honest about any limitations of their experience.
  - Not knowingly use software that is obtained or retained either illegally or unethically.
  - Keep private any confidential information gained in their professional work

CoE Principle #3: *Product*

- *Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.*
  - strive for high quality, acceptable cost and a reasonable schedule.
  - Ensure proper and achievable goals and objectives for any project on which they work or propose.
  - Identify, define and address ethical, economic, cultural, legal and environmental issues related to work projects.

CoE Principle #4: Judgment

- *SW engineers shall maintain integrity and independence in their professional judgment.*
  - Only endorse documents either prepared under their supervision or within their areas of competence and with which they are in agreement.
  - Maintain professional objectivity with respect to any software or related documents they are asked to evaluate.
  - Not engage in deceptive financial practices such as bribery, double billing, or other improper financial practices.

CoE Principle #5: Management

- *Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.*
  - Attract potential software engineers only by full and accurate description of the conditions of employment.
  - Not ask a software engineer to do anything inconsistent with this Code.
  - Not punish anyone for expressing ethical concerns about a project.

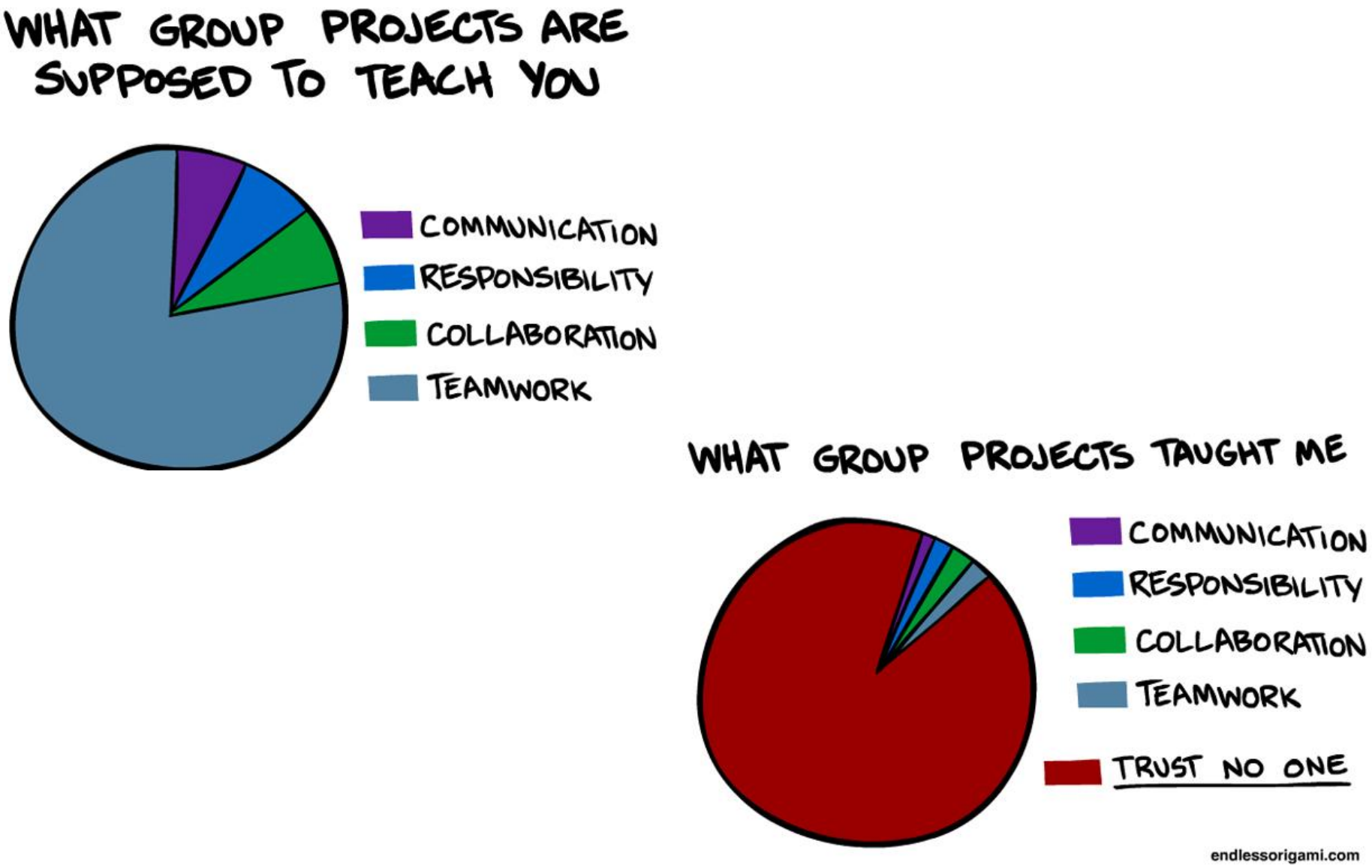
CoE Principle #6: Profession

- *Software engineers shall advance the integrity and reputation of the profession consistent with the public interest*
  - Help develop an organizational environment favorable to acting ethically.
  - Promote public knowledge of software engineering.
  - Extend software engineering knowledge by appropriate participation in professional organizations, meetings and publications.

CoE Principle #7: Colleagues

- *Software engineers shall be fair to and supportive of their colleagues.*
  - Encourage colleagues to adhere to this Code.
  - Assist colleagues in professional development.
  - Credit fully the work of others
  - Review the work of others in an objective, and properly-documented way.

CoE Principle #7: Colleagues



CoE Principle #8: Self

- *Software engineers shall participate in life-long learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.*
  - Further their knowledge of developments
  - Improve their ability to create safe, reliable, and useful quality software at reasonable cost and within a reasonable time.
  - Improve their ability to produce accurate, informative, and well-written documentation.

The ACM/IEEE Code of Ethics

Key Points

- *Software engineers have responsibilities to the engineering profession and society. They should not simply be concerned with technical issues.*
- *Professional societies publish codes of conduct which set out the standards of behavior expected of their members.*
- *But wait, it is not that easy !!!*

Review Questions

1. *To help counter terrorism, many countries are planning or have developed computer systems that track large numbers of their citizens and their actions. Clearly this has privacy implications. Discuss the ethics of working on the development of this type of system.*
2. *In the Software Engineering Code of Ethics what is the Colleagues principle?*
3. *Which of the eight principles are most important from your point of view?*