SYSTEMS PLANNING

Introduction

Class content

What to expect from this class?

- An **introduction to software engineering** and its goals
- An introduction to the **software process**
- **Fundamental ideas and aspects** of software engineering
- Types of **software systems**
- Software engineering **ethics**

Source: https://pixabay.com/en/checklist-check-list-marker-2077020/
by TeroVesalainen |CC0 Creative Commons

Class learning goals

MCI MANAGEMENT CENTER INNSBRUCK

What should be your learning outcome?

- Understand that software engineering is **more than programming**
- Understand the **software process**
- Know some **general issues** software engineering needs to deal with
- Be able to distinguish between **different types of software systems**
- Be aware of **ethical rules** with respect to software engineering

Source: https://pixabay.com/en/learning-hint-school-subject-3245792/ by harishs |CC0 Creative Commons

Introduction

MCI MANAGEMENT CENTER INNSBRUCK

Professional Software Development

Programming as (side) activity
- Business people write **spreadsheets**
- Scientists write **programs to process data**
- Hobbyists write programs for their **own interest**

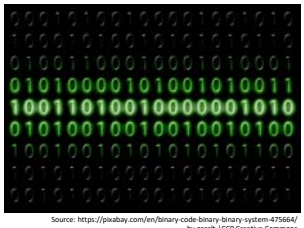
Professional programming
- Software for **specific business purposes**
  E.g. information systems, CAD systems
- Used by someone **apart from the developer**
- **Maintained** and **changed** over time
- Needs **software/system engineering**

Source: https://pixabay.com/en/mark-marker-hand-write-516279/ by geralt |CC0 Creative Commons

Introduction

Programming vs. Software Engineering

Software is not just the programs themselves but
- Associated **configuration files**
- **System documentation** describing the structure of the system
- **User documentation** describing how to use the system
- **Websites** to download recent product information and updates



Source: https://pixabay.com/en/binary-code-binary-binary-system-475664/ by geralt |CC0 Creative Commons

"Software engineers are concerned with developing **software products** (i.e. software which can be sold to a customer)."

Introduction

Two kinds of software products

Generic products
- **stand-alone systems** sold on the open market
  E.g. word processors, drawing packages, project management tools, etc.
- Also so-called **vertical applications** such as accounting systems

Customized (or bespoke) products
- **Commissioned** by a customer
- Built by a **contractor**
  E.g. **control systems** for electronic devices, systems to **support particular business processes**, etc.

Source: https://pixabay.com/en/software-cd-dvd-digital-disc-417880/ by geralt |CC0 Creative Commons

Introduction

MCI MANAGEMENT CENTER INNSBRUCK

Two kinds of software products

Generic products
- **Developers control** the software specification
- The **development company estimates** what clients may need

Customized (or bespoke) products
- Customer develops the software specification
- The **client tells** the developer **what he/she needs**

But
- The **distinction** is becoming increasingly **blurred**
- Now often **generic products build the base** and are then adapted to the requirements of the customer
- E.g. ERP systems such as SAP

Source: https://pixabay.com/en/it-business-icons-computers-722950/ by ZeroTolerance |CC0 Creative Commons

Introduction

MCI MANAGEMENT CENTER INNSBRUCK

Software Quality

Various aspects to take into account
- Software **behavior while executed**
- Structure and organization of the components
- Documentation

These are non-functional software attributes
- E.g. response time, understandability, reliability, security
- Things that you would **expect from a software** (depending on its area of application)

Source: https://pixabay.com/en/label-quality-badge-sticker-emblem-1197365/ by GDJ |CC0 Creative Commons

Software Engineering as a discipline

Engineering discipline
- Engineers **make things work**
- Apply **theories**, **methods**, and **tools**
- **Discover solutions** to problems without theories
- Work within organizational and financial **constraints**

Different aspects of software production
- Not only code
- Project management activities
- Development of **tools**, **methods** an **theories** to support software production

Source: https://pixabay.com/en/macro-cogwheel-gear-engine-vintage-1452987/ by Pavlofox |CC0 Creative Commons

Software Engineering as a discipline

Constraints
- Getting results **within schedule and budget**
- Deal with **compromises**
- **Adapt** to the organization
- Find most appropriate method for given circumstance
- More **creative**, less formal approaches may be effective in **some cases; more traditional approaches** may work for **other settings**

"People writing programs for themselves, however, can spend as much time as they wish on the program development."

Source: https://pixabay.com/en/robot-artificial-intelligence-woman-507811/ by geralt |CC0 Creative Commons

Importance

Software Engineering is important because

- More and more individuals **rely on advanced software systems**. Hence, we need trustworthy systems.
- It is usually **cheaper in the long** run to use software engineering methods and techniques rather than just coding away (**Note: the majority of costs are costs of changing the software**)

Source: https://pixabay.com/en/note-thumbtack-reminder-notes-34670/ by Clker-Free-Vector-Images | CC0 Creative Commons

Systematic approach: A software process

A **software process** is a sequence of activities that leads to the production of a software product.

The four fundamental activities are:

- **Software specification**: customers and engineers define the software and its constraints
- **Software design and implementation**: the software is designed and programmed
- **Software validation**: the software is checked against requirements
- **Software evolution**: the software is modified

Source: https://pixabay.com/en/mark-marker-hand-write-516277/ by geralt | CC0 Creative Commons

Different types of processes

Different types of systems need different development processes.

For example:
- **Real-time** software in an aircraft needs to be **completely specified**
- For **e-commerce systems specification** and **development** may go **hand in hand**

Consequently…
"Activities may be organized in different ways and described at different levels of detail depending on the type of software being developed."

Source: https://pixabay.com/en/cockpit-plane-airplane-jet-100624/ by 12019 |CC0 Creative Commons

Software engineering disciplines

Computer science
- **Methods** that underlie computers and software systems
- **Theory**
- **Not always 100% applicable** to large, complex systems

Systems engineering
- All aspects of **development** and **evolution** of **complex systems**
- **Hardware** development
- **Process design** and system deployment
- **System specification** and integration

Source: https://pixabay.com/en/artificial-intelligence-brain-think-3382507/ by geralt |CC0 Creative Commons

General issues with software

- **Heterogeneity:** Operated as distributed systems accessed by different types of devices (e.g. mobile phones).
  Challenge: ***build flexible software***
- **Business and social change:** business and society changes and so do technologies.
  Challenge: ***traditional SE techniques are time consuming***
- **Security and trust:** Software is intertwined with all aspects of our lives (e.g. web services).
  Challenge: ***maintain information security***

Source: https://pixabay.com/en/doors-choices-choose-open-decision-1767564/ by qimono |CCO Creative Commons

Software engineering diversity

A systematic approach is taking into account
- Cost, schedule, and dependability issues

Implementations vary depending on
- The **organization** developing the software
- The **type** of software
- The **people** involved in the process

Source: https://pixabay.com/en/umbrella-color-atmosphere-mood-1587967/ by geralt |CCO Creative Commons

"There is no universal method or technique. The type of application may decide over the used method."

Types of application

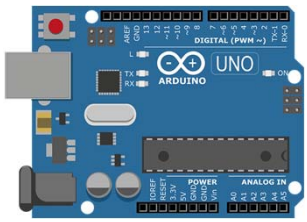
Stand-alone applications
- Run on a **local computer**
- Include all necessary functionality
- Do **not require network** access
- E.g. CAD programs, photo manipulation

Source: https://pixabay.com/en/office-windows-word-excel-1356793/ by Pixaline |CCO Creative Commons

Interactive transaction-based applications
- Execute on a **remote computer** but accessed from a user's PC or terminal
- May include **large data** that is **accessed** and **updated** at each transaction
- E-commerce systems
- **Cloud-based** business systems

Types of application

Embedded control systems
- Software that **controls hardware** devices
- Probably the **majority** of systems
- E.g. **Mobile phone** software, software in a microwave, software in cars



Source: https://pixabay.com/en/arduino-arduino-uno-technology-2168193/ by Seven_au |CCO Creative Commons

Batch processing systems
- Systems that process **large batches of data**
- Input-Output systems
- E.g. phone **billing systems**, salary payment systems

Types of application

Entertainment systems
- Focus on **personal** use
- Usually **games** of some sort
- Quality of **user interaction** is key

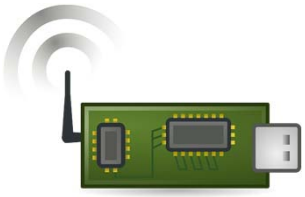
Systems for modelling simulation
- Developed by **scientists and engineers**
- Used to **model** situations
- Often **computationally intensive**
- Require high-performance parallel systems

Source: https://pixabay.com/en/super-mario-red-jumping-man-1138462/ by InspiredImages |CC0 Creative Commons

Types of application

Data collection systems
- **Collect data** from the environment using a set of **sensors**
- Often installed in **hostile environments** e.g. inside an engine or in a remote location

Systems of systems
- Composed of a **number of software systems**
- Some may be generic (e.g. a spreadsheet program) others may be **specifically written for that environment**



Source: https://pixabay.com/en/wlan-wireless-lan-card-sender-155909/ by OpenClipart-Vectors |CC0 Creative Commons

No clear boundaries

The boundaries between systems are blurred

- A game for a mobile phone has **the same constraints** as phone software
- **Batch processing** systems are often used in conjunction with **web-based systems** (e.g. phone bill)

Software engineering techniques may be different

- An embedded control system in a car can be **safety-critical** i.e. it needs extensive verification as it is expensive to recall cars after sale
- Web-based systems often use an **iterative development** approach
- Systems of systems usually need to be **specified in advance** so they can be developed separately

Source: https://pixabay.com/en/country-road-road-traffic-tar-599457/ by markusspiske | CC0 Creative Commons

Software engineering fundamentals

But some fundamentals apply to all types of software

- Use a **managed and understood development process** i.e. have a clear idea of what will be produced and when it will be produced
- Software should be **dependable and perform efficiently**. It should be safe to operate and secured against external attacks.
- **Software expectations** should be understood and managed using **software requirements** and **specification** documents
- Make **effective use of existing resources** i.e. reuse already developed components

Source: https://pixabay.com/en/bridge-web-bridge-piers-pillar-246913/ by Antranias | CC0 Creative Commons

Software engineering and the web

The web had a profound effect on software engineering

- **Initially** universally accessible **information store**
- Around **2000** the **Web started** to evolve
- More and **more functionality** was added to the browser
- **Web-based systems** started replacing special purpose interfaces
- Led to the development of **new and innovative services**
- **Web browsers improved** to run small programs
- Instead of writing software and deploying it on PCs it was **deployed on web servers**
- **Changes** and **upgrades** of software became **cheaper**
- Costs were also saved on the **user interface development**

Source: https://pixabay.com/en/browser-web-www-computer-773215/ by geralt |CC0 Creative Commons

Software engineering and the web

The next step: web services and software as a service

- **Web services** deliver specific functionality over the web
- Applications **integrate these web services** (e.g. google translate)
- Linking can be **dynamic** so that it is executed when it is called (allows for updating the service)

Source: https://pixabay.com/en/social-networking-marketing-business-2187996/ by 472301 |CC0 Creative Commons

- **Software as a service (SaaS)** brings entire programs into the “cloud” (e.g. web-based email)
- A computing cloud is a huge number of **linked computers** that are shared by many users
- Users **pay according to how much they use** the software (often it is free but financed through advertising)

MANAGEMENT CENTER INNSBRUCK

Software engineering and the web

Changes in system engineering

- **Software reuse** has become the **dominant** approach
- Systems are **not specified in advance** but rather **developed iteratively**
- **User interfaces are constrained** by the capabilities of the web browser
- Web-based application interfaces are still **more limited** than specially designed interfaces on PCs
- Progress is, however, made (e.g. **AJAX**)

Source: https://pixabay.com/en/code-html-digital-coding-web-1076533/ by jamesosborne_ | CCO Creative Commons

MANAGEMENT CENTER INNSBRUCK

Software ethics

Don't be evil!

- Software engineering is carried out **within a social and legal framework**
- The job involves **responsibilities** beyond technical skills
- This includes the **application of standards** and **morality**
- Software engineers do have **a lot of power**…

Source: https://pixabay.com/en/arrow-ethics-morality-credibility-2110610/ by 3dman_eu |CCO Creative Commons

Software ethics

Standards that may not be bound by law

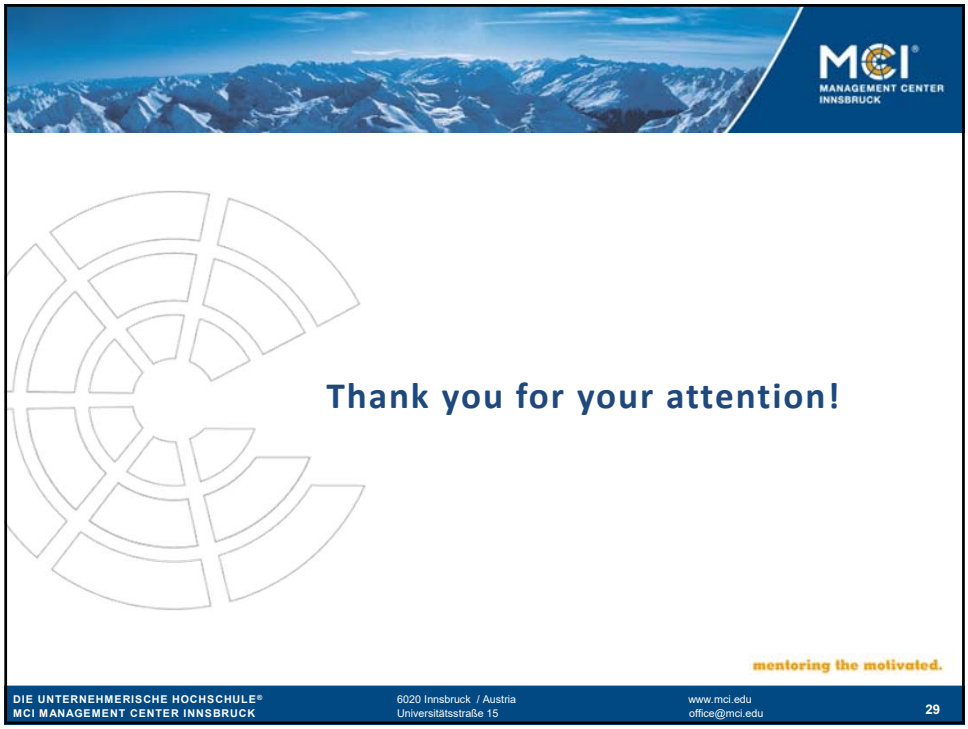
- ***Confidentiality:*** respect the **confidentiality of employees** and clients, also if no confidentiality agreement is signed
- ***Competence:*** do not accept work that is outside your competence
- ***Intellectual properties:*** be aware of **intellectual property** such as patents and copyrights
- ***Computer misuse:*** do not use technical skills to **misuse other people's computer** (e.g. game playing, dissemination of viruses, etc.)

Source: https://pixabay.com/en/shield-ethics-morality-credibility-2110613/ by 3dman_eu |CCO Creative Commons

A code of conduct or a code of ethics for software engineering is published by the **ACM** and the **IEEE**!

What you should have taken away from this class:

- Software is **not just a program or programs but also includes documentation**.
- The software process **includes all of the activities involved in software development**. The high-level activities of **specification, design and implementation, validation, and evolution are part of all software processes**.
- There are **many different types of systems** 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 systems**. These fundamentals include **managed software processes**, **software dependability and security**, **requirements engineering**, and **software reuse**.
- **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** that set out the standards of behavior expected of their members.


MCI
MANAGEMENT CENTER
INNSBRUCK
Thank you for your attention!
mentoring the motivated.
DIE UNTERNEHMERISCHE HOCHSCHULE®
MCI MANAGEMENT CENTER INNSBRUCK
6020 Innsbruck / Austria
Universitätsstraße 15
www.mci.edu
office@mci.edu
29