TAB2XML System Requirements

2021 March 26

Contents

1 Introduction **2**

2 Functional Requirements **2**
2.1 Functionality 2
2.2 Usability 2

3 Non-Functional Requirements **3**
3.1 Reliability 3
3.2 Performance 3
3.3 Supportability 3

4 Use Cases **3**
4.1 Convert Text Tab 3
4.1.1 Extensions 4
4.2 Edit Text Tab 4
4.2.1 Extensions 4
4.3 Change Settings 4
4.4 Use Case Diagram 5

5 User Stories **5**

1 Introduction

TAB2XML is a system that can convert text tablature to MusicXML. It is being created for the purpose of allowing people to take advantage of MusicXML's many features, such as rendering, transposing and automatic playing. Its primary audience is musicians.

2 Functional Requirements

2.1 Functionality

- Must convert text tablature to MusicXML tablature
- Must be able to read guitar, drum and bass text tabs
- Must be able to access and edit the original text tab after conversion
- Must be able to save any edits to the text tab
- Must be able to finetune MusicXML output by editing input, including metadata
- Must be able to account for drop tunings
- Must be able to account for an unusual amount of strings in guitar tabs
- Must be compatible with music in any time signature or key
- Should be able to notate techniques such as bends, slides, hammer-ons, pull-offs, etc.
- Must support repeated measures
- Must support grace notes

2.2 Usability

- Must have an intuitive visual interface
- Must be able to accept copy-pasted text or text read from a file
- Must automatically detect which instrument the tab is for
- Must allow the user to override this instrument detection
- Must deal with errors in a user-friendly way

3 Non-Functional Requirements

- Should have an API that can be used by other programs

3.1 Reliability

- Must be able to work with lots of variation in the format of the text tab
- The converted MusicXML tablature must be error free

3.2 Performance

- Must convert the text tab in a reasonable amount of time

3.3 Supportability

- Should allow the user to configure their preferences for how they want the drums sheet music to be displayed (selecting the value and noteheads). Can have a default notation but should be able to be customized through preferences
- Must be testable via automated testing

4 Use Cases

4.1 Convert Text Tab

Primary Actor: Musician
Goal: The musician has a text tab, and they want a MusicXML file
Success Scenario:

1. Musician starts program
2. Musician inputs the text tab
3. Musician inputs any MusicXML metadata (e.g. title, time signature(s))
4. System identifies what instrument it is for
5. Musician tells system to convert text tab
6. System converts text tab to MusicXML

7. Musician saves output

8. Musician closes program

4.1.1 Extensions

3a. If system cannot identify instrument, or it identifies the wrong instrument, musician can choose instrument manually.
5a. If text tab is unrecognizable, notify musician and restart at step 2

4.2 Edit Text Tab

Primary Actor: Musician
Goal: The musician has an incorrect text tab, and they want to make it correct.
Success Scenario:

1. Musician starts program

2. Musician inputs the text tab

3. System identifies what instrument it is for

4. Musician tells system to convert text tab

5. System identifies errors in text tab, and also converts tab.

6. Musician edits text tab to fix identified errors

7. (Optional) Musician converts text tab to MusicXML (scenario of 4.1, start at step 3)

4.2.1 Extensions

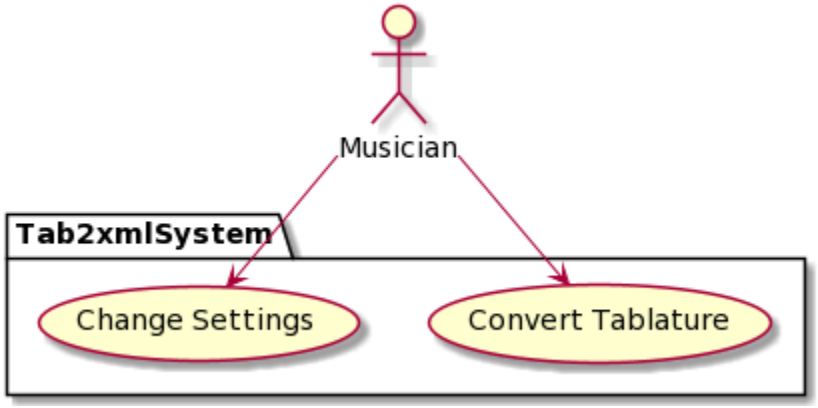
3a. If system cannot identify instrument, or it identifies the wrong instrument, musician can choose instrument manually.

4.3 Change Settings

Primary Actor: Musician
Success Scenario:

1. Musician starts program (if it isn't already started)
2. Musician changes settings
3. System updates settings internally
4. Musician stops program or continues using program for something else

4.4 Use Case Diagram



5 User Stories

1. "As a musician, I want to convert this text tablature to MusicXML so I can take advantage of MusicXML features like rendering, transposing and automatic playing."