Chapter 1- Introduction

Software engineering

- The economies of ALL developed and many developing nations are dependent on software.
- More and more systems are software controlled
- Software engineering is concerned with theories, methods and tools for professional software development.
- Expenditure on software represents a significant fraction of GNP in all developed countries.

Largest Public Companies in the World (2000)

This *Financial Times*–based list is up to date as of 31 March 2001.[79]

Rank	Name	Headquarters	Primary industry	Market value (USD million)
1	General Electric	United States	Conglomerate	477,406
2	Cisco Systems	United States	Networking hardware	304,699
3	Exxon Mobil	United States	Oil and gas	286,367
4	Pfizer	United States	Health care	263,996
5	Microsoft	United States	Software industry	258,436
6	Wal-Mart	United States	Retail	250,955
7	Citigroup	United States	Banking	250,143
8	Vodafone	United Kingdom	Telecommunications	227,175
9	Intel Corporation	United States	Computer hardware	227,048
10	Royal Dutch Shell	The Netherlands	Oil and gas	206,340

Source: Wikipedia

Largest Public Companies in the World (2010)

This *Financial Times* Global 500–based list is up to date as of December 31, 2009. Indicated changes in market value are relative to the previous quarter.

Rank	First quarter[59]	Second quarter[60]	Third quarter[61]	Fourth quarter[62]
1	Exxon Mobil ▼336,527	PetroChina ▲366,662.9	Exxon Mobil ▼329,725	PetroChina ▲353,140.1
2	PetroChina ▲287,185	Exxon Mobil ▲341,140.3	PetroChina ▼325,097.5	Exxon Mobil ▼323,717.1
3	Wal-Mart ▼204,365	ICBC ▲257,004.4	ICBC ▼237,951.5	Microsoft ▲270,635.4
4	ICBC ▲187,885	Microsoft ▲211,546.2	Microsoft ▲229,630.7	ICBC ▲268,956.2
5	China Mobile ▼174,673	China Mobile ▲200,832.4	HSBC ▲198,561.1	Wal-Mart ▲203,653.6
6	Microsoft ▼163,320	Wal-Mart ▼188,752.0	China Mobile ▼195,680.4	China Construction Bank ▲201,436.1
7	AT&T ▼148,511	China Construction Bank ▲182,186.7	Wal-Mart ▲189,331.6	BHP Billiton ▲201,248
8	Johnson & Johnson ▼145,481	Petrobras ▲165,056.9	Petrobras ▲189,027.7	HSBC ▲199,254.9
9	Royal Dutch Shell ▼138,999	Johnson & Johnson ▲156,515.9	China Construction Bank ▲186,816.7	Petrobras ▲199,107.9
10	Procter & Gamble ▼138,013	Royal Dutch Shell ▲156,386.7	Royal Dutch Shell ▲175,986.1	Apple Inc. ▲189,801.7

Source: Wikipedia

Largest Public Companies in the World (2021)

This list is up to date as of June 30, 2021. Indicated changes in market value are relative to the previous quarter.

Rank	First quarter	Second quarter
1	Apple ▼2,050,000[16]	Apple ▲2,286,000[16]
2	Microsoft ▲1,778,000[17]	Microsoft ▲2,040,000[17]
3	Amazon ▼1,558,000[18]	Amazon ▲1,735,000[18]
4	Alphabet ▲1,395,000[19]	Alphabet ▲1,680,000[19]
5	Facebook ▲838,720[20]	Facebook ▲985,920[20]
6	Tencent ▲766,970[21]	Tencent ▼721,460[21]
7	Tesla ▼641,110[22]	Tesla ▲654,780[22]
8	Alibaba Group ▼615,010[23]	Berkshire Hathaway ▲637,280[24]
9	TSMC ▲613,410[25]	TSMC ▲623,160[25]
10	Berkshire Hathaway ▲590,050[24]	Alibaba Group ▲615,140[23]

Source: Wikipedia

Which industry is growing the fastest?

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. For systems with a long life, maintenance costs may be several times development costs.
- Software engineering is concerned with cost-effective software development.

Software project failure

✧ *Increasing system complexity*

- As new software engineering techniques help us to build larger, more complex systems, the demands change. Systems have to be built and delivered more quickly; larger, even more complex systems are required; systems have to have new capabilities that were previously thought to be impossible.

✧ *Failure to use software engineering methods*

- It is fairly easy to write computer programs without using software engineering methods and techniques. Many companies have drifted into software development as their products and services have evolved. They do not use software engineering methods in their everyday work. Consequently, their software is often more expensive and less reliable than it should be.

Professional software development

Frequently asked questions about software engineering

Question	Answer
What is software?	Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.
What are the attributes of good software?	Good software should deliver the required functionality and performance to the user and should be maintainable, dependable and usable.
What is software engineering?	Software engineering is an engineering discipline that is concerned with all aspects of software production.
What are the fundamental software engineering activities?	Software specification, software development, software validation and software evolution.
What is the difference between software engineering and computer science?	Computer science focuses on theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software.
What is the difference between software engineering and 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.

Frequently asked questions about software engineering

Question	Answer
What are the key challenges facing software engineering?	Coping with increasing diversity, demands for reduced delivery times and developing trustworthy software.
What are the costs of software engineering?	Roughly 60% of software costs are development costs, 40% are testing costs. For custom software, evolution costs often exceed development costs.
What are the best software engineering techniques and methods?	While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system. For example, games should always be developed using a series of prototypes whereas safety critical control systems require a complete and analyzable specification to be developed. You can't, therefore, say that one method is better than another.
What differences has the web made to software engineering?	The web has led to the availability of software services and the possibility of developing highly distributed service-based systems. Web-based systems development has led to important advances in programming languages and software reuse.

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; CAD software; software for specific markets such as appointments systems for dentists.
- ✧ Customized products
  - Software that is commissioned by a specific customer to meet their own needs.
  - Examples – embedded control systems, air traffic control software, traffic monitoring systems.

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.

Essential attributes of good software

Product characteristic	Description
Maintainability	Software should be written in such a way so that it can evolve to meet the changing needs of customers. This is a critical attribute because software change is an inevitable requirement of a changing business environment.
Dependability and security	Software dependability includes a range of characteristics including reliability, security and safety. Dependable software should not cause physical or economic damage in the event of system failure. Malicious users should not be able to access or damage the system.
Efficiency	Software should not make wasteful use of system resources such as memory and processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilisation, etc.
Acceptability	Software must be acceptable to the type of users for which it is designed. This means that it must be understandable, usable and compatible with other systems that they use.

Software engineering

✧ Software engineering

- Engineering discipline
  - Using appropriate theories and methods to solve problems bearing in mind organizational and financial constraints.

- Concerned with all aspects of software production
  - from the early stages of system specification through to maintaining the system after it has gone into use.
  - Not just technical process of development. Also project management and the development of tools, methods etc. to support software production.

Importance of software engineering

- Individuals and society rely on advanced software systems.
- Need to produce reliable and trustworthy systems economically and quickly.
- It is usually cheaper
  - To use software engineering methods and techniques for software systems
  - Rather than just write the programs
  - For most types of system, the majority of costs are the costs of changing the software after it has gone into use.

Software process activities

- Software specification
  - Customers and engineers define the software that is to be produced and the constraints on its operation.
- Software development
  - The software is designed and programmed.
- Software validation
  - The software is checked to ensure that it is what the customer requires.
- Software evolution
  - The software is modified to reflect changing customer and market requirements.

General issues that affect software

- Heterogeneity
  - Increasingly, systems are required to operate as distributed systems across networks that include different types of computer and mobile devices.
- Business and social change
  - 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.

General issues that affect software

- Security and trust
  - As software is intertwined with all aspects of our lives, it is essential that we can trust that software.
- Scale
  - Software has to be developed across a very wide range of scales, from very small embedded systems in portable or wearable devices through to Internet-scale, cloud-based systems that serve a global community.

Software engineering diversity

- Many different types of software
  - No universal set of software techniques

- The software engineering methods and tools
  - Depend on the type of application being developed
  - Requirements of the customer
  - 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. They include all necessary functionality and do not need to be connected to a network.
- 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.
- Embedded control systems
  - These are software control systems that control and manage hardware devices. Numerically, there are probably more embedded systems than any other type of system.

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.
- Entertainment systems
  - These are systems that are primarily for personal use and which are intended to entertain the user.
- 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.

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.
- 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.

Internet software engineering

- The Web is now a platform for running application and organizations are increasingly developing web-based systems rather than local systems.
- Web services allow application functionality to be accessed over the web.
- Cloud computing is an approach to the provision of computer services where applications run remotely on the 'cloud'.
  - Users do not buy software but pay according to use.

Web-based software engineering

- Web-based systems
  - Complex distributed systems
  - The fundamental principles of software engineering are as applicable to them as they are to any other types of system.
- Software reuse is the dominant approach for constructing web-based systems.
  - When building these systems, you think about how you can assemble them from pre-existing software components and systems

Key points

- Software engineering is an engineering discipline that is concerned with all aspects of software production.
- Essential software product attributes are maintainability, dependability and security, efficiency and acceptability.
- The high-level activities of specification, development, validation and evolution are part of all software processes.

Key points

- There are many different types of system and each requires appropriate software engineering tools and techniques for their development.

- The fundamental ideas of software engineering are applicable to all types of software system.

Room/Group leader

✧ In each room leader will be chosen based on the following:

Assume:

$rem_i = id_i \% 7 \; i = 1 \ldots n$

The student with smallest rem_i is the leader. If two or more students have similar rem_i , the student with smaller id will be the leader.

Group discussion

✧ How hardware and software are different?

✧ Hardware can be any tangible item not just computers.

✧ 12 min to discuss in your group

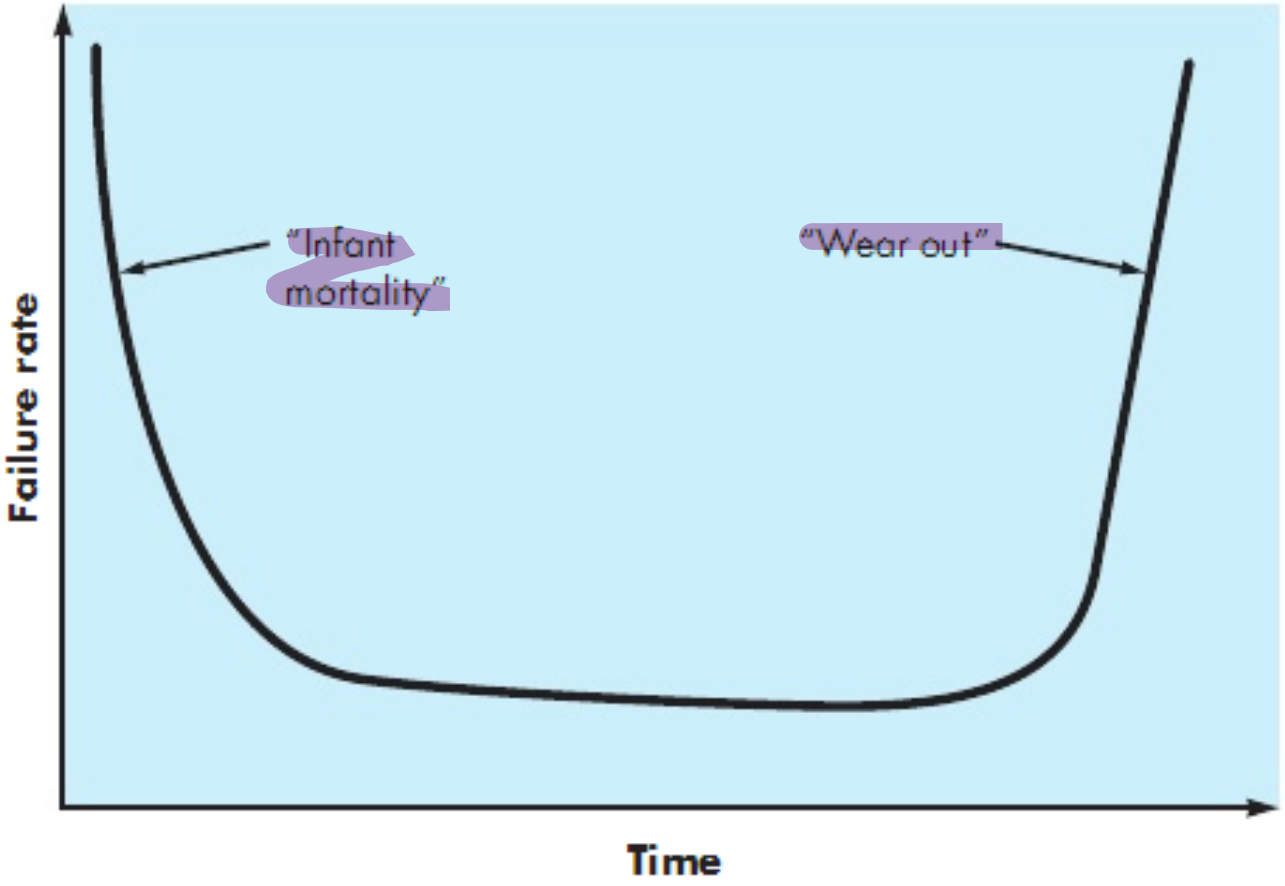
✧ Leaders will let us know what the group thinks

Hardware vs. software

✧ Hardware systems

- Physical
- Cost: mainly material cost and manufacturing cost
- Wear out
  - Dust, vibrations, extreme temperatures, abuse, etc.

Hardware



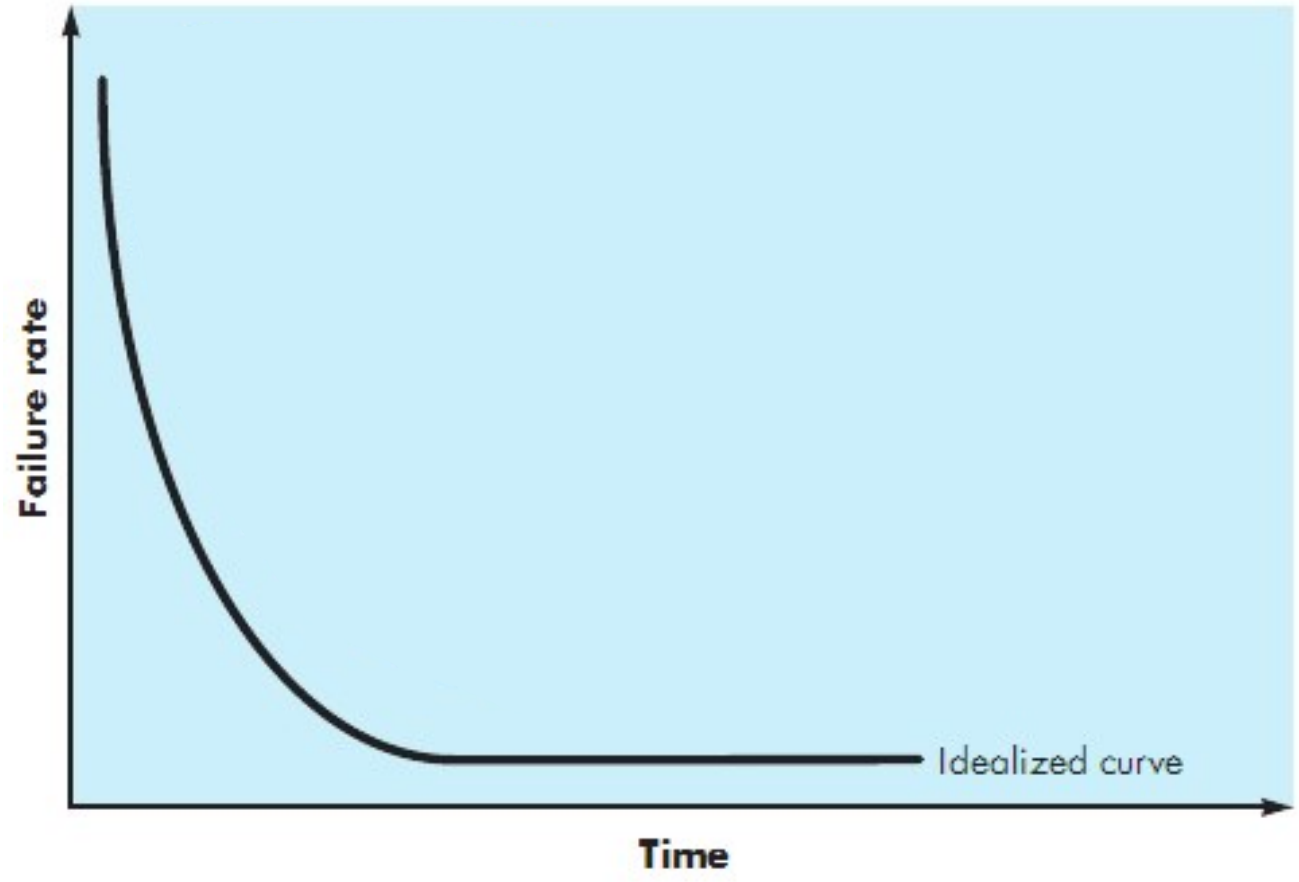
Hardware vs. software

✧ Software

- Logical product
  - Itself is a product
  - Vehicle for other software products
- Not affected by environmental maladies
- Doesn’t wear out
  - But will deteriorate

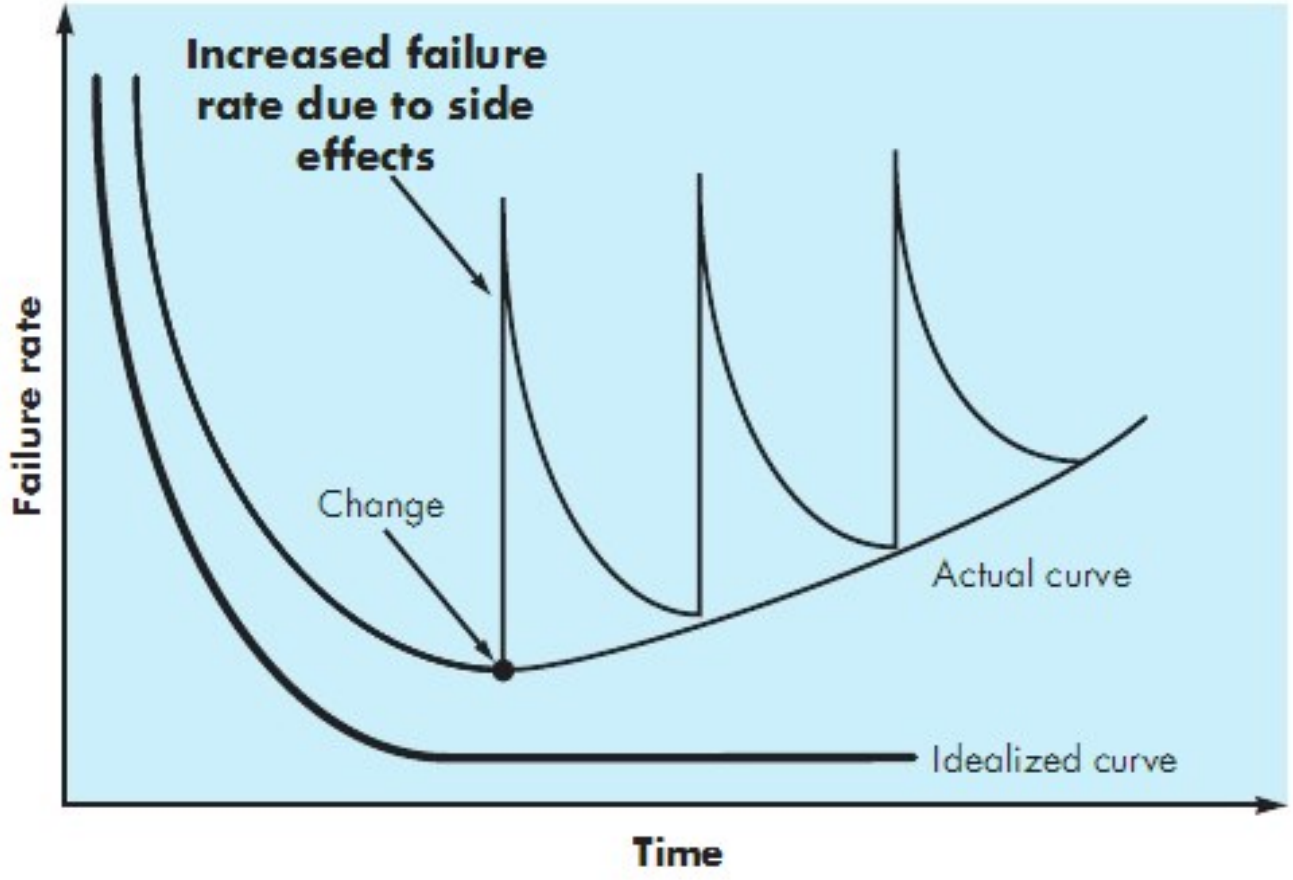
Software


Software Engineering
Ian Sommerville


Failure rate
Idealized curve
Time

Software


Software Engineering
Ian Sommerville


Increased failure rate due to side effects
Failure rate
Change
Actual curve
Idealized curve
Time

Software engineering ethics

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 above 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).

ACM/IEEE Code of Ethics

- The professional societies in the US have cooperated to produce a code of ethical practice.
- Members of these organisations sign up to the code of practice when they join.
- The Code contains eight Principles related to the behaviour of and decisions made by professional software engineers, including practitioners, educators, managers, supervisors and policy makers, as well as trainees and students of the profession.

Rationale for the code of ethics

- *Computers have a central and growing role in commerce, industry, government, medicine, education, entertainment and society at large. Software engineers are those who contribute by direct participation or by teaching, to the analysis, specification, design, development, certification, maintenance and testing of software systems.*
- *Because of their roles in developing software systems, software engineers have significant opportunities to do good or cause harm, to enable others to do good or cause harm, or to influence others to do good or cause harm. To ensure, as much as possible, that their efforts will be used for good, software engineers must commit themselves to making software engineering a beneficial and respected profession.*

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.