Universiteit Antwerpen
Faculteit Wetenschappen

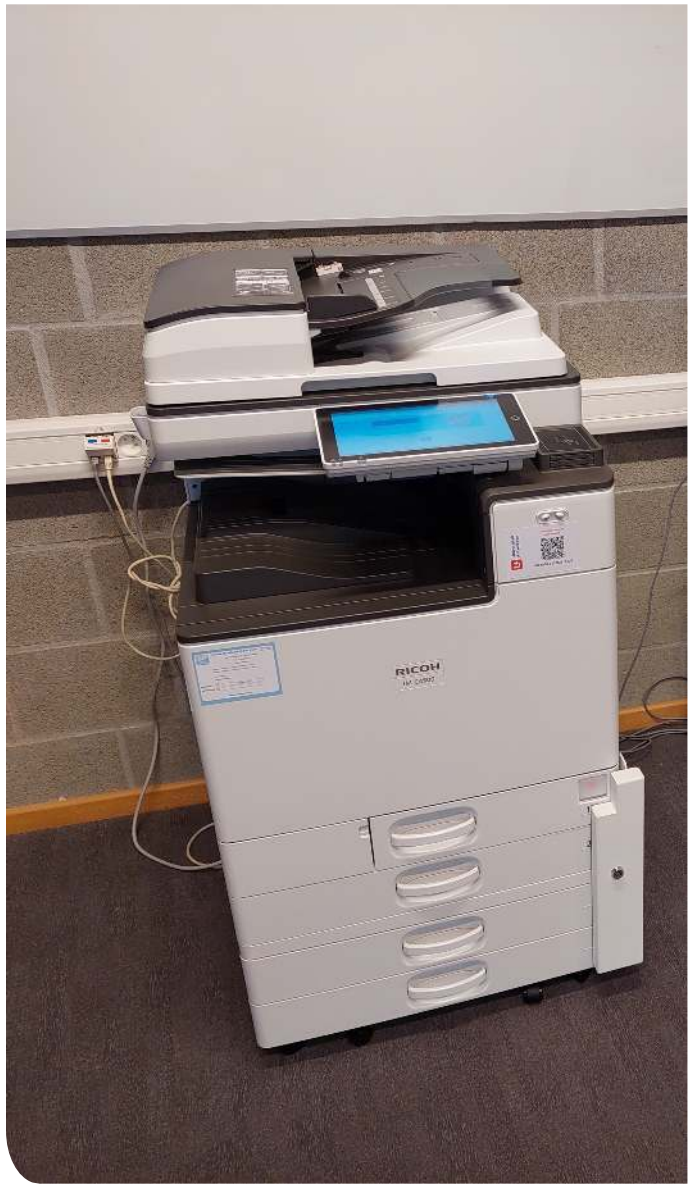
Project Software Engineering

Project 23-24: Printing system

What you will learn

- **Quality control: tests, contracts**
- **Collaboration in group (e.g., version control with GitHub)**
- **Object-oriented design**
  - Separation of concerns: split into multiple files and classes
  - Managing dependencies: Includes, contructor parameters, …
  - Decoupling
  - **=> Make your code flexible and adaptable**
- **Customer interaction:**
  - Interpreting a specification
  - Negotiating

Printing system



Software for printing system

- Printing devices
- Printing jobs
- Tracking CO2 emissions
- Functionality described in use-cases
  - Now: specification 1.0
  - Later: specification 2.0

Use-cases for specification 1.0

Use-Case	Priority
1: Input	
Use Case 1.1: Reading printers and jobs	REQUIRED
2: Output	
Use Case 2.1: Simple output	REQUIRED
3: Simulation	
Use Case 3.1: Manual processing	REQUIRED
Use Case 3.2: Automated processing	IMPORTANT

General requirements

Tools

- C++17
- Buildsystem (CMake, CLion IDE)
- Tests (Google Test)
- Contracts (DesignByContract.h)
- XML input files (TinyXML)
- Documentation (Doxygen)
- Collaboration (GitHub)

Evaluation criteria

- Example evaluation form (Ansymore -> “Project”)
- Non-functional requirements (Ansymore -> “Project”)
- Quality control: automatic tests, contracts
- Code quality: OO design, code conventions (Ansymore)
- Good project management: Documentation, planning

Example evaluation form

- **Ansymore -> “Examen”**
- **Pay close attention to this!!**

Algemeen

Samenwerking: ❑ blijven samenwerken ❑ werk wordt gelijk verdeeld
❑ beide studenten begrijpen de ingediende oplossing
Beheersing werktuigen: ❑ gtest ❑ TinyXML ❑ CMake ❑ CLion ❑ Doxygen ❑ GitHub
Commentaar:

Tests

❍ Nt aanwezig (= geen tests)	❍ Vrkrd. gebruikt (= manuele test)	❍ Beperkt (=nt. alles getest) - domein tests	❍ Voldoende (= invoer) - invoer tests - meerdere fouten	❍ Goed (= uitvoer) - uitvoer tests - scenario’s	❍ Excellent (= testcode goed) - modulair - code duplicatie

Commentaar:

Contracten

❍ Nt aanwezig (= geen assert)	❍ Vrkrd. gebruikt (= geen pre-post) - niet in .h file - private/protected	❍ Beperkt (=minimum) - properlyInit	❍ Voldoende (= alleen pre) - getters	❍ Goed (= some pre-post) - setters - output	❍ Excellent (= soms pre-post) - domeinmodel bewegingen

Commentaar:

Plan

❍ Nt aanwezig (= geen planning ingediend)	❍ Vrkrd. gebruikt (= te laat)	❍ Beperkt (=nt. alles)	❍ Voldoende (= alle verplicht)	❍ Goed (= meer) - nt. verplichte - zoals beloofd	❍ Excellent (= bijna alles) - graf. impressie - graphics engine

Commentaar:

Objectgericht Ontwerp en Programmeervaardigheden

❑ Basis ontbreekt (= beoordeling niet mogelijk)	❑ Basis - code compileert - 0, 1, infinity principe - efficiënt gebruik van datastructuren - code opgesplitst in overzichtelijke files - gebruik van pointers	❑ Encapsulatie - private variabelen + getters en setters - functie/ctor parameters beperkt - afschermen van interne datastructuren
❑ Domeinmodel - klassen modelleren het domein - klassenamen = substantieven - Gedeeltelijk aanwezig: 1-op-1 overeenkomst tussen klasses en verantwoordelijkheden	❑ ADT - functies = acties dat het object onderneemt - functienamen = werkwoorden - functionaliteit bij data (cohesion) - correct gebruik inheritance	❑ Modulariteit (= separation of concerns) - loose coupling - dependency injection - input/output is duidelijk afgescheiden (strings aangemaakt in aparte output klasses) + ook in de testcode

Commentaar:

Planning

Planning

- **Specification 1.0**
  - Keep track of time (“tijdsblad”)
  - Demonstration of system 1.0 during evaluation 1.0
- **Specificatie 2.0**
  - Promise use-cases in plan 2.0
  - Demonstration of system 2.0 during evaluation 2.0
  - Extra time? => Extra functionalities in plan 2.1
  - Demonstration of system 2.1 during FINAL evaluation 2.1

Planning

- **Reference: Ansymore website -> “Tijdsschema”**
- **Keep an eye on updates!**

Date	Deadline
Monday, March 18	System 1.0
Thursday, March 28	Plan 2.0
Monday, April 29	System 2.0
Wednesday, May 8	Plan 2.1
???	Exam: System 2.1