L1 - Introduction

Topics covered

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

Software engineering

- The economies of ALL developed 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.

Think-Pair-Share

- Can we use the same method and tool to develop i) an information system, ii) a controller, and iii) a game? Why?

Information system

Controller

Game

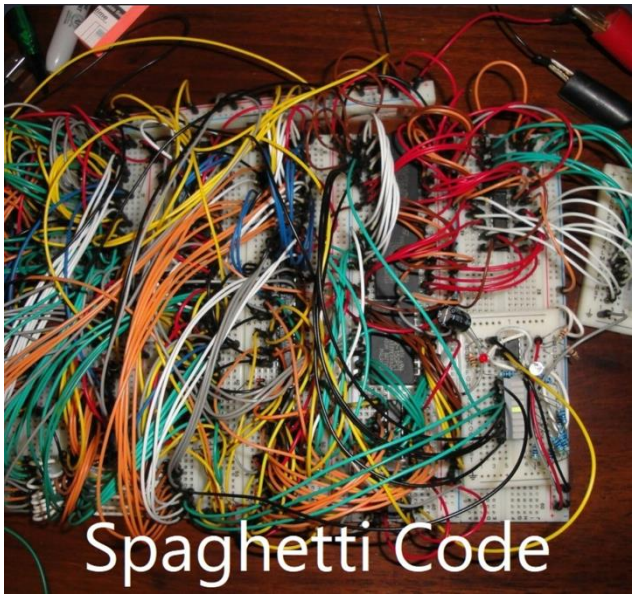
Think Individually → Discuss with Groupmates → Share in Class

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
  - New SE techniques help us to build **larger** and **more complex** systems, and delivered **more quickly**
- ✧ Failure to use software engineering methods
  - It is fairly easy to write computer programs **without using SE methods and techniques**
  - Consequently, their software is often **more expensive and less reliable** than it should be.

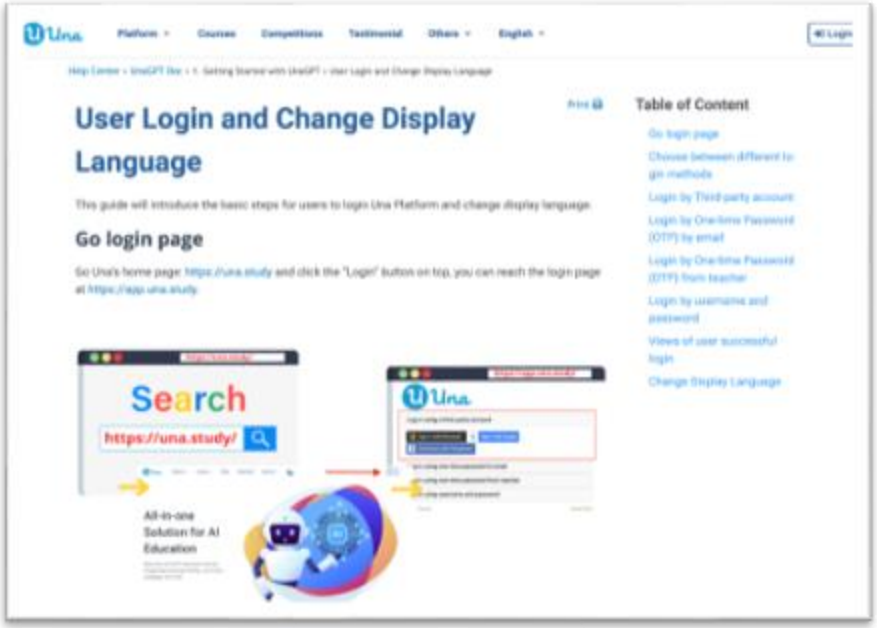


Professional software development

Real examples - UnaGPT on Una Platform: AI Auto Grading and Feedback System

Associated documentation of Una Platform

✧ For example, the user guides of Una Platform are created by WordPress (https://una.study/help-center/)



User view of a document

WordPress's editor view

Professional software engineers create good software

- What is software?
  - Computer programs and **associated documentation**
  - may be developed for a particular customer or a general market
- What are the attributes of good software?
  - should deliver the required **functionality** and **performance** to the user and should be **maintainable**, **dependable** and **usable**.

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 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

- ✧ Product specification is the specification of **what the software should do**
- ✧ Generic products are owned by the **software developers**
- ✧ Customized products are owned by the **customer**
- ✧ Decisions on software change are made by the owners.

e.g. Una Platform: Real-time interactive AI education platform

e.g. Making Smart Choices 2: Sexuality Education Experiential Learning Game

Question: What if an accident happens?

Software engineering

✧ Definition of SE:

- *"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

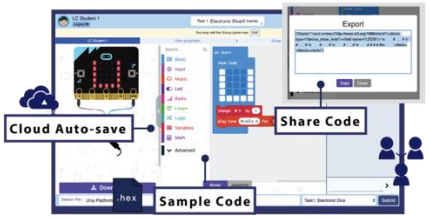
- More and more, individuals and society rely on advanced software systems. We need to be able to produce **reliable** and **trustworthy** systems **economically** and **quickly**.
- It is usually **cheaper**, in the long run, to **use software engineering methods and techniques** for software systems rather than just write the programs as if it was a personal programming project.

Software process activities

1. Software **specification**, where customers and engineers define the software that is to be produced and the constraints on its operation.
2. Software **development**, where the software is designed and programmed.
3. Software **validation**, where the software is checked to ensure that it is what the customer requires.
4. Software **evolution**, where the software is modified to reflect changing customer and market requirements.

General issues that affect software

- ✧ Heterogeneity



Example: Una Platform for AI Education does not support mobile version, due to the requirement of a large screen size for blocks coding

- ✧ Business and social change

Example: Web browser does not support Flash games anymore. Thus, we revamp the games to an HTML5 version.

- ✧ Security and trust

Hackers?

AI?

- ✧ Scale

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 (1)

- Stand-alone applications
  - run on a local computer without connecting to a network
- Interactive transaction-based applications
  - execute on a remote computer and accessed by users from their own PCs.
- Embedded control systems
  - control and manage hardware devices.
- Batch processing systems
  - process data in large batches.

Application types (2)

- Entertainment systems
  - primarily for personal use and intended to entertain the user
- Systems for modelling and simulation
  - model physical processes or situations, which include many, separate, interacting objects.
- Data collection systems
  - collect data from their environment using a set of sensors and send that data to other systems for processing.
- Systems of systems
  - 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.

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 outwith their competence.

VR or METAVERSE?

Issues of professional responsibility

- Intellectual Property (IP) 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 IP 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 (e.g. game playing on an employer's machine) to extremely serious (e.g. dissemination of viruses or other malware).

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.
- 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.