SE 3XA3: Software Requirements Specification
ReTouch

Team #7, ReTouchers
Abrar Attia - attiaa1
Susan Fayez - fayezs
Mediha Munim - munimm

December 6, 2017

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	Version	Notes
Oct 4	1.0	Updated template with project/team names
Oct 5	1.1	Added func. requirements and definitions
Oct 6	1.2	Added project drivers and issues
Oct 6	1.3	Added charts and work partitioning
Oct 6	1.4	Added non-func. requirements
Oct 25	2.0	Updated FREQ5 and FREQ15, updated list of terms
Oct 25	2.1	Removed NFREQ4 , NFREQ17, and NFREQ23
Oct 25	2.2	Updated NFREQ10, NFREQ24, and NFREQ25
Oct 25	2.3	Updated open issues and OTS solutions
Dec 4	3.0	Updated use case diagram
Dec 4	3.1	Added fit criteria to the non-functional requirements
Dec 4	3.2	Prefixes for non-functional requirements updated
Dec 6	3.3	Removed LF3
Dec 6	3.4	Added FREQ16 and FREQ17

1 Project Drivers

1.1 Purpose

The purpose of this project is to re-implement the open source project K-Touch. K-Touch is a utility that allows users to track their speed and accuracy in typing, and results in improved typing skills through practice and repetition. The re-implementation will improve upon the original project by making it more user friendly and providing more comprehensive documentation.

1.2 Stakeholders

The stakeholders in this project are the clients, the developers, and the consumers. The clients are Dr. Bokhari and the TA's, as they are the ones who commissioned the project and it is their expectations that the developers are trying to meet. We, Abrar Attia, Mediha Munim, and Susan Fayez, are the developers of the project. The consumers or users for this project include any person who wishes to improve their typing skills.

1.3 Mandated Constraints

One of the main constraints for this project is the time frame in which it must be completed. The final prototype must be completed and demonstrated by November 27, 2017 and the final source code must be submitted by December 6, 2017.

Beyond time constraints, a major limiting factor for this project is that it must have similar functionality to the original K-Touch project. It must offer various text "lessons" for the user to type as quickly and as accurately as they can while offering real-time statistics on their accuracy and speed.

The functionality of the actual program will be constrained by the speed and processing power of the computer on which it is run. This is an issue that is present with the original project, as the program experiences heavy delays between the user input and the interface response.

1.4 Naming Conventions and Terminology

- **Lessons:** The selections of text provided by the application for the user to type as quickly and accurately as they can.

- **K-Touch:** The original project. A utility that is designed to help users improve their typing skills.

- **Completed character:** A character is completed if the user has successfully typed it on the keyboard when it was the current character.

- **Current character:** The character that the user is expected to input at a given moment.

- **Incorrect character:** A character is incorrect if the wrong keyboard character was inputted by the user when it had been the current character.

- **Noncompleted character:** A character is noncompleted if it is not a completed character.

- **Lesson beginning:** The moment when the list of words is first generated and displayed on screen.

- **Lesson end:** The moment when the last character in the current list of words has been completed.

- **Results**: The time, typing accuracy, and typing speed of the user calculated at the end of a lesson.

1.5 Relevant Facts and Assumptions

The original project comes with very little user instruction and often experiences heavy delays between the user input and the interface's representation of the user's progress. It also has an unfair policy of requiring the user to type with 100% accuracy in order to advance progress in the lesson. These are issues that need to be addressed in the re-implementation.

The project will be implemented in such a way that it is assumed that the user has basic knowledge of computers, enough to open and run the program. Beyond that, the project will be user friendly and provide comprehensive instructions to ensure the user has an optimal experience. It will

also be assumed that the program will be used by one user at a time on a desktop or laptop computer with a physical keyboard.

2   Functional Requirements

2.1   The Scope of the Work and the Product

2.1.1   The Context of the Work
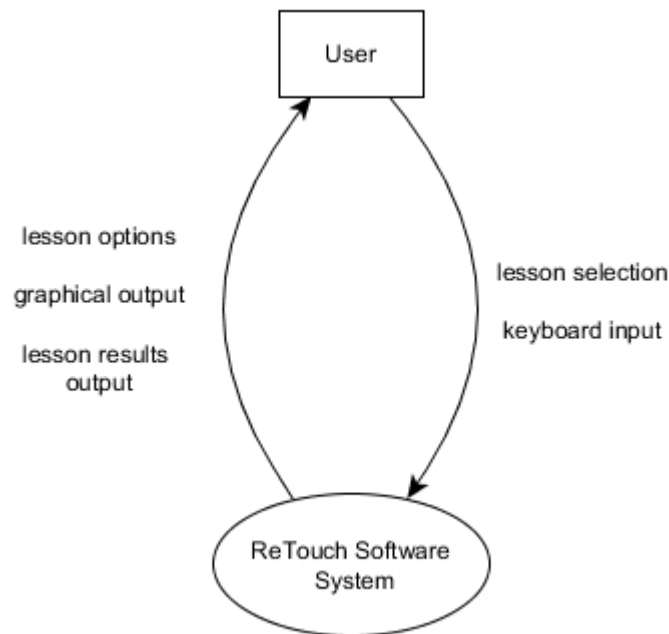
Figure 1: The context diagram for ReTouch.

2.1.2 Work Partitioning

Table 2: The work partitioning for ReTouch.

Event Name	Input/Output	Summary
Lesson Selection	OUTPUT: List of lessons INPUT: User-specified lesson	The user chooses a lesson from those available on screen.
Keyboard	INPUT: Keyboard input	The user's input, consisting of the characters that the user types in.
Graphical Output	OUTPUT: Characters to be typed, real-time user information	The system outputs the GUI, consisting of the characters that need to be typed, as well as the user's current typing speed, accuracy, and timing.
Lesson Results	OUTPUT: User's typing information.	The system outputs the user's overall typing speed, accuracy, and timing data gathered from the completion of the lesson by displaying it on the screen.

2.1.3 Use Cases

Figure 2: The use cases for ReTouch.

2.2 Functional Requirements

Table 3: The functional requirements for ReTouch.
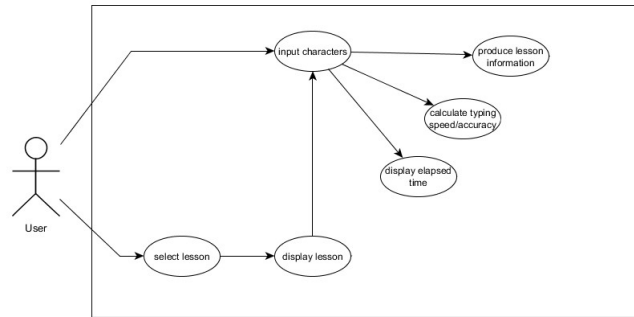
Identifier	Priority	Requirement
FREQ1	2	The system should allow the user to choose between various lessons, each lesson consisting of a different combination of keyboard symbols/letters.
FREQ2	5	The system shall generate a list of characters consisting of the specified letter/symbol combinations. The list shall consist of less than MAX_LESSON characters, including spaces.
FREQ3	5	The system shall display the list of characters using the specified character combinations for the user to type up. The list of characters will be presented on separate lines, with each line being no greater than constant MAX_LINE.
FREQ4	5	The system shall begin the program at the first character (which will become the current character) and wait for the user to type a character.
FREQ5	5	The system shall move the current character to the next character (to the right) when the user inputs a character (incorrect or not), unless the current character is at the end of a line. If it is at the end of a line, the current character will not change.
FREQ6	5	The system shall set a current character as completed if the user presses the same character that is indicated by the current character.
FREQ7	4	The system shall indicate when an incorrect character has been typed by highlighting the incorrect character. All characters typed after an incorrect character will be considered incorrect, and highlighted as well.
FREQ8	4	The system shall allow the user to move on to the next line only when they have reached the last character on the current line and all the characters on the line are correct. When the ENTER key is pressed, the current character will become the first character on the next line.

FREQ9	4	The system shall remove the character to the left of the current character and move the current character to the left when the BACKSPACE key is pressed. However, if the current character is the first character of a line, nothing will happen when BACKSPACE is pressed.
FREQ10	2	The system should count the number of times an incorrect character is typed.
FREQ11	2	The system should display the elapsed time from when a lesson begins to when the lesson is completed.
FREQ12	2	The system should display the typing accuracy of the user.
FREQ13	2	The system should display the typing speed of the user.
FREQ14	4	The system shall end the typing lesson when all characters are completed and ENTER is pressed.
FREQ15	3	The system should display the results of the lesson after the lesson is done.
FREQ16	3	The system should allow the user to return to the lesson selection page during a lesson.
FREQ17	3	The system should allow the user to return to the lesson selection page at the lesson results page.

3 Non-functional Requirements

3.1 Look and Feel Requirements

Table 4: The look and feel requirements for ReTouch.

Identifier	Priority	Requirement
LF1	5	The application will open to an introductory page that has multiple lesson options displayed.
LF2	5	Once the lesson begins, the application will display all the required characters for the lesson in a text box. The current character will be highlighted to notify the user on what character to enter next.
LF3	3	The GUI's components will be consistent in colour, text fonts, and window properties in order to keep the application organized and easy to navigate.

3.2 Usability and Humanity Requirements

Table 5: The usability and humanity requirements for ReTouch.

Identifier	Priority	Requirement
UH1	5	The application interface will be easy to use for any individual who is able to use a computer and keyboard and requires typing practice. The user should be able to navigate all three pages without instruction
UH2	5	The application will be easy to follow and will not require any additional training to use. The user should be able to complete the lesson without instruction
UH3	4	The application will provide typing lessons that appeal and work for individuals who are beginners with typing, people who are considered experts and can type proficiently, and everyone in between. There should be at least one lesson that the user finds challenging.
UH4	4	The application will be easy to install by any user. The user should be able to run the program with the provided instructions.

3.3 Performance Requirements

Table 6: The performance requirements for ReTouch.

Identifier	Priority	Requirement
P1	5	The application will respond to user input within 1 second. Furthermore, every completed character inputted by the user will allow the application to move to the next character.
P2	5	The applications recorded elapsed time will be accurate and begin right when the lesson loads and will stop when the user presses ENTER. This should be apparent from the GUI
P3	4	The application will be reliable and will only fail or crash in extreme unexpected circumstances.
P4	4	The application will be a single-user system.
P5	4	The application will not interfere with the user's machine.

3.4 Operational and Environmental Requirements

Table 7: The operational and environmental requirements for ReTouch.

Identifier	Priority	Requirement
OE1	4	The application will be functional in Windows, Mac, and Linux.
OE2	4	The application will not depend on any other partner applications.

3.5 Maintainability and Support Requirements

Table 8: The maintainability and support requirements for ReTouch.

Identifier	Priority	Requirement
M1	3	The source code of the application will be open to the public.

3.6   Security Requirements

Table 9: The security requirements for ReTouch.

Identifier	Priority	Requirement
S1	4	The application will remain confidential and will not store user results.
S2	4	The applications source code will not be augmented by any changes or updates made by the user when using the application.

3.7   Cultural Requirements

Table 10: The cultural requirements for ReTouch.

Identifier	Priority	Requirement
C1	3	The application shall not be offensive in any way to any potential user.
C2	3	All the text within the application will be available in English.

3.8   Legal Requirements

No legal requirements at this time.

3.9   Health and Safety Requirements

Table 11: The health and safety requirements for ReTouch.

Identifier	Priority	Requirement
HS1	3	The application shall not impose any health risks upon its users. For instance, there will be no extreme flashing lights that may impact users with epilepsy.
HS2	3	A warning shall be placed in the instructions that extreme repetitive usage of the application that could cause strain on muscles and eyes.
HS3	3	The application shall not impose any safety risks upon its users. For instance, the users information will not be made public.

4   Project Issues

4.1   Open Issues

In the re-implementation of this project, the source code will be converted from C/C++ to Java. This presents several issues, the most critical of which being the use of required libraries. These libraries may not be portable to Java, which could hinder the functionality of our implementation.

Another challenge this project faces is the lack of familiarity the developers have with the language the original project is implemented in. One group member has limited knowledge of C/C++, but it will still be problematic for the developers to understand the original source code enough to re-implement and improve upon it.

Similarly, the developers of this project have never worked on a project of this scope. It will be a challenge to ensure that each component of the application runs smoothly, accurately, and concurrently.

The project itself also has issues that can be improved upon. The project we are basing our implementation on runs only on Linux, meaning portability can be greatly improved upon. Additionally, this application only has single user functionality, missing out on the benefits that a competitive element may bring to the user. By putting them against other users, they may be further motivated to improve their performance.

4.2 Off-the-Shelf Solutions

Many similar applications to K-Touch exist that are readily available to users. There are competitive online applications that allow users to race others to accurately complete a selection of text, such as TypeRacer, FreeTypingGame, Nitro Type, and Rapid Typing. These applications solve the competitiveness issue. There are also applications that are more similar to K-Touch, in that they focus singularly on the user and helping them improve their speed and accuracy, such as Typng Master, Key Hero, and 10FastFingers. All of the mentioned applications are available online, solving the portability issue.

4.3 New Problems

The use of this application can potentially cause new problems for the user. One such problem is a potential Repetitive Strain Injury such as Carpal Tunnel Syndrome if the application is used for a prolonged period of time in a non-ergonomic way.

4.4 Tasks

The tasks for this project are essentially to complete the deliverables prescribed by Dr. Bokhari within the time frame he provides. The final code should be completed by November 27, 2017 for the final demonstration. Other deliverables include: the Problem Statement, Development Plan, Requirements Document, Proof of Concept Demonstration, Test Plan, Design Document, Revision 0 Demonstration, Peer Evaluation, Test Plan, Test Report, and User Guide.

4.5 Risks

The biggest risk for this project is that the developers may be too unfamiliar with aspects of the original project to effectively re-implement it. Given the time constraints on the project, they must learn quickly in order to have a viable product at the end of the project term. Another risk is in the conversion from C/C++ to Java. It is possible the new project won't be able to have the same level of functionality as the original in the transition.

4.6   Costs

The cost for this project will simply be the time and energy of the developers. They will invest their energy in coding, learning elements of C/C++, researching libraries, learning how to develop GUI's, potentially learning how to implement wrappers, writing documentation, and presenting their work.

4.7   User Documentation and Training

The final application shall be very user-friendly and easy to use. Comprehensive instructions will be included to diminish any chance of uncertainty in using the product.

4.8   Ideas for Solutions

For the issue of having incompatible required libraries, the first solution would be to search for libraries in Java that have the same functionality as the original C/C++ libraries. Should this solution fail and no viable Java libraries are found, the next solution would be to implement a wrapper for the original libraries to make them usable in Java.

For the issues of inexperience and unfamiliarity, the solution is for the developers to commit themselves to researching and learning about the areas in which they are unclear. Online resources and the help of TA's will be implemented when needed.

4.9 Symbolic Parameters

- MAX_LESSON: The maximum number of characters per lesson.

- MAX_LINE: The maximum number of characters per line.