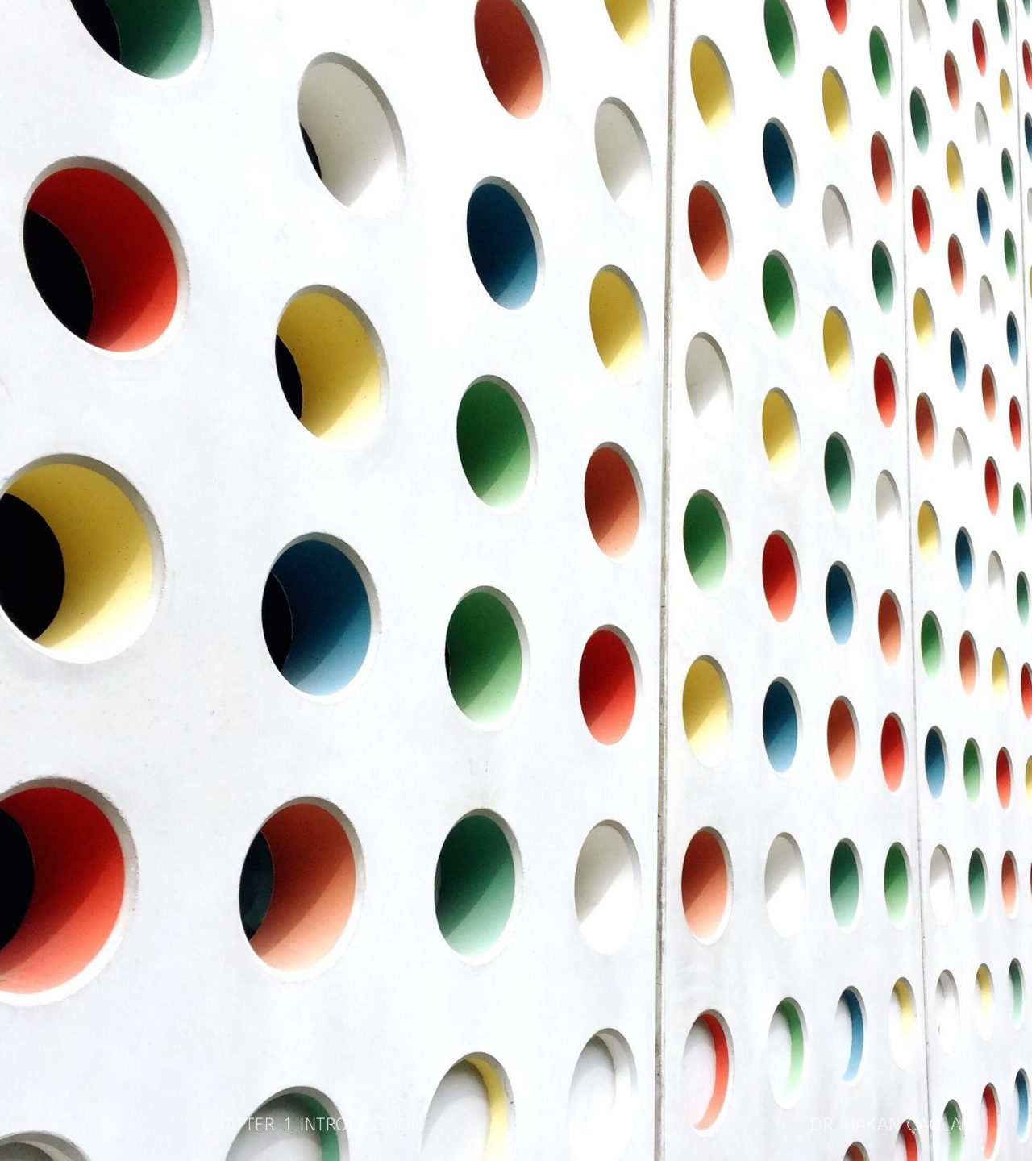


SENG-321 Software Engineering

Prof.Dr. Hakan Çağlar

Chapter 1- Introduction

Text Books

- Software Engineering 10th Edition by Ian Sommerville, Pearson Education, 2015

- Software Engineering: A Practitioner' S Approach; 8th Edition by Roger S. Pressman, Bruce R. Maximm Mc Graw Hill Education
- IEEE Computer Society, "A Guide to the Software Engineering Body of Knowledge"

Class Hours:

SENG 321 -1	Soft. Eng.	Monday	12:00-15:00	C026
SENG 321 -2	Soft. Eng.	Tuesday	15:00-18:00	C026
SENG 321 -3	Soft. Eng.	Thursday	15:00-18:00	C036

Office Hour:

Monday 10:30 – 11:30

Tuesday 10:30 – 11:30

Assistant: Arş.Gör Cansu Yoruk

e-mail: cansu.yoruk@ankarabilim.edu.tr

Course Grading

- Mid Term Exam % 20
- SW Project (SRS, UML Diagrams) % 40
- Final Exam % 40

Software Engineering Course Outline

- Introduction to Software Engineering
  - Professional software development
    - What is meant by software engineering.
  - Software engineering ethics
    - A brief introduction to ethical issues that affect software engineering.
  - Case studies

Software Engineering Course Outline

- The Software Process
  - Waterfall Model
  - Incremental Process Model
  - Evolutionary Process Model
  - Agile Development Process Model
  - V-Model and Work Products
  - Software Engineering Standards

Software Engineering Course Outline

- Sytems Engineering
  - Introduction to System Engineering
  - Systems Engineering History
  - Systems Engineering Definition
  - Key Concepts & Tasks of Systems Engineering
  - Systems Engineering Process
  - Systems Engineering Standarts

Software Engineering Course Outline

- Requirement Engineering & Requirement Analysis
  - Definition and Type of Requirements
  - Requirements Elucidate (Capture)
  - Requirement Development
  - Requirement's Template
  - Requirement Quality (Automatic Requirement Quality Management),
  - Requirement Parsing, (IRQ, Case studies)

Software Engineering Course Outline

- Analysis Model & Requirement Allocation
  - Requiremet Analysis
  - Analysis Modelling and Views
  - Context Diagram, Opperational view, Functional view, Physical view
  - Functional Decomposition, ER-Diagrams, Data Flow Diagrams
  - Business Process Reengineering (BPR), Business Process Improvment (BPI)
  - Object Oriented Analysis & Design (UML Modelling)

Object Oriented Analysis & Design

Object Oriented Design Fundamentals

- Object & Class
- Abstraction & Encapsulation
- Inheritance, Polymorphism

Unified Modeling Language (UML)

- Use-Case Diagrams
- Class Diagrams
- Sequence Diagrams
- Activity Diagrams

Software Engineering Course Outline

- System Design & Architecture
  - System Modelling
  - Interaction models
  - Structural models
  - Architectural design decisions
  - Architectural views
  - Architectural patterns
  - Application architectures

Software Engineering Course Outline

- Software Project Management
  - Software Project Management Techniques
  - Project Scheduling Management
  - Work Breakdown Structure (WBS) & Budget Management
  - Software Organizations & HR Management
  - Software Project Metrics
  - Risk Management
  - Software Quality Management and CMMI
  - Software Sizing and Estimation Techniques (Funcition points, COCOMO)

Software Engineering Course Outline

- Software Project Performance Tracking and Monitoring
  - Importance of tracking and monitoring
  - Tracking the performance
  - Monitoring the progress and resource
  - Visualizing Techniques
  - Earned Value Analysis

Software Engineering Course Outline

- Software Quality Management & Software SEI-CMMI
  - ISO 9001
  - CMM & CMMI

Software Engineering Course Outline

- Software Configuration Management
  - Software evolution (types of changes)
  - Configuration management (need for it)
  - Aspects of SCM
  - Change control board
  - Change management
  - Auditing and status accounting

Software Engineering Course Outline

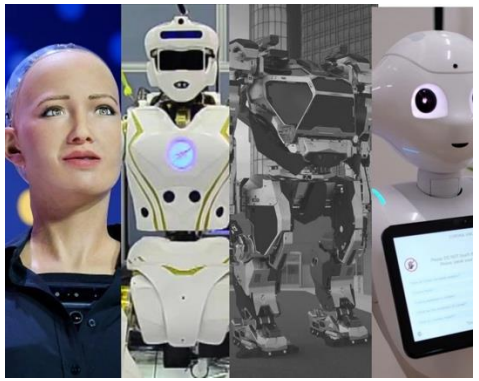
- Software Testing (Validation & Verification)
  - Software Testing Strategies
  - Test Case, Test Procedures, Test Tools
  - Unit Testing, System Testing, Entegrasyon Testing
  - Debugging Process and Reporting

Software Engineering Course Outline

- Information & Internet Security
  - What is Information Warfare
  - What is Security
  - Confidentiality, Integrity, Avaliability, Accountability, Reliability
  - Security Threats & Risk Analysis
  - Elements of Cryptography
  - Public Key Cryptography (PKI)
  - RSA, Digital Signiture, Hash Functions

Project Examples

- Humanoid Robots
- University Information Systems
- University Social Media
- Language Translation
- Smart Home Appliances - IoT
- Hospital Information Systems
- Smart Personnel Assitant
- Social Media Apps, …



AI-Powered English Education

1. Level Assessment (AI Diagnostic Engine)

- Before starting the course, the student takes an AI-powered placement test.
- This test analyzes speaking, writing, listening, and reading skills using Natural Language Processing (NLP).
- The test results are used to create a personalized learning plan (Adaptive Learning Path).

2. Personalized Learning Content (AI-Powered Adaptive Curriculum)

- Content is automatically selected or generated based on the student's level and goals.
- Skills such as grammar, vocabulary, pronunciation, and speaking are addressed individually.
- Lessons, exercises, (roleplays) are adapted by AI.

AI-Powered English Education

3. AI Tutor (Conversational AI / Virtual Tutor)

- Students can have written or spoken conversations with a GPT-based virtual tutor.
- This tutor:
  - Corrects pronunciation (via speech-to-text analysis),
  - Instantly points out grammar mistakes,
  - Provides meaningful feedback,
  - Explains new words in context.

- Role-play scenarios (e.g., conversation at the airport, job interview) are simulated by AI.

AI-Powered English Education

4. Automated Assessment and Feedback (AI Assessment & Feedback)

- After each speaking, writing, or quiz activity, AI evaluates the student's performance.
- Mistakes, challenges, and areas for improvement are identified.
- A weekly "progress report" is provided to the student (progress tracking).

5. Real-Time Speaking Partner (AI Speaking Companion)

- Students can practice English 24/7 with a ChatGPT-like system.
- Both voice and text communication are supported.
- Spoken responses are analyzed using NLP; feedback is given based on fluency, word choice, and other criteria.

6. Content Generation (AI Content Generator)

- Texts, stories, or news articles are generated based on the student's interests.
- For example, if the student is interested in football, football news is provided as English learning material.

Topics covered

- Professional software development
  - What is meant by software engineering.
- Software engineering ethics
  - A brief introduction to ethical issues that affect software engineering.
- Case studies

Software engineering

The economies of ALL developed nations are dependent on software.

More and more systems are software controlled (Autonomous systems, Cars, Smart Home Appliances, Smart TV, Airplane, UAV,)

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

Software development process (products, people, process 3P)

Expenditure on software represents a significant fraction of GNP in all developed countries.

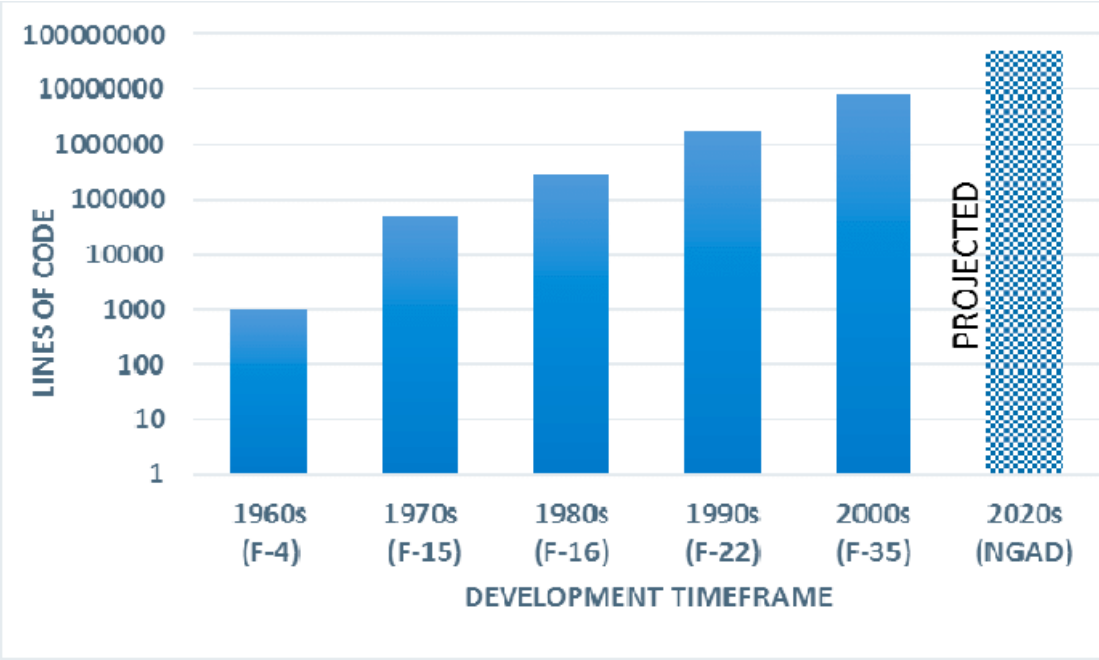
Software costs

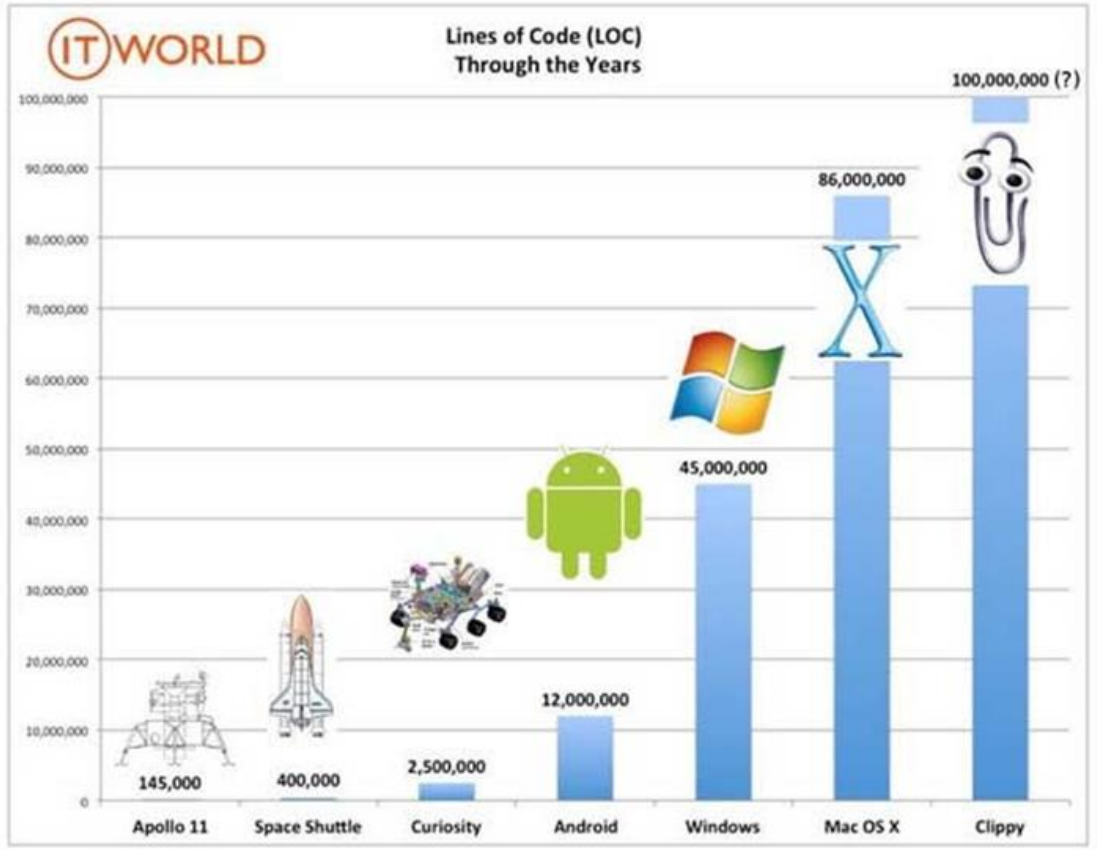
Software costs often dominate computer system costs. The costs of software on a PC are often greater than the hardware cost.

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., cost-effective software maintenance & update program.

Software Lines of Codes





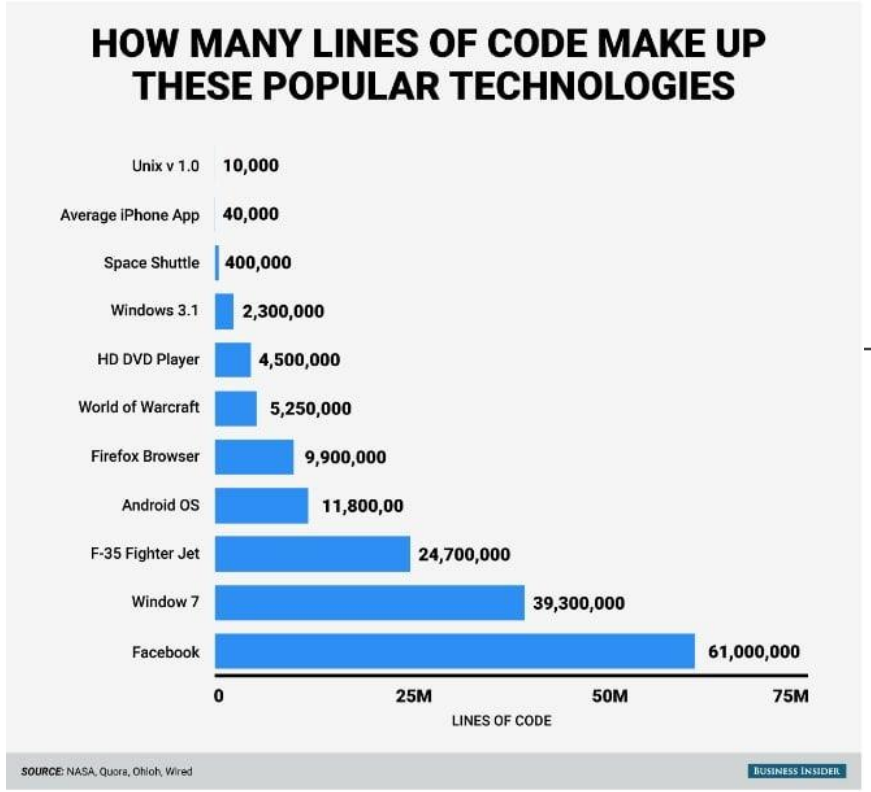
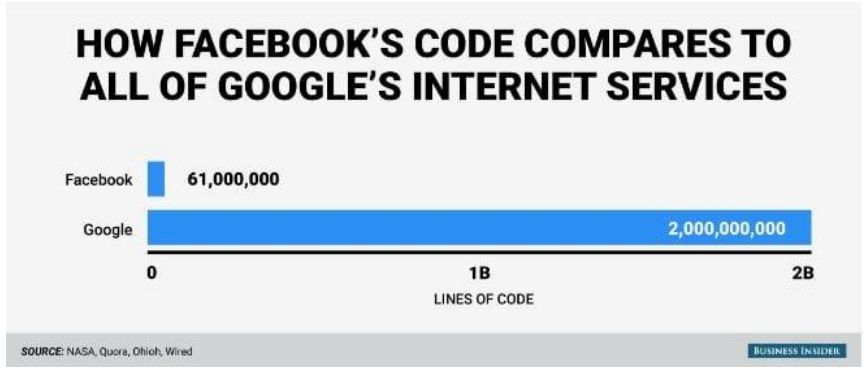


Image: Business Insider



50 SLOC about 1 page
100 Mil. SLOC 2.000.000 pages
1 Mil. SLOC 20.000 pages

Software project failure

Increasing system complexity;

As new software engineering techniques help us to build larger, more complex systems, the demands change. Failure rate of Large SW Projects are more than 85%.

Failure Reasons:

Insufficiently defined needs (req.)	13.1%	Lack of resources	10.6%
No user engagement	12.4%	Lack of top management support	9.3%
Change of needs and specifications	8.7%	Lack of planning	8,1%
Unrealistic expectations	9.9%		

Software project failure

Increasing system complexity;

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.

It is fairly easy to write computer programs without using software engineering methods and techniques.

Many companies do not use software engineering methods in their everyday work. Consequently, their software is often more expensive and less reliable than it should be.

ANKARA
SCIENCE UNIVERSITY

Professional software development

ANKARA SCIENCE UNIVERSITY

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.

Frequently asked questions about software engineering

Question	Answer
What are the fundamental software engineering activities?	Software specification, software design & development, software validation (Test) and software evolution (Upgrade & Maintanece).
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 systems 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.

ANKARA SCIENCE UNIVERSITY

Frequently asked questions about software engineering

Question	Answer
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.

ANKARA SCIENCE UNIVERSITY

Frequently asked questions about software engineering

Question	Answer
What are the key challenges facing software engineering?	Coping with increasing diversity & complexity, 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.

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. COTS software.

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, factory management information system.

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 is an engineering discipline that is 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.

Engineering discipline

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

All aspects of software production

- Not just technical process of development. Also project management and the development of tools, methods etc. to support software production.

Importance of software engineering

More and more, individuals and society rely on advanced software systems. We need to be able to produce reliable and trustworthy systems economically and quickly.

It is usually cheaper, in the long run, to use software engineering methods and techniques for software systems rather than just write the programs as if it was a personal programming project.

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, where customers and engineers define the software that is to be produced and the constraints on its operation.

Software development, where the software is designed and programmed.

Software validation, where the software is checked to ensure that it is what the customer requires.

Software evolution, where 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

There are many different types of software system and there is 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. 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 buy pay according to use.

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 level.

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.

Ethical principles

I. PUBLIC - Software engineers shall act consistently with the public interest.

II. 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.

III. PRODUCT - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.

IV. JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.

Ethical principles

V. MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

VI. PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.

VII. COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.

VIII. 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.

Ethical dilemmas

Disagreement in principle with the policies of senior management.

Your employer acts in an unethical way and releases a safety-critical system without finishing the testing of the system.

Participation in the development of military weapons systems or nuclear systems.

Case studies

Case studies

A personal insulin pump

- An embedded system in an insulin pump used by diabetics to maintain blood glucose control.

A mental health case patient management system

- Mentcare. A system used to maintain records of people receiving care for mental health problems.

Insulin pump control system

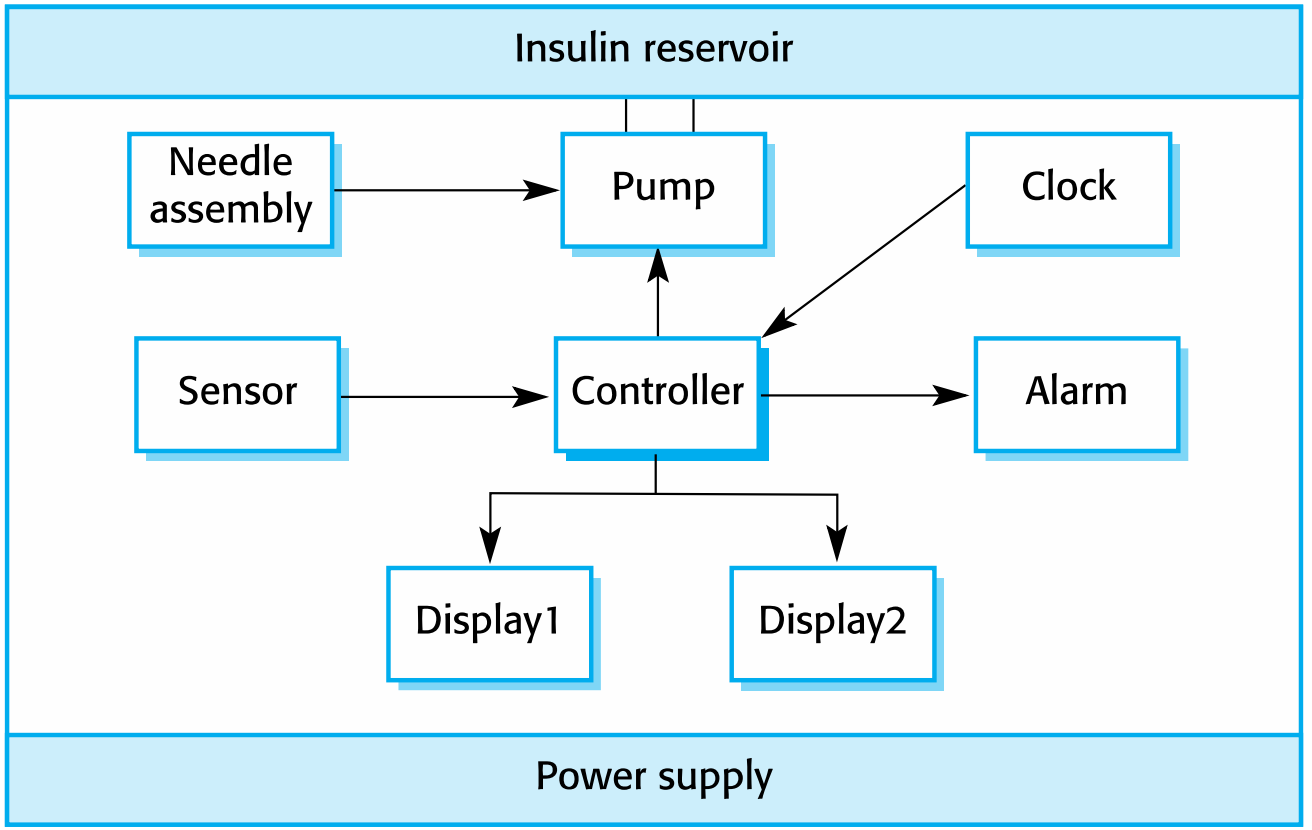
Collects data from a blood sugar sensor and calculates the amount of insulin required to be injected.

Calculation based on the rate of change of blood sugar levels.

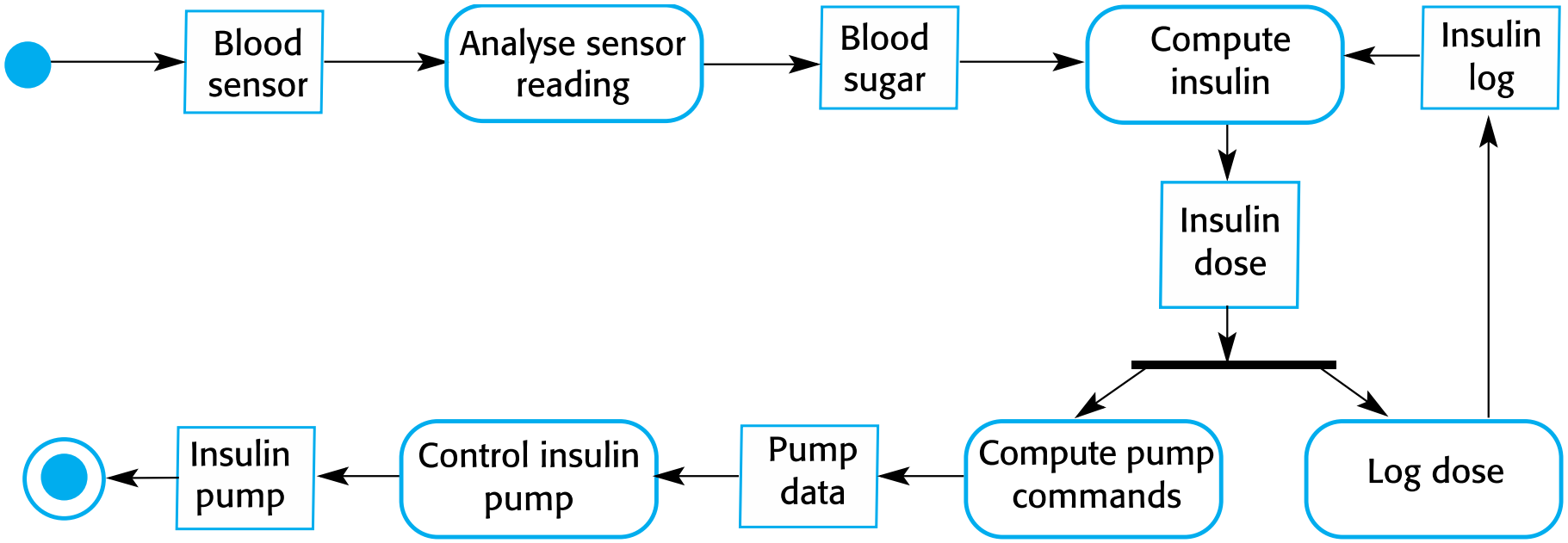
Sends signals to a micro-pump to deliver the correct dose of insulin.

Safety-critical system as low blood sugars can lead to brain malfunctioning, coma and death; high-blood sugar levels have long-term consequences such as eye and kidney damage.

Insulin pump hardware architecture



Activity model of the insulin pump



Essential high-level requirements

The system shall be available to deliver insulin when required.

The system shall perform reliably and deliver the correct amount of insulin to counteract the current level of blood sugar.

The system shall therefore be designed and implemented to ensure that the system always meets these requirements.

Mentcare: A patient information system for mental health care

A patient information system to support mental health care is a medical information system that maintains information about patients suffering from mental health problems and the treatments that they have received.

Most mental health patients do not require dedicated hospital treatment but need to attend specialist clinics regularly where they can meet a doctor who has detailed knowledge of their problems.

To make it easier for patients to attend, these clinics are not just run in hospitals. They may also be held in local medical practices or community centres.

Mentcare

Mentcare is an information system that is intended for use in clinics.

It makes use of a centralized database of patient information but has also been designed to run on a PC, so that it may be accessed and used from sites that do not have secure network connectivity.

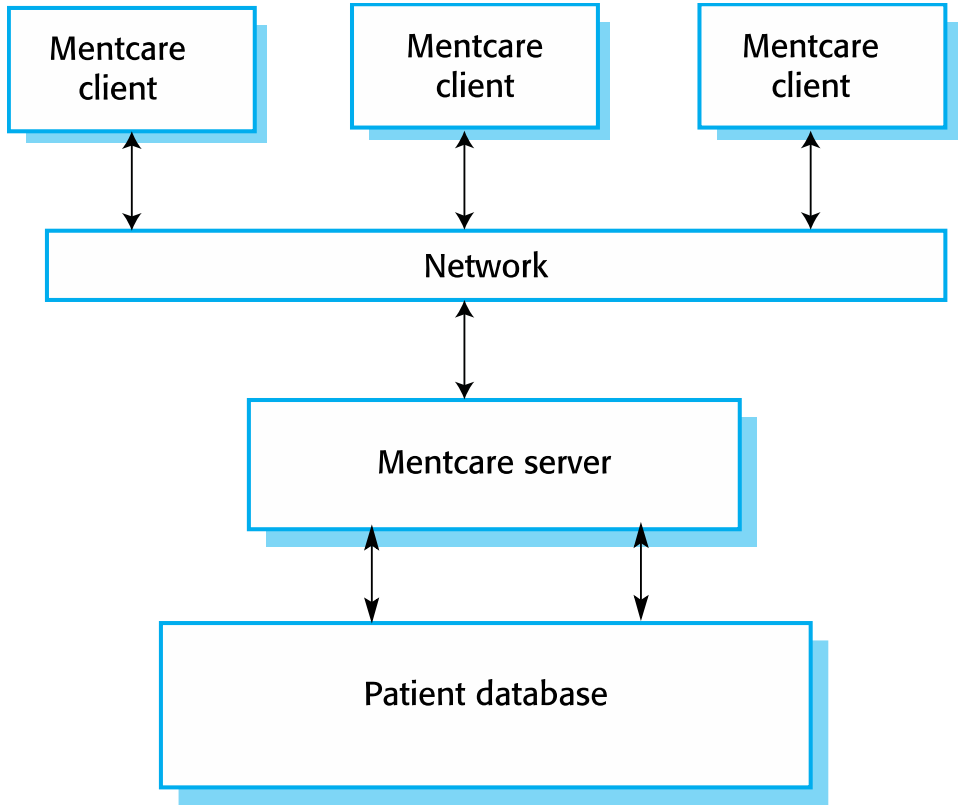
When the local systems have secure network access, they use patient information in the database but they can download and use local copies of patient records when they are disconnected.

Mentcare goals

To generate management information that allows health service managers to assess performance against local and government targets.

To provide medical staff with timely information to support the treatment of patients.

The organization of the Mentcare system



Key features of the Mentcare system

Individual care management

- Clinicians can create records for patients, edit the information in the system, view patient history, etc. The system supports data summaries so that doctors can quickly learn about the key problems and treatments that have been prescribed.

Patient monitoring

- The system monitors the records of patients that are involved in treatment and issues warnings if possible problems are detected.

Administrative reporting

- The system generates monthly management reports showing the number of patients treated at each clinic, the number of patients who have entered and left the care system, number of patients sectioned, the drugs prescribed and their costs, etc.

Mentcare system concerns

Privacy

- It is essential that patient information is confidential and is never disclosed to anyone apart from authorised medical staff and the patient themselves.

Safety

- Some mental illnesses cause patients to become suicidal or a danger to other people. Wherever possible, the system should warn medical staff about potentially suicidal or dangerous patients.
- The system must be available when needed otherwise safety may be compromised and it may be impossible to prescribe the correct medication to patients.

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, design & development, validation and evolution are part of all software processes.

The fundamental notions of software engineering are universally applicable to all types of system development.

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.

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 behaviour expected of their members.