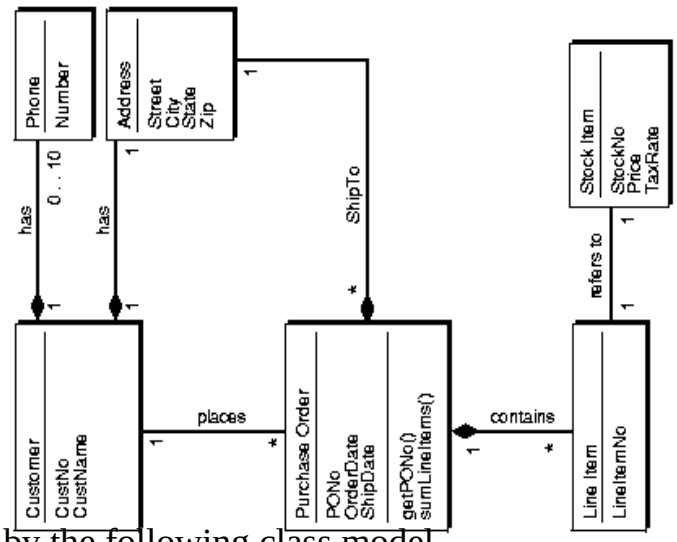
AIN SHAMS UNIVERSITY
FACULTY OF ENGINEERING
Computer and Systems Engineering Department
Post Graduate Students

1st Semester, 2015-2016 **Course Code: CSE608/CSE619** ***Time : 3.00 Hrs***

Advanced Software Engineering

The exam consists of **six** Questions in **six** Pages. ***Total Marks: 70 Marks*** **1/**10



Question 1 [12 marks]

Consider a purchase system that is described by the following class model

a) Create a suitable use-case diagram showing all needed use-cases **[3 marks]**

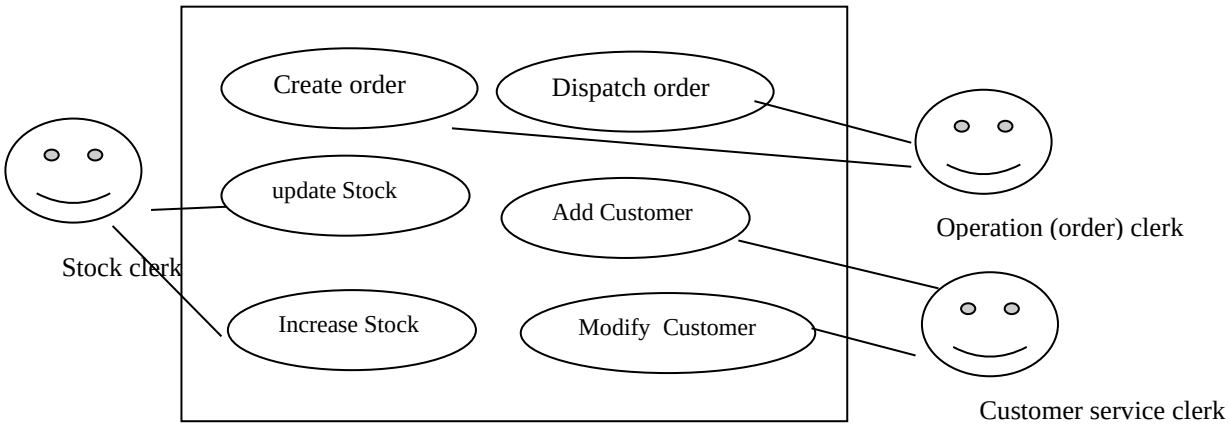
Looks like we need to add customers, delete customers update or modify customer data adding addresses or phones which will be part of modify or add customers

Create purchase orders, modify purchase orders, delete old ones, possibly we need add stock items purchase more items etc

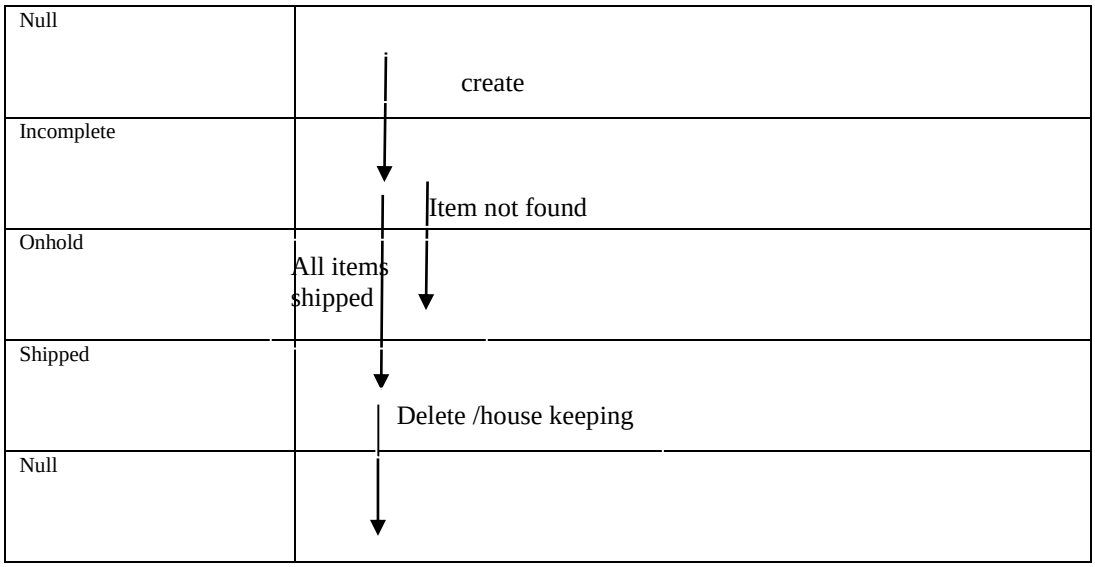
Assume the following actors

Customer service Clerk to add the info of the customers

Purchase order clerk to deal with orders and stock clerk to deal with stock items



b) Create a lifecycle diagram for a purchase order whose possible states are (null, incomplete, on hold, shipped) **[2 marks]**



c) List four suitable OCL invariants of this system **[2 marks]**

Item stocks cannot be negative

Contaxt stock_item

Inv: stock>-1

Each customer has exactly one address

Context customer

Inv: address.size = 1

Each purchase order has at least one line and at most 10 lines

Context purchase_order

Inv: order_line.size>0 and order_line.size<11

Shipping date must be after order date

Contxt purchase_order

Inv: ship_date>order_date

d) List OCL pre and post conditions for sumLineItems()**[2 marks]**

Context purchase_order::sumLineItems()

Pre: all(l:line_item | l.stock_item.price.isdefined()) all items in the order has price

Post: value = iterate(l:line_item ; sum: integer=0| sum+l.stock_item.price)

e) Create a CRUD matrix between the use-cases and classes in the diagram **[3 marks]**

	customer	phone	address	Purchase order	line item	Item stock
create order	R			C	C*	R
Dispatch order	R		R	R	R*	U*
Create customer	C	C*	C			
Modify customer	U	U*/C*/D*	U			
Update stock (Add item)						C
Increase stock						U

Question 2 [8 marks]

Decide whether the following statement true or false and validate your answer

a) Software life cycle refers to how software application is used
   False, it refers to the steps where it has been created
b) Problem specification is less important than program testing
   False, both are important and problem specification shape and affect the way the program is tested
c) Integration tests are done before unit tests
   False, integration is done after unit test to make sure the unit work together
d) Test coverage is based on black-box concept
   False , usually white or glass box so we need to cover all statements of the written code
e) Object oriented design produces faster programs

Exam. Date : 12 January 2016
Dr. Islam El-Maddah

Yes, OO produces programs quickly as it uses reusability
False, But the code itself is slower than the equivalent none OO version

f) Inheritance discourages code reuse
False inheritance is code reusability

g) The reusability of the software application is a client quality attribute.
False it is a developer quality attribute

h) Quality assurance is not related to production of software documentation.
False it is important to document program development stages, user-guide, installion guide etc, even the development process steps it has to be documented so to make sure every time the same process is followed

Question 3 [13 marks]

a) For the following code fragment, describe 3 different test cases, and for each, describe the class of test cases it represents. **[3 marks]**

TC1 (5,5,5, "equilateral") test triangles with equal sides
TC2 (5,5,7, "isosceles") test triangles with two equal sides
TC3 (3,4,5,"scalene") tests normal triangles

```
char * triangle (int x, int y, int z) {
/*
Pre-condition: The parameters are in ascending order (i.e. x <= y
<= z)
Post condition: If x, y and z are the lengths of the sides of a
triangle, this function classifies the triangle using one of the
three strings, "scalene", "isosceles" or "equilateral". If x, y,
and z do not form a triangle, the empty string is returned.
*/
char *r;
r="scalene";
if (x==y || y==z)
r="equilateral";
if (x==z)
r="isosceles";
if (x <= 0 || (x+y) <= z)
r="";
return (r); }
```

b) Full path coverage testing requires that every possible path through the code be tested at least once. Why is full path coverage testing desirable? **[2 marks]** For the code fragment above, how many test cases would be needed for full path coverage? List them **[2 marks]**

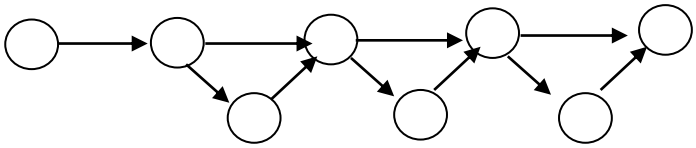
Exam. Date : 12 January 2016
Dr. Islam El-Maddah

full path test all possible ways the program can run, we have three if statements each could be true or false so basically we could have 8 ways to test this code covering all paths

c) Branch (Decision point) coverage testing requires that each outcome of each decision point be tested at least once. Why does this criteria usually require fewer test cases than full path coverage? **[2 marks]** What kinds of error might this testing **miss**? **[2 marks]**

branch covers tests the if statmenets each alone so for 3 if statements we could have two test cases one make all of them false other make all of them true!
Branch coverage could miss errors relating one branch to another branch

d) Draw a control flow diagram (CFD) for the triangle function **[1 mark]** and hence compute its cyclomatic complexity **[1 mark]**
cc=V(G) = e-n+2 = 10-8+2 = 4 (three closed areas +1) =4



Question 4 [12 marks]

a) Explain the following using illustrative diagrams when possible:

i. CRUD matrix analysis advantages **[2 marks]**
CRUD matrix shows all incomplete operation or data read, create, update , delete, It also shows unwanted process that does not do any thing with data
group processes together that can be developed together relating to the same data

ii. Difference between behavior and function UML models **[2 marks]**
Behavioral model show how the system reacts to events like state chart
Function show what is the main use(s) of the program like use case model

iii. How to decrease the testing cost without affecting the quality of software **[2 marks]**
Use suitable coverage methods, use both black and white the black should drive the white, model checking could decrease the testing effort, enough formal specification should also do the benefit f decreasing the testing effort

b) List the following very briefly

i. Two main advantage of object oriented development **[2 marks]**

Reusability, fast development, easier use of UML diagram to develop code maintain or test code.

ii. Two reasons why software engineering is different from other engineering disciplines **[2 marks]**
Software programs are not touchable like other products, customer needs may change during the development process

iii. Two software Application Examples where it is important to use model checking tools such as SMV **[2 marks]**
Safety critical systems, like reactors, power plants, boilers, ATM machines

---

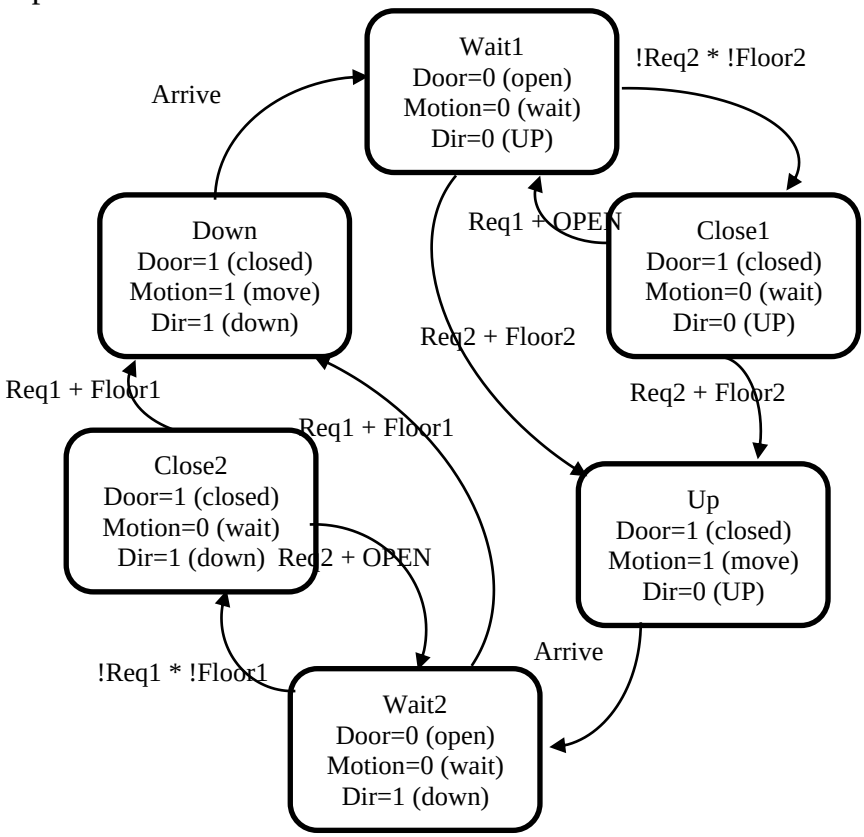
Question 5 [11 marks]

Consider the following state chart for a lift system, where * means logical and, + means logical or, ! means not. Each state has one name and three Boolean values for each output variables (Door, motion, and direction) Req1, Req2 are buttons from inside the lift car, and floor1, floor2 are buttons on each floor panel.

a) List all possible events sensed by the state machine **[2 marks]**
Arrive Req1, Req2, Floor 1, Floor 2, OPEN

b) Once been in state "CLOSE1", is state "DOWN" reachable? List event sequence needed to reach the state **[2 marks]**
yes the sequence is : Req 2 → arrive → Floor 1

c) List Properties in LTL or CTL for the following:
- there will be no motion in the next state if the door is open
  AG [door = open → AX (no motion)]
- Req2 implies sometime in the future state wait2 is reached



Exam. Date : 12 January 2016
Dr. Islam El-Maddah

AG(Req2 → EF (state = wait2))

- There will be always possibility to reach state close1 **[3 marks]**
- **AG(EF(state = close1))**

d) Write down an SMV module to describe this behavioral model **[4 marks]**

```
Module main
VAR
State ={wait1, wait2, close1, close2, up, down};
Arrive Req1, Req2, Floor 1, Floor 2, OPEN: Boolean;

ASSIGN
Init(state) := wait1;
Next(state):= case
State=wait1 & (!Req2 | !floor2):closed1;
State=wait1 & (Req2 | floor2): UP;
:
1: state;
esac;
```

Question 6 [14 marks]

A software program is to be developed to manage a research conference processes. The conference runs every year. Researchers write two types of research papers; Type A (a research paper) where the maximum number of pages is 12 and type B (a poster paper) where the maximum number of pages is 4. Each paper has a title, an abstract and a body with a maximum of 10 standard keywords from related topics (published on the conference webpage). The conference has a community of ten members with one of them assigned as a conference chair. A paper has between one author and up to a maximum of 6 authors. Only one author email is used on behalf of the paper to communicate information. Each author supplies name, details, an email and an affiliation address.

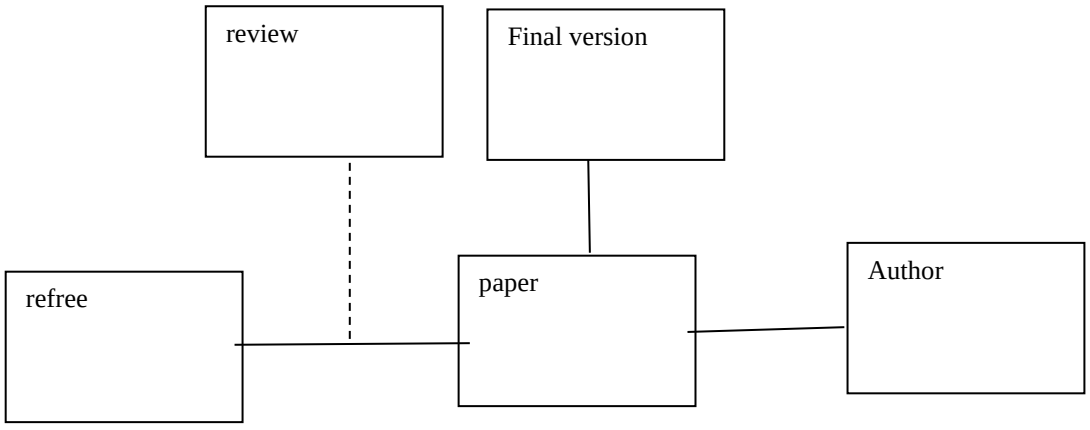
Papers once sent to the conference are assigned to three referees. Each of them sends a review within one month to judge the paper. Papers are to be judged as rejected, accepted as it is or accepted with changes. Once the final paper is sent to the conference and the publication fees are paid, the paper is added to the conference final program. At the conference venue, accepted paper are marked as presented, when one of the authors present the paper during the conference program.

a) List four stakeholders of this application **[2 marks],** three innovative requirements to enhance the business of this application process **[3 marks]**

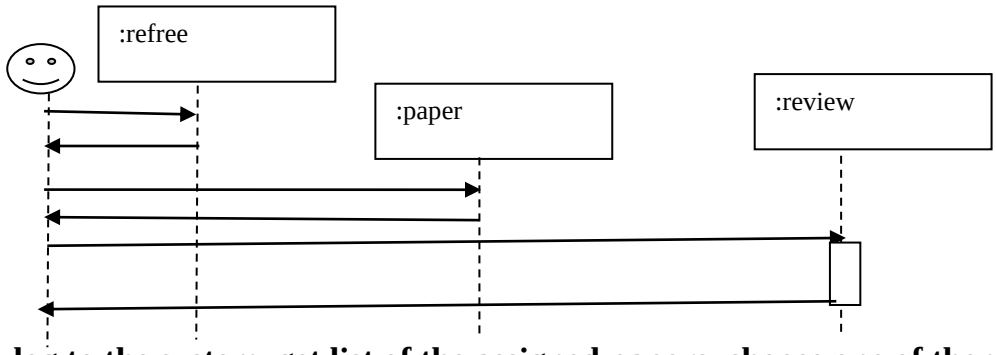
Author/Researcher, Reviewer, Community Member, Chair, Operator to enter data
Send early call for papers for old authors
Ask for author feedback about reviewing, website, etc
Ask for feedback about quality of papers
Ask fr reviewer feedback about best papers

b) List two needed non-functional requirements in details **[2 marks]**
Information security, access control
System should be scalable even though with big number of papers and authors (performance)

c) Create a suitable class diagram with all possible attributes and methods for the needed classes **[5 marks]**



d) Based on part c, Create a sequence diagram (SD) for a referee sending a review of one paper to the conference **[2 marks]**



Refree log to the system, get list of the assigned papers, choose one of them get its text, start creating new review object related to this paper.

Exam. Date : 12 January 2016
Dr. Islam El-Maddah

Sample of SMV module

```
MODULE main  -- comment
VAR  -- variable section
request: boolean; -- true of false which is the same as 0 or 1
state: {ready, busy};

ASSIGN
init(state) := ready;  -- initial values
next(state) := case  -- the next value
state=ready & request: busy;
1: {ready, busy};
esac;

SPEC AG (request -> EX state=busy) --specification in CTL
LTLSPEC G(request -> F (state = busy)) -- specification in linear temporal logic
```

OCL Constraints

Can define invariants over model state or sets or pre/post conditions for operations or bodies (i.e. definitions) for queries.

Invariants:

```
context Class1
inv attr1 > 100
context Class2
inv secondInvariant: attr2 < 10
```

Can have as many "inv" statements as required, optionally named, and the resulting invariant is their conjunction ("i1 and i2 and ...").

Pre and Post Conditions:

```
context Class1::method1(v1: Integer) : Integer
pre valueIsLargeEnough: v1 >= 100
post: attr1 >= attr1@pre + 100 and result > v/10
```

The "@pre" notation refers to the "before" state and "result" is a reserved word for the result of the operation the returning value (if it has one).

Exam. Date : 12 January 2016
Dr. Islam El-Maddah

Can have as many "pre" and "post" statements as required (optionally named) and the resulting pre/post conditions are their conjunction ("p1 and p2 and ...").

Collection Manipulation Operations

Operations are applied to collections using the "->" operator (e.g. items->isEmpty(), where "items" is a collection).

OCL provides a rich set of operations for use in collection expressions, including:

= / <> Are the collections identical (not identical).

- Return the value of the set difference of the arguments (Set and OrderedSet only).

append(obj) Append obj to an ordered collection.

asBag(), asSet(), asOrderedSet(), asSequence() Type conversion operations (available to/from all collection types).

at(idx) Return object at index of ordered collection.

count(obj) Number of times that obj appears in a collection.

excludes(obj) Does count(obj) = 0 ?

excludesAll(coll) Does count(obj) = 0 hold for all items in collection coll?

excluding(obj) Value of collection with object obj removed. first() The first item in the ordered collection.

includes(obj) Is count(obj) > 0 ?

includesAll(coll) Does count(obj) > 0 hold for all items in collection coll?

including(obj) Value of collection with object obj added.

intersection(coll) Value of the intersection of the unordered collection and the unordered collection coll.

isEmpty() Is collection's size() = 0 ?

last() The last item in the ordered collection.

notEmpty() Is collection's size() > 0 ?

prepend(obj) Value of the ordered collection with the object obj prepended to it.

size() Number of items in the collection.

subOrderedSet(start,end) Value of a subset of an ordered set based on indexing.

subSequence(start,end) Value of part of a sequence, based on indexing.

sum() Sum all items in the collection (Integer/Real only).