Muhammad Mubeen Alam

ID#6906 BS(SE)

5/20/2020

IQRA NATIONAL UNIVERSITY

Sessional 2020 Examination

Course Name	Max. Marks	Deadline	Date	Instructor
Software Requirement Specification	20	10th June, 2020	10th May, 2020	Aasma Khan

- **Attempt all questions.**
- **Marks will be given as per the DEPTH of the answer, not LENGTH.**

Question No: 01 **(20)**

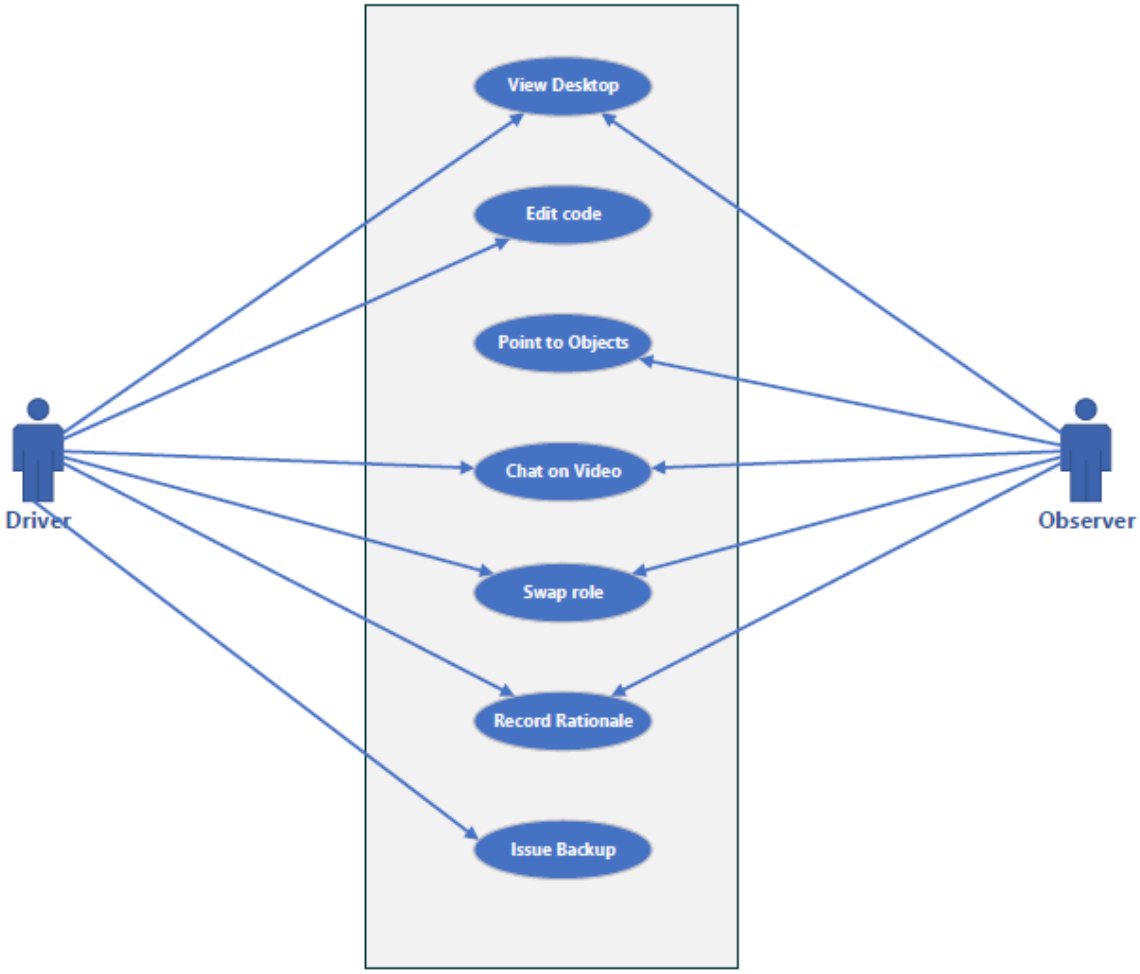
Pair programming is an agile software development technique in which two programmers work together at one work station. One types in code while the other reviews each line of code as it is typed in. The person typing is called the driver. The person reviewing the code is called the observer. The two programmers switch roles frequently (possibly every 30 minutes or less).

Suppose that you are asked to build a system that allows Remote Pair Programming. That is, the system should allow the driver and the observer to be in remote locations, but both can view a single desktop in real-time. The driver should be able to edit code and the observer should be able to "point" to objects on the driver's desktop. In addition, there should be a video chat facility to allow the programmers to communicate. The system should allow the programmers to easily swap roles and record rationale in the form of video chats. In addition, the driver should be able to issue the system to backup old work.

- Draw a use case diagram to show all the functionality of the system.
- Describe in detail four non-functional requirements for the system.
- Give a prioritized list of design constraints for the system and justify your list and the ordering.
- Propose a set of classes that could be used in your system and present them in a class diagram.

ANS :

PART-A:



Notes:
* use case names should be verb phrases.
* your answer may be slightly different from what is shown on the left. Please put in brief statements to explain your design.

PART-B:

Any of the following would be sufficient as four answers. You can have additional NFRs that are not on my list below. Be sure to provide an explanation for each NFR associated with the program (other than "ease of use", "reliability", "security", etc.), which is the preferred NFR for most applications.

* Easy to use - The front-end interface should be simple and easy to use.

* Real-time performance - changes made by the driver can be detected quickly without delay; Video chat should be smooth without delay.

* Availability - The program should be available to both program editors at all times.

* Load - programmers can use the program no matter what computer and operating system they use.

* Security - The backup code must be kept secure and protected from unauthorized access.

* Cost - Users may pay more than $ 5 per month to use RPP.

* Reliability - The system must be reliable, that is, it should not crash when the Internet speed is slow, and the user can resume the session later if the Internet connection is abruptly slowed down.

PART-C:

Note that since you need to provide only 4 NFRs in part b), so you only need one describe your ordering of the associated design agreements.

In your reply, you must specify your order, e.g., if you include "convenience" internally position 1 and "real-time performance" in position 2, and then point out the reason why you think "free." use "is more important than" real-time "use. You should also talk briefly whether there are conflicting design issues again, based on yours to order, which design obstacle should be emphasized.

Note: NFRs become bottlenecks in design and, as a result, "design issues"

They are often treated in the same way as "NFRs". However, there are differences between them.

Example # 1:

The System Are managed - by NFR.

This NFR may lead to trouble in the programming language used for system operations (e.g., i

Java programming language (rather than C and C ++) can be selected for compilation and NFR).

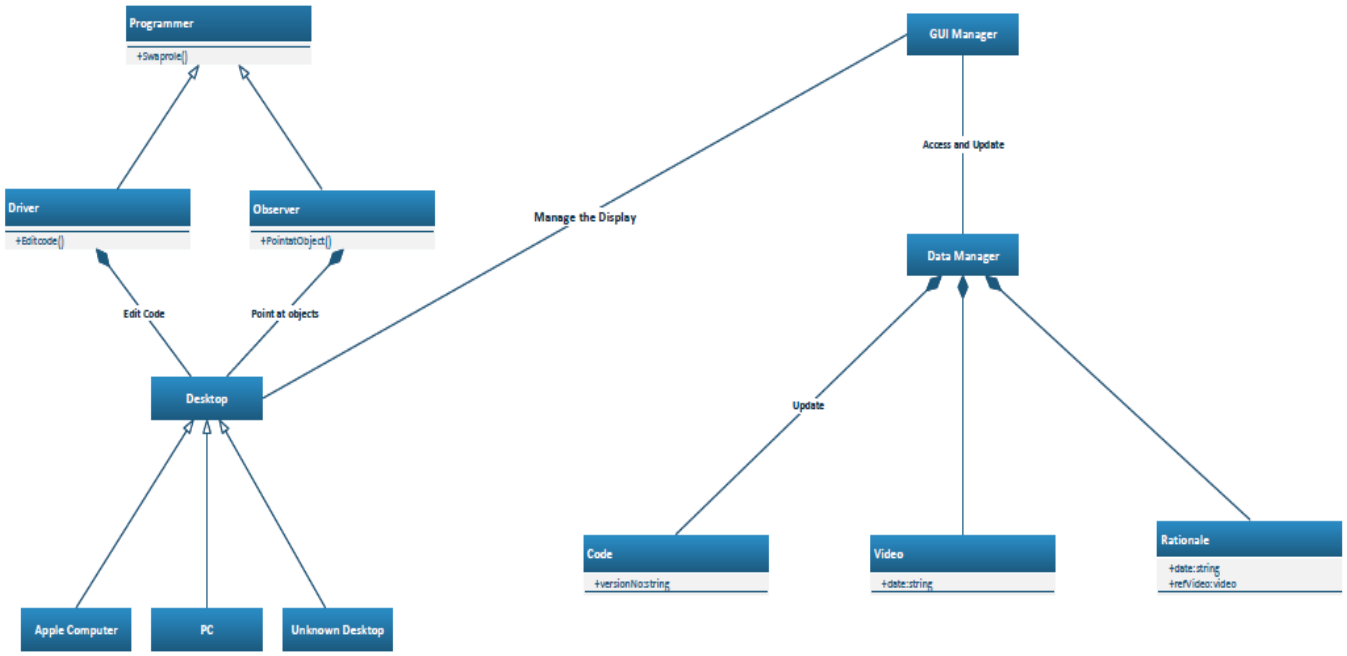
Example # 2:

Security : the system must be protected - by NFR.

Design issues whether user authentication must be localized, the communication rule must be

text, and / or information must be stored on the server after the fire.

PART-D:



Now We Explain About this Diagram….

Example of Class Diagram and above captures the most of the classes and relationships. i have installed the Anonymous Desktop class to extend the RPP in the future. Obviously when I did this class diagram I already have the Abstract a Factory design pattern in my mind.

In your class drawing you don't need to specify attributes and functionality unless the ones that really matter. Other functions may be excluded from the use case diagram PART-A. For example, you can quickly see that, under Driver section, I have to have issue Backup () functionality. Also, under the Programmer section, Me it should have desktop View (), chat On Video (), and Recorder () recording. Maybe we should you have a session as a class and also associated with Data Manager? I took it out because it is not explicitly stated in the description. Note that from Apple Computer and PC are not explicitly mentioned in the description, if you do not have the material below under Desktop is okay too.

The second type is to have the Programmer section associated with the desktop section

in turn. However, in this second version we cannot write that the organization is so

'edit' or 'point things out'.

The GUI Manager class and the Data Manager class (both can be meeting places) are included help manage the display and access of data. Software-view-controller (MVC) software the build pattern is accepted here: Data Manager takes care of 'model',

GUI Manager, which communicates with Data Manager and Desktop, takes care of 'viewing', and the tasks performed by the programmer of the two programs constitute the 'controller' component.

THE END ☺