CS4125 SYSTEMS ANALYSIS AND DESIGN

TEAM-BASED PROJECT

Specification V1

Semester II: 2014-2015

J.J. Collins

19th February 2015 (Week 4)

1. Objectives

1. To apply an Object-Oriented (OO) Analysis and Design (OOAD) method using the Unified Modelling Language (UML) to the specification of an enterprise software system.
2. To apply architectural and design patterns where appropriate.
3. Gain insights into team dynamics and project management from the experience.

2. Scenario

You are required to create a situation in which you are commissioned by a client to either automate an existing manual system, or adapt an existing system to incorporate changes in the business model. A list of examples is as follows:

- Traffic flow simulation.
- Games hosting framework
- Football fantasy league.
- Warehouse distribution or retail.
- Accounts processing for a new mobile phone company.

The system should have a significant algorithmic requirement as opposed to being one that primarily post queries and updates to a database from a GUI.

3. Specification

You are required to submit a design document, which contains in the following order:

1. Front cover with business scenario title, student names and IDs, and module details. This page should convey a professional impression, for example, by designing a logo that reflects the concept being developed..
2. Table of contents.

3. A one-page (maximum) narrative description of your development project.
4. A one-page (maximum) discussion on the software lifecycle model adopted to manage the process of software development.
5. Project plan and allocation of roles.
6. Requirements
    - Functional:
        - One or more use case diagram(s). Do not use decomposition!
        - For each use case, a structured use case description.
        - For at least one use, the use case description to be specified using the template distributed in lectures.
    - Non-functional requirements specified as add-ons in the use case description.
    - A short discussion on tactics that one might consider adopting to support specific quality attributes.
    - A brief selection of:
        - Screen shots of GUI prototypes, especially for web based front ends.
        - Layouts/format for both electronic and hardcopy reports.
7. A two-page (maximum) discussion of system architecture to include at least one package diagram, and detail the architectural decisions taken at this point in the project.
8. Analysis supporting documentation drawn using the generic OOAD method presented in lectures. Diagrams should be hand drawn **sketches**: Include the following:
    - List of candidate objects derived by a0pplying Data Driven Design (DDD) to use case descriptions.
    - Sketch of the class diagram, with significant methods and attributes identified. The sketch should demonstrate inheritance, aggregation and composition, associations, dependencies, visibility, etc. There should be no duplication of attributes or methods in the diagram. The diagram should contain at least two class interfaces.
    - A sketch of a communication diagram or sequence diagram.
    - Entity Relationship diagram with cardinality.
9. Code
    - Proof of concept prototype.
    - Implement those classes necessary to support a subset of the use cases.

- Should include Model-View-Controller (MVC) architectural pattern, and collection classes.
- Automated Test Cases as an acknowledgement of test Driven Development (TDD).
- **ADDED VALUE**: application of an architectural and/or design pattern - MVC, iterator, and singleton excluded, AND/OR support for concurrency through threading AND/OR GUI.
- No need to implement data persistence layer, simulate it.

Note: coding can only begin after completion of sections 6, 7, and 8.

10. Two page discussion of added value.
11. Design **blueprints** - based on the implementation but can also incorporate concepts that could be developed in future releases. Draw blueprints of:
    - Architectural diagram
    - Class diagram.
    - Interaction diagram
    - One state chart for an object in either the sequence or communication diagram, with annotated transition strings.
    - Descriptions of the patterns used in item (9), and support for concurrency.
12. Critique: compare and contrast the analysis sketches in (8) versus the blueprints created in (11).
14. References

4. Notes and Guidelines

- This assignment **constitutes approximately 50%** of the total marks awarded for this module.
- You will work in a team of 3 or 4. A team of 4 is expected to submit a higher quality submission than a comparable team of three.
- The award of an F or NG will automatically result in the award of an F or NG for the module.
- **Submission deadline is 11:00 Wednesday 22nd April 2015 - Week 12.**
- NO SUBMISSIONS WILL BE ACCEPTED AFTER THIS DATE!

- Hand the submission to the lecturer at the start of the scheduled lecture on Wednesday 22nd April 2015 (Week 12).
- **Prior to commencement of coding, email sections 1-8 of the report to J.J.Collins@ul.ie with cover stating identity of team members.**
- You will be required to provide the lecturer with a walk through of your project submission in Week 13. The timeline for this will be agreed with the class rep.
- The project will not be marked if a walkthrough is not provided.
- Submission should be stapled or glue-bound with a clear front cover.
- You are required to submit an electronic copy of your submission at the beginning of the walkthrough.
- All team members must contribute to the implementation.
- Programming language and development environment is at your discretion.
- Presentation is critical. The report should be concise, easy to read, with diagrams that conform to aesthetic guidelines, i.e. minimise line crossings.
- If the presented work is not to the standard expected at this level, the submitted work will be severely penalised. Please ensure that:
    - Layout conforms to specification.
    - Page numbers in table of contents and throughout report.
    - Spell and grammar checks done.
- Analysis diagrams can be drawn on paper, captured using a mobile phone camera, and inserted in jpeg/gif/etc form into the report.
- You must use a diagramming workbench for item (11) of the specification - numerous lists of community edition tools are available on the net, select one and justify your choice in the report. Using Word or PowerPoint is definitely excluded.
- You may base your project on an existing system, but you must:
    - Demonstrate added value – for example, by extending the feature set
    - Demonstrate an in-depth understanding of structure and dynamics.
    - Adhere to the rules with respect to the prevention of plagiarism.
    - Clearly differentiate between what already exists and added value.
- Quality not quantity. Depth not breadth. Do not attempt to develop a full enterprise system! Instead, focus on a more constrained and feasible objective. For example, an accounts subsystem that interfaces with other subsystems in the enterprise.

- Marks are awarded for having attempted to use the methodologies and tools, for developing a lightweight proof of concept prototype, and evaluating the utility of diagrams. Marks are not awarded for the complexity or expansiveness of your selected scenario.
- Separation of the business tier and UI is critical. Make sure that you have read and adopted one of the standard approaches to the design of user interface objects such as the Model View Controller (MVC) approach.
- The lecturer reserves the right to make minor amendments to the assignment up to and including week 8. Amendments will be sympathetic to any work done to date.
- You should backup your work at all times. Accidental loss of work will not be accepted as an excuse.
- Plagiarism will automatically result in an F grade for the module and referral to the disciplinary team.

5. Note on Team Dynamics

- Problems with team dynamics should be reported to me in person A.S.A.P.
- If it becomes evident during the walkthrough that one of the team members did not contribute equally to the project, the following penalties will be imposed:
    - F for the non-contributing member
    - Capped at a C3 for the other member because he/she did not bring the problem to the lecturer's attention.
- Each team member must contribute equally to the effort required for the project. This is particularly applicable to the coding element of the specification.
- It is not your responsibility to deal with a team member who refuses to contribute fairly, that is the lecturer's job. However, it is your responsibility to report problems as soon as possible, which will be handled in a sensitive manner.
- The primary mechanism to resolve problems with team dynamics will be to resort to individual submissions.

Division of labour amongst team members is essential to a successful outcome, as is the necessity to commence immediately. Work clever, and do not focus excessively on ensuring fidelity of problem with real world scenarios.