Software Requirements Specification

for

EECS2311 - TAB2XML

Version 2.0 approved

Prepared by

Group 1: Lian Attily, Alborz Gharabaghi, Robert Mealey, Derui Liu, Isaiah Linares

04/13/2021

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Lian Attily, Alborz Gharabaghi, Robert Mealey	2/24/2021	Updated user stories and fixed layout	2.0
Lian Attily	04/13/2021	Update new requirements	3.0

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detailed description of the “Tab to XML” (T2X) software. It will illustrate the purpose and complete declaration for the development of the system. It will also explain system constraints, interface and interactions with other external applications.

1.2 Intended Audience and Reading Suggestions

The intended audience for this document are the developers & testers for this software, as well as the customer. The audience should be somewhat familiar with musical theory and be able to read standard musical notation as well as tablature for various instruments.

1.3 Product Scope

The software should be able to convert guitar/drums/bass tablature in a text(.txt) file into a musicXML(.xml) file, which can be loaded into standard sheet music creation software (e.g. MuseScore) for viewing and editing. It will be useful to any user that might have trouble reading music tablature and prefers sheet music. It should be able to support tablature from as many websites and formats as possible, and provide a very simple but effective GUI to guide the user through.

2. Overall Description

2.1 Product Perspective

This software is a new, self-contained product that our customer has hired our team to develop.

2.2 Software Functions

The major functions of the software should include the following:

- Importing ascii tablature through copy/paste or an import button.
- Visual representation of the tablature that is able to be edited in real time.
- The ability to specify a Title, Composer and Lyricist, as well as other attributes like Time Signature, Key Signature, the instrument to parse (Guitar, Bass, Drums), and a Tab or Sheet music output.
- Various advanced editing features for the imported tablature
- A conversion of the tablature to a music XML format.

2.3 Operating Environment

This software is designed to operate on any operating system that has Java 12 installed and can run Java desktop applications.

2.4 Assumptions and Dependencies

The software assumes that the user inputs a correct TAB that does not contain any errors. If the input file contains incorrect symbols, the user should be able to preview the tab and fix the errors if they wish.

3. External Interface Requirements

3.1 User Interfaces

The software will interact with the user through a Graphical User Interface (GUI) by prompting the user to paste their tabs into a text-area or upload a text file (shortcut: Ctrl+O) containing well-formatted tabs. The user must then pick between guitar/drum/bass and other required attributes, click the convert button, and specify the name of the *.xml file and its location on their machine.

3.1.1 Input Handling:

- Input file (*.txt) or pasted text must be in ASCII tablature format.
- If the input is not in proper ASCII tab format the user is responsible for an incorrectly outputted MusicXML file.
- If the input file contains symbols that are not recognized as ASCII tablature symbols, the software must ignore them or alert the user if possible.
- As many versions of ASCII tablature as possible should be supported.

3.2 Hardware Interfaces

The user must have access to a keyboard and a mouse (or a touchscreen), to paste the tab and click the convert button and/or add additional information.

3.3 Software Interfaces

- The software will allow importing of a tablature file via text file format.
- The software will allow exporting of a MusicXML file.

3.4 Communications Interfaces

There will not be any external communication interfaces required or used by the software.

3.5 Usability

The usability of the user interface should be tested by studying the experience of users who have not worked on the high level details of the software, and the GUI should change depending on the feedback of these users and their experience. The user interface should be clean, easy on the eyes, intuitive, and simple to understand with little to no guidance. The GUI should be revised if the user study reports any confusion or ambiguity in the design.

4. System Requirements

4.1 Text Area

4.1.1 Description and Priority

Priority: High. The customer would like a text area where the user will upload/paste and edit their tablature with ease. The text area should be large enough for the user to easily review and edit the tabs input before converting them to xml.

4.1.2 Stimulus/Response Sequences

When the user is uploads a *.txt file, the text in the file should be appended to the text area and look identical to the*.txt file they imported.

4.1.3 Functional Requirements

REQ-1: Monospaced fonts should be used to wrap text and present tabs well.
REQ-2: The user should be able to view the input file and edit the text area as necessary.

4.2 Changing Selected Measures’ Time Signatures

4.2.1 Description and Priority

Priority: High. The user should be able to change the time signatures of one or more specific measures.

4.2.2 Stimulus/Response Sequences

There should be an “ADVANCED SETTINGS” button that allows the user to select the measures that need to be changed and the target time signature.

4.2.3 Functional Requirements

REQ-1: There should be a combo box with a list of all the measure numbers that allow the user to select a single measure
REQ-2: There should be FROM measure and TO measure text fields that allow the user to enter a range of measures
REQ-3: There should be a combo box that allows the user to select the time signature.

4.3 Error Detection and Handling

4.3.1 Description and Priority

Priority: High. The software should be able handle error in a smooth and user-friendly way, by either ignoring minor errors or warning the user through the GUI without becoming unresponsive or crashing.

4.3.2 Stimulus/Response Sequences

The user should be able to have typos and mistakes and expect the software to handle them gracefully.

4.4 Instrument Detection

4.4.1 Description and Priority

Priority: Medium. The software should include a feature that will detect the type of instrument that the tab input is in.

4.4.2 Stimulus/Response Sequences

Once the user provides an input tab, the instrument type should automatically appear in the Instrument Type combo box.

4.4.3 Functional Requirements

REQ-1: There should be a combo box that allows the user to see the detected instrument type and change it if needed

REQ-2: The software should be able to detect guitar, bass, and drum tab as input.

REQ-3: The software should produce a valid MusicXML Document according to the detected instrument type.

4.5 Save edited tab and its attributes

4.5.1 Description and Priority

Priority: Medium. The user should be able to edit a tab and its attributes (including title, composer, time signature, conversion type, etc.) and then click a button to save the current changes and settings. Later on during their session, or on a session, the user should be able to retrieve the saved changes.

4.5.2 Stimulus/Response Sequences

The user should be able to click on a button to save changes, and another button to retrieve the saved changes when they’d like to.

4.5.3 Functional Requirements

REQ-1: There should be a “save changes” button that stores the (edited) input

REQ-2: There should be a “load recent” button that loads the saved input

4.6 Additional Information Text Fields

4.5.1 Description and Priority

Priority: Low. The software should include a feature that will allow the user to enter their name, title of the piece, composer, and lyricist if they wish. If the user chooses not to add additional information, these attributes in the .xml file should be left empty.

4.5.2 Stimulus/Response Sequences

The user should be able to enter additional information about the music piece by typing into the respective text fields.

4.5.3 Functional Requirements

REQ-1: There should be text fields that allow the user to enter additional layout attributes if they wish.

REQ-2: The software should be able to accept guitar, bass, and drum tab as input.

REQ-3: The software should produce a valid MusicXML Document.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

REQ-1: The performance of the software depends on the hardware components of the client/user. The software should not be taxing on hardware and should not require a powerful machine to run.

REQ-2: The operation of the software should conclude within a reasonable timeframe. Processing of input files should take an unreasonably long time.

5.2 Safety Requirements

REQ-1: The user could potentially overwrite and lose existing files if they set the output to an existing file, possibly damaging their device if an important system file was replaced. Safeguards should be put into place, preventing the creation and modification of files that normally require administrator access to do so, or possibly restricting the creation of files to a predetermined workspace directory.

5.3 Security Requirements

None of what the user inputs will be transferred or uploaded anywhere online. They will be picking local files from their device to use, and saving the output files where they select on their own device.

REQ-1: Sensitive files could be uploaded to the program, and viewed through the preview pane. Safeguards should be in place to prevent malicious users from using the program to view normally restricted or private files.

5.4 Software Quality Attributes

The user can both import a text file as well as copy and paste text directly into the software's text area.

5.5 Business Rules

The user is allowed to interact with the software solely through the GUI. The user is not expected to go into any of the programs files or code in order to use the software.

6. Use Cases

6.1 **Name:** Import an ASCII tab as a file
Actors: User
Pre-conditions: The user must have a tab stored on a known folder on their PC as a *.txt file format.
Success Scenario: The user may click a button to upload a text file containing tablature. There is also an editable text area for the user to edit their input and review it.

6.2 **Name:** Add additional attributes
Actors: User
Pre-conditions: The user uploaded/pasted an ascii tab into the appropriate text area
Success Scenario: The user types the title, composer, lyricist, their name, and other additional layout attributes into the appropriate text fields before exporting the MusicXML file.

6.3 **Name:** Save changes
Actors: User
Pre-conditions: The user uploaded/pasted an ascii tab into the appropriate text area, and perhaps made some changes to the tab or its attributes
Success Scenario: The user must click on the "save changes" button, a confirmation window will pop up if saving was successful, and an error message will pop up if the textarea is empty and there is nothing to save

6.4 **Name:** Load recently saved tab
Actors: User
Pre-conditions: The user has saved a tab earlier this session or in an earlier session
Success Scenario: The user must click on the "load recent" button under the File menu item. If the textarea is not empty, the GUI will confirm if the user would like to overwrite the textarea's data or append the saved file to the end of the current tab. If the textarea is empty, the recently saved tab and its attributes will appear on the GUI and when exporting the XML file.

6.5 **Name:** Specify: Key and Time Signature, Instrument type, conversion type (Tab or sheet music)
Actors: User
Pre-conditions: The user uploaded/pasted an ascii tab into the appropriate text area
Success Scenario: The user must use the combo box drop down menu and choose the correct Key Signature, Time Signature, Instrument type, and conversion type (Tab or sheet music) before exporting the MusicXML file. If nothing is chosen, the default settings are C Major and 4/4,

guitar, and tab respectively.

6.6 **Name:** Export a MusicXML file
Actors: User
Pre-conditions: The user uploaded/pasted an ascii tab into the appropriate text area
Success Scenario: The user clicks a button to export their input tab as a MusicXML file. The user

provides the name of the file and its path then clicks save. An .xml file representing the correct music notation should be found at the specified path.

6.7 **Name:** Export a MusicXML file without required attributes chosen
Actors: User
Pre-conditions: The user uploaded/pasted an ascii tab into the appropriate text area
Success Scenario: The user clicks a button to export their input tab as a MusicXML file. The user

then provides the name of the file and its path then clicks save. The GUI will warn the user that they have not selected the required attributes and will proceed to select default values for them. An .xml file representing the correct music notation should be found at the specified path.

7. User Stories

7.1 **Scenario:** I am in a hurry and decide not to fill out the "Required Attributes" in the GUI, I quickly paste in my tab and click on "Export", awaiting a successful conversion.
What happened: The GUI prompted me with some warnings for every required attribute I did not fill in, and decided to fill in default values for each attribute letting me know what the default values are. It said the conversion was successful. I threw my new XML file in MuseScore and am delighted to see the output looks accurate to the default values that were chosen for me.

7.2 **Scenario:** I imported a tab into the text area and clicked on export, little did I know there was a small error in the tab I selected.
What happened: The GUI threw me an error, warning that a tab line was null. It then told me that the conversion was not successful.

7.3 **Scenario:** I quickly edited my tab that was in the text area, fixing the tiny error. I then clicked on export awaiting a successful conversion.
What happened: The GUI did not throw me any error message and said the conversion was completed. I checked out the music XML in musescore delighted to see it looks correct!

7.4 **Scenario:** I copy and pasted a guitar tab, filled in every single attribute and clicked export.
What happened: The GUI prompted me with a confirmation message saying it was exported. I check out the XML file on musescore, happy to see it looks perfect.

Appendix A: Glossary

Term	**Definition**
SRS	Software Requirements Document; this document outlining the requirements that the software must fulfil.
GUI	Graphical User Interface
REQ	Describes a *requirement* of the system from the user
User	The person using the software; generally someone who can play an instrument (YouTuber, musician, instructor, student..etc)
MusicXML	XML-based file format representing music notation
ASCII Tablature (tab)	Text file format representing guitar/bass/drum tablature - which is a form of musical notation