# CS 414 Object Oriented Design
# Fall 2017
# Project 3.0: The fun part begins!

Due date: 10/23/2017

## 1 Introduction

Having analyzed the requirements, defined the use cases, and modeled the system domain, you are ready to go ahead with the design and implementation of the *X game*.

*Read carefully this document before proceeding!*

## 2 Design and implementation

As a development team and based on the artifacts developed in previous iterations, you are to design and implement the first version of the *X game*.

With this purpose, consider the next tasks:

- Prepare sequence diagrams as part of the design process.
- Prepare a design class diagram.
- Make a list of system test cases.
- Write JUnit test cases.
- Implement the first version of the system. You must use the GRASP patterns (to be) reviewed in class. Read chapters 17 and 18 of Larman's text to have a better understanding of the patters.

Mind that existing artifacts (i.e., use cases, domain model and glossary) might need to be refined/updated during the design and implementation of each version of the *X game*. When updating these artifacts, highlight the performed changes. You should strive for consistency among all the development artifacts. Although previous documents will not be re-graded, consistency among the latest versions ones will.

The tasks above are to be executed **only for the set of core features**.

## 3 Deliverables

There are six deliverables for this assignment:

1. **Design document**. The document should contain:
    a. At least three key/difficult/interesting sequence diagrams.
    b. The design class diagram.
2. **Testing document**. The document should contain the system test cases.
3. **Source code**. The code should include the JUnit test cases. Well-documented code will be rewarded.
4. **Development manual**. The document should describe how to set up the development environment to work on the project, how to run the system as a developer, and how to run the tests (put yourself in the place of a newcomer—what are the necessary steps for her to start working on the project?).
5. **Traceability link matrix**. This is a spreadsheet document showing the traceability links between the use cases and the code. While each row represents a use case, each column represents an implemented class in the system. A mark in the cell intersecting the use

case *UC-1* and the class *MyClassB* indicates that the class *MyClassB* is directly involved in the implementation of the use case *UC-1*, as shown next:

| | MyClassA | MyClassB | … | MyClassZ |
|---|---|---|---|---|
| UC-1 | X | X | | |
| UC-2 | | X | | X |
| … | | | | |
| UC-n | X | | | X |

6. **Presentation**. The progress on the project is to be presented during class. Besides presenting the design artifacts, you are to present any update to the previous artifacts. You are also to present the technologies and tools you are using to develop the project, as well as a demo of the *X game*.

All documents should be uploaded to the GitHub repository. Mind that the commit log of the repository will help us determine the level of individual participation in the project, time management, and how the version control system is being used. This applies to both source code and documentation.

The grades for this assignment will apply to the deliverables uploaded to the GitHub repository before class (i.e., 3:00pm).

# 4 Notes

- Deadlines associated with deliverables will be verified in the repository. Late work policies apply.
- Grading criteria:
  - Design document: 20%
  - Testing document: 10%
  - Source code: 20%
  - Development manual: 10%
  - Traceability link matrix: 10%
  - Presentation: 30%
- Points will be deducted if:
  - The submission requirements are not met.
  - You are late with the submission.
- You will not receive credit for this assignment if:
  - You do not submit the deliverables.
  - You do not present during class.