CHƯƠNG 1

2. Phân biệt giữa công nghệ phần mềm và khoa học máy tính. (The difference between software engineering and computer science)(Slide 13 – Chương 1)

The difference between software engineering and computer science

❖ Computer science is concerned with theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software.

❖ Computer science theories are still insufficient to act as a complete underpinning for software engineering.

3. Phân biệt giữa công nghệ phần mềm và kỹ thuật hệ thống. (The difference between software engineering and system engineering) (Slide 14 – Chương 1)

The difference between software engineering and system engineering

❖ System engineering is concerned with all aspects of computer-based systems development including hardware, software and process engineering. Software engineering is part of this process concerned with developing the software infrastructure, control, applications and databases in the system.

❖ System engineers are involved in system specification, architectural design, integration and deployment.

CHUONG 4

4. Context diagram dùng để làm gì? Trình bày vai trò của nó trong việc xác định ranh giới hệ thống.

Context models

❖ Context models are used to illustrate the operational context of a system - they show what lies outside the system boundaries.

❖ Social and organisational concerns may affect the decision on where to position system boundaries.

❖ Architectural models show the system and its relationship with other systems.

System boundaries

❖ System boundaries are established to define what is inside and what is outside the system.

 They show other systems that are used or depend on the system being developed.

❖ The position of the system boundary has a profound effect on the system requirements.

❖ Defining a system boundary is a political judgment

 There may be pressures to develop system boundaries that increase / decrease the influence or workload of different parts of an organization.

5. Mô hình Use Case được sử dụng như thế nào trong việc xác định yêu cầu phần mềm? Một Use Case bao gồm những thành phần nào? (Slide 19-20 Chapter 5)

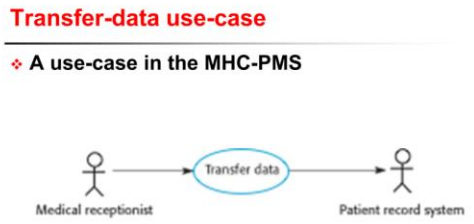
Use case modeling

❖ Use cases were developed originally to support requirements elicitation and now incorporated into the UML.

❖ Each use case represents a discrete task that involves external interaction with a system.

❖ Actors in a use case may be people or other systems.

❖ Represented diagramatically to provide an overview of the use case and in a more detailed textual form.



7. Trình bày khái niệm và mục tiêu của mô hình trạng thái (State Machine Model). Đưa ví dụ minh họa một hệ thống thực tế. (Slide 44-45 Chapter 5)

State machine models

❖ These model the behaviour of the system in response to external and internal events.

❖ They show the system's responses to stimuli so are often used for modelling real-time systems.

❖ State machine models show system states as nodes and events as arcs between these

nodes. When an event occurs, the system moves from one state to another.

❖ Statecharts are an integral part of the UML and are used to represent state machine models.

24. Mẫu thiết kế (design pattern) là gì? Trình bày các thành phần cấu thành nên một mẫu thiết kế. Cho ví dụ về mẫu Observer và mục đích sử dụng của nó. (Slide 29 -35 Chương 7)

Design patterns

❖ A design pattern is a way of reusing abstract knowledge about a problem and its solution.

❖ A pattern is a description of the problem and the essence of its solution.

❖ It should be sufficiently abstract to be reused in different settings.

❖ Pattern descriptions usually make use of object-oriented characteristics such as inheritance and polymorphism.

Pattern elements

❖ Name

 A meaningful pattern identifier.

❖ Problem description.

❖ Solution description.

 Not a concrete design but a template for a design

solution that can be instantiated in different

ways.

❖ Consequences

 The results and trade-offs of applying the pattern.

The Observer pattern

❖ Name

 Observer.

❖ Description

 Separates the display of object state from the object itself.

❖ Problem description

 Used when multiple displays of state are needed.

❖ Solution description

 See slide with UML description.

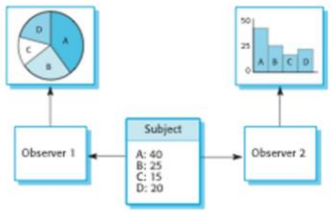
❖ Consequences

 Optimisations to enhance display performance are impractical.

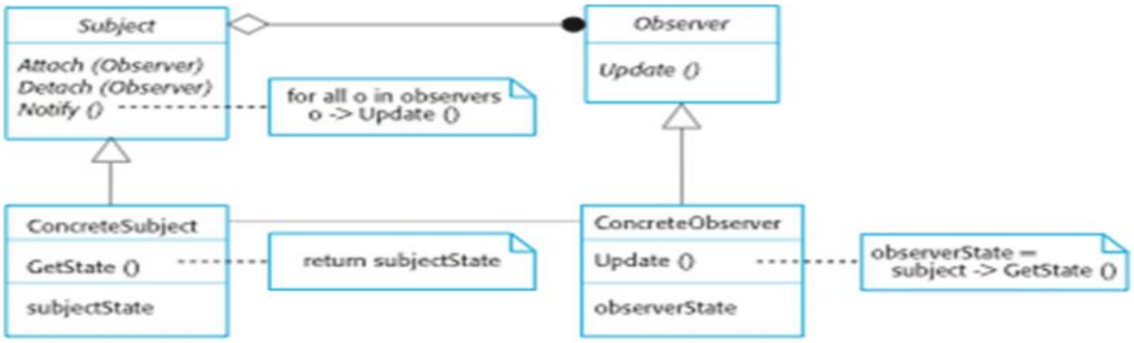
The Observer pattern

Pattern name	Observer
Description	Separates the display of the state of an object from the object itself and allows alternative displays to be provided. When the object state changes, all displays are automatically notified and updated to reflect the change.
Problem description	In many situations, you have to provide multiple displays of state information, such as a graphical display and a tabular display. Not all of these may be known when the information is specified. All alternative presentations should support interaction and, when the state is changed, all displays must be updated. This pattern may be used in all situations where more than one display format for state information is required and where it is not necessary for the object that maintains the state information to know about the specific display formats used.

Multiple displays using the Observer pattern



A UML model of the Observer pattern



The Observer pattern

Pattern name	Observer
Solution description	This involves two abstract objects, Subject and Observer, and two concrete objects, ConcreteSubject and ConcreteObject, which inherit the attributes of the related abstract objects. The abstract objects include general operations that are applicable in all situations. The state to be displayed is maintained in ConcreteSubject, which inherits operations from Subject allowing it to add and remove Observers (each observer corresponds to a display) and to issue a notification when the state has changed. The ConcreteObserver maintains a copy of the state of ConcreteSubject and implements the Update() interface of Observer that allows these copies to be kept in step. The ConcreteObserver automatically displays the state and reflects changes whenever the state is updated.
Consequences	The subject only knows the abstract Observer and does not know details of the concrete class. Therefore there is minimal coupling between these objects. Because of this lack of knowledge, optimizations that enhance display performance are impractical. Changes to the subject may cause a set of linked updates to observers to be generated, some of which may not be necessary.

29. Open source development là gì? Nêu các lợi ích và thách thức khi phát triển phần mềm theo mô hình mã nguồn mở. (Slide 54 – Chapter 7)

Open source development

❖ Open source development is an approach tosoftware development in which the source code of a software system is published and volunteers are invited to participate in the development process

❖ Its roots are in the Free Software Foundation (www.fsf.org), which advocates that source code should not be proprietary but rather should always be available for users to examine and modify as they wish.

❖ Open source software extended this idea by using the Internet to recruit a much larger population of volunteer developers. Many of them are also users of the code.

9. Unit/component/system testing: đặc điểm? Component: Slide 37 chapter 8 Unit: Slide 23 Chapter 8 System: Slide 43 Chapter 8

Component testing

❖ Software components are often composite components that are made up of severalinteracting objects.

 For example, in the weather station system, the reconfiguration component includes objects that deal with each aspect of the reconfiguration.

❖ You access the functionality of these objects through the defined component interface.

❖ Testing composite components should therefore focus on showing that the component interface behaves according to its specification.

 You can assume that unit tests on the individual objects within the component have been completed.

Unit testing

❖ Unit testing is the process of testing individual components in isolation.

❖ It is a defect testing process.

❖ Units may be:

 Individual functions or methods within an object.

 Object classes with several attributes and methods.

 Composite components with defined interfaces used to access their functionality.

System testing

❖ System testing during development involves integrating components to create a version of the system and then testing the integrated system.

❖ The focus in system testing is testing the interactions between components.

❖ System testing checks that components are compatible, interact correctly and transfer the right data at the right time across their interfaces.

❖ System testing tests the emergent behaviour of a system.

Vẽ mô hình mô tả qui trình xử lý nhập học trong ứng dụng quản lý sinh viên bằng PowerDesigner

Qui trình xử lý nhập học được thực hiện như sau:

- Sinh viên nộp hồ sơ cho phòng đào tạo. PDT kiểm tra hồ sơ, nếu có sai sót thì yêu cầu sinh viên cập nhật lại. Khi hồ sơ hoàn chỉnh, PDT nhập hồ sơ (lưu trữ lý lịch sinh viên và biên nhận hồ sơ vào CSDL), gởi thông tin về cho giáo vụ Khoa. Đồng thời họ xuất biên nhận hồ sơ và giấy vào lớp để sinh viên nộp về giáo vụ Khoa.
- GV Khoa nhận giấy vào lớp và cập nhật danh sách lớp, lưu trữ vào CSDL riêng thuộc Khoa và cấp giấy chứng nhận cho sinh viên

Đối tượng, bộ phận, phòng ban tham gia: Sinh viên, phòng đào tạo, văn phòng Khoa, phòng tài chính (nếu cần).

Những việc cần thực hiện:

- Chuẩn bị hồ sơ nhập học
- Nộp hồ sơ
- Tiếp nhận hồ sơ
- Nộp giấy vào lớp

- Tiếp nhận giấy vào lớp

Student Admission Process Description

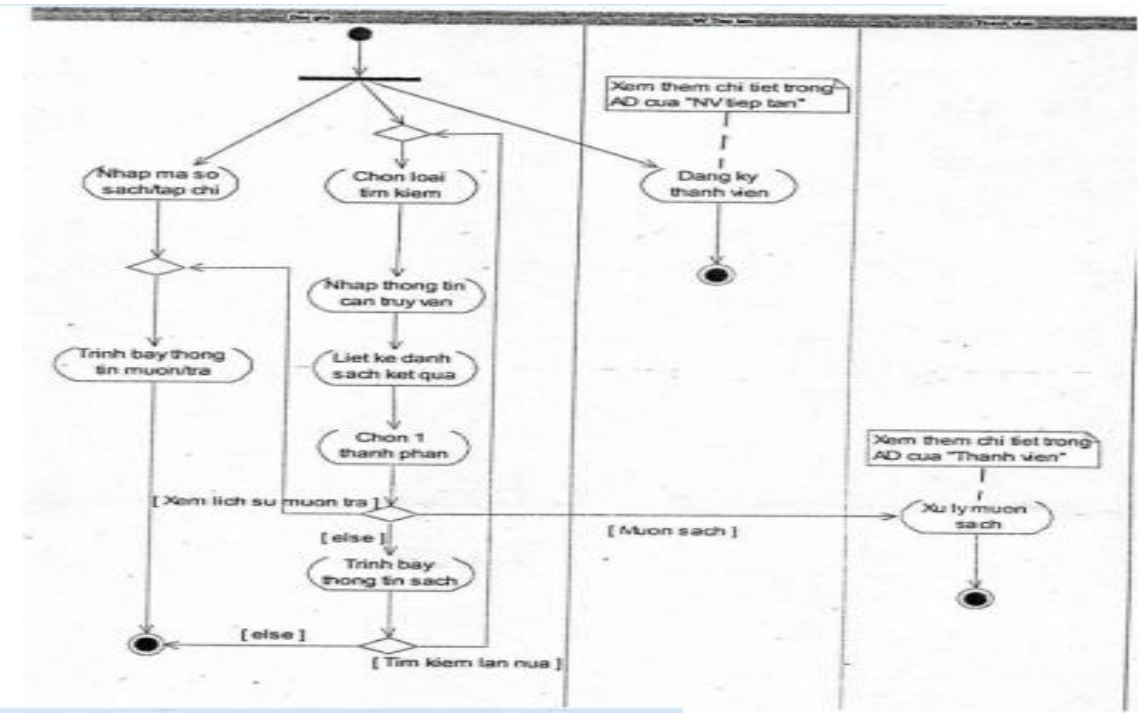
The student admission process is carried out as follows:

- The student submits their admission application to the Training Department (TD). The TD reviews the application. If there are any errors, the TD requests the student to update the application. Once the application is complete, the TD enters the application into the system (storing the student's profile and application receipt in the database), and sends the information to the Faculty Administration Office (FAO). At the same time, the TD issues an application receipt and a class admission slip for the student to submit to the FAO.
- The FAO receives the class admission slip, updates the class roster, stores it in the Faculty's separate database, and issues a certificate of admission to the student.

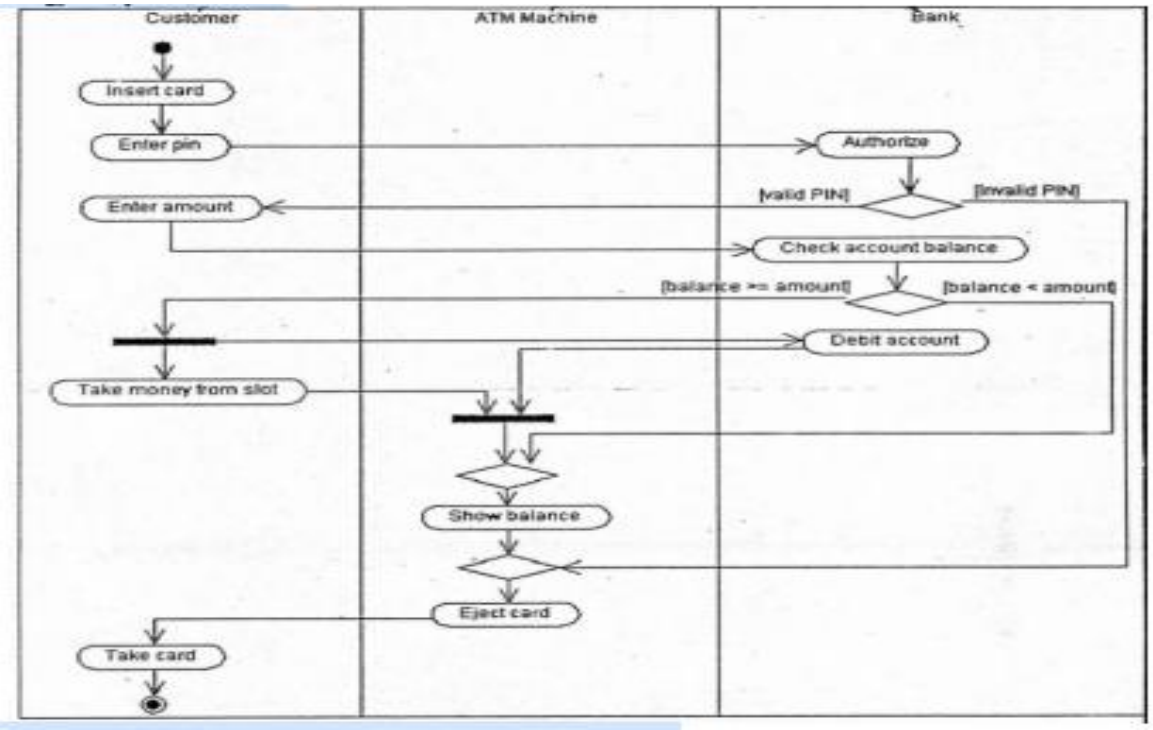
Entities, departments, and offices involved: Student, Training Department (TD), Faculty Administration Office (FAO), Finance Department (if necessary).

Tasks to be performed:

- Prepare admission application
- Submit application
- Receive application
- Submit class admission slip
- Receive class admission slip

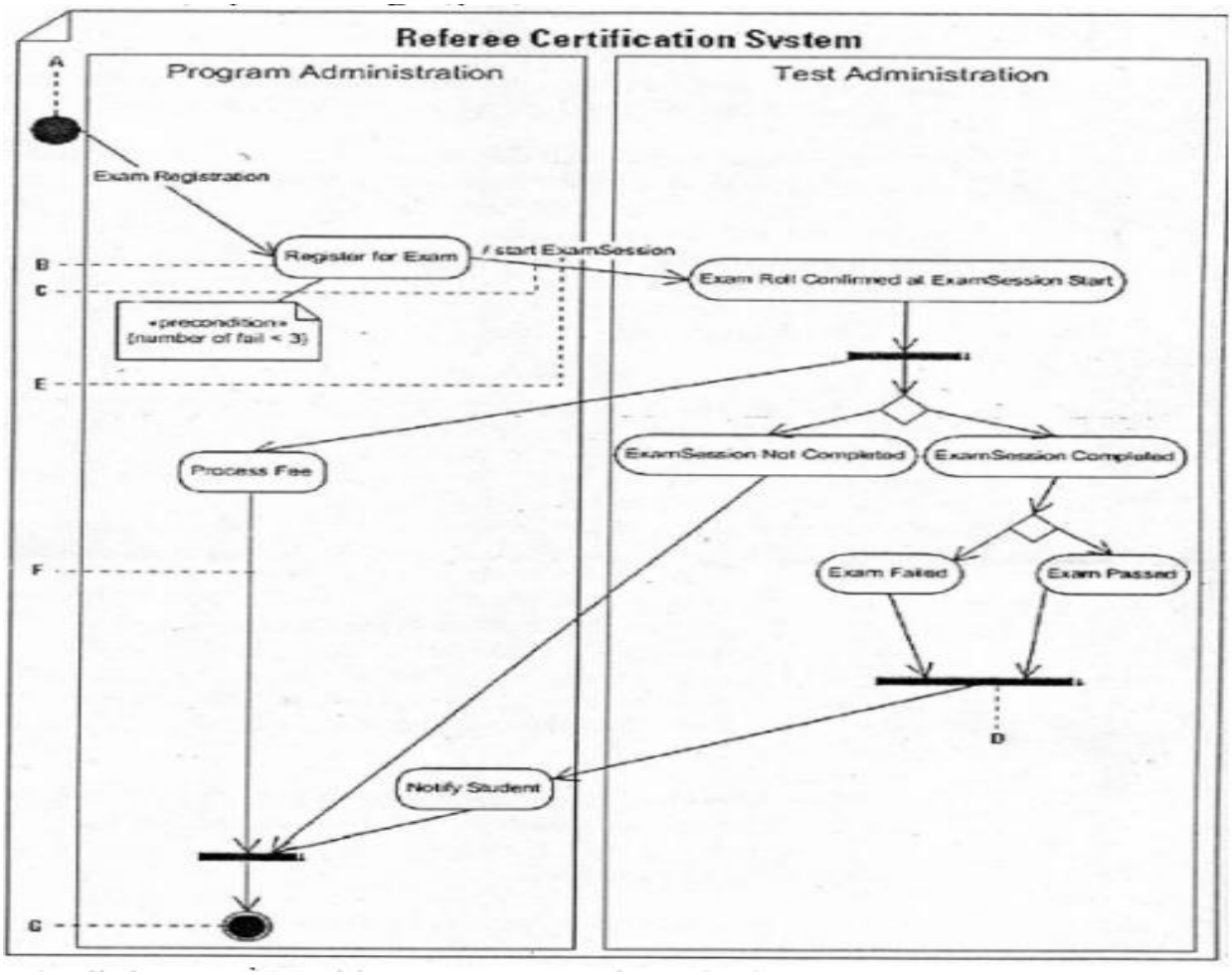


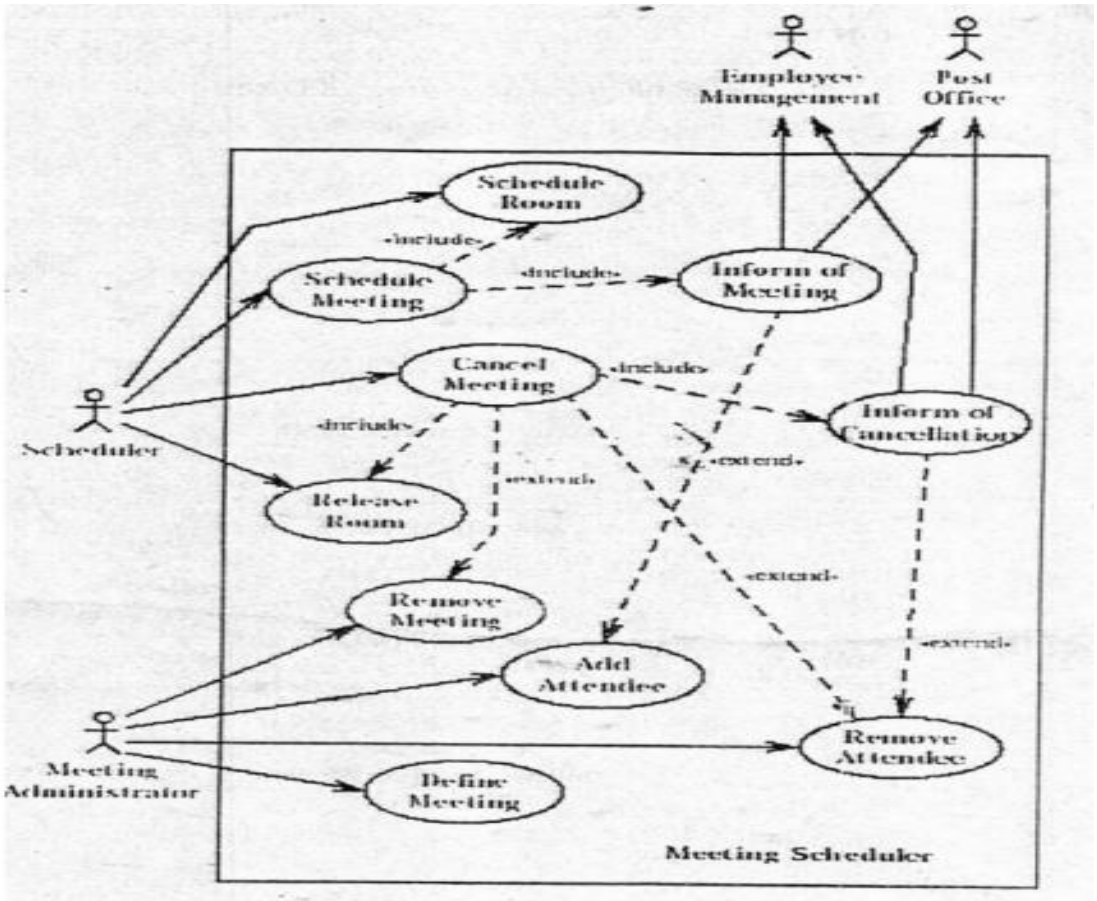
Bài 2. Vẽ lại qui trình nghiệp vụ sau:

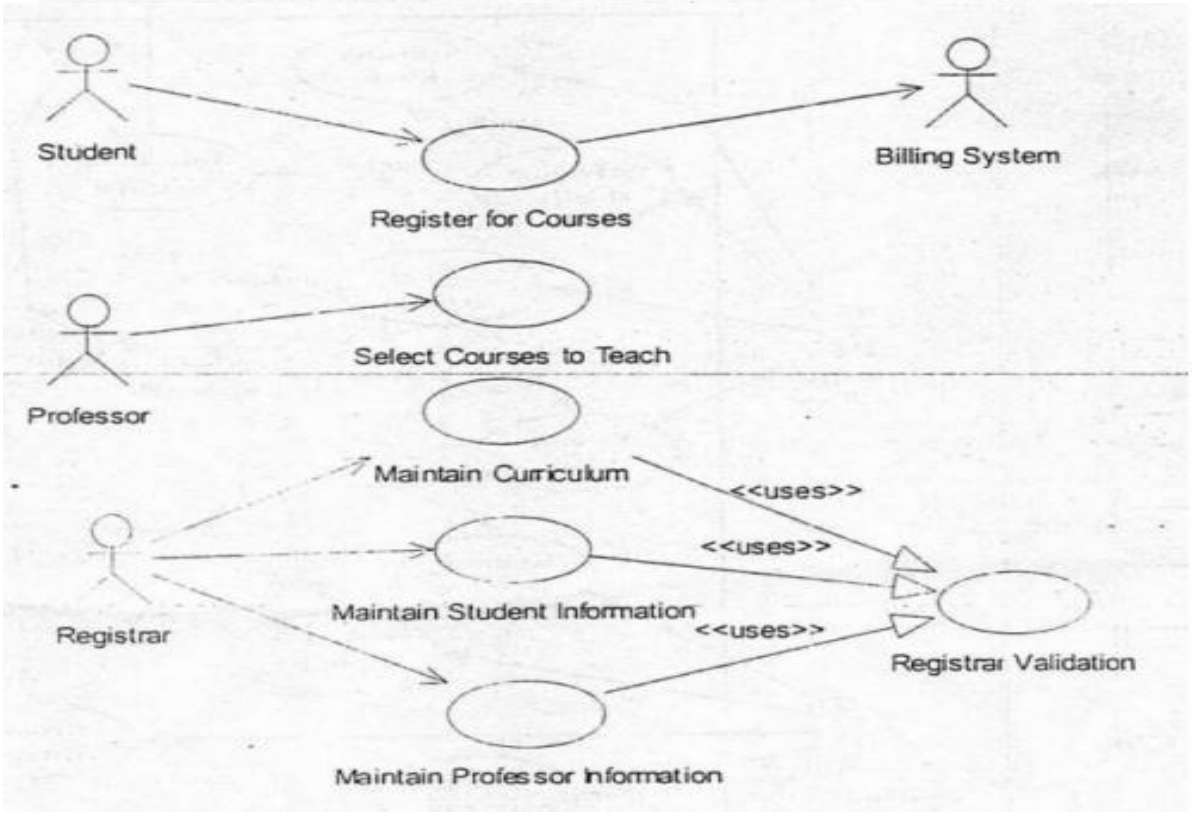


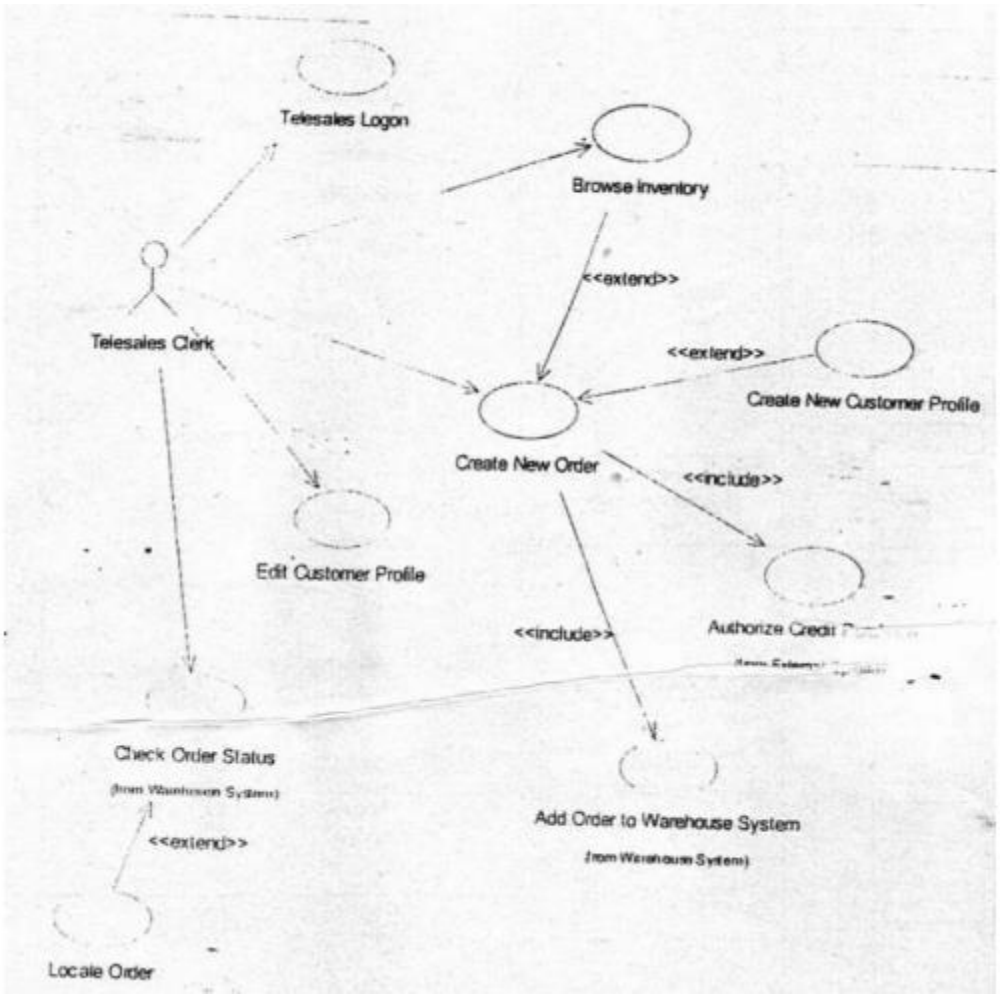
Xác định yêu cầu nghiệp vụ, user stories, và các use cases.

Bài 3. Vẽ lại qui trình nghiệp vụ:

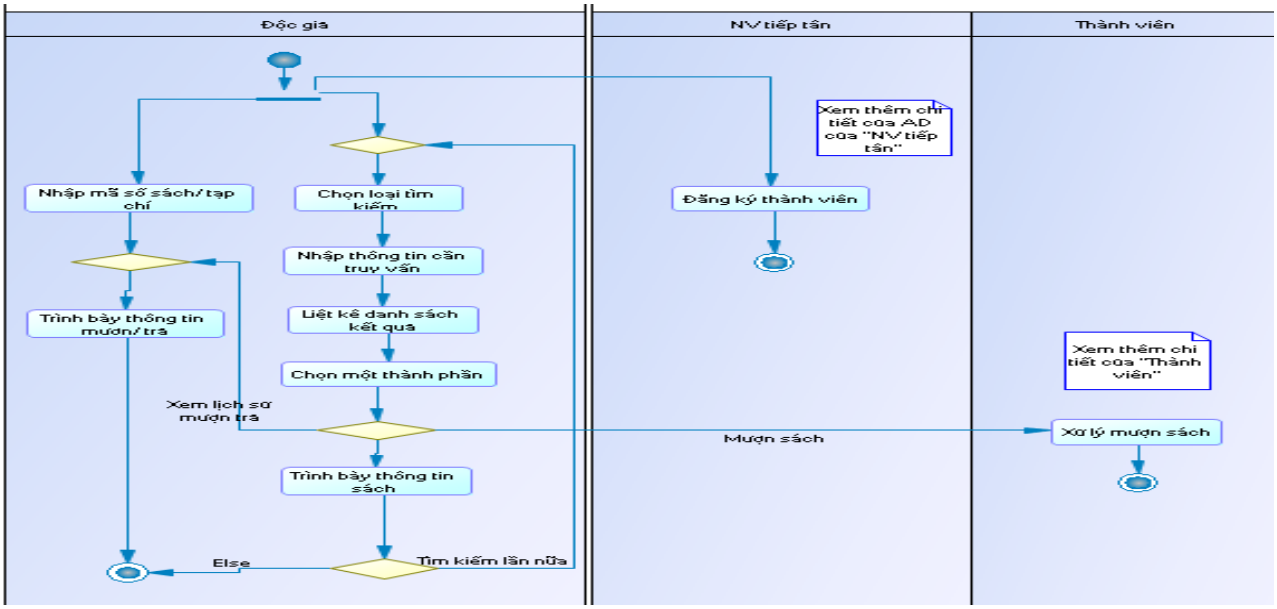



Employee Management
Post Office
Schedule Room
«include»
Schedule Meeting
«include»
Inform of Meeting
Cancel Meeting
«include»
Inform of Cancellation
Scheduler
«include»
«extend»
Release Room
«extend»
«extend»
Remove Meeting
Add Attendee
«extend»
Meeting Administrator
Remove Attendee
Define Meeting
Meeting Scheduler


Student
Billing System
Register for Courses
Select Courses to Teach
Professor
Maintain Curriculum
<<uses>>
<<uses>>
Registrar
Maintain Student Information
<<uses>>
Registrar Validation
Maintain Professor Information


Telesales Logon
Browse Inventory
Telesales Clerk
<<extend>>
<<extend>>
Create New Customer Profile
Create New Order
<<include>>
Edit Customer Profile
<<include>>
Check Order Status
<<extend>>
Add Order to Warehouse System
Locate Order

Bài 1 :


Độc giả
NV tiếp tân
Thành viên
Xem thêm chi tiết của AD của "NV tiếp tân"
Nhập mã số sách/ tạp chí
Chọn loại tìm kiếm
Đăng ký thành viên
Nhập thông tin cần truy vấn
Trình bày thông tin mượn/ trả
Liệt kê danh sách kết quả
Xem thêm chi tiết của "Thành viên"
Chọn một thành phần
Xem lịch sử mượn trả
Mượn sách
Xử lý mượn sách
Trình bày thông tin sách
Else
Tìm kiếm lần nữa

Yêu cầu nghiệp vụ : Quản lý thư viện

 Tìm kiếm tài liệu (sách/tạp chí)

- Người dùng có thể nhập mã số sách/tạp chí hoặc chọn loại tìm kiếm.
- Hệ thống yêu cầu nhập thông tin tìm kiếm.
- Hệ thống hiển thị danh sách kết quả.
- Người dùng có thể chọn một mục trong danh sách.
- Hiển thị thông tin sách hoặc lịch sử mượn/trả nếu có.

 Đăng ký thành viên

- Độc giả có thể đăng ký thành viên.
- Thông tin đăng ký sẽ được xử lý bởi hệ thống.

 Mượn và trả sách

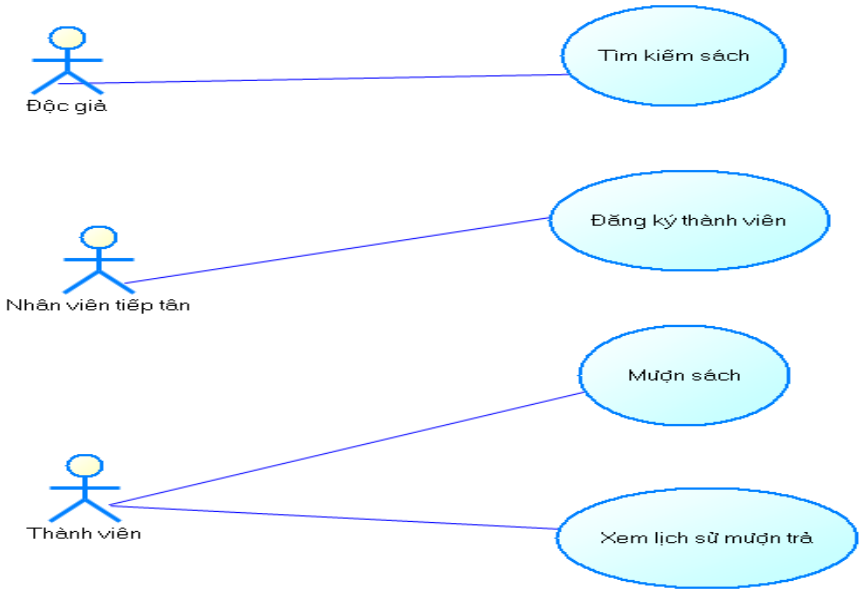
- Thành viên có thể mượn sách.
- Hệ thống xử lý yêu cầu mượn sách.
- Nhân viên tiếp tân có thể xem thông tin chi tiết về thành viên.

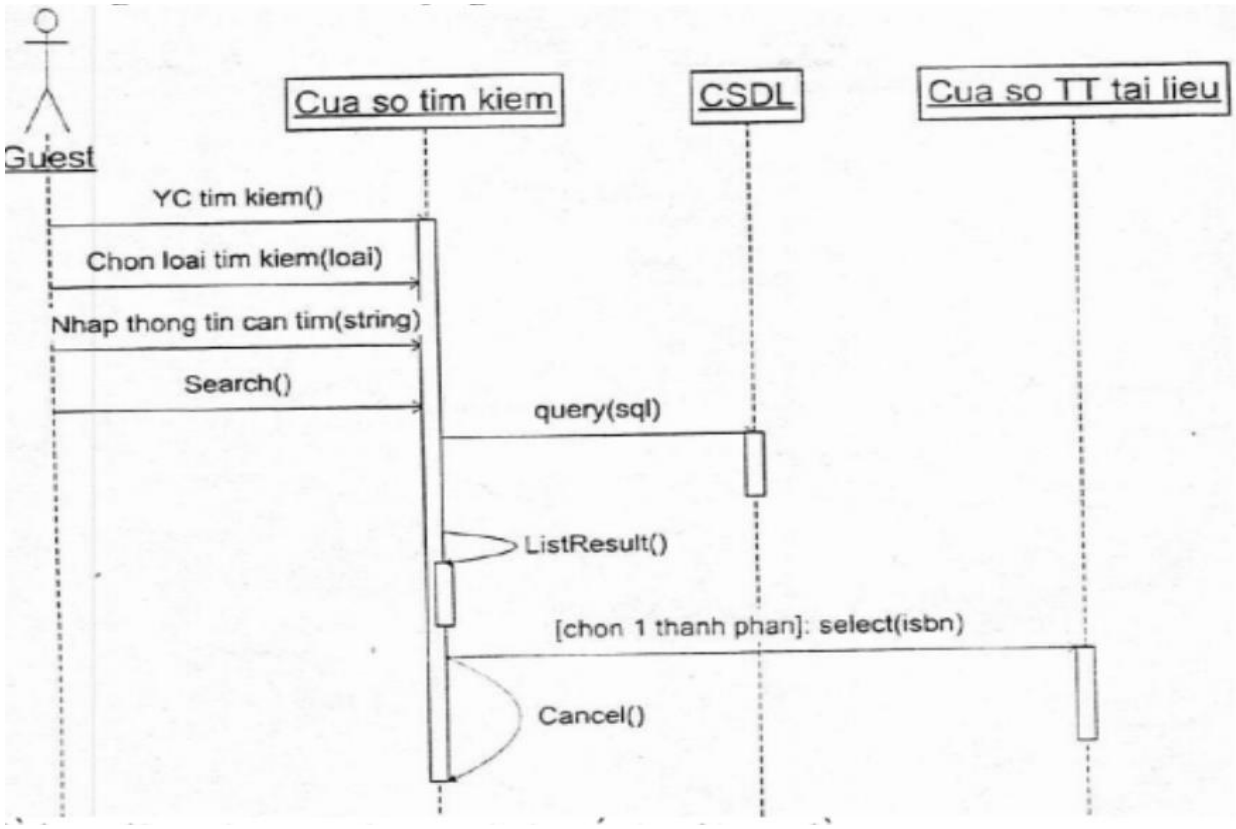
User stories

EPIC	Story ID	User story
Tìm kiếm sách	B_01	Tìm kiếm theo sách theo mã Xem chi tiết sách Tìm kiếm sách theo từ khóa
Đăng ký thành viên	B_02	Điền thông tin đăng ký Xác nhận tài khoản

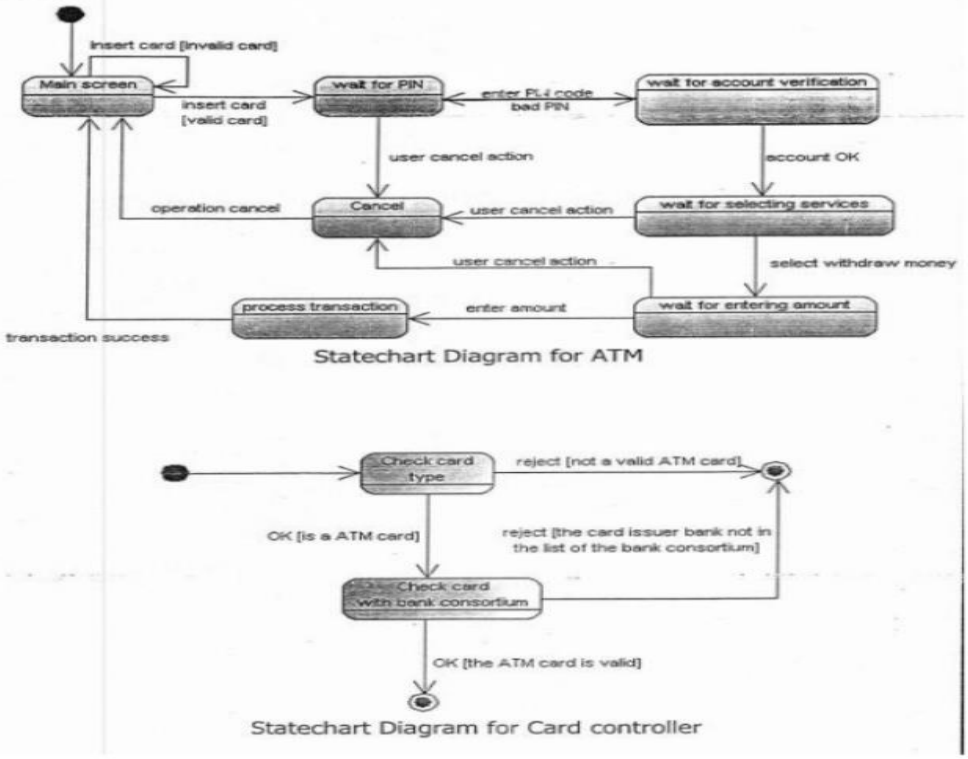
		Duyệt đăng ký thành viên
Mượn trả sách	B_03	Mượn sách từ thư viện Trả sách đã mượn Xem lịch sử mượn trả Xem thông tin thành viên

Use case

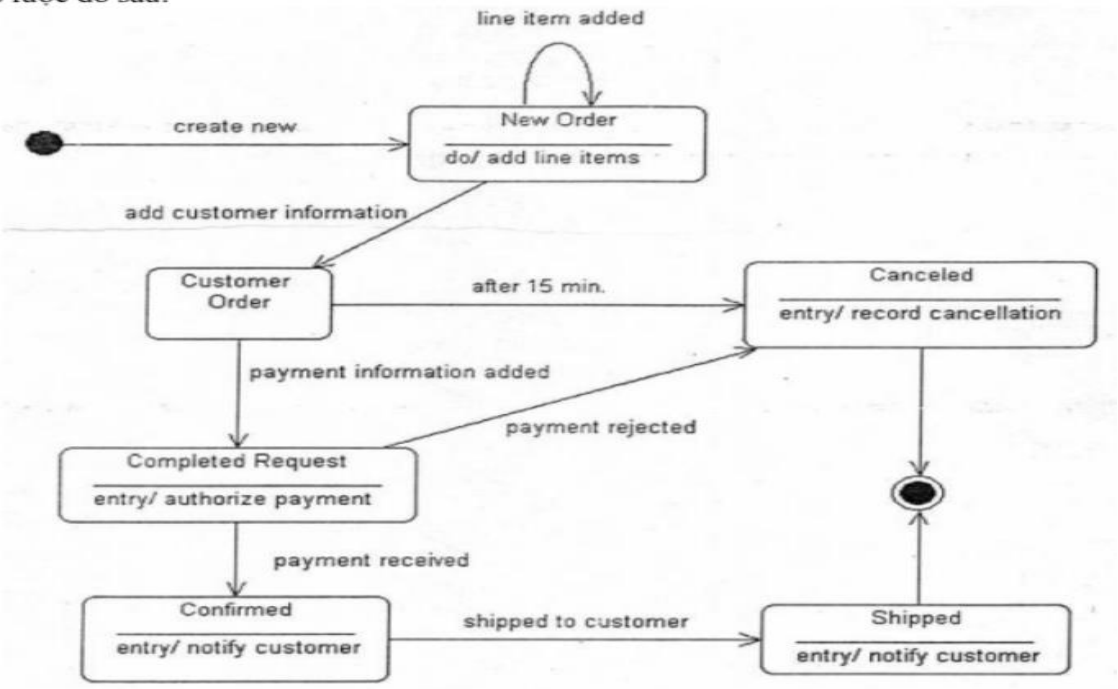


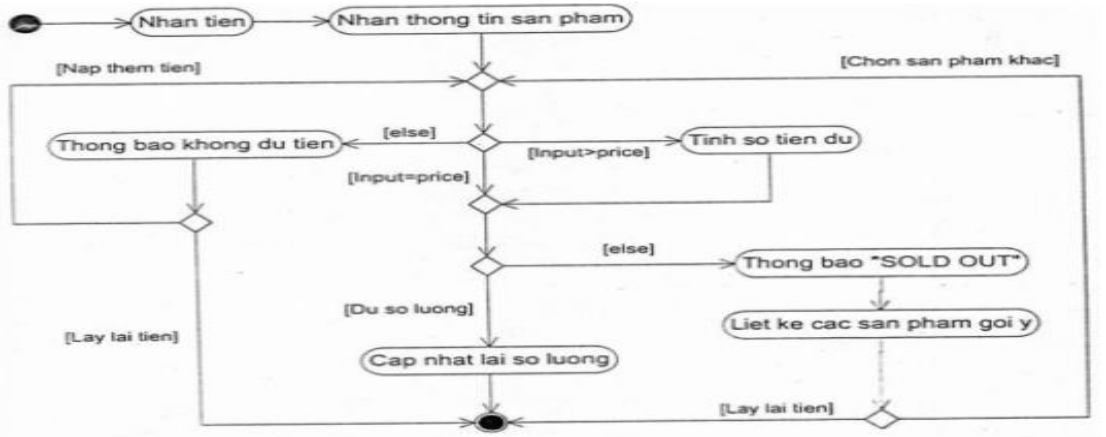


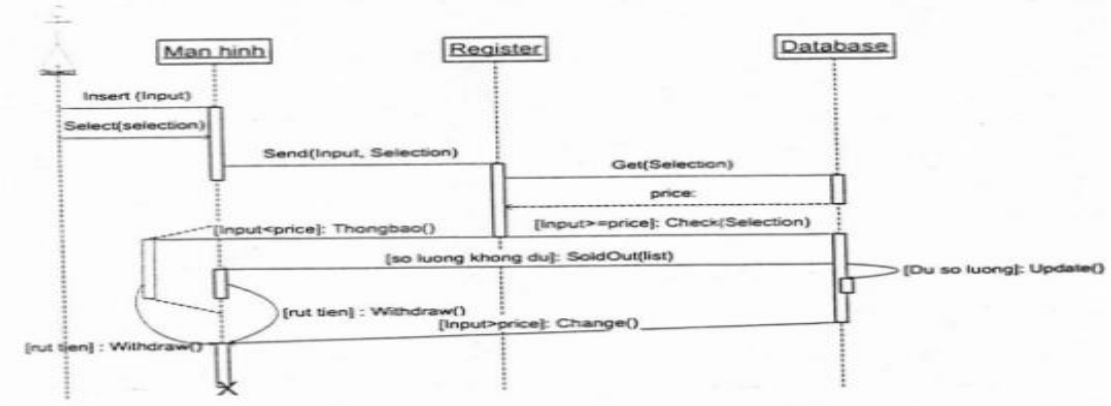
Bài 2. Vẽ lược đồ sau:

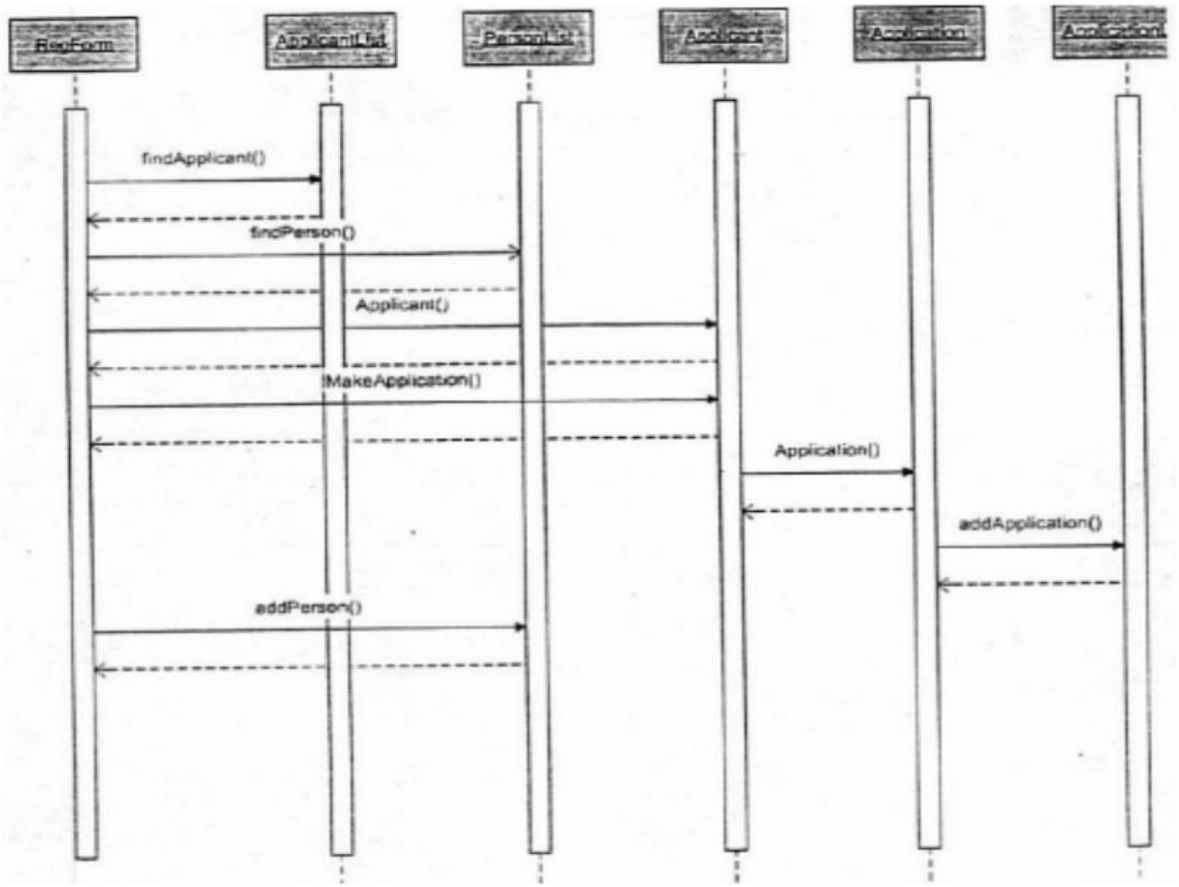


Bài 3. Vẽ lược đồ sau:





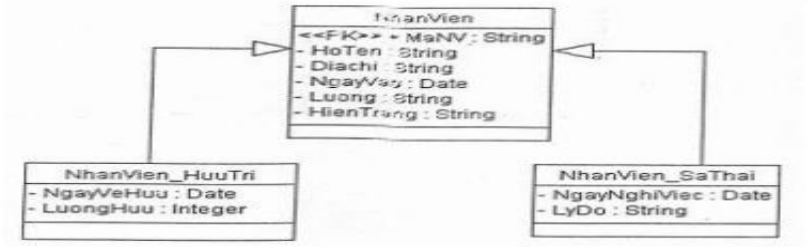


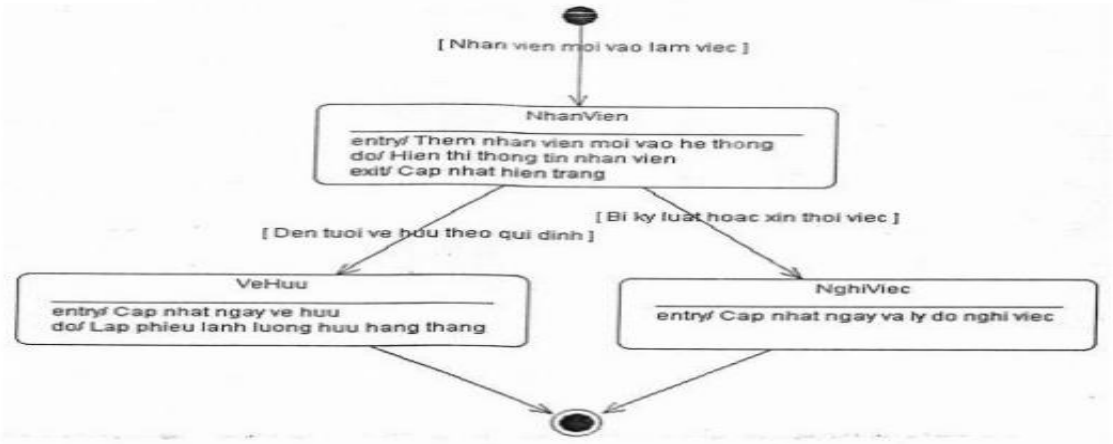


MÔ HÌNH HÓA YÊU CẦU: STATE-DEPEDENT DIAGRAMS

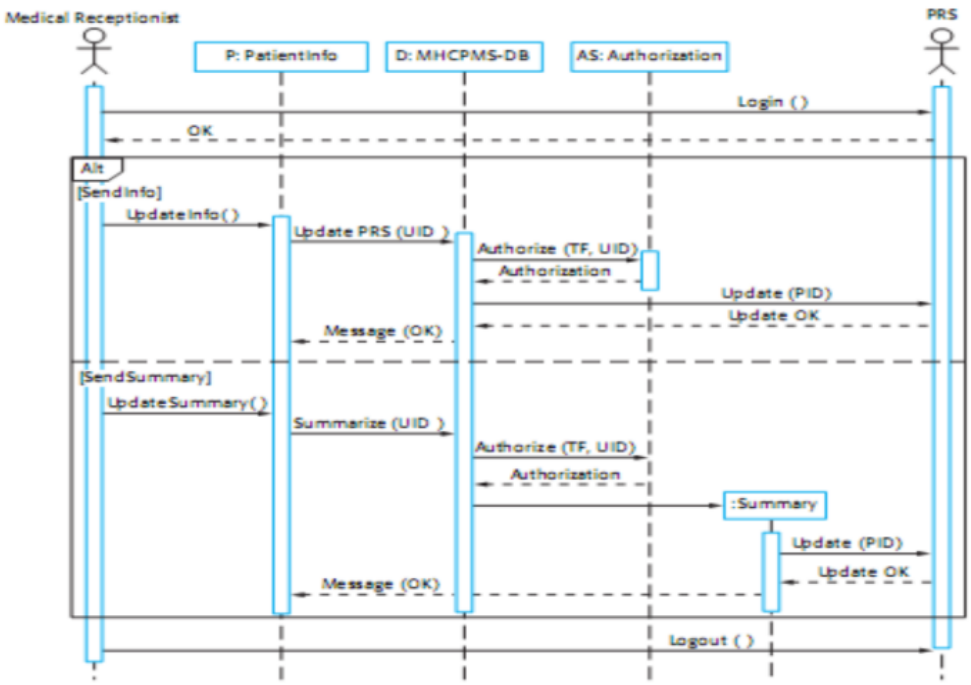
---o0o---

Bài 1. Vẽ lược đồ sau:





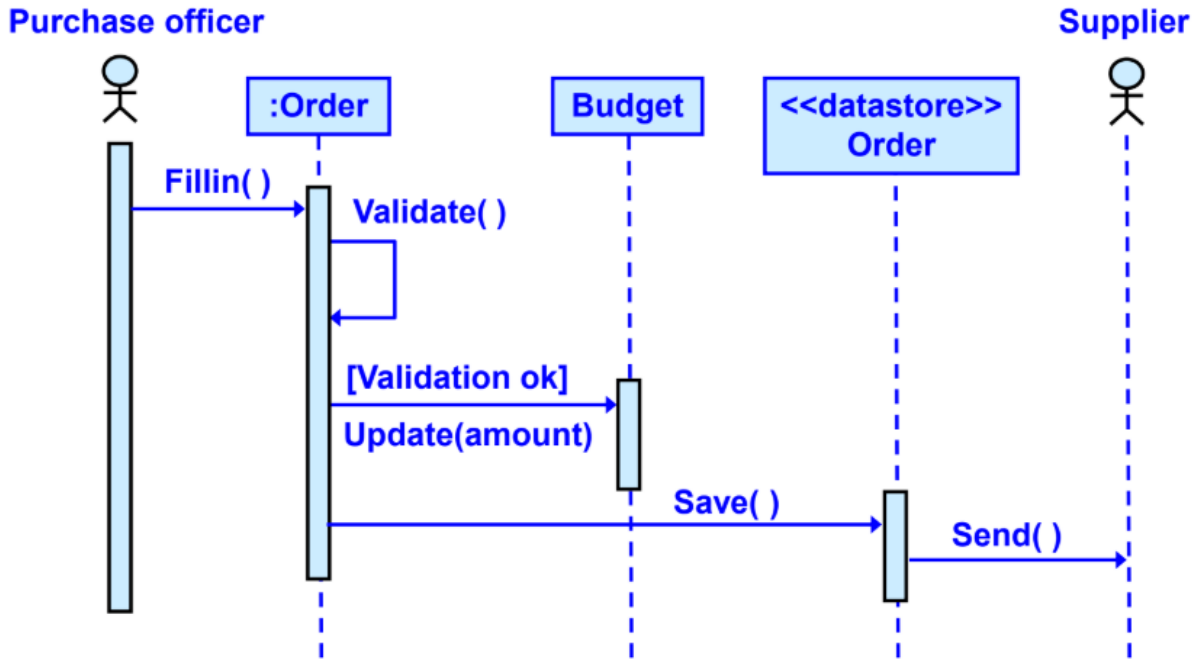
Sơ đồ tuần tự Transfer Data



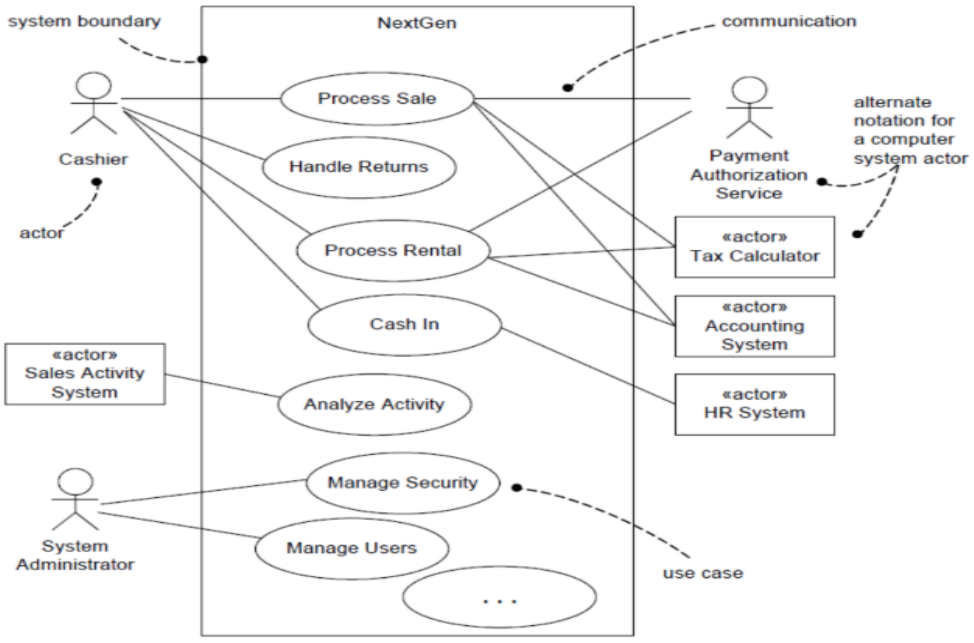
Lớp Consultation

Consultation
Doctors Date Time Clinic Reason Medication prescribes Treatment prescribed Voice notes Transcript …
New( ) Prescribed( ) RecordNotes( ) Transcribed( ) …

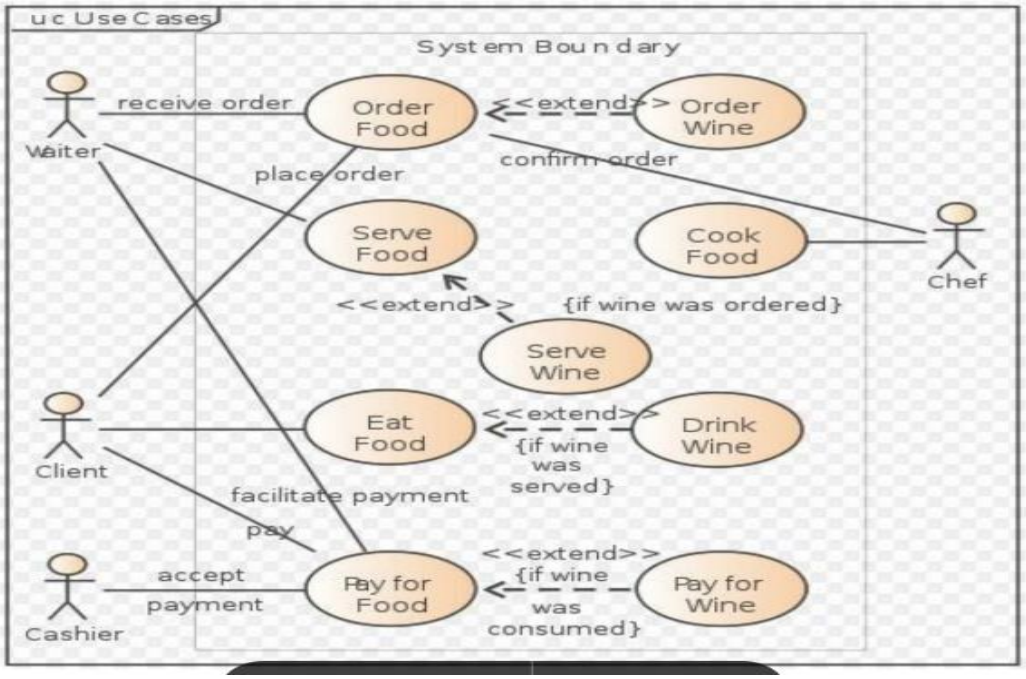
Xử lý đơn hàng



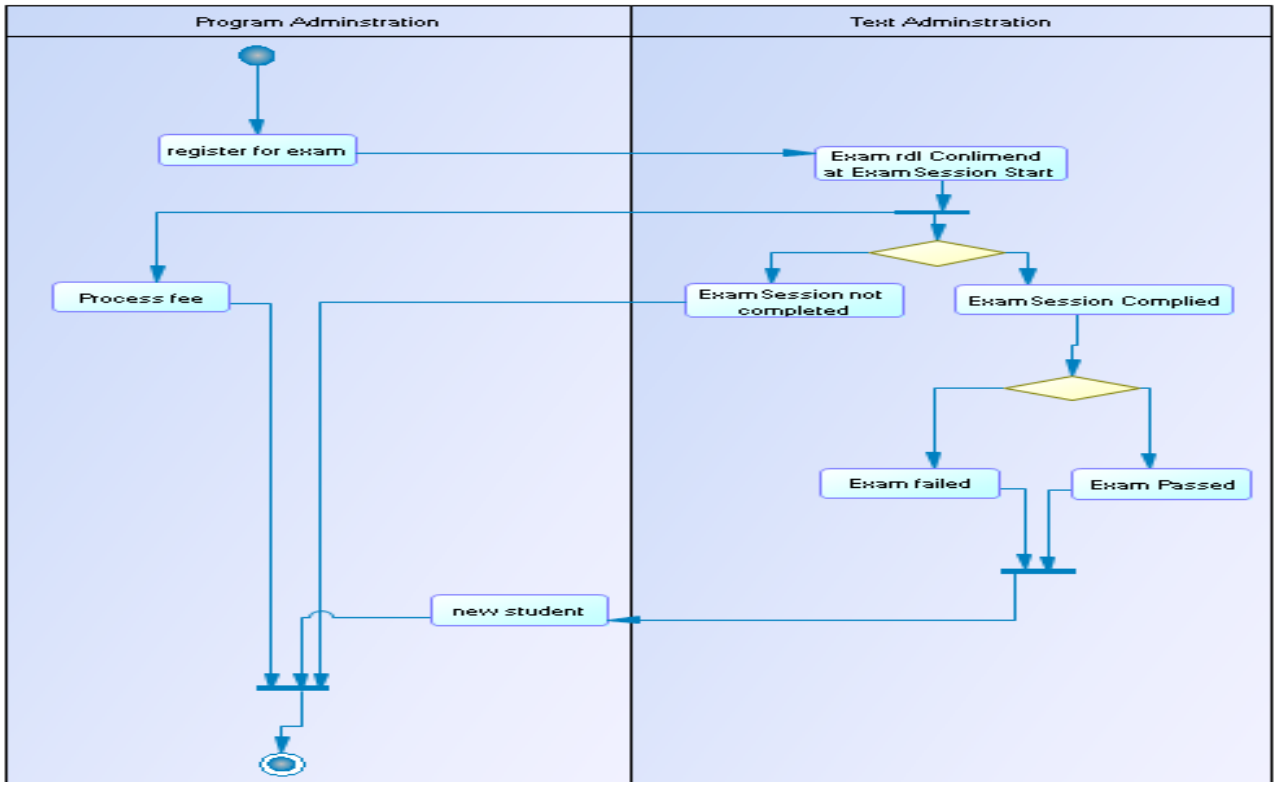
Một minh họa về UC



Ví dụ 4


uc Use Cases
System Boundary
Waiter
receive order
Order Food
<<extend>>
Order Wine
confirm order
place order
Serve Food
Cook Food
Chef
<<extend>>
{if wine was ordered}
Serve Wine
Client
Eat Food
<<extend>>
Drink Wine
{if wine was served}
facilitate payment
pay
Cashier
accept payment
Pay for Food
<<extend>>
{if wine was consumed}
Pay for Wine

Bài 3


Program Adminstration
Text Adminstration
register for exam
Exam rdl Conlimend at Exam Session Start
Process fee
Exam Session not completed
Exam Session Complied
Exam failed
Exam Passed
new student

Yêu cầu nghiệp vụ : hệ thống chứng nhận trọng tài

 Đăng ký kỳ thi:

- Sinh viên có thể đăng ký thi nếu số lần trượt < 3.
- Khi đăng ký thành công, hệ thống tạo một phiên thi.

 Xác nhận danh sách thí sinh:

- Khi kỳ thi bắt đầu, hệ thống xác nhận danh sách thí sinh có mặt.

 Tiến hành kỳ thi:

- Nếu sinh viên hoàn thành bài thi, hệ thống sẽ chấm điểm.
- Nếu không hoàn thành, hệ thống thông báo thất bại.

 Xử lý kết quả:

- Nếu thi đậu → cập nhật trạng thái chứng nhận.
- Nếu thi rớt → ghi nhận số lần trượt và thông báo sinh viên.

 Thanh toán lệ phí thi:

- Sinh viên phải đóng lệ phí trước khi hoàn tất đăng ký.

User story

EPIC	Story ID	User story
Thi	B_21	Là một sinh viên, tôi muốn đăng ký kỳ thi nếu số lần trượt dưới 3.

		Là một sinh viên, tôi muốn đóng lệ phí thi để hoàn tất đăng ký. Là một sinh viên, tôi muốn biết thông tin về kỳ thi sau khi đăng ký. Là một sinh viên, tôi muốn tham gia kỳ thi khi đến thời gian quy định Là một sinh viên, tôi muốn biết kết quả bài thi sau khi hoàn tất.. Là một sinh viên, tôi muốn nhận thông báo nếu tôi rớt kỳ thi.