# MSc in Communications Software 2011
Design Patterns
Summer Paper
Eamonn de Leastar

***Instructions:***

Below are are four short descriptions of problems in specific contexts. Two solutions are required for each problem - the second an elaboration on the first. Each solution is to be structured as follows:

I. Briefly outline of the relevant design pattern(s) that are viable candidates in the ensuing answer.

5 Marks

II. A solution to Version 1 expressed as a class diagram and with a high level sketch of the classes involved. Methods in the class can be expressed in any suitable pseudocode.

10 Marks

III. A Solution to Version 2, which may be expressed as modifications to Version 1 with further classes and pseudocode if necessary.

10 Marks

IV. A short summary of the benefits of adopting the selected patterns.

8 Marks

Select *any three* of the problems and propose the outline solutions structured as described above. All problems are awarded equal marks.

***Problem 1: Micro Blogging***

A new micro blogging service - to be called MessageFeed - is to be designed, with a particular emphasis on a simplified API for developers. This API is to provide a mechanism for account signup and log in, message “posting” to specific message identifiers (modeled on twitter hash tags). One version of this API is to be exposed as a set of Java interfaces, which should provide the expected functionality.

- Version 1: Propose a flexible API to encapsulate the MessageFeed service. The API is to be expressed as a set of Interfaces, which can be implemented incrementally as the project advances.

- Version 2 : A subscription feature is to be introduced, where by an application can subscribe to message identifiers. Propose an extension to the API to deliver this feature.

## *Problem 2: GUI Tree Component*

A tree component is being designed as part of a new GUI component library, enabling hierarchical data to be displayed. The component is so support a simplified model/view/controller approach. Thus, the data is to be provided to the tree component in the form on a data model object, which the component will access when the tree is to be redrawn.

- Version 1: provide a simple example of how this might be realised.

- Version 2: in this version, illustrate how data held in pre-designed data structure (not related to the table model) can be rendered by the component.

## *Problem 3: Trouble Tickets*

A development team is implementing a desktop application for managing and scheduling for trouble tickets within a technical support application. Its features include the ability to create/delete tickets, group related tickets, reorder previously grouped tickets, and submission of ticket groups to technicians for execution. Technicians then will be tasked either with individual tickets, or groups of tickets.

- Version 1:  Propose a simple design for such a system.

- Version 2 : A new feature is required, whereby ticket groups can be nested. i.e. a group of related tickets can be embedded within another group, all allocated to one technician. Devise an extension to the existing design to support this feature.

## *Problem 4. A, B, C & D.*

A developer is present with the following problem. Class A has been written to operate with Class B, with a reference to B provided to A's constructor. The source for Class A is "closed", i.e. it cannot be modified. However, the source for class B is available. Note also that there are classes C and D in the problem space, which are unrelated to B.

- Version 1: Propose a design pattern that would enable class A to interoperate with class C or D. Use implementation inheritance in the proposal.

- Version 2:  Propose an alternative configuration whereby B is an interface. What variation of the design pattern proposed in 1 can now be adopted. Explore the advantages of this approach.