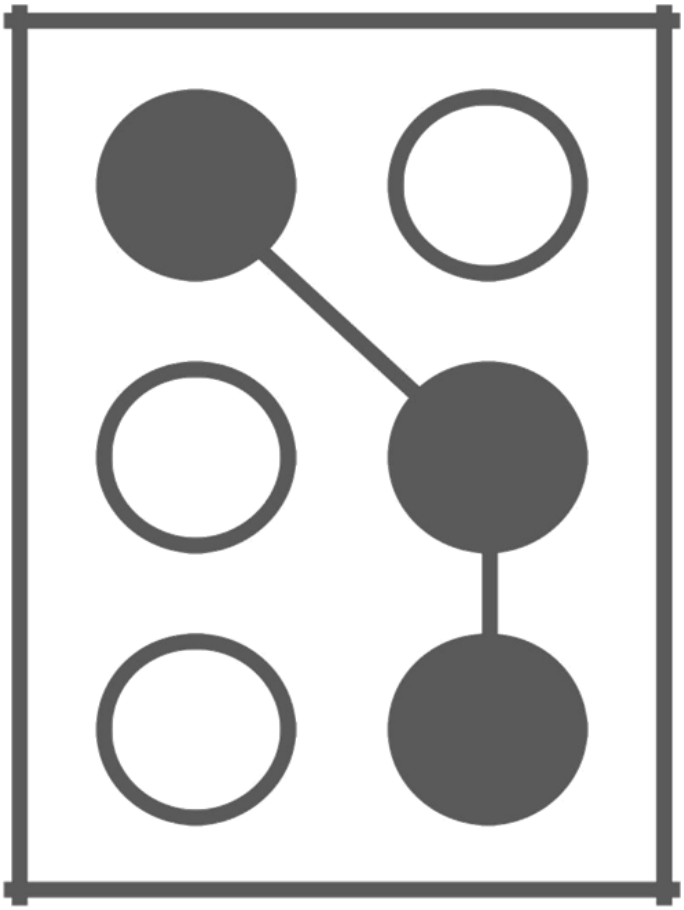
# Bengali Braille to Text Translator

Software Requirements Specification and Analysis



25-Sep-2016

IIT
University of Dhaka

# Bengali Braille to Text Translator

Software Requirements Specification and Analysis


Submitted by

Minhas Kamal

Roll: BSSE-0509

Student of BSSE 8th Semester

Institute of Information Technology

University of Dhaka

Supervised by

Dr. Mohammad Shoyaib

Associate Professor

Institute of Information Technology

University of Dhaka



Submission Date

25th September, 2016

# LETTER OF TRANSMITTAL

25th September, 2016.

BSSE 4th Year Exam Committee
Institute of Information Technology
University of Dhaka

Sir,

I have prepared the report on Software Requirements Specification of Bengali Braille to Text Translator.

The primary purpose of this report is to summaries my findings that I have gathered during the requirements specification process. This report also includes details of each step I have followed while collecting the requirements.

Sincerely Yours,

Minhas Kamal
Roll: BSSE-0509
Student of BSSE 8th Semester
Institute of Information Technology
University of Dhaka

Enclosure: Software Requirements Specification Report

# Executive Summary

Braille is a specialized writing system for visually impaired people, where raised dots on embossed paper are used as tactile alphabet. The tool Bengali Braille to Text Translator will take in scanned image of Bengali Braille writing, apply pattern recognition, and translate it to text. The user does not need much theoretical or technical skill to run this software.

# Acknowledgements

By the grace of Almighty Allah I have completed my report on Software Requirements Specification of Bengali Braille to Text Translator.

I am grateful to my supervisor Dr. Mohammad Shoyaib for his direction throughout the working time. It was almost impossible for me to complete this SRS document without him.

I am also thankful to the teachers and students of The Institute of Education and Research, University of Dhaka. They greatly helped me in collecting information among all business.

# Table of Contents


**Chapter 1: Introduction ..... 1**

1.1 Purpose 1
1.2 Intended Audience 1

**Chapter 2: Inception ..... 3**

2.1 Introduction 3
2.1.1 Identifying Stakeholders 3
2.1.2 Asking the First Questions 4
2.1.3 Recognizing Multiple Viewpoints 4
2.1.4 Working towards Collaboration 5
2.2 Conclusion 6

**Chapter 3: Elicitation ..... 7**

3.1 Introduction 7
3.2 Eliciting Requirements 7
3.3 Collaborative Requirements Gathering 7
3.4 Quality Function Deployment 8
3.4.1 Normal Requirements 8
3.4.2 Expected Requirements 8
3.4.3 Exciting requirements 9
3.5 Usage Scenario 9
3.6 Elicitation Work Product 10

**Chapter 4: Scenario-Based Model ..... 11**

4.1 Introduction 11
4.2 Use Case Scenario 11
4.3 Use Case Descriptions 13
4.3.1 Bengali Braille to Text Translator 14
4.3.1.2 Pre-Processing 18
4.3.1.3 Translation 31
4.3.1.4 Post-Processing 41

# Table of Contents


**Chapter 5: Data Model** ..... **48**

5.1 Introduction 48
5.2 Data Object Selection 48
5.3 Data Objects & Attributes 51
5.4 Conclusion 52

**Chapter 6: Class-Based Model** ..... **53**

6.1 Introduction 53
6.2 General Classification 53
6.3 Selection Characteristics 55
6.4 Attribute Selection 56
6.5 Defining Methods 57
6.5.1 Verb List 57
6.5.2 Selected Methods 58
6.6 Class Diagram 60
6.7 Class Card 61

**Chapter 7: Behavioral Model** ..... **62**

7.1 Introduction 62
7.2 Identifying Events 62
7.3 State Transition Diagram 64
7.4 Sequence Diagram 65

**Chapter 8: Conclusion** ..... **76**

**Appendix** ..... **77**

# List of Figures


| Figure No. | Figure Name | Page No. |
|---|---|---|
| **Figure 4.3** | Use Case Diagram of BBTT (Level-0) | 13 |
| **Figure 4.3.1** | Use Case Diagram of BBTT (Level-1) | 14 |
| **Figure 4.3.1.1a** | Activity Diagram- Image Acquisition | 16 |
| **Figure 4.3.1.1b** | Swim-Lane Diagram- Image Acquisition | 17 |
| **Figure 4.3.1.2** | Use Case Diagram of Pre-Processing (Level-1.2) | 18 |
| **Figure 4.3.1.2.1a** | Activity Diagram- Enhancement | 20 |
| **Figure 4.3.1.2.1b** | Swim-Lane Diagram- Enhancement | 21 |
| **Figure 4.3.1.2.2a** | Activity Diagram- Noise Reduction | 23 |
| **Figure 4.3.1.2.2b** | Swim-Lane Diagram- Noise Reduction | 24 |
| **Figure 4.3.1.2.3a** | Activity Diagram- Connectivity Improvement | 26 |
| **Figure 4.3.1.2.3b** | Swim-Lane Diagram- Connectivity Improvement | 27 |
| **Figure 4.3.1.2.4a** | Activity Diagram- Quantization | 29 |
| **Figure 4.3.1.2.4b** | Swim-Lane Diagram- Quantization | 30 |
| **Figure 4.3.1.3** | Use Case Diagram of Translation (Level-1.3) | 31 |
| **Figure 4.3.1.3.1a** | Activity Diagram- Segmentation | 33 |
| **Figure 4.3.1.3.1b** | Swim-Lane Diagram- Segmentation | 34 |
| **Figure 4.3.1.3.2a** | Activity Diagram- Pattern Recognition | 36 |
| **Figure 4.3.1.3.2b** | Swim-Lane Diagram- Pattern Recognition | 37 |
| **Figure 4.3.1.3.3a** | Activity Diagram- Raw Text Generation | 39 |
| **Figure 4.3.1.3.3b** | Swim-Lane Diagram- Raw Text Generation | 40 |
| **Figure 4.3.1.4** | Use Case Diagram of Post-Processing (Level-1.4) | 41 |
| **Figure 4.3.1.4.1a** | Activity Diagram- Spell Check | 43 |
| **Figure 4.3.1.4.1b** | Swim-Lane Diagram- Spell Check | 44 |
| **Figure 4.3.1.4.2a** | Activity Diagram- Clean and Generate Output | 46 |
| **Figure 4.3.1.4.2b** | Swim-Lane Diagram- Clean and Generate Output | 47 |
| **Figure 6.6** | Class Diagram (BBTT) | 60 |
| **Figure 7.3** | State Transition Diagram- User | 64 |
| **Figure 7.4.1** | Sequence Diagram- Take Image | 65 |
| **Figure 7.4.2** | Sequence Diagram- Run Enhancement | 66 |
| **Figure 7.4.3** | Sequence Diagram- Reduce Noise | 67 |
| **Figure 7.4.4** | Sequence Diagram- Improve Connectivity | 68 |
| **Figure 7.4.5** | Sequence Diagram- Apply Quantization | 69 |
| **Figure 7.4.6** | Sequence Diagram- Extract Braille Pattern | 70 |

# List of Figures


| Figure No. | Figure Name | Page No. |
|---|---|---|
| **Figure 7.4.7** | Sequence Diagram- Convert Braille to Code | 71 |
| **Figure 7.4.8** | Sequence Diagram- Map Code to Character | 72 |
| **Figure 7.4.9** | Sequence Diagram- Check Spell | 73 |
| **Figure 7.4.10** | Sequence Diagram- Run Cleaning Procedure | 74 |
| **Figure 7.4.11** | Sequence Diagram- Customize Template | 75 |

# Bengali Braille to Text Translator

## Software Requirement Specification and Analysis

# Chapter 1
# Introduction

## 1.1 Purpose

This is the Software Requirements Specification (SRS) for the tool- Bengali Braille Character Recognizer. The document contains detailed functional, non-functional, and support requirements for the project. It also establishes a requirements baseline for the development of the system.

The SRS serves as the official means of communicating user requirements to the developer [1]. It provides a common reference point for both the developer team and stakeholder community. This document's content may evolve over time as users and developers work together in developing the system. But its fundamental output will remain unchanged.

## 1.2 Intended Audience

This SRS is intended for all the stakeholders- customers, project managers, designers, developers, and quality assurance engineers [1] [2]. They will use it for the following reasons-

- Customers will use this SRS to verify acceptability of the development team's work.
- The project managers will design their plan on the basis of this document. They will set milestones and delivery dates. They will also ensure that the development team is on track.
- The designers will prepare the systems design based on this document. They will continually refer back to this SRS to ensure that the system they are designing will fulfill the customers' needs.

- Developers will use this document as a basis for developing the system's functionality. The developers will link the requirements defined in this SRS to the software they create to ensure that they have created software that will fulfill all of the customers' documented requirements.
- Quality assurance engineers will use the SRS to derive test plans and test cases. When portions of the software are complete, they will run their tests on that software to ensure that the software fulfills the requirements documented in this SRS. They will again run their tests on the entire system when it is complete and ensure that all requirements documented in this SRS have been completed.

# Chapter 2
# Inception

## 2.1 Introduction

Requirements Engineering starts with Inception phase. Its goal is to identify parallel needs and conflicting requirements among the stakeholders of a project. The foundation was established by following the subsequent factors-

### 2.1.1 Identifying Stakeholders

Any person, group, or organization which will affect or be affected by the system directly or indirectly is a stakeholder. It includes both project developers and end-users. Here only client side stakeholders will be focused.

Although, we intend to develop Bengali Braille to Text Translator for public use, we are currently building it for using only in the Institute of Education and Research (IER), University of Dhaka premises. For this reason, we have selected the stakeholders from the scope of IER only. We have identified following stakeholders for our project:

1. **Teachers of IER:** Teachers working with visually impaired student at IER are our primary client side stakeholders. They will directly interact with the system.

2. **Visually Impaired Students of IER:** Although the visually impaired students will not interact with the system directly, but they are the biggest group affected by the system. The software will process their writing and convert it to text.

3. **General Students of IER:** Visually normal students will also use the software. But they will not be using it for the same reason as the teachers.

### 2.1.2 Asking the First Questions

First set of context-free questions focuses on the client side stakeholders and tries to reveal overall project goals and benefits. Now we ask question to help us gaining a better understanding of problem, and to allow customers to voice their perceptions about the solution.

### 2.1.3 Recognizing Multiple Viewpoints

We have collected these view points by discussing with the teachers, visually impaired students, and general students of IER.

1. **Teachers of IER:**
    a. Full control of the system
    b. Low cost
    c. Minimum error rate
    d. Handing batch of images
    e. Intuitive user interface
    f. Ability to see intermediate results

2. **Visually Impaired Students of IER:**
    a. Error free solution
    b. Process double sided writing
    c. Special character recognition ability
    d. Fast processing

3. **General Students of IER:**
    a. Easy to use
    b. User friendly interface

c. Easy to install

d. Work with any scanner

e. No need of theoretical or technical knowledge

### 2.1.4 Working towards Collaboration

Every stakeholder has his own set of requirements from his point of view. We followed following steps to merge these requirements:

i. Identify the common and conflicting requirements

ii. Categorize the requirements

iii. Take priority points for each requirement from stakeholders and on the basis of this voting prioritize the requirements

iv. Make final decision about the requirements

**Common Requirements:**

User friendly

interface Easy to use

**Conflicting Requirements:**

Low cost and process double sided writing

Fast and minimum error rate

Full control of the system and no need of theoretical or technical knowledge

**Final Requirements:** We have finalized following requirements for the system through categorization and prioritization process:

Easy to use

User friendly interface

Users with theoretical and technical knowledge will be able to fully customize and control the system

Users with no technical knowledge will be able to use templates (created by specialists) for running the system

Minimize error rate

Can work with any scanner

## 2.2 Conclusion

Inception phase strongly supported us in establishing basic understanding about Bengali Braille to Text Translator. It also helped us to identify people who will be benefited by the software. Most importantly it established a preliminary communication with the stakeholders.

# Chapter 3
# Elicitation

## 3.1 Introduction

Elicitation helps the customer to define the requirement more specifically. This phase faces many problems like- problems of scope, problems of volatility, and problems of understanding. To overcome these problems, we have worked in an organized and systematic manner.

## 3.2 Eliciting Requirements

Unlike Inception, where Question and Answer approach is used, Elicitation makes use of a requirements elicitation format that combines the elements of problem solving, elaboration, negotiation, and specification. It requires the cooperation of a group of end-users and developers to elicit requirements.

## 3.3 Collaborative Requirements Gathering

There are many different approaches to collaborative requirements gathering. Each approach makes use of a slightly different scenario. We followed the subsequent steps to do it:

i. Meetings were conducted with teachers and students of IER, DU. They were questioned about their requirements and expectations from the tool.

ii. They were asked about the problems they are facing with exam papers written in Braille. We also inquired regarding the efficiency of the current process.

iii. At last we selected our final requirement list from these meetings.

## 3.4 Quality Function Deployment

The technique which translates the needs of the customer into technical requirements for software is called Quality Function Deployment (QFD).

QFD concentrates on maximizing customer satisfaction from the Software Engineering process. With respect to our project the following requirements are identified by QFD-

### 3.4.1 Normal Requirements

Objectives and goals that are stated during the meeting with the customers are Normal Requirements. Our project's Normal Requirements are:

1. User can customize the work sequence, and store the configuration for future use.
2. Will be able to run the tool and get output with only single 'run' button.
3. User can view result from each level of Braille to text translation process.
4. User will be able to run different types of image enhancement and other pre-processing algorithms to improve input image quality.
5. There will be a spell checking system, which will minimize error rate.
6. There will be some default configuration templates available with the software.
7. User will also be able to select different pattern recognition algorithms for recognizing Braille patterns on the basis of his need.

### 3.4.2 Expected Requirements

Expected Requirements are implicit to the system. These requirements may be so fundamental that the customers do not explicitly state them. Expected Requirements of our project are:

1. The system will be able to support all popular image formats like- JPG, JPEG, PNG, and so on.

2. User will be able to process both single image and batch of images of Braille writing.

3. The user interface of the system shall be easy to use. It will make use of drop-down boxes, radio buttons, and other selectable fields wherever possible instead of fields that require the user to type in data.

### 3.4.3 Exciting Requirements

Features that go beyond the customer's expectations are called Exciting Requirements. They prove to be very satisfying when present-

1. The user interface should provide appropriate error messages for invalid input as well as tool-tips and help.

2. The system will also be able to work with multiple languages of Braille writing.

### 3.5 Usage Scenario

## Bengali Braille to Text Translator

Bengali Braille Character Recognizer is a tool that will take scanned image of Braille writing, run different types of image-preprocessing techniques, translate Braille to text through pattern recognition, and apply text correction procedures for final output.

Braille is a specialized writing system for visually impaired people. Here raised dots on embossed paper are used as tactile alphabet [3] [4]. These papers will be scanned with a scanner, and resulting image files will be stored in the computer.

At first, the user will select scanned images of Braille writing. Now, different types of image enhancement processes will be run. These image enhancement processes will improve Braille patterns and make it distinguishable from the

background. After that, various noise reduction algorithms will be applied for reducing artifacts and improve pattern quality. For further development of Braille dots, connectivity improvement and quantization will be applied.

In the second section, Braille is translated to text. Firstly, Braille patterns are extracted from the image. Then these patterns are converted to code- Braille Pattern Code (BPC- each dot of Braille character is encoded in '1' and '0'). Now these resulting codes are mapped with Code to Character Map (CCM-like a hash table, where each unique code relates to only one character) and plain text file is generated. This text file contains probable raw characters for the Braille image.

In the post processing stage, spell checking will be performed for minimizing error rate. Here, words for spell checking process will be stored in a Domain Word List (DWL) file. Lastly, a cleaning procedure will produce the final text file.

All these steps, settings, algorithms and their parameters can be automated using a configuration file- Braille Processing Template (BPT). Some templates (BPT) will be available to the user in default with the system. Users will also be able to define customized template.

## 3.6 Elicitation Work Product

Our elicitation work product includes:

Statement of our requirements for Bengali Braille to Text Translator

A bounded statement of scope for our solution

Set of usage scenarios

A list of clients, users, and other stakeholders who participated in requirement specification process

# Chapter 4
## Scenario-Based Model

### 4.1 Introduction

The user's point of view is used to describe Scenario-Based Model. In SRS this is the first modeling phase. So, it serves as an input for the creation of other models.

### 4.2 Use Case Scenario

With the advancement of requirements gathering, functionalities and responsibilities of the software starts to materialize. The following table enlists primary components of the system:

**Table-4.2 Use Case Scenario**

| Level-0 | Level-1 | Level-2 | Actors |
|---|---|---|---|
| Bengali Braille to Text Translator | Pre-Processing | Image Acquisition | User, Scanner |
| | | Enhancement | User, Braille Processing Template |
| | | Noise Reduction | User, Braille Processing Template |
| | | Connectivity Improvement | User, Braille Processing Template |
| | | Quantization | User, Braille Processing Template |
| | Translation | Segmentation | User, Braille Processing Template |
| | | Pattern Recognition | User, Braille Processing Template, Braille |

| | | | |
|---|---|---|---|
| | | | Pattern Code |
| | | Raw Text Generation | User, Braille Processing Template, Code to Character Map |
| | Post-Processing | Spell Check | User, Braille Processing Template, Domain Word List |
| | | Clean and Generate Output | User, Braille Processing Template |

## 4.3 Use Case Description

In this section Use Case Scenario will be elaborated to Use Case Diagram, Description, Activity Diagram, and Swim-Lane Diagram. Figure-4.3 is the Use Case Diagram of level-0 for Bengali Braille to Text Translator (BBTT):

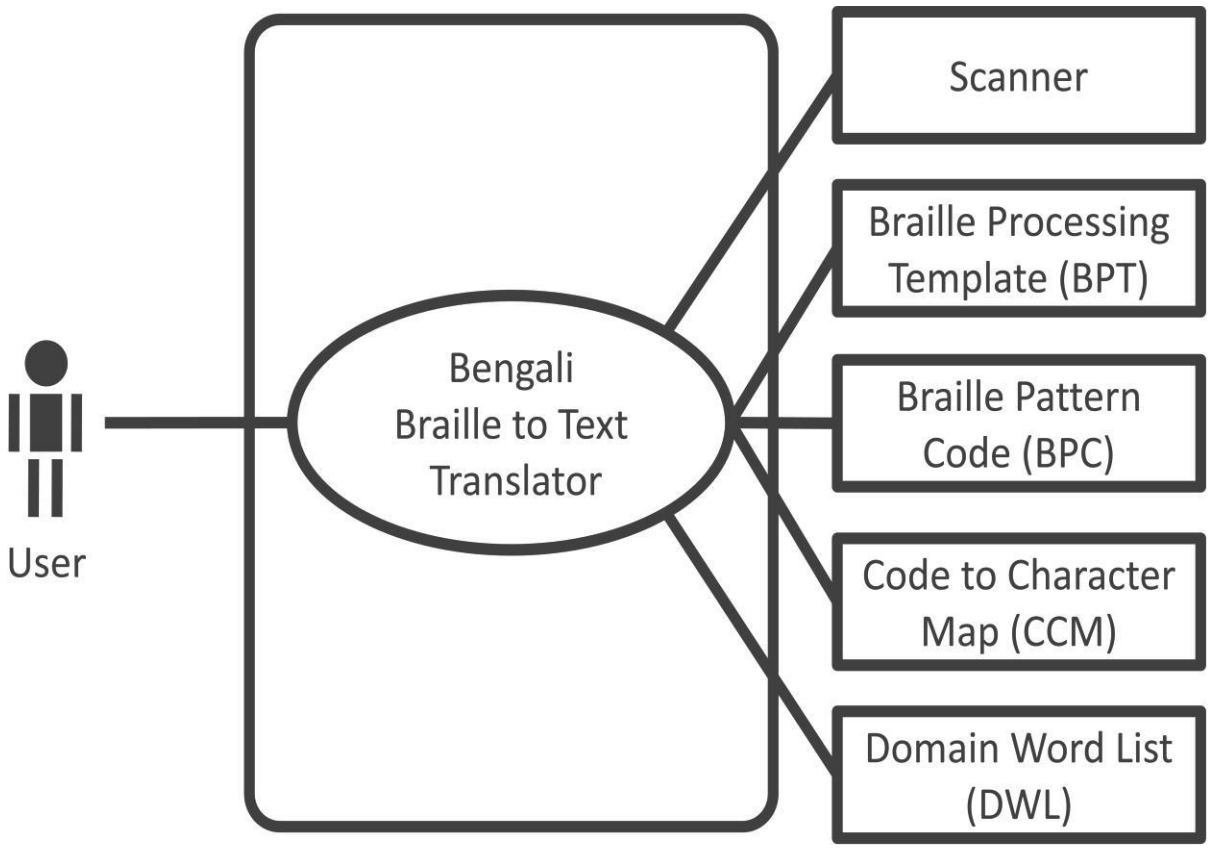



Figure 4.3: Use Case Diagram of BBTT (Level-0)

### 4.3.1 Bengali Braille to Text Translator

Here Figure-4.3.1 is the detailed form of level-0:

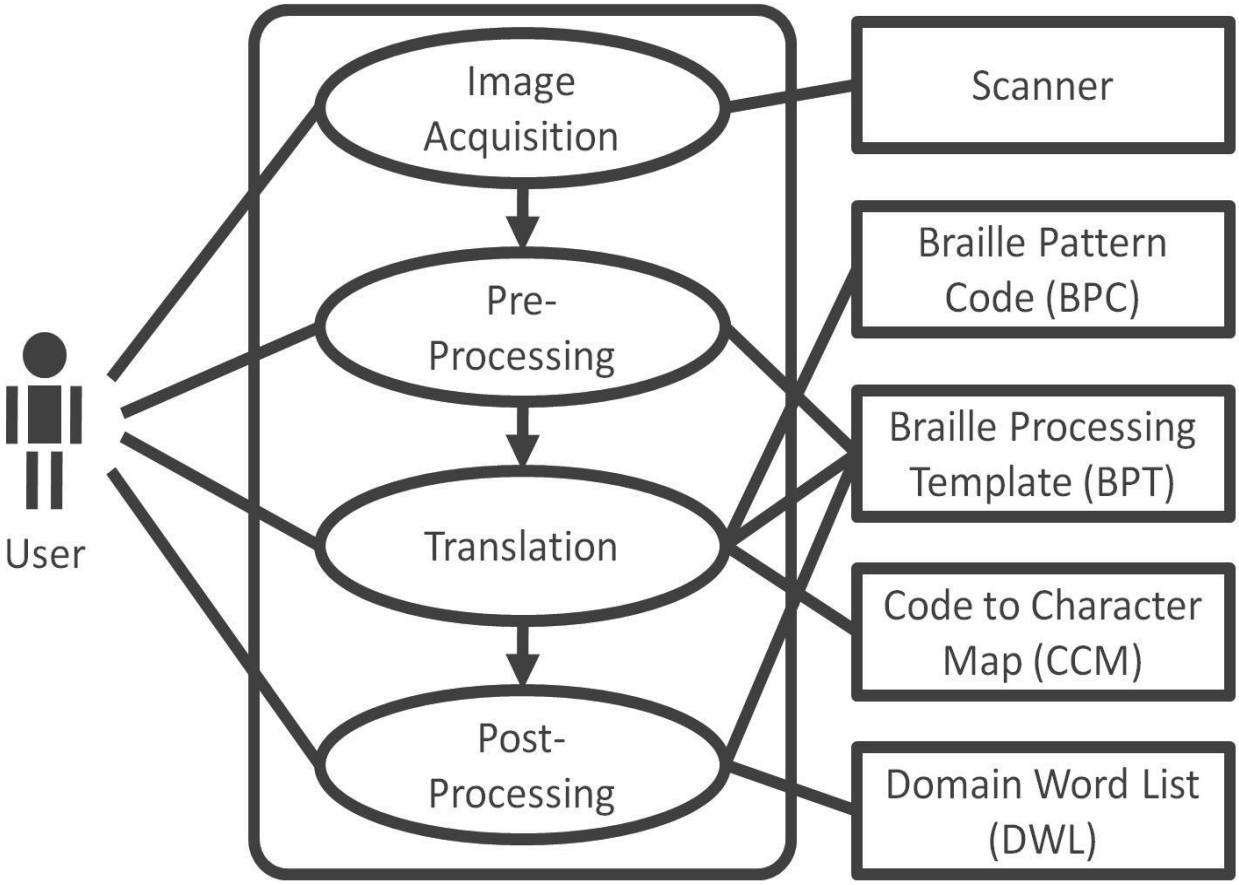



Figure 4.3.1: Use Case Diagram of BBTT (Level-1)

#### 4.3.1.1 Use Case: Image Acquisition

**Primary Actor:** User.

**Secondary Actor:** Scanner, Braille Processing Template.

**Goal in Context:** Use scanner to scan Braille writing and initialize the system.

**Scenario:**

1. Place Braille writing sheets in scanner.
2. Scan and store images.
3. Input images into the system.
4. Select Braille Processing Template (BPT) if necessary.

**Exceptions:**

1. System failure.
2. Error in scanner connection.

**Priority:** Moderate, may be implemented.

**When Available:** First increment.

**Frequency of Use:** Several times per week.

Figure 4.3.1.1a is the Activity Diagram of Image Acquisition-

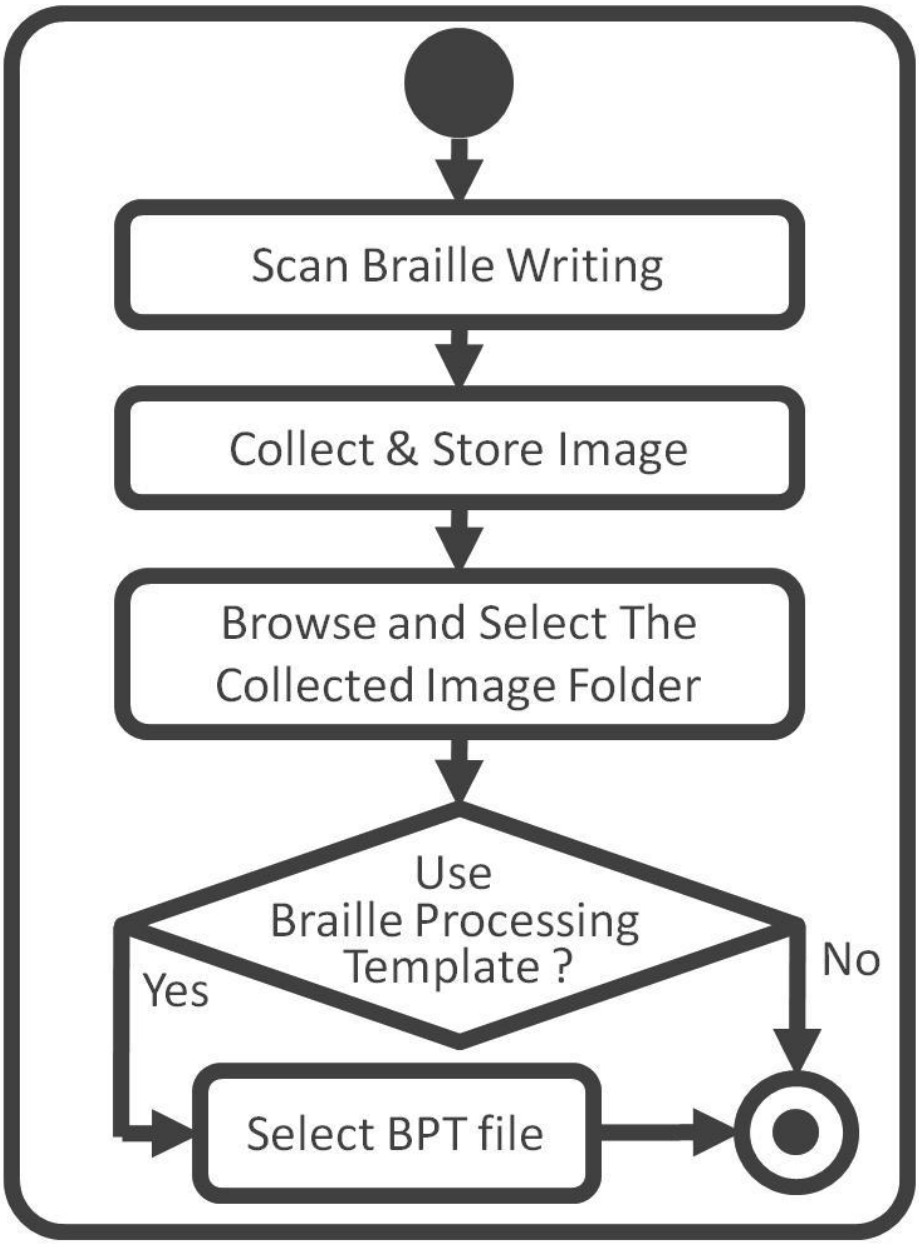



Figure 4.3.1.1a: Activity Diagram- Image Acquisition

Figure 4.3.1.1b is the Swim-Lane Diagram of Image Acquisition-

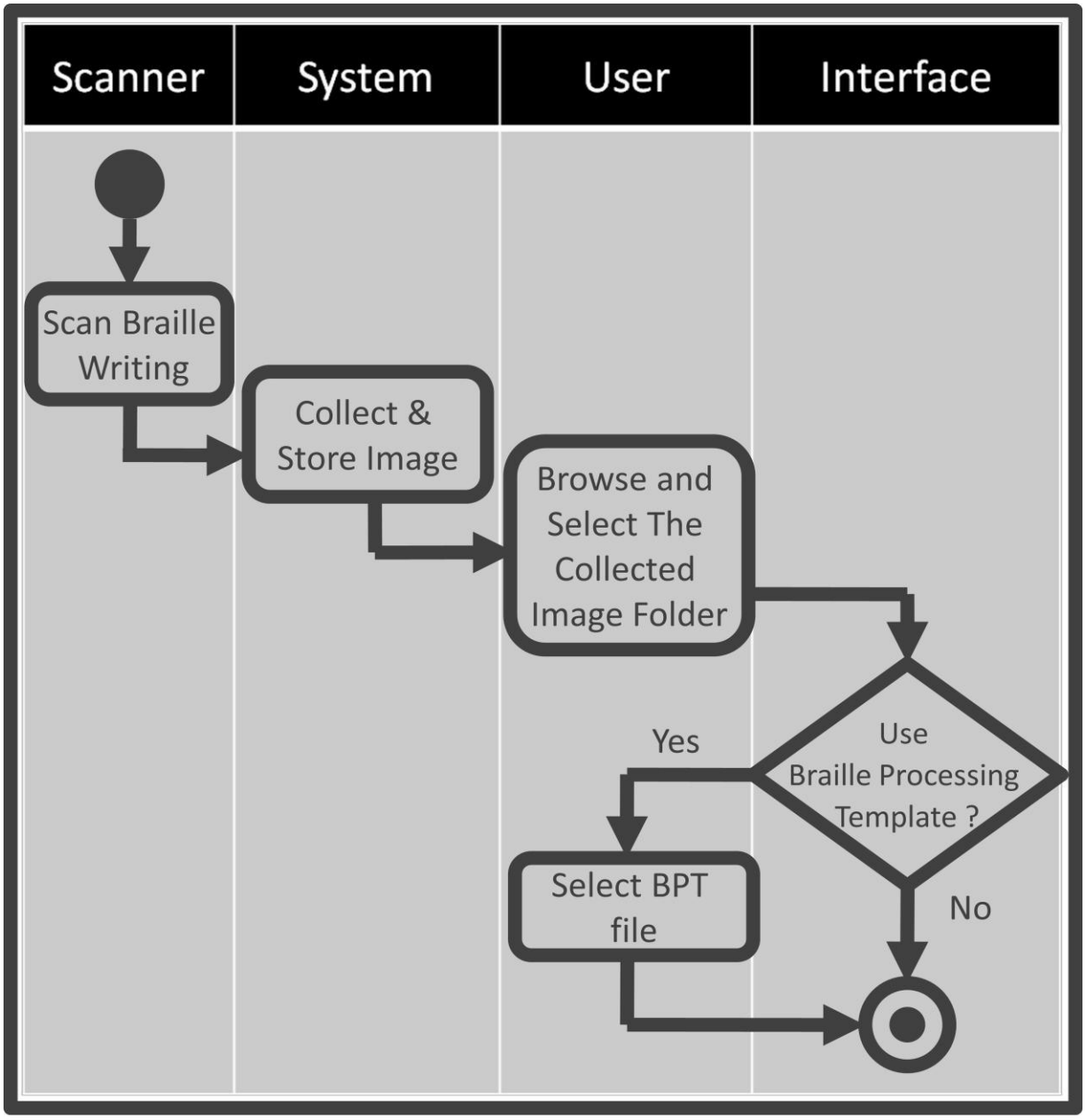



Figure 4.3.1.1b: Swim-Lane Diagram- Image Acquisition

### 4.3.1.2 Pre-Processing

Pre-Processing phase can be divided into five sub-phases. Figure-4.3.1.2 shows this:

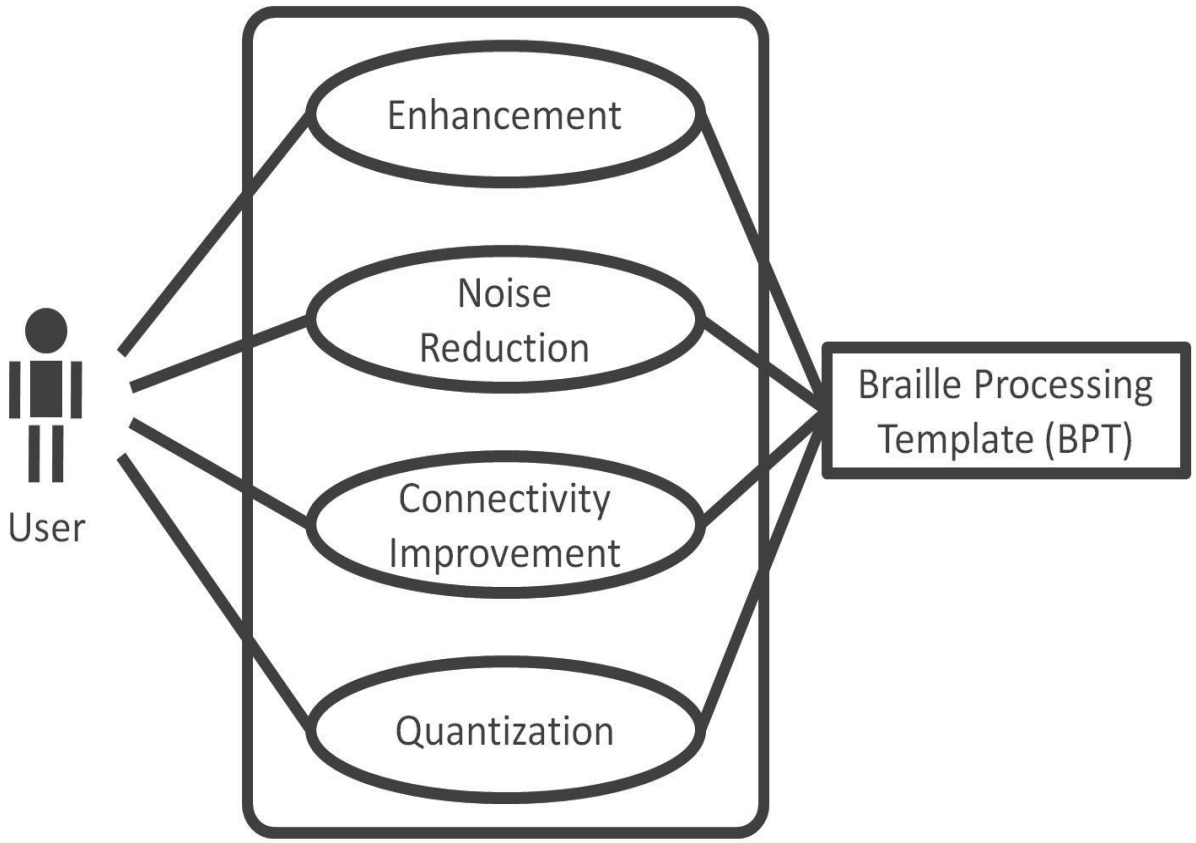



Figure 4.3.1.2: Use Case Diagram of Pre-Processing (Level-1.2)

#### 4.3.1.2.1 Use Case: Enhancement

**Primary Actor:** User.

**Secondary Actor:** Braille Processing Template.

**Goal in Context:** Improve Braille patterns and make it distinguishable from the background.

**Scenario:**

1. Select image enhancement algorithm.
2. Set parameters.
3. Run process and store result.

**Exceptions:**

1. System failure.
2. Image file size is too large.
3. Unsupported image file format.

**Priority:** Moderate, may be implemented.

**When Available:** Second increment.

**Frequency of Use:** Several times per day.

Figure 4.3.1.2.1a is the Activity Diagram of Enhancement-

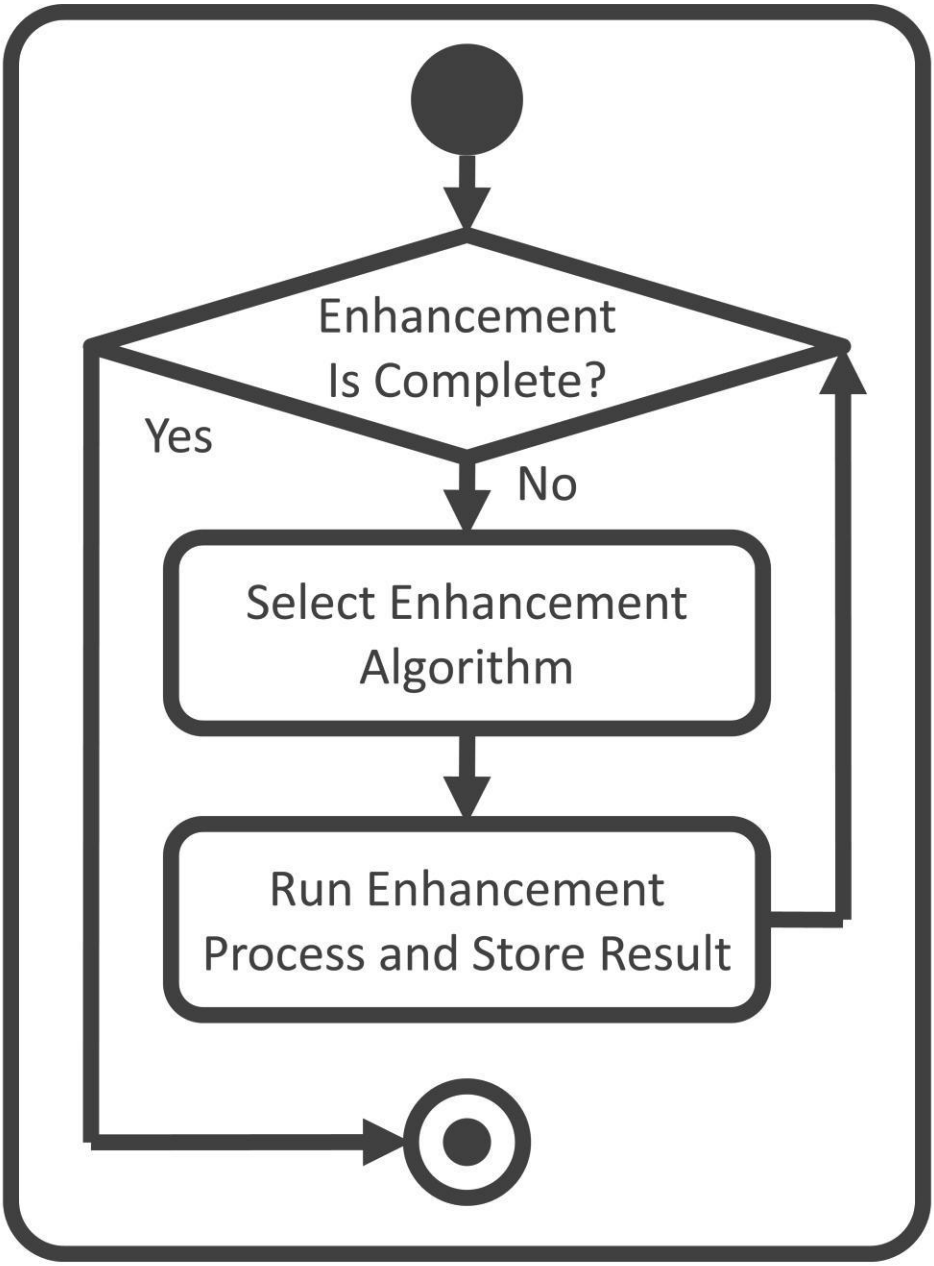



Figure 4.3.1.2.1a: Activity Diagram- Enhancement

Figure 4.3.1.2.1b is the Swim-Lane Diagram of Enhancement -

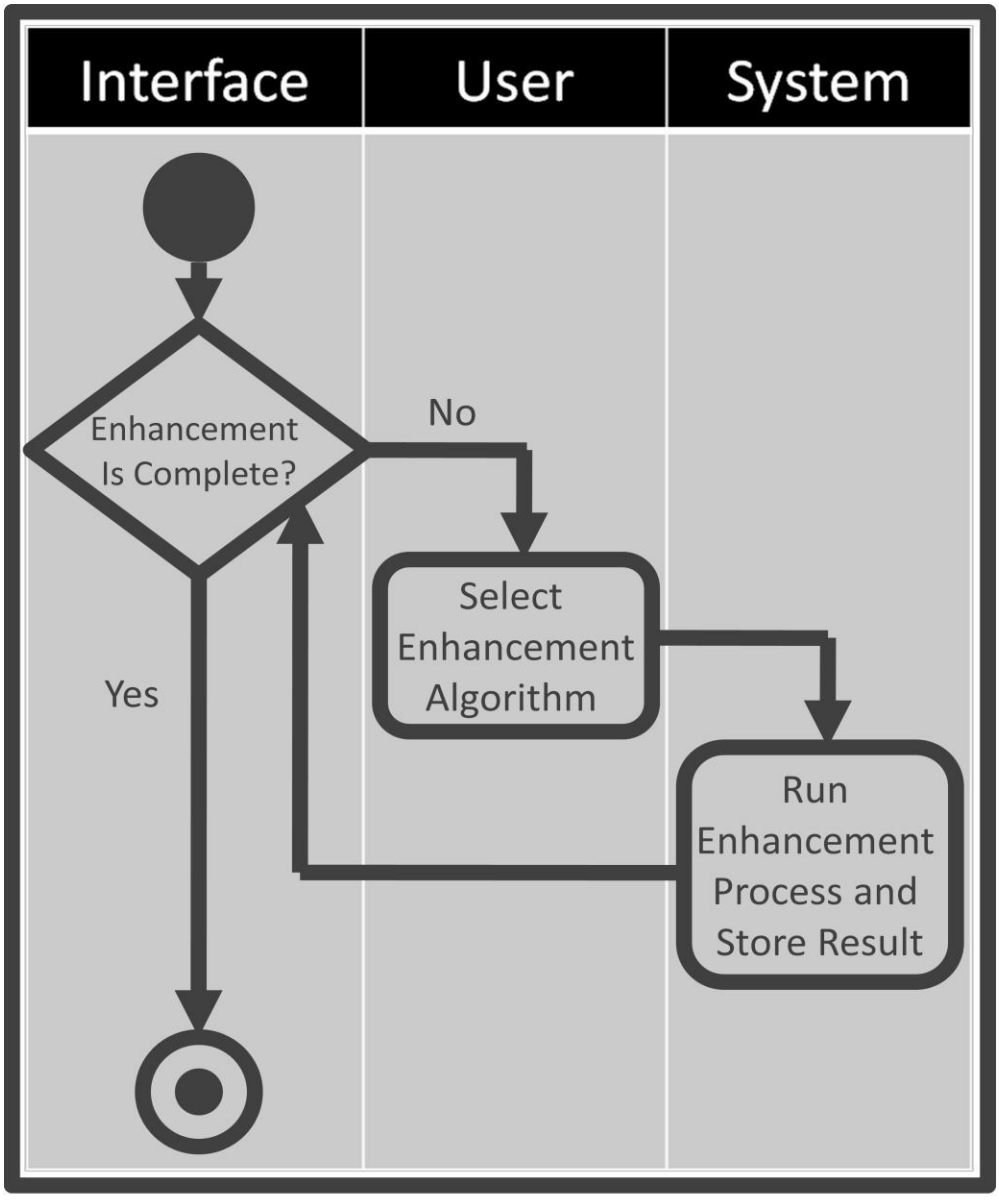



Figure 4.3.1.2.1b: Swim-Lane Diagram-Enhancement

#### 4.3.1.2.2 Use Case: Noise Reduction

**Primary Actor:** User.

**Secondary Actor:** Braille Processing Template.

**Goal in Context:** Reducing artifacts and improve pattern quality.

**Scenario:**

1. Select noise reduction algorithm.
2. Set parameters.
3. Run process and store result.

**Exceptions:**

1. System failure.
2. Unsupported image file format.

**Priority:** Moderate, may be implemented.

**When Available:** Second increment.

**Frequency of Use:** Several times per day.

Figure 4.3.1.2.2a is the Activity Diagram of Noise Reduction-

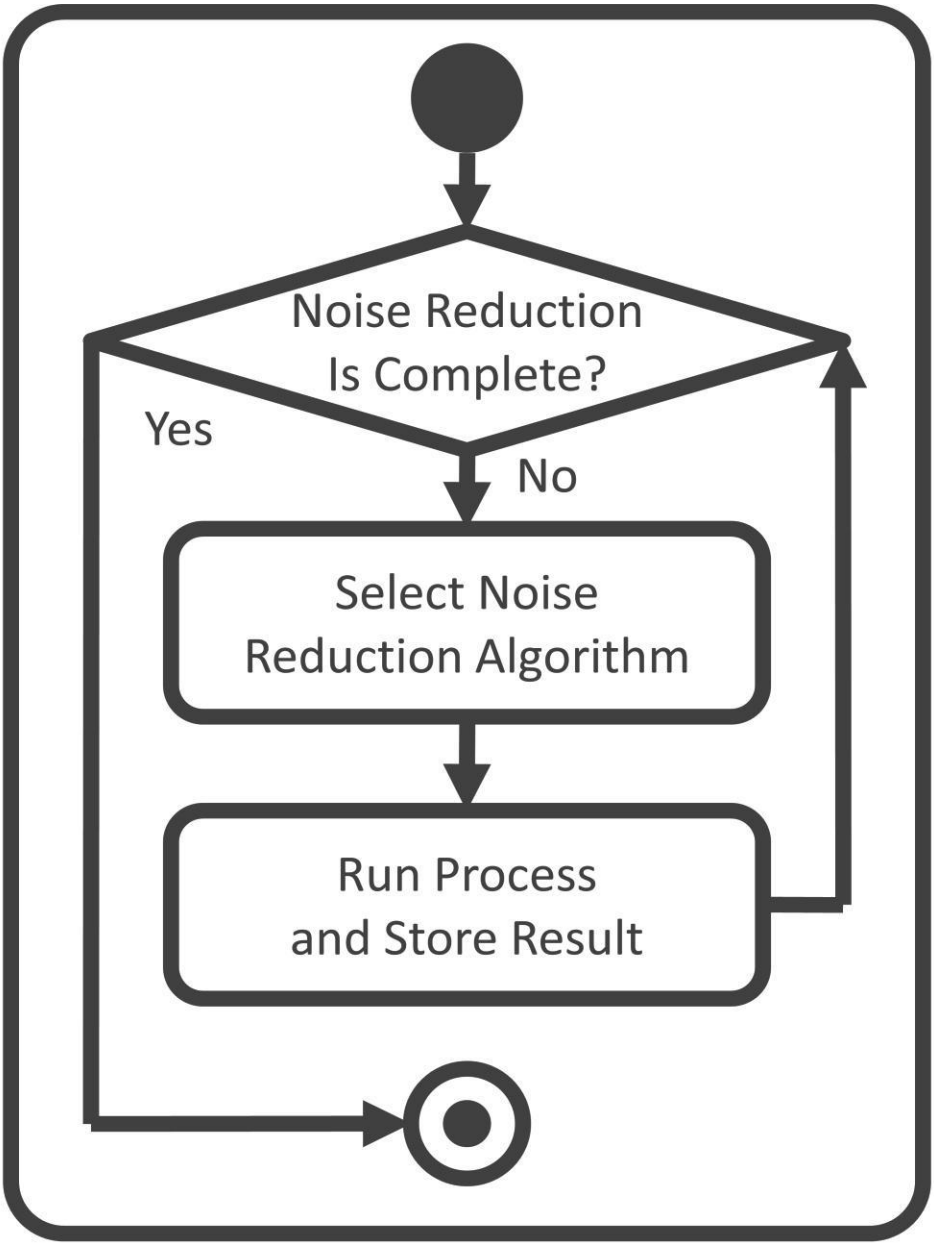



Figure 4.3.1.2.2a: Activity Diagram- Noise Reduction

Figure 4.3.1.2.2b is the Swim-Lane Diagram of Noise Reduction-

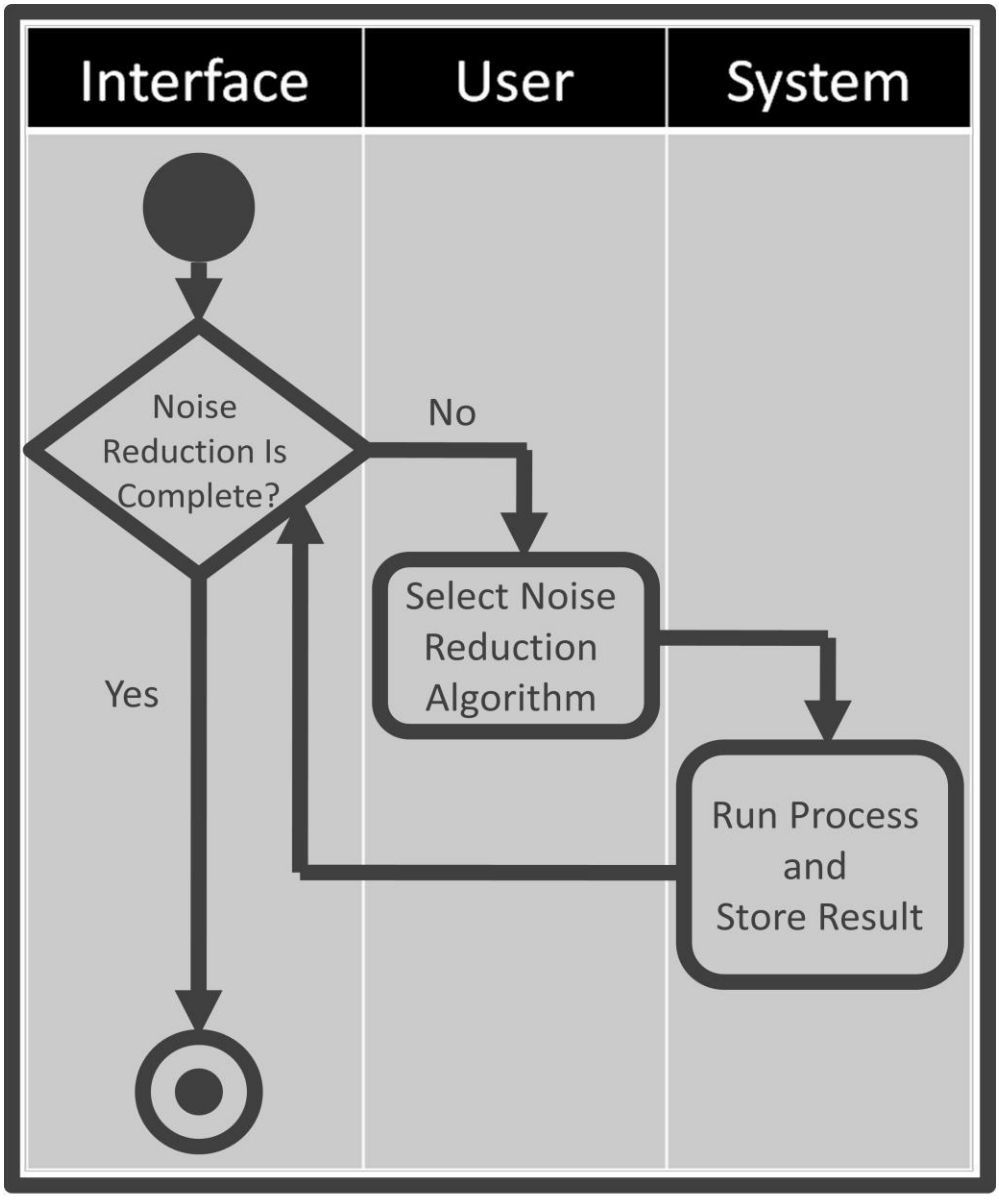



Figure 4.3.1.2.2b: Swim-Lane Diagram- Noise Reduction

#### 4.3.1.2.3 Use Case: Connectivity Improvement

**Primary Actor:** User.

**Secondary Actor:** Braille Processing Template.

**Goal in Context:** Improve Braille writing patterns.

**Scenario:**

1. Select algorithm.
2. Run process and store result.

**Exceptions:**

1. System failure.
2. Image file size is too large.
3. Unsupported image file format.

**Priority:** Moderate, may be implemented.

**When Available:** Second increment.

**Frequency of Use:** Several times per day.

Figure 4.3.1.2.3a is the Activity Diagram of Connectivity Improvement-

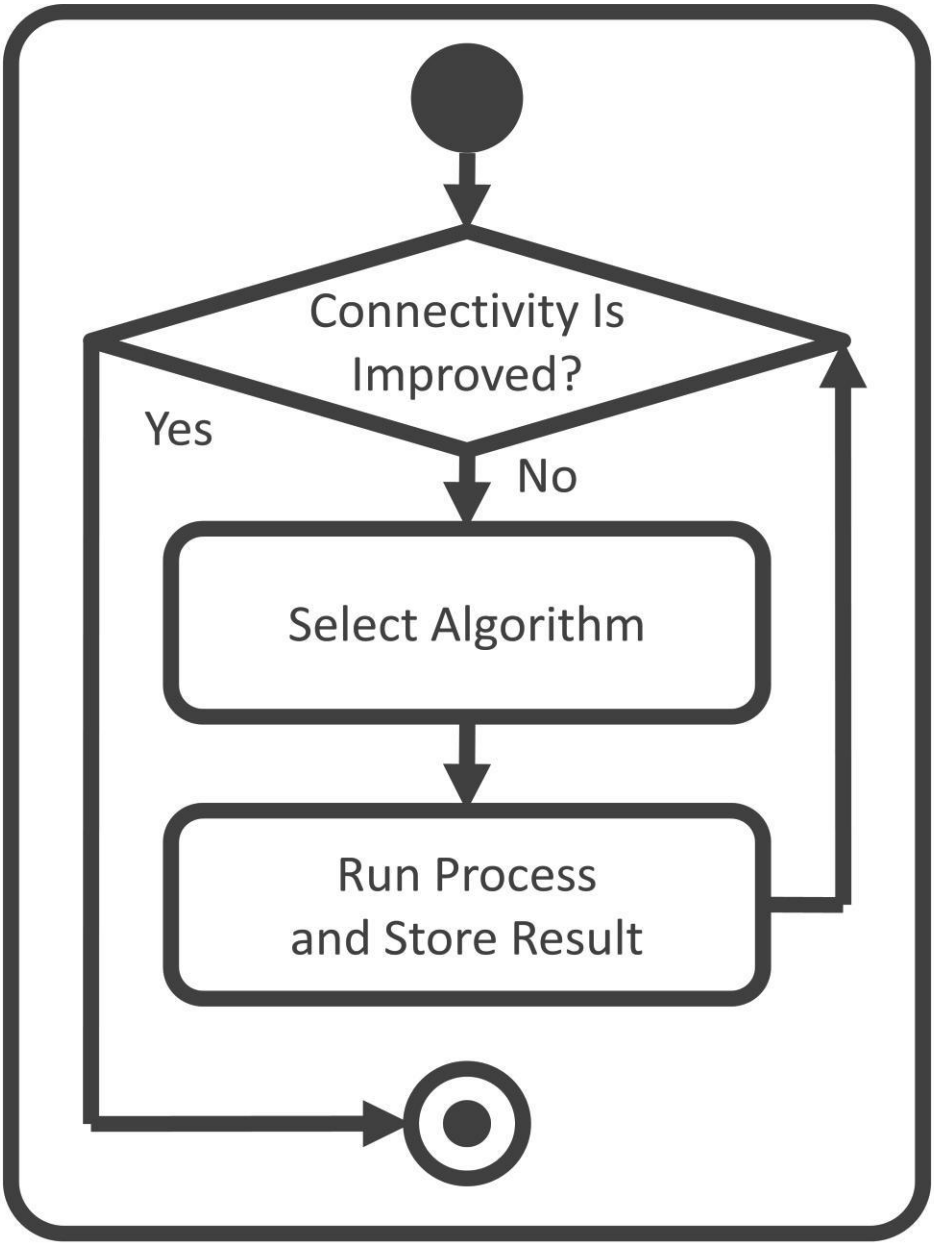



Figure 4.3.1.2.3a : Activity Diagram- Connectivity Improvement

Figure 4.3.1.2.3b is the Swim-Lane Diagram of Connectivity Improvement-

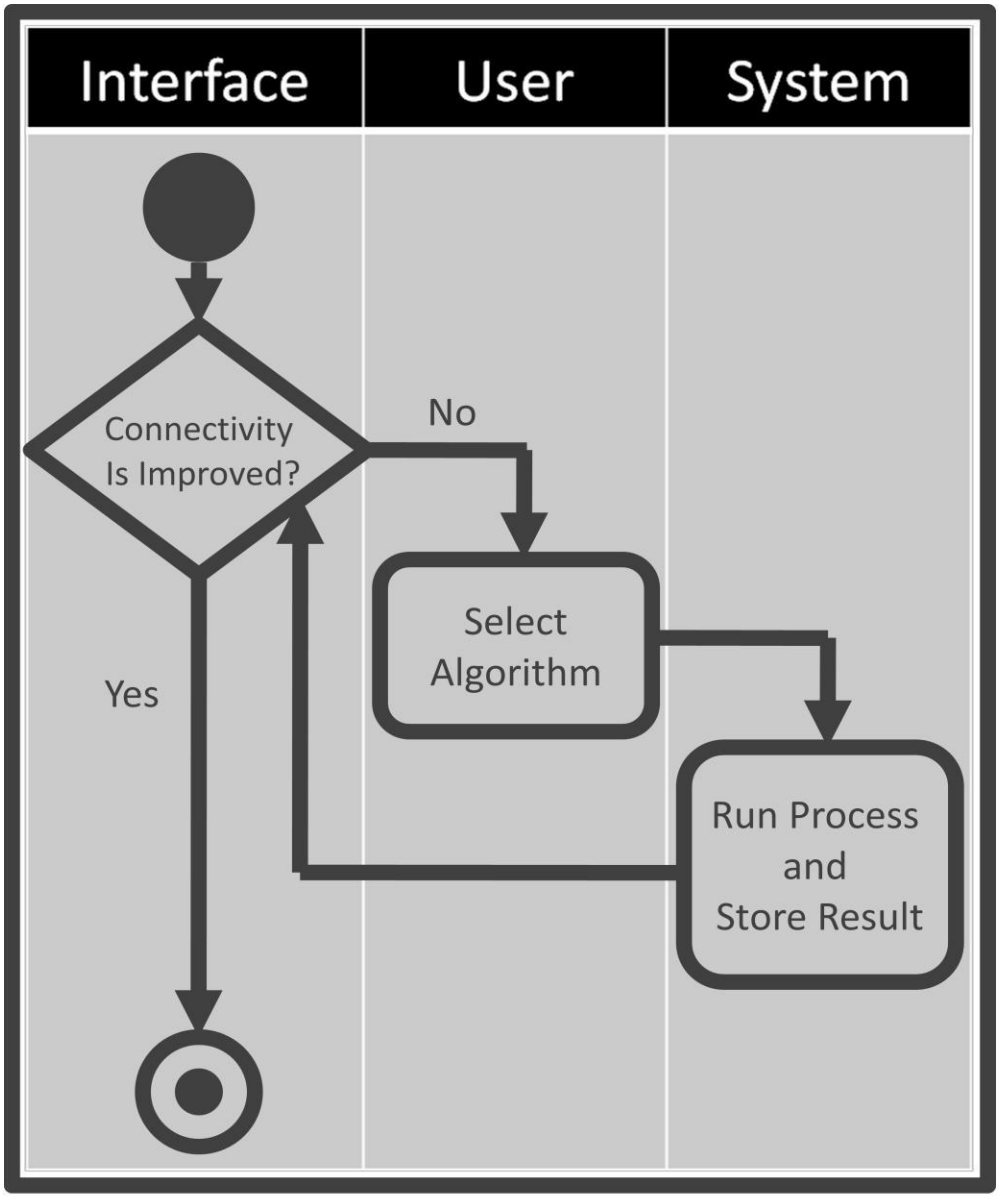



Figure 4.3.1.2.3b: Swim-Lane Diagram-
Connectivity Improvement

#### 4.3.1.2.4 Use Case: Quantization

**Primary Actor:** User.

**Secondary Actor:** Braille Processing Template.

**Goal in Context:** Improve Braille writing patterns.

**Scenario:**

1. Set parameters.
2. Run process and store result.

**Exceptions:**

1. System failure.
2. Unsupported image file format.

**Priority:** Moderate, may be implemented.

**When Available:** Second increment.

**Frequency of Use:** Several times per day.

Figure 4.3.1.2.4a is the Activity Diagram of Quantization-

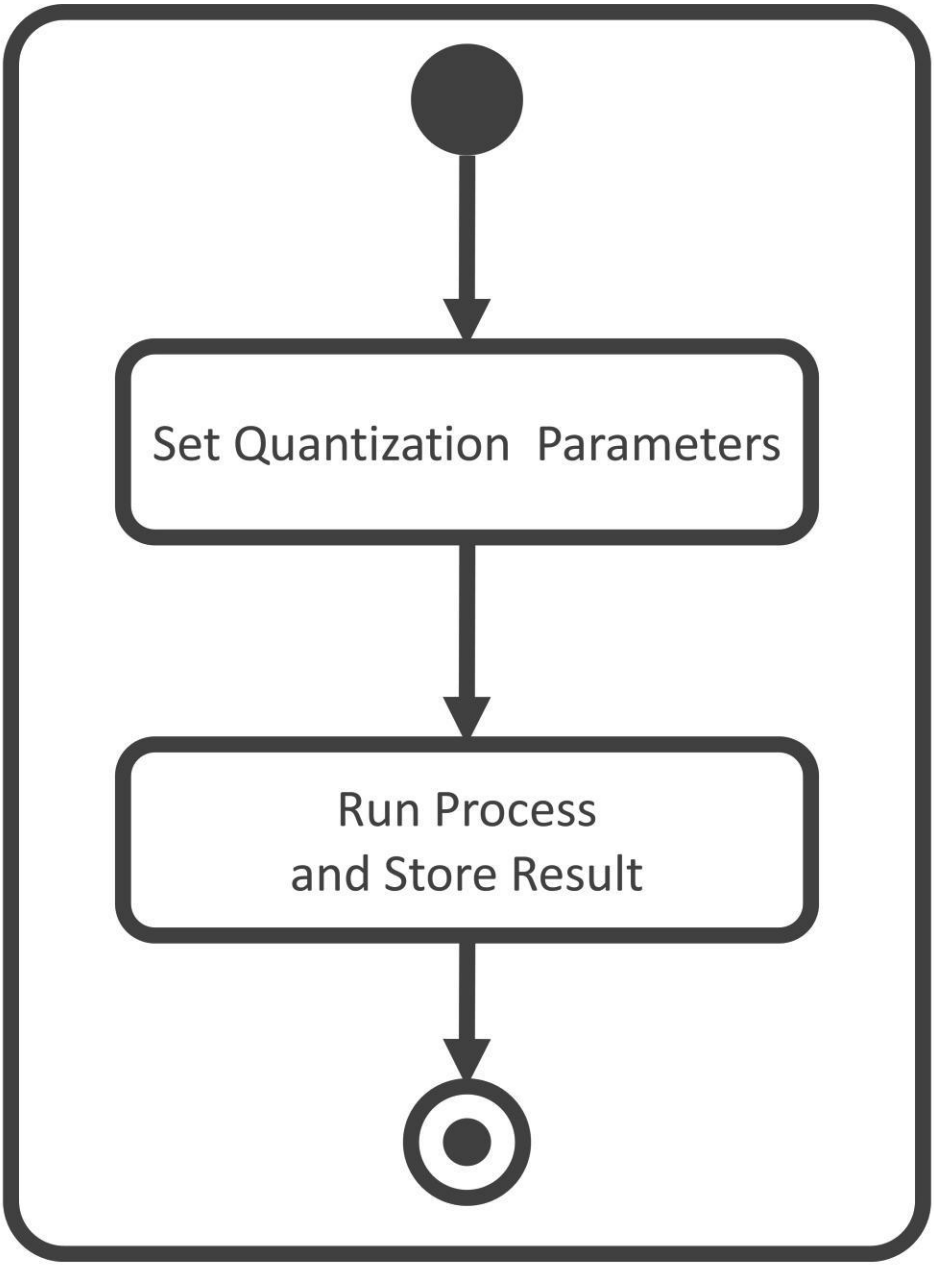



Figure 4.3.1.2.4a : Activity Diagram- Quantization

Figure 4.3.1.2.4b is the Swim-Lane Diagram of Quantization-

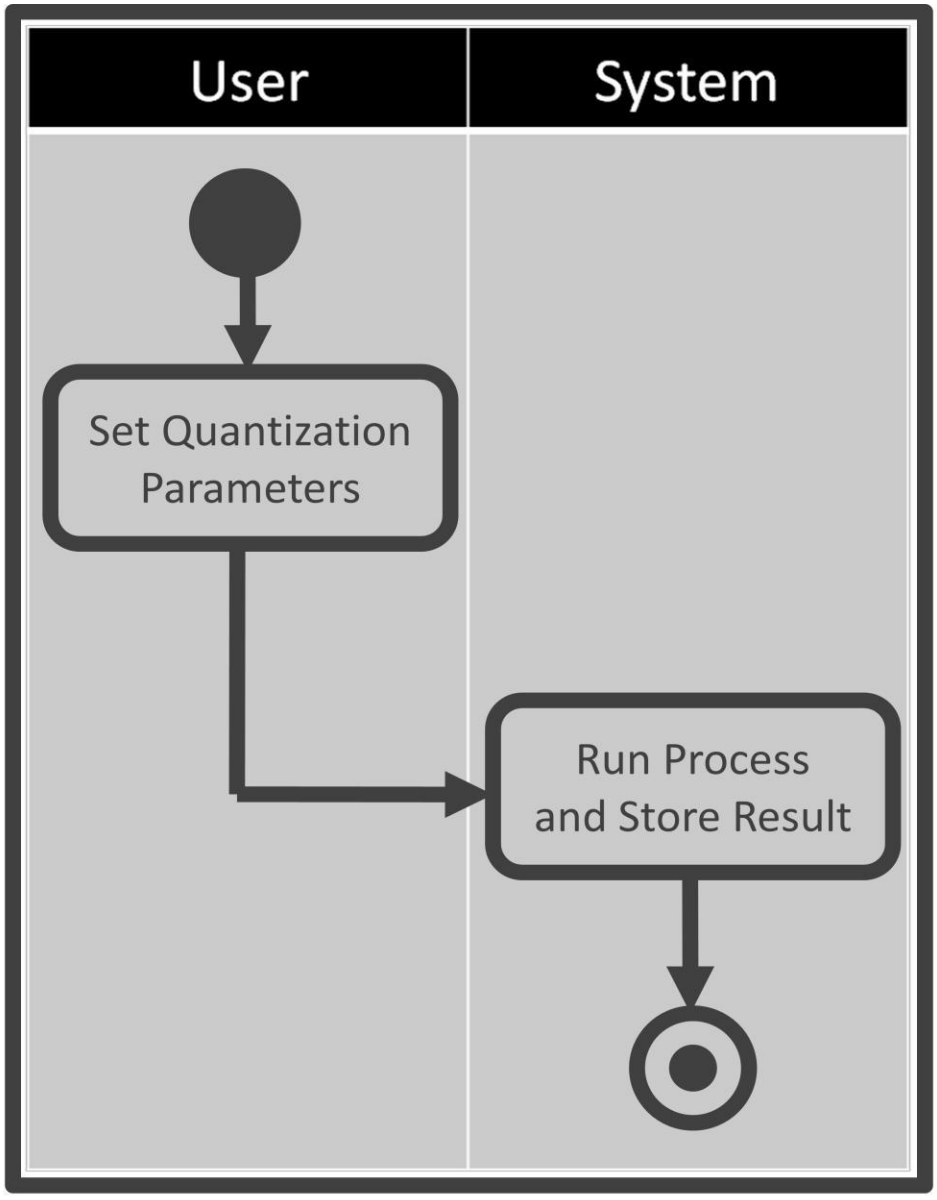



Figure 4.3.1.2.4b: Swim-Lane Diagram-Quantization

### 4.3.1.3 Translation

Further section of Translation system creates three sub-systems:

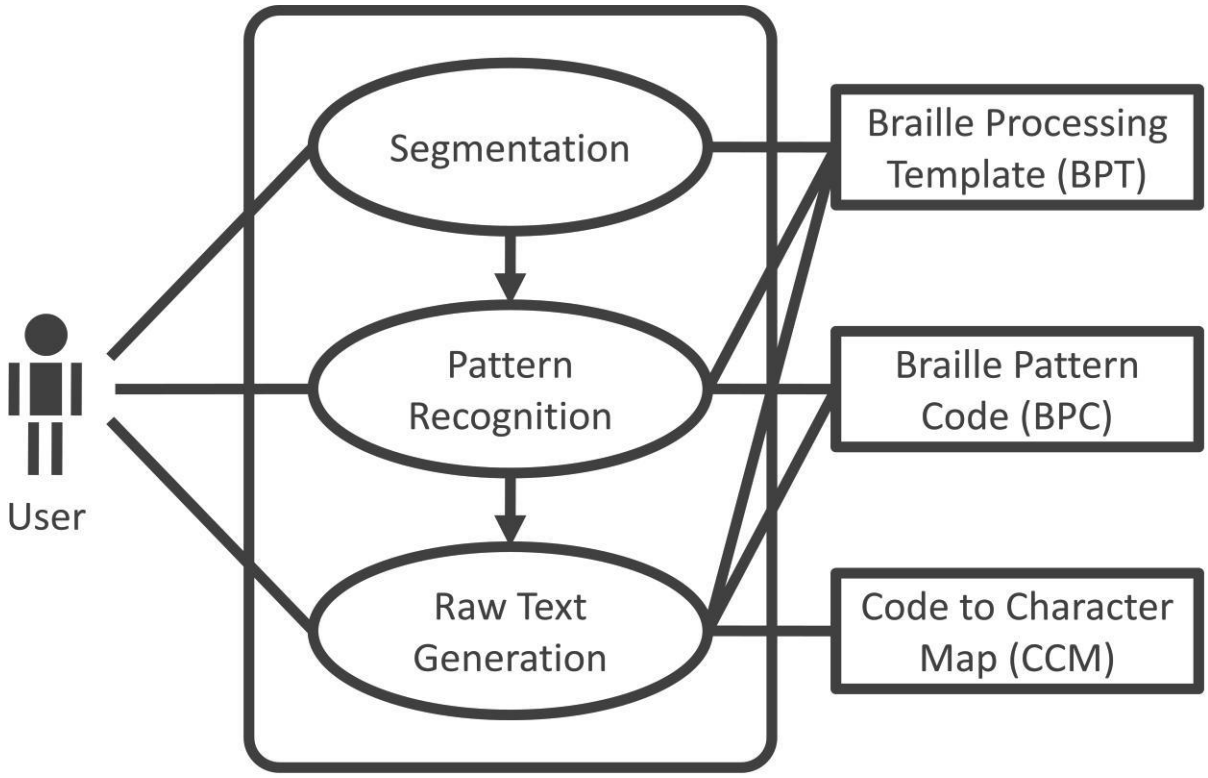



Figure 4.3.1.3: Use Case Diagram of Translation (Level-1.3)

#### 4.3.1.3.1 Use Case: Segmentation

**Primary Actor:** User.

**Secondary Actor:** Braille Processing Template.

**Goal in Context:** Extract Braille patterns from the image.

**Scenario:**

1. Select segmentation procedure.
2. Set parameters.
3. Run process.
4. Store resulting binary image.

**Exceptions:**

1. System failure.
2. Unsupported image file format.
3. Big image file.

**Priority:** Essential, must be implemented.

**When Available:** Third increment.

**Frequency of Use:** Several times per day.

Figure 4.3.1.3.1a is the Activity Diagram of Segmentation-

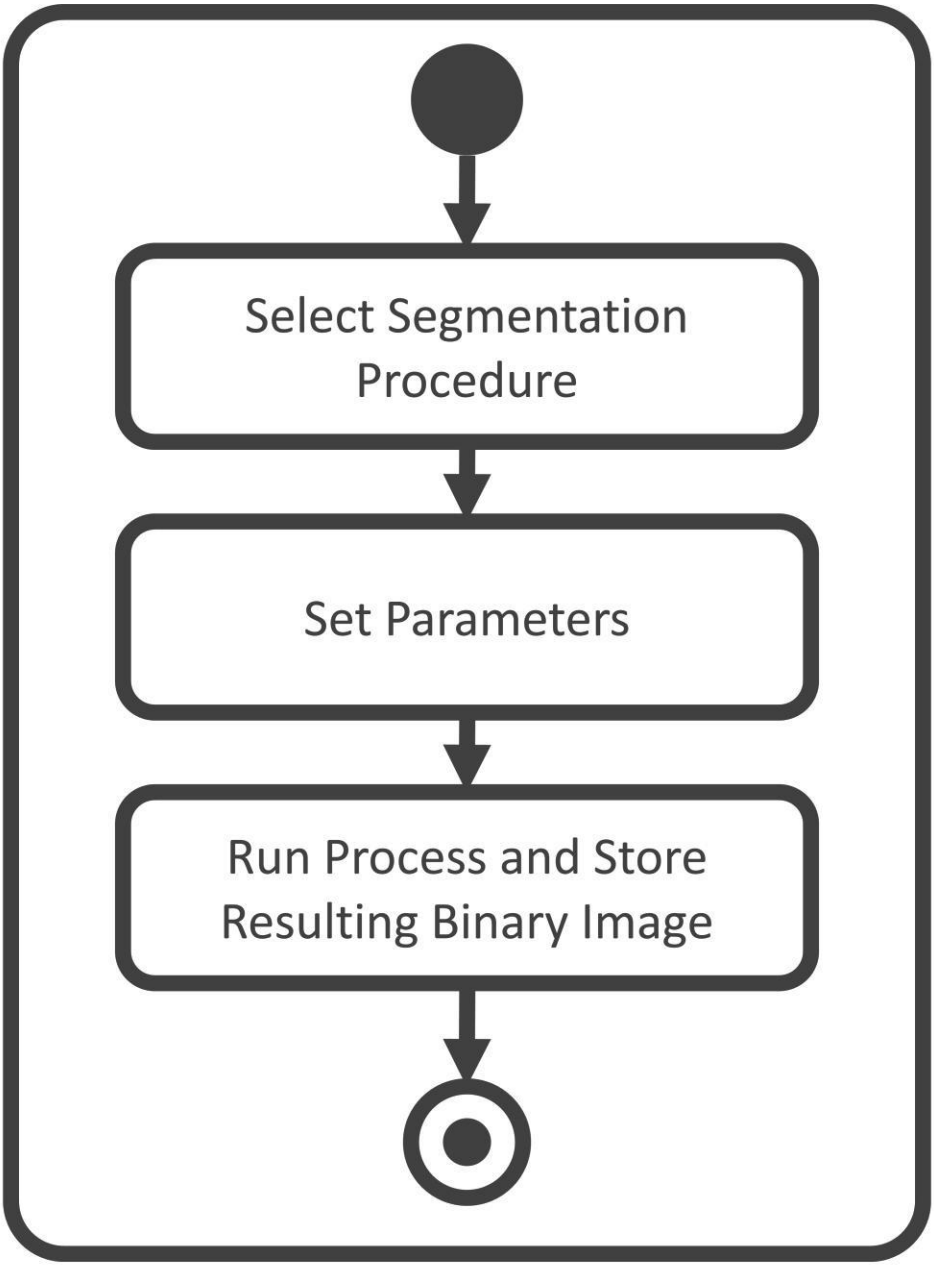



Figure 4.3.1.3.1a: Activity Diagram- Segmentation

Figure 4.3.1.3.1b is the Swim-Lane Diagram of Segmentation-

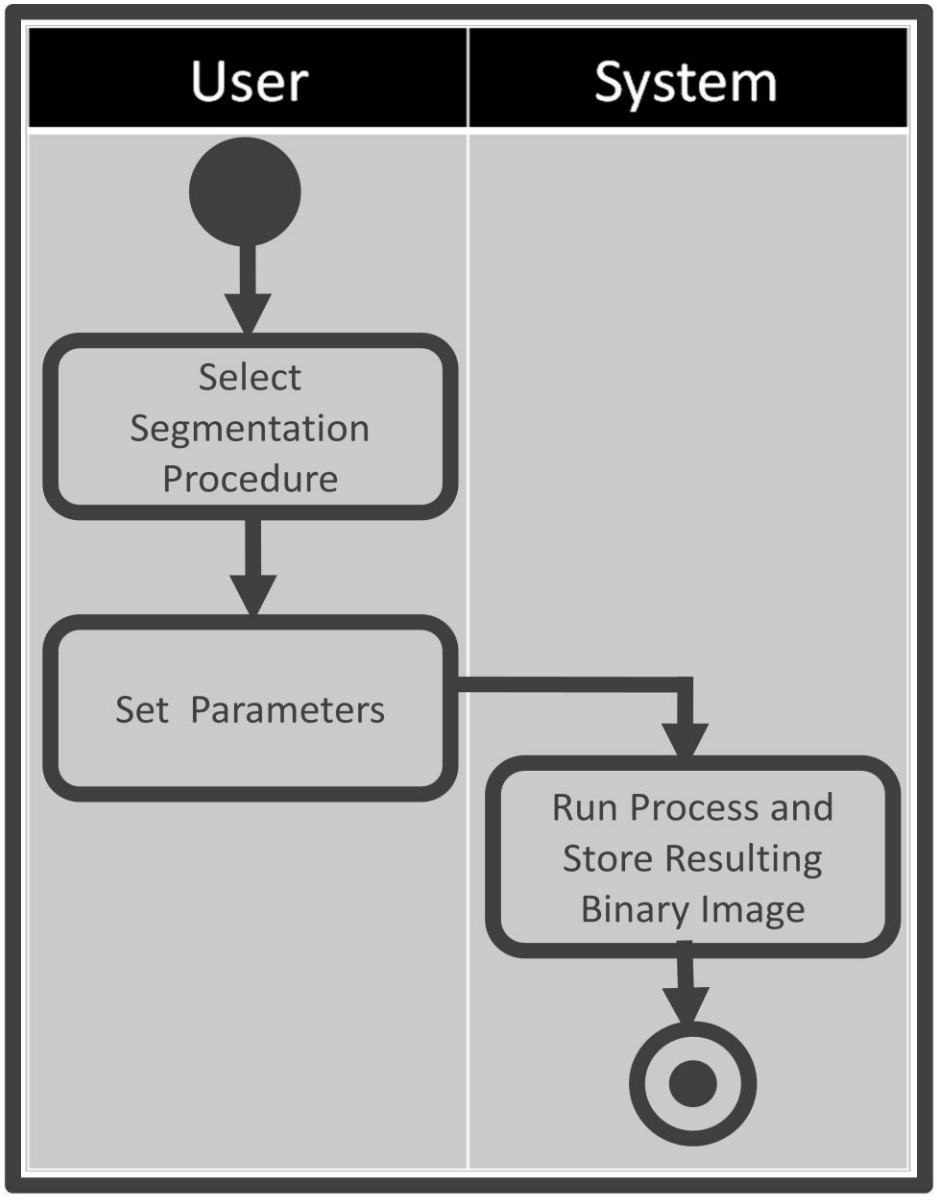



Figure 4.3.1.3.1b: Swim-Lane Diagram-Segmentation

#### 4.3.1.3.2 Use Case: Pattern Recognition

**Primary Actor:** User.

**Secondary Actor:** Braille Processing Template, Braille Pattern Code.

**Goal in Context:** Convert Braille patterns to code.

**Scenario:**

1. Select pattern recognition procedure.
2. Set parameters.
3. Run process.
4. Store code in text file.

**Exceptions:**

1. Image with no Braille pattern.
2. System failure.
3. Big image file.

**Priority:** Essential, must be implemented.

**When Available:** Fourth increment.

**Frequency of Use:** Several times per day.

Figure 4.3.1.3.2a is the Activity Diagram of Pattern Recognition-

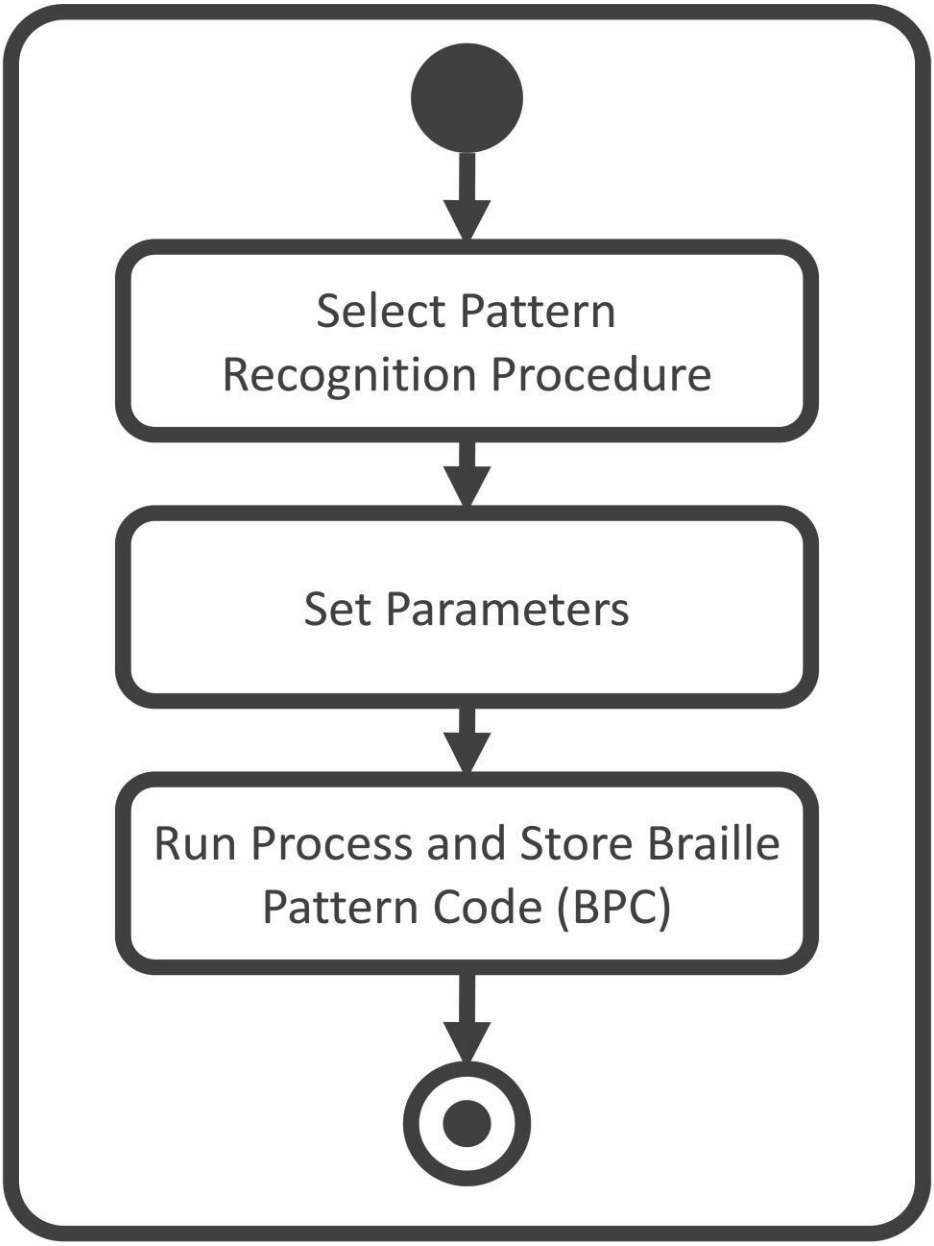



Figure 4.3.1.3.2a: Activity Diagram- Pattern Recognition

Figure 4.3.1.3.2b is the Swim-Lane Diagram of Pattern Recognition-

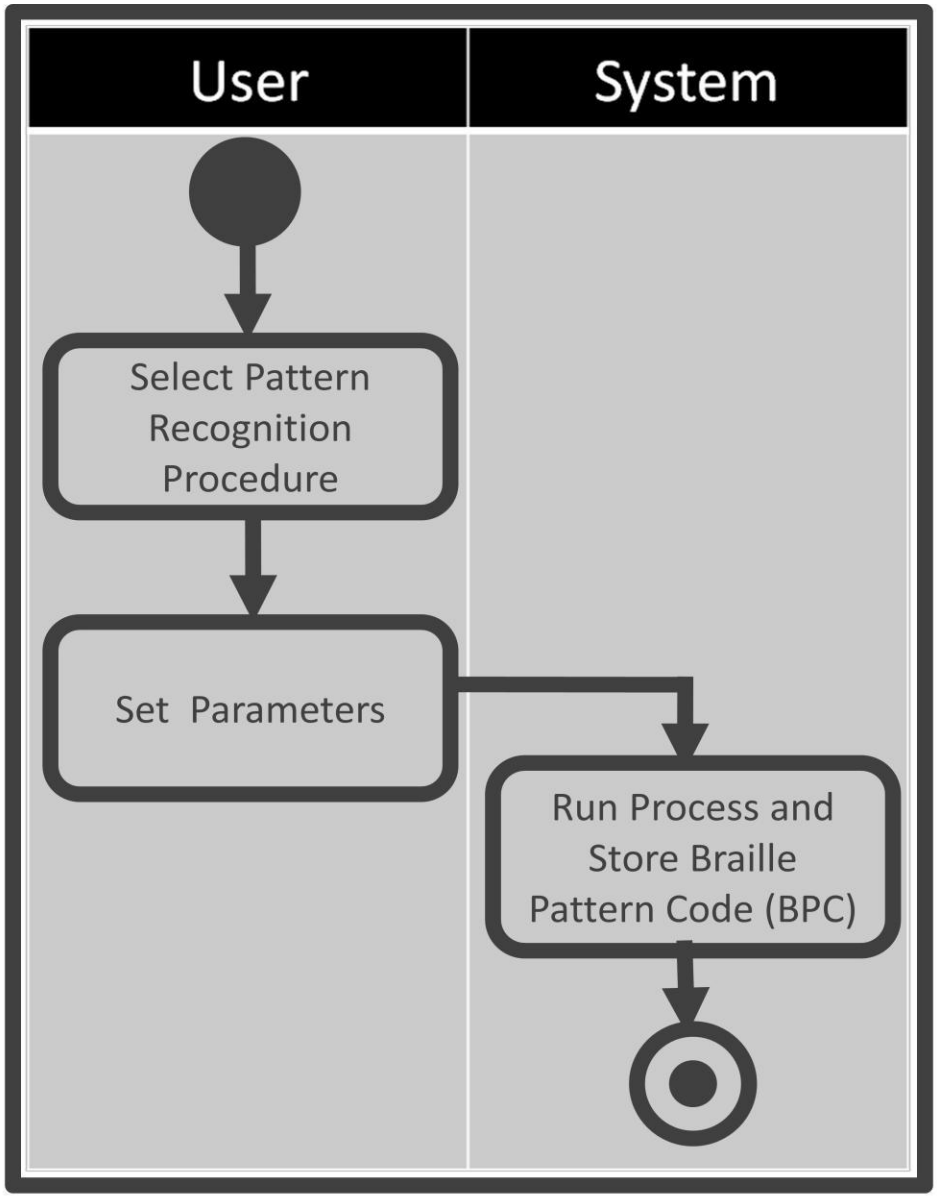



Figure 4.3.1.3.2b: Swim-Lane Diagram- Pattern Recognition

#### 4.3.1.3.3 Use Case: Raw Text Generation

**Primary Actor:** User.

**Secondary Actor:** Braille Processing Template, Braille Pattern Code, Code to Character Map.

**Goal in Context:** Generate plain text file.

**Scenario:**

1. Select Braille Pattern Code file.
2. Select Code to Character Map file.
3. Run process.
4. Store result in text file.

**Exceptions:**

1. Invalid pattern code.
2. System failure.

**Priority:** Essential, must be implemented.

**When Available:** Fifth increment.

**Frequency of Use:** Several times per day.

Figure 4.3.1.3.3a is the Activity Diagram of Raw Text Generation-

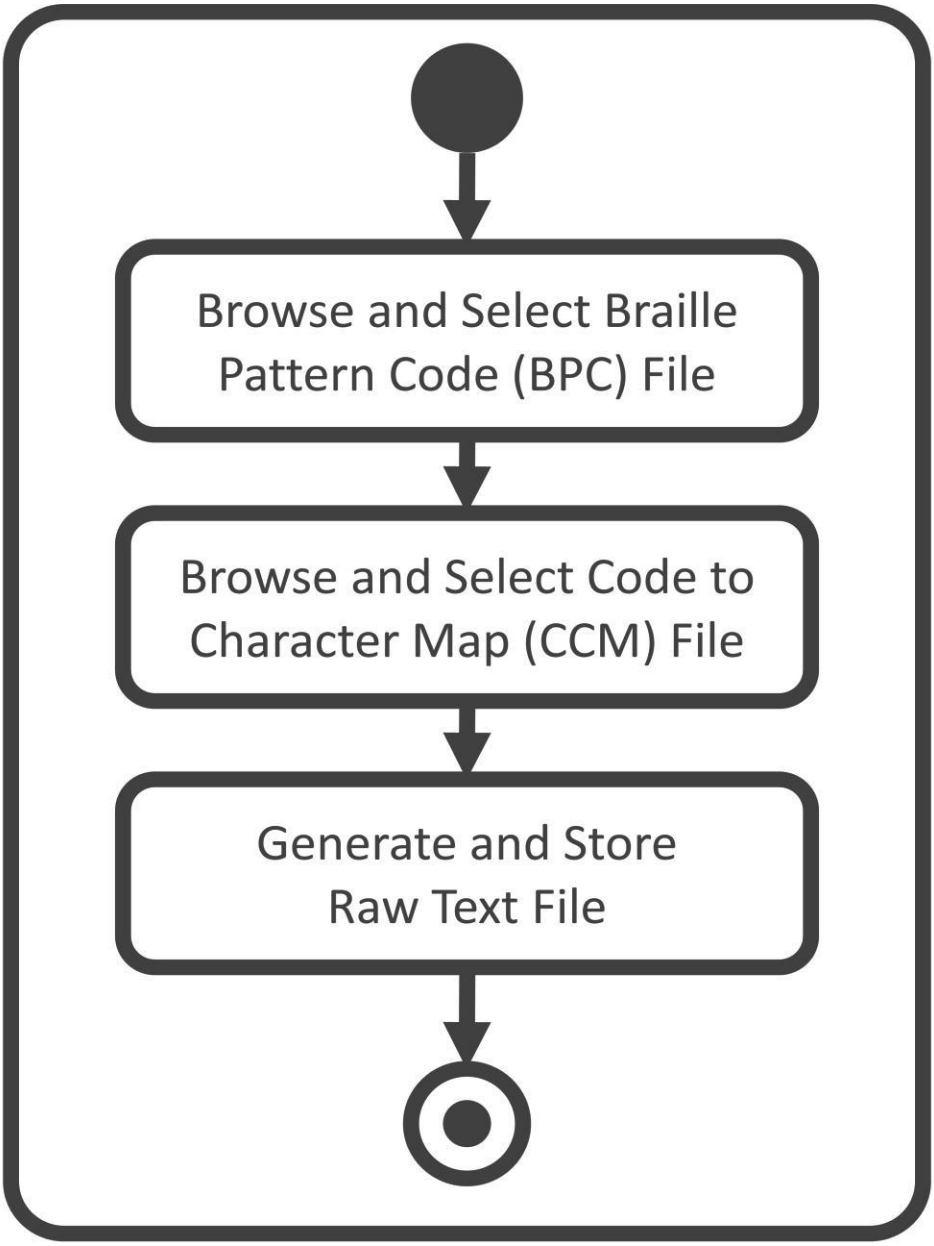



Figure 4.3.1.3.3a: Activity Diagram- Raw Text Generation

Figure 4.3.1.3.3b is the Swim-Lane Diagram of Raw Text Generation-

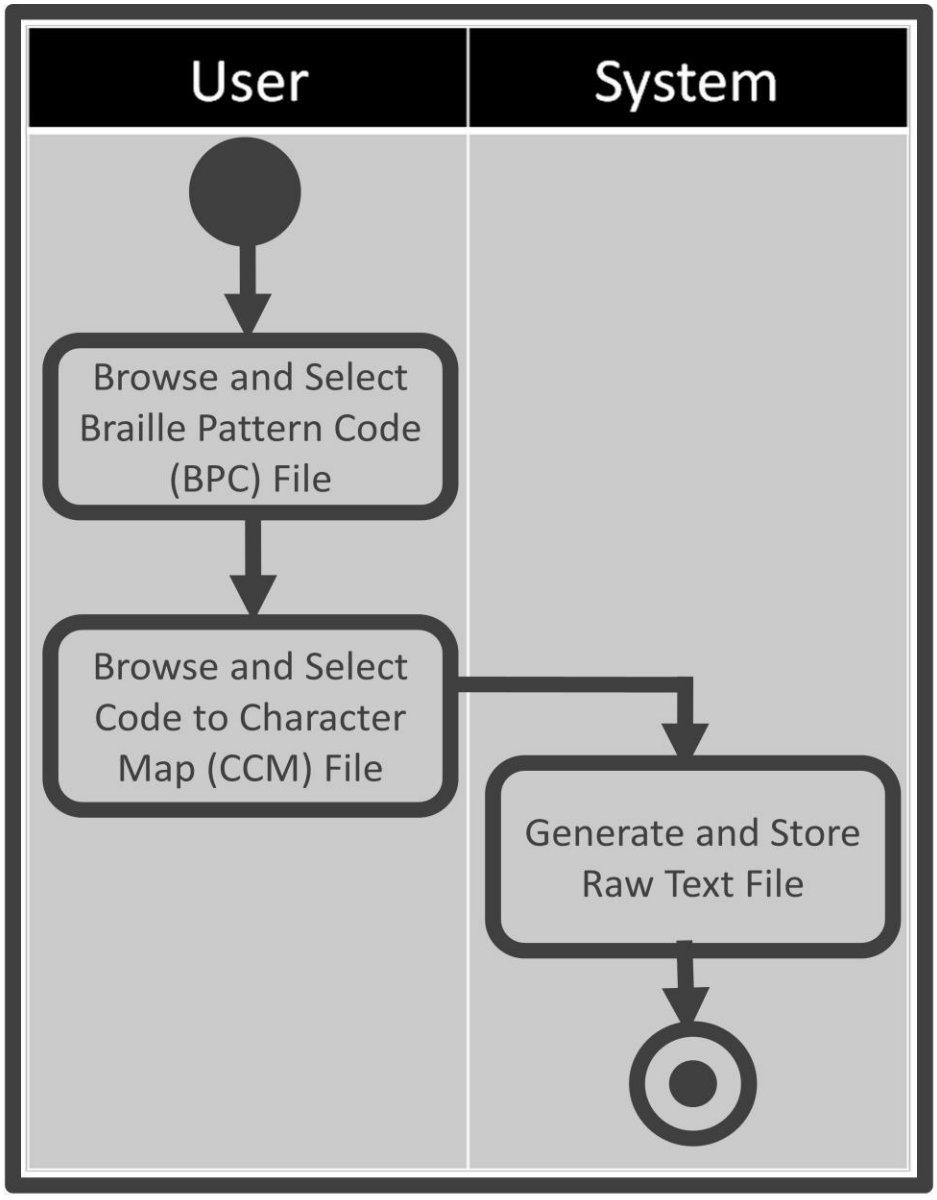



Figure 4.3.1.3.3b: Swim-Lane Diagram- Raw Text Generation

### 4.3.1.4 Post-Processing

Post-Processing has two major phases:

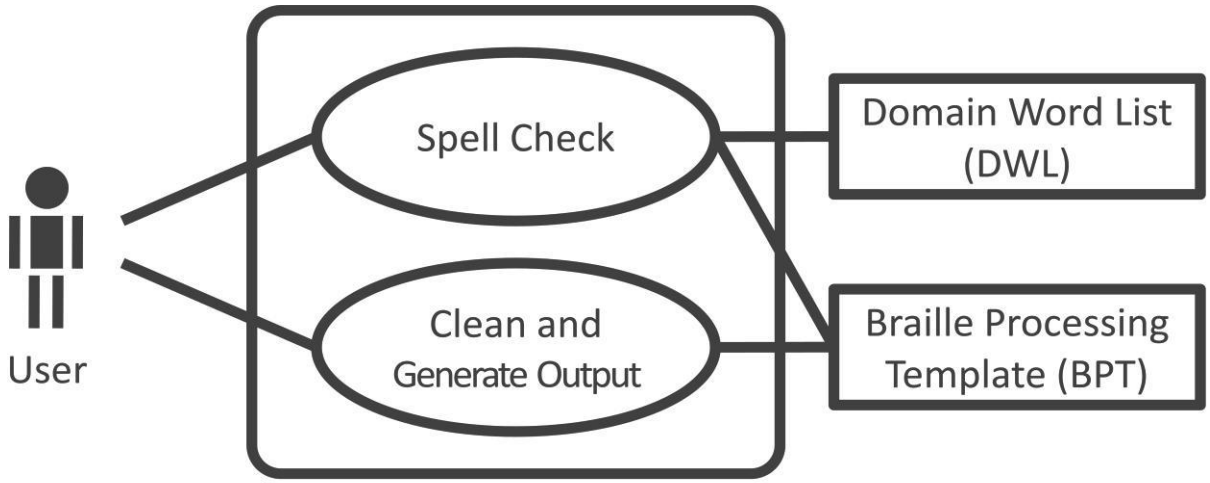



Figure 4.3.1.4: Use Case Diagram of Post-Processing
(Level-1.4)

#### 4.3.1.4.1 Use Case: Spell Check

**Primary Actor:** User.

**Secondary Actor:** Braille Processing Template, Domain Word List.

**Goal in Context:** Minimizing error rate.

**Scenario:**

1. Select raw text file.
2. Select Domain Word List file.
3. Run process.
4. Store corrected result in text file.

**Exceptions:**

1. System failure.

**Priority:** Moderate, may be implemented.

**When Available:** sixth increment.

**Frequency of Use:** Several times per day.

Figure 4.3.1.4.1a is the Activity Diagram of Spell Check-

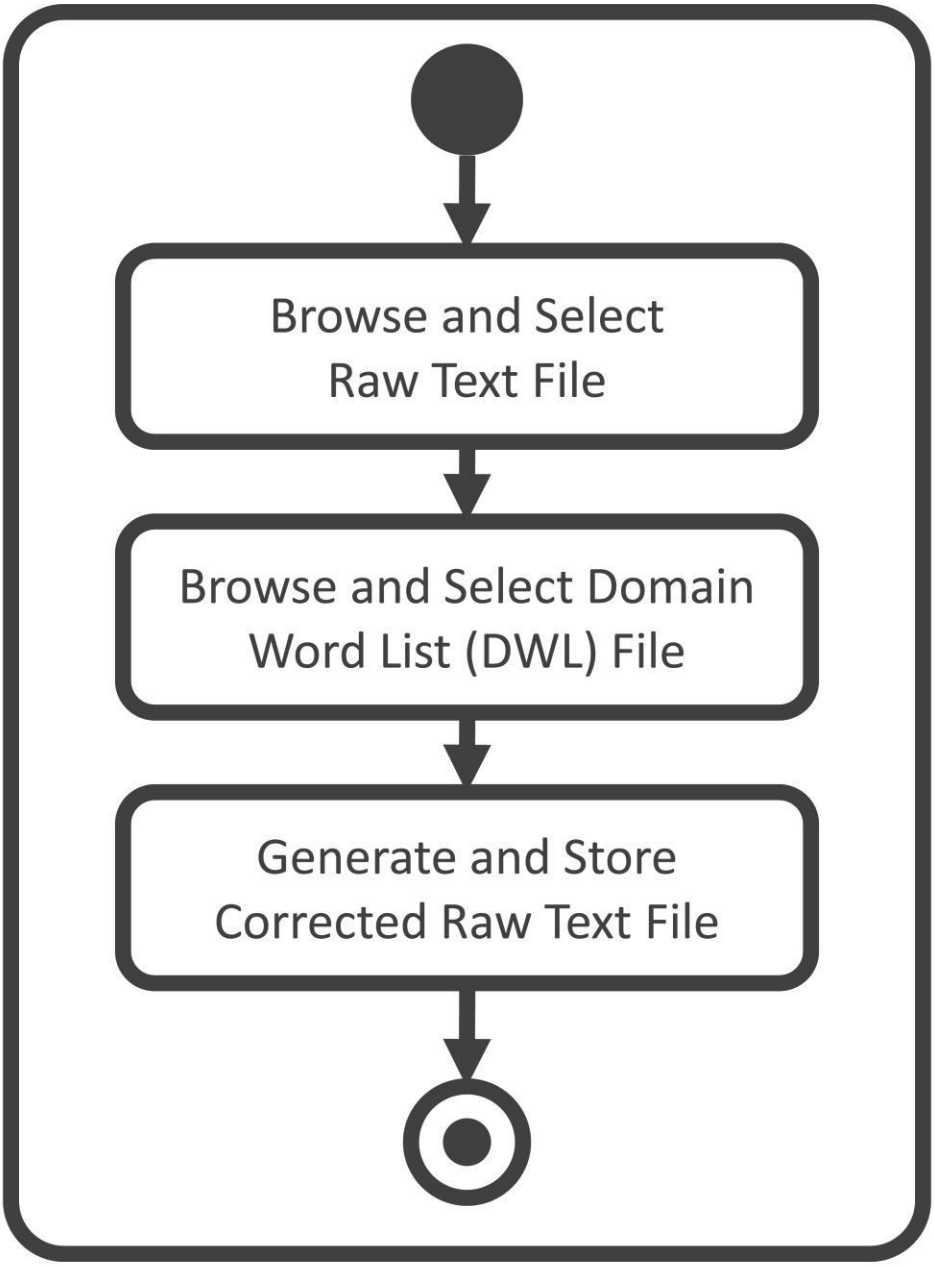



Figure 4.3.1.4.1a: Activity Diagram- Spell Check

Figure 4.3.1.4.1b is the Swim-Lane Diagram of Spell Check-

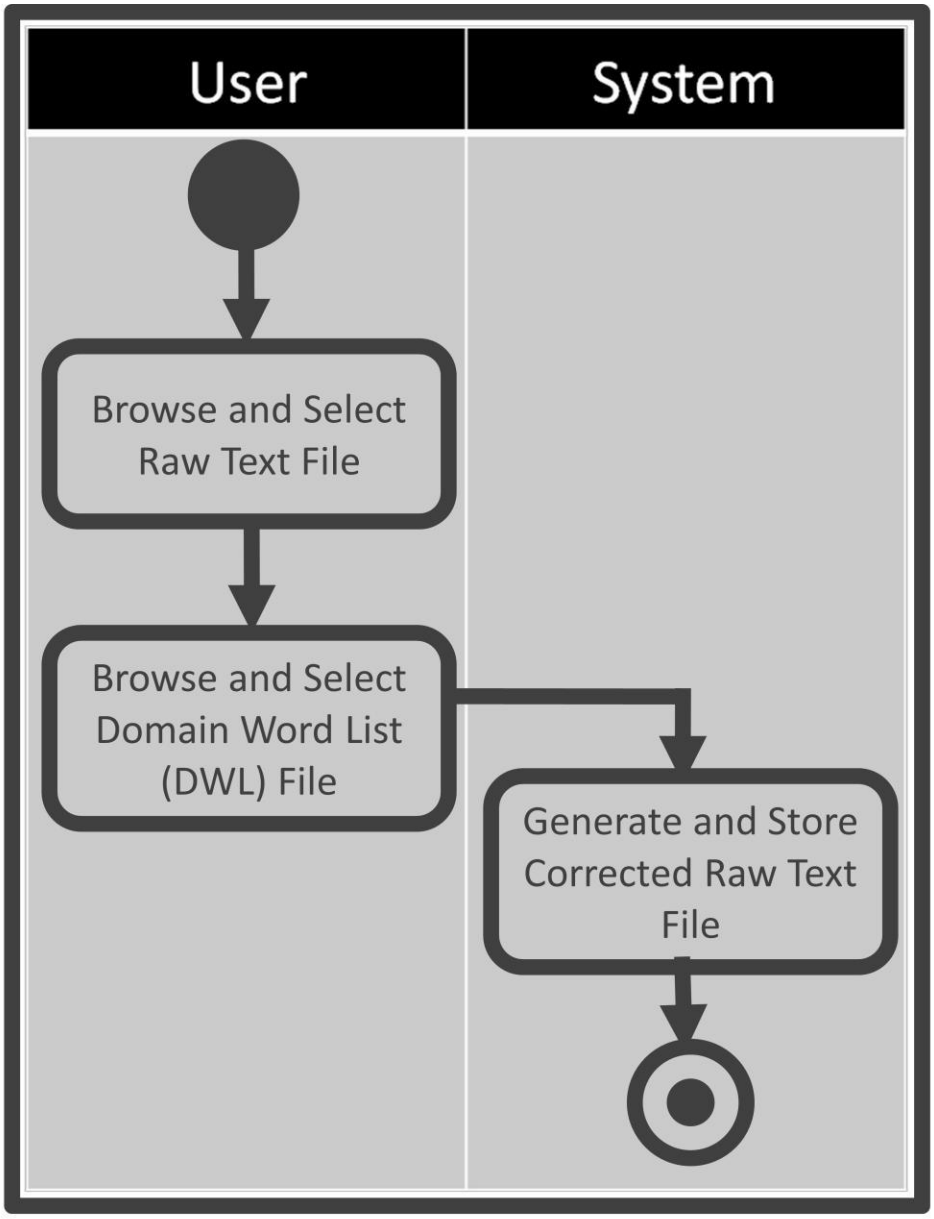



Figure 4.3.1.4.1b: Swim-Lane Diagram- Spell Check

#### 4.3.1.4.2 Use Case: Clean and Generate Output

**Primary Actor:** User.

**Secondary Actor:** Braille Processing Template.

**Goal in Context:** Generate final text file.

**Scenario:**

1. Select raw text file.
2. Run cleaning process.
3. Store result in text file.

**Exceptions:**

1. System failure.

**Priority:** Essential, must be implemented.

**When Available:** Last increment.

Figure 4.3.1.4.2a is the Activity Diagram of Clean and Generate Output-

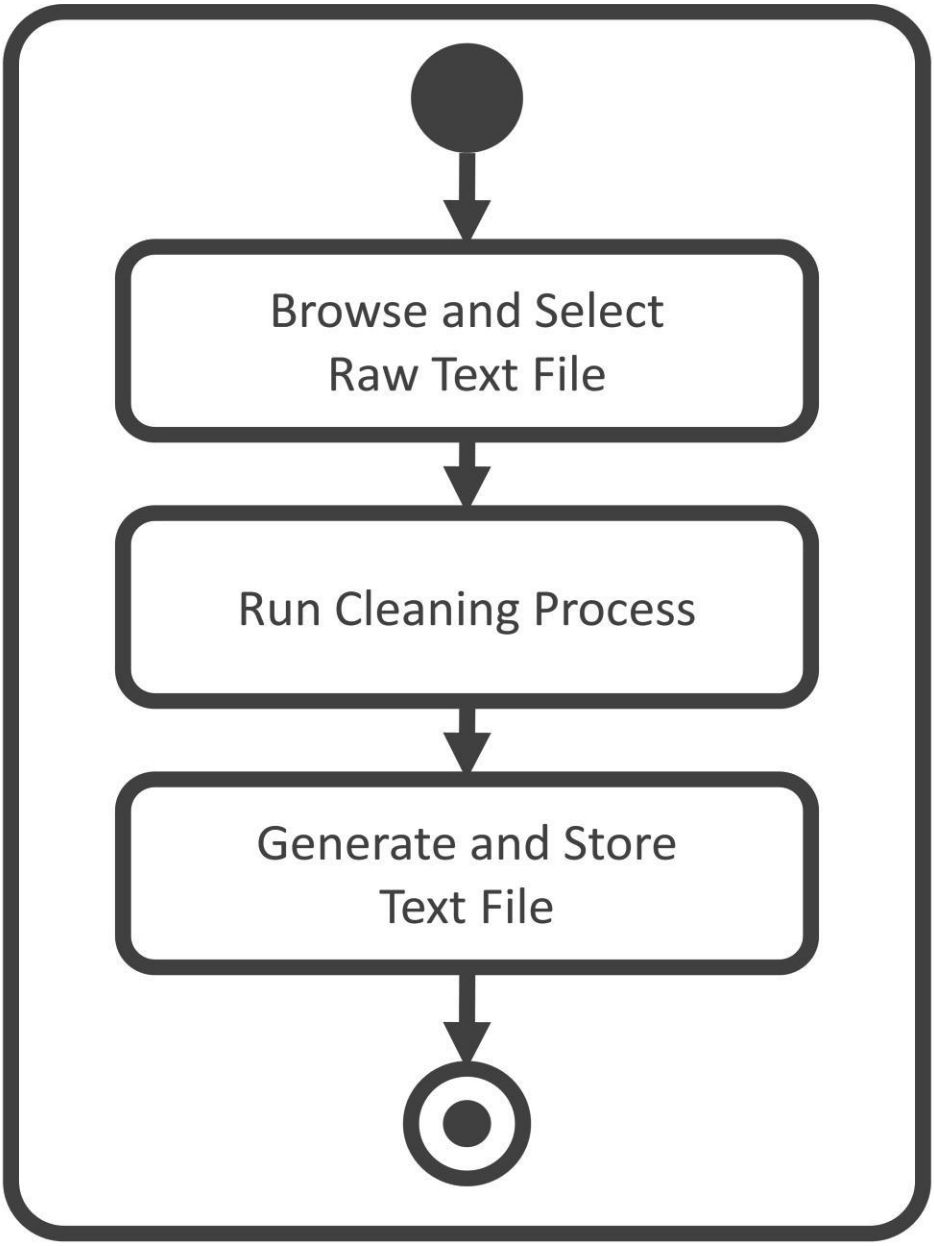



Figure 4.3.1.4.2a : Activity Diagram- Clean and Generate Output

Figure 4.3.1.4.2b is the Swim-Lane Diagram of Clean and Generate Output-

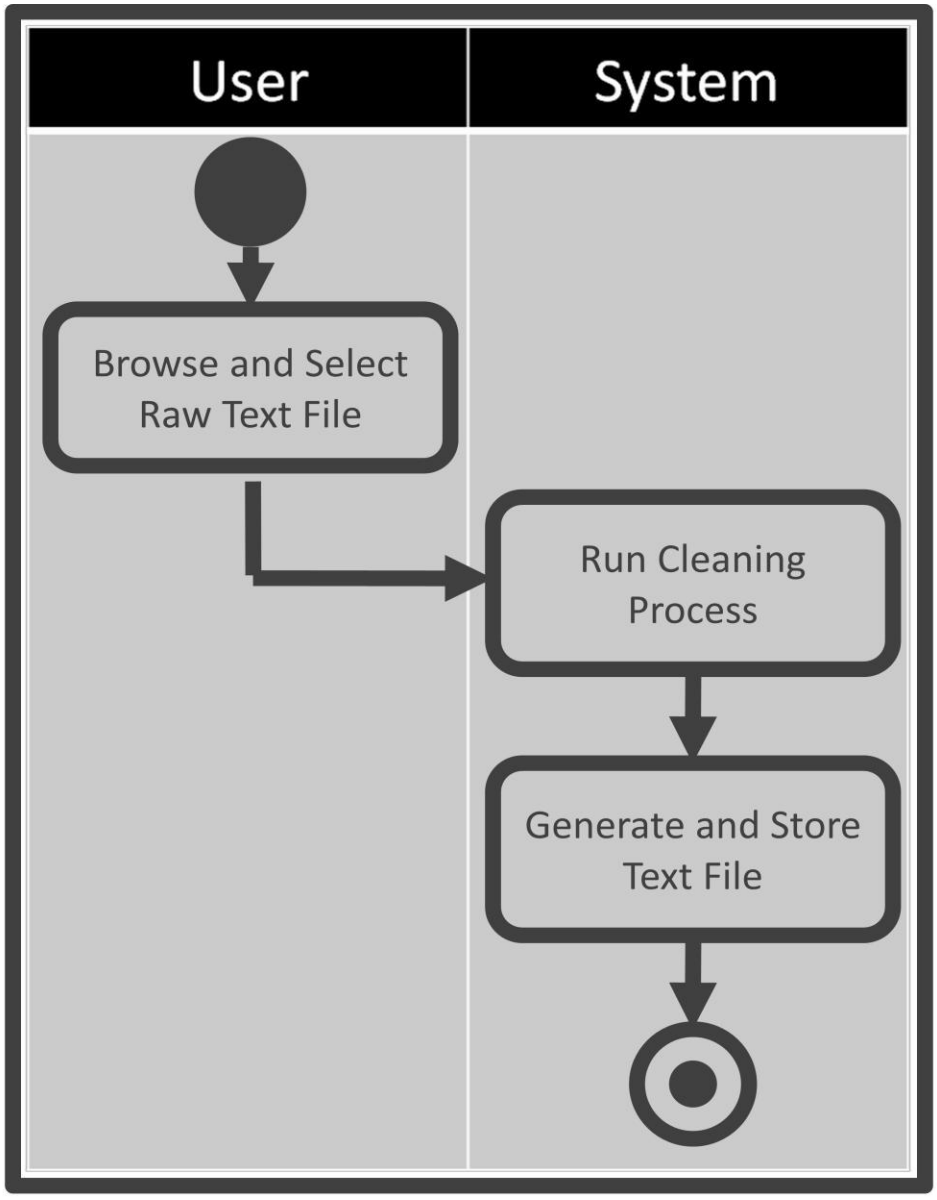



Figure 4.3.1.4.2b: Swim-Lane Diagram- Clean and Generate Output

# Chapter 5
# Data Model

## 5.1 Introduction

When software requires interfacing with a database or complex data structures need to be constructed and manipulated, data model is performed as part of overall requirements modeling.

## 5.2 Data Object Selection

Data objects are representation of information which has different attributes.Table-5.2 enlists all nouns from Usage Scenario and marks potential data objects:

Table-5.2 Data Object Selection

| Noun | Attributes | Description | Remark |
|---|---|---|---|
| bengali braille character recognizer | | represents the whole system | rejected |
| image | | alias of braille writing | rejected |
| braille writing | | input for the system | rejected |
| image-preprocessing technique | | alias of algorithm | rejected |
| text | | alias of text file | rejected |
| pattern recognition | | alias of algorithm | rejected |
| final output | | output for the system | rejected |
| braille | | alias of braille writing | rejected |
| system | | alias of bengali braille | rejected |

| | | | |
|---|---|---|---|
| | | character recognizer | |
| visually impaired people | | out of scope | rejected |
| raised dots | | out of scope | rejected |
| embossed paper | | out of scope | rejected |
| tactile alphabet | | out of scope | rejected |
| paper | | alias of embossed paper | rejected |
| scanner | | a device, external entity | rejected |
| image files | | alias of braille writing | rejected |
| computer | | represents the whole system | rejected |
| scanned images | | alias of braille writing | rejected |
| image enhancement process | | alias of algorithm | rejected |
| section | | out of scope | rejected |
| braille pattern | | alias of braille writing | rejected |
| background | | out of scope | rejected |
| noise reduction algorithm | | alias of algorithm | rejected |
| artifact | | out of scope | rejected |
| pattern quality | | out of scope | rejected |
| braille dots | | alias of braille writing | rejected |
| connectivity improvement | | alias of algorithm | rejected |
| quantization | | alias of algorithm | rejected |
| code | | an attribute of braille pattern code | rejected |
| **braille pattern code** | id, code | potential data object | accepted |
| braille character | | alias of braille writing | rejected |
| **code to character map** | code, character | potential data object | accepted |
| hash table | | alias of braille code to character map | rejected |
| text file | | alias of final output | rejected |
| braille image | | alias of braille writing | rejected |

| | | | |
|---|---|---|---|
| stage | | alias of algorithm | rejected |
| spell checking | | alias of algorithm | rejected |
| error rate | | out of scope | rejected |
| **domain word list** | word, frequency | potential data object | accepted |
| cleaning procedure | | alias of algorithm | rejected |
| step | | alias of algorithm | rejected |
| setting | | alias of braille processing template | rejected |
| algorithm | | an attribute of braille processing template | rejected |
| parameter | | an attribute of braille processing template | rejected |
| configuration file | | alias of braille processing template | rejected |
| **braille processing template** | algorithm, parameters | potential data object | accepted |
| template | | alias of braille processing template | rejected |
| user | | no separated authentication system required | rejected |
| system | | alias of braille writing | rejected |
| customized template | | alias of bengali braille character recognizer | rejected |

## 5.3 Data Objects and Attributes

This is a brief view of all attributes we have found so far:

**Braille Pattern Code-** Id + Code

**Code to Character Map-** Code + Character

**Domain Word List-** Word + Frequency

**Braille Processing Template-** Algorithm + Parameters

## 5.4 Conclusion

Data objects found here have no relationship to each other. These data objects will be stored in a file system and will work independently. So there is no Data Object Relational Diagram here. For this same reason no E-R Diagram or Schema Diagram exists for this project.

# Chapter 6
## Class-Based Model

### 6.1 Introduction

The objects that the system will manipulate, the operations that will be applied to the objects and relationships between the objects are represented by Class-Based Modeling. It also defines the collaborations that occur between the classes.

### 6.2 General Classification

Analysis classes can be marked by one of the following ways:

1. External Entity
2. Thing
3. Occurrence
4. Role
5. Organizational Unit
6. Place
7. Structure

Table-6.2 General Classification

| No. | Noun | General Classification | Remark |
|---|---|---|---|
| 1 | bengali braille character recognizer | 2 | Problem Space (represents whole system) |
| 2 | image | 2, 7 | Problem Space |
| 3 | braille writing | NULL | Problem Space |
| 4 | image-preprocessing technique | 3, 7 | Solution Space (same as- 25) |

| | | | |
|---|---|---|---|
| 5 | text | 2, 7 | Problem Space |
| 6 | pattern recognition | 3 | Solution Space |
| 7 | final output | 7 | Problem Space |
| 8 | braille | Null | Problem Space |
| 9 | system | 2 | Problem Space (represents whole system) |
| 10 | visually impaired people | 1 | Problem Space |
| 11 | raised dots | 1 | Problem Space |
| 12 | embossed paper | 1 | Problem Space |
| 13 | tactile alphabet | 1 | Problem Space |
| 14 | paper | 1 | Problem Space |
| 15 | scanner | 1, 2 | Problem Space |
| 16 | image files | 1, 2 | Problem Space |
| 17 | computer | 1, 2, 7 | Problem Space (represents whole system) |
| 18 | scanned images | 1, 2 | Problem Space |
| 19 | image enhancement process | 3 | Solution Space |
| 20 | section | NULL | Problem Space |
| 21 | braille pattern | 1, 2 | Problem Space |
| 22 | background | NULL | Problem Space |
| 23 | noise reduction algorithm | 3 | Solution Space |
| 24 | artifact | NULL | Problem Space |
| 25 | pre-processing | 3 | Solution Space |
| 26 | braille dots | 1, 2 | Problem Space |
| 27 | connectivity improvement | 3 | Solution Space |
| 28 | quantization | 3 | Solution Space |
| 29 | code | NULL | Problem Space |
| 30 | translation | 3 | Solution Space |
| 31 | braille character | NULL | Problem Space |
| 32 | code to character map | 4, 7 | Solution Space |
| 33 | hash table | 4, 7 | Solution Space (same as- 32) |
| 34 | text file | 1, 2 | Problem Space |

| | | | |
|---|---|---|---|
| 35 | braille image | 1, 2 | Problem Space |
| 36 | post-processing | 3 | Solution Space |
| 37 | spell checking | 3 | Solution Space |
| 38 | error rate | NULL | Problem Space |
| 39 | domain word list | 4, 7 | Solution Space |
| 40 | cleaning procedure | 3 | Solution Space |
| 41 | step | NULL | Problem Space |
| 43 | setting | 4, 7 | Solution Space (same as- 47) |
| 44 | algorithm | 3 | Solution Space (represents all procedures) |
| 45 | parameter | 7 | Solution Space (represents all parameters of the system) |
| 46 | configuration file | 4, 7 | Solution Space (same as- 47) |
| 47 | braille processing template | 4, 7 | Solution Space |
| 48 | template | 4, 7 | Solution Space (same as- 47) |
| 49 | user | 1, 4 | Solution Space |
| 50 | system | 2 | Problem Space (represents whole system) |
| 51 | customized template | 4, 7 | Solution Space (same as- 47) |

## 6.3 Selection Characteristics

Coad and Yourdon suggest six selection characteristics that should be used to consider each potential class for inclusion in the analysis model:

1. Retained Information
2. Needed Services
3. Multiple Attributes
4. Common Attributes
5. Common operations
6. Essential Requirements

| Table-6.3 Selection Characteristics | | | |
|---|---|---|---|
| **No.** | **Potential Class** | **Characteristics** | **Remarks** |
| 1 | pattern recognition | 1, 2, 3, 6 | No |
| 2 | image enhancement process | 1, 2, 3 | No |
| 3 | noise reduction algorithm | 1, 2, 3 | No |
| 4 | pre-processing | 1, 2, 3, 4, 5, 6 | Yes |
| 5 | connectivity improvement | 1, 2, 3 | No |
| 6 | quantization | 1, 2, 3 | No |
| 7 | translation | 1, 2, 3, 4, 5, 6 | Yes |
| 8 | code to character map | 1, 2, 6 | No |
| 9 | post-processing | 1, 2, 3, 4, 5, 6 | Yes |
| 10 | spell checking | 1, 2, 3 | No |
| 11 | domain word list | 1, 2 | No |
| 12 | cleaning procedure | 1, 2, 3 | No |
| 13 | braille processing template | 1, 2 | No |
| 14 | user | 1, 2, 3, 4, 5, 6 | Yes |

## 6.4 Attribute Selection

Here we find attributes for selected classes.

| Table-6.4 Attribute Selection | | |
|---|---|---|
| **No.** | **Class** | **Attributes** |
| 1 | User | selected_algorithm_List, initialized_parameters |
| 2 | Pre-Processing | image_input_file_list, output_image_path_list |
| 3 | Translation | image_list, code_data, text_data, |

| | | |
|---|---|---|
| | | code_to_character_map |
| 4 | Post-Processing | Input_text_data, output_file_path |

## 6.5 Defining Methods

In this section we find all the verbs from usage scenario and include necessary external verbs in a list. Then we select useful verbs as methods.

### 6.5.1 Verb List

Here we list all verbs from usage scenario.

Table-6.5.1 Verb List

| No. | Verb | Remarks |
|---|---|---|
| 1 | take scanned image | out of scope |
| 2 | run image pre-processing | yes |
| 3 | translate braille to text | yes |
| 4 | apply text correction | yes (same as- run post-processing) |
| 5 | scan paper | out of scope |
| 6 | store image | out of scope |
| 7 | select image | out of scope |
| 8 | run image enhancement | yes |
| 9 | improve braille pattern | yes (same as- run image enhancement) |
| 10 | make distinguishable | yes (same as- run image enhancement) |
| 11 | apply noise reduction | yes |
| 12 | improve connectivity | yes |
| 13 | run quantization | yes |

| | | |
|---|---|---|
| 14 | extract braille pattern | yes |
| 15 | convert braille pattern to code | yes |
| 16 | map code to character | yes (same as- convert braille pattern to code) |
| 17 | generate plain text from code | yes |
| 18 | contain character | out of scope |
| 19 | run post processing | yes |
| 20 | run spell checking | yes |
| 21 | store word | out of scope |
| 22 | run cleaning procedure | yes |
| 23 | automate whole process | no need |
| 24 | customize braille processing template | yes |

### 6.5.2 Selected Methods

From the verb list above we have selected following methods for classes.

Table-6.5.2 Selected Methods

| Class | Methods | Remarks |
|---|---|---|
| User | pre_process()<br>translate()<br>post_process()<br>customize_braille_processing<br>_template() | pre_process() is run image pre-processing, translate() is translate braille to text, post_process() is run post-processing |
| Pre-Processing | enhance_image()<br>reduce_noise()<br>improve_connectivity()<br>run_quantization() | enhance_image() is run image enhancement, reduce_noise() is apply noise reduction |
| Translation | apply_segmentation()<br>recognize_braille_pattern() | apply_segmentation() is extract braille |

| | | |
|---|---|---|
| | create_plain_text() | pattern, recognize_braille_pattern() is convert braille pattern to code, create_plain_text() is generate plain text from code |
| Post-Processing | check_spell()<br>generate_final_output() | generate_final_output() is run cleaning procedure |

## 6.6 Class Diagram

We have shown here (Figure- 6.6), how the classes interact together to accomplish certain goal.

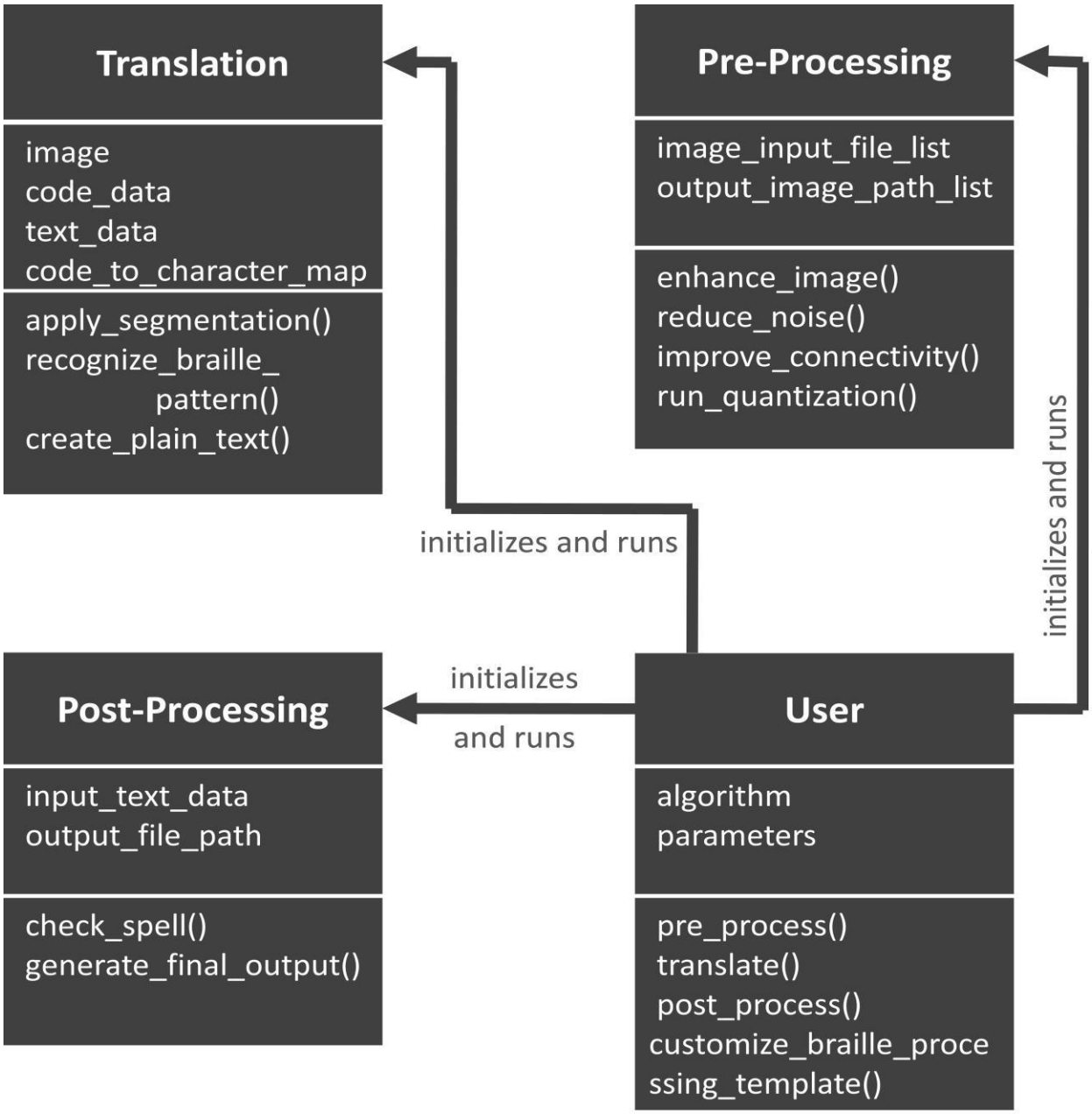



Figure 6.6: Class Diagram (BBTT)

## 6.7 Class Card

Class card represents a graphical view of responsibility and collaborator for each class.

| User | |
|---|---|
| Responsibility | Collaborator |
| • Initialize System<br>• Set Parameters<br><br>• Run Process | • Null<br>• Pre-Processing, Translation, Post-Processing<br>• Pre-Processing, Translation, Post-Processing |

| Pre-Processing | |
|---|---|
| Responsibility | Collaborator |
| • Enhance Input Image Quality | • Null |

| Translation | |
|---|---|
| Responsibility | Collaborator |
| • Convert Pattern to Code<br>• Convert Code to Text | • Null<br><br>• Null |

| Post-Processing | |
|---|---|
| Responsibility | Collaborator |
| • Check Spell<br>• Clean Output | • Null<br>• Null |

# Chapter 7
## Behavioral Model

### 7.1 Introduction

When the system is perceived in terms of states and transitions, it is called Behavior Modeling. It is also known as State Modeling, State Machines, or State Transition Matrix.

This requires both identifying all of the interesting states of being that software or its components are likely to be in. And also, at a high level, abstracting what events are likely to cause software or its components to change between states of being.

### 7.2 Identifying Events

Here we have identified events from the **Usage Scenario** and listed their corresponding initiators & collaborators.

**Table-7.2 Identifying Events**

| Event | Initiator | Collaborator |
|---|---|---|
| take scanned image of Braille writing | User | Scanner |
| run different types of image-preprocessing techniques | User | |
| translate Braille to text through pattern recognition | User | Braille Pattern Code, Code to Character Map |
| apply text correction procedures | User | Domain Word List |

| | | |
|---|---|---|
| select scanned images of Braille writing | User | |
| run image enhancement | User | |
| apply noise reduction | User | |
| improve connectivity | User | |
| apply quantization | User | |
| extract braille pattern | User | |
| convert braille to code | User | Braille Pattern Code |
| map code to character | User | Braille Pattern Code, Code to Character Map |
| check spell | User | Domain Word List |
| run cleaning procedure | User | |
| automate the configuration | User | Braille Processing Template |
| customize template | User | Braille Processing Template |

## 7.3 State Transition Diagram

State Transition Diagram represents active states for each class and the events (triggers) that cause changes between these active states. Here Figure-7.3 provides diagram for the actor- User.

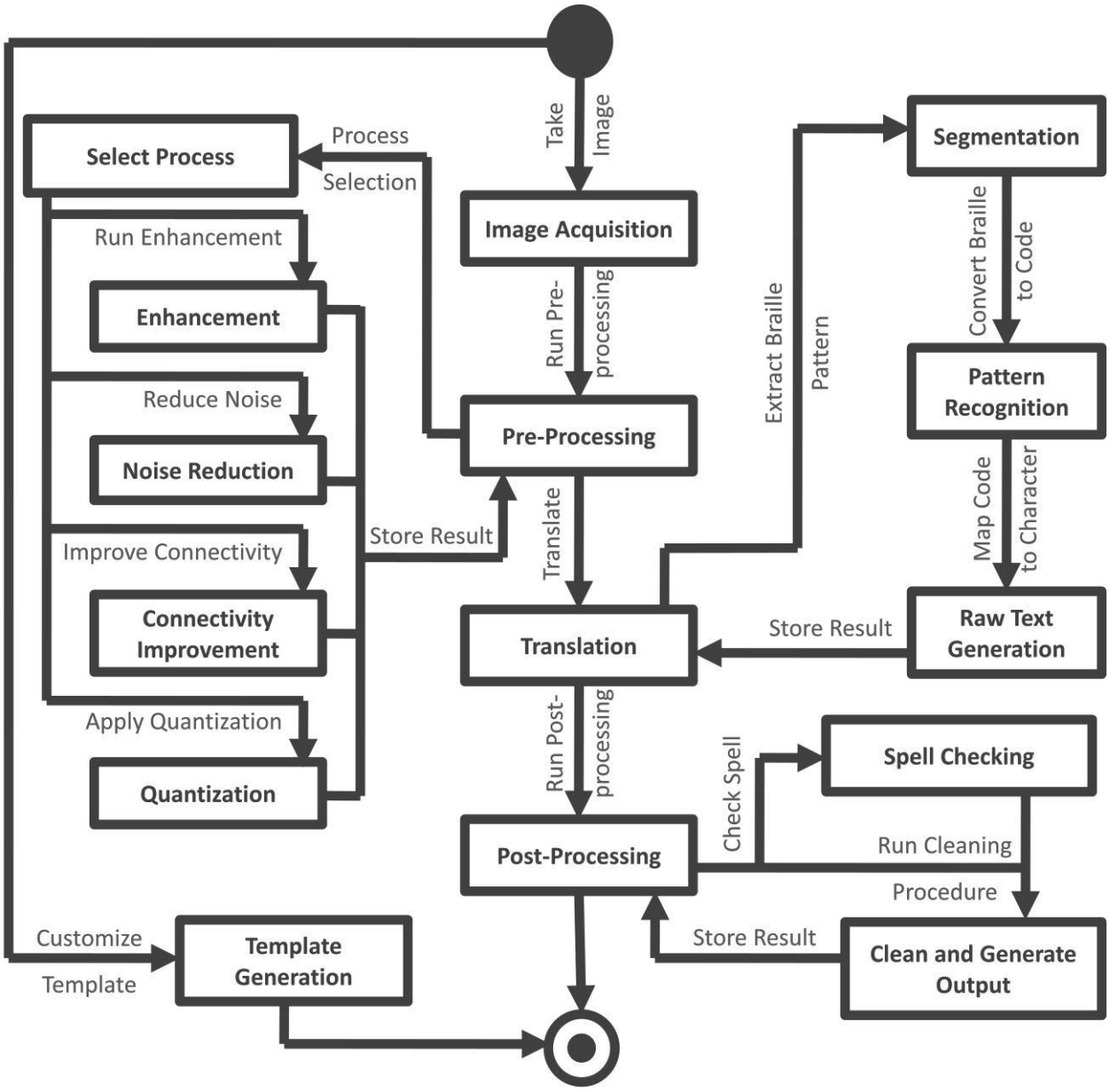



Figure 7.3: State Transition Diagram- User

## 7.4 Sequence Diagram

The way events cause transitions from object to object as a function of time is represented by Sequence Diagram. Figure-7.4.1 to Figure-7.4.11 represents Sequence Diagram for all major events of this project.

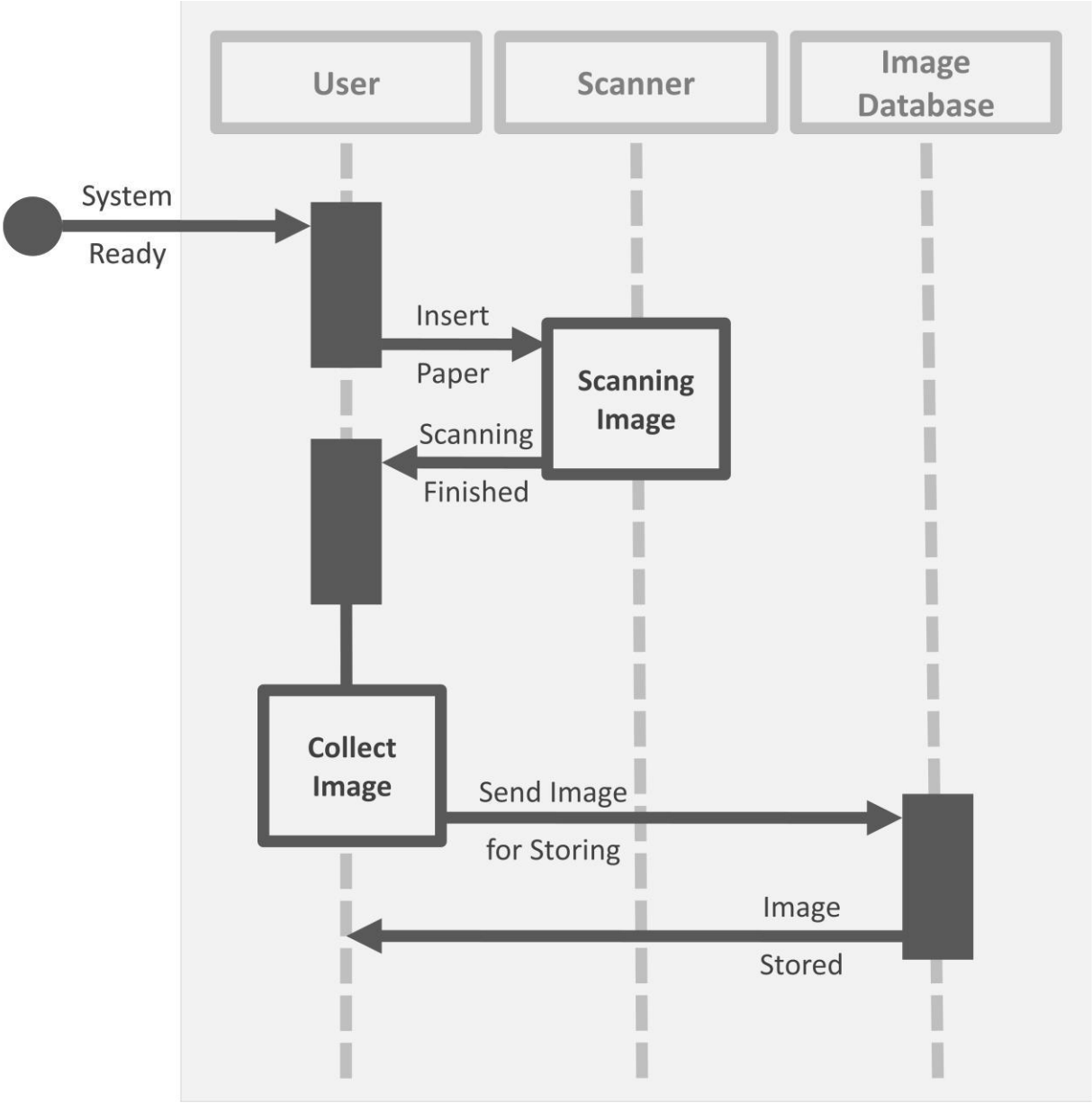



Figure 7.4.1: Sequence Diagram- Take Image

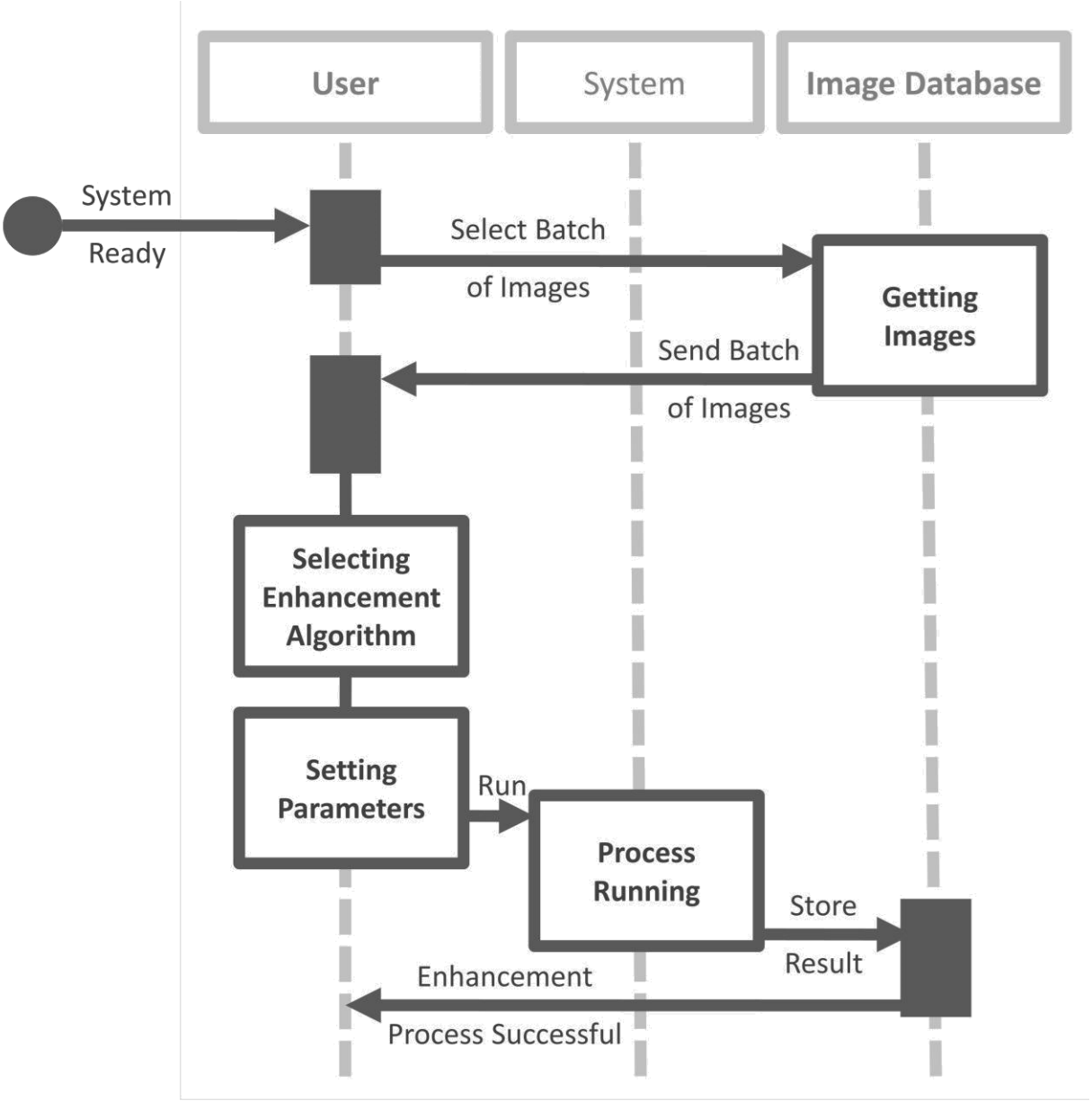



Figure 7.4.2: Sequence Diagram- Run Enhancement

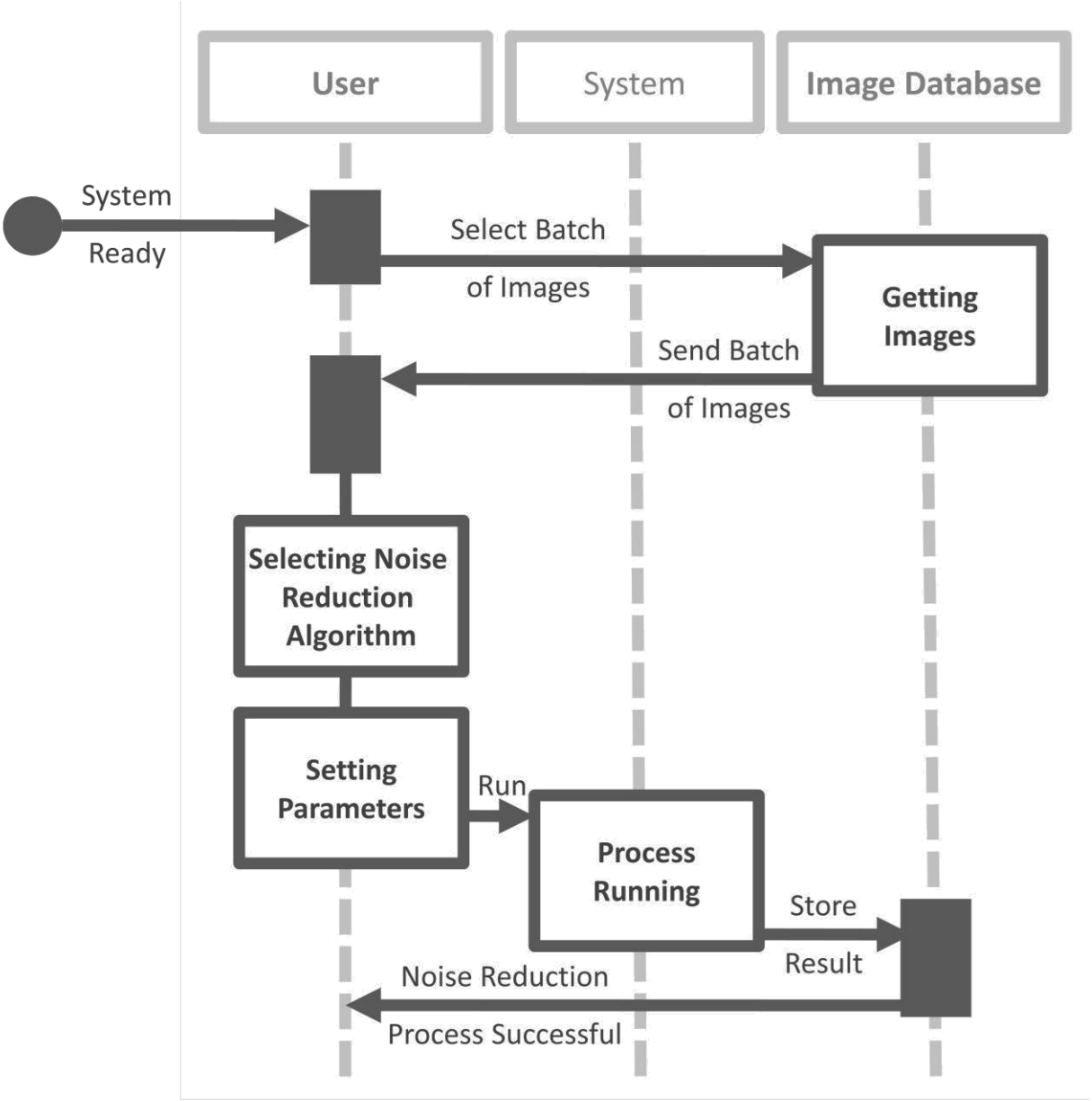



Figure 7.4.3: Sequence Diagram- Reduce Noise

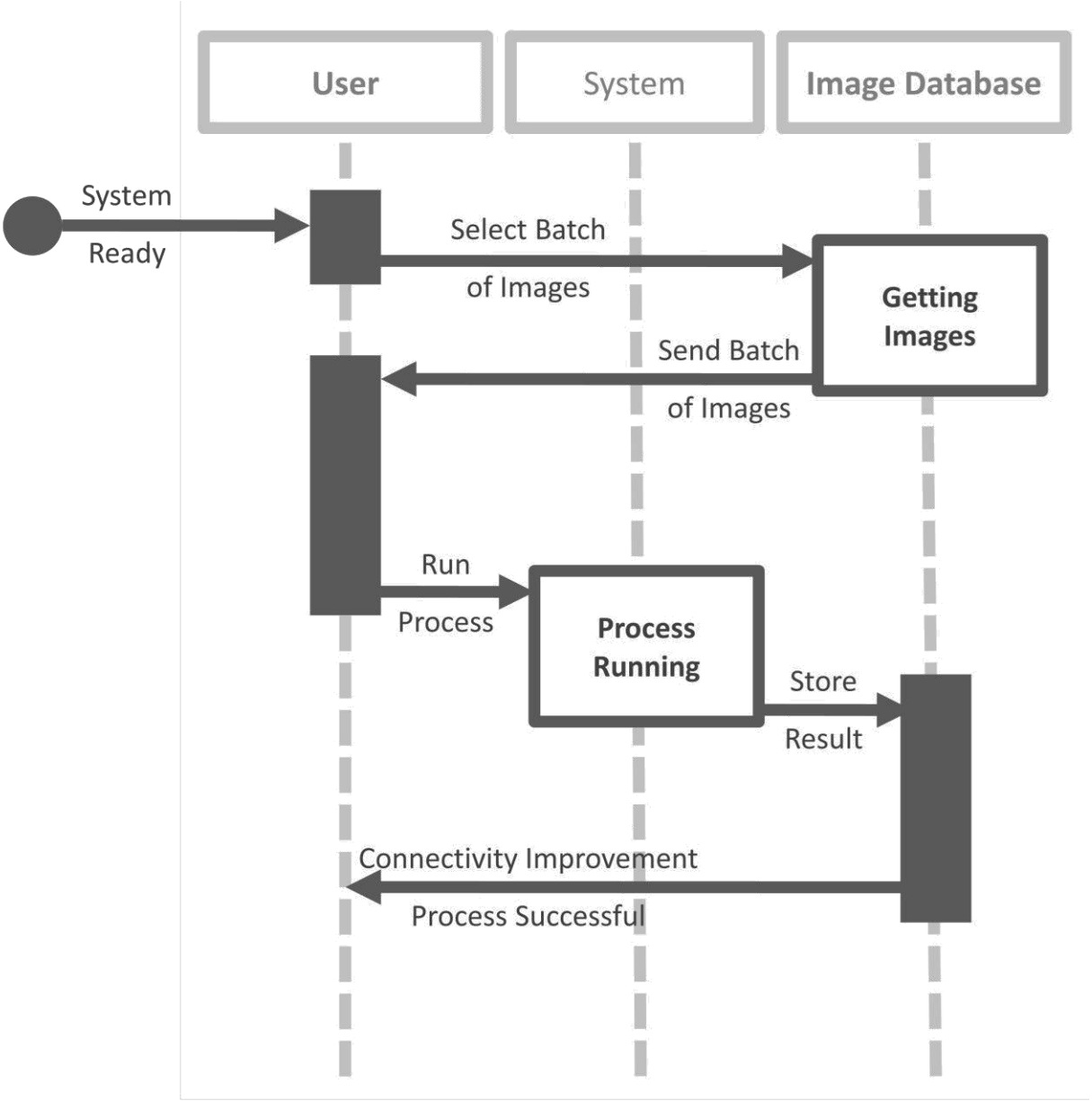



Figure 7.4.4: Sequence Diagram- Improve Connectivity

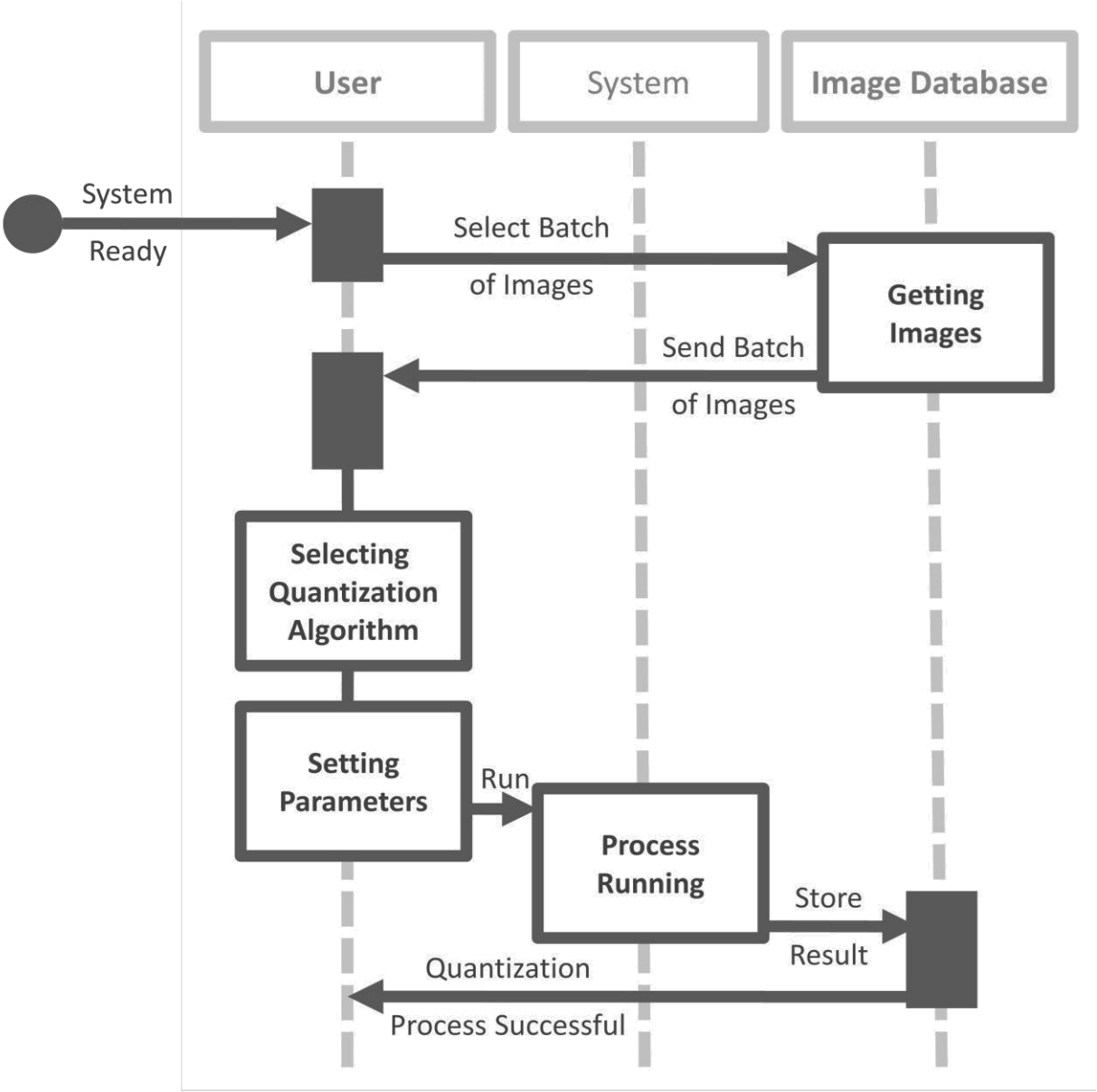



Figure 7.4.5: Sequence Diagram- Apply Quantization

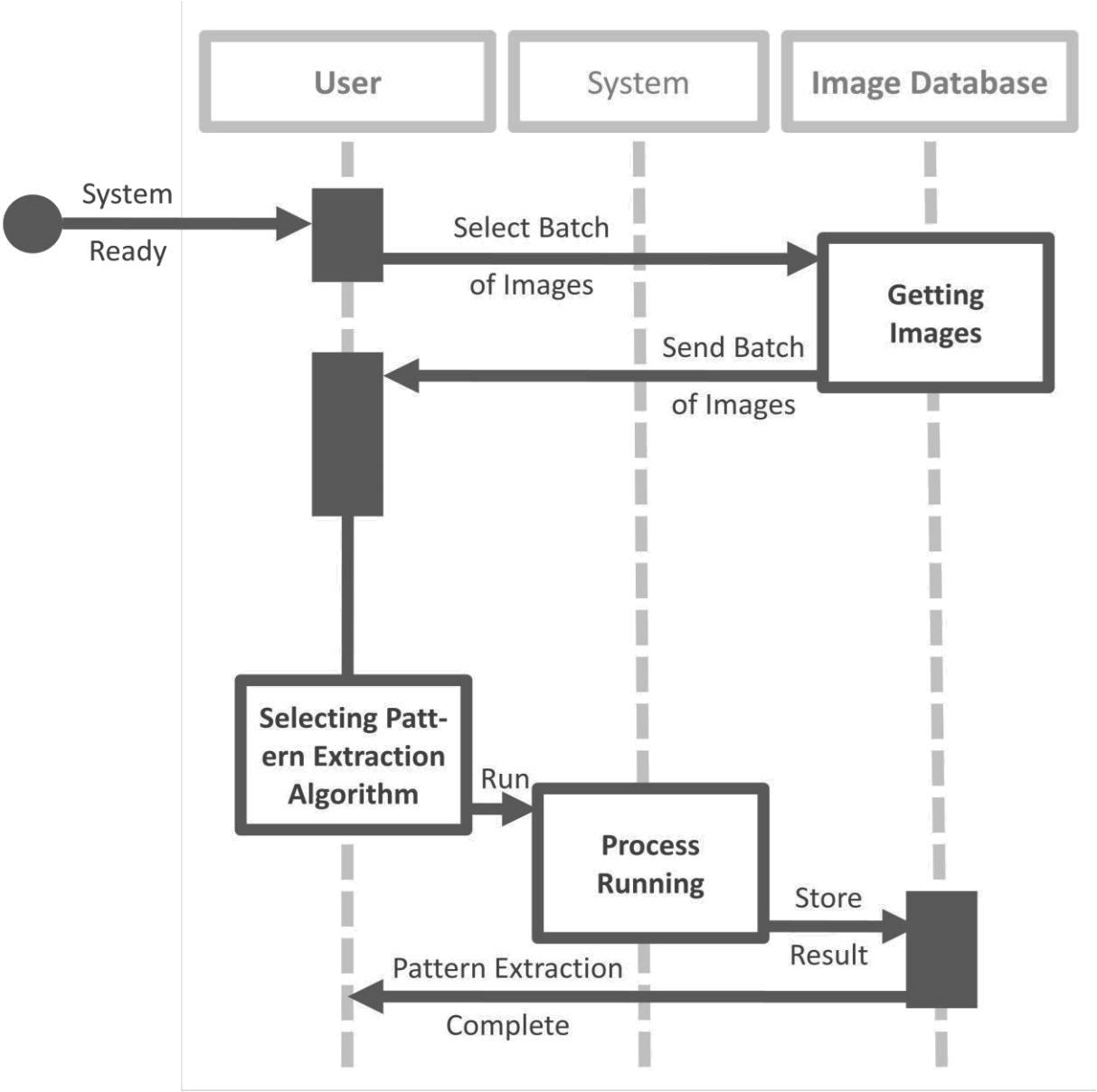



Figure 7.4.6: Sequence Diagram- Extract Braille Pattern

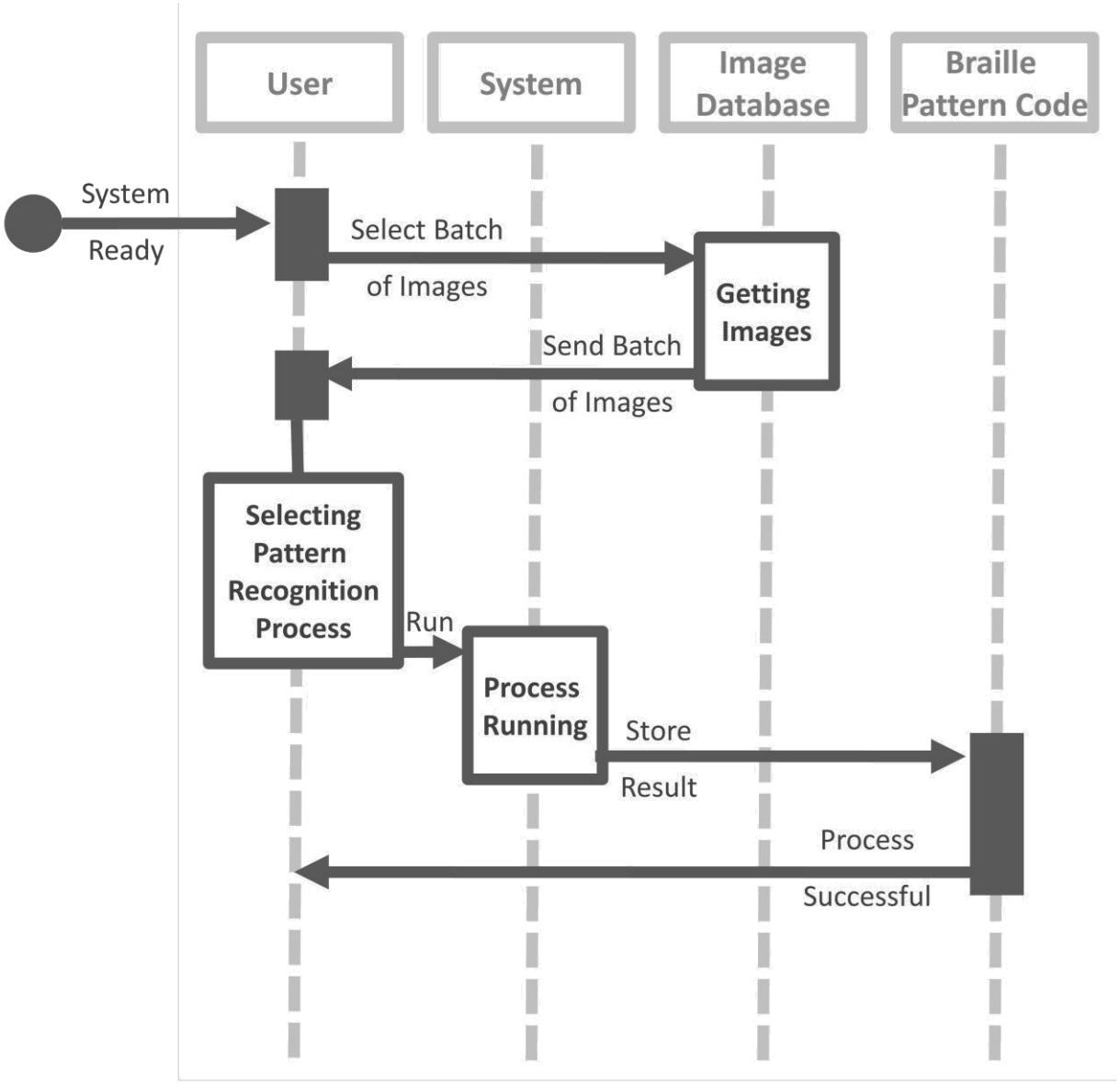



Figure 7.4.7: Sequence Diagram- Convert Braille to Code

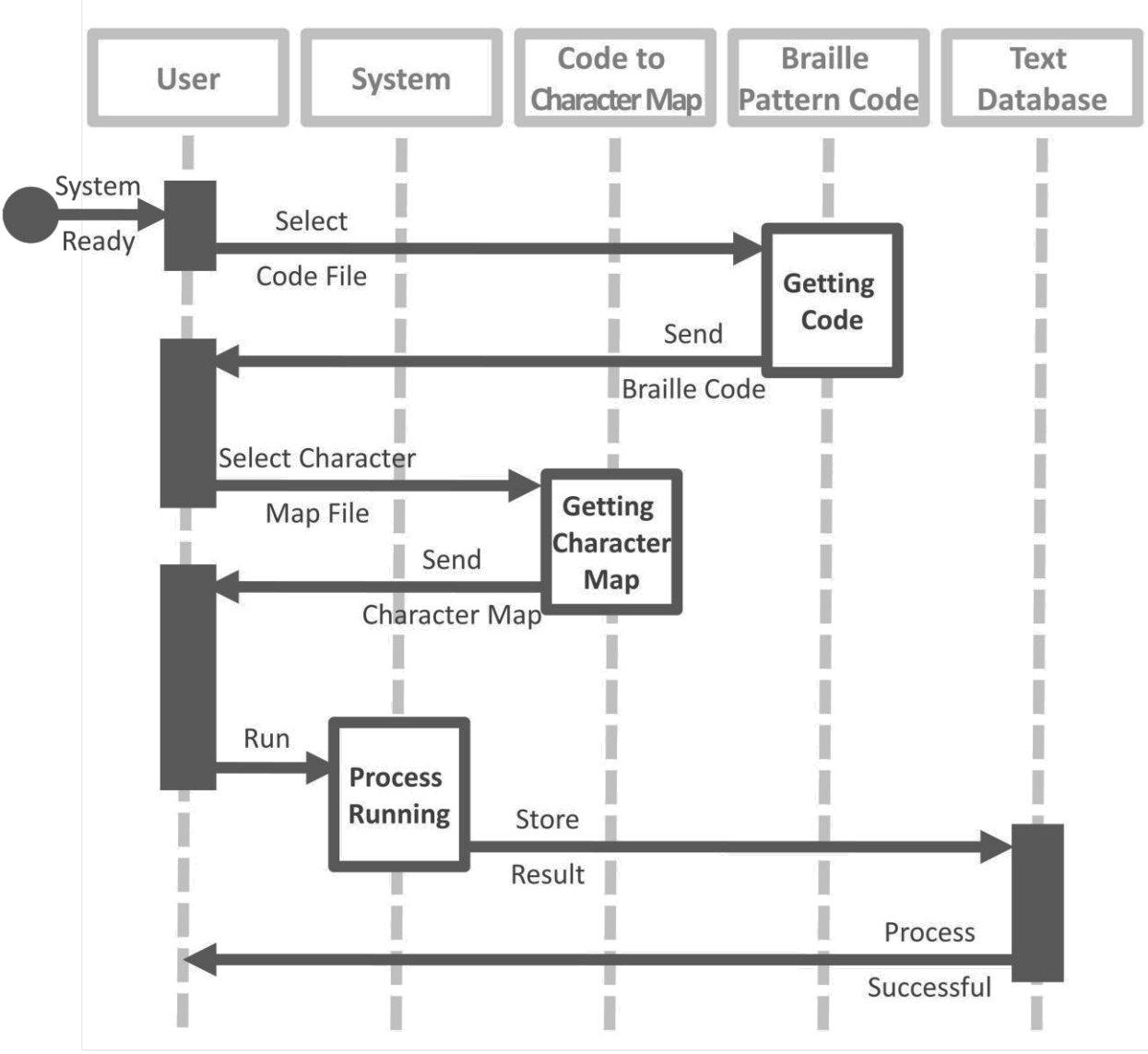



Figure 7.4.8: Sequence Diagram- Map Code to Character

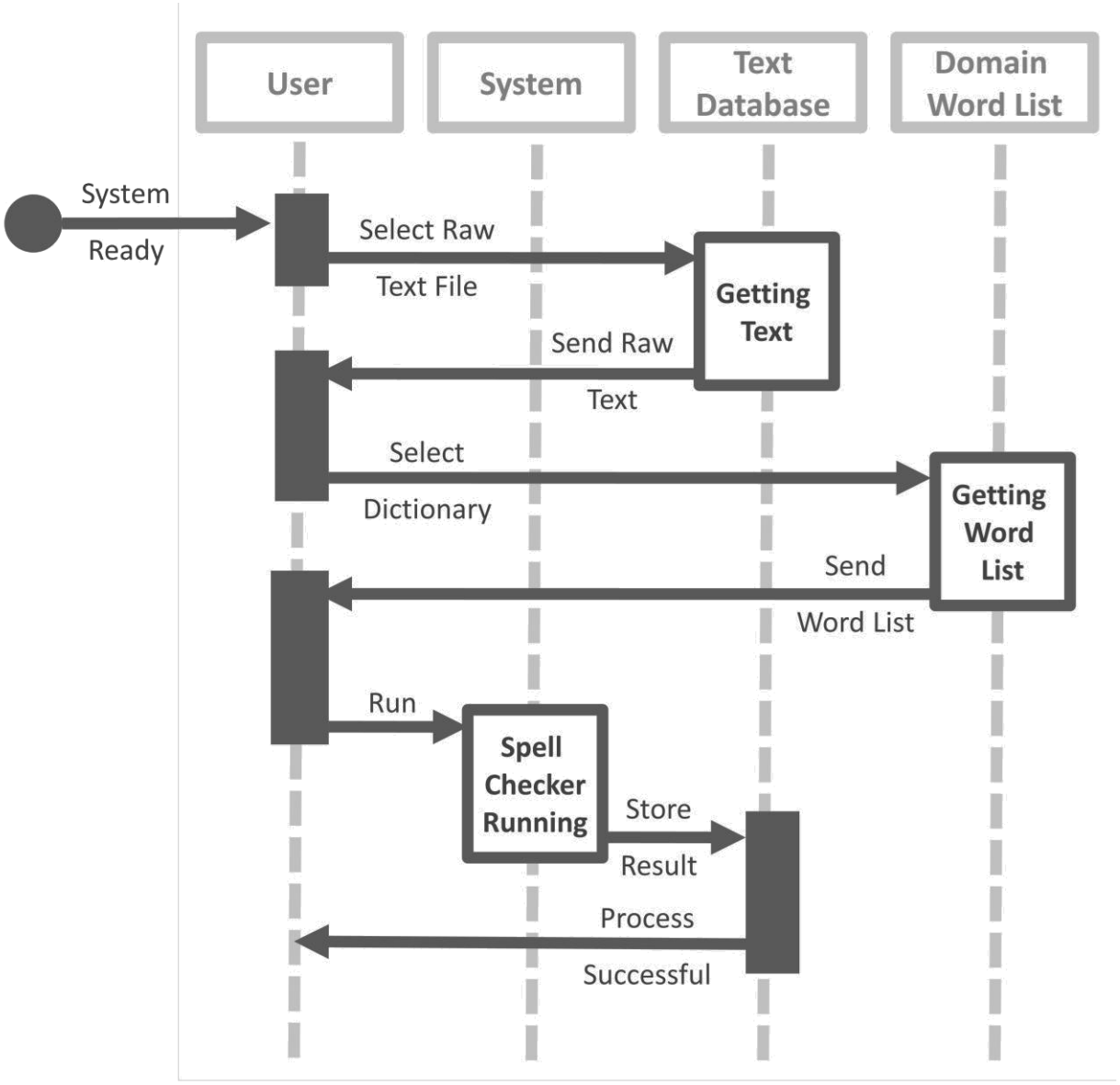



Figure 7.4.9: Sequence Diagram- Check Spell

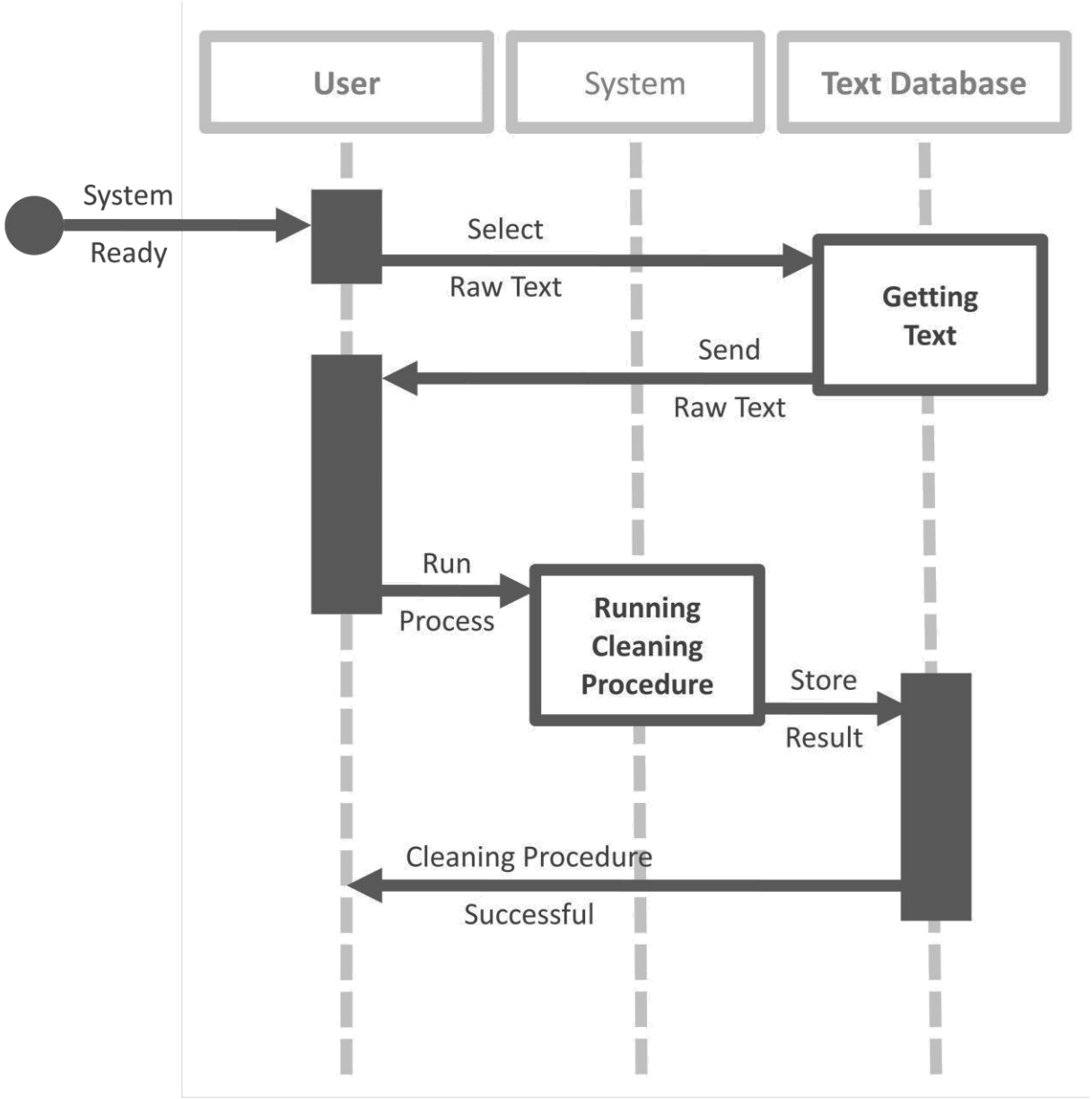



Figure 7.4.10: Sequence Diagram- Run Cleaning Procedure

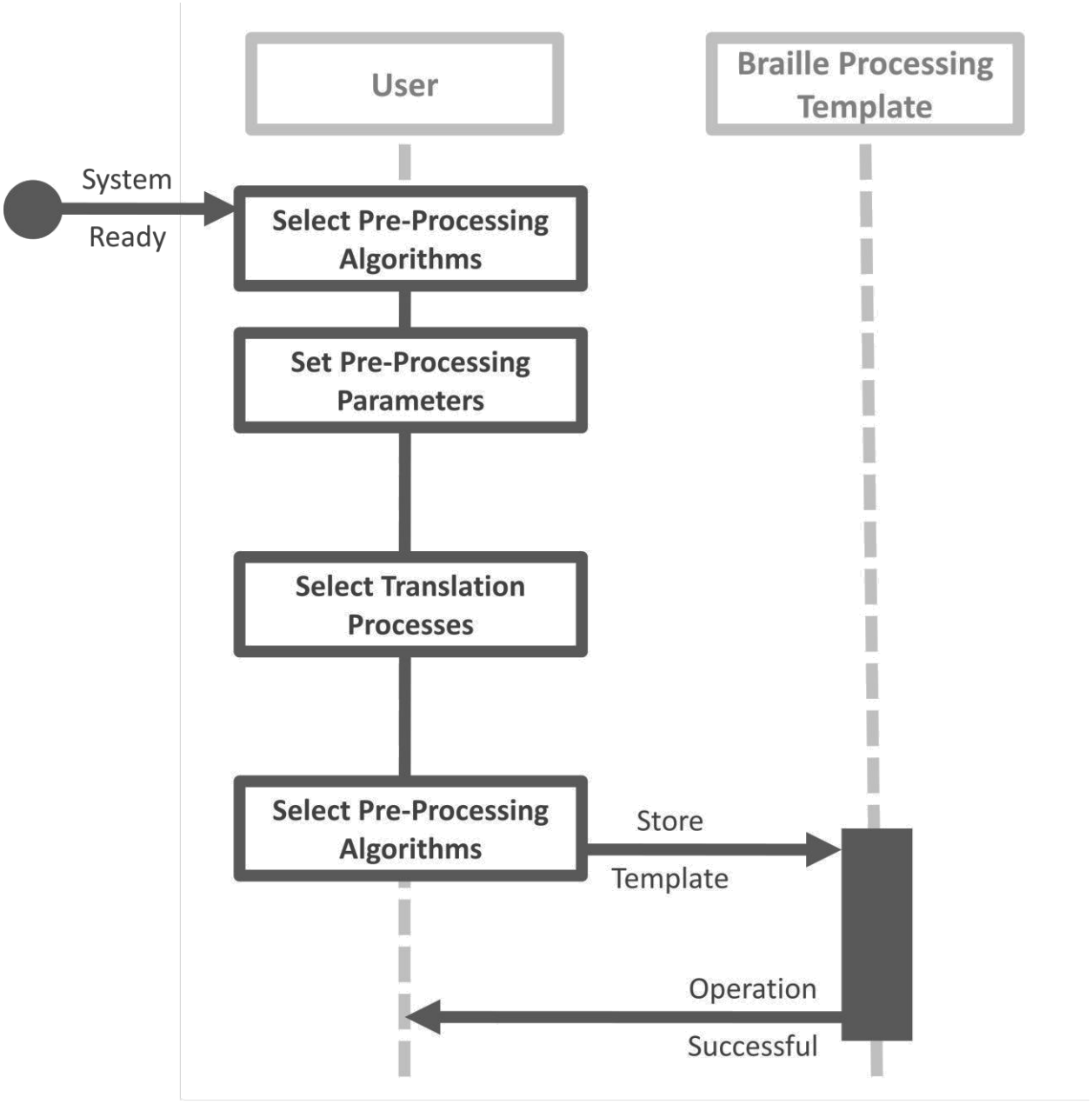



Figure 7.4.11: Sequence Diagram- Customize Template

# Chapter 8
## Conclusion

I am pleased to submit the final SRS report on Bengali Braille to Text Translator. From this, the readers will get a clear and easy view of tactile writing system. He will also get a good understanding of the translation process.

This SRS document can be used effectively to maintain the software development cycle for the project. I have presented a detailed description of the total system. It will be much easy to conduct the whole project using this SRS. It will also help me to determine the pitfalls that may come ahead. Hopefully, this document can also help other software engineering students as well as practitioners.

I have tried my best to make effective and fully designed SRS. I believe that the readers will find it in order.

# Appendix

## References

[1] Pressman, Roger S. Software Engineering: A Practitioner's Approach (7th ed.). Boston, Mass: McGraw-Hill. ISBN 0-07-285318-2.

[2] Ralph, Paul. "The illusion of requirements in software development." Requirements Engineering 18.3 (2013): 293-296.

[3] W. David, A. Adler. “A Picture of Louis Braille.” New York, McGraw Hill, 1999.

[4] Durre, K.P., W. Tuttle. “A Universal Computer Braille Code for Literacy and Scientific Texts.”International Technology Conference, 1991.