Software Requirements Specification

for

TAB2XML

Prepared by Ziqi Zhou
Amer Alshoghri
Hargovind Singh
Uwais Kazi
Ali Yamany

Group 13

Jan 28th, 2021

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to cover all the software requirements for TAB2MXL for it to function as described. TAB2MXL is a standalone desktop application designed to convert different formats of ASCII music tablatures to a popular music description file format called MusicXML. This software will be able to convert different types of tablature (guitar, drums, and bass) into platform and instrument independent musicXML.

1.2 Intended Audience and Reading Suggestions

This document is intended for developers, project managers, and shareholders. Potential users, such as hobbyists and professional musicians, can also use sections 1.1-1.3 to gain a deeper insight on the software description and it's intended goals.

1.3 Product Scope

The main goal of this product is to provide a convenient local software for tablature-to-MusicXML conversion. This product will include a simple graphical interface, convert formats accurately, and provide intuitive ways for users to manage input and output. The software must run locally and must be platform independent.

1.4 References

MusicXML website https://www.musicxml.com/

2. Use Cases

2.1 Product Perspective

The music research community has developed a free format, called MusicXML, that can be used to precisely denote a piece of music. This format is supported by many music apps to display the song in an easy to read fashion, transpose to another key, play the song etc. Conversion from tablature to MusicXML is still performed manually. This product is designed to break the barrier of tablature-to-MusicXML conversion. TAB2XML is designed to be a brand-new and self-contained product coded in Java suitable for all platforms with Java Runtime Environment installed.

2.3 Use cases

1. Save translated MusicXML file

Use-case field	**Description**
Use case name	Output translated MusicXML file
Actors	XML Generation Service
Overview	Actors use TAB2XML to save correctly translated MusicXML files
Preconditions	● The input tablatures must be that of guitar, bass, or drum ● The input tablatures must conform to a valid format.
Main Success Scenario	1. User inputs a music tablature file 2. Parsing service reads the file 3. Parsing service classifies the file as one of Guitar, Bass, or Drum tablature 4. XML Generation Service generates a correctly translated file 5. System asks user for a location to save the musicXML file 6. User selects an appropriate location 7. MusicXML file is saved to the system
Extensions	4a. Translation error: The translation is inaccurate, resulting in an unusable MusicXML file. The program should display an informative error to the user 6a. Output error: User does not have the sufficient permissions to store the file in the selected directory. System displays an error and asks the user to upload the file to another location

2. Display translated MusicXML

Use-case field	Description
Use case name	Display translated MusicXML
Actors	XML Generation Service
Overview	Instead of saving the result to a file, the XML Generation service can display the resulting musicXML content directly
Preconditions	● The input tablatures must be that of guitar, bass, or drum and of correct format ● The input tablatures must conform to a valid format. ● the Parsing service successfully parsed the tablature and saved the appropriate values in memory
Main Success Scenario	1. The MusicXML translation is generated 2. XML Generation Service outputs the contents of the musicXML to the display
Extensions	1a. Translation error: The translation is inaccurate, resulting in incorrect musicXML translation

3. Upload tablature files

Use-case field	Description
Use case name	Upload tablature files
Actors	User
Use case overview	Actors are able to upload tablature files to TAB2XML and have them processed for translation
Preconditions	The input file must be a text file. The text file must exist on actors' local computers.
Main Success Scenario	1. User selects option to upload a tablature file 2. User selects the file from their local system 3. The Parsing service reads, classifies, and translates the file contents 4. The XML Generation Service Generates correct MusicXML 5. The system saves the file or displays the output depending on the option the user selected
Extensions	2a. File I/O error due to insufficient permissions to read the indicated file: System should display an informative error message and ask the user to upload another file

4. Input tablature as text

Use-case field	**Description**
Use case name	Type in the tablatures directly in the input
Actors	User
Use case overview	Actors are able to directly type in the tablatures in the TAB2XML input and have them processed for translation.
Preconditions	None
Main Success Scenario	1. User inputs tablature text into the display 2. Parsing service classifies and translates the tablature into musicXML format 3. XML generation service will save resulting translation into a file or display the translated text based on user-selected option
Extensions	2a. Input error: User did not input correctly formatted tablature. Program should display an informative error and ask the user for an appropriate input

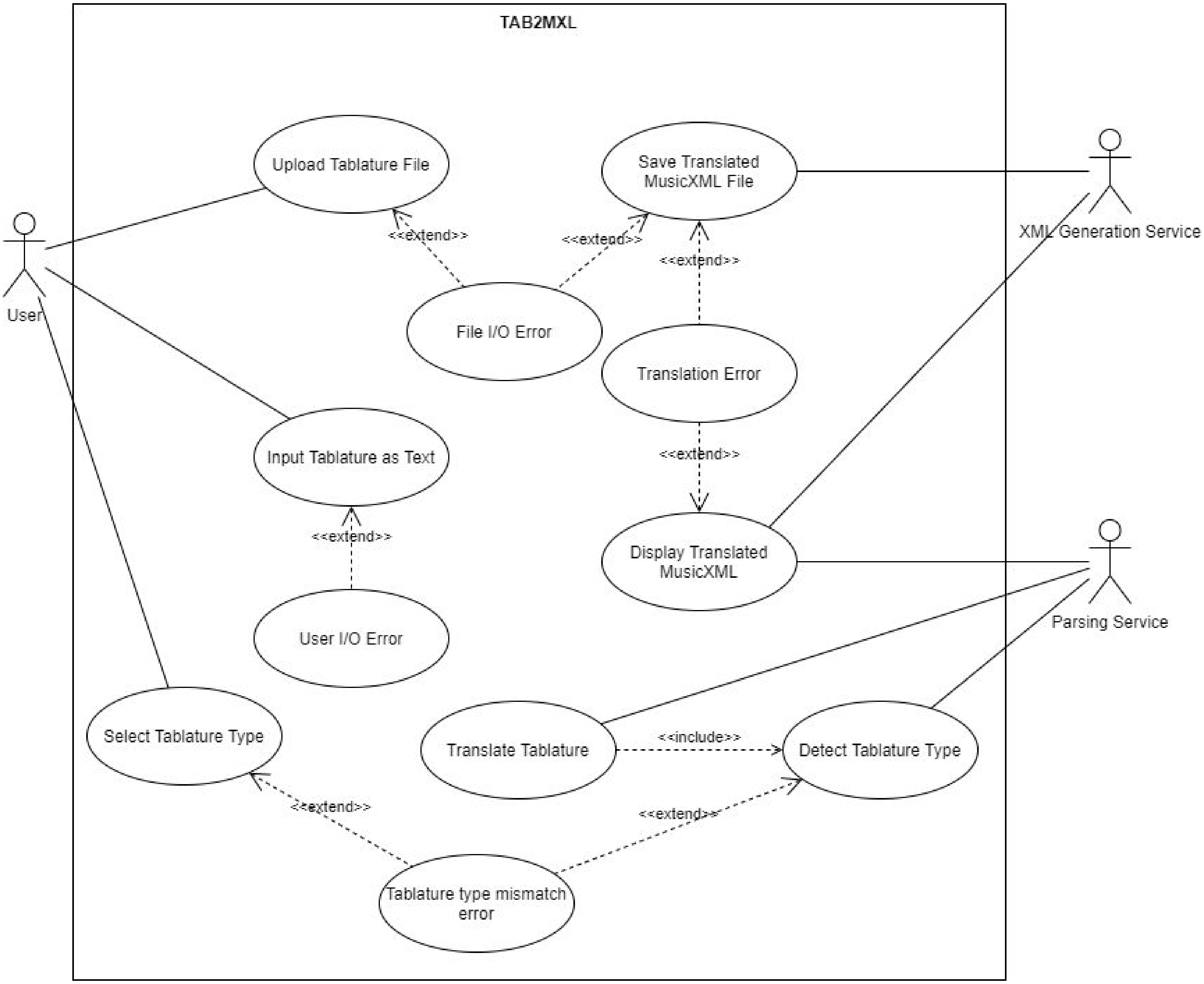
5. Select tablature type

Use-case field	**Description**
Use case name	Select tablature type
Actors	User
Use case overview	Actors are able to choose the tablature types manually either preemptively, or if the system classifies them incorrectly
Preconditions	The desired tablature type is that of bass, guitar, or drum
Main Success Scenario	1. User choose the desired tablature type from the display 2. User inputs tablature through text or by uploading a file 3. Parsing service attempts to translate given tablature using the given tablature type 4. XML generation service will save resulting translation into a file or display the translated text based on user-selected option
Extensions	2a. Tablature type mismatch Error: The selected tablature type does not match the type of the input tablature

6. Detect Tablature Type

Use-case field	Description
Use case name	Detect tablature type
Actor	Parsing Service
Use case overview	The program detects the tablature type directly to educate for MusicXML learner, and to make the process easier for musicians
Preconditions	The input tablature type is that of bass, guitar, or drum. The input content/file is valid tablature.
Main Success Scenario	1. User inputs valid tablature through file upload or text input 2. Parsing Service reads the file into memory 3. Parsing Service detects the tablature type and prepares to apply MusicXML translation to the tablature type 4. XML Generation Service generates valid MusicXML and stores it in memory 5. XML Generation service outputs the resulting translation or saves it into a file depending on user preference
Extensions	3a. Invalid tablature type: The program does not have the ability to detect some available tablatures

2.4 Use case diagram



2.5 Design and Implementation Constraints

TAB2XML must be developed in Java
TAB2XML must be designed, implemented, and deployed by the end of April
TAB2XML secured a limited budget, therefore costly features cannot be accepted into design

2.6 System Requirements

REQ-1: Software must convert music tablature in ASCII form to MusicXML
REQ-2: Software must support guitar, bass, and drum tablature
REQ-3: Software must automatically detect the type of tablature input
REQ-4: Software must allow the user to manually choose the type of tablature to parse
REQ-5: Software must allow both text input and file upload input
REQ-6: Software must allow outputting musicXML as text and as a file

3. Other Nonfunctional Requirements

3.1 Performance Requirements

REQ-8: Software must parse uploaded/pasted text, convert, and output the new format in less than 5 seconds

3.2 Usability Requirements

REQ-9: Software must offer the users the ability to access all available functionality in 2 clicks or less
REQ-10: Software interface must be have evenly spaced buttons no less than 10px on a 1920x1080 resolution screen

3.3 Maintainability Requirements

REQ-11: Software components must be modular and maintainable to support addition, removal, or overhaul of core features
REQ-12: Software must use the git fork-pull request workflow to help developers maintain the project

3.4 Supportability Requirements

REQ-13: Software must perform critical functionality on Windows operating systems
REQ-14: Software must render user interface correctly on 1920x1080 displays
REQ-15: Software must not consume more than 500 megabytes of RAM storage on the device it’s running on