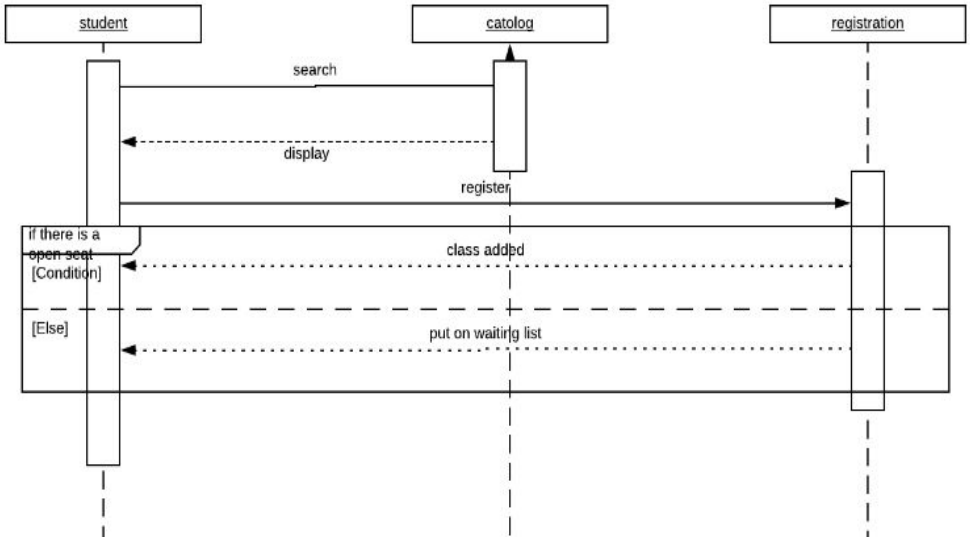
Class Exercise 1:

System Modeling:

1. Develop a sequence diagram showing the interactions involved when a student registers for a course in a university. Courses may have limited enrolment, so the registration process must include checks that places are available. Assume that the student accesses an electronic course catalog to find out about available courses.

   Answer



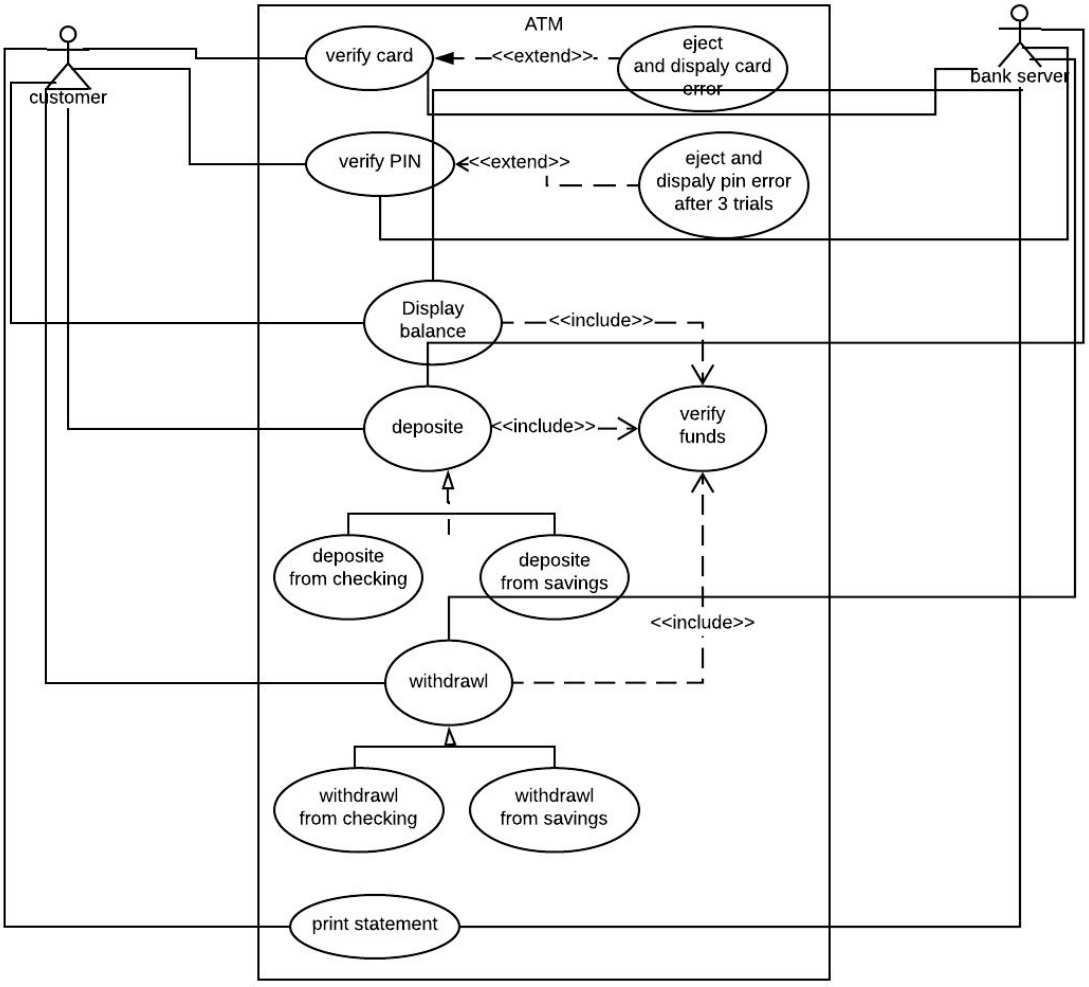
2. Look carefully at how messages and mailboxes are represented in the email system that you use. Model the object classes that might be used in the system implementation to represent a mailbox and an e-mail message.(attributes, methods)

   Answer

Mail Message	Mailbox
Sender Receive list Cc list Bcc list Date	Name Pathname Creatdate Messages Unread messages

Subject Message info Message body Attachment Signature	Flagged messages Deleted messages
read() reply() Reply all() print() forward() send()	Move message() Copy message() Delete message() Fetch mail() Create() Rename() delete()

3. Using your knowledge of how an ATM is used, develop a set of use cases that could serve as a basis for understanding the requirements for an ATM system.
   Answer



Requirement:

4. Discover ambiguities or omissions in the following statement of requirements for part of a ticket-issuing system:

   An automated ticket machine sells rail tickets. Users select their destination and input a credit card and a personal identification number. The rail ticket is issued and their credit card account charged.

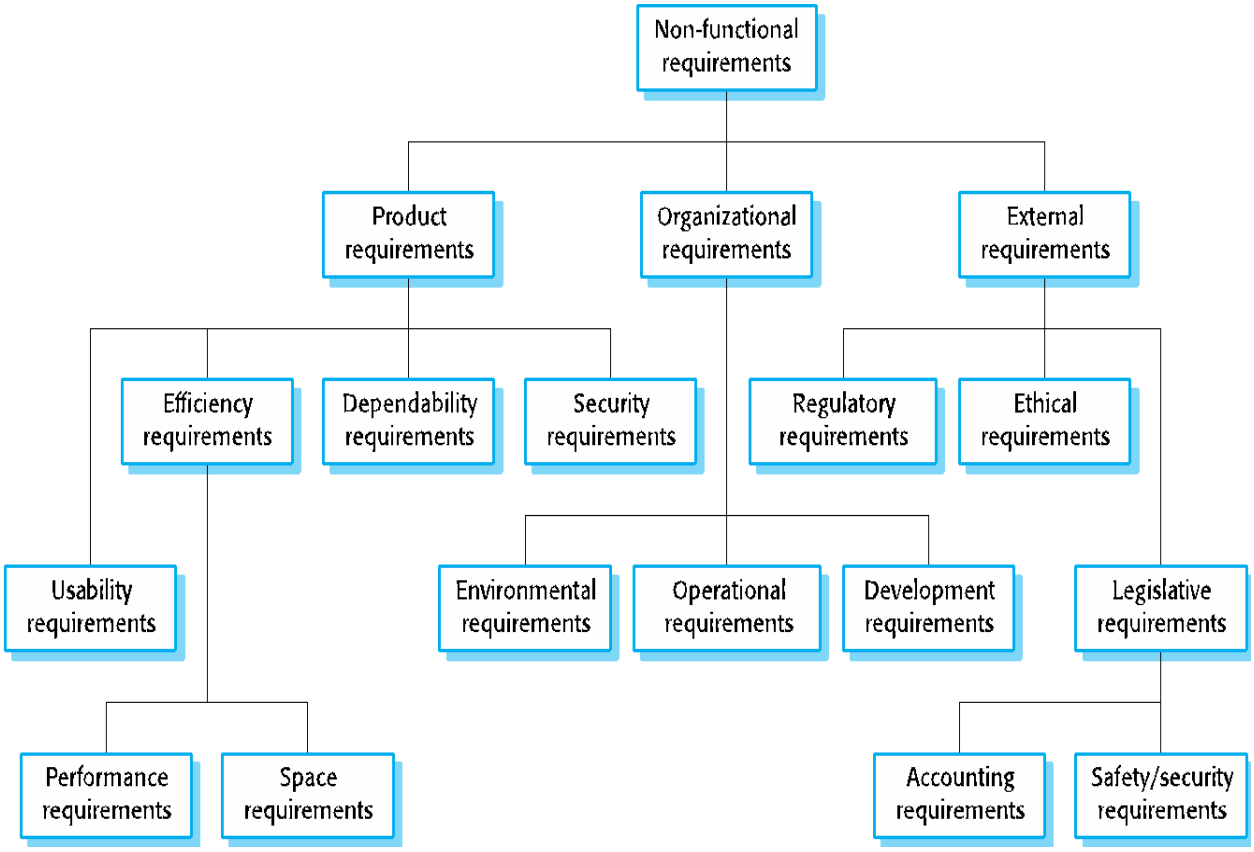
   When the user presses the start button, a menu display of potential destinations is activated, along with a message to the user to select a destination and the type of ticket required. Once a destination has been selected, the ticket price is displayed and customers are asked to input their credit card. Its validity is checked and the user is then asked to input their personal identifier (PIN). When the credit transaction has been validated, the ticket is issued.

Answer:

1. Can a customer buy multiple tickets?
2. Can a customer cancel a request if a mistake has been made?

3. How should the system respond if an invalid card is input?
4. What happens if customers try to put their card in before selecting a destination (as they would in ATM machines)
5. Must the user press the start button again if they wish to buy another ticket to a different destination?
6. Should the system only sell tickets between the station where the machine is situated and direct connections or should it include all possible destinations?

5. Write a set of non-functional requirements for the ticket-issuing system, setting out its expected reliability and response time.(Usually a ticket-issuing system more active working time is from 06:00 - 23:00)



Answer:

1. Between 06:00 - 23: 00 in any day, total down time should not exceed 5 minutes.

2. Between 06:00 - 23: 00 in any day, recovering time after a system failure should not exceed 2 minutes.

3. Between 23:00 - 06:00 in any day, total down time should not exceed 20 minutes.

4. When a customer presses a button, the system update time should be less than 0.5 seconds.

5. The ticket issuing time after a credit card validation has been received should be less than 10 seconds.

6. When validating, a status message pointing out what the machine is processing and what will be expected should show.

7. The maximum acceptable failure rate for ticket issue request is 1:10000.

6. Suggest why it is important to make a distinction between developing the user requirements and developing system requirements in the requirements engineering process.

Answer:

There is a fundamental difference between the user and the system requirements that mean they should be considered separately.

1. The user requirements are intended to describe the system's functions and features from a user perspective and it is essential that user understand these requirements. They should be expressed in natural language and may not be expressed in great detail, to allow some implementation flexibility. The people involved in the process must be able to understand the user's environment and application domain.
2. System requirements are much more detailed than the user requirements and intended to be a precise specification of the system that my be part of a system contract. They may also be used in situations where development is outsourced and the development team need a complete specification of what should be developed. It is after user requirement have been established.

Agile Development:

7. Explain how the principles underlying agile methods lead to the accelerated development and deployment of software.

Answer:

The principles underlying agile development are:

1. *Individual and interactions over processes and tools*. By taking advantages of individual skills and ability and by ensuring that the development team know what each other are doing, the overheads of formal communication and process assurance are avoided. This means that the team can focus on the development of working software.

2. *Working software over comprehensive documentation*. This contributes to accelerated development because time is not spent developing, checking and managing documentation. Rather, the programmer's time is focused on the development and testing of code.

3. *Customer collaboration over contract negotiation*. Rather than spending time developing, analyzing and negotiating requirements to be included in a system contract, agile developers argue that it is more effective

to get feedback from customer's directly during the development about what is required. This allows useful functionality to be developed and delivered earlier than would be possible if contracts were required.

4. *Responding to change over following a plan*. Agile developers argue (rightly) that being responsive to change is more effective than following a plan-based process because change is inevitable whatever process is used. There is significant overhead in changing plans to accommodate change and the inflexibility of a plan means that work may be done that is later discarded.

Software Processes:

8. Giving reasons for your answer based on the type of system being developed, suggest the most appropriate generic software process model that might be used as a basis for managing the development of the following systems:
    a. A system to control anti-lock braking in a car
    b. A virtual reality system to support software maintenance
    c. A university accounting system that replaces an existing system
    d. An interactive travel planning system that helps users plan journeys with the lowest environmental impact

Answer:

1. *Anti-lock braking system* This is a safety-critical system so requires a lot of up-front analysis before implementation. It certainly needs a plan-driven approach to development with the requirements carefully analysed. A waterfall model is therefore the most appropriate approach to use, perhaps with formal transformations between the different development stages.

2. *Virtual reality system* This is a system where the requirements will change and there will be an extensive user interface components. Incremental development with, perhaps, some UI prototyping is the most appropriate model. An agile process may be used.

3. *University accounting system* This is a system whose requirements are fairly well-known and which will be used in an environment in conjunction with lots of other systems such as a research grant management system. Therefore, a reuse-based approach is likely to be appropriate for this.

4. *Interactive travel planning system* System with a complex user interface but which must be stable and reliable. An incremental development approach is the most appropriate as the system requirements will change as real user experience with the system is gained.