Carleton University

cuCare

Requirements Analysis Document

Team: The Four Puppeteers
Sergey Vershinin (Team Lead)
David Carson
Devin Denis
Mike Yuill

Submitted to:
Dr. Christine Laurendeau
COMP 3004 Object-Oriented Software Engineering
School of Computer Science
Carleton University

FINAL
2012-10-18

Table of Contents

Table of Figures

1 Introduction

The Four Puppeteers team is proud to present the cuCare requirements analysis document for your review. Below you will find a detailed response to the system description you provided, including a comprehensive analysis the functionality and workflow of the cuCare system. Also included, as per your instructions, are the selected elements of the system's object and dynamic models.

1.1 Purpose of System

As gateways into the health care system, medical clinics providing primary physician care are vital to ensuring the best possible patient health outcomes. A present day medical practice faces the dual pressures of dealing with a complex web of health care providers, including a multitude of specialists and diagnostics professionals, as well as a rapidly aging population.

Although this is true across the board, older patients, in addition to having a higher likelihood of complicated health issues, are more likely to require assistance in dealing with the complexities of the modern health care system. This assistance is vital as missed follow-up appointments or failures to renew medical prescriptions have the potential of turning what seems to be a minor oversight into a fatal tragedy.

The cuCare system proposed herein aims to provide a medical practice with the means of leveraging modern information technology to streamline record management, provide value-added services to its clients, and ultimately assist the practice with improving patient health outcomes.

Health records contain sensitive information and require access control as per provincial and federal regulations. The proposed system takes this into account and is designed to be used by both physicians and administrative personnel. While physicians are provided complete access to all health records within the system, administrative personnel are limited in their ability to alter records in specific instances so as to ensure health record integrity.

The full range of health record types is supported by the system in an integrated manner, including patient profiles, as well as consultation and follow-up records. This approach to health record management provides a number of benefits:

1. Electronic storage of patient records makes it easy to access and update patient information as required
2. Keeping all of the patient's medical records in one place enables physicians to easily access the patient's medical history, and
3. Tracking post-consultation follow-up requests enables the practice to provide timely reminders to its patients concerning follow-up care.

In all, the system is designed to improve the flow of information between physicians and their support staff, as well as between the medical practice and its clients.

1.2 Overview of Document

The requirements analysis document consists of five major elements, including:

Functional Requirements

The functional requirements section presents an itemized list of the proposed system's functionality from the user's point of view. In other words, this section provides the list of features offered by the proposed cuCare system.

Non-functional Requirements

The non-functional requirements section provides a listing of considerations and implementation constraints that concern the usability, reliability, performance, and supportability of the proposed system. This includes both expectations the user should have when interacting with the system, system operating environment, and extensibility specifications (e.g. programming language used to develop the system).

Use Case Model

The use case model establishes the scope of the proposed system through the use of abstracted user scenarios (i.e. examples of user workflow elements). This section includes a list of use cases for the entire feature set of the cuCare system, including use cases describing regular workflow, as well as specific error conditions that may be encountered by the user (e.g. entering an invalid username at login).

Object Model

The object model presents an analysis of the entity classes that will store persistent data within the cuCare system. In addition to describing the data stored by the various entity objects, this section also outlines the relationships between the different types of entity objects within the system.

Dynamic Model

The dynamic model presents an analysis of the runtime interactions within the system. The lifetime of entity objects (e.g. a consultation record) is illustrated through the use of state diagrams. This section also includes sequence diagrams for a selected set of use cases, including:

- Creation of a patient consultation record (**CreateConsultation)**
- Editing of a patient's consultation record (**ModifyConsultation**)
- Viewing a list of patients, filtered by follow-up status and by physician (**RunFollowupStatusReport**)
- The execution of the system audit for identification of overdue follow-ups (**RunAudit**)

2 Functional Requirements

This section presents a detailed listing of the features offered by the proposed cuCare system.

Table 1 – Functional Requirements

1	The system provides access to the cuCare client application through the use of **user accounts**
1.a	User accounts contain the following information:
1.a.i	Username (unique)
1.a.ii	First name
1.a.iii	Last name
1.a.iv	Title
1.a.v	Home address
1.a.vi	Contact information
1.b	Users must log in to the system using a valid username to gain access to client application's functionality
1.c	There are three types of user accounts maintained:
1.c.i	System administrator
1.c.ii	Administrative assistant
1.c.iii	Physician

2	A user logged-in as a **system administrator** can:
2.a	Create user accounts for physicians and administrative assistants
2.b	View and modify stored user account information of physicians and administrative assistants
2.c	Remove user accounts of physicians and administrative assistants
2.d	View and modify the time for execution of the daily audit process
2.e	View and modify the client application's setting for the IP and port number of the cuCare server

3	Users logged-in as an **administrative assistant** can:
3.a	Create patient records
3.b	View patient records, including all of their associated consultation and follow-up records
3.c	Modify and remove patient records
3.d	Create consultation records
3.e	View, modify, and cancel consultation records before they have taken place, in the same fields as were editable when the record was created
3.f	Modify a follow-up record by entering information, including test results and specialist diagnoses, detailing the results of a follow-up
3.g	Run reports
3.h	List patients filtered by follow-up status

4	Users logged-in as a **physician** can:

4.a	Perform any action available to an administrative assistant
4.c	View, modify, and remove consultation records, including
4.c.i	Patient diagnosis
4.c.ii	Create new follow-up requests
4.c.iii	Notes – additional follow-up information including test results and specialist diagnoses
4.d	View modify, and cancel follow-up records

5	**Patient records** contain the following information:
5.a	Unique identifier
5.b	Date the patient was added to the system
5.c	First name
5.d	Last name
5.e	Mailing address
5.f	Contact information
5.g	Date of birth
5.h	Health card number
5.i	Health card expiry date
5.j	Primary physician
5.k	Notes – additional patient information

6	**Consultation records** contain the following information:
6.a	Patient name (uniquely identified)
6.a.i	The consultation recorded must be associated with an existing patient record
6.b	Have one of three types of status:
6.b.i	Pending – initial status, set at the creation of the follow-up record
6.b.ii	Completed – set by a physician, pending review of all outstanding follow-up requests
6.b.iii	Cancelled
6.c	Date and time of the consultation
6.d	Reason for the consultation
6.e	Attending physician
6.i	A consultation record must always specify an attending physician
6.f	Patient diagnosis

7	**Follow-up records** contain the following information:
7.a	Have one of four types of status:
7.a.i	Pending – initial status, set at the creation of the follow-up record
7.a.ii	Overdue – set by a scheduled automated audit process
7.a.iii	Received – set by an administrative assistant upon entry of test results or specialist diagnoses
7.a.iv	Completed – set by the physician upon review of test results, specialist

	diagnosis, or issuance of a drug prescription renewal
7.b	Date by which the follow-up is due
7.b.i	If the follow-up status is not updated to received or completed by the due date, the system audit process will automatically change its status to "overdue"
7.c	There are four types of follow-ups
7.c.i	Drug prescription renewal – the due date specifies the date by which the patient should get their prescription renewed
7.c.ii	Medical test – the due date specifies the date by which test results should be received by the doctor's office
7.c.iii	Referral to a specialist – the due date specifies the date by which a specialist diagnosis should be received by the doctor's office
7.c.iv	Return consultation – the due date specifies the date by which the patient should return for a follow-up consultation
7.d	Follow-up records can store large amounts of text for entry of test results and specialist diagnoses

8	Users can run **reports** on the data contained within the system
8.a	Reports that can be run by administrative assistants include:
8.a.i	A ranking of types of follow-up requests that are most often overdue
8.a.ii	A list of patients who have had no consultations for a specified amount of time
8.a.iii	A list of patients with a high number of consultations (the user can specify the minimum number of consultations for inclusion in the report)
8.a.iv	A list of patients with a high number of overdue follow-ups (the user can specify the minimum number of follow-ups for inclusion in the report)
8.a.v	A list of the total number of consultations associated with each billing number
8.b	Reports that can be run by physicians include all those that can be run by administrative assistants, as well as
8.b.i	A list of patients filtered by follow-up status
8.c	All reports can be filtered by primary physician, patient age range, and follow-up type
8.d	Users can save reports to a text file accessible outside of the system (i.e. export reports)

9	The system conducts an automatic, regularly scheduled **audit** of the records stored therein
9.a	The audit process runs each day at a time specified by a system administrator
9.b	The audit process checks all follow-ups which currently have the status 'pending' that are past their due date

3 Non-functional Requirements

This section presents a listing of considerations and constraints for the implementation of the proposed cuCare system.

Table 2 – Non-functional Requirements

1	**Usability**
1.a	The GUI uses the standard Ubuntu visual style and themes
1.b	When the client must wait for a process to be completed, they are presented with a progress bar (i.e. the GUI does not 'hang').
1.c	An average user with no prior experience using the system should be able to learn to perform their main work tasks with one day of instruction
1.d	The GUI must be scalable to all standard resolutions
1.e	The GUI can be viewed without the need for horizontal scrolling at a screen resolution of 1024x768 or higher.
1.f	The same GUI is used for both physicians and administrative assistants. Features which are not available to administrative assistants will appear greyed out.
1.g	Upon installation, a default system administrator account will already exist. Details for logging into that account are provided in the system manual.
1.h	Modifying the client application's cuCare server IP and port number settings does not require a restart of the application to re-establish communication.
1.i	Error messages displayed by the system are descriptive and easily understandable by a non-technical user.

2	**Reliability**
2.a	Server must be able to handle at least 4 client connections simultaneously
2.b	Server must be able to run for at least 48 hours without restart
2.c	Client must be able to run and maintain communication with the server for at least 8 hours without restart

3	**Performance**
3.a	The GUI reacts in real time to all user commands
3.b	In the testing environment (32-bit VM) the system is able to:
3.b.i	Search 10,000 records in 10 seconds
3.b.ii	Create a report based on 10,000 records in less than 30 seconds.

4	**Supportability**
4.a	Designed for use under the Ubuntu 12.04 OS
4.b	Developed in C++ using the Qt Creator SDK
4.c	Changes in the client application's GUI and behaviour do not require changes to the cuCare server application
4.d	Data storage is implemented using a SQLite database, linked-in as a library. Installation of a separate database application is not required
4.e	The system source code and documentation is delivered to the client on a DVD

4 Use Case Model

This section presents a comprehensive set of use cases for the entire feature set of the cuCare system. Each use case represents an abstracted user scenario, generalized from specific examples of a user's interaction with the proposed cuCare system.

Note: To save space in the tables and diagrams in this section, the three user types will be referred to using the following shorthand labels:

Administrative assistant - *AdminAssistant*
Physician - *Physician*
System Administrator - *SysAdmin*

4.1 Use Case Model Overview

This section presents the list of all of cuCare use cases, including a short description for each one.

Identifier	Name	Description
	SYSTEM ACCESS	
UC-100	**LogIn**	User (any type) logs on to the system
UC-110	**LogOut**	User (any type) logs out of the system
	PATIENT MANAGEMENT	
UC-200	**CreatePatient**	AdminAssistant or Physician creates a new patient profile
UC-210	**ViewPatient**	AdminAssistant or Physician views an existing patient record
UC-220	**ModifyPatient**	AdminAssistant or Physician makes changes to an existing patient record
UC-230	**RemovePatient**	AdminAssistant or Physician removes a patient record from the system
	CONSULTATION MANAGEMENT	
UC-300	**CreateConsultation**	AdminAssistant or Physician schedules a new consultation
UC-310	**ViewConsultation**	AdminAssistant or Physician views an existing consultation record
UC-320	**ModifyConsultation**	AdminAssistant or Physician makes changes to an existing consultation record
UC-330	**PhysicianModifyConsultation**	Physician makes changes to an existing consultation record (describes additional functionality available only to Physicians)
UC-340	**CancelConsultation**	AdminAssistant or Physician cancels a consultation
	FOLLOW-UP MANAGEMENT	
UC-400	**CreateFollowup**	Physician creates a follow-up request
UC-410	**ViewFollowup**	AdminAssistant or Physician views a follow-up record

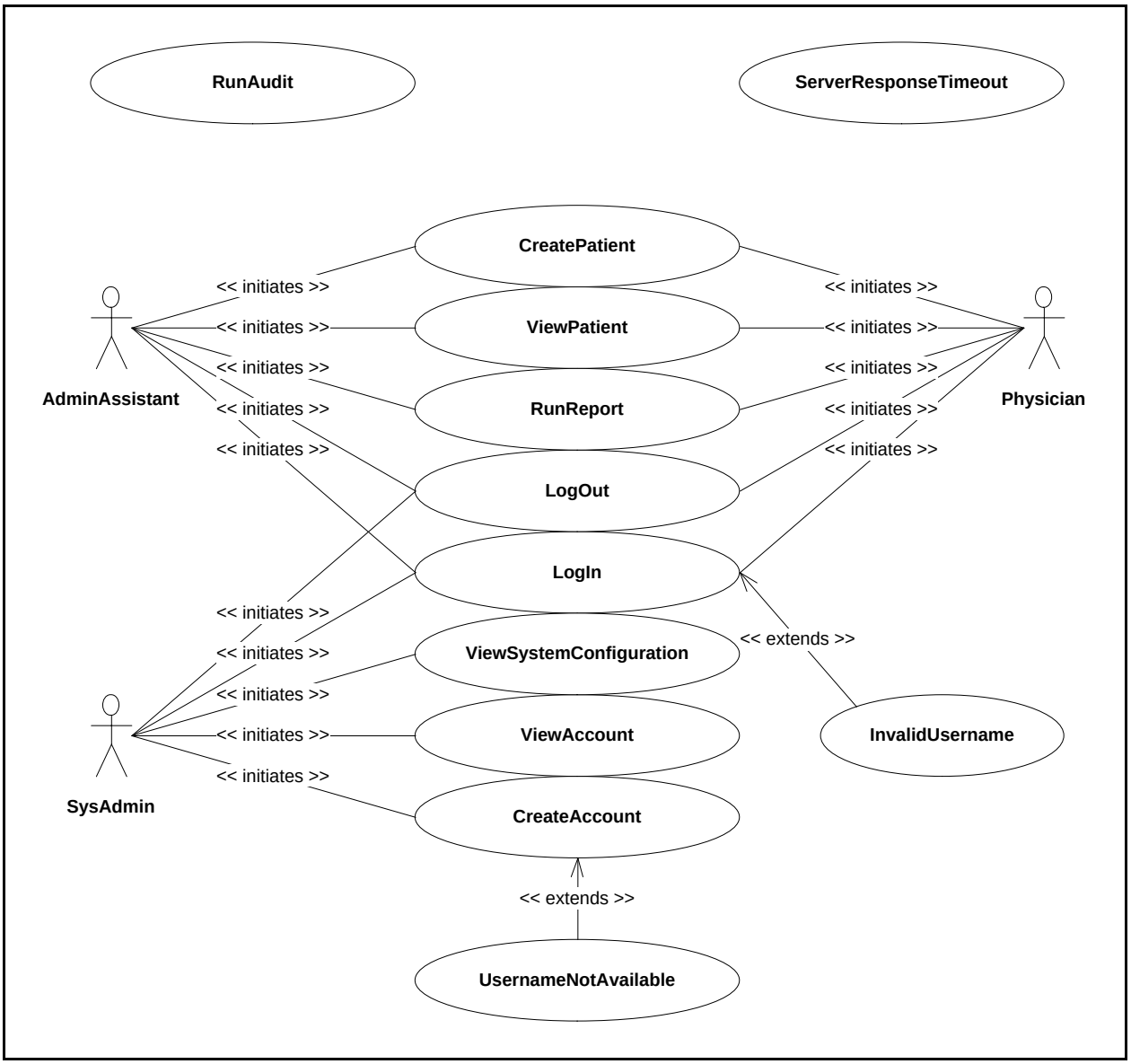
UC-420	**ModifyFollowup**	A general description of how a user would make changes to an existing follow-up record
UC-430	**EnterFollowupResults**	AdminAssistant or Physician enters the results of a medical test or a specialist diagnosis into a follow-up record
UC-440	**PhysicianModifyFollowup**	Physician makes changes to a follow-up record
UC-450	**MarkFollowupComplete**	Physician marks a follow-up record as complete
UC-460	**CancelFollowup**	Physician cancels a follow-up request
	REPORT GENERATION	
UC-500	**RunReport**	AdminAssistant or Physician runs a statistical report
UC-510	**RunFollowupRankingReport**	AdminAssistant or Physician runs a report that provides a ranking of types of follow-up requests that are most often overdue
UC-520	**RunInactivePatientReport**	AdminAssistant or Physician runs a report that lists patients who have had no consultations for a specified amount of time
UC-530	**RunPatientConsultationCountReport**	AdminAssistant or Physician runs a report that lists patients with a high number of consultations
UC-540	**RunOverdueFollowupCountReport**	AdminAssistant or Physician runs a report that lists patients with a high number of overdue follow-ups
UC-550	**RunBillingCounsultationsCountReport**	AdminAssistant or Physician runs a report that lists the total number of consultations associated with each billing number
UC-560	**RunFollowupStatusReport**	Physician runs a report that lists patients filtered by follow-up status
UC-570	**AddReportFilter**	AdminAssistant or Physician applies a filter to a report they are currently viewing
UC-580	**ModifyReportFilter**	AdminAssistant or Physician makes changes to a filter that is being or will be applied to a report they are currently viewing
UC-590	**GenerateReport**	Having selected the relevant report options and applied any of the desired filters, AdminAssistant or Physician generates the desired statistical report
UC-595	**SaveReport**	AdminAssistant or Physician saves a reports they have generated to a text file that can be accessed outside of the cuCare system
	SYSTEM ADMINISTRATION	
UC-600	**CreateAccount**	SysAdmin creates a new user account
UC-610	**ViewAccount**	SysAdmin views an existing user account
UC-620	**ModifyAccount**	SysAdmin makes changes to an existing user account
UC-630	**RemoveAccount**	SysAdmin removes a user account from the system
UC-640	**ViewSystemConfiguration**	SysAdmin views configuration settings for the cuCare client application

UC-650	**SetAuditTime**	SysAdmin changes the time of the daily audit process
UC-660	**SetServerIP**	SysAdmin changes the IP address of the cuCare server
UC-670	**SetServerPort**	SysAdmin changes the port number of the cuCare server
UC-680	**TestConnection**	SysAdmin tests the connection to the cuCare server using current system settings
	AUTOMATION	
UC-700	**RunAudit**	Execution of the daily audit process
	EXTENDS	
UC-800	**SelectPhysician**	Extends use cases involving data entry by providing a list of available physicians
UC-810	**ServerResponseTimeout**	Extends use cases involving client server communication by providing the user with a prompt to inform then that the cuCare client application has failed to establish a connection with the server
UC-820	**InvalidUsername**	Extends the **LogIn** use case in instances where an invalid username has been entered in an attempt to log in to the system
UC-830	**InvalidDate**	Extends use cases involving data entry by notifying the user that the date they have entered is invalid
UC-840	**InvalidTime**	Extends use cases involving data entry by notifying the user that the time they have entered is invalid
UC-850	**UsernameNotAvailable**	Extends the CreateAccount use case in instances where a username is already being used

4.2 Use Case Diagrams

This section presents diagrams that demonstrate the relationships between the use cases for the proposed cuCare system.

Figure 1 – Use Case Diagram – Top Level



Note: **ServerResponseTimeout** extends every use case that involves communication with the cuCare server. This is not shown on the diagrams for practical reasons (too many arrows).

Figure 2 – Use Case Diagram – Patient Management

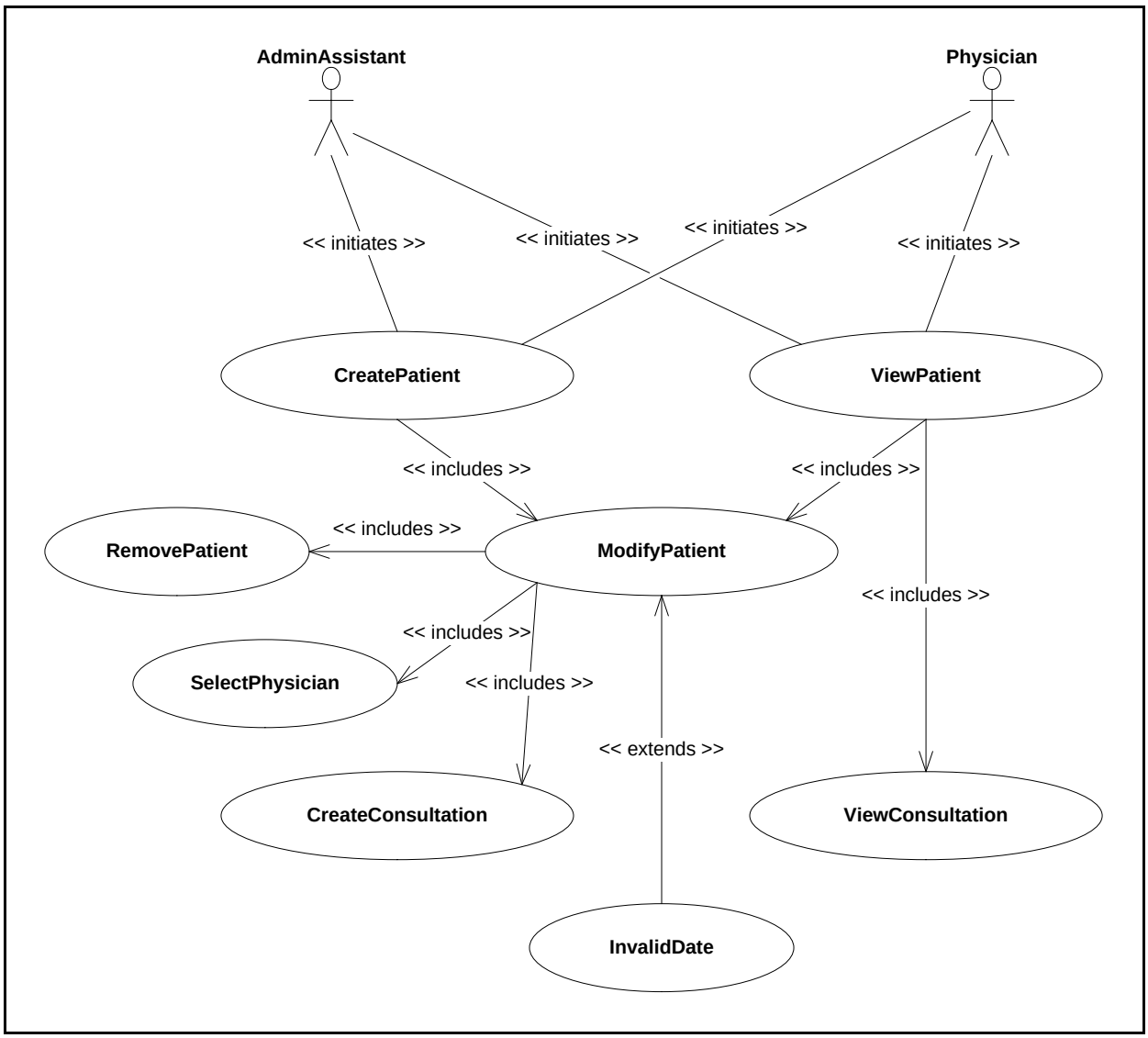


Figure 3 – Use Case Diagram – Consultation Management

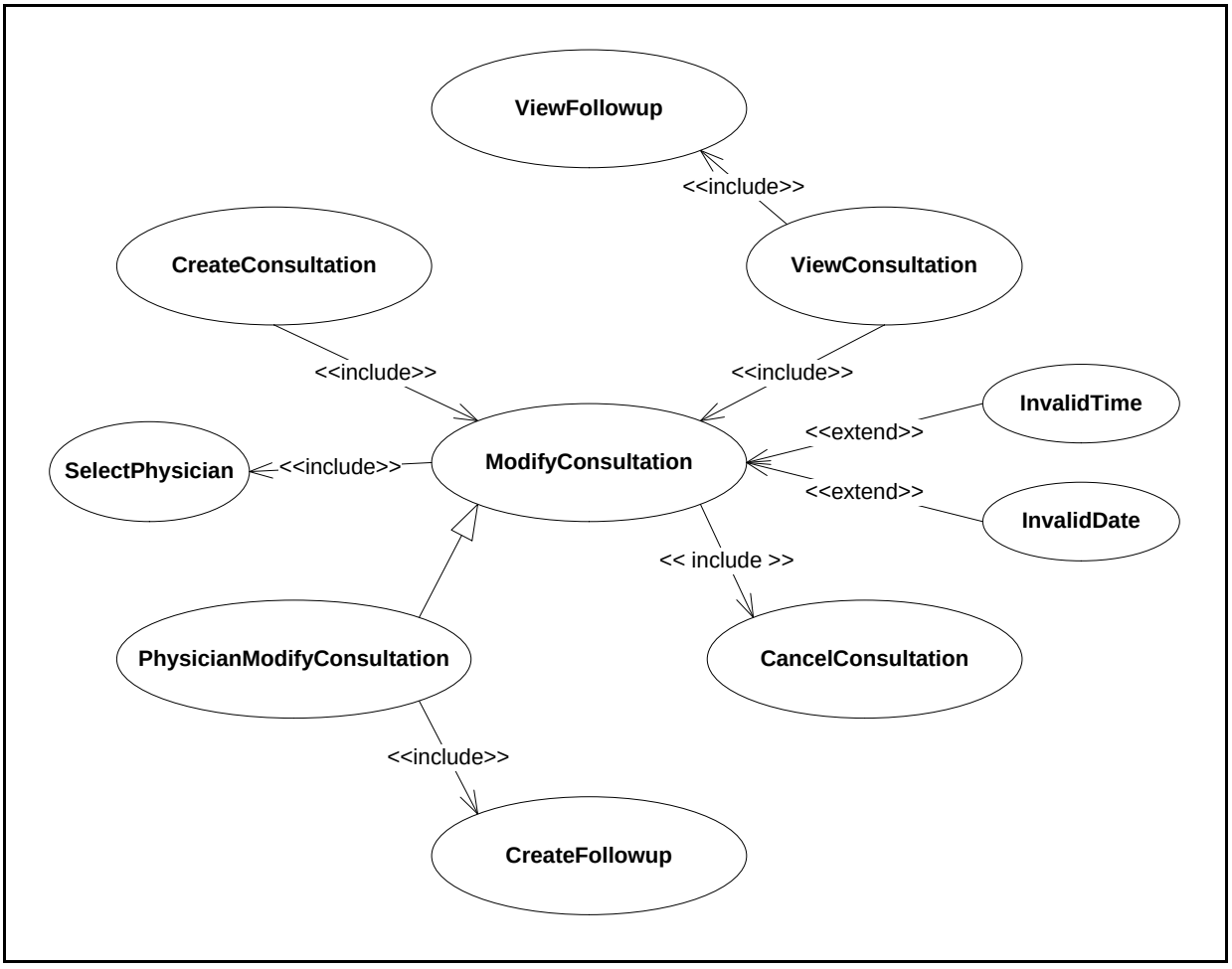


Figure 4 – Use Case Diagram – Follow-up Management

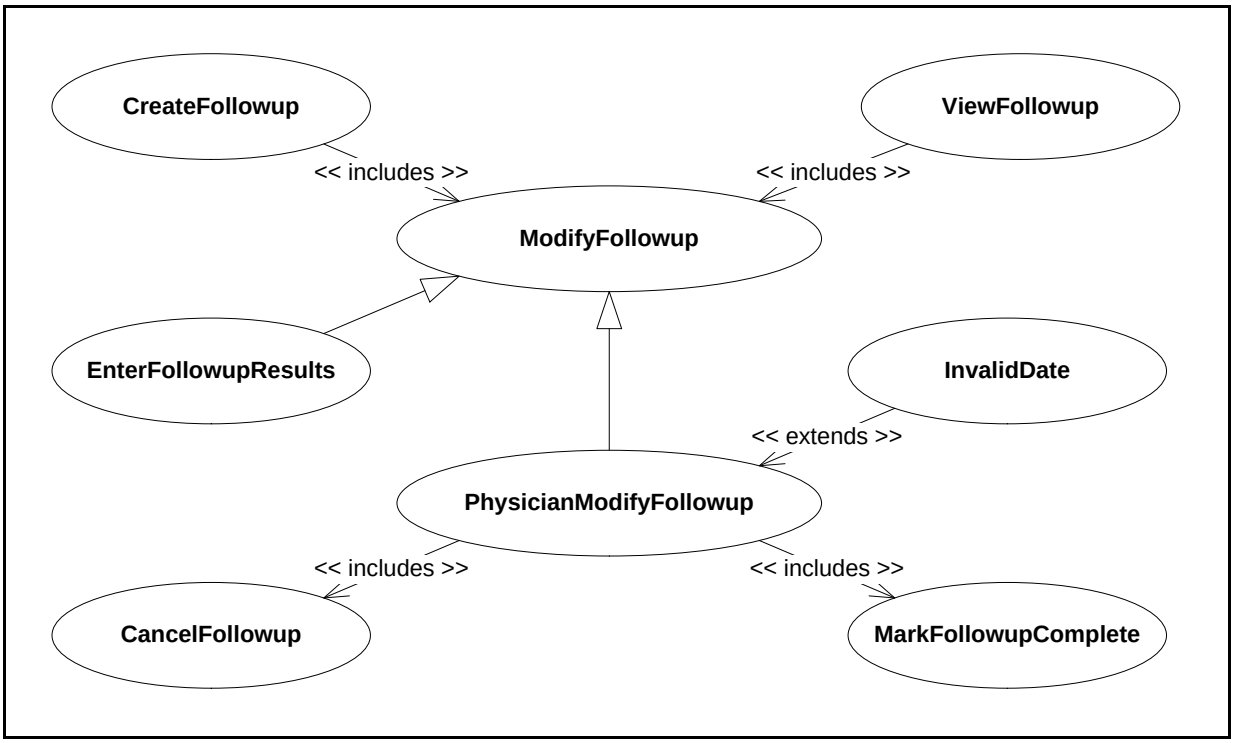


Figure 5 – Use Case Diagram – System Management

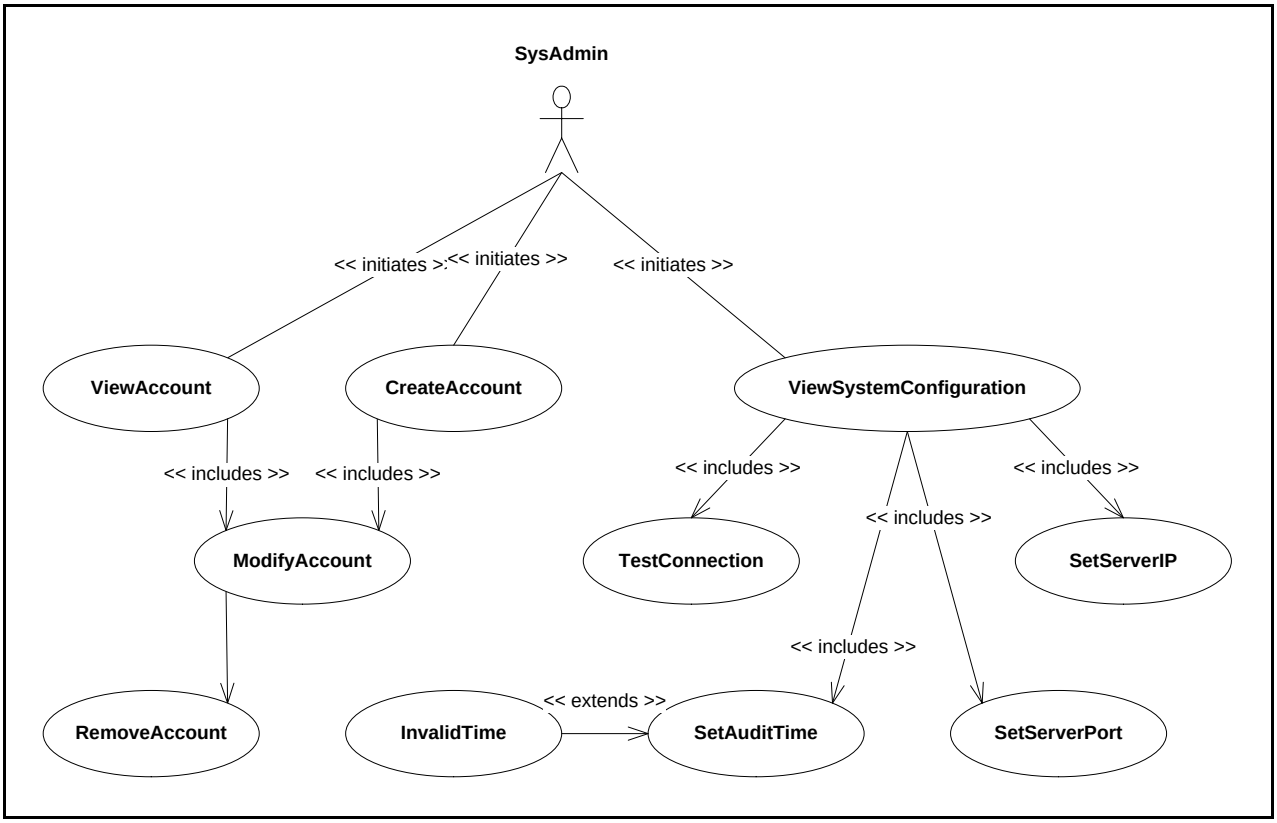
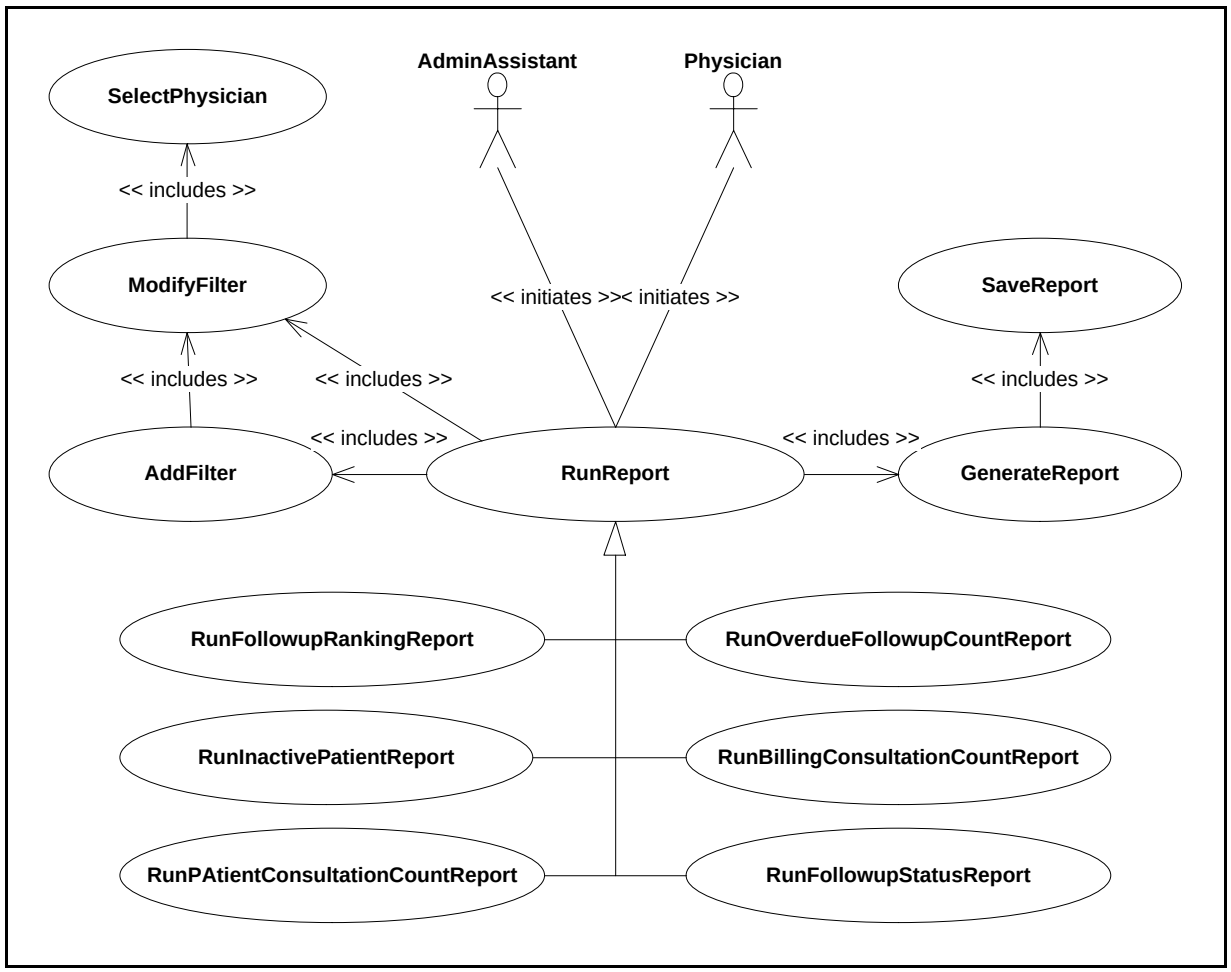


Figure 6 – Use Case Diagram – Reports



4.3 Detailed Use Case Descriptions

This section presents the detailed descriptions of all of the use cases for the proposed system. Each use case description provides a step by step flow of events, the context within which the use case is expected to occur, and identifies the corresponding functional requirements.

Use Case Identifier	UC-100
Use Case Name	**LogIn**
Participating Actors	Initiated by SysAdmin OR AdminAssistant OR Physician
Flow of Events	1. The user enters their username, and selects the "Log In" option. 2. The system uses the supplied username to log the user in with the corresponding access level and view for their account type.
Entry Conditions	The client application has been launched and the user is presented with a login view.
Exit Conditions	The user is logged-in to the system. The system displays the top level view corresponding with their account type.
Traceability	FR 1.b

Use Case Identifier	UC-110
Use Case Name	**LogOut**
Participating Actors	Initiated by SysAdmin OR AdminAssistant OR Physician
Flow of Events	1. The user selects the "Log Out" option. 2. The system logs the user out of the system, and opens the "Log In" view for another user to log in.
Entry Conditions	The user is logged-in to the system.
Exit Conditions	The user is logged out of the system, which is now displaying the log-in view.
Traceability	FR 1.b

Use Case Identifier	UC-200
Name	**CreatePatient**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects the "Create Patient" option. 2. The system creates a new patient record with default values, and then opens this record for modification (**ModifyPatient**).
Entry Conditions	The user is logged in as an AdminAssistant or Physician.
Exit Conditions	A new patient record that has been filled out by the user is present in the database.
Traceability	FR 3.a FR 4.a FR 5

Use Case Identifier	UC-210
Name	**ViewPatient**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects the "Select Patient" option. 2. The system displays a searchable list of all patient records, optionally filterable by a follow-up status and primary physician.

	3. The user selects a patient from the list. 4. The system opens that patient record for modification (**ModifyPatient**). 5. The system also displays a tree of the patient's consultations and the associated follow-ups, which are selectable (**ViewConsultation**).
Entry Conditions	The user is logged in as an AdminAssistant or Physician.
Exit Conditions	The user has set which patient is currently selected and performed any actions involving that patient's consultations and follow ups they wished.
Traceability	FR 3.b FR 3.h FR 4.a FR 5

Use Case Identifier	UC-220
Name	**ModifyPatient**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user makes any changes to the patient record fields they wish. They then select the "Submit Changes" option. 2. The system records the changes in the database. 3. The user may also select the "New Consultation" option at any point. This will begin the process of creating a new consultation (**CreateConsulation**).
Entry Conditions	The user is logged in as an AdminAssistant or Physician. A patient has been selected to be modified.
Exit Conditions	The user has made whatever changes they wished to the patient record fields and those changes have been saved to the database.
Traceability	FR 3.c FR 4.a FR 5

Use Case Identifier	UC-230
Name	**RemovePatient**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects the "Remove Patient" option. 2. The system asks the user for confirmation. 3. The user confirms the remove. 4. The system records the changes in the database.
Entry Conditions	The user is logged in as an AdminAssistant or Physician. A patient has been selected to be modified.
Exit Conditions	The patient record has been marked as removed from the system, and will no longer show up in the list of patient records.
Traceability	FR 3.c FR 4.a FR 5

Use Case Identifier	UC-300
Use Case Name	**CreateConsultation**
Participating Actors	Initiated by AdminAssistant OR Physician
Flow of Events	1. User (AdminAssistant or Physician) selects the "Schedule Consultation"

	option for a particular patient. 2. The system presents a blank form into which information about the consultation can be entered. 3. The user makes modifications to the date, time, reason for consultation, and Physician fields in the form, and submits (include use case ModifyConsultation). **Note:** All of the listed fields are mandatory. **Note:**OHIP billing numbers are kept in patient records, and so they do not need to be specified by the user here. 4. The system saves the newly created form to storage and notifies the user of the successful consult creation.
Entry Conditions	The user is logged in as an AdminAssistant or Physician. A patient has been selected to be modified.
Exit Conditions	The new consultation has been added to storage, and the user has received acknowledgement that this has happened.
Traceability	FR 3.d FR 4.a FR 6

Use Case Identifier	UC-310
Use Case Name	**ViewConsultation**
Participating Actors	Initiated by AdminAssistant OR Physician
Flow of Events	1. Having already selected a patient, the user (AdminAssistant or Physician) selects a consultation belonging to that patient. 2. The system brings up an editable form containing the information on that consultation. 3. The user views the details of the consultation and/or makes a modification to the consultation (include use case ModifyConsultation).
Entry Conditions	The user is logged in as an AdminAssistant or Physician. A patient has been selected to be modified.
Exit Conditions	The user has completed the task they viewed the consultation for.
Traceability	FR 3.b FR 3.e FR 4.a FR 6

Use Case Identifier	UC-320
Use Case Name	**ModifyConsultation**
Participating Actors	Initiated by AdministrativeAssistant OR Physician
Flow of Events	1. Having already selected a consultation, the user modifies a number of fields in the consultation form. The modifiable fields are: date, time, reason consultation, attending physician (include use case **SelectPhysician**), and three others included in the **PhysicianModifyConsultation** use case (these can only be modified by a Physician user). 2. The user selects the “Save Changes” option on the form. 3. The modification of the consultation is saved to storage and the user

	is notified of the completion of the transaction.
Entry Conditions	The user is logged in as an AdminAssistant or Physician. A consultation has been selected to be modified.
Exit Conditions	The modification has been processed, storage updated accordingly, and the user has been notified that the transaction is complete.
Traceability	FR 3.e FR 4.a FR 6

Use Case Identifier	UC-330
Use Case Name	**PhysicianModifyConsultation**
Participating Actors	Initiated by Physician
Flow of Events	1. Physician makes any number of modifications that only Physicians are allowed to make. These include: editing the "diagnosis" field, editing the "real reason for consult" field and selecting the "Schedule New Follow-up" option (include use case **CreateFollowup**).
Entry Conditions	The user is logged in as a Physician. A consultation has been selected to be modified.
Exit Conditions	The user has made the desired modifications.
Traceability	FR 4.c FR 4.d FR 6.d FR 6.f

Use Case Identifier	UC-340
Use Case Name	**CancelConsultation**
Participating Actors	Initiated by AdministrativeAssistant OR Physician
Flow of Events	1. The user selects the "Cancel Consultation" option on the selected Consultation (specializes use case **ModifyConsultation**). 2. A confirmation window is brought up, asking the user if they're sure they want to cancel the consultation. 3. The user selects "Yes" if they wish to proceed, or "No" if they wish to reconsider. Selecting "No" brings the user back to the editable form of the selected consultation. Selecting "Yes" moves the user to step 4. 4. The system updates the status of the stored consultation record to "Cancelled", and notifies the user that the cancellation has been carried out successfully.
Entry Conditions	The user is logged in as an AdminAssistant or Physician. A consultation has been selected to be modified.
Exit Conditions	The consultation record has been updated with "Cancelled" status in storage, and the user has been notified of the successful cancellation.
Traceability	FR 3.e FR 4.c

Use Case Identifier	UC-400
Use Case Name	**CreateFollowup**
Participating Actors	Initiated by Physician

Flow of Events	1. A Physician selects the "Schedule Follow-up Request" option for a particular patient. 2. The system presents a blank form into which information about the follow-up request can be entered. 3. The user makes modifications to the type of follow-up being requested, and the date by which the follow-up request is expected to be completed in the form, and submits (include use case **ModifyFollowup**). **Note:** All of the listed fields are mandatory. 4. The system saves the newly created form to storage and notifies the user of the successful creation of a new follow-up record.
Entry Conditions	The user must be viewing an existing consultation record.
Exit Conditions	The new follow-up record has been saved to storage, and the user has received acknowledgement that this has happened.
Traceability	FR 4.c.ii FR 7

Use Case Identifier	UC-410
Use Case Name	**ViewFollowup**
Participating Actors	Initiated by AdminAssistant OR Physician
Flow of Events	1. The user selects a follow-up record associated with a consultation. 2. The system brings up an editable form containing the information on that follow-up request. 3. The user views the details of the follow-up request and/or makes a modification to the follow-up request (include use case **ModifyFollowup**).
Entry Conditions	The user must be viewing an existing consultation record.
Exit Conditions	
Traceability	FR 3.b FR 4.d FR 7

Use Case Identifier	UC-420
Use Case Name	**ModifyFollowup**
Participating Actors	Initiated by AdminAssistant OR Physician
Flow of Events	1. Having already selected a specific follow-up record, the user modifies a number of fields in the follow-up request form. The type of user performing the modifications determines what fields can be modified. Both AdminAssistant and Physician users are able to enter follow-up results, causing the status of the follow-up request to be set to "Received" (include use case **EnterFollowupResults)**. Other fields can only be modified by a Physician (include use case **PhysicianModifyFollowup)**. 2. The user selects the "Save Changes" option on the form. 3. The system updates the follow-up record in storage to reflect the changes and the user is notified of the completion of the transaction.
Entry Conditions	The user must be viewing an existing follow-up record.
Exit Conditions	The follow-up record has been modified, storage is updated accordingly, and the user has been notified that the transaction is complete.
Traceability	FR 3.f

	FR 4.d FR 7

Use Case Identifier	UC-430
Use Case Name	**EnterFollowupResults**
Participating Actors	Initiated by AdminAssistant OR Physician
Flow of Events	1. The user enters the results of the follow-up request (e.g. medical test results or specialist diagnosis) (specializes use case **ModifyFollowup**). The user selects the "Save Changes" option to save the changes to the system. 2. The system updates the follow-up record in storage to reflect the changes and the user is notified of the completion of the transaction.
Entry Conditions	The user must be viewing an existing follow-up record.
Exit Conditions	The follow-up record has been modified, storage is updated accordingly, and the user has been notified that the transaction is complete.
Traceability	FR 3.f FR 4.d FR 7.c.ii FR 7.c.iii

Use Case Identifier	UC-440
Use Case Name	**PhysicianModifyFollowup**
Participating Actors	Initiated by Physician
Flow of Events	1. The user makes modifications to a follow-up request they are currently viewing (specializes use case **ModifyFollowup**). These include: selecting the type of follow-up being requested, the date by which the follow-up request must be completed or the results of the follow-up must be received. 2. The user selects the "Save Changes" option to save the changes to the system OR the "Cancel Follow-up Request" option (include use case **CancelFollowup**) OR the "Mark Complete" option (include use case **MarkComplete**). If the user selects the "Save changes" option: 3. The system updates the follow-up record in storage to reflect the changes and the user is notified of the completion of the transaction.
Entry Conditions	The user must be viewing an existing follow-up record.
Exit Conditions	The follow-up record has been modified, storage is updated accordingly, and the user has been notified that the transaction is complete.
Traceability	FR 4.d FR 7

Use Case Identifier	UC-450
Use Case Name	**MarkFollowupComplete**
Participating Actors	Initiated by Physician
Flow of Events	1. The user selects the "Mark Complete" option on the selected follow-up request form. 2. A confirmation window is brought up, asking the user if they're sure they want to set the status of the follow-up to "Completed". 3. The user selects "Yes" if they wish to proceed, or "No" if they wish to

	reconsider. Selecting "No" brings the user back to the editable form of the selected follow-up record. Selecting "Yes" moves the user to step 4. 4. The system updates the status of the stored follow-up record to "Completed", and notifies the user that the change of status has been carried out successfully.
Entry Conditions	The user must be viewing an existing follow-up record.
Exit Conditions	The follow-up record has been updated with "Completed" status in storage, and the user has been notified of the successful status update.
Traceability	FR 4.d FR 7.a.iv

Use Case Identifier	UC-460
Use Case Name	**CancelFollowup**
Participating Actors	Initiated by Physician
Flow of Events	1. The user selects the "Cancel Follow-up Request" option on the selected Consultation. 2. A confirmation window is brought up, asking the user if they're sure they want to cancel the consultation. 3. The user selects "Yes" if they wish to proceed, or "No" if they wish to reconsider. Selecting "No" brings the user back to the editable form of the selected follow-up record. Selecting "Yes" moves the user to step 4. 4. The system updates the status of the stored follow-up record to "Cancelled", and notifies the user that the cancellation has been carried out successfully.
Entry Conditions	The user must be viewing an existing follow-up record.
Exit Conditions	The follow-up record has been updated with "Cancelled" status in storage, and the user has been notified of the successful cancellation.
Traceability	FR 4.d FR 7

Use Case Identifier	UC-500
Name	**RunReport**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects the "Run Reports" option 2. The system displays a window where the patient may specify the report they wish to run 3. The user selects a report type (include use cases **RunFollowupRankingReport**, **RunInactivePatientReport**, **RunPatientConsulationCountReport**, **RunOverdueFollowupCountReport**, **RunBillingConsulationsReport**, and **RunFollowupStatusReport**) 4. If desired, the user selects the "Add Filter" option (include use case **AddFilter**) 5. The user selects the "Generate Report" option (include use case **Generate Report**)
Entry Conditions	The user is logged in as an AdminAssistant or Physician.
Exit Conditions	The desired report has been generated and the user may either return to the main window or continue with another report.

Traceability	FR 3.g FR 8

Use Case Identifier	UC-510
Name	**RunFollowupRankingReport**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects 'Follow Up Ranking Report' from the list of report types (specializes use case **RunReport**).
Entry Conditions	The user is logged in as an AdminAssistant or Physician.
Exit Conditions	A report ranking which types of follow-ups are most often overdue has been generated and the user may either return to the main window or continue with another report.
Traceability	FR 8.a.i

Use Case Identifier	UC-520
Name	**RunInactivePatientReport**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects 'Inactive Patient Report' from the list of report types (specializes use case **RunReport**). 2. The system displays fields for 'Start Date' and 'End Date'. 3. The user fills out the field.
Entry Conditions	The user is logged in as an AdminAssistant or Physician.
Exit Conditions	A report listing patients with no consultations for the specified period of time has been generated and the user may either return to the main window or continue with another report.
Traceability	FR 8.a.ii

Use Case Identifier	UC-530
Name	**RunPatientConsulationCountReport**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects 'Patient Consultation Count Report' from the list of report types (specializes use case **RunReport**). 2. The system displays a field for 'Min Consultations'. 3. The user fills out the field.
Entry Conditions	The user is logged in as an AdminAssistant or Physician.
Exit Conditions	A report listing all patients with a number of consultations in the specified range during the specified time period has been generated and the user may either return to the main window or continue with another report.
Traceability	FR 8.a.iii

Use Case Identifier	UC-540
Name	**RunOverdueFollowupCountReport**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects 'Overdue Follow-up Count Report' from the list of report types (specializes use case **RunReport**). 2. The system displays a field for 'Min Overdue'.

	3. The user fills out the field.
Entry Conditions	The user is logged in as an AdminAssistant or Physician.
Exit Conditions	A report listing all patients with a number of overdue follow-ups in the specified range has been generated and the user may either return to the main window or continue with another report.
Traceability	FR 8.a.iv

Use Case Identifier	UC-550
Name	**RunBillingConsulationsCountReport**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects 'Billing Consultations Report' from the list of report types (specializes use case **RunReport**).
Entry Conditions	The user is logged in as an AdminAssistant or Physician.
Exit Conditions	A report listing number of consultations by billing number has been generated and the user may either return to the main window or continue with another report.
Traceability	FR 8.a.v

Use Case Identifier	UC-560
Name	**RunFollowupStatusReport**
Participating actors	Initiated by Physician
Flow of events	1. The user selects 'Follow-up status Report' from the list of report types (specializes use case **RunReport**). 2. The system displays a field for 'Follow-up Status'. 3. The user fills out the field.
Entry Conditions	The user is logged in as a Physician.
Exit Conditions	A report listing follow-ups with the 'received' state has been generated and the user may either return to the main window or continue with another report.
Traceability	FR 8.b.i

Use Case Identifier	UC-570
Name	**AddReportFilter**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects the "Add Filter" option from the report screen. 2. The system adds a default filter to the report. 3. The user can then modify the default filter (include use case **ModifyFilter**).
Entry Conditions	The user is logged in as an AdminAssistant or Physician. The user is in the report generation dialog.
Exit Conditions	The desired filter has been added to the report.
Traceability	FR 8.c

Use Case Identifier	UC-580
Name	**ModifyReportFilter**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user changes one of the options in any of the filters currently added to

	the report. Different types of filters include 'primary physician', 'patient age range', and 'follow-up type'. Primary Physician has an associated field to select the physician (include use case **SelectPhysician**). Patient age range has two associated fields- max age and min age. Follow-up type has one associated field in which the follow-up type can be selected from a list. 2. The system records the changes to the filter.
Entry Conditions	The user is logged in as an AdminAssistant or Physician. The user is in the report generation dialog.
Exit Conditions	The filter changes are recorded and the next time the report is generated the new filter will be used.
Traceability	FR 8.c

Use Case Identifier	UC-590
Name	**GenerateReport**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects the "Generate Report" option from the report view. 2. The system polls the database with the parameters specified by report type, any user entered information, and filters. Once the data has been retrieved, it is displayed to the user. 3. The user may select the "Save Report" option (include use case **SaveReport**).
Entry Conditions	The user is logged in as an AdminAssistant or Physician. The user is in the report generation dialog.
Exit Conditions	The report has been generated and the user may view and save it.
Traceability	FR 8

Use Case Identifier	UC-595
Name	**SaveReport**
Participating actors	Initiated by AdminAssistant OR Physician
Flow of events	1. The user selects the "Save Report" option from the report view. 2. The system opens a save file dialog box. 3. The user selects a file name and location using the save file dialog. 4. The system saves the report as a text file with the name and location the user specified.
Entry Conditions	The user is logged in as an AdminAssistant or Physician. The user is in the report generation dialog. A report has been generated.
Exit Conditions	The report is saved as a text file and can be accessed outside of the system.
Traceability	FR 8.d

Use Case Identifier	UC-600
Use Case Name	**CreateAccount**
Participating Actors	Initiated by SysAdmin
Flow of Events	1. The SysAdmin selects the "Create New Account" option. 2. The system opens up a dialog window with editable fields for the different types of user account information (See FR 1.a)

	3. The SysAdmin enters in the account information and selects whether the new account will be a Physician or an AdminAssistant (include use case **ModifyAccount**) 4. The SysAdmin selects the "Submit" option. 5. The system stores the supplied data in a new user account on the server and closes the dialog window.
Entry Conditions	The user is logged in as a SysAdmin.
Exit Conditions	A new user account has been created and stored on the server.
Traceability	FR 1.a FR 1.c FR 2.a

Use Case Identifier	UC-610
Use Case Name	**ViewAccount**
Participating Actors	Initiated by SysAdmin
Flow of Events	1. The SysAdmin selects the "Select Existing Account" option. 2. The system opens a dialog window with a list of all user accounts stored on the system, searchable by account name. 3. The SysAdmin selects the user account he wishes to view from the list, and selects the "Select" option. 4. The system closes the dialog window and present's the selected user account's info in the SysAdmin's view. 5. The SysAdmin may modify this information if he chooses to (include use case **ModifyAccount**).
Entry Conditions	The user is logged in as a SysAdmin.
Exit Conditions	The selected user account is displayed in the SysAdmin's view.
Traceability	FR 1.a FR 1.c FR 2.b

Use Case Identifier	UC-620
Use Case Name	**ModifyAccount**
Participating Actors	Initiated by SysAdmin
Flow of Events	1. The SysAdmin modifies one or more of the fields in the selected user account. 2. The SysAdmin selects the "Save Changes" option. 3. The system synchronizes the changes with the server and saves the modified account.
Entry Conditions	The user is logged in as a SysAdmin. An account has been selected to be modified.
Exit Conditions	The changes to the selected user account are stored in the system.
Traceability	FR 1.a FR 1.c FR 2.b

Use Case Identifier	UC-630
Use Case Name	**RemoveAccount**

Participating Actors	Initiated by SysAdmin
Flow of Events	1. The SysAdmin selects the "Remove Account" option. (specializes use case **ModifyAccount**) 2. The selected user is removed from the system.
Entry Conditions	The user is logged in as a SysAdmin. An account has been selected to be modified.
Exit Conditions	The selected user account has been removed from the system.
Traceability	FR 1.a FR 2.c

Use Case Identifier	UC-640
Use Case Name	**ViewSystemConfiguration**
Participating Actors	Initiated by SysAdmin
Flow of Events	1. The SysAdmin selects the "Configure System" option. 2. The system opens a new dialog window displaying fields with the server IP address and port number, the set time for the audit process to run, and a "Test Connection" option. 3. The SysAdmin may modify any of the fields or select the "Test Connection" option. (include use cases **SetAuditTime, SetServerIP, SetServerPort,** and **TestConnection**) 4. The SysAdmin selects the "Close" option. 5. The system closes the dialog window and returns the SysAdmin to his default view.
Entry Conditions	The user is logged in as a SysAdmin.
Exit Conditions	The system has returned the SysAdmin to his default view.
Traceability	FR 2.d FR 2.e

Use Case Identifier	UC-650
Use Case Name	**SetAuditTime**
Participating Actors	Initiated by SysAdmin
Flow of Events	1. The SysAdmin modifies the set time for the audit process to run. 2. The SysAdmin selects the "Save" option. 3. The system stores the new audit time in the server.
Entry Conditions	The user is logged in as a SysAdmin. The user is in the system configuration dialog.
Exit Conditions	The new time for the audit process has been saved on the server.
Traceability	FR 2.d FR 9.a

Use Case Identifier	UC-660
Use Case Name	**SetServerIP**
Participating Actors	Initiated by SysAdmin
Flow of Events	1. The SysAdmin modifies the set server IP address. 2. The SysAdmin selects the "Save" option. 3. The system stores the server IP address in the client.
Entry Conditions	The user is logged in as a SysAdmin.

	The user is in the system configuration dialog.
Exit Conditions	The new server IP address has been saved in the client process.
Traceability	FR 2.e

Use Case Identifier	UC-670
Use Case Name	**SetServerPort**
Participating Actors	Initiated by SysAdmin
Flow of Events	1. The SysAdmin modifies the set server port number. 2. The SysAdmin selects the "Save" option. 3. The system stores the new server port number in the client.
Entry Conditions	The user is logged in as a SysAdmin. The user is in the system configuration dialog.
Exit Conditions	The new server port number has been saved in the client process.
Traceability	FR 2.e

Use Case Identifier	UC-690
Use Case Name	**TestConnection**
Participating Actors	Initiated by SysAdmin
Flow of Events	1. The SysAdmin selects the "Test Connection" option. 2. The client attempts connection to the server process using the stored server IP address and port number. 3. The system displays whether or not the connection test was a success.
Entry Conditions	The user is logged in as a SysAdmin. The user is in the system configuration dialog.
Exit Conditions	The client has attempted connection to the server, and displayed the results to the SysAdmin.
Traceability	FR 2.e

Use Case Identifier	UC-700
Use Case Name	**RunAudit**
Participating Actors	
Flow of Events	1. The system initiates the audit process daily at the specified time stored in the server. 2. The audit selects all of the follow-up records stored in the system that: have a status of "Pending" AND are past the due date specified by the physician. 3. The system changes the status of all of the selected follow-up records to "Overdue".
Entry Conditions	The time of day is currently the specified time for the audit process to run.
Exit Conditions	The audit process has been run on the system, and overdue follow-up records have been marked as such.
Traceability	FR 9

Use Case Identifier	UC-800
Use Case Name	**SelectPhysician**

Participating Actors	AdminAssistant OR Physician
Flow of Events	1. Having already selected a consultation, the user (AdminAssistant or Physician) selects the "Select Physician" option from the form. 2. The system brings up a list of all physicians currently practicing at the clinic. 3. The user selects the desired physician. 4. The user is brought back to the form, where the selected physician is shown as the attending physician for the consultation.
Entry Conditions	The user is logged in as an AdminAssistant or Physician.
Exit Conditions	The desired physician has been selected and is displayed on the consultation form.
Traceability	FR 3.e

Use Case Identifier	UC-810
Use Case Name	**ServerResponseTimeout**
Participating Actors	SysAdmin OR AdminAssistant OR Physician
Flow of Events	1. The system notifies the user that it cannot connect to the server process.
Entry Conditions	A connection attempt to the server process from the client failed.
Exit Conditions	The current client operation is aborted.
Traceability	NFR 1.i

Use Case Identifier	UC-820
Use Case Name	**InvalidUsername**
Participating Actors	SysAdmin OR AdminAssistant OR Physician
Flow of Events	1. The system notifies the user that the username they have tried to log in with is invalid.
Entry Conditions	A user has attempted to log in with an invalid username.
Exit Conditions	The current log in operation is aborted.
Traceability	FR 1.b NFR 1.i

Use Case Identifier	UC-830
Use Case Name	**InvalidDate**
Participating Actors	AdminAssistant OR Physician OR SysAdmin
Flow of Events	1. The system disallows the submission, and returns the user to the screen they entered the invalid date into.
Entry Conditions	A user has entered a date that is not valid, and attempted to submit the information.
Exit Conditions	The attempted submission has been stopped, and the user has been returned to a screen from which they can rectify their error.
Traceability	FR 5.g FR 6.c FR 7.b NFR 1.i

Use Case Identifier	UC-840
Use Case Name	**InvalidTime**
Participating Actors	AdminAssistant OR Physician OR SysAdmin
Flow of Events	1. The system disallows the submission, and returns the user to the screen they entered the invalid date into.
Entry Conditions	A user has entered a time that is not valid, and attempted to submit the information.
Exit Conditions	The attempted submission has been stopped, and the user has been returned to a screen from which they can rectify their error.
Traceability	FR 6.c NFR 1.i

Use Case Identifier	UC-850
Use Case Name	**UsernameNotAvailable**
Participating Actors	SysAdmin
Flow of Events	1. The system notifies the user that the username they have tried to use for a new user account on the system is already taken.
Entry Conditions	A user has attempted to create a new user account with a username that is already associated with an existing account.
Exit Conditions	The attempted submission has been stopped, and the user has been returned to a screen from which they can rectify their error.
Traceability	FR 1.a.i FR 2.a NFR 1.i

5 Object Model

This section presents an analysis of the entity classes that will store persistent data within the proposed cuCare system. In addition to describing the data stored by the various entity objects, this section also outlines the relationships between the different types of entity objects within the system through the use of class diagrams.

5.1 Data Dictionary

The data dictionary lists the entity objects included in the proposed system, provides a short description for each, and identifies their associated attributes and associations with other entity objects within the system.

Entity Object	Attributes/Associations	Definition
User	• Username • First Name • Last Name • Title • Address • Contact Info • Date (added to system)	A user who can use the CuCare system.
Physician		User account type for physicians in the CuCare system.
Administrative Assistant		User account type for administrative assistants in the CuCare system.
System Administrator		User account type for system administrators of the CuCare system.
Patient	• ID • First Name • Last Name • Notes • Contact Info • Date (of birth) • Date (added to system) • Health Card • Physician • Consultations	A patient record in the CuCare system that stores the patient's profile information as well as a list of all of their consultations.
Consultation	• Reason • Diagnosis • Comments • Status • Date (scheduled) • Time (scheduled) • Physician • Follow-ups	A consultation record and all of its associated data, including all associated follow-up requests.

Followup	• Status • Date (follow-up is received) • Date (follow-up is due) • Date (follow-up is completed)	A follow-up record that contains a status and the due dates associated with it.
ReturnConsultation	• Consultation	Type of follow-up record indicating that a return consultation is required.
MedicationRenewal	• Medication	Type of follow-up record that requires a renewal of medication prescription.
ResultantFollowup	• Results	Type of follow-up record that requires results to be sent to the practice.
Referral	• Specialist Name	Type of follow-up record that requires a referral appointment with a specialist, and to have the specialist diagnosis sent to the practice.
Test	• Test Type	Type of follow-up record that requires a medical test to be completed, and to have the test results sent to the practice.
Report	• Results	A collection of data retrieved from the repository, defined by parameters (and optionally filters) set by the user.
List of Patients with Received Results		Type of report only available to physicians listing all patients who have follow-ups with the 'Received' status.
Types of Follow-ups Most Overdue		Type of report listing follow-ups in order of most overdue.
Patients with High Number of Consultations	• Minimum Consultations	Type of report listing patients with a number of consultations above a certain threshold specified by the user.
Patients with No Consultations in Specified Time	• Date (range start) • Date (range end)	Type of report listing all patients with no recorded consultations within a user specified date range.
Patients with Frequently Overdue Follow-ups	• Minimum Overdue Follow-ups	Type of report listing patients who have a number of overdue follow-ups over a threshold specified by the user.
Number of Consultations by Billing Number		Type of report listing all billing numbers and their respective number of consultations.

Filter		A filter used to narrow down results within reports.
Filter by Primary Physician	• Physician	Type of filter limiting report results to a user specified physician.
Filter by Patient Age Range	• Minimum Age • Maximum Age	Type of filter limiting report results to patients who fall in a certain user specified age range.
Filter by Follow-up Type	• Follow-up Type	Type of filter limiting by a specific follow-up type

5.2 Class Diagrams

Class diagrams outline the relationships between the different types of entity objects within the proposed system.

Figure 7 – Class Diagram – Users and Patients

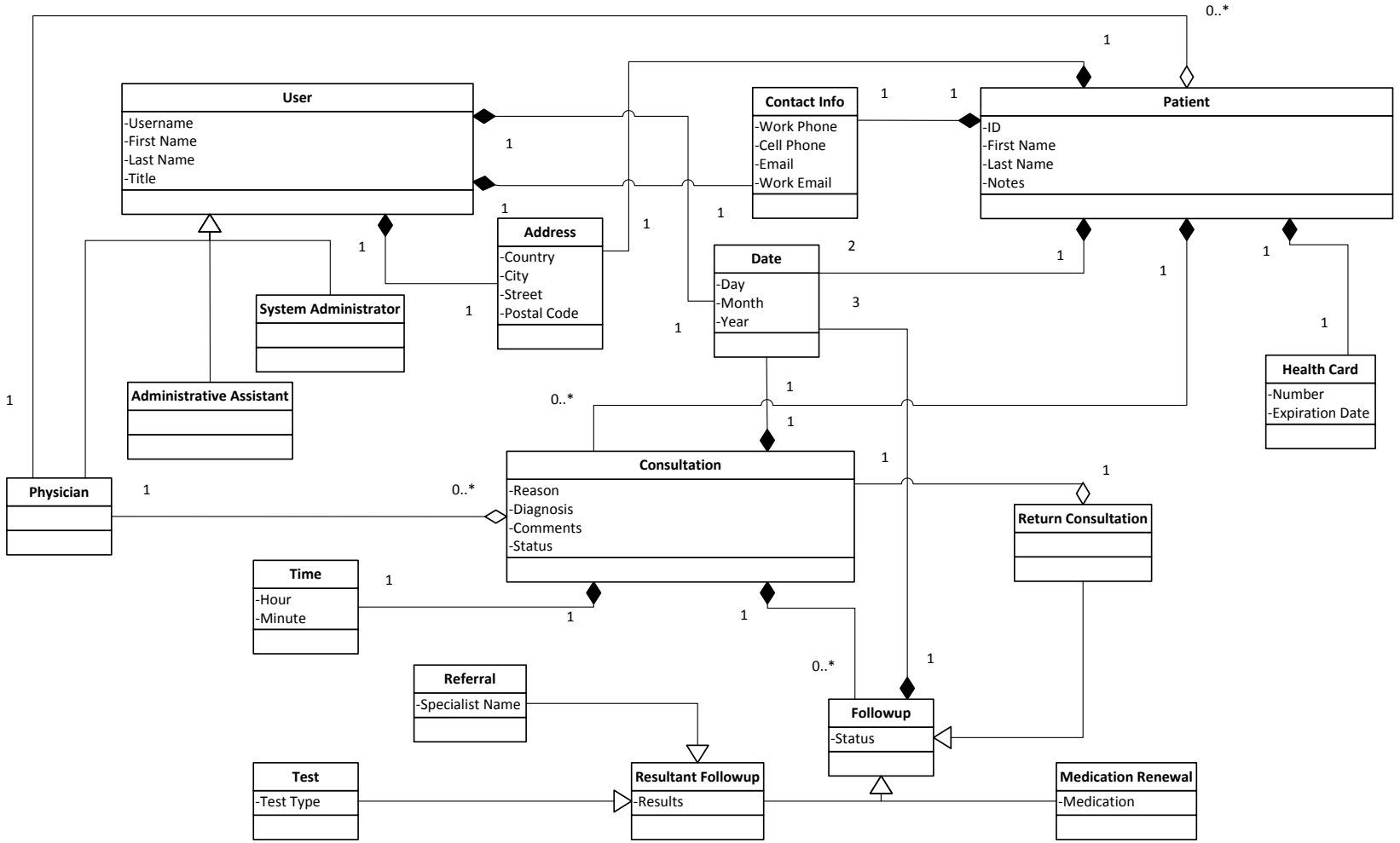
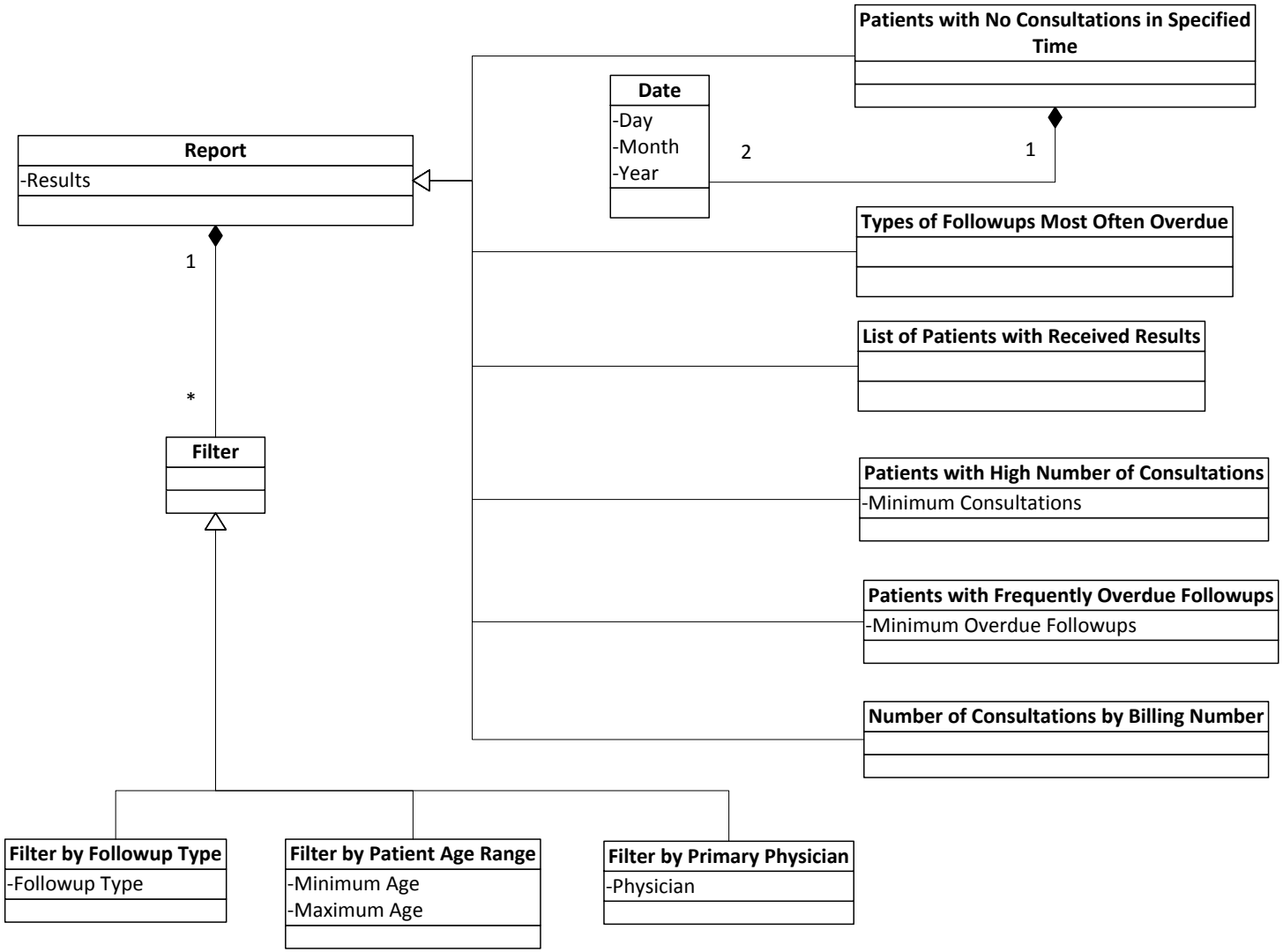


Figure 8 – Class Diagram – Reports



6 Dynamic Model

This section presents an analysis of the runtime interactions within the system. The lifetime of entity objects (e.g. a consultation record) is illustrated through the use of state diagrams. This section also includes sequence diagrams, representing interaction of system components for a selected set of use cases.

6.1 State Machine Diagrams

This section illustrates the lifetime of entity objects for the proposed system. The diagrams below present the possible states that the entity objects can have, as well as the activities that can cause their state to change.

Figure 9 – State Machine Diagram – Consultation Record

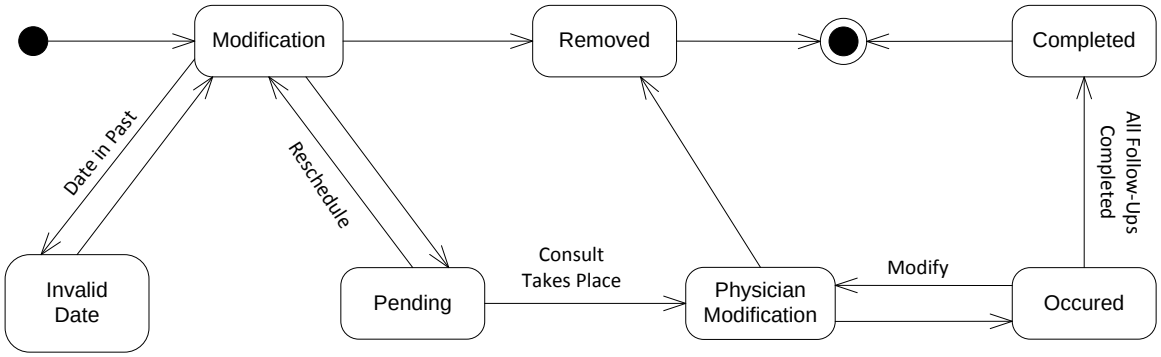


Figure 10 – State Machine Diagram – Resultant Follow-up

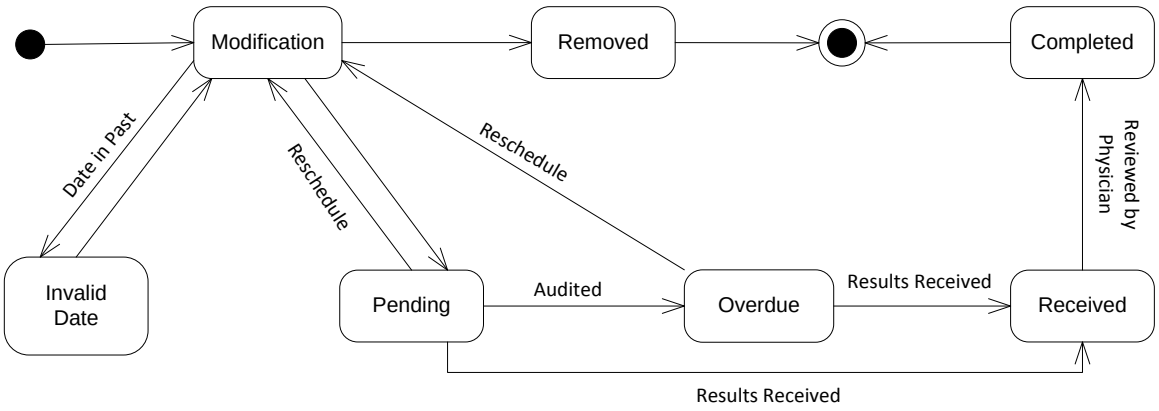


Figure 11 – State Machine Diagram – Return Consultation

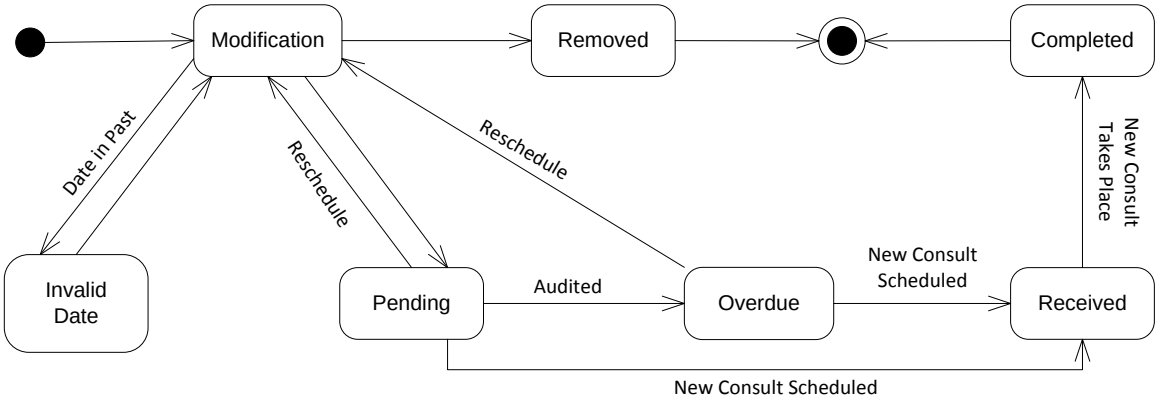
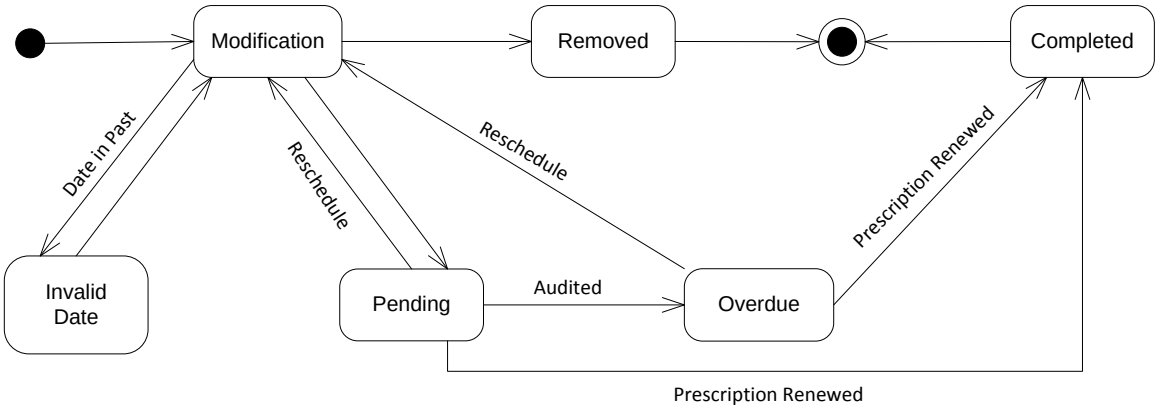


Figure 12 – State Machine Diagram – Medication Renewal



6.2 Sequence Diagrams

This section illustrates interaction of system components for a selected set of use cases through the use of sequence diagrams. The use cases represented include:

- Creation of a patient consultation record (**CreateConsultation)**
- Editing of a patient's consultation record (**ModifyConsultation**)
- Viewing a list of patients, filtered by follow-up status and by physician (**RunFollowupStatusReport**)
- The execution of the system audit for identification of overdue follow-ups (**RunAudit**)

Figure 13 – Sequence Diagram – CreateConsultation

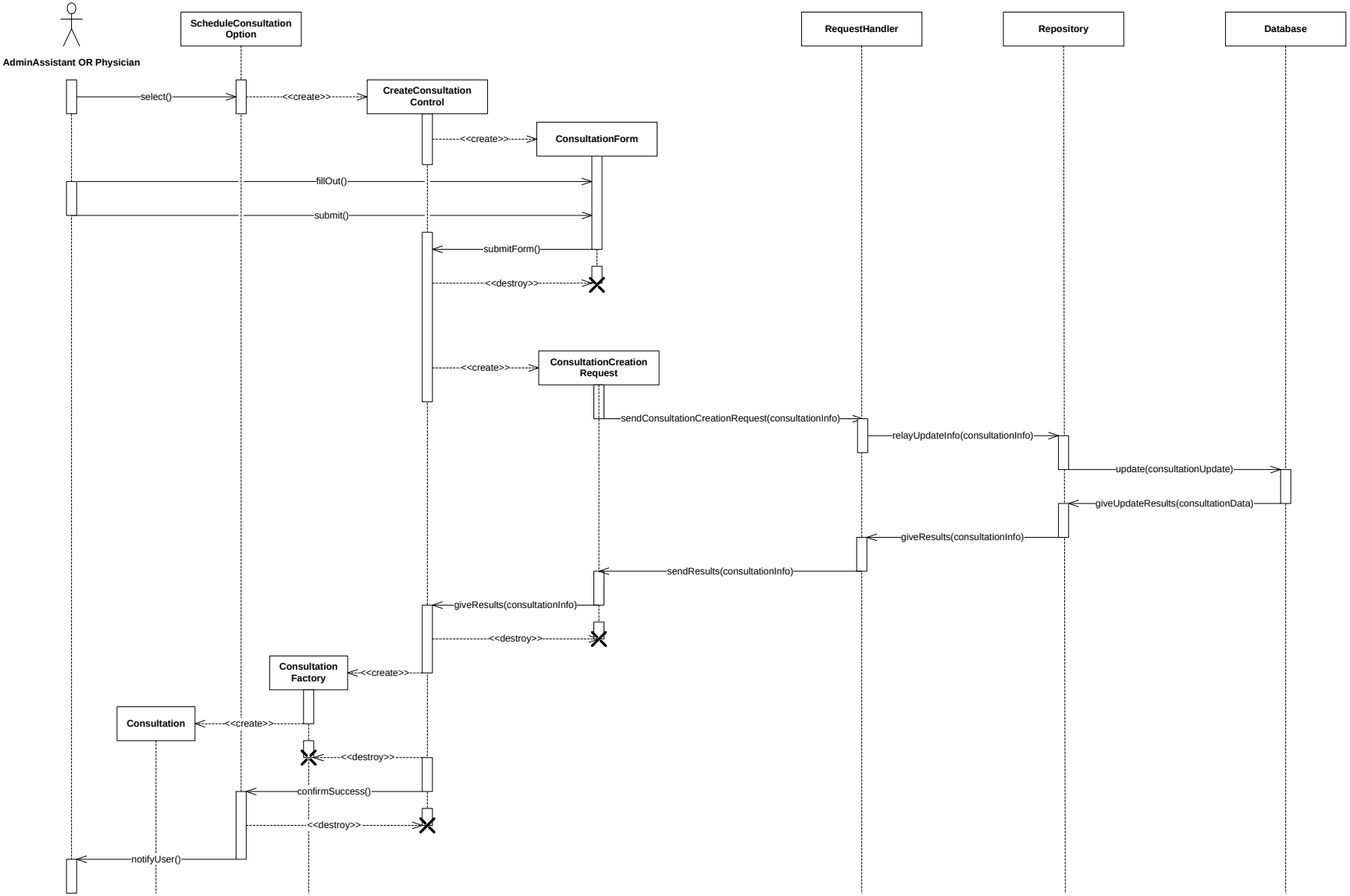


Figure 14 – Sequence Diagram – ModifyConsultation

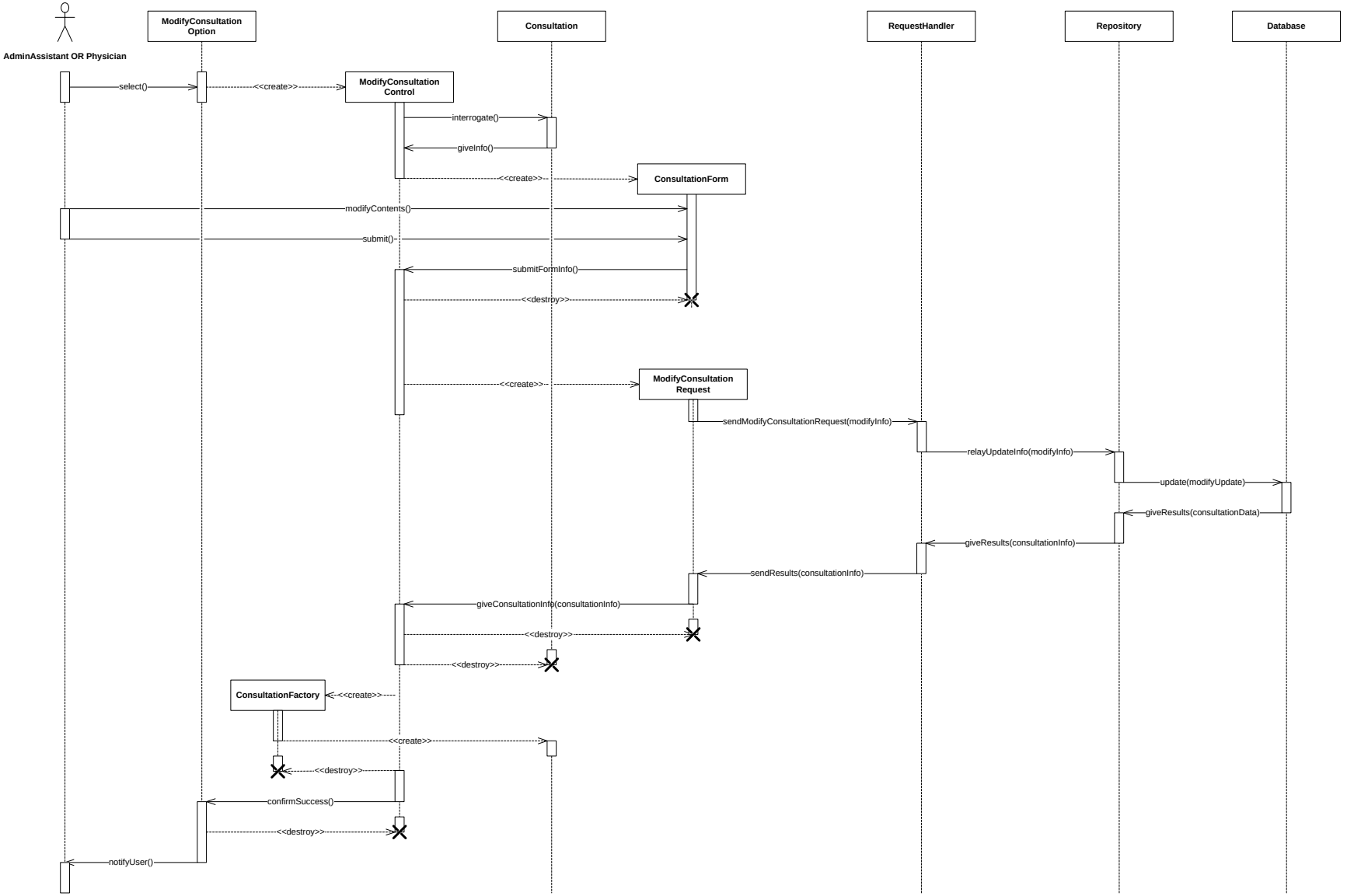


Figure 15 – Sequence Diagram – RunFollowupStatusReport

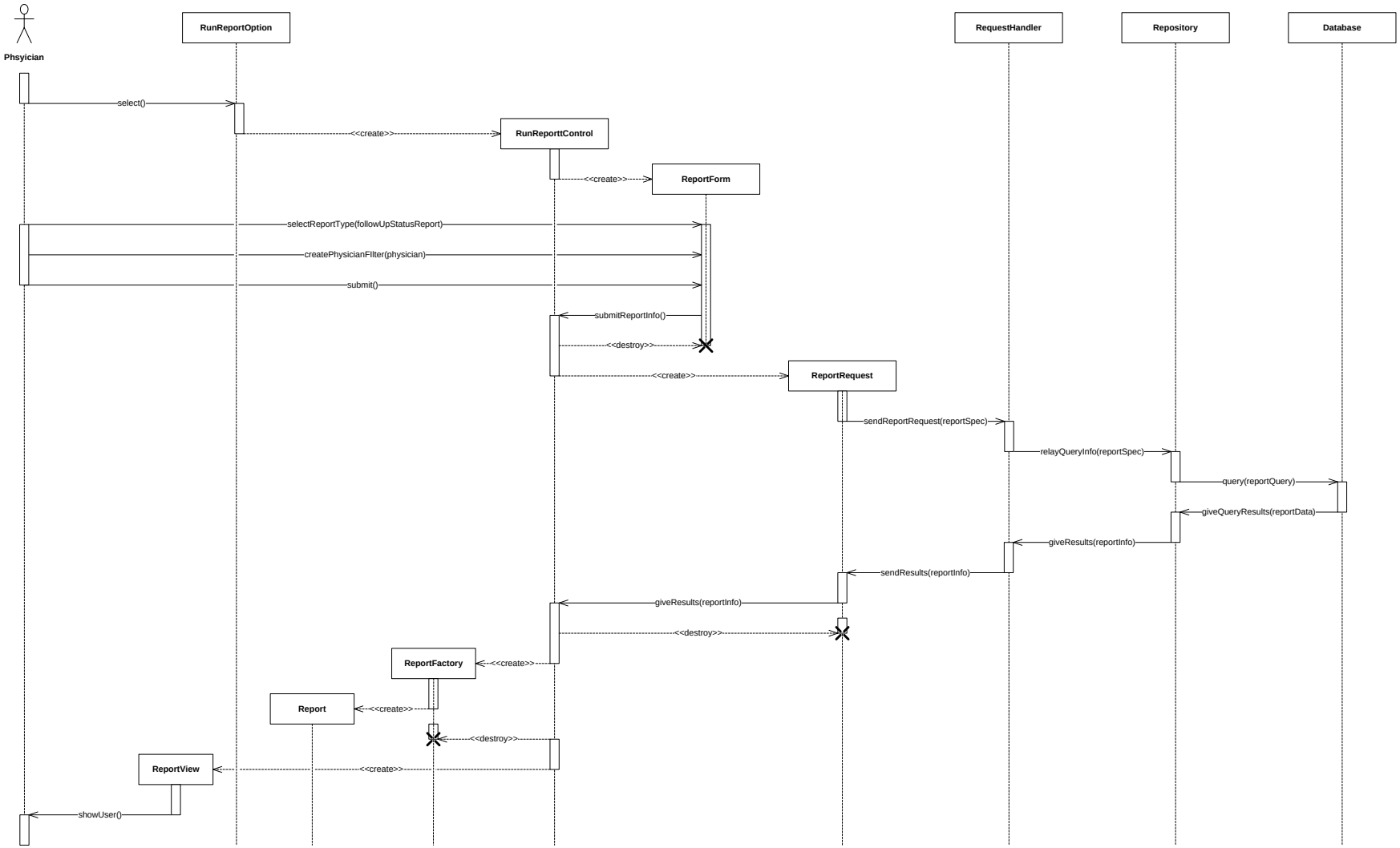


Figure 16 – Sequence Diagram – RunAudit