CONTENTS

• Practical Matters

• Course overview – topics, text book

• Project

• Achievement-driven approach to learning

• Analysis and Design primer

• Forming groups
ARE YOU REGISTERED?

- Admitted students must web-register themselves for this course in the Student Portal.

- Students who are unable to web-register (or who were registered before) should contact the Student Office (it-kansli@it.uu.se).

- Do this as soon as possible! You will be able to web-register until November 11 at the latest.
DROPPING COURSES

• If you decide to drop it, you must inform the Student Office (it-kansli@it.uu.se).

• If less than 3 weeks have passed since the course started, your course registration will be removed.

• After 3 weeks a “course intermission” will be reported to UPPDOK instead.
COURSE OVERVIEW
TOPICS COVERED

• Requirements Analysis

• Structural Modelling – classes, components, subsystems, deployment.

• Software Architecture

• Behavioural modelling

• Design Patterns

• Design principles (GRASP)

• Design evaluation and improvement
TEXTBOOK AND REFERENCES
PROJECT

• Analysis, Design and implementation of DungeonQuest board game (with various extensions)

• Groups of 4 or 5.

• Follow iterative design process:
  • elicit requirements, develop use cases, perform design, write code of small parts of whole
  • each iteration develops more, and improves design based on experience gained

• No expectation that you will complete it
MEETINGS WITH TAS

• Weekly meetings with TAs:
  • discuss issues that arise in lectures
  • discuss project, obtain feedback
  • be assessed on various points
DELIVERABLES

• Check-pointing:
  • important to see progress/improvements to ongoing project
• Final design
• Review of other's design
PROJECT

• Volume produced is not what will be measured.

• Rather, understanding and application of design processes and design principles, response to feedback, improvement of design through iterations, evaluation of others’ design.

• Assessment based on achievement-driven learning
ACHIEVEMENT-DRIVEN LEARNING
ACHIEVEMENT-DRIVEN LEARNING

• A new assessment approach based on a fine-grained breakdown of learning outcomes.

• Detailed list of learning outcomes (achievements) known in advance.

• Students need to work towards ticking-off achievements.

• Different achievements required for 3, 4, and 5 – higher grades require deeper levels of mastery, not more stuff.

• Idea of achievement-driven learn was developed by Tobias Wrigstad.
THE PROCESS

• You need to have a plan
• You decide how to convince us that you have *unlocked an achievement*
• TA will determine whether we agree – usually during weekly meeting.
• Achievements are *individual*
  • when discussing with a TA need all members of team to participate
  • missing meeting may mean missing/delayed opportunity
• Some time scheduled later in course to tick-off achievements.
EXPECTATIONS

• Don’t leave everything until the end
• Do research to find out how to do things – teach your team!
• Keep your group alive and active – or quickly get a new one
• Do not try to do too much (or too little)
• Structure your work towards ticking-off achievements
EMAIL ADDRESS

• If your email is broken –mailbox full, etc–, you will miss crucial information.

• Failing to exist is not an excuse for failure.
COMMENTS, CONSTRUCTIVE CRITICISM, SUGGESTED IMPROVEMENTS

Please come to me first.
COURSE WEB PAGE
HTTP://WWW.IT.UU.SE/EDU/COURSE/HOMEPAGE/ASD/HT13
ANALYSIS AND DESIGN PRIMER
WHAT IS GOOD DESIGN?
GOOD DESIGN:
DESIGN THAT COPES WITH CHANGE.
EMBRACE CHANGE
KEY: ASSIGN RESPONSIBILITY TO CLASSES
NOT RELIGIOUS
ABOUT DESIGN

Design choices need motivation:
• Use cases.
• Expected future use cases.
• Non-functional properties.
• Organisational constraints.
SOFTWARE DEVELOPMENT PROCESSES

• Ad hoc coding

• Waterfall model

• Iterative development method, agile methods
MODELLING AND UML
UML IS ABOUT COMMUNICATION NOT DOCUMENTATION
MODELLING STARBUCKS

• As an example, we’ll model some of the activities that take place at Starbucks (or your favourite American coffee chain).

• We’ll model the activity of the customer ordering a cup of coffee.
The customer orders coffee at the cashier and pays. The cashier writes the order on a cup and places it in the queue for the barista. The barista retrieves the cup from the queue, makes the coffee, and places it in another queue for the customer to retrieve. When the customer’s coffee becomes available, the customer takes it and leaves.
FULLY DRESSED USE CASE

Use case: Process Sale

**Scope:** Starbucks

**Level:** user goal

**Primary actor:** cashier

**Stakeholders and interests:**
- Cashier
- Customer
- Barista

**Preconditions:** Cashier is wearing Starbucks uniform

**Success Guarantee (or postcondition):** Customer receives cup of coffee.

**Main Success Scenario (or basic flow):**
1. Customer arrives at the cashier.
2. Cashier asks customer for order.
3. Customer provides order.

... 

**Extensions (or alternative flows):**
...
DOMAIN MODELLING

- **Customer**
  - processes
  - takes order
  - removesFrom

- **Cashier**
  - writes
  - addsTo

- **Coffee**
  - kind

- **InputQueue**
  - size
  - reads
  - removesFrom

- **OutputQueue**
  - size
  - addsTo

- **Barista**
  - processes
  - addsTo
STATIC MODELLING

Customer
put(Cup) take() : Cup

Queue <<Cup>>
put(Cup) take() : Cup

Barista
inqueue : Queue<<Cup>> outqueue : Queue<<Cup>>
brew(Cup)

Cup
customer : Customer
setCustomer()
getCustomer() : Customer

Cashier
order(Customer)

Customer

 Wednesday 30 October 13
DYNAMIC MODELLING

Dynamics of a coffee shop system:

- Customer orders a cup.
- Cup is created.
- Cup is put into the queue.
- Barista takes the cup.
- Cup is taken from the queue.
- Cup is put into the output queue.

**Activities:**

- `order(c)`
- `cup = create()`
- `put(cup)`
- `cup = take()`
- `put(cup)`
DYNAMIC MODELLING

Barista

**Start**

- **Idle** → **Busy**
  - newCup

- **Busy** → **Idle**
  - finished / filledCup
TEAMWORK
TEAMWORK

- Assigning people to roles
- Swapping roles
- Peer learning: splitting up learning of material, and teaching remainder of group
FORM GROUPS

• Groups of 4 or 5.

• Ensure that you have at least one hour time slot that you can meet with a TA per week.

• When you have your group, sign up with a TA (provisional).

• Leave room when you are signed up.

• Do not leave room without joining a group.