**COMP 3700**

**Software Modeling and Design**

**Homework 4**

**Due: Tuesday, February 19th, 2019 by 11:59PM (midnight)**

Although this homework relates to your group project, it is an INDIVIDUAL assignment. Your task is to initiate the inception process and start exploring the functional requirements of your project.

**Note:** 1. Association multiplicity for \* cannot properly be modeled in ArgoUml; instead it has been written as 0..\* or 1..\* (permission granted by professor).

2. Constraints have been written under the third bullet point for question 1 instead of being added to the model, as ArgoUML is buggy with accepting OCL syntax (permission granted by professor).

3. Constraints have been made for every attribute and every operation for the class diagram.

4. There are 5 main use cases for the Use Case Diagram, and other important ones have been either included or extended from the others to model every possible use case within the design.

5. ArgoUML does not have any way to show qualifiers; the words next to the association ends are implied to be the qualifiers for each association.

**Table of Contents:**

* Question 1 is answered on pages 2 – 5.
* Question 2 is answered on page 6.
* Question 3 is answered on page 7.
* Question 4 is answered on pages 8 – 9, with the sequence diagrams on page 10 and 11, respectively.
* The reference page is included on page 12.

(1- 25 pts) In this problem, you will transform a concept statement into a formal domain model and

develop constraints for its structure:

* Develop a one-page **concept statement**.

We are going to design a software program that allows any user within a medical facility to communicate with each other or any patients in real time to help further aid the medical system. This application will be designed for all sorts of medical staff, but also provides a UI for patients to help communicate with the medical staff. The medical staff that will be using the software includes, but is not limited to, Doctors, nurses, secretaries, pharmacists, lab testers, radiologists, the hospital billing department, and an admin. Others that will use the system includes new patients and current patients. Staff members are identified by name, email, phone number, employee ID, job title, and SSN. Patients are identified by name, email, phone number, and SSN.

The system will handle communication between all these entities to allow easier management and accessibility of information for everyone. It will also solve the problem of patient communication, so that patients can more easily schedule an appointment, receive necessary medication, and access important medical information about themselves. This is important for providing a seamless form of accessibility towards all staff members as well, as they will now be able to provide instant information and permissions towards other staff members to increase the speeds at which people are treated. This will also help provide more capacity for new patients and allow more medical care to be received for those that need it.

This system will be adopted for most hospitals in America; but can be designed and translated for use in other countries. It will help strengthen the medical care system universally so that more people will be treated at faster rates. This is not restricted to only being used in standard hospitals; it can be used in a variety of medical facilities. This means it can be used in a wide variety of hospitals; general hospitals, mental health treatment facilities, smaller hospitals in more urban areas, etc. If the medical facilities have computers and accessibility to good internet speeds, it can be implemented.

This system will be needed when any staff member desires to communicate with another, i.e., providing a good form of communication between departments. As an example, when lab testing completes a study of a new drug that was requested by a different department, the desired departments can receive this information (such as pharmacy, radiology, etc.). This is an example of how it can be used for extremely important data communication, but it can also be used for more trivial tasks. For another example, a new/current patient can now schedule an appointment, request medication refills be sent via the mail, or view medical information without the hassle of driving to a facility. The UI will allow any user, medical staff or not, to login/logout, change the password to their account (provided that the admin is contacted when medical staff attempt this), or to make/delete an account (again, provided admin permissions are necessary on the staff end of this). For staff members, it will allow referrals to be made, appointments to be scheduled (and arrival/departure confirmed within the system), or for new patients to be added to the system.

This system is needed for daily usage so that the time spent in communication can be as minimal as possible, as time is an important resource in the medical community. This will help improve the ability we have of aiding people during crisis situations. The medical centers that implement this program will have an easier time with accepting and referring new patients. It is especially important for any medical facility because health is a crisis; people need medication and procedures in order to stay alive in many parts of the world. The most important design criteria are that this system be implemented with a different UI for both staff and patients. It will work by allowing patients to contact medical care facilities in both crisis and non-crisis situations. It will work for the staff by allowing them to send data to each other and to patients. All departments of any facility will have an easier time with accepting patients, increasing speeds with which aid is provided, and issuing/receiving payments when medical aid is completed.

* Transform the concept statement into a **conceptual domain model** defined in terms of a **UML**

**class diagram** that specifies the classes (problem domain concepts), relevant attributes, associations (with role names, multiplicity, qualifiers), and specializations/generalizations. To develop the domain model from the concept statement, perform **linguistic analysis** to identify model components and explicitly indicate which components are eliminated and for what reason.

**Good classes:**

Information

Appointment

Prescription

Account

Test

Result

Staff

Patient

Medical Facility

Bank

Bank Account

Billing Department

Payment

**Bad classes:**

Redundant: Change password, access important medical information, Make new patient

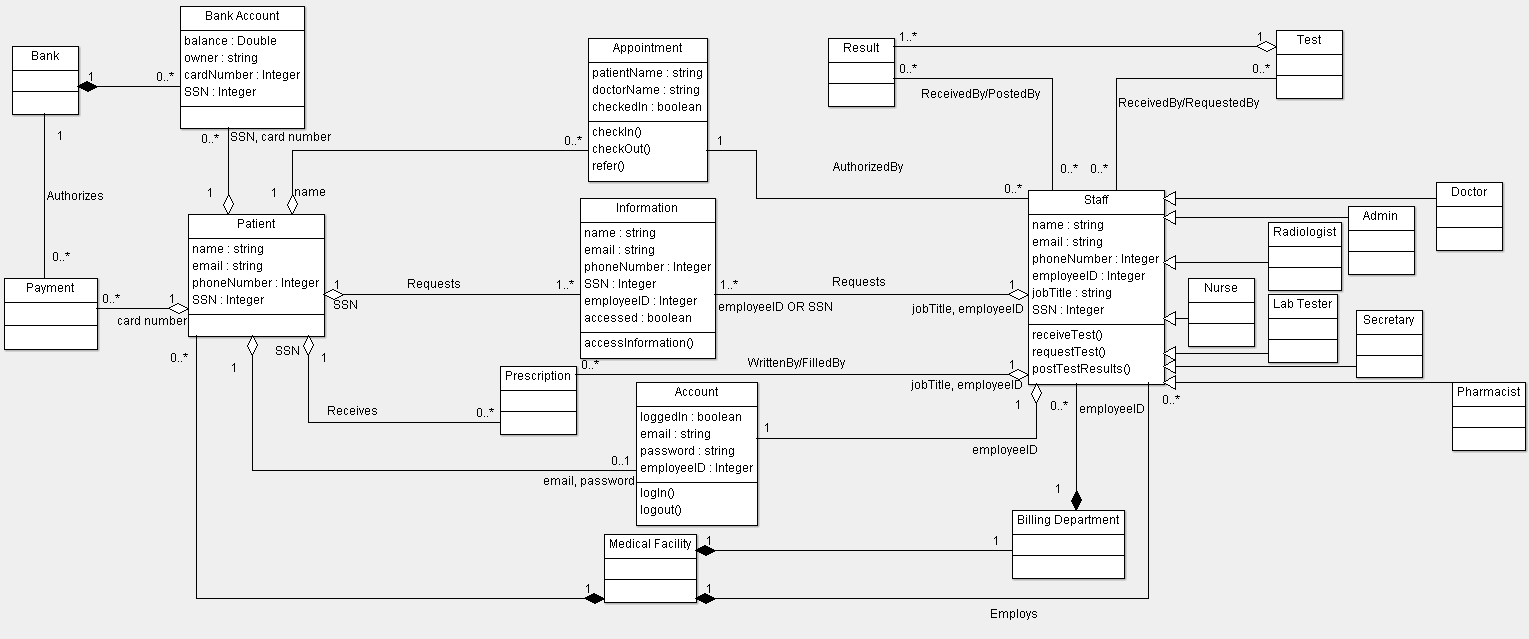
Vague: System

Irrelevant: Permissions

Attribute: Name, Email Address, Phone Number, Employee ID, Job Title, SSN, Card Number

Operations: Request Test, Receive Test Request, Post Test Results, Write Prescription, Fill Prescription, Make/Delete Account, Access Information, Check In/Out, Refer, Log In, Log Out

Roles: Lab Tester, Pharmacist, Doctor, Nurse, Secretary, Admin, Radiologist, New Patients, Current Patients

Implementation: Software, UI

* Define three **constraints/invariants** over the components of the domain model and express them in terms of the **UML Object Constraint Language** (OCL) – recall the OCL examples/features covered in class. You may do research and choose to apply other OCL features/keywords you think are appropriate.

**Classes:**

Bank Account:

{owner->notEmpty() AND balance >= 0 AND cardNumber->size() = 16 AND SSN->size() = 9}

Information:

{name->notEmpty() AND email->notEmpty() AND phoneNumber->size() = 10 AND SSN->size() = 9 AND employeeID->size() = 9

Context: Information::accessInformation() : boolean

pre: accessed = false

post: accessed = true, result = true}

Account:

{email->notEmpty() AND password->notEmpty() AND employeeID->size() = 9

Context: Account::login() : boolean

pre: loggedIn = false

post: loggedIn = true, result = true

Context: Account::logout() : boolean

pre: loggedIn = true

post: loggedIn = false, result = true}

Appointment:

{patientName->notEmpty() AND doctorName->notEmpty()

Context: Appointment::checkIn() : boolean

pre: checkedIn = false

post: checkedIn = true, result = true

Context: Appointment::checkout() : Boolean

pre: checkedIn = true

post: checkedIn = false, result = true

Context: Appointment::refer() : string

pre: doctorName->notEmpty()

post: doctorName = “New Doctor Name”}

Patient:

{name->notEmpty() AND email->notEmpty() AND cardNumber->size() = 16 AND SSN->size() = 9}

Staff:

{name->notEmpty() AND email->notEmpty() AND phoneNumber->size() = 10 AND employeeID->size() = 9 AND jobTitle->notEmpty() AND SSN->size() = 9

Context: Staff::receiveTest() : boolean

pre: jobTitle = “Lab Tester”

post: result = true

Context: Staff::requestTest() : Boolean

pre: jobTitle = “Pharmacist” OR jobTitle = “Doctor”

post: result = true

Context: Staff::postTest() : Boolean

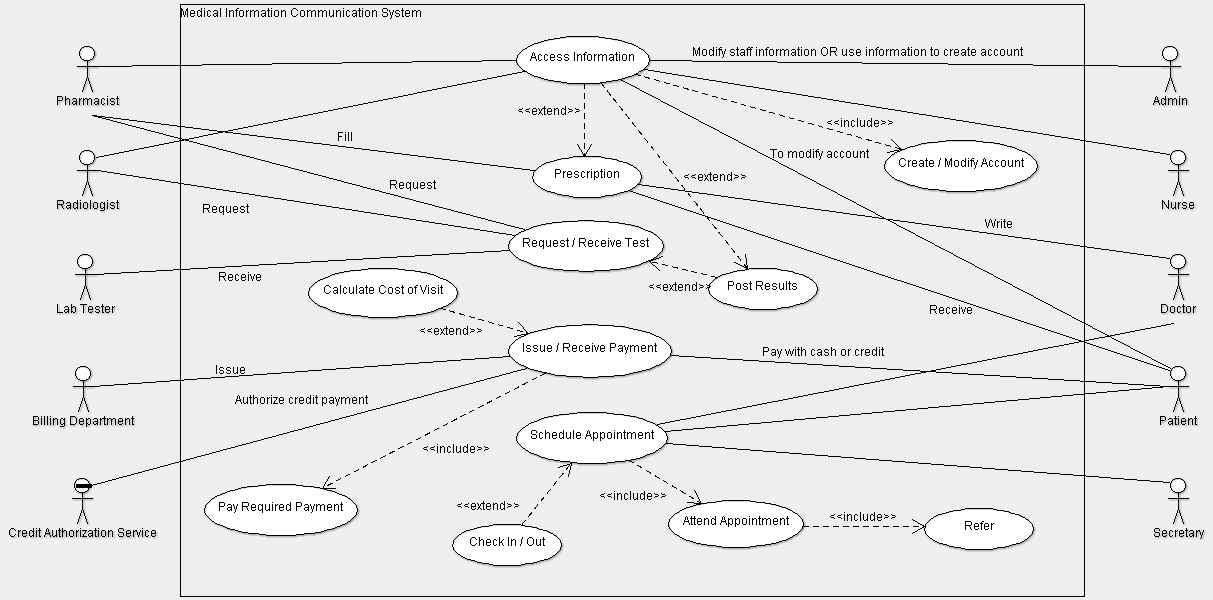
pre: jobTitle = “Lab Tester”

post: result = true}

(2- 25 pts) Draw a **UML Use-Case Diagram** that depicts the boundary of the system, the actors, and

**5 use cases**. You need to present the associations between the actors and the use cases, as well as

(when applicable) the relations (i.e., includes, extends, uses) among the use-cases.



(3- 25 pts) Select a single use-case from part (2) and develop a detailed text description in terms of sequence of actions that the system and the actors perform (**in two-column format**) to yield observable results of value to the actors involved in the use-case. Make sure that each section of the use-case is explicitly represented (i.e., name, list of participants, precondition, flow of events, alternative flow of events, postcondition).

**Use Case: Schedule Appointment**

**Typical Course of Events:**

|  |  |
| --- | --- |
| **Actor Intentions** | **System Responsibility** |
| 1. Patient logs into account to schedule an appointment. | 1. Presents available times / dates. |
| 1. Patient picks an available time / date. | 1. Confirms chosen time / date and saves it to database. |
| 1. Patient arrives at medical facility. |  |
| 1. Secretary checks patient in. | 1. Saves check in time / date to database. |
| 1. Patient finishes appointment and the secretary checks the patient out. | 1. Saves check out time / date to database. |
|  | 1. Billing statement for the appointment is received from the billing department. |
| 1. Patient pays secretary by cash or credit. |  |
| 1. Secretary records payment into system. | 1. If a credit payment, authorizes it. |
|  | 1. Generates receipt and records payment. |
| 1. Patient schedules new appointment. | 1. Presents available times / dates. |
| 1. Secretary confirms new appointment. | 1. Confirms chosen time / date and saves it to database. |
| 1. Secretary gives patient an appointment reminder card to the patient, who then leaves with the reminder card and any prescriptions that were received from the pharmacy. | 1. Sends appointment details to patient’s email address registered to the account and schedules a reminder to be sent 3 days in advance of the appointment. |

**Alternative Courses:**

* Step 1. Patient provides incorrect verification to the system and cannot login. Requests password reset link to be sent to email or phone attached to the account.
* Step 3. Patient cannot schedule an appointment due to an unpaid bill from the last appointment or a non-confirmed email address / phone number attached to the account. Patient must confirm attached information and/or pay the previous bill or cancel scheduling an appointment.
* Step 11. Patient has insufficient cash. Request a credit payment or cancel scheduling until payment is received.
* Step 13. Failure to authorize credit payment, either because of insufficient credit or inactive authorization service. Request cash payment instead.
* Step 15. Appointment time / date is not saved in system until previous appointment bill has been paid.

(4-25 pts) From the selected use-case, identify **two scenarios**, present their textual description, and

specify a **High-Level System Sequence Diagram** (e.g., interactions among actors and the system) for

each scenario to illustrate the sequence and order of events specified in the scenarios.

**Scenario 1: Normal Scenario**

Patient logs in.

System establishes secure communications.

System displays account information.

Patient selects option for scheduling an appointment.

System displays all available times / dates for appointment.

Patient selects available time / date.

System confirms appointment time / date for specified patient.

Patient logs out.

System establishes insecure communication.

System displays good-bye screen.

System sends appointment details to Patient’s email address that is attached to the account.

System generates reminder to be sent to Patient’s email address 3 days before the appointment date.

Patient visits medical center and checks in with Secretary.

System confirms check in and saves check in time / date to database.

Patient finishes appointment and checks out with Secretary.

System confirms check out and saves check out time / date to database.

System displays billing statement for the appointment to the Secretary.

Secretary requests cash or credit payment for appointment.

Patient pays secretary in credit.

Credit Authorization Service confirms payment for appointment.

System generates receipt and records payment in the database.

Patient schedules new appointment with Secretary.

System displays all available times / dates for appointment.

Patient selects available time / date.

System confirms appointment time / date for specified patient.

Secretary gives appointment reminder card to patient.

System sends appointment details to Patient’s email address that is attached to the account.

System generates reminder to be sent to Patient’s email address 3 days before the appointment date.

Patient leaves the medical center.

**Scenario 2: Patient Cannot Pay Bill**

Patient logs in.

System establishes secure communications.

System displays account information.

Patient selects option for scheduling an appointment.

System displays all available times / dates for appointment.

Patient selects available time / date.

System confirms appointment time / date for specified patient.

Patient logs out.

System establishes insecure communication.

System displays good-bye screen.

System sends appointment details to Patient’s email address that is attached to the account.

System generates reminder to be sent to Patient’s email address 3 days before the appointment date.

Patient visits medical center and checks in with Secretary.

System confirms check in and saves check in time / date to database.

Patient finishes appointment and checks out with Secretary.

System confirms check out and saves check out time / date to database.

System displays billing statement for the appointment to the Secretary.

Secretary requests cash or credit payment for appointment.

Patient pays secretary in credit.

Credit Authorization Service denies payment for appointment.

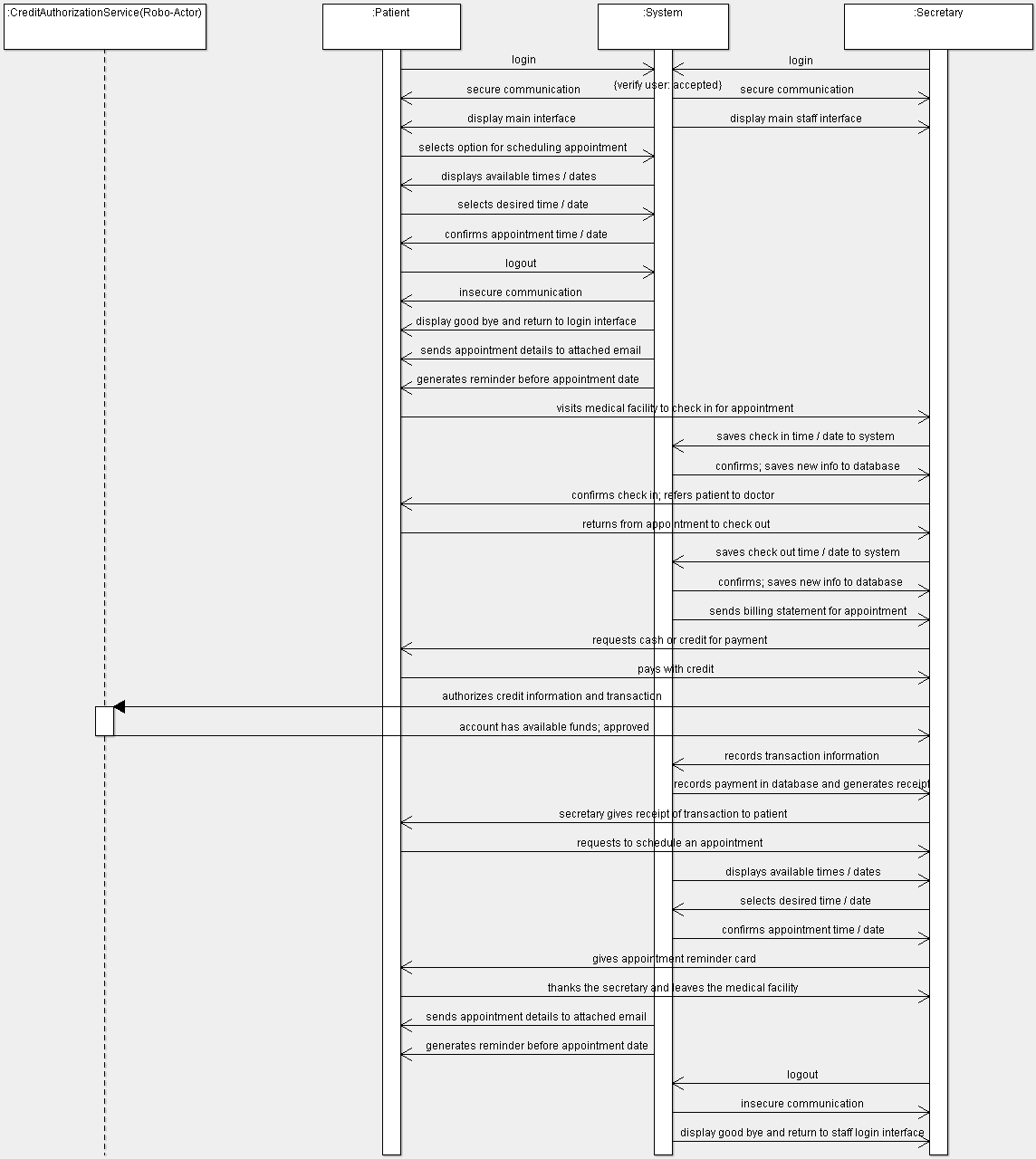
Secretary informs Patient that credit was declined, and requests cash instead.

Patient has insufficient funds in cash to pay bill.

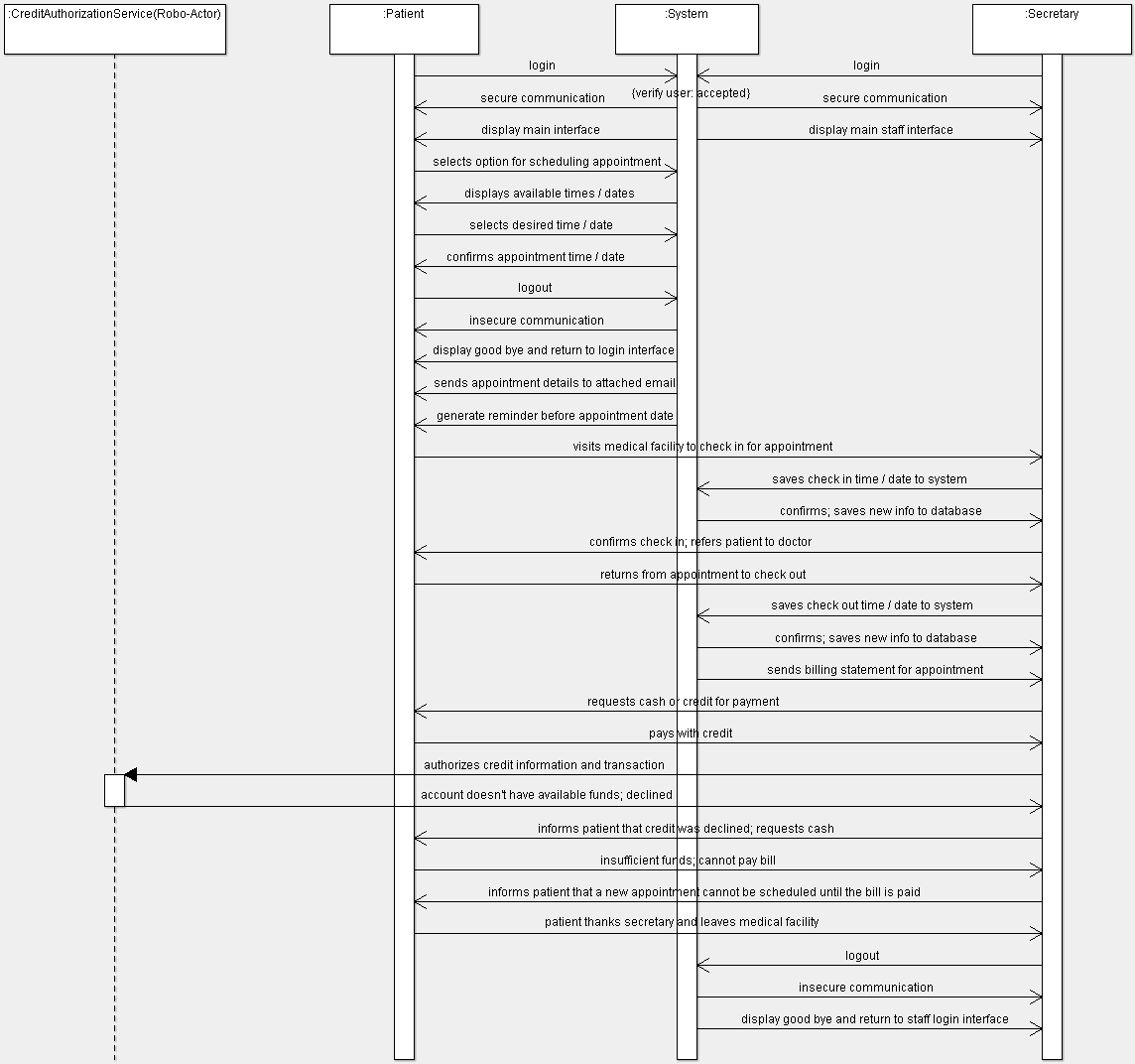
Secretary informs Patient that a new appointment cannot be successfully scheduled until the previous bill is paid either through the system or in person.

Patient leaves the medical center.

**Scenario 1:**



**Scenario 2:**



**References:**

1. *Computer Association Lysator*, [www.lysator.liu.se/xenofarm/argouml/work/argouml/www/documentation/defaulthtml/manual/ch12s07.html](http://www.lysator.liu.se/xenofarm/argouml/work/argouml/www/documentation/defaulthtml/manual/ch12s07.html).
2. “The Object Constraint Language.” *Sample: The Object Constraint Language*, csci.csusb.edu/dick/samples/ocl.html.