Functional Specification

Automatic Transliteration between English and Russian

Student: Kieron Drumm (13314446)

Supervisor: Prof. Qun Liu

Date: November 24th 2016

Table of Contents

1. Introduction

1.1 Overview

The main purpose of this system will be to perform automatic transliteration between English and Russian and vice-versa. As a result it will be converting between two different writing systems, namely the Latin alphabet and the Cyrillic alphabet. Also, as this application will be a Windows application it should easily be runnable on any up-to-date Windows machine.

The main function of this application will be to graphically display the transliteration of a given English or Russian word to the user. The user need only input a word in English or Russian and its transliterated form will be shown below.

The intended graphical user interface will be simple and similar to most standard Windows applications to make it integrate with Windows more easily. This will be done with the intention of making the application more usable for users who are only familiar with the Windows operating system.

1.2 Business Context

In the context of business, this application could potentially be used as a teaching tool in classrooms. The target users in such a case would be beginner students that are learning either English or Russian. This application could be used as a tool to teach the student’s a new writing system, be it Latin or Cyrillic. The application could also be used by a language teacher to introduce an entire classroom of students to a new writing system.

Another quite specific use of this application could be to act as a helpful tool when creating a language course, be it in an educational or commercial setting. For example, most language courses, both online and written are normally split into sections that cover specific parts of everyday life. Throughout these sections, whenever some new vocabulary is introduced to the reader/student, there is normally a phonetic spelling included. This application could speed up the process of writing such a course by allowing the writer to automatically generate phonetic spellings for a large list of words.

1.3 Glossary

This section contains descriptions for some words whose meaning may not be common knowledge:

- Transliteration – The process of converting a word from one writing system to another writing system while ensuring that the pronunciation of that word is maintained.
- Cyrillic – The phonemic alphabet used to write words in a large number of Slavic and Non-Slavic languages such as Russian, Kazakh, Mongolian and Ukrainian.

2. General Description

2.1 Product/System Functions

The intended programming languages for this application will be as follows:

- C# - For the graphical user interface, this language has been chosen as it is the ideal language for developing the user interface for a Windows application. It will also be used to execute any scripts in other languages in order to carry out some back-end processing.
- Python – For the data scraping, back end processing and general machine learning. Python has been chosen for its useful libraries in relation to natural language processing and data scraping. Namely the natural language tool kit (nltk) for natural language processing, and the Scrapy framework for data scraping.

The first step in developing this application will be to acquire large amounts of parallel data with which to train the application. Once said training has been complete, the code for the back-end can be developed. The main back-end function will be to make calculated decisions, based on the parallel data provided, as to how certain transliterations should be carried out.

The main function of the user interface will be to read and store a user's input in either English or Russian, to process said input behind the scenes and to then output the transliterated input to the user in a clear and concise manner. The interface will not be overly cluttered in order to make the transliteration process as easy as possible.

2.2 User Characteristics and Objectives

The target users for this application are those involved in the language learning process, be they teachers or students. The expected expertise of the target user's will be a basic understanding of how to use applications in the Windows operating system. The process of inputting a word and transliterating said word will only involve some typing and button clicking for the user. This will be done in order to make the process as simple as possible for the user.

From the user's perspective, the objectives of the application will be simple. The user should expect to be able to open the application on a Windows computer and to input a word in either English or Russian into the text area provided. They should then be able to click a button signalling the start of the transliteration process. And lastly they should clearly see the transliterated form of the word they typed in displayed on the screen.

2.3 Operational Scenarios

a. Standard Transliteration Between English and Russian

The user opens the Windows application on their machine. A user interface is displayed with a text area in which the user can input a word in English. The user inputs a word or words in English with the use of the Latin alphabet. They then click a button with the label “transliterate” on it. A small message is displayed to them to signify that some back-end processing is being carried out. Once the processing has finished and a decision has been made as to what transliteration should be carried out, the result is displayed to the user in Russian using the Cyrillic alphabet.

b. Standard Transliteration Between Russian and English

This scenario would be extremely similar to the scenario in which an English word is translated into Russian. The key differences would be as follows:

- The keyboard language would need to be changed to Cyrillic.
- The inputs would be reversed i.e. the input would be in Cyrillic and the output would be in the Latin alphabet.

2.4 Constraints

a. Time

As this project is due near the end of May there is a clear time constraint involved in the development of this application. In order to manage this constraint I intend to construct a Gantt chart. I believe that this will keep me on track as it will allow me to manage my time wisely. It should be noted that when constructing this Gantt chart I will have to ensure that an overly large amount of time is not allocated to a single task, for example were I to spend too much time on data scraping, my code that trains the application may suffer.

b. Data Set

The data set used to train my application will have to be custom made. As a result, I will have to write a spider using the Scrapy framework in Python. With the spider I will have to gather large amounts of parallel data. This parallel data will be used to train my application using a predefined machine learning algorithm. The main constraint here will be whether or not enough parallel data can be gathered, or at the very least, how and where said data will be found.

c. User Interface

When developing a user interface for this application I will have to ensure that the interface in question is easy and intuitive to use. As this application only has one or two major functions, the interface will be quite minimal. However, it will still have to be designed in such a way that it is aesthetically pleasing to the user.

3. Functional Requirements

Description: This application should be able to perform transliterations from either English-Russian or Russian-English. The result of this transliteration should then be displayed to the user.

Criticality: Of the utmost importance, as this is the main function of the entire application.

Technical Issues: Naturally the only possible issue that could arise when implementing this feature is the possibility of not being able to gather enough parallel data to train the application. As a result training the application will become very difficult. However, I believe that such a scenario is unlikely.

Dependencies: This requirement depends on some of the requirements such as the user interface functioning correctly.

Description: This application's user interface should function correctly and should be easy and intuitive to use.

Criticality: This requirement is extremely important, as without a functioning user interface, even in the event that the application performs a correct transliteration, such a thing will not be apparent to the user.

Technical Issues: At the moment, the main technical issue that comes to mind when implementing a user interface is the possibility of some Cyrillic characters not being visible in the user interface, namely the output from English-Russian and the input from Russian-English. These characters may not be visible if they are not supported. Although Unicode supports these characters, some things may have to be taken into account in order to support the input and output of Cyrillic.

Dependencies: This requirement does not have any particular dependencies.

Description: This application should be runnable on any up-to-date Windows machine.

Criticality: At the moment the aim for this application will be that it will be runnable on any up-to-date Windows machine. As a result this requirement is very important but not extremely urgent as other machines on which it does not run can be accounted for later in the development process.

Technical Issues: One issue that may arise is the event in which a user's machine is missing a specific DLL or does not have the .NET framework installed on their machine. I will have to account for this by installing the .NET framework on the target machine automatically (if necessary). This should all be handled when creating a setup project with which to deploy any executable code, along with installing the .NET framework.

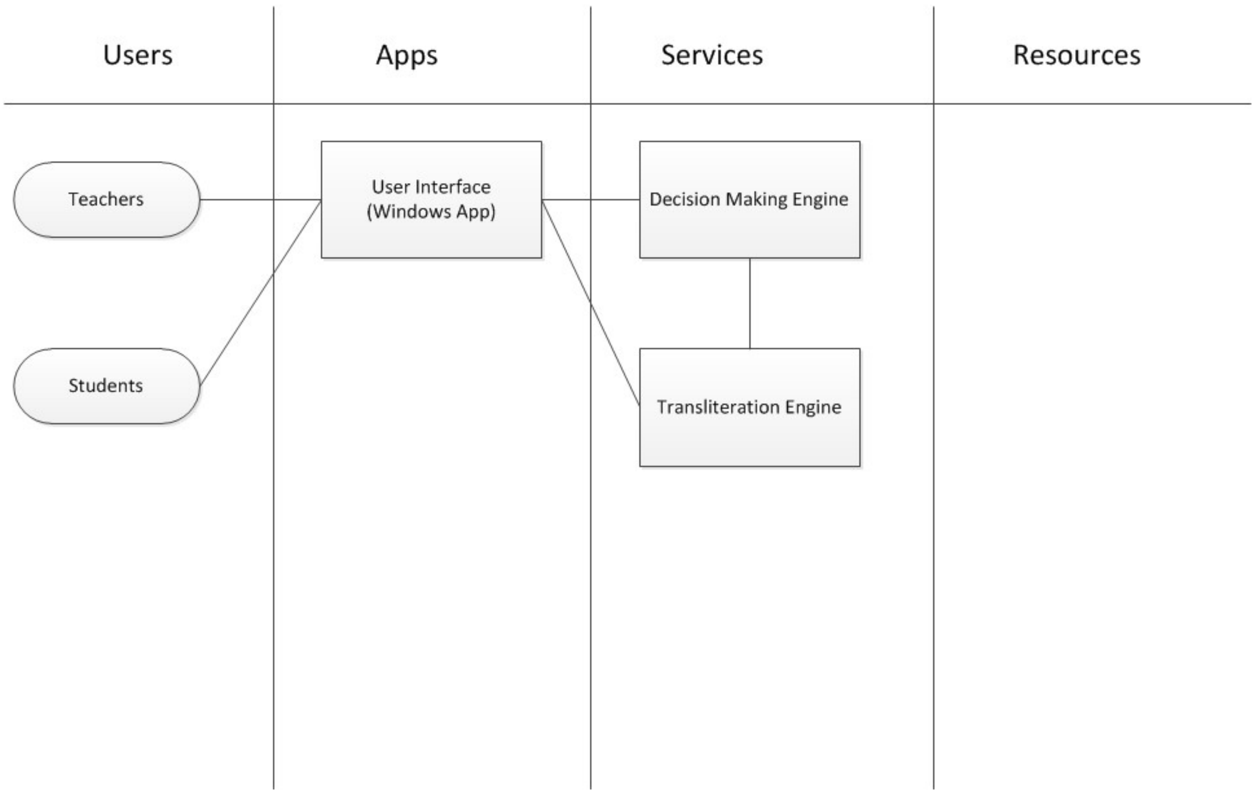
Dependencies: This requirement does not have any particular dependencies.

4. System Architecture

The following is a short explanation of the system architecture diagram shown below:

- Users – There are expected to be two main types of users, teachers and students. Both users will have the same interface shown to them, the difference between these users is the way in which they will use this application. Either to teach using examples or to teach oneself in the case of a student.
- Apps – This mainly describes the user interface that will allow users to carry out transliterations easily.
- Services – Two main services will be called by the UI, the decision making engine and the transliteration engine. The transliteration engine will do as expected, and will take in some input and output the transliterated form of that input. The decision making engine will take in some input and decide how it will be transliterated i.e. what function should be used to do so.
- Resources – As I do not have any information to store for this application I will not require any resources.

System Architecture Diagram

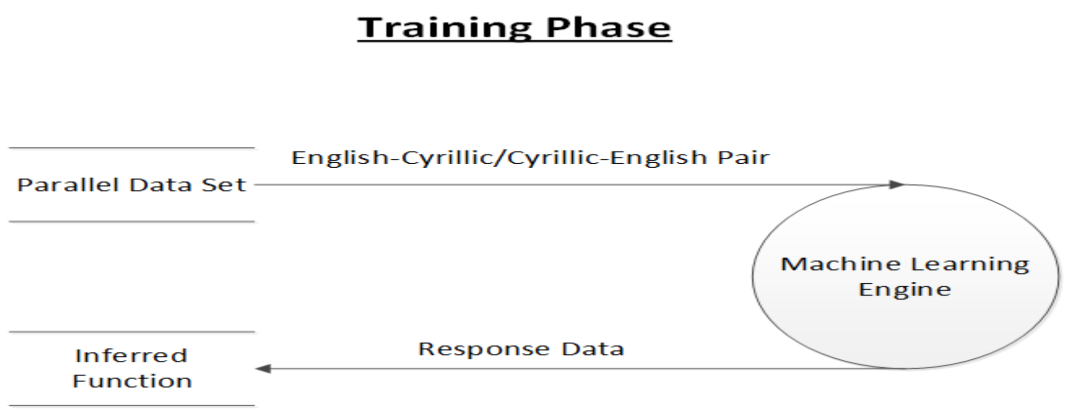


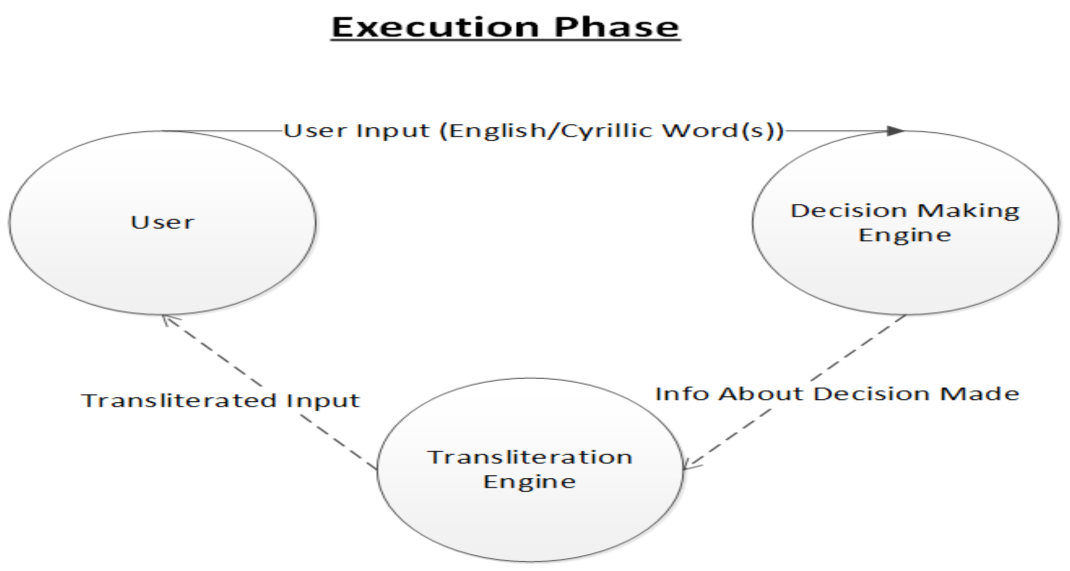
5. High-Level Design

Data Flow Diagram

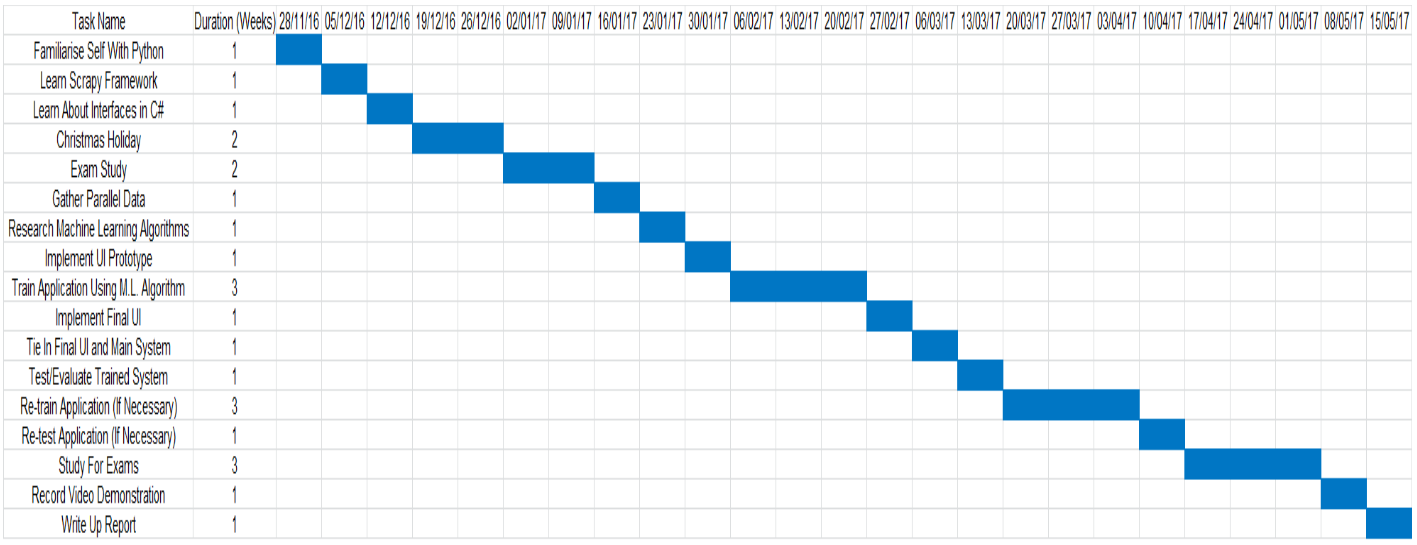
The following is a short explanation of the data flow diagram below. Namely, some of the entities present in the diagram:

- Decision Making Engine – This application component is responsible for deciding on the most accurate transliteration given the input provided by the user. It makes such a decision by referring to everything that it has learned during the training phase.
- Transliteration Engine – This component is responsible for transliteration any input given to it from Latin-Cyrillic or Cyrillic-Latin.
- Machine Learning Engine – This component will implement a predetermined machine learning algorithm, and using the parallel data provided will slowly train itself until it produces a function that can later be used to carry out transliteration.





6. Preliminary Schedule



Task Name	Duration (Weeks)	28/11/16	05/12/16	12/12/16	19/12/16	26/12/16	02/01/17	09/01/17	16/01/17	23/01/17	30/01/17	06/02/17	13/02/17	20/02/17	27/02/17	06/03/17	13/03/17	20/03/17	27/03/17	03/04/17	10/04/17	17/04/17	24/04/17	01/05/17	08/05/17	15/05/17
Familiarise Self With Python	1	🟦																								
Learn Scrapy Framework	1		🟦																							
Learn About Interfaces in C#	1			🟦																						
Christmas Holiday	2				🟦	🟦																				
Exam Study	2						🟦	🟦																		
Gather Parallel Data	1								🟦																	
Research Machine Learning Algorithms	1									🟦																
Implement UI Prototype	1										🟦															
Train Application Using M.L. Algorithm	3											🟦	🟦	🟦												
Implement Final UI	1														🟦											
Tie In Final UI and Main System	1															🟦										
Test/Evaluate Trained System	1																🟦									
Re-train Application (If Necessary)	3																	🟦	🟦	🟦						
Re-test Application (If Necessary)	1																				🟦					
Study For Exams	3																					🟦	🟦	🟦		
Record Video Demonstration	1																								🟦	
Write Up Report	1																									🟦