Allegro Tab Converter

DIVERSIFY MUSIC.

Allegro Tab Converter: Requirements Document

Version – 2.0

EECS 2311 – Software Development Project

Prepared For:

Dr. Vassilios Tzerpos, PhD

Department of Electrical & Computer Science

Lassonde School of Engineering

York University

Prepared By:

Rafael Dolores

Mohammed Fulwala

Shawn Verma

Junhyeong Park

Yashraj Rathore

Date of Submission: April 13, 2021

Group Number: 5

Table of Contents

1.0 Introduction

This section introduces the purpose of the Allegro Tab Converter Application, the motivation behind it, and the specific group of audience it targets.

Tablatures are continuously used by many musicians to assist them in understanding the rhythm and duration of a note within a musical piece. Due to their popularity and the difficulty it takes to read them, the music community has developed an open format called "MusicXML" which can denote a music piece precisely. The only issue is that there is no standard way to convert the tab into a MusicXML file.

The Allegro Tab Converter is a software that allows the user/customer to input a .txt or a simple text containing a guitar, bass, or drum tablature for a song and produces a MusicXML file that can be used for various purposes.

1.1 Purpose

The purpose of this document is to provide a comprehensive description of the expected system requirements for the Allegro Tab Converter. It outlines all the features that are required by the customer. To elaborate further, there will be subsequent sections that provide different use cases and user stories where each tackles a specific need or scenario.

1.2 Intended Audience

The development of this software is dedicated to everyone in the music community. It can be used by anyone who wants to learn music, teach music, or for any other recreational purposes. This includes, but not limited to, experts with decades of experiences, an amateur in their second year of playing an instrument, and a beginner who recently decided to learn a new instrument. As the it is made available online, it is expected that the user has a basic knowledge of web/browser surfing to utilize the features and functionality of the application.

1.3 Intended Use

The Allegro Tab Converter is intended to be used for the purpose of converting a guitar, bass, or drum tablature into the online playable format of MusicXML. Alternatively, the application can also be used as an editing tool through an implemented "save current edits" feature.

1.4 Scope

Text tablatures that are stored under a text file format are the only acceptable type of inputs. It is an assumption that the user will provide a text file that obeys the standardized syntax of text tablatures[1]. Western notation or other existing music formats will be rejected upon sight.

In terms of symbols, the Allegro Tab Converter can recognize notes that are presented numerically along with the following techniques:

- h(or H) = hammer-on
- p(or P) = pull-off
- s = slides
- [*note*] = natural harmonics
- g = grace notes
- |* = forward repeats
- *| = backward repeats
- n| = variable number of repetitions (where n is some integer greater than 0)

Consequently, the following techniques will not be recognized:

- b = bend
- / = slide up
- \ = slide down
- PM------- = palm muting (above or below tab)
- ~~~ = vibrato
- x = muted hit

2.0 User Information

This section describes the use of this software to our target audience pertaining to the stated needs of the customer. Several use cases and scenarios were allocated based on these needs.

2.1 Customer Needs

The customer requires a software application that can accurately translate the content of a text tablature into MusicXML fine. This translation is expected to be fast, reliable, free of any errors, and easy to use. It is also their desire for the application to accepts inputs from a directory via a browsing mechanism and a copy-paste feature onto a textbox.

In terms of symbols, the software must be able to recognize any of the techniques outlined by *Capricho Arabe*(guitar) and *War Pigs*(drums). Consequently, the application must be able to generate the correct MusicXml of the two songs.

To establish user-friendliness, an error-detection system was also required by the customer. It should be able to alert the user of mistakes or any unrecognized behaviours from the given input.

2.2 Use Cases

2.2.1 Generating MusicXML Files Through Copy and Paste

Primary Actor: User

Success Scenario

1. The user copies and pastes a portion of their favourite guitar piece into the designated textbox located at the right-hand side of the interface.

2. The user presses the convert button, and the system recognizes it as a guitar tablature

3. The system translates the tablature successfully and displays the output on the designated textbox located at the left-hand side of the interface

4. The user presses the save button and stores the output into the directory of their choice

2.2.2 Generating MusicXML Files Through Directory Browsing

Primary Actor: User

Success Scenario:

1. The user selects a drum tablature from their desktop by pressing the browse button.
2. The user presses the convert button the and the system recognizes that the inputted tab is a drum.
3. The system proceeds with the conversion procedure and displays the output on the designated textbox at the left-hand side of the interface.
4. The user saves the tablature and stores it in their desktop.

2.2.3 Editing Tablatures

Primary Actor: User

Success Scenario

1. The user copies and pastes a portion of some bass tablature they found online.
2. The user performs some editing – addition of new notes, hammering two notes together, etc..
3. The user is satisfied with the changes but decides that they want to convert the entire music piece rather than just one portion, so they saved the current changes to their desktop.

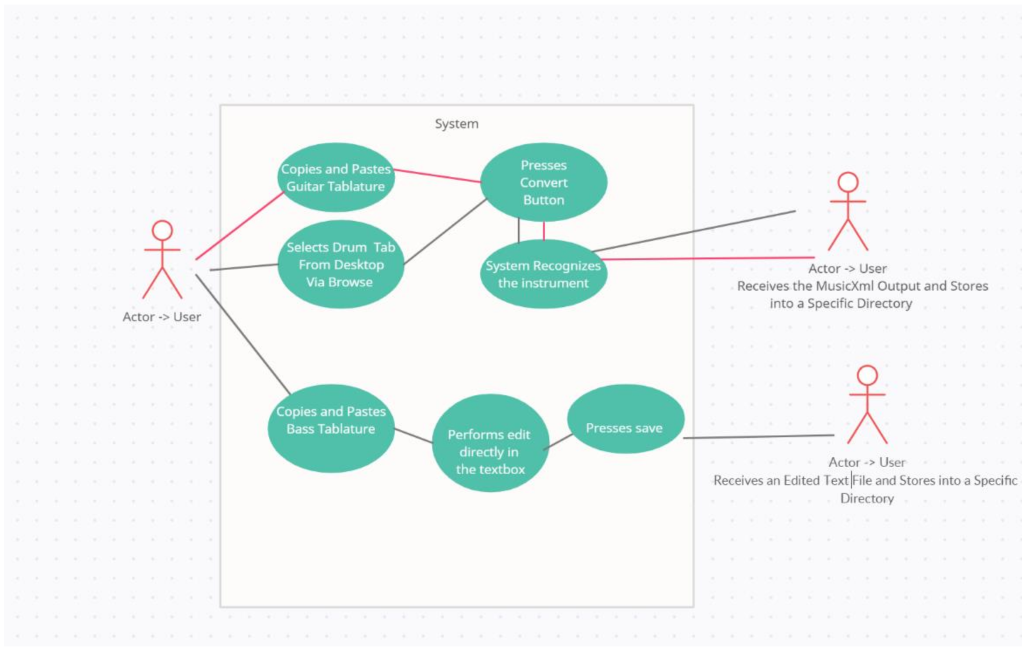

System
Copies and Pastes Guitar Tablature
Presses Convert Button
Selects Drum Tab From Desktop Via Browse
System Recognizes the instrument
Actor -> User
Actor -> User
Receives the MusicXml Output and Stores into a Specific Directory
Copies and Pastes Bass Tablature
Performs edit directly in the textbox
Presses save
Actor -> User
Receives an Edited Text File and Stores into a Specific Directory

Figure 1: The Three Use Cases Under One Use Case Diagram

2.2 User Stories

- As a user, I want the copy-paste feature to display the tab aesthetically and generate the corresponding MusicXML without any error.
- As a user, I want the browse button to access all the directories of my computer so that I can select a file wherever it's stored.
- As a user, I want the application to save my edits so that I can integrate them with a different tablature later.

3.0 FURPS & System Requirements

3.1 Functionality

3.1.1 Capability

This application should accept a text file which consists of the music tablature of

a guitar, bass, or drum tablature or the direct text of the tablature into a text box. It

should be capable of converting this tablature into a MusicXML file and

storing it into the system in which the application is being used.

3.1.2 Reusability

The application can support various types of music tablatures and is run on a stable Java environment. The algorithm is developed with the intention of inherently recognizing the set various instruments and thus can be reused to cater additional requirements with ease. The application delivers a converted .musicxml file, which means there will always be a need for this application.

3.1.3 Security

The application will not store any unnecessary data from the system. It should also not extract any data while the application is running, especially when unauthorized by the user. Furthermore, this application should not corrupt the text file, the MusicXML file, or any other file in the user's system.

3.2 Usability

3.2.1 Human Factors

This application will be user friendly. If the user has not browsed any file, then the user should be able to convert anything. It will allow text to be copied and pasted into a designated textbox. The application will also allow the user to save any edits made to the tablature.

3.2.2 Aesthetics

The interface of the application is designed intuitively to achieve maximum liking from the user. It will be focused on functionality without any unnecessary functions for the users.

3.2.3 Consistency

The application supports guitar, drum, and bass tablatures that are stored in a text file as an input. The application does not allow the user to convert any file format outside of text files. It will warn the user if such event were to occur. The application will also warn the user when conversion fails due to errors in the tablature.

3.2.4 Documentation

A user manual of the software will be provided to the audience upon its initial launch. The code will consist of clear and concise documentation using the JavaDocs feature.

3.3 Reliability

3.3.1 Availability

This application runs on a Java SE runtime environment and must have it

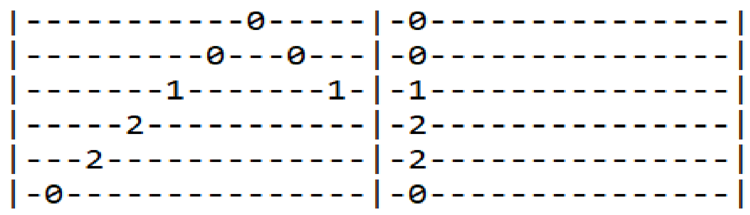
installed on the device to run. The application is supported for all types of OS and

it requires approximately 30 MB for installing the application and to run the

application.

3.3.2 Failure Rate & Duration

The failure rate of the application solely depends on the .txt file that the user has selected. As long it is a .txt file and its contents are syntactically like that of a regular tab(Figure 2), the failure rate of the application remains minuscule.



```
|-----------0-----|-0---------------|
|---------0---0---|-0---------------|
|-------1-------1-|-1---------------|
|-----2-----------|-2---------------|
|---2-------------|-2---------------|
|-0---------------|-0---------------|
```

Figure 2: Sample Tablature Found From the EECS 2311 Course Website

3.3.3 Predictability

The application returns a tablature in .musicxml format upon successful run. It will return with a warning to the user when a file that is not in .txt format is inserted or when the conversion fails due to errors within the .txt file.

3.4 Performance

3.4.1 Speed

The average runtime of the conversion must be less than 5 seconds depending on the size of the .txt file that the user desires to convert.

3.4.2 Efficiency

Assuming that the tablature in .txt format is written is syntactically similar to that of musical tab that the software supports (Figure 2), the application will run successfully.

3.4.3 Resource Consumption

The idea of the application is that it accepts a .txt file or a raw text containing the music tablature and creates a MusicXML file. The loading of the .txt file or the raw text into memory is the only resource that should be consumed during the process.

3.4.4 Scalability

The use of the application is fixed for tablatures in .txt format and only returns. musicxml file. It may support more tablature for different instruments upon development.

3.5 Supportability

3.5.1 Testability

The application returns warnings every time an error occurs within the expectation. It does not return any detailed description on the reasoning of the failure.

3.5.2 Extensibility

This application extends it features to guitar, bass, or drum tablature. If the customer is to request for another instrument to be included, sufficient amount of time will be allocated.

3.5.3 Serviceability

This application is a desktop application that can be run on all desktops with either Mac OS or Windows as the OS that has Java SE runtime environment installed.

3.5.4 Configurability

This application does not support any configure options to the users as it may give the user a non-intuitive experience.

3.6. Plus Category

3.6.1 Design Constraints

- The application must be finished by April 13, 2021 at 11:59 pm.
- The application cannot detect the time signature just based on the tab alone and will require the user to state them explicitly.
- The application will automatically assume a fixed number of strings from each instrument(6 for guitar, 4 for bass, and 6 for drums).
- The application will not be able to detect every error that occurs; thus, it is encouraged for the input of the user to contain minimal number of mistakes.

3.6.2 Implementation

The application is to be developed solely in the Java Environment and will utilize its plug- ins and libraries. The Gradle Build Automation Tool will be used to run the application to save the user from the trouble of importing every used package.

3.6.3 Interface

The Graphical User Interface(GUI) is to contain three textbox and four buttons. Respectively, they are the following:

- A textbox at the left-hand side to accept inputs from the user.
- A textbox at the right-hand side to produce the MusicXML output.
- A textbox at the bottom, in between of the previous two that accepts time-signature inputs from the user.
- A browse button to access the directories of the user’s system and select the desired text tablature.
- Two save buttons; one for saving edits and one for saving the final output.
- A convert button to perform the translation from text tablature into MusicXml.

3.6.4 Hardware Constraints

The Allegro Tab Converter will not require substantial amount of memory space from the CPU and/or a fast internet connection. However, it is required that the application is launched from a desktop or laptop.

4.0 References

[1] “How to Read Guitar Tabs | Simplifying Theory.” https://www.simplifyingtheory.com/how-to-read-guitar-tabs/ (accessed Apr. 14, 2021).