UNIVERSITY OF BIRMINGHAM

Object Oriented Programming Using C#

Assignment 2010

Dr M. Spann

1. Aims and Objectives

This C# programming assignment involves the design and implementation of the software which enables the operation of a lift to be modeled, simulated and the simulation graphically animated. It specifically involves the development of a multi-threaded and event-based software architecture.

2. Lift Controller Model

A lift controller comprises a number of floor selection buttons that are activated by the users of the lift along with up/down selection buttons situated on each floor. It contains a floor sensor that is activated when the lift reaches a given floor and the number of the floor can be determined by lift control software accessing the sensor. The lift controller has a direction sensor and display panel that indicates the direction of motion of the lift (up or down) or if the lift is stationary, neither direction sensor is illuminated and it also contains a floor indicator panel which specifies the current floor visited by the lift.

The lift services user requests by visiting the next floor required either by an on-board user or a user waiting at a floor to be picked up in the current direction of travel and according to a direction of travel requested by the user. Thus if a user at the 5th floor selects the 'up' floor button but the lift is currently traveling down to pick up a user at the 3rd floor, it will not stop at the 5th floor to pick up the user even if it passes the 5th floor. The lift will only alter its current direction of travel if there are no further request to stop at further floors in that direction. This is the normal mode of operation of a passenger lift. A service lift for example might go straight to its target floor ignoring all service requests at intermediate floors.

3. Design Overview

There are a number of system components making up the overall application. This is a demanding programming exercise and you should design your system in separate stages with each stage undergoing thorough testing. Note, it is expected that you include details of all testing in your report. You will need to implement the following independent units of software.

- Lift control system. This models the interaction between the sensors and the lift's motor controller and moves the lift either up or down in order to service user requests based on the sensor states at each of the floors and the current direction of travel.

- Graphical animation. This animates the lift's position, motion and sensor states. Figure 1 is a screenshot of a suggested layout including the lift and lifts sensor indicators on the floor and lift buttons and an animation control button.

- External event generator. This generates lift and floor button presses according to some simple random event process. I would suggest that you use a simple random process where lift service requests are generated with a specified average time between them. We could model people entering and leaving the lift as a separate random process but, since this has nothing to do with the actual lift control mechanism, we won't.

4. Programming Hints and Suggestions

Implement the event generation process and the control of the lift as independent threads of execution. The events generated by lift and floor buttons imply state changes of the corresponding sensors. These states are appropriately read and handled by the lift controller. The lift controller moves the lift either up or down depending on the state of the lift and floor button sensors. The lift controller software must be regarded as an independent unit of software (it would in practice be an embedded system implemented on a microcontroller) as it has no knowledge about the event generation or animation software. It could, for example, be re-programmed to implement different 'rules' defining in what order it visits floors with no changes required to the rest of the application.

Because the lift controller software has no knowledge of the animation software, sensor state changes need to be issued as events which the animation application registers its interest in and contains event handling code to update itself in accordance with the controller sensor states. Also updating the animation GUI from multiple threads is not a good idea as GUI components are not usually thread safe. Use *Invoke* to do your GUI updating.

5. Assessment

This coursework represents all of the assessment for this module. The assessment will be based on a submitted formal report as well as my assessment of your program's functionality. Please submit your program **written using VS2008 (.NET framework 3)** on CD to accompany your report. Please include all of the solution files under a single solution directory Make sure your CD has your name/ID on it in case it gets separated from your report. I randomly check submitted code using anti-plagiarism software (see below). **Your program must run on the School's networked Visual Studio 2008.** Even if you develop your application in Visual Express, it is your responsibility to check compatibility with the fully installed Visual Studio and please be aware we have had compatibility issues in the past.

The assessment form that I use is in appendix V so this should give you an idea of the criteria I will use in marking your report. You should be aiming for a report length of around 15 to 20 pages excluding appendices. I am happy for you to include your code listing in an appendix but it is not obligatory. I do not expect you to use formal design notation (such as UML) but you can if you are familiar with it. Use pseudo-code to explain algorithm implementation (and not flow charts!) **and do not include explicit code snippets in your main report**.

Finally, I am sure you are aware there is a lot of published code on the internet for just about every application imaginable**. If you are going to use downloaded code for any part of this exercise, make sure you attribute it in your report (referencing the URL is sufficient)**. Obviously your mark will reflect the amount of original code in your program but you will not be penalised for using small amounts of attributed downloaded code. **If you use code from the internet (or code from a colleague) without an adequate reference in the source text, this will count as plagiarism. Any significant plagiarism will result in a zero mark for the exercise. Also, if you submit the same or similar code to a colleague, you will both receive a zero mark.**

Key dates
Report deadline: **Monday 6th December**. Please hand in to the Postgraduate Office by 12 noon.

Appendix I

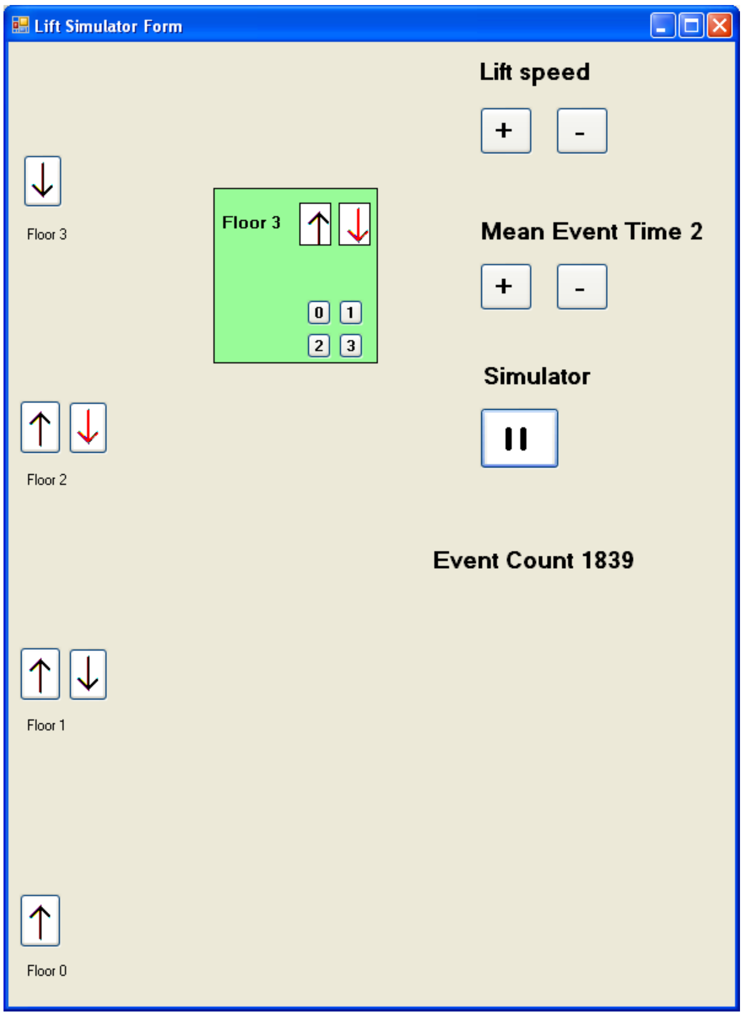

Figure 1. Screenshot of the lift animation

Appendix II

Object-Oriented Programming and Design

Programming Assessment

Dr M Spann

<u>Report Presentation</u> Cover page Page numbering Grammar and spelling Section layout Figure labelling and clarity Correct use of references		**/10**
<u>Program Design</u> Effective use of classes and object interactions Discussion of object oriented issues related to design Effective use of clear formal or semi formal design diagrams		**/20**
<u>Program Implementation</u> Code layout including use of comments Effective use of dll's Algorithm efficiency and correctness Effective use of multithreading and event handling		**/20**
<u>Program Functionality</u> No, limited, full or extended functionality Clarity and usability of the graphical animation		**/30**
<u>Testing</u> Use of systematic approach to sub system and full system testing Use of suitable output to verify test results such as screen shots		**/10**
<u>Conclusions</u> Discussion of possible design and implementation improvements and extensions Discussion of how well the program meets the specification and, if not, why not Overall summing up of what has been achieved and what has been learnt		**/10**
<u>Total Mark</u>		**/100**