

Systems Development: Object Oriented Analysis and Design

(H172 35)

Outcomes 2 and 3

“To produce static and dynamic models of a system”

An open-book assessment

You are required to provide the analysis and design documentation in preparation for an Object Oriented Programming specific implementation based on the Client Brief provided (provided as a separate document)

Your analysis and design documentation will span both the “Business” and “User Interface” models integrated within a modern agile approach to software development

You will subsequently implement your design as the requirement for Outcomes 1, 2 and 3 of the

“Software Development: Object Oriented Programming” (H171 35) unit

These outcomes can be carried out under supervised and unsupervised conditions

i.e. you may work on it in your own time

Your assessor will check the authenticity of any unsupervised work and will NOT accept anything other than what they consider to be a student’s own submission

Read the following brief and then complete the stages detailed overleaf

The Project Brief

You are a software engineer, employed by the software bureau “OOP Solutions Ltd.”

Retro Games Ltd. has commissioned you to design and develop a simplified version of Sokoban

Sokoban is a popular single player game by which a player pushes crates around a map to get them all in the right location

The game is played on a 2 dimensional grid, but the rooms are not usually of regular shape

The edges of the room are indicated by a wall, and the player and boxes cannot get through the wall

There is a warehouse keeper, who the player must control in order to move the crates from their starting positions onto the diamonds

The diamonds are the end points for the crates

You can only push a crate when you are to one side of it and its opposite side is clear, which makes the task somewhat tricky for more complicated maps

There are walls all around the map, and also in the middle in various configurations

Crates cannot be pushed through walls

Once a crate is up against a wall you can only push it along the wall, as you need to get behind a crate in order to push it

Once a crate is in a corner it is impossible to move it again

The warehouse keeper is unable to climb over crates, and is only strong enough to move one crate at a time

Crates can only be pushed, not pulled

The game will require at least five levels

Each level is to be harder to solve than the previous one, either by having more crates or obstacles, or tighter corridors, or a more complex starting arrangement of crates

The program is to record how many moves a player takes to solve a level, and output this information visually

The Analysis stage

You are required to analyse the system requirements from the client’s brief

As a preliminary you will also be required to investigate the Sokoban game itself - it’s environment and how it is played

Download an open source version of the game and try it out for yourself

To successfully complete this stage will be required to provide the following:

* Documentation associated with your analysis activities and discussions, based on the client’s brief, to agree upon a minimum scope and specification for the proposed implementation (a Requirements Specification)

Your analysis could be carried out using various techniques, such as Natural Language Analysis, interviewing the client, research into similar existing systems, or any other suitable approaches

* Associated UML analysis documentation: Use Case diagram and the use of CRC cards
* Four or more Use Case scenarios which include pre- and post- conditions, trigger events and the best case scenario flow of events

Alternative or exceptional behaviour must be included in at least one Use Case scenario (this could be represented by the use of an Activity diagram)

The Detailed Design stage

This stage will result in the provision of the following deliverables:

* A Class diagram that includes:
  + - visibility of attributes and operations (public, private or protected)
    - specification of appropriate association, aggregation and inheritance relationships between classes
* At least one Sequence diagram showing the flow of messages between three or more objects, for one use case
* At least one Activity diagram
* A visualisation (part storyboard) of at least one User Interface, it’s associated algorithm describing the required user interaction with the interface and it’s binding with the core application / resulting object diagram

All diagrams provided must be appropriate for the Client brief scenario given

Deadline date: 8th March 2017