

Tutorial Letter 202/2/2015

Object-Oriented Analysis

ICT2622

Semesters 2

School of Computing

This tutorial letter contains important information about your module.

Bar code

Dear student

This tutorial letter, designated ICT2622/**202**/2015, contains the solution to Assignment 02.

Solutions in this tutorial letter are taken from the prescribed textbook. These solutions are meant to provide you with guidelines for answering the questions in your assignment.

NOTE: Assignment 01 solution was provided with assignment 01 results via SMS.

These solutions served as guidelines for marking your assignment. Some answers were not be exactly the same as the solution. Marks were awarded to answers that make sense.

■ At this stage you should have received the following study material:

- ICT2622/**101**/2015 (start-up letter)
- ICT2622/**301**/2015 (general Information)
- ICT2622/**202**/2015 (this tutorial letter)

If you have not received any of the above tutorial letters, please contact the UNISA at

- e-mail: study-info@unisa.ac.za

Alternatively, download the relevant tutorial matter from:

- our departmental website: <http://osprey.unisa.ac.za>
- myUNISA at <http://my.unisa.ac.za>

Both assignments will contribute 50/50 towards the year mark. And the year mark will contribute 20% towards the final mark, where the exam mark will contribute 80% towards the final mark.

BEST WISHES

THE LECTURERS

ICT2622

Assignment 02 Semester 2 – Compulsory assignment

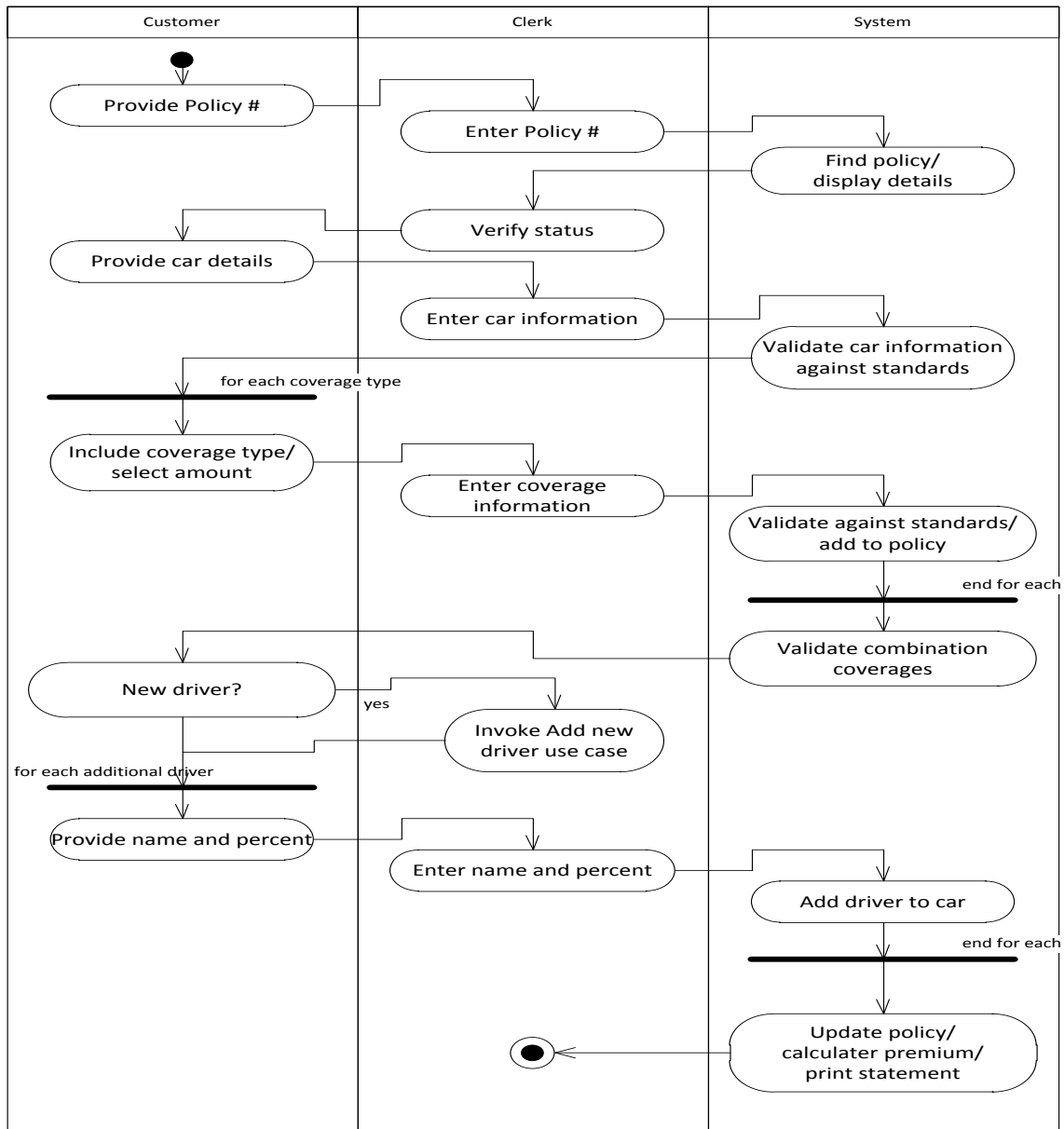
Question 1

[40]

Based on the following narrative, develop either an activity diagram or a fully developed description for the use case of *Add a new vehicle to an existing policy* in a car insurance system.

Narrative: *A customer calls a clerk at the insurance company and gives his policy number. The clerk enters this information, and the system displays the basic insurance policy. The clerk then checks the information to make sure the premiums are current and the policy is in active.*

The customer gives the make, model, year, and vehicle identification number (VIN) of the car to be added. The clerk enters this information, and the customer selects the type of coverage desired and the amount of each. The clerk enters the information, and the system records it and validates the request amount against the policy limits. After the coverage has been entered, the system insures the total coverage against all other ranges, including other cars on the policy. Finally, the customer must identify all the drivers and the percentage of the time they drive the car. If a new driver is to be added, then another use case Add new driver could be invoked (not for this assignment)



Use Case Name:	Add a new vehicle to an existing policy
Scenario:	Telephone instance with customer and clerk

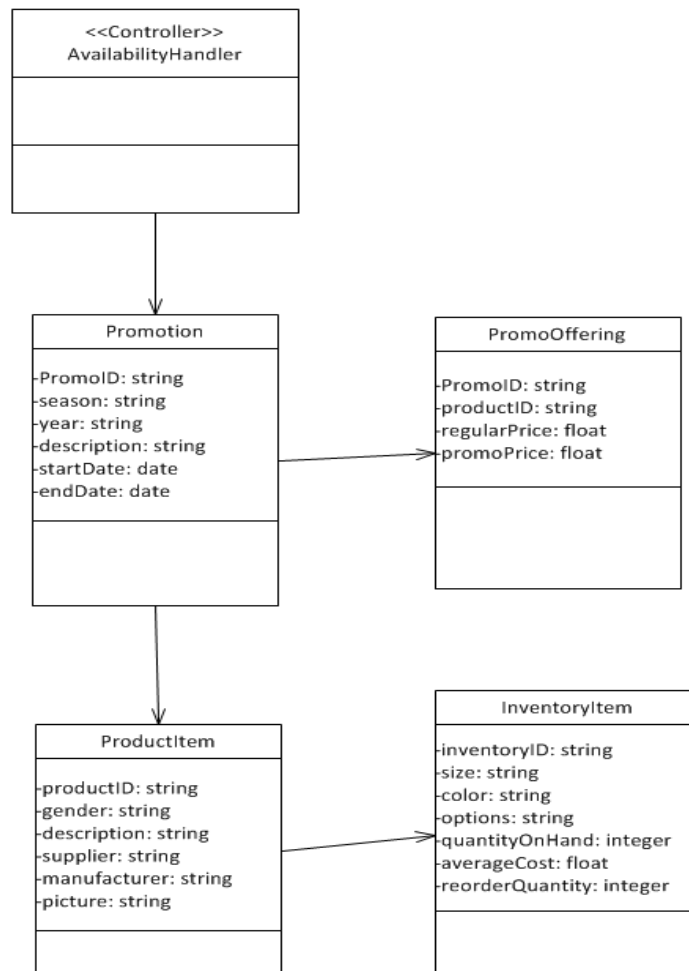
Triggering Event:	Customer buys a new vehicle.	
Brief Description:	Customer provides car information, requests coverage with amounts, identifies drivers of the new car. System updates the policy.	
Actors:	Customer service clerk	
Stakeholders:	Customer Customer service department	
Preconditions:	Customer policy must exist and be up to date. StandardVehicle control tables for this vehicle type and year must exist. StandardCoverage tables exist.	
Postconditions:	New vehicle object created and connected to policy. Also connected to StandardVehicle. New coverage objects created and connected to vehicle. Also connected to StandardCoverage. New driver (InsuredPerson) (if necessary) created and added to policy. Existing drivers and percentages updated. Policy updated with new premiums.	
Flow of Activities:	Actor	System
	1. Clerk enters customer information. 2. Clerk verifies policy is current. 3. Clerk enters car identification information. 4. Clerk enters each type of coverage customer requests, including deductibles and coverage amount. 5. Clerk indicates all coverages have been entered. 6. Clerk invokes <i>Add new person</i> use case if necessary. 7. Clerk changes driver percentages on this car and other cars. 8. Clerk indicates everything is complete.	1.1 System finds policy and displays details. 3.1 System validates that car has known standard. 4.1 System validates coverage requests. 5.1 System does combination validation on policy. 7.1 System updates driver information. 8.1 System updates policy, calculates new premium, prints new statement.
Exception Conditions:	2.1 If policy is not current, clerk requests payment or collects necessary information. 3.1 If car type is not in system, clerk refers customer to underwriting to handle this situation. 4.1 If coverage requests are out of range, clerk asks customer for changed amount. 5.1 If some combination is invalid, return to step 4.	

Question 2

[10]

In chapter 10, you learnt to develop a first-cut DCD, a set of CRC cards, and a final DCD for the *Create phone sale* use case for RMO. Create the first-cut DCD for the *Look up item availability* use case. (State your assumptions)

[ANSWER]



Question 3

[6]

Explain the syntax of a message on a sequence diagram.

[ANSWER]

'*' – means multiply occurring or looping

[true/false] – means test condition which is tested before the message is sent

return-value – is the return value which is returned to the originator of the message

:= -- is used to denote that a return value exists

message-name – this is the message name or service requested

(parameter-list) – this is the list of data parameters being sent with the message

Question 4**[6]**

Describe the risk factors associated with database design.

[ANSWER]

Since the database is an integral part of any information system, the performance and operation of the database is critical. Hence design decisions about what DBMS to use, how to configure it, and how to optimize it are usually quite complex and critically important.

Another risk factor is how to integrate a new database with existing databases. New systems normally are not completely independent of existing systems and database, and usually must interface with existing architectures. It is important that the new database not only integrate well, but that it does not cause problems with existing configurations.

Finally, good database design depends on having a complete or mostly complete problem domain model. While enhancing and adding tables and attributes is possible to already constructed databases, doing so can sometimes cause sub-optimization of data structures