SOFTWARE REQUIREMENTS AND ANALYSIS SPECIFICATION

Table of Contents

1 Introduction

This section provides an overview of the entire requirement and analysis document. This document describes all data, functional and behavioral requirements for software.The goal is to develop a software for ChocAn Organization that will enable the organization to perform their tasks nonchalantly and efficiently.

1.1 Goals and objectives

Goal

The goal of the project is to develop a system software for the ChocAn Organization which provides services to its members. Concrete aim is to develop a user interface containing a functionality specification to specify and manage services and health care providers , and to specify the general technological and structure requirements.

Objectives

The objective of the project is to find ways of handling communication between the providers and the system and making the information exchange between different actors in the system easily accessible and more efficient. The main purpose is to keep all information in one place so it becomes a helpful tool for the actors using it.

1.2 Statement of scope

A ChocAn software is a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of the organization and that are developed from a common set of core assets in a prescribed way. The organization deals with day to day transactions and communication between its members ,providers and operators. ChocAn software has to manage the efficacy of the workflow.

Major inputs include:

- Add/update/delete Member
- Add/update/delete Provider
- Add/update/delete Services

Major outputs include:

- Service directory
- Reports

External inquiries:

- Member verification
- Existing member/Provider information
- Service fee

External Interface Files

- EFT data Transfer

1.3 Software context

Chocolate is consistently ranked as one of the most problematic foods for addictive-like eating behaviors. It leads to health issues like obesity. Chocoholics Anonymous is an organization dedicated to deal with these health issues by providing unlimited consultations and services to its members in exchange of the monthly fee. The health care providers like dieticians, internists, and exercise experts can enroll themselves in the organization to provide services. The organization wants a software system that is efficient in dealing day to day procedures and transactions. It would handle the billing services with accuracy and efficacy and would be able to perform its major functions in interactive mode. The software will be environment friendly in a way that it would help providers and operators to perform major tasks electronically.

1.4 Major constraints

We followed waterfall approach throughout the entire project as the project's requirements were clearly defined in the project document it helped the team to:

- Focus on understanding the core desired functionality for the process specification
- Conservatively manage project scope based on team member availability

- Establish a working relationship within the project team, and clarify roles and responsibilities
- The design and implementation phase are mostly assumptions due to the absence of programming.

2 Usage scenario

2.1 User profiles

- **Members / Potential Members :** ChocAn members who joined Chocoholics Anonymous to receive health service, Potential Members are the person who wants to join ChocAn
- **Providers :** Health Care provider who will be responsible for providing health services to ChocAn members
- **ChocAn Operator :** Operator manages member, maintain services provided to ChocAn member and, maintain providers
- **ChocAn Manager :** Manager interacts with the system for accounting procedure for bill services, generating account payables report for EFT data transfer with ACME accounting system.

2.2 Major software functionality

- Maintain members database
- Manage Providers database
- Maintain services
- Health care provider should be able to provide services to ChocAn members using Provider Terminal
- Provider Bills ChocAn for the service provided to ChocAn members.
- Member validation.
- Weekly report generation
- On-demand reports

- Generate service/provider directory
- Managing Membership.
- Run Accounting Procedures.

2.3 Special usage considerations

-NA -

3 Data Model and Description

3.1 Data Description

The primary data objects involved in the application are as follows:

- **Provider:** A master data entity is tracked representing a provider of ChocAn services. Properties include the unique provider number and contact information.
- **Member**: A master data entity is tracked representing a member of ChocAn. Properties include the unique member number, member status, and contact information.
- **Service Directory**: A master data entity is tracked representing a service provided by ChocAn. Properties include the service name, code, and fee paid for the service.
- **Service Log**: A transactional entity is tracked representing an instance of a service provided to a member. Properties include the service date, fee charged, and additional comments.

3.1.1 Entity-Relationship Diagram

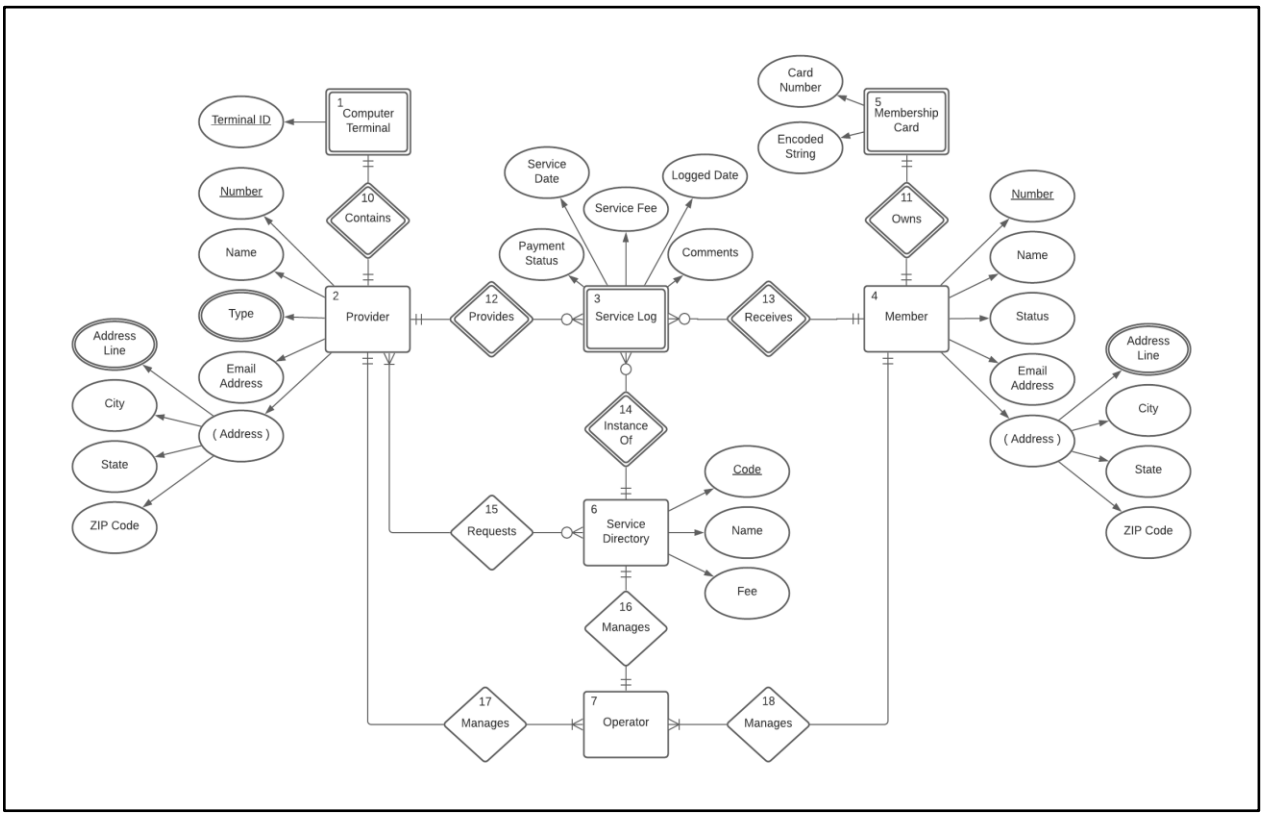


Figure X: Build ERD

Traced Requirements

1 Use Case 1: Manage Members

1.1 Operators can add, delete, or update Members: 7, 18, 4

2 Use Case 2: Manage Providers

2.1 Operators can add, delete, or update Providers: 7, 17, 2

3 Use Case 3: Manage Services

3.1 Operators can add, delete, or update Services: 7, 16, 6

4 Use Case 4: Manage Session

4.1 Providers can initiate a session at a terminal: 2, 10, 1

5 Use Case 5: Provide Services

5.1 Providers can provide services to members: 1, 12, 3, 13, 4

6 Use Case 6: Bill Services

6.1 Providers can log services: 4, 13, 3

7 Use Case 7: Validate Member

7.1 Membership status is tracked: 4

7.2 Members have a member ID and card: 4, 11, 5

8 Use Case 8: Request Provider Directory

8.1 Providers can request the provider directory: 2, 15, 6

9 Use Case 9: Receive Membership Updates

9.1 Membership status is tracked and can be updated: 4

10 Use Case 10: Run Accounting Procedure

10.1 Reports email services provided to members: 3, 13, 4, 14, 6

10.2 Reports email services provided by providers: 3, 12, 2, 14, 6

10.3 Reports write payment information to disk: 3, 14, 6

3.1.2 Data Flow Diagrams

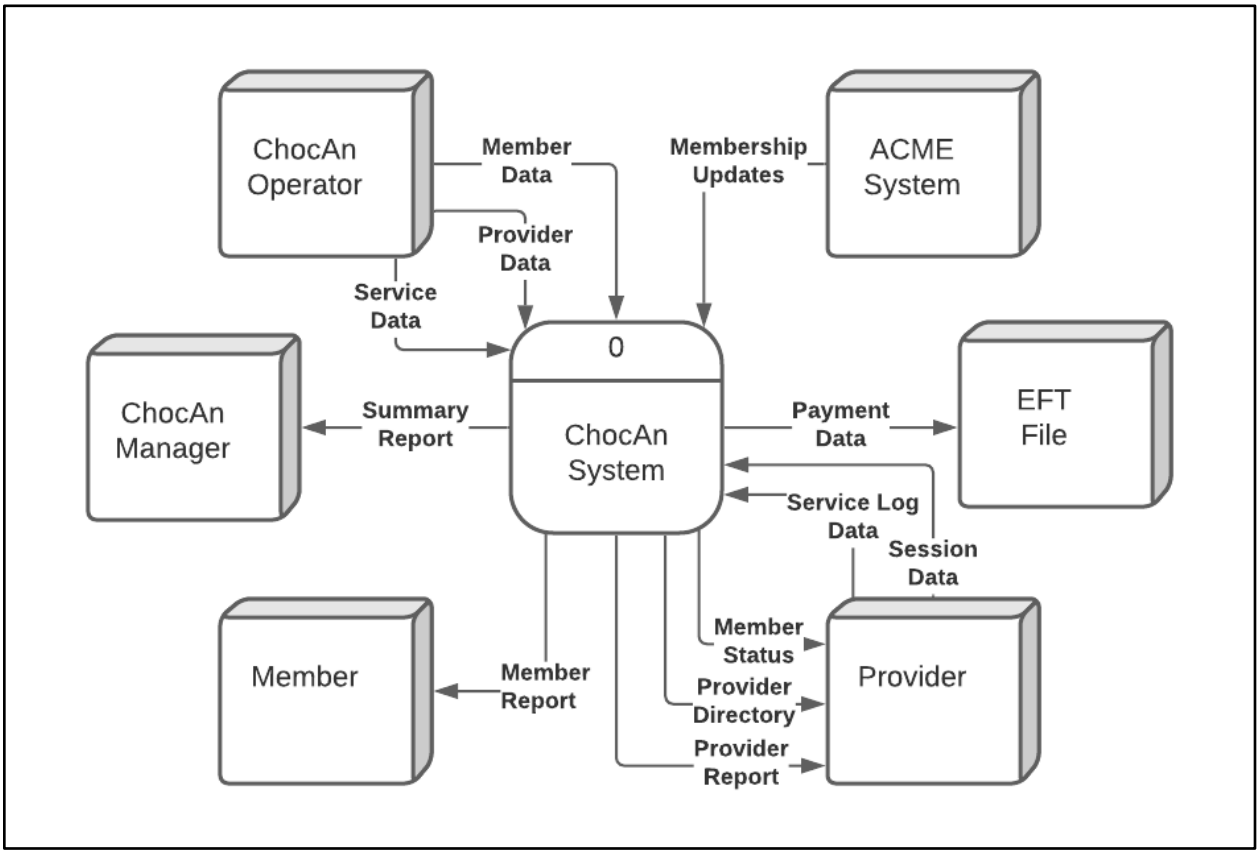


Figure X: Context Data Flow Diagram

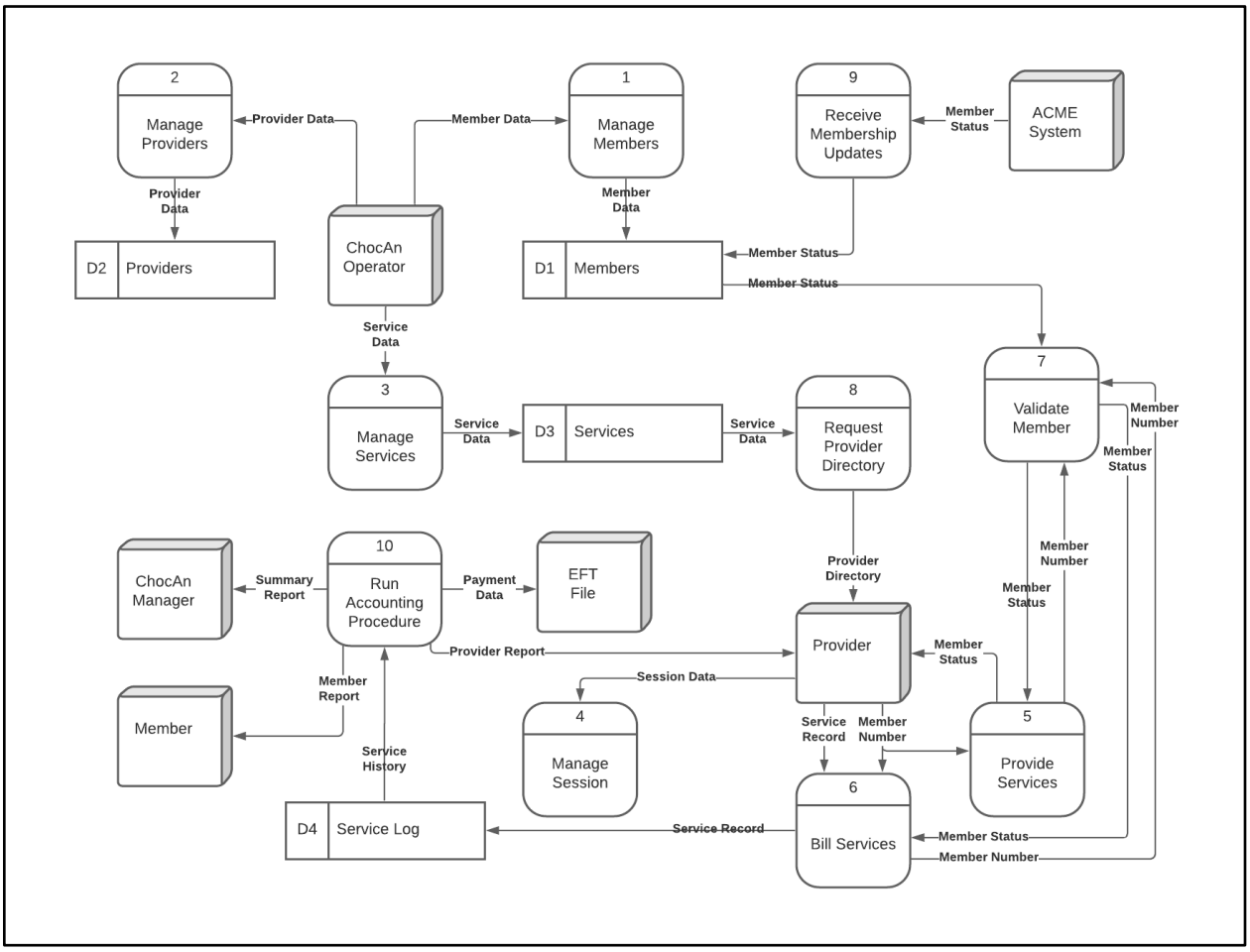


Figure X: 0 Data Flow Diagram

3.1.3 Object Relationships

Membership Update Interface	
Description: Interface into ChocAn for membership updates from ACME system	
Type: Boundary	
Responsibility	**Collaborators**
Update membership status in ChocAn	Member Data Store

Figure X: Membership Update Interface CRC Card

Operator Interface	
Description: Interface into ChocAn for operators to manage master data	
Type: Boundary	
Responsibility	**Collaborators**
Manage Member Data	Member Data Store
Manage Provider Data	Provider Data Store
Manage Service Data	Service Directory

Figure X: Operator Interface CRC Card

Manager Interface	
Description: Interface into ChocAn for managers to request reports	
Type: Boundary	
Responsibility	**Collaborators**
Request Report Generation	Generate Reports Class

Figure X: Manager Interface CRC Card

Terminal Interface	
Description: Interface into ChocAn for operators to manage master data	
Type: Boundary	
Responsibility	**Collaborators**
Accept member number for validation	Validate Member Class
Accept service log data to bill	Submit Service Class
Allow provider directory to be requested	Request Provider Directory Class
Manage provider session	Manage Session Class

Figure X: Terminal Interface CRC Card

Generate Reports	
Description: Generates user-facing and EFT reports based on service log data	
Type: Control	
Responsibility	**Collaborators**
Generate member report for each member	Member Data Store
Generate provider report for each provider	Provider Data Store
Summarize service log data	Service Directory Data Store
Generate summary report for manager	Service Log Data Store
Generate EFT report for external EFT system	Manager Interface

Figure X: Generate Reports Class CRC Card

Validate Member	
Description: Validates a member's status given their member number	
Type: Control	
Responsibility	**Collaborators**
Accept member number	Member Data Store
Return member status & reason	Terminal Interface

Figure X: Validate Member Class CRC Card

Submit Service	
Description: Submits a service log record to ChocAn	
Type: Control	
Responsibility	**Collaborators**
Accept service log data	Service Log Data Store
Validate service data	Terminal Interface
Return success message	

Figure X: Submit Service Class CRC Card

Request Provider Directory	
Description: Requests a provider directory to be emailed to the provider	
Type: Control	
Responsibility	**Collaborators**
Compile provider directory from services	Service Directory Data Store
Format provider directory document	Terminal Interface
Send email with attachment	

Figure X: Request Provider Directory Class CRC Card

Manage Session	
Description: Manages session data for a ChocAn provider terminal	
Type: Control	
Responsibility	**Collaborators**
Accept provider number	Terminal Interface
Maintain session data	
Expire session based on inactivity	

Figure X: Manage Session Class CRC Card

Member	
Description: Data about a ChocAn member	
Type: Entity	
Responsibility	**Collaborators**
Store member details	Membership Update Interface
Allow members to be added, updated, and deleted	Operator Interface
Allow member details to be queried	Generate Reports Class
Allow membership status to be changed	Validate Member Class

Figure X: Member Entity CRC Card

Provider	
Description: Data about a ChocAn provider	
Type: Entity	
Responsibility	**Collaborators**
Store provider details	Operator Interface
Allow providers to be added, updated, and deleted	Generate Reports Class
Allow provider details to be queried	

Figure X: Provider Entity CRC Card

Service	
Description: Data about a ChocAn service	
Type: Entity	
Responsibility	**Collaborators**
Store service details	Operator Interface
Allow services to be added, updated, and deleted	Generate Reports Class
Allow service details to be queried	Request Provider Directory Class

Figure X: Service Directory Entity CRC Card

Service Log	
Description: Data about an instance of a ChocAn service provided to a member	
Type: Entity	
Responsibility	**Collaborators**
Store service log details	Submit Service Class
Allow service logs to be added, updated, and deleted	Generate Reports Class
Allow service log details to be queried	

Figure X: Service Log Entity CRC Card

Member Report	
Description: Displays data about services provided to a member in the previous period	
Type: Boundary	
Responsibility	**Collaborators**
Format and display member's service log data	Generate Reports Class

Figure X: Member Report CRC Card

Provider Report	
Description: Displays data about services provided by a provider in the previous period	
Type: Boundary	
Responsibility	**Collaborators**
Format and display provider's service log data	Generate Reports Class

Figure X: Provider Report CRC Card

Summary Report	
Description: Displays summarized data about all services provided in the previous period	
Type: Boundary	
Responsibility	**Collaborators**
Format and display summarized service log data	Generate Reports Class

Figure X: Summary Report CRC Card

EFT Report	
Description: Summarizes payments to be made to a provider for services in a period	
Type: Boundary	
Responsibility	**Collaborators**
Format summarized payment data for file	Generate Reports Class

Figure X: EFT Report CRC Card

3.1.4 Complete data model

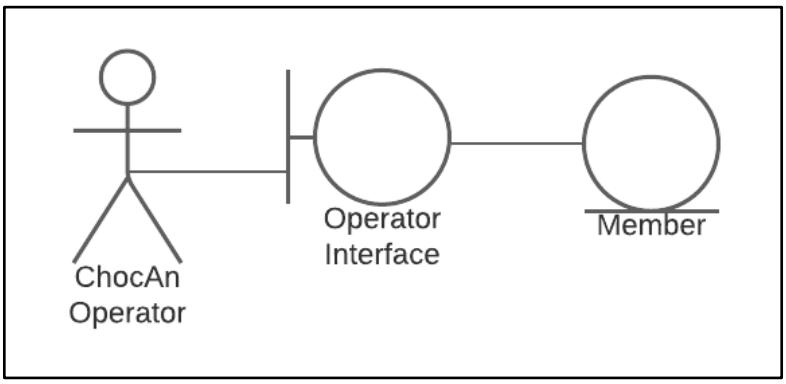


Figure X: Robustness diagram for 1: Manage Members use case

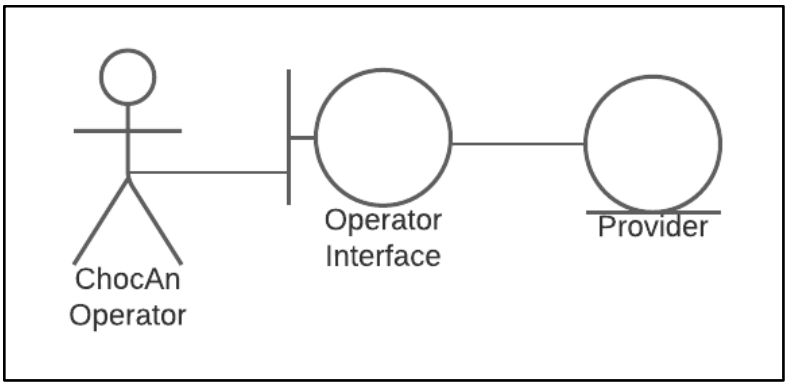


Figure X: Robustness diagram for 2: Manage Providers use case

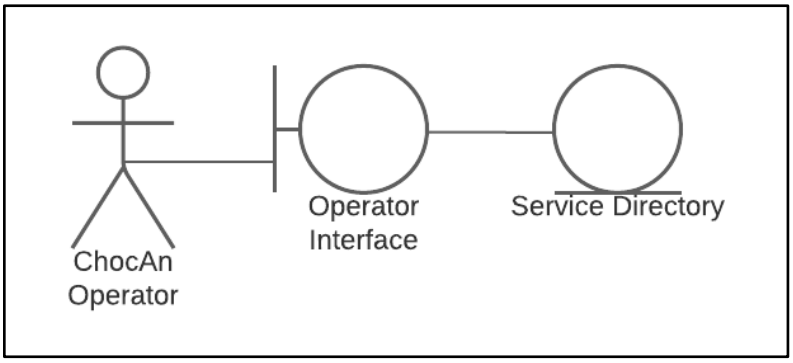


Figure X: Robustness diagram for 3: Manage Services use case

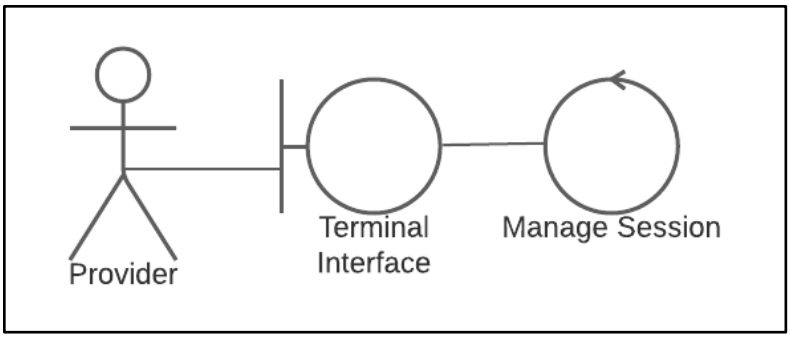


Figure X: Robustness diagram for 4: Manage Session use case

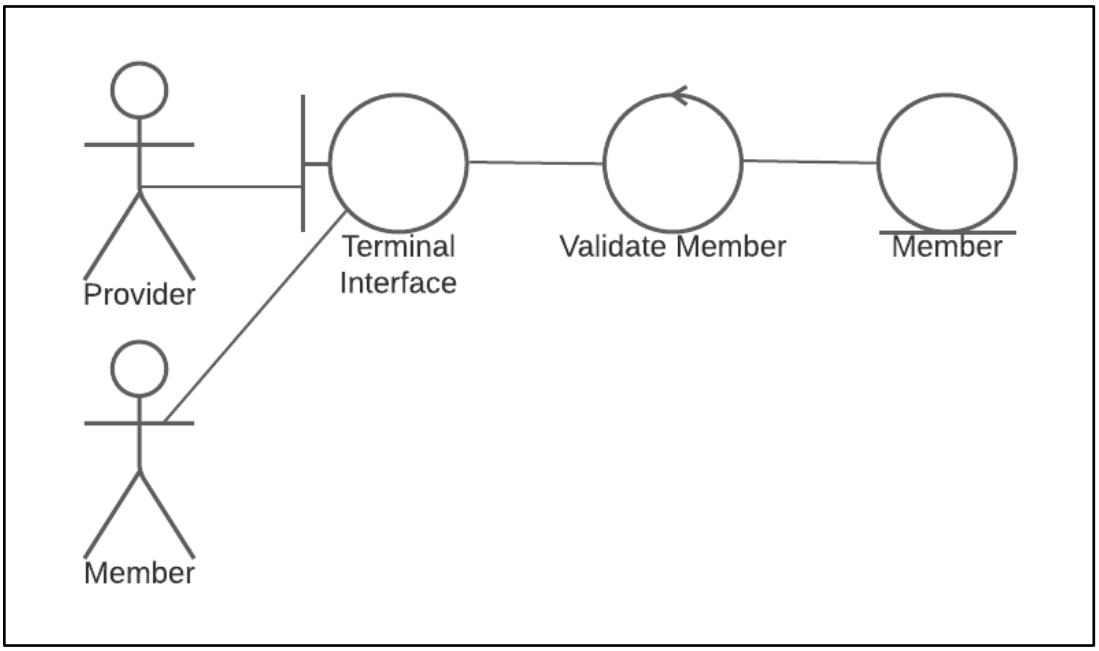


Figure X: Robustness diagram for 5: Provide Services use case

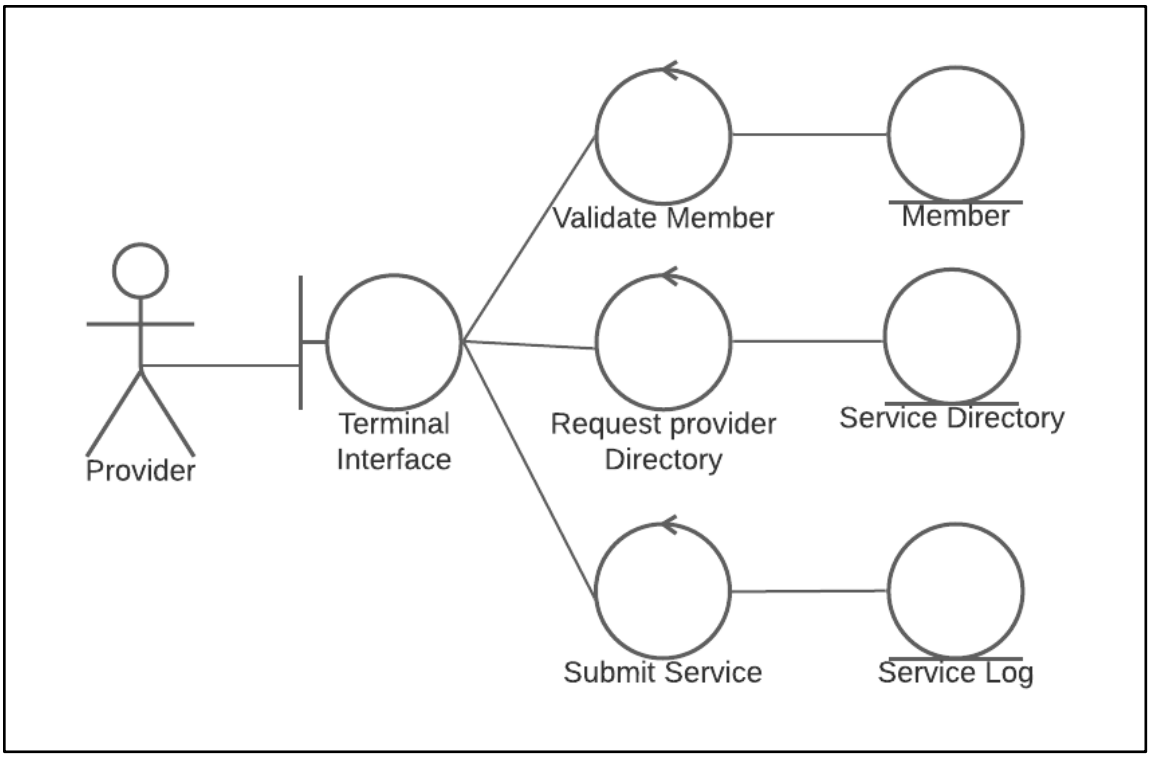


Figure X: Robustness diagram for 6: Bill Services use case

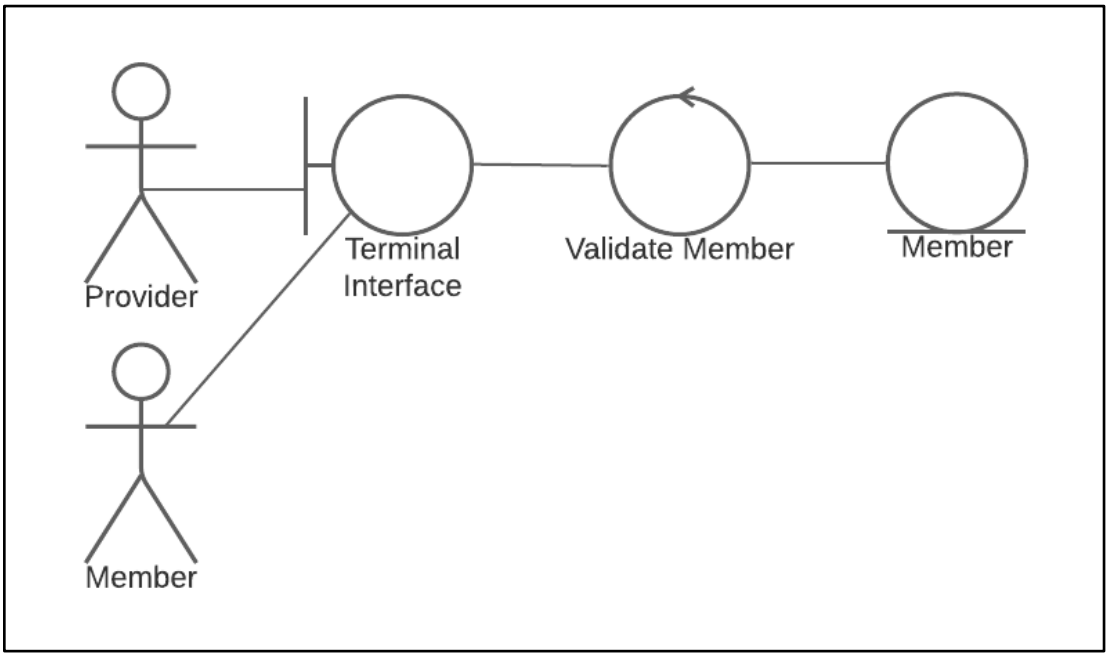


Figure X: Robustness diagram for 7: Validate Member use case

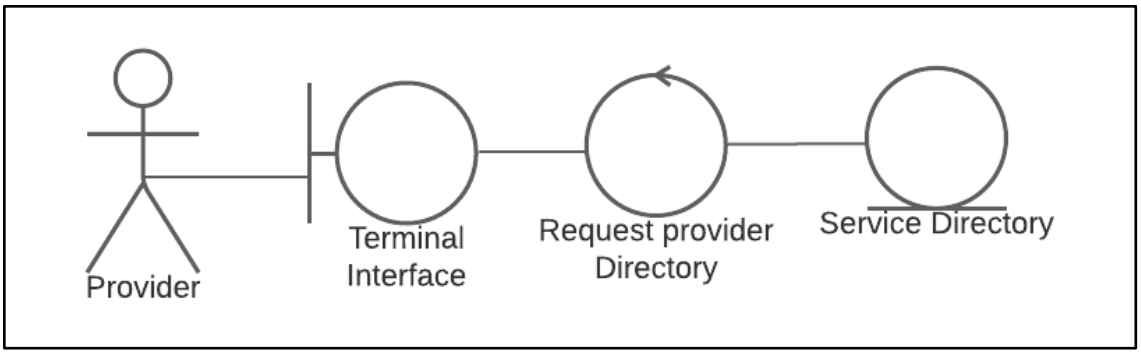


Figure X: Robustness diagram for 8: Request Provider Directory use case

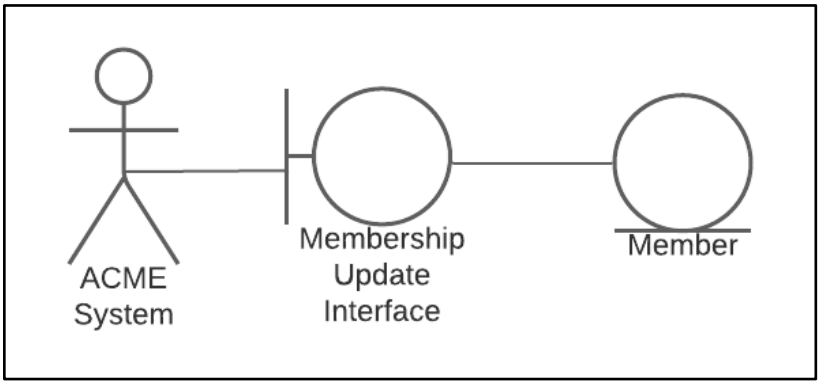


Figure X: Robustness diagram for 9: Receive Membership Updates use case

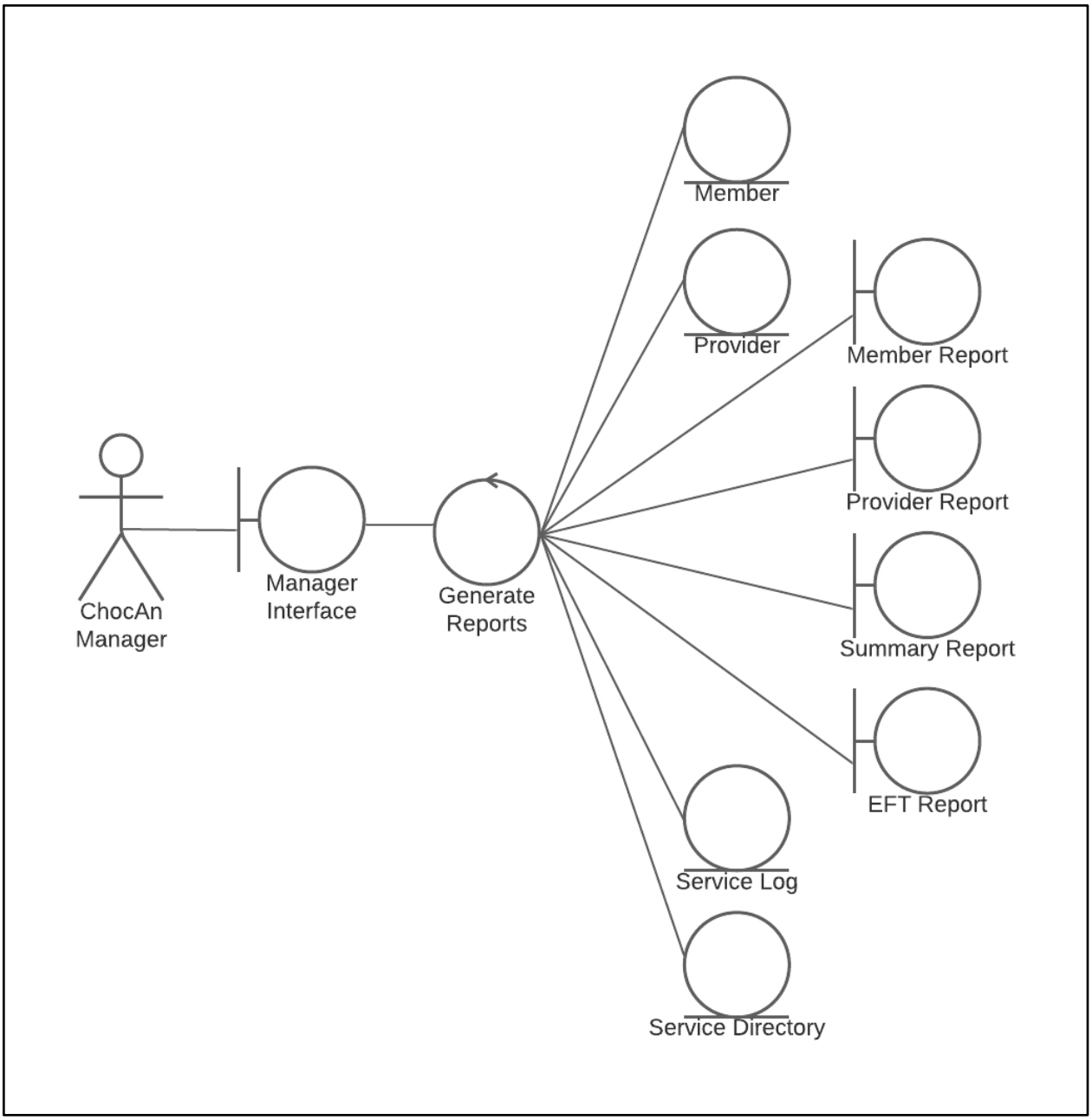


Figure X: Robustness diagram for 10: Run Accounting Procedure use case

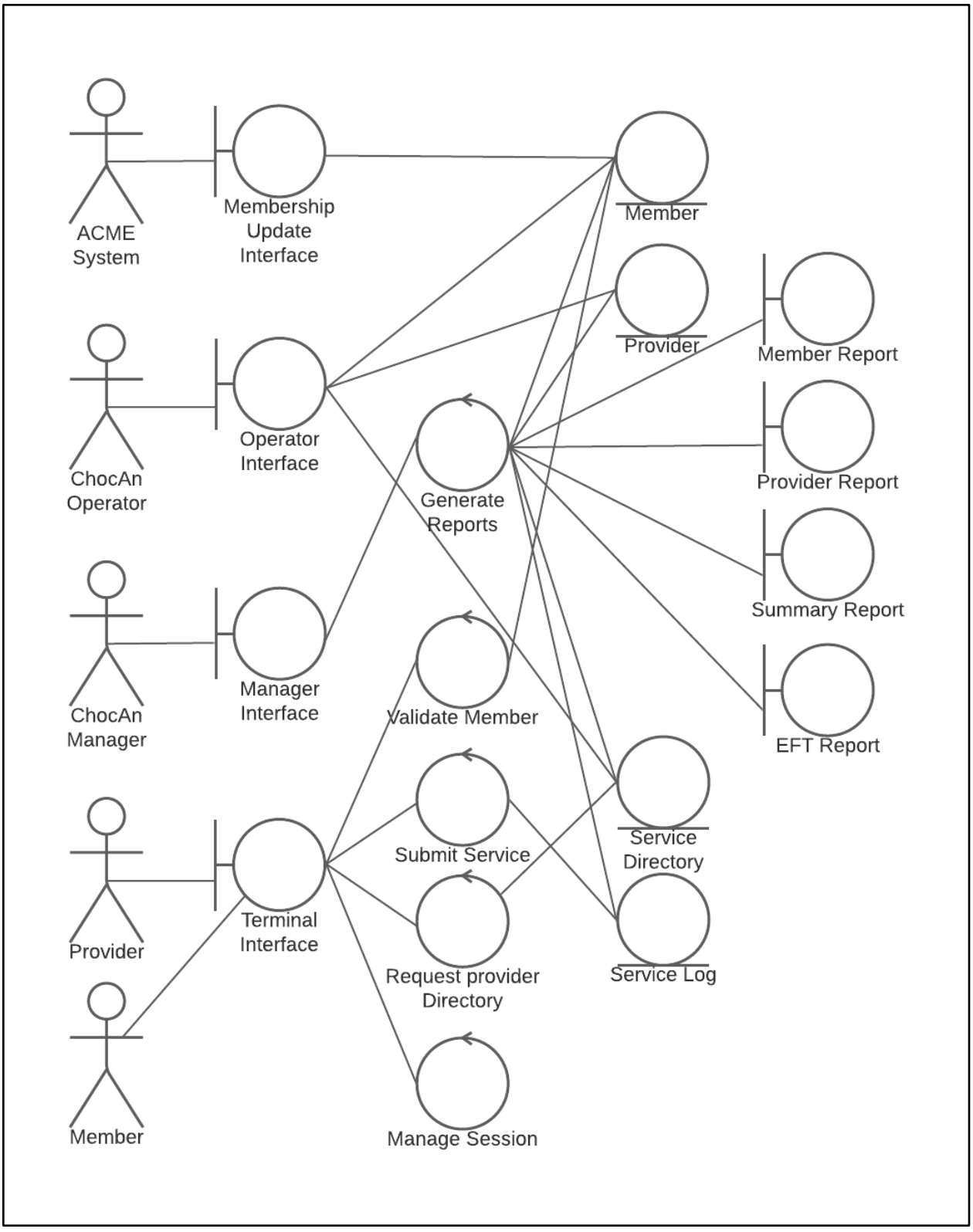


Figure X: Class Realization Diagram

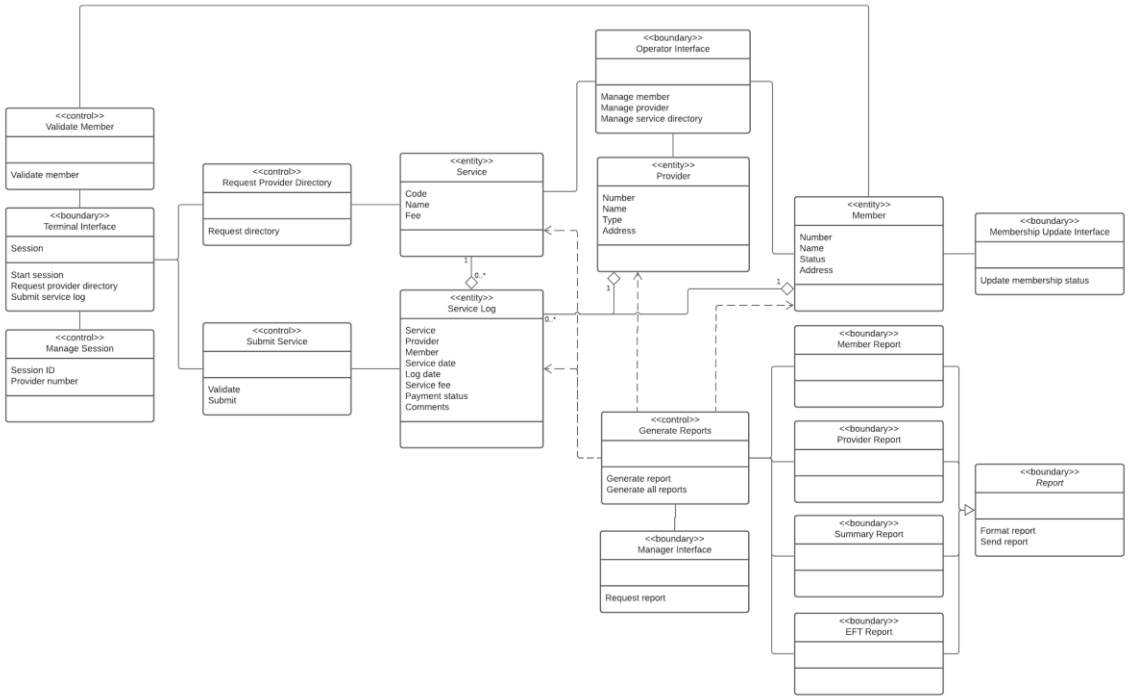


Figure X: Initial Class Diagram

3.1.5 Data dictionary

A reference to the data dictionary is provided. The dictionary is maintained in electronic form. (Data dictionary is available in DataDictionary_V1.xlsx)

4 Functional Model and Description

4.1 Use cases

In the ChocAn system we found a total of ten use cases. The following table shows the name of use case and the link of corresponding use case.

Sl. no	Name of use case	Link
1	Manage Members	**https://www.dropbox.com/scl/fi/dji6s0h4yzlyrmagslrb1/Use_Case_1_Specification.docx?dl=0&rlkey=oajvpifwwdm926uzf5zccvscx**
2	Manage Providers	**https://www.dropbox.com/scl/fi/aduo2r78qzw143nvj2pmc/Use_Case_2_Specification_ManageProvider.doc?dl=0&rlkey=4vw0dijiy91f52bctsjaohu7u**
3	Manage Services	**https://www.dropbox.com/s/s8pc4m4nwgezlcu/Use_Case_Specification_%203_Manage%20Services.pdf?dl=0**
4	Manage Session	**https://www.dropbox.com/scl/fi/e5mkj8narz5us3sv9mjq7/Use_Case_Specification_4_Manage-Session.docx?dl=0&rlkey=n7pbg0ul215hpbhkn26mlreig**
5	Provide Services	**https://www.dropbox.com/s/sqora86ajspgv66/Use_Case_Specification_5_Provide_Services.pdf?dl=0**
6	Bill Services	**https://www.dropbox.com/s/b1izt73d24bqw87/Use_Case_Specification_6_Bill_Services.pdf?dl=0**
7	Validate Member	**https://www.dropbox.com/scl/fi/3lfudz06racdkitvioaj2/Use_Case_7_Specification_ValidateMember.doc?dl=0&rlkey=89trblb23fyre9v5v9leop1fe**
8	Request Provider Directory	**https://www.dropbox.com/s/nekayy404k3b3ku/Use_Case_Specification_8_Request_Provider_Directory.pdf?dl=0**
9	Receive Membership Updates	**https://www.dropbox.com/s/fwb1ebg1pe1u3to/Use_Case_Specification_9_Receive_Membership_Updates.pdf?dl=0**
10	Run Accounting Procedure	

4.2 Software Interface Description

4.2.1 External machine interfaces

External machine interfaces will include other computers or servers that are not included in the ChocAn system but have interaction with this system. In this case the terminal equipment and machineries, the ACME accounting machineries and servers and the EFT system servers and machineries can be considered as external machine interfaces.

4.2.2 External system interfaces

The external system interfaces may include all the software systems that are not included in the ChocAn software system but ChocAn interacts with them in some point. According to the contract, our organization will only write the ChocAn data processing software. another organization will be responsible for the communications software, for designing the ChocAn provider's terminal, for the software needed by Acme Accounting Services, and for implementing the EFT component. So, the external systems are communications software, terminal system software, ACME system and EFT processing system. Acme Accounting Services, a third-party organization is responsible for financial procedures such as recording payments of membership fees, suspending members whose fees are overdue, and reinstating suspended members who have now paid what is owing. The EFT (Electronic Fund Transfer) processing system will take input data from the ChocAn data processing software and then will generate payment invoices. After generating payment invoices, EFT system will disburse the payments to several accounts payable through banking channel.

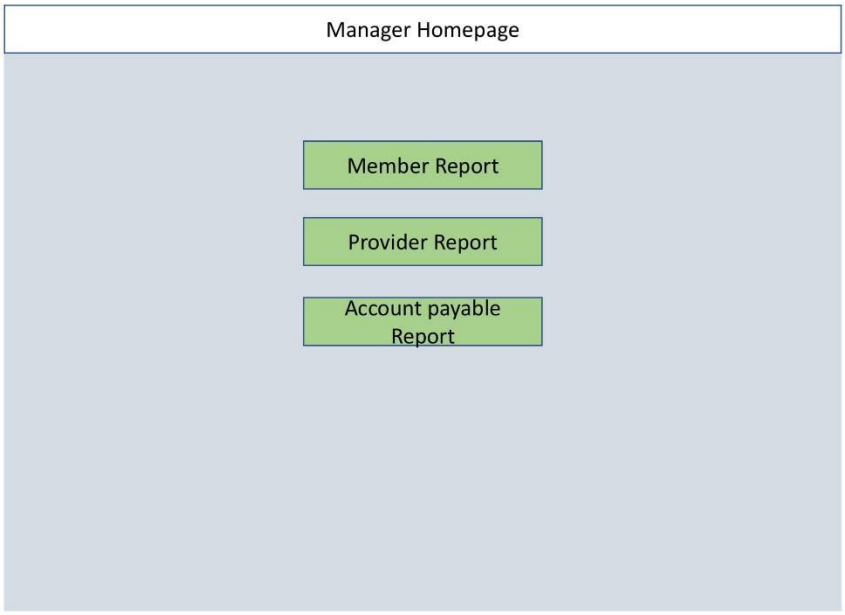
4.2.3 Human interface

4.2.3.1 User screen interface layouts

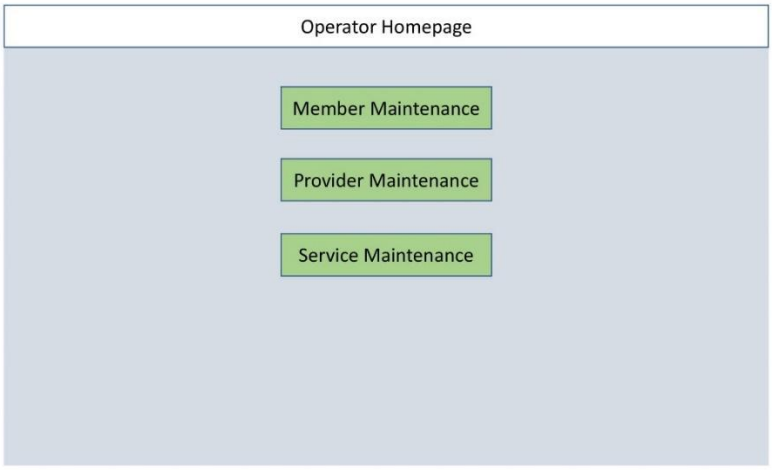
There are three user screen interfaces are portrayed in this case. The manager scree, the operator screen and the provider screen.

4.2.3.1.1 Manager Screen interface layout

The manager screen interface generates three reports, i.e. member report, provider report and account payable report. If the manager clicks on Member report, it will generate the reports of members. The reports are shown in 4.3.2.2 the report layout section.



4.2.3.1.2 Operator Screen interface layout

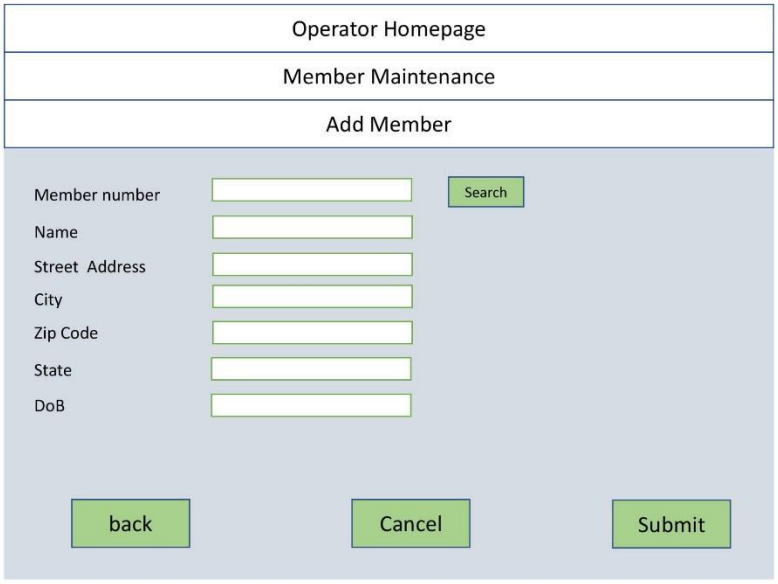


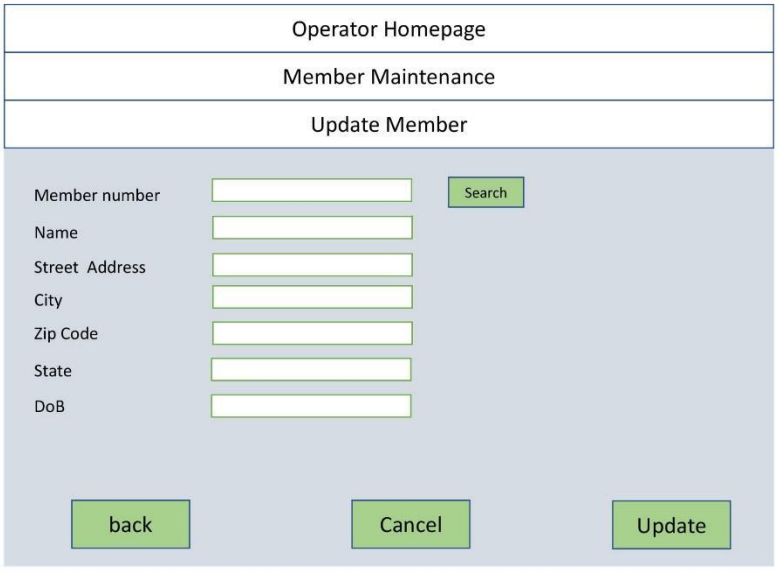
The operator screen will go through the following path:

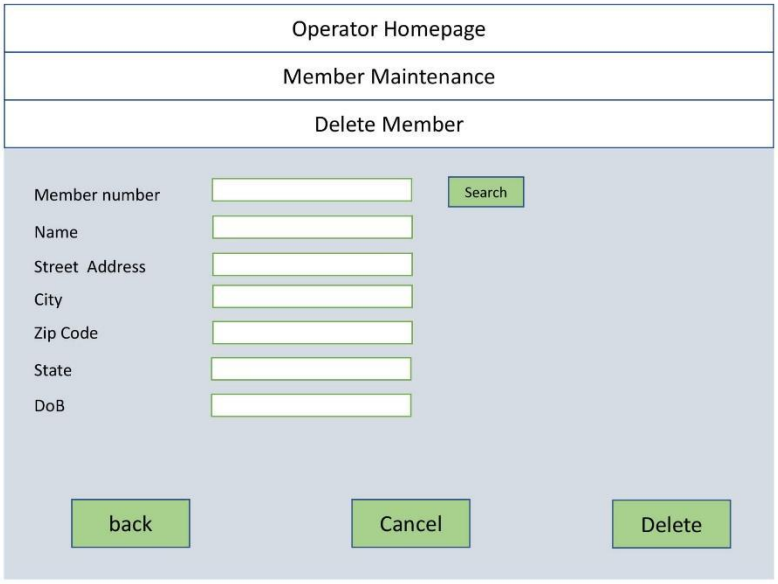
- Member maintenance
    - Add member
    - Update member
    - Delete member
- Provider maintenance
    - Add Provider
    - Update Provider
    - Delete Provider
- Service maintenance
    - Add Service
    - Update Service
    - Delete Service

The following four diagrams show the member maintenance interfaces.


Operator Homepage
Member Maintenance
Add Member
Update Member
Delete Member

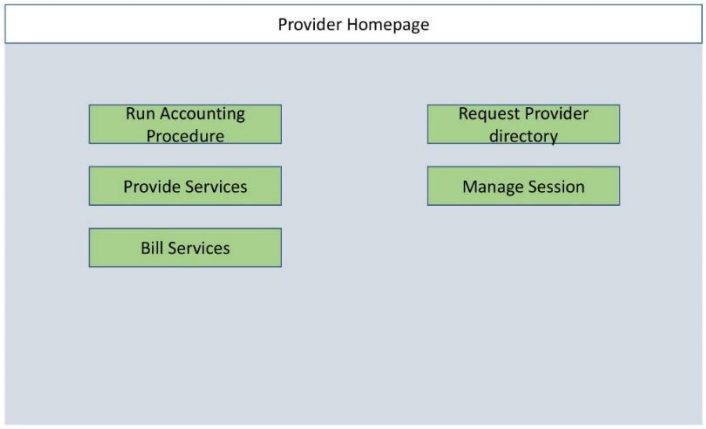

Operator Homepage
Member Maintenance
Add Member
Member number
Search
Name
Street Address
City
Zip Code
State
DoB
back
Cancel
Submit

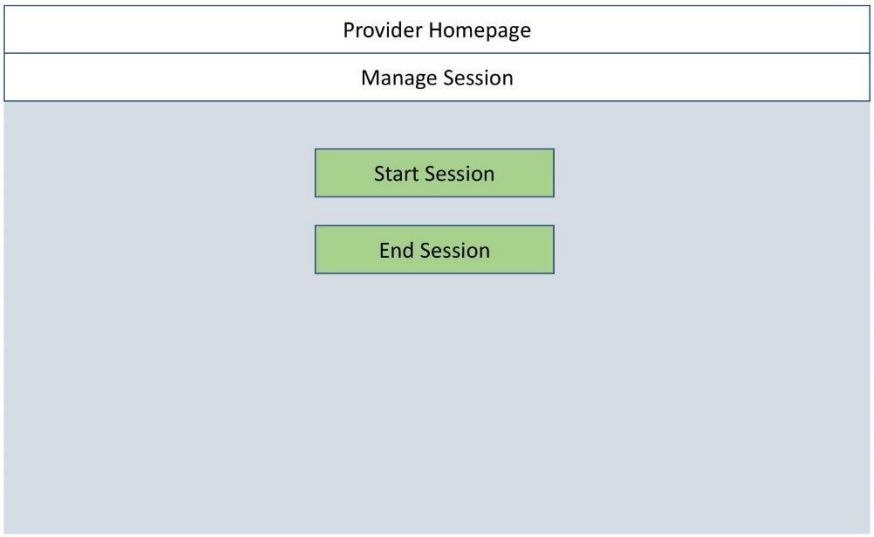

Operator Homepage
Member Maintenance
Update Member
Member number
Search
Name
Street Address
City
Zip Code
State
DoB
back
Cancel
Update

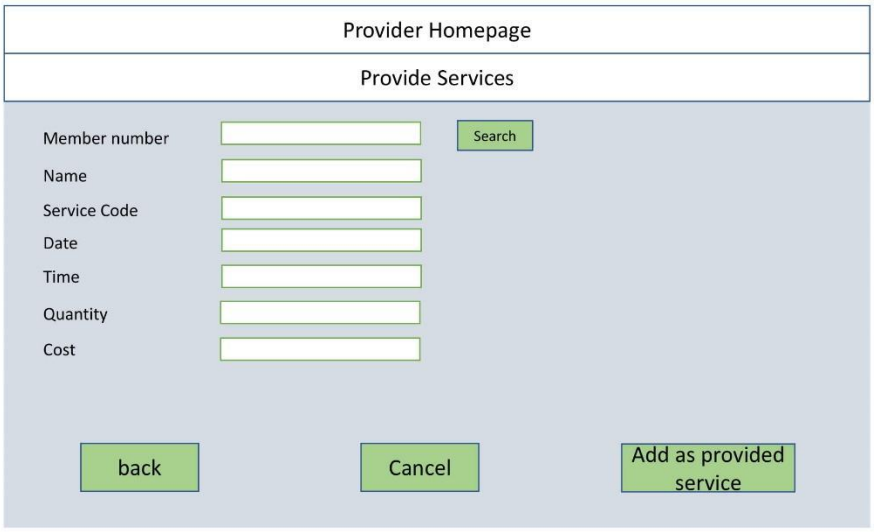

Operator Homepage
Member Maintenance
Delete Member
Member number
Search
Name
Street Address
City
Zip Code
State
DoB
back
Cancel
Delete

4.2.3.1.3 Provider Screen interface layout

The provider screen interface has five options to click. In this case, we showed manage session and provide services.



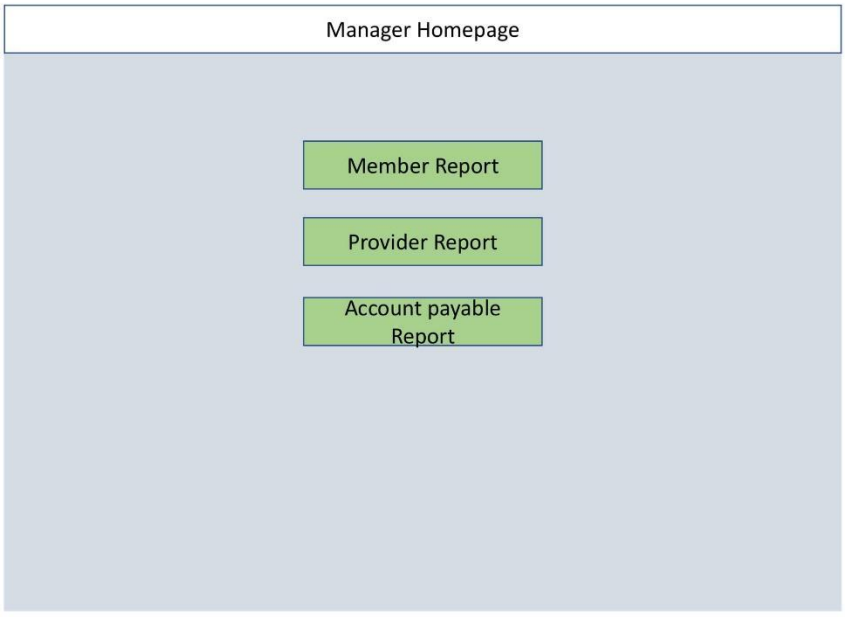




4.2.3.2 Report layouts

4.2.3.2.1 Manager Report Layouts

The manager report layout shows member report and provider report as an example.



Member Report				
Member number	**Name**	**Address**	**E-mail**	**Status**

Back

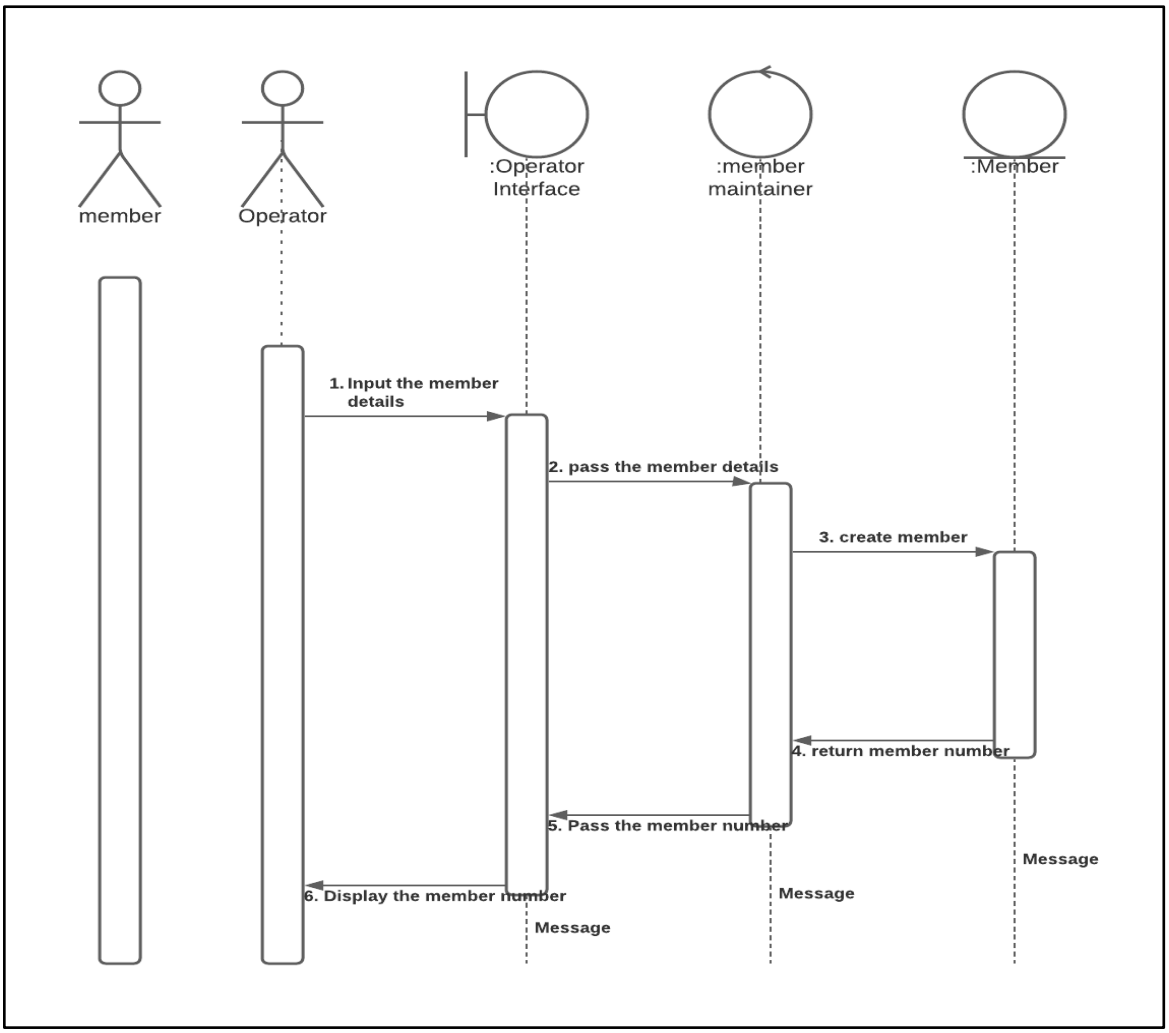
Provider Report				
Provider number	**Name**	**Address**	**E-mail**	**Status**

Back

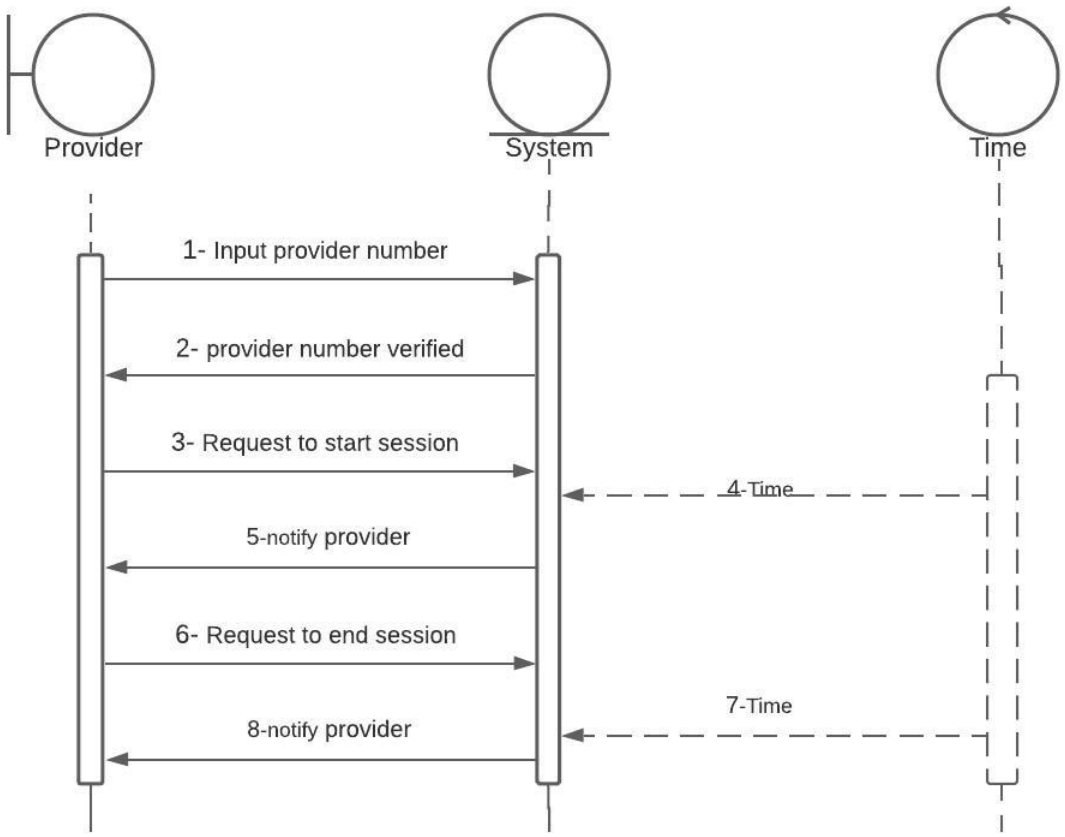
4.3 Sequence Diagrams

Used to model the class interactions needed for the use cases.

4.3.1 Member Maintenance



4.3.2 Manage Session



Sequence diagram for use_case_4_Manage_session

4.3.3 Provide Services

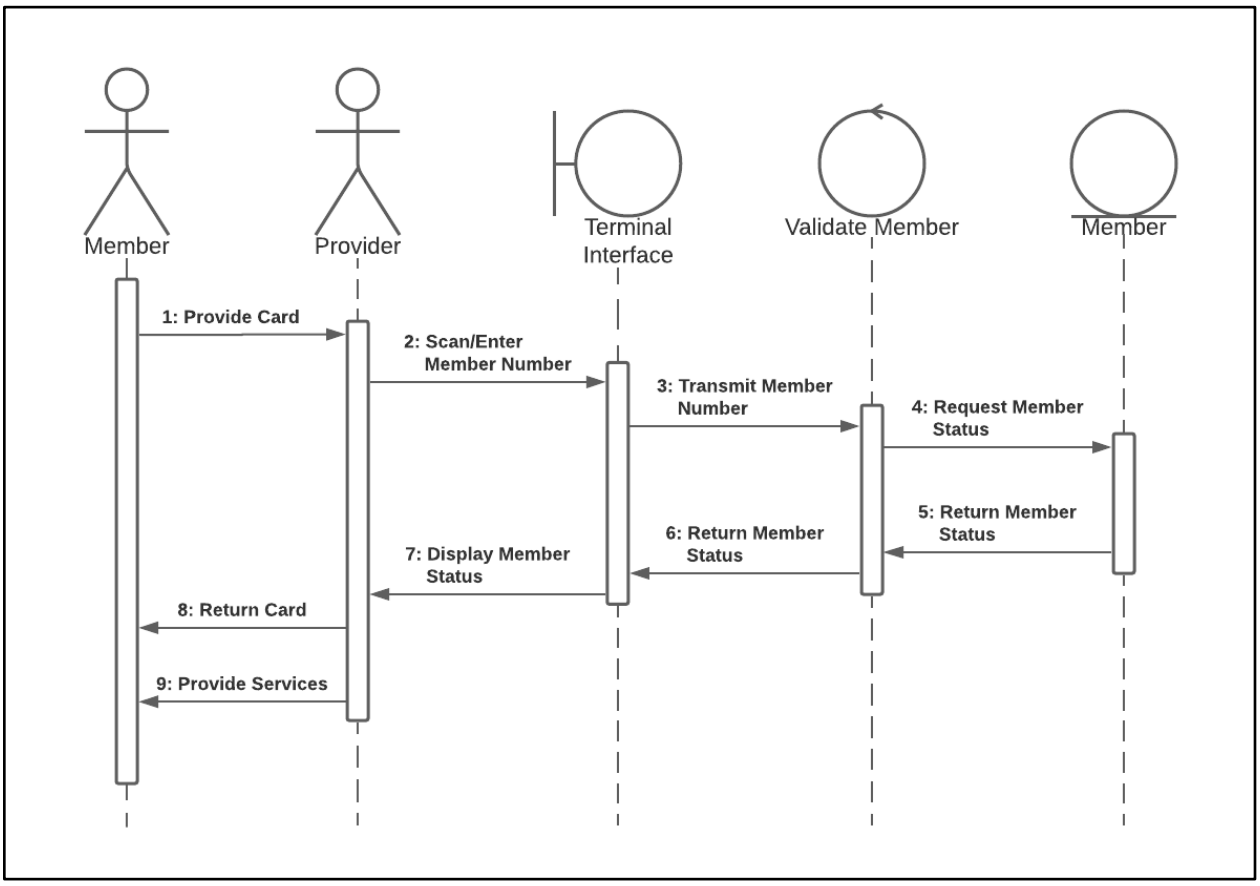
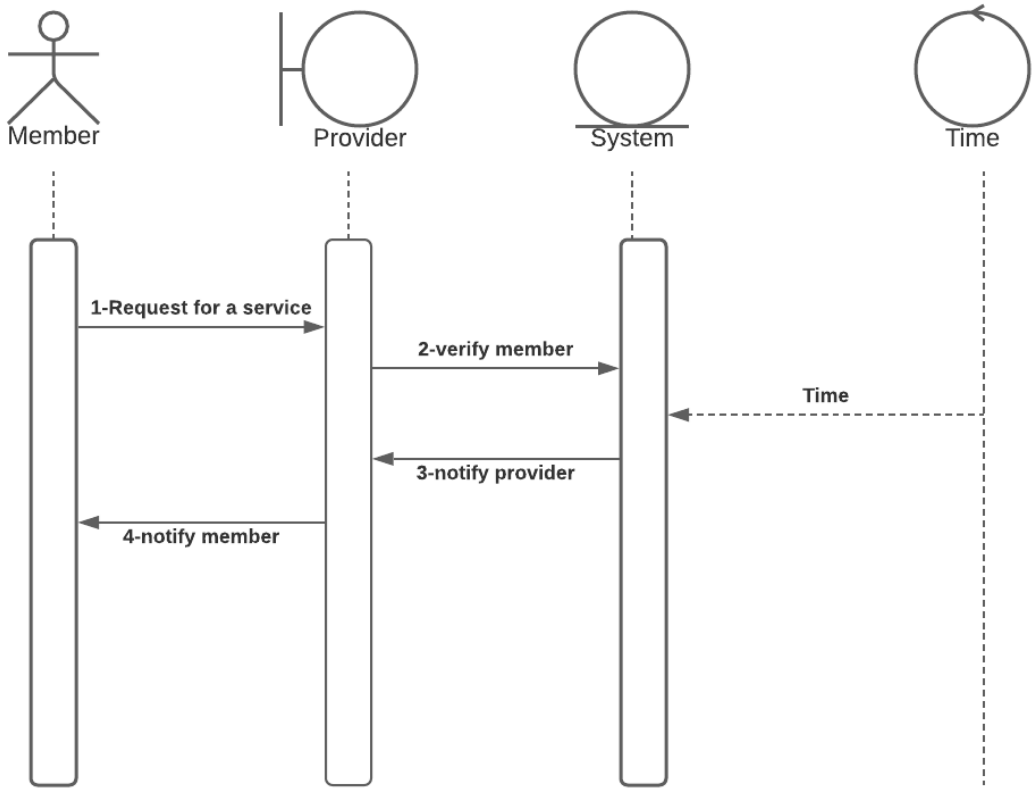


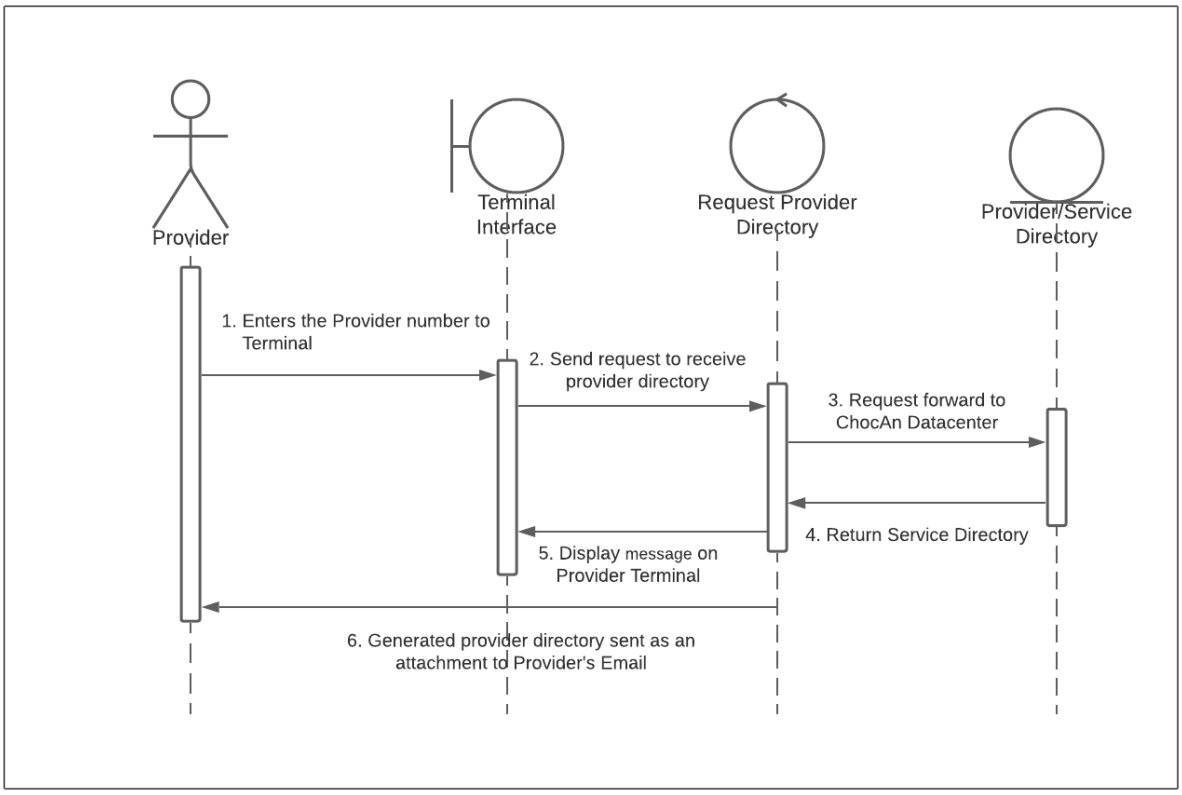
Fig X: Provide Services Sequence Diagram

4.3.4 Validate Member



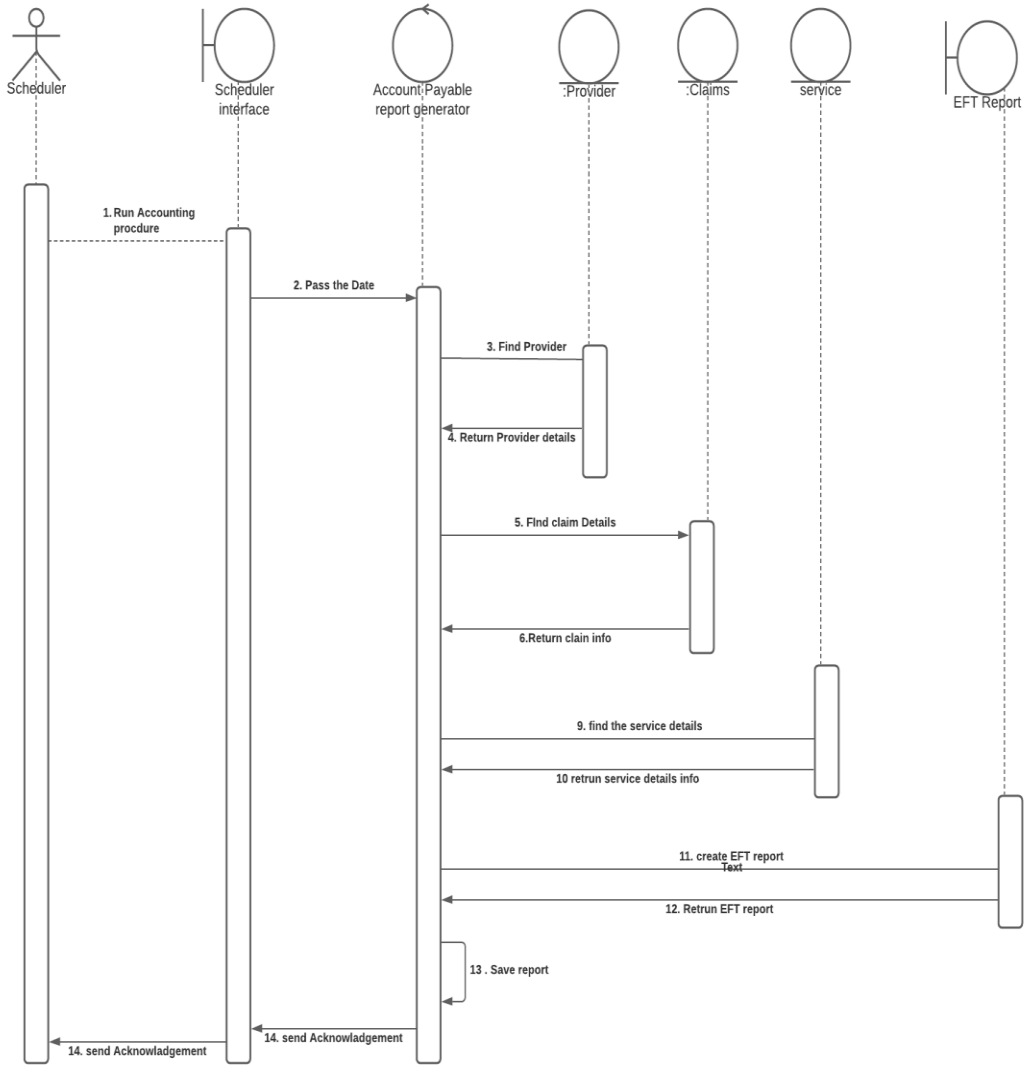
Sequence Diagram Use_case_7_Validate_Member

4.3.5 Request Provider Directory

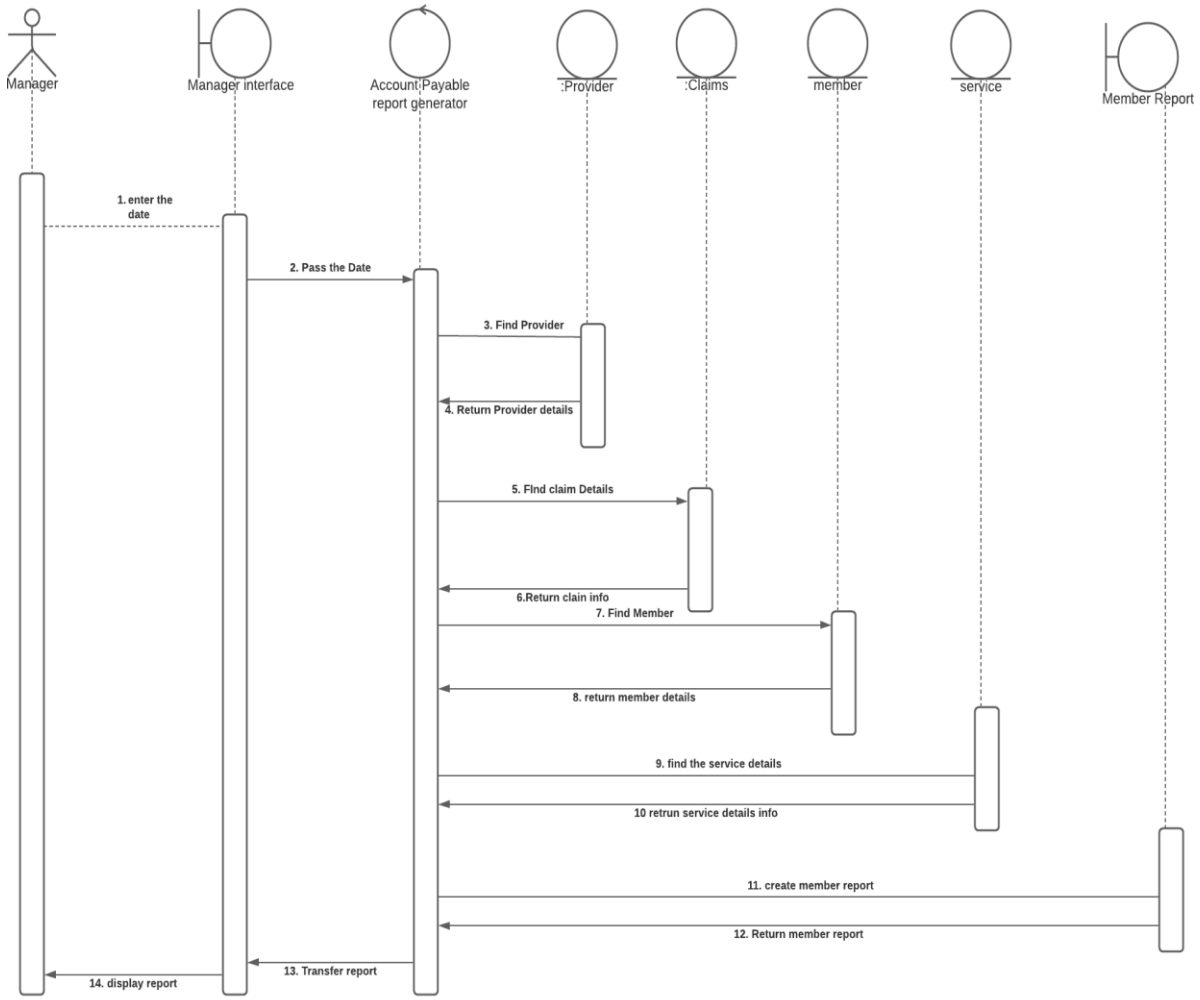


4.3.6 Run Accounting Procedure

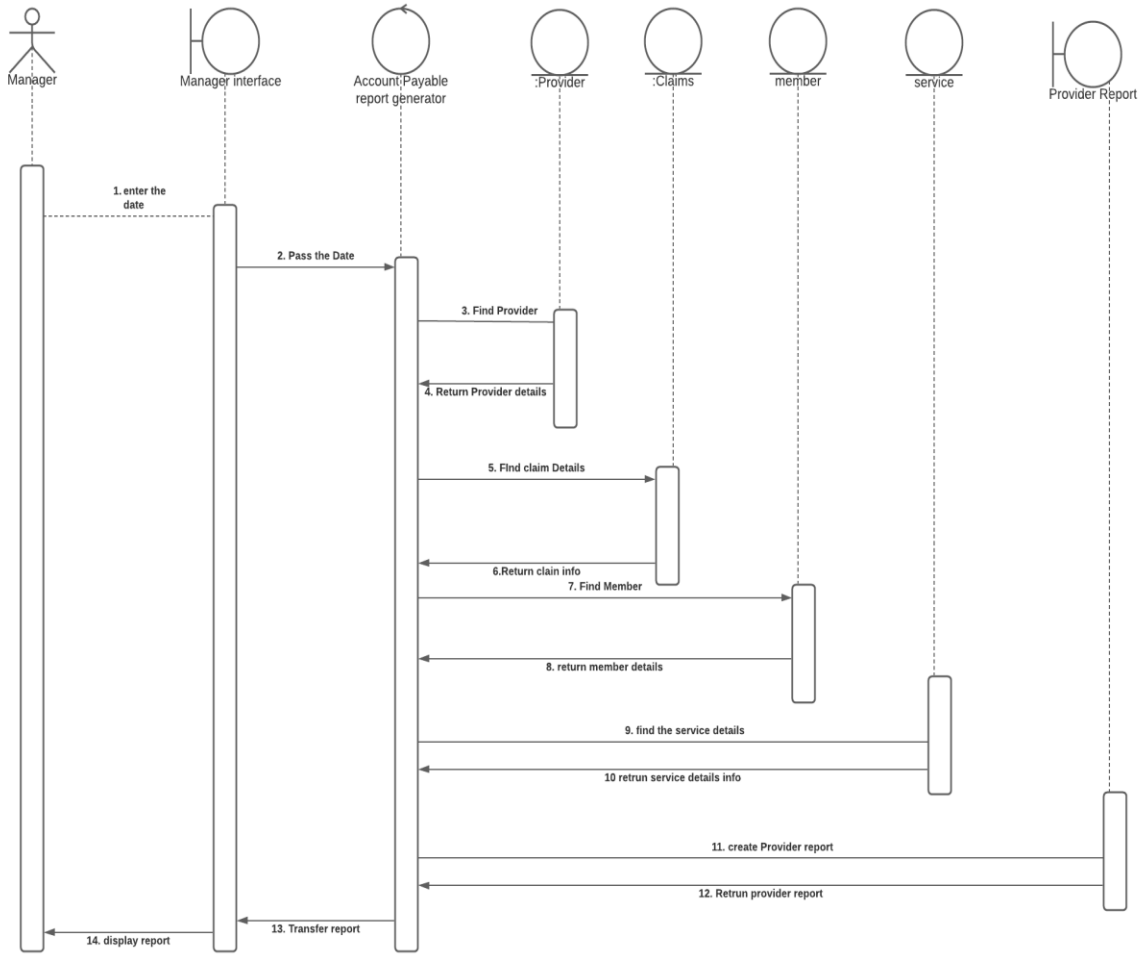
4.3.6.1 EFT Report Generation



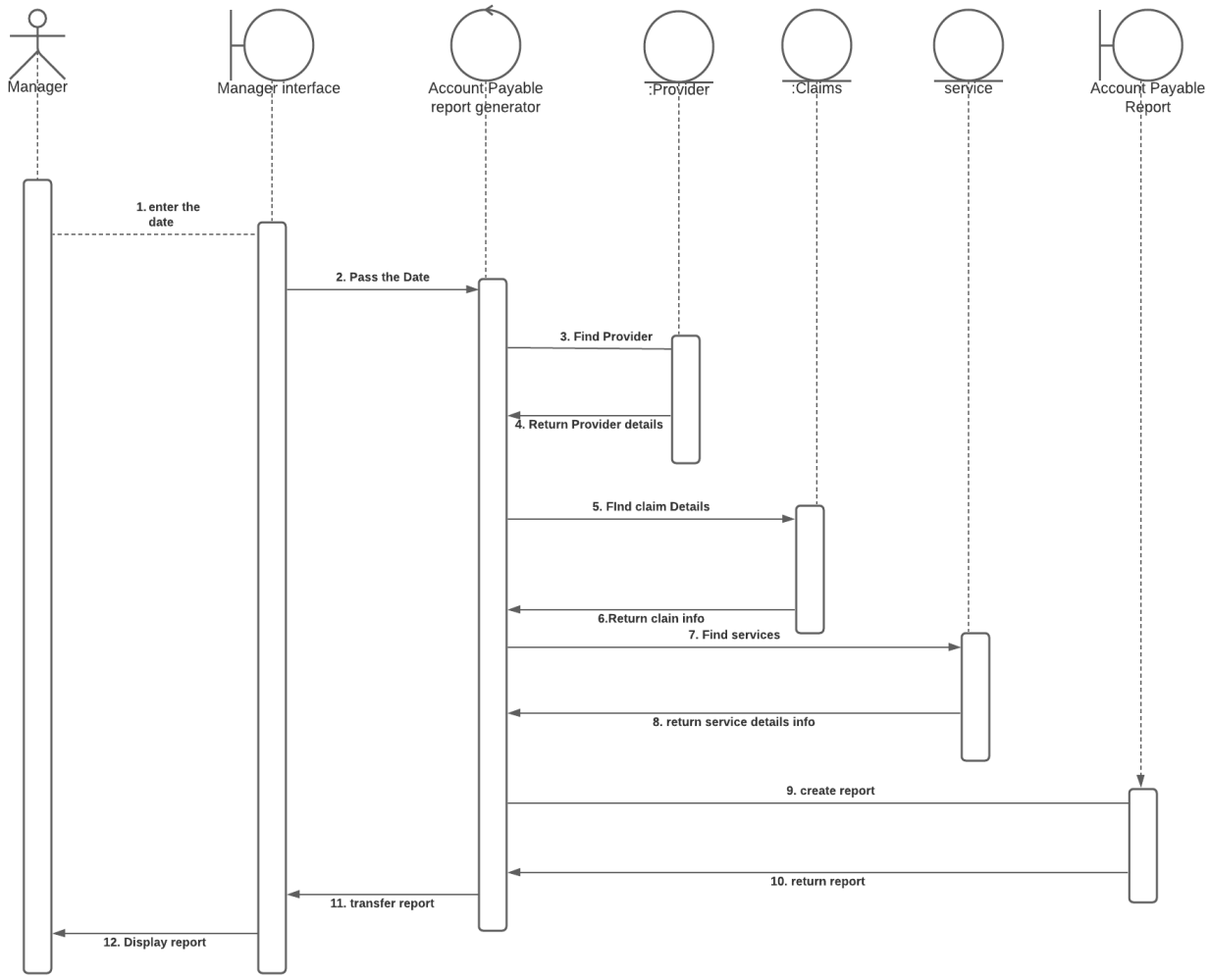
4.3.6.2 Member Report Generation



4.3.6.3 Provider Report Generation



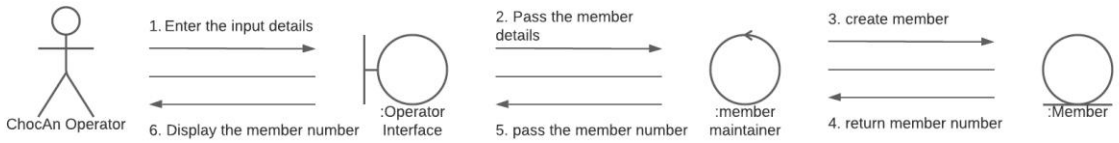
4.3.6.4 Account Payable Report Generation



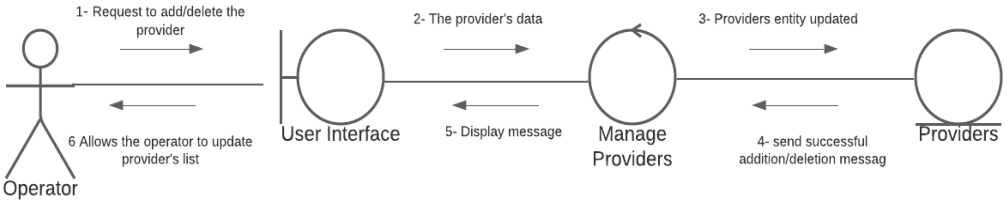
4.4 Communication Diagrams

Used to model the message passing structure of the system functions.

4.4.1 Member Maintenance

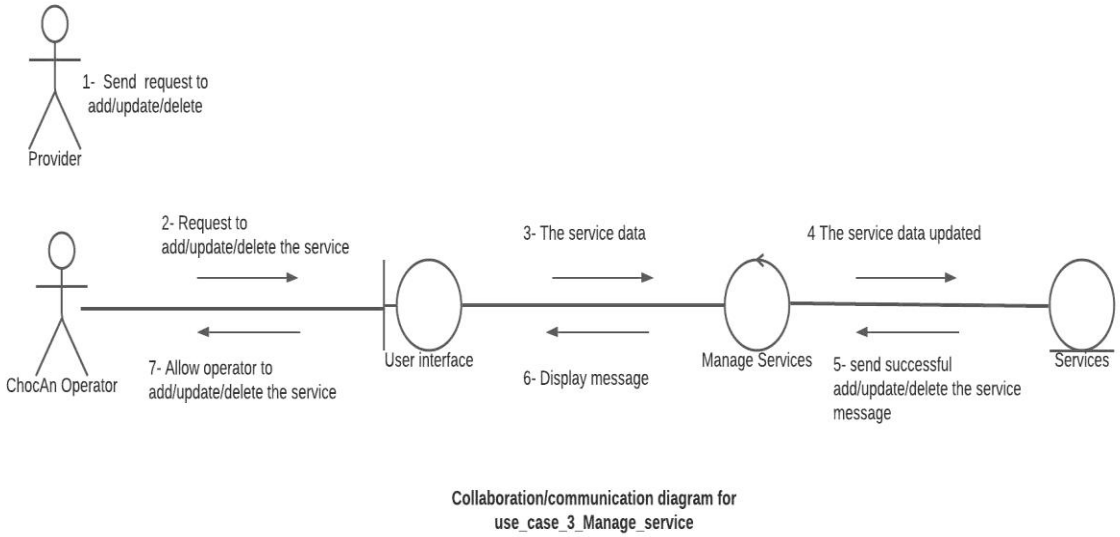


4.4.2 Manage Providers



Collaboration/communication
Diagram
Use_case_2_Manage_Providers

4.4.3 Manage Services



Collaboration/communication diagram for use_case_3_Manage_service

4.4.4 Bill Services

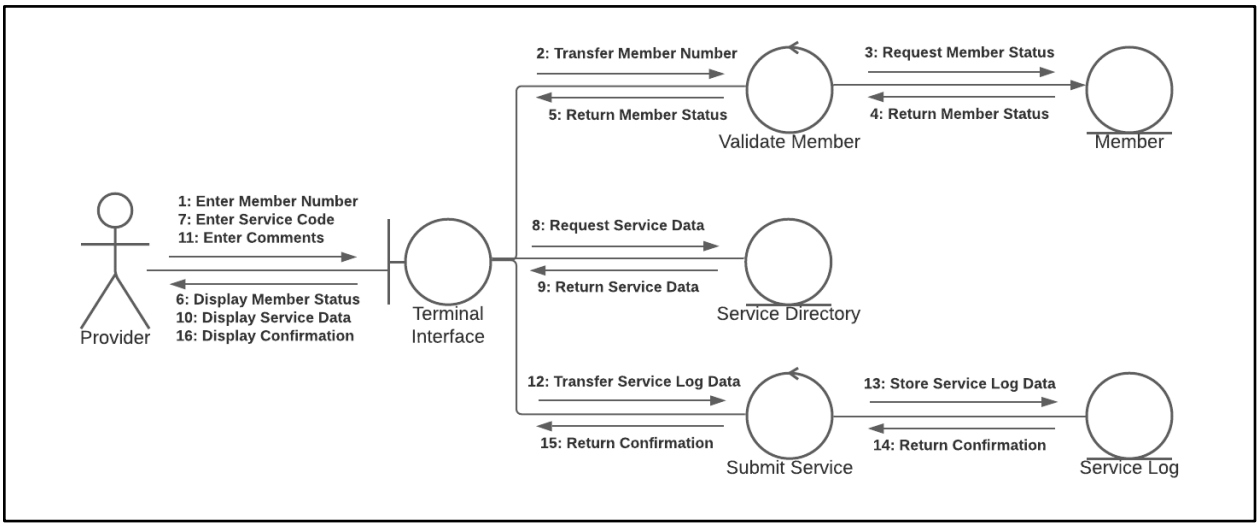
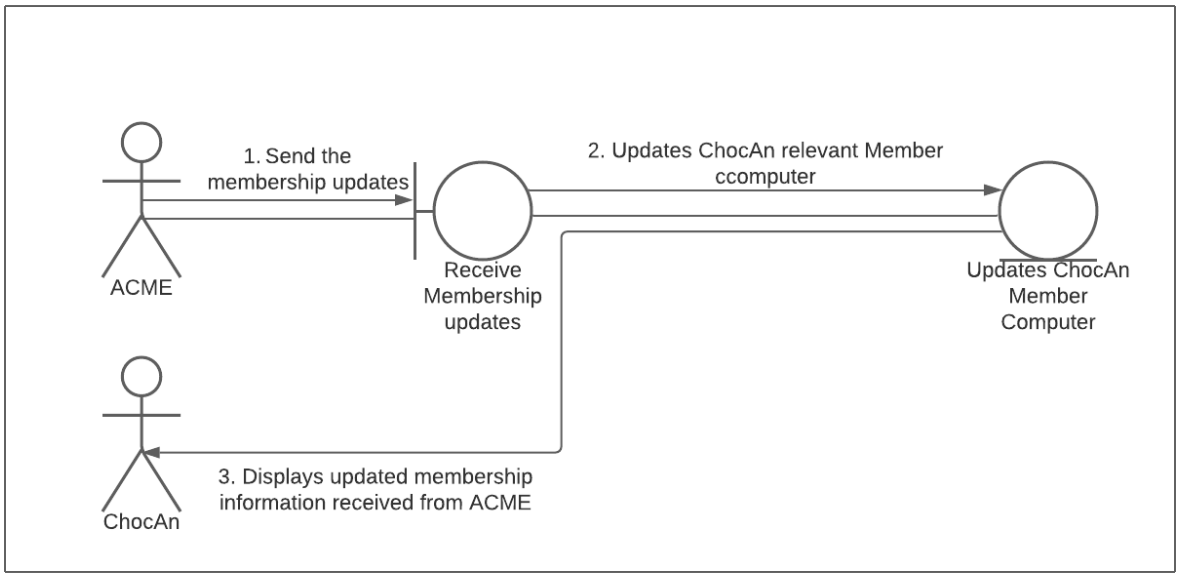


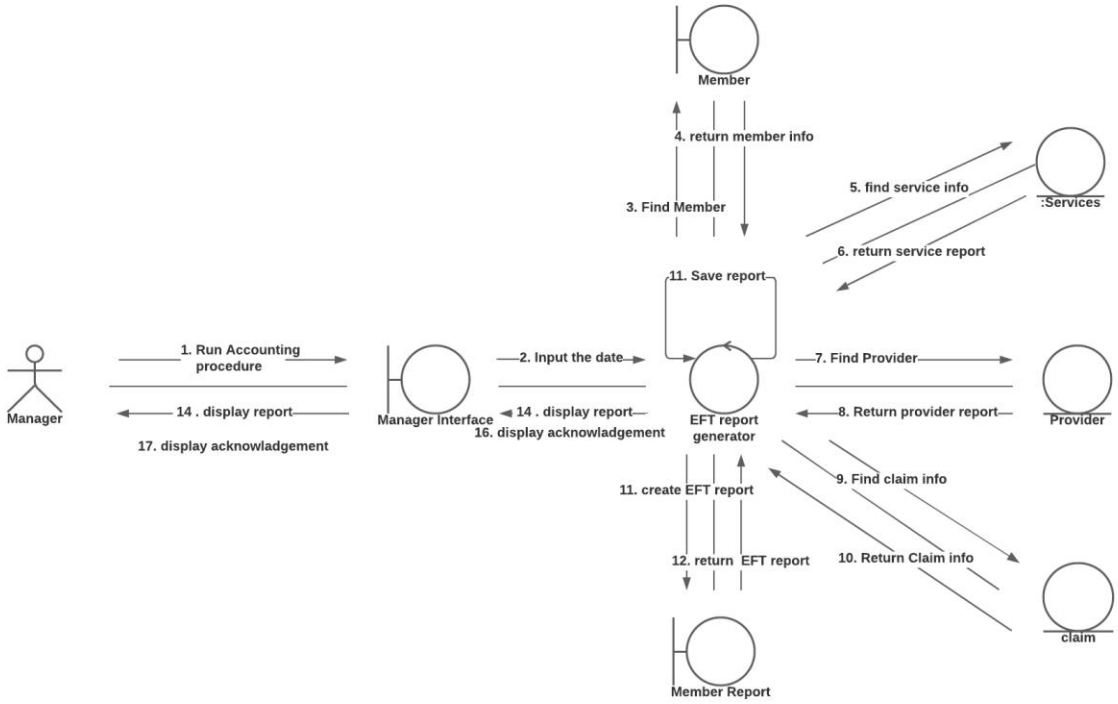
Figure X: Bill Services Communication Diagram

10.3.1

4.4.5 Receive Membership updates



4.4.6 Run Account Procedure



5 Behavioral Model and Description

A description of the behavior of the software is presented.

5.1 Description for software behavior

5.1.1 Events

The ChocAn system has four major events as follows.

- **Provider Section**

  System provides options to handle the provider requests.

- **Maintenance Subsystem Selection**

  System provides Member maintenance, Provider maintenance and service maintenance. Usually the operator will be responsible for handling these requests.

- **Manager Report generation**

  System provides provision to handle the manager requests like generating reports in interactive mode.

- **Running Account procedure**

  System has a scheduler internally which helps to run the account procedure which internally generates reports and sends it to corresponding.

5.1.2 States

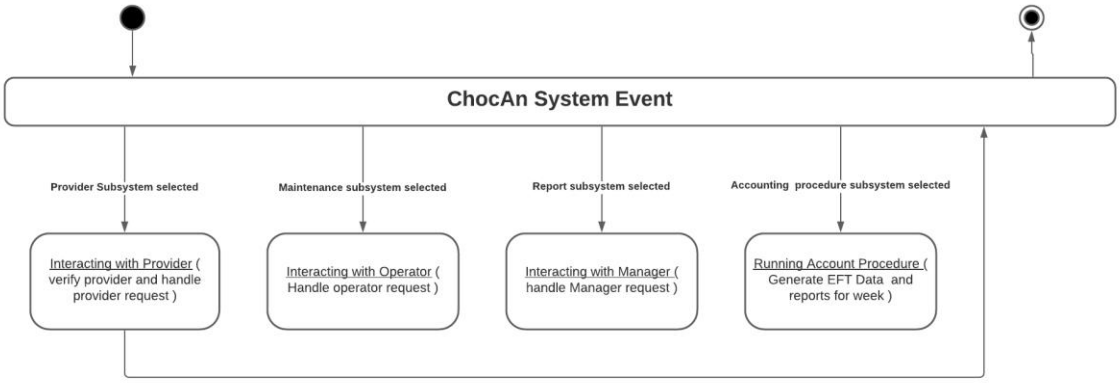
- **Provider Section**
  - Provider can verify the membership for any member
  - Provider can submit the claims

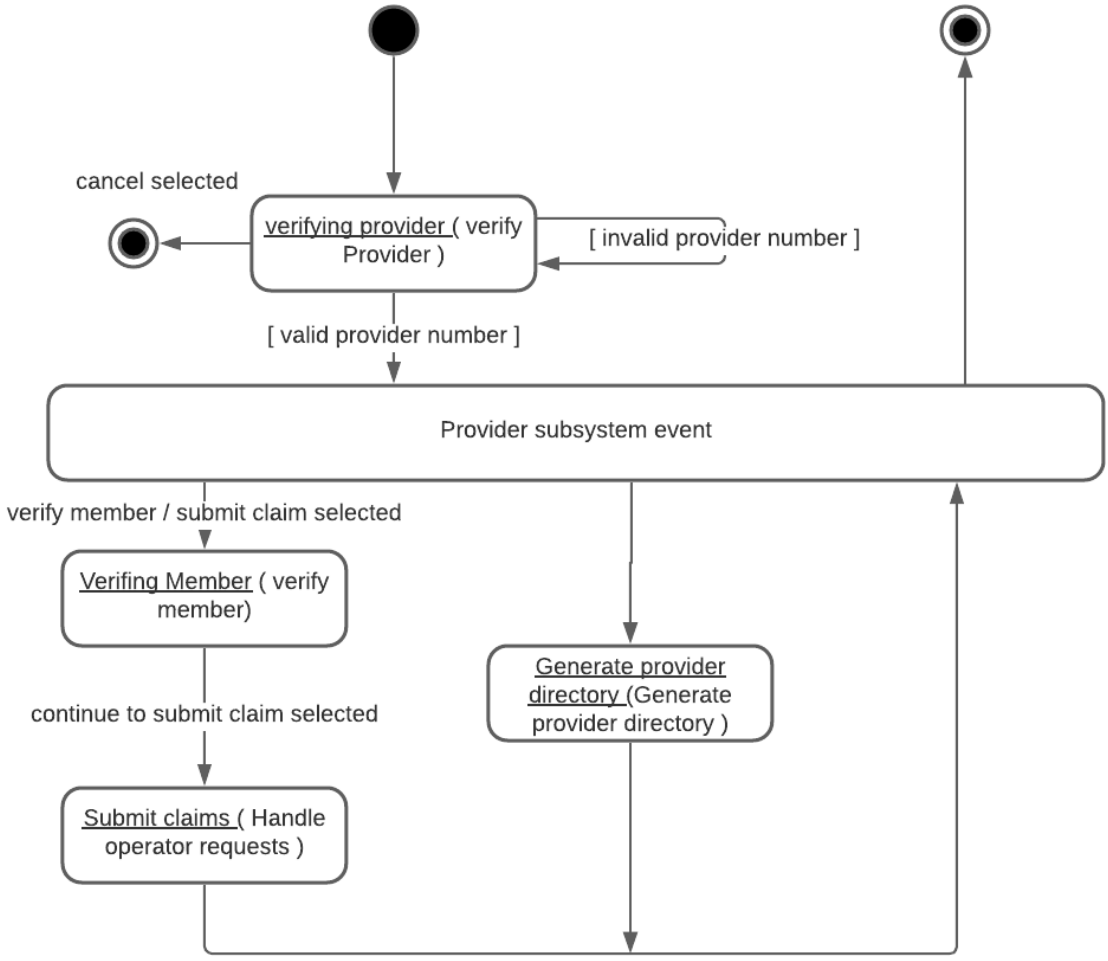
- Providers request the provider directory

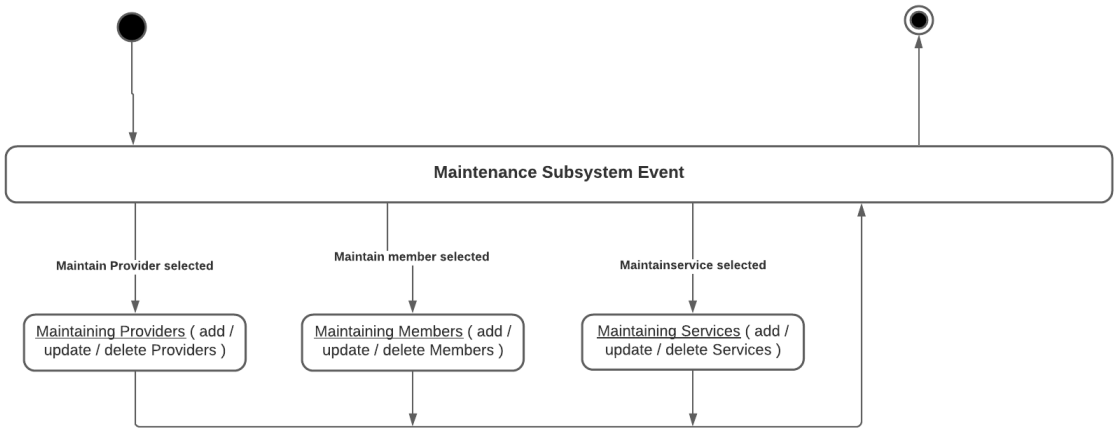
- **Maintenance Subsystem Selection**
  - Member maintenance
    - Add new member
    - Update existing member
    - Delete existing member
  - Provider maintenance
    - Add new provider
    - Update existing provider
    - Delete existing provider
  - Service maintenance.
    - Add new service
    - Update existing service
    - Delete existing service
- **Manager Report generation**
  - Generate the member report
  - Generate provider report
  - Generate the account payable report.
- **Running Account procedure**
  - Generate the member report and send to members
  - Generate provider report and send to providers
  - Generate the account payable report and send to manager
  - Generate EFT Data and send to EFT System (ACME system)

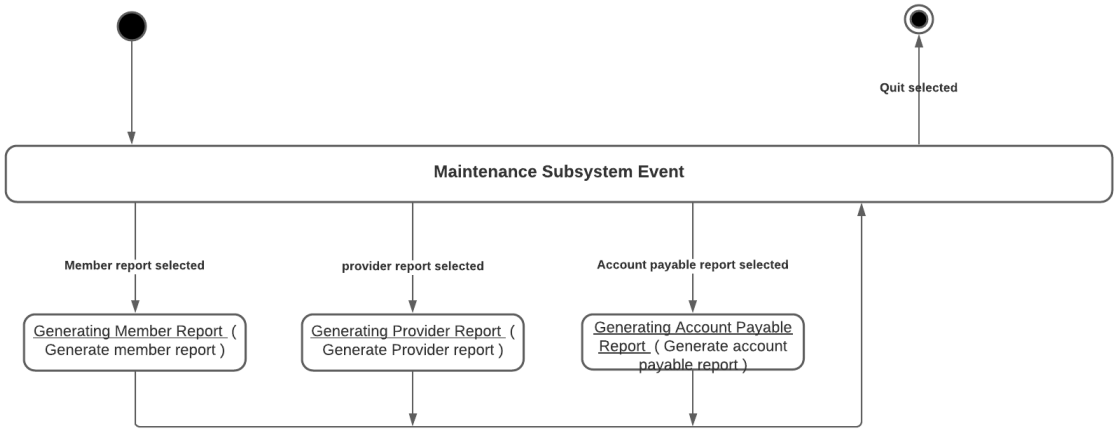
5.2 State Transition Diagrams

Depict the manner in which the software reacts to external events.


ChocAn System Event
Provider Subsystem selected
Maintenance subsystem selected
Report subsystem selected
Accounting procedure subsystem selected
Interacting with Provider (verify provider and handle provider request)
Interacting with Operator (Handle operator request)
Interacting with Manager (handle Manager request)
Running Account Procedure (Generate EFT Data and reports for week)


cancel selected
verifying provider (verify Provider)
[invalid provider number]
[valid provider number]
Provider subsystem event
verify member / submit claim selected
Verifing Member (verify member)
continue to submit claim selected
Submit claims (Handle operator requests)
Generate provider directory (Generate provider directory)


Maintenance Subsystem Event
Maintain Provider selected
Maintain member selected
Maintainservice selected
Maintaining Providers (add / update / delete Providers)
Maintaining Members (add / update / delete Members)
Maintaining Services (add / update / delete Services)


Quit selected
Maintenance Subsystem Event
Member report selected
provider report selected
Account payable report selected
Generating Member Report (Generate member report)
Generating Provider Report (Generate Provider report)
Generating Account Payable Report (Generate account payable report)

6 Restrictions, Limitations, and Constraints

The system must provide data protection

Since it process medical and billing information at the same time, the system must provide protection against data loss and ensure that detailed information is not lost. The system should make sure that the system saves the data or notify the user that an error has occurred. More information on this is detailed in the functional requirements section.

System Performance

The following are the general performance requirements for the client and server.

- **Shortest system response time**

  To allow the provider to execute the transaction quickly, the system will provide a response to your terminal as soon as possible. Users should always receive an immediate response when using the system, even if the response indicates an error or delay.

- **System uptime will exceed 99.99%**

  The server must have high availability and minimal downtime. Server components must be designed for redundancy and load balancing. The client component must be specifically designed to avoid failure and always be available.

- **System reboot time**

  If an outage occurs, the system will recover and reboot as soon as possible. There are no specific restart requirements for server components because they can be load balanced to avoid general downtime. It should take no more than 20 seconds for the client device to recover from a reboot event or power outage.

- **The system will be powered by renewable energy**

  To maintain the green corporate image, the system must be powered by the last renewable resource: baby's tears. This requirement may require the participation of the hardware engineering team, but its design requirements are beyond the scope of this document.

- **The system will be easy to use**

  The client and management tools should be functional but easy to use. Due to the massive nature of working with chocolate addicts, the focus should be on the ability to complete tasks effectively.

- **The client application should require minimal training.**

  Provider personnel should be able to use the system with minimal training, and should be designed to minimize the training time required for the operating system without errors.

The system must comply with all HIPPA regulations

When it comes to medical regulations, there are many mandatory government regulations on consumer data protection. The system must ensure that customer data is protected and that only authorized users can access it to avoid the possibility of leaking restricted information. The full definition of the HIPPA regulations is beyond the scope of this document.

- **Access to patient data**

  Access to patient data must be authorized and limited to authorized providers and administrators. Patient health data must be protected from external access. Only authorized users can access this data.

- **Automatic logout**

  If the session is idle for more than 240 seconds, the interactive interface will terminate the administrator or provider session. This feature is included in the HIPAA security rules to protect access to medical information.

- **Activity log**

  There should be a record of all supplier and manager interactions with the system. These records must be available at the time of the audit, and therefore must be available to managers who wish to audit records.

The system has several key limitations that affect the architecture design:

1. Client-server model: The system must be implemented as a client-server architecture, and our development team must maintain the server.

2. Response time-Client devices will be used in remote locations that may have different Internet connection speeds. The hardware terminal will have a dial-up connection as a backup option, and all data transmission must take into account the possibility of these slow connections.

3. The system must store data in an SQL database: Due to data retention and reporting requirements, the project data storage must be in a structured format that is easy to refer to.

7 Validation Criteria

7.1 Classes of tests/Test Strategy

ChocAn System testing will include low-level system and unit testing to ensure program correctness and accuracy, and high-level integration testing to ensure user requirements and workflows are met. Each type of testing will have it's own challenges, timelines, and requirements, which are documented below.

7.1.1 Unit testing

Low-level unit tests will be performed automatically using the testing framework and should be performed before checking any code in the source code control system. (As part of the integration and release cycle, the build, integration, and deployment teams will also perform unit tests.). All code in development will be compiled and tested at the same time. The build and integration manager will reject any code without proper testing coverage. The overall test coverage should be at least 85%, but should not include testing the database itself or any framework code, as these systems are covered by their own test suites. Both the client application software and the server software require unit testing. There is no requirement or

deadline for unit testing because it is an integral part of the development process. The challenge of unit testing is to ensure that developers do it as part of the process. This is resolved through a code coverage requirement of 85%, but it will also be monitored by project owner during the development process to ensure continued compliance. Failed unit tests are resolved as part of the development process, so there is no significant impact other than the developers fixing the issues in the code they submit.

7.1.2 System Testing

In addition to low-level unit tests, automated system tests will also be performed to test the functionality of the system. System testing reflects integration testing in its role and scope, and provides end-to-end testing of server systems. Because they are automated, they are considered a different type of test from integration testing. These tests will be performed on the server system to ensure correct system capabilities and behavior, and will focus on end-to-end performance, security, and other results for API calls. Please note that system tests will not be used for client applications, only for server components. The server integration and implementation team will test the system before each server is built and released in a test or production environment. Since this is built into the integration and deployment process, there are no special requirements or schedules related to system testing. The system test challenge revolves around solving any problems encountered during testing. Failed system tests generally indicate performance issues or other non-blocking errors and, in most cases, will not impede development or deployment progress. When a bug is found, you need to determine if the bug is serious enough to suspend the release or should only be resolved in the next release. This will lead to the risk of affecting the development plan, but for the major version that is carried out at the same time as the client application update, additional time should be built into the plan to fix the problems found in the integration test phase, which will solve the pending system test Questions provide opportunities as well.

7.1.3 Integration Testing

Integration testing will be performed manually by humans in accordance with specific step-by-step scripts. Due to the nature of the interaction between hardware and software, this level of testing is difficult to automate through computer running tests. It is best to test the "script" that the human tester will follow to ensure that the test is repeatable and there are no errors caused by the process. These tests will be performed using the production version of the application being tested to ensure that no errors will be introduced during the software build process. Integration Tests will be divided into several levels:

• Level 1: These tests are the features and workflow required for the system to be considered as containing the minimum acceptable functionality.

• Level 2: These tests describe features and workflows that do not need to meet the minimum functional requirements but still need to be tested.

Integration testing is the only form of testing that requires additional resources or time away from the developers, so it will bring some additional challenges and requirements. The tester setup requirements are initially low, but if the test coverage increases, additional resources may be required to complete the test within a reasonable amount of time. The QA will perform integration testing during the development cycle and prior to each release of any client applications. Development milestones should include the time to complete this testing phase

7.2 Expected software response

When these steps are followed and the expected results are obtained, the test passes. These tests are designed to be run before each production version of the application to ensure that the entire system works as expected.

7.2.1 Level 1 Test

The Level 1 test describes the minimum required features and must be passed with a 100% completion rate for the entire test plan to be successful. The Administration Client

Level 1 test should include tests for all major workflows, some of which are listed below:

- Chocan administrators can use the administration interface to add, edit, and delete members. This test ensures that admin interface users can successfully add using the admin tool, edit and delete members. The test should include steps to ensure that locally cached data does not lead to erroneous results.
- As Chocan administrator, the report generation must send the report to designated email This test should ensure that when a request is made using the administration client application, the correct report is sent to the requested email address. In this test, no effort should be made to ensure the accuracy of the entire report set, but a brief check should be performed to ensure that the report contains data and no "blanks" are provided.
- Provider can log into my terminal with the correct ID. This test ensures that the provider correctly logs into the system when they enter their provider ID number, and entering the wrong provider ID will prevent access.
- Provider can retrieve reports of all available services. This test ensures that vendors can request service reports and that the reports are sent to the email associated with your vendor information.
- Provider can check the status of the member's account by entering the member number. This report should test if the correct status of the provider is returned for the user account. Multiple accounts of different types must be tested to ensure that all expected statuses are working properly.
- Provider can keep a record of a service provided to a member This test should verify the entire workflow to keep a record of user access to the service provider. Test

7.2.2 Level 2 Test

Level 2 describes the required functionality and workflow, but only needs to pass with an overall completion rate of 75% to meet the publishing criteria. Level 2 tests should solve more abstract problems or problems designed to secure non-technical goals. Here are some examples of Level 2 tests for illustration:

- **A login failure should provide an easy to understand error**

  When a login attempt fails, the user should understand what happened and why without resorting to checking the error code book.

- **Server connection errors should provide easy to understand errors**

  If the server does not respond, the client application should notify the user in non-technical language, explaining what they are going to do next.

- **The sent report contains a correct and useful text message that explains what the report is and why the recipient received it.**

  Users receiving reports from the system should not simply receive the report as an attachment. The email they receive should explain what the report is and who they can contact if they receive an error. It should also include any other relevant text.

7.3 Performance bounds

The test should be designed to run at least one script of each type of request to test the response time of each request and the number of simultaneous requests that the server can handle. The objective is to simulate a large number of clients accessing the system simultaneously. Due to the nature of this request, it will need to run in parallel. Ideally, this test will be built as a multi-step workflow through the API. This will ensure that the test meets the expectations step by step, because all the data in the workflow will be independent. These workflows should be run in an offset manner to prevent all tests from attempting to write at the same time and then read at the same time. To be considered successful, this test must ensure that the API can handle an average of 200 requests per second, and that no single request takes more than 2 seconds to complete. In addition to simple test performance, this test should provide a high-level verification that each API route works as expected and returns relevant results. This will be done by checking the HTTP return status to make sure it is correct and checking the payload of each output to make sure it meets the expected output of the workflow.

8 Appendices

8.1 System traceability matrix

A matrix that traces stated software requirements back to the system specification. File is enclosed along with this file here

8.2 Product Strategies

The methodologyof the project will be a mixture of waterfall development and incremental development. According to the waterfall model, we start the project in phases. These stages include requirements definition, systems and software design, unit testing and implementation, system integration and testing, and operation and maintenance. None of the stages are concurrent because they only start when the previous stage is completed, from one stage to another. In the traditional waterfall model, there is no backtracking between processes. However, we expect the deployment and unit testing phases to have some impact on the functionality of our system, leading to changes in design and requirements. Therefore, at that stage, we can carry out specification, development and verification activities at the same time. Therefore, a hybrid cascade and an incremental development model is performed. At the beginning of this project, the client showed us a comprehensive plan of their needs and goals. Driven by this plan, and keeping the waterfall approach in mind, we started the development process by defining requirements, which resulted in requirements documents. After completing the previous stage, we resolved the design of the system and software, which is shown in this design document. This document specifies the design and design strategy of the project, and is based on the requirements defined in the previous phase. Finally, equipped with the peculiarities established in the design phase, we will enter the system integration and testing phase. At this stage, we will develop a test plan, while addressing the feasibility of implementing our design and adjusting our testing and design accordingly.

8.3 Analysis metrics to be used

The design documents will be completed on 07/01/2021. Designed to freeze the document design, once this milestone is reached, development can begin. The development is completed on July 10, 2021. Once this milestone is achieved, a fully functional product should be ready for testing. At the end of this milestone, code changes should be limited to bug fixes or incomplete functionality found in the UAT phase. UAT will end on July 20, 2021. It refers to a product that has been accepted by related parties and is considered to be fully functional and as error-free as possible. Once the UAT is completed, the product will be deployed, or if it is determined that it has too many defects to be released, then the second phase must restart from the design phase. Due to hardware design requirements, the keyboard and mouse of the simulated device will be used for testing at this stage, and the same keyboard and mouse will be used for the simulated management terminal. Until the hardware integration is completed, the system implementation will remain pending, which is beyond the scope of this document. The software has been implemented and will be available to end users on August 1, 2021. System maintenance starts at this time. This is a project beyond the scope of this document and will not be discussed further.Supplementary information (as required)