Theory ISW

Software Engineering

Computer Science School

DSIC – UPV

Year 2020-2021

Act 1 (online)

Theory Evaluation

12-11-2020

2h 30 min

Rules:

- This evaluation will be recorded to verify the identity of the student and to guarantee the evaluation rights and review of the test.
- The participating student is responsible of guaranteeing that no other people will be present
- The Universitat Politècnica de València is not responsible for the accidental recording of images of the private sphere. The student must take preventive actions to avoid this situation.

Goal:

- Development of an evaluation exam.
- Public service provision in high education institution (article 1 of University Organic Law).
- Responsible: Universitat Politècnica de València.
- You may exercise your rights of access, rectification, removal, portability, limitation or position to the treatment of images from a UPV email account by sending an email to dpd@upv.es. Do not forget to include any necessary justificative documentation.

Act 1

Theory ISW

Theory Exam

12-11-2020

2h 30 min

Rules:

- As a participating student in this test I accept individually that I will not receive or give any help to other students. I guarantee that I am the only author of the solutions provided. If I violate this honor clause the UPV norms related to academic honor violations could be applied
- The meeting will be video recorded
- The video camera and microphone will be always connected.
- The desktop will be always shared on Teams.
- No questions will be answered during the exam. Clarify all your assumptions in written form in your answers.
- LucidChart will be used to create the class diagram and Visual Studio for the design problem
- Exam Hand over:
    - A pdf file will be submitted by email to fjaen@upv.es and let the lecturer know using the meeting chat.
    - The file will include the student's name, the answers to the theoretical questions, the image of the class diagram made with LucidChart and the C# code of the design problem.

Software Engineering

Computer Science School

DSIC – UPV

Year 2020-2021

NAME:
GROUP:

Time: 2 hours 30 min

Questions *(3 points)*

1. (*1 point*) Associate the following 'problems' with OO modelling concepts (UML notation). A given problem can be associated with several concepts and vice versa. Not every modelling concept has to be related to a problem mandatorily.

Problem		**OO Modelling concept**
a) A vehicle must have an environmental qualifier (0, Eco, A, B, C, ...)		1) Attribute in a class
b) A car is a vehicle		2) Association relationship
c) A vehicle is owned by a user		3) Aggregation relationship
d) When a customer is registered in an activity the date of inscription must be indicated		4) Cardinality 1..n
e) A vehicle must have at least a registered owner		5) Specialization/Generalization
		6) Association class
		7) Cardinality 0..1

2. (*1 point*) Explain at least two differences between using a closed and an open multi-layered architecture.

3. *(1 point)* Explain briefly the implications caused by software being a logical element.

Problems *(7 points)*

4. *(3.5 points)* Using the following description, build the UML class diagram for the proposed system including class attributes and the names of all the relationships (**do not include any methods nor attribute types**).

The company ISWSoft needs a software system for small and medium sized companies having customers with mobility restrictions. The app to be developed (*DeliveryApp*) manages the delivery of products (packages) to customers who make orders by using different mechanisms (phone, WhatsApp, Web, etc). In this first version of the product *DeliveryApp* will not handle payments because these will be handled by other means.

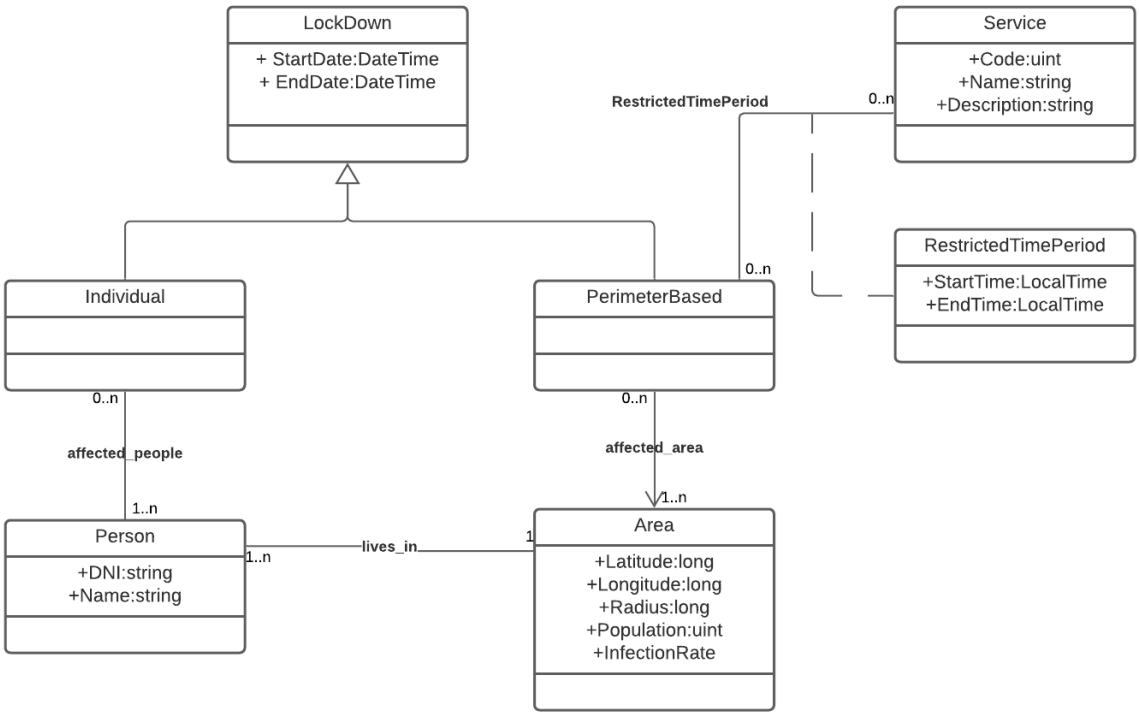
Any company can be registered in *DeliveryApp* by indicating its name, address, Company Id, and its activity. The possible activities of companies that can be selected when registering it are indicated by *DeliveryApp* such as "grocery store", "cafeteria", "restaurant", etc. Customers are registered by companies as soon as they use the system by using an ID number (if a customer already exists it is not registered twice). In addition to the ID number, it is also provided the customer`s name, phone number, address and postal code.

Whenever a delivery must be done, the company inserts the delivery in the system indicating whether it is an urgent delivery or not. If it is an urgent delivery the company must indicate the deadline delivery time (in the same day) to deliver it and the extra cost that the company will pay to the deliverer in addition to the normal pre-agreed delivery cost. In the case of a non-urgent delivery, a delivery time span is assigned (this time span has been previously agreed with the customer). All deliveries are labelled as small, medium size or large depending on its weight and it is also indicated whether the delivery is fragile or not. A description of the delivery may also be included.

Deliverers are also registered in the system by providing an ID number, alias, phone number and delivery zones (expressed as postal codes) in which they provide delivery services. In case the deliverer has its own vehicle (motorbike, car, van) he/she will indicate the vehicle's plate number and the maximum weight that can be delivered. In this case more than one delivery zone can be registered. Deliverers without a vehicle may only deliver small deliveries and in just one delivery zone (postal code). In addition, every deliverer must have a substitute deliverer who will oversee the deliveries in case any unexpected event occurs to the initially assigned deliverer. A substitute deliverer may only be substituting one deliverer.

DeliveryApp implements an algorithm to assign deliveries to deliverers depending on the delivery zone(s) of each deliverer and the deliverer's availability when the delivery is created. The availability is indicated by a deliverer in real time. If a deliverer is available, the system can assign him/her deliveries. The system must have information about all deliveries assigned to each deliverer (but not picked up by the deliverer yet), picked-up deliveries for each deliverer but not delivered, the current delivery being delivered by each deliverer and, finally, the delivered deliveries of each deliverer. As soon as a delivery is delivered the system will record the date and time of delivery. Every company will be able to display the status of its deliveries and the assigned deliverer. The administrator of *DeliveryApp* will issue the payments to deliverers and the charges to companies every 15 days

5. *(3.5 points)* Given the following UML class diagram:



Note1. There is a navigation restriction between *PerimeterBased* and *Area*

Note 2. Do not write class methods only the attributes.

Note 3. *LockDown* is an abstract class.

a) *(1.5 points)* Obtain the C# design using the design patterns studied in this course.

b) *(1 point)* Provide the header of the required constructors (without the body).

c) *(1 point)* Implement the needed C# code to create a LockDown object of type *PerimeterBased* with a restricted service affecting one area. The system must also have another Individual LockDown affecting just to one person. Use any arbitrary parameter values in the constructors to have a consistent system after its creation.