TECHNOLOGIES OF MUSIC

Amandip Singh[1], Amit Vishwakarma[2], Bhavik Boricha[3]

[1,2,3] *Final year Student, Dept. of M.C.A, V.E.S. Institute of Technology, India.*

Abstract - *This article presents simple and easy to understand aspects of technology of music in the terms of what they allow us to do. Any subject that requires tool interaction can benefit from the model's analytical and pedagogical applications. This article focuses on music application programming. The findings will be of particular interest to music educators, composers, performers, and researchers seeking a new perspective on the relationship between music and the tools and technologies used to compose and perform. The model incorporates technology in a broad sense, combining traditional acoustic tools with non-electronic as well as electronic and computer technologies. An account of how the frameworks and cost model can be used effectively for teaching and research, and specific examples, is provided.*

Key Words: Music Programming, Music Software Development, Music Technology.

1. INTRODUCTION

It has been a common practice in the field of music production to use musical instruments to create music. In early days the music was recorded using these instruments only. These instruments were also used to record music and to store it. Different types of devices were used to record them. These devices include VHS, Tape recorders, Recording cassettes, Gramophone records, etc. Music has always been a combination of melodies and harmony.

Ever since technology started growing into various sectors, It has also grown in the music Industry. Notes are an important part of music. As these notes are categorized to understand different musical instruments in a single way. For example a song on harmonium could be written down in notes and the same notes could be used to play the same melody with guitar also. In this way notes are a vital part of music. Music has always been re-discovered by experiments.

These experiments in result gave us new types of instruments and skills. Just like beatboxing and playing drums with kitchen utensils. Film scoring is also done using vegetables, water, etc.

These sounds could be used to create some mind blowing music. These days computers can be used to make full music without any instruments which sounds amazing. This kind of music creation has come into the picture using the technology of music. How computers understand music and how we can record, create, edit, use filters, synthesizers, etc to create music.

2. PROBLEM STATEMENT

It is self understood that the IT industry is very different from the music industry. Since for the development of music softwares ought to be done by one or more musicians in the development. A musician might not have any idea about coding and software development.

A musician has different roles in their respective field. Their roles include consistent practice, being ready for performances and keeping their instruments in check. Many well-known singers, writers, and composers may have never used a computer before. It becomes difficult to understand what a software is, how a software works, and most importantly how a software is developed since they might have never written a single line of code. Every artist has started from their respective community and their music shows the touch of their community and culture.

These are the factors which are a fuel to an artist. The most difficult challenge is to figure out how to make music technology simple to learn and use. Overcoming this hurdle will help an artist to synchronize their music potential with technology so that no value of their style is lost in the transition. Finding the best technology for development is another challenge.

There are a number of characteristics which should be fulfilled by the respective technology. The factors that must be considered are related to those that must be considered during any software development stage. Requirements, research and development.

3. LITERATURE REVIEW

Currently available papers are based on the theory of how the nature of music technology should be approached to start learning the process and the advantages of technology of music in creating music.

Focusing on all the stages of the music creation since the development of technology of music started in the industry. The applications of the technology in music creation. One of the papers mentioned the affordances of music making, frameworks and models from the perspective of a musician.

4. TEXT BASED MUSIC PROGRAMMING

Alda

- Alda is a text-based musical programming language which is used in music composition.
- Alda allows us to write and play back music.
- A developer can program using only a text editor and the command line also.
- Alda provides easy to learn markup like syntax.
- Alda is pertinent for musicians who may not know how to code.

Following are some basic building blocks of an Alda program:

- Notes: notes are the keys of music.
- Octave: we can use greater than symbol to go up a higher octave.
- Accidentals: sharps and flats can be added using "+" and "-" symbols.

Examples:

Writing a score:

bassoon:

```
"o2 d8 e (quant 30) f+ g (quant 99) a2"
```

trumpet:

```
"o4 c8 d e f g a b > c4."
```

trombone:

```
" o3 e8 f g a b > c d e4."
```

Summary:

- Alda is suitable for musicians who don't know how to code.
- Alda provides easy to learn markup like syntax.

5. PROGRAMMING SYNTHESIZERS

5.1 SynthEdit SDK

- Synth Edit is a service API for MIDI and audio plugins.
- C++ is used while working with Synth SDK.
- Synth Edit is simple to use and learn.
- Synth Edit has been made to save too much typing or code. Therefore, most of the Synth Edit module is described in an XML file.
- We can also use code blocks for synth edit sdk plugins development.
- There are many examples of modules available in the documentation.

Some of these fields for a sample code are :

- id
- name
- category
- GUI
- parameters

Example of an XML pin:

```
<Pin id="8" name="VoiceReset" direction="in" datatype="float"
hostConnect="Voice/Active" isPolyphonic="true" />
```

Example of one audio pin, one gui pin, one parameter:

```
<Plugin id="SE PatchMemory Float Out" name="PatchMemory Float
Out3"category="Sub-Controls" >

<Parameters>

<Parameter id="0"datatype="float" direction="out"/>

</Parameters>

<Audio>

<Pin id="0"name="PM Value Out" direction="out" datatype="float" parameterId="0" />

</Audio>

<GUI>

<Pin id="0"name="PM Value In" direction="in" datatype="float" parameterId="0" />

</GUI>

</Plugin>
```

Summary:

Synth Edit is good for building synthesizers. Synthesizers are the software or hardware which take an input sound and make some changes of its basic attributes like amplitude, pitch, etc and then gives the modified sound as its result. It could be live or for recording purposes according to the directions given. The output gives the modified sound.

This sound could be produced in different pitches laid on a keyboard. Basic working of a synthesizer is to use one sound and lay it on a keyboard which enables us to create a different type of instrument with the input sound used as a basic building sound of the instrument. Therefore Synth Edit is suitable for developing synthesizers.

6. GUI BASED MUSICAL PROGRAMMING

Pure Data

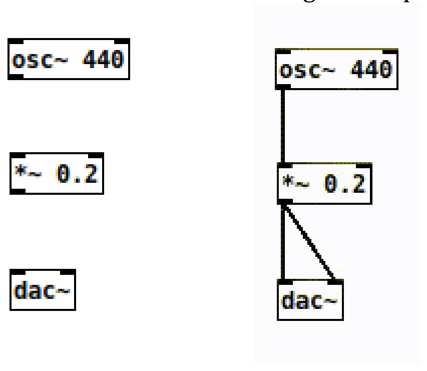
- Pure Data is a Graphical User Interface based visual programming environment.

- Pure Data allows a developer, musician, visual artist, performer, or researcher to graphically create software without writing code.

- Pure Data can also be used to produce and create audio, video, and 2D/3D graphics.

- Pure Data can integrate wearable devices, motor systems, lighting rigs, and other equipment over local and remote networks.

We can start by creating some basic building objects. These objects represent some basic attributes for generating sound in any system Like Frequency, Amplitude, Output path. Using the GUI, we can connect these objects. This will allow us to make a working sound which could be heard after enabling the output source.



We can make a basic sound this way.

Summary:

- Pure Data is useful in learning the foundations of audiovisual processing and development environments.

- Pure Data is also suitable for complex and large scale projects.

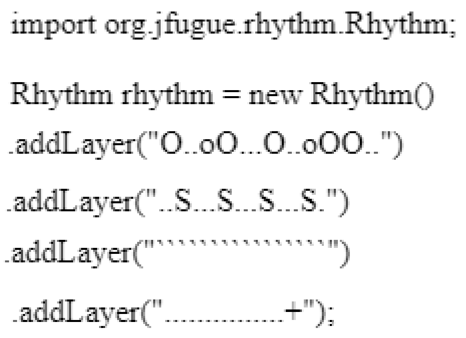
7. JAVA LIBRARIES FOR MUSICAL PROGRAMMING

JFugue

JFugue, an open source programming library, allows a person to program music in Java without the complexities of Musical Instrument Digital Interface (MIDI). MIDI is an electrically powered communication protocol for musical instruments. Java programs in JFugue can be written that will play C-major scale. The program will have a Player library imported and then a class can be created with the main function. Now an object of Player will be created and the play function of Player can be called along with 7 letters or string to produce sound. We can use notes, durations, tempo, patterns, chords and rhythms in JFugue. Consider 'player.play("C D E F G A B");', here notes are alphabets and if we consider this 'player.play("C5 D5 E5 F5 G5 A5 B5");', here notes are present along with octaves which is by default 5. Octaves can be in between 1 and 10, if notes are repeated but octaves are different the sound will differ as well (e.g. 'player.play("G3 G2 A5 B3");') Duration will result in the faster time to play a sound, in above strings a.k.a Staccato, the music will sound interesting as like octaves where default duration 'q' is used. Common durations are 'Q' for quarter note, 'W' for whole, 'H' for half and 'E' for eighth. The duration is shorter or faster than the previous duration by half time and twice slower or longer than the next duration. 'player.play("G4qi G3s A3is B2is");', this will give an interesting sound experience and also we can use multiple duration as we have used in 'G4qi'. It's advisable to provide them with order of length, leading with the longest and end with the shortest. Another important concept is Rest, which means silence, pause or absence of music. We can also use pipes as shown in this string for better readability - 'player.play("G5is E5i Ri | G5s Ris E5q Rs | G5q E5i Rs D5q rs C5h Rs");'. We can also use the '+' operator in our Staccato to concatenate the strings. The musical element in Staccato string is a token, a Pattern wraps it and allows us to manipulate in convenient ways. SetTempo() (to increase the Tempo), setInstrument() (to alter the Instrument), setVoice() (to change the Voice), and add() are all methods in the Pattern class (to add multiple patterns or multiple Staccatos together). The ChordProgression class's setKey(), distribute(), and allChordsAs() methods can be used to construct and edit chords and chord progressions. Using the class we can automatically create chords and set the key using setKey(), distribute all chords using distribute() and can select each chord by typing $ with an index number of the chord as show below :

```
import org.jfugue.theory.ChordProgression;

ChordProgression cp = new ChordProgression("ii V I")

.setKey("C")

.distribute("7")

.allChordsAs("$0hqit Ri $1hqi Ri $2wh Rht");

player.play(cp);
```

Rhythms in JFugue can be used with built-in Rhythm class, addLayer() method works similarly as Voices and getPattern() method will change into rhythm into pattern



```
import org.jfugue.rhythm.Rhythm;

Rhythm rhythm = new Rhythm()
.addLayer("O..oO...O..oOO..")
.addLayer("..S...S...S...S.")
.addLayer("````````````````")
.addLayer("...............+");
```

player.play(rhythm.getPattern().repeat(2));

Staccato is a form of musical expression, it means a short-term note, separated from the note. The string provided to the function comprises a set of musical instructions, which JFugue parses and converts into musical events, which are presented as MIDI by default. The "Staccato" format, which can express all MIDI musical elements and is specifically designed to be easy to read and write, is used here. JFugue has been used in a wide range of applications, including the installation of software projects. In just one or two lines of code, JFugue allows you to accomplish something interesting.

JMusic

JMusic is an open source music programming library written in Java to help melodists and music software inventors by furnishing support for music data structures, variations, and input/ affair to colorful train formats. JMusic was an intimately released exploratory design written by Johannes Vazha Tavdgiridze and Andrew Brown in November 1998. Simple to learn, important to use. jMusic is fluently understood because it builds on conventions of western music. JMusic will not let you hear what you have written until you have completed a program that generates SMF which is also known as Standard MIDI file or audio file. JMusic program labors can be saved to a SMF or. au audio.The introductory process behind computer music composition is: Decide the composition which needs to be composed. Select compositional ways to achieve this. Writing a program that explains compositional ways to the computer system. Run or execute the program. Playback the composition as a MIDI or an audio file. The musical information is stored in a hierarchical fashion grounded upon a conventional score on paper. Score Contains any number of its corridors. Part Contains any number of its Expressions. Any number of Notes can be included in an expression. Please remember that this is unique information concerning a musical piece. The objects contain lots of useful information. Pitch: Pitch of the note. Dynamic: Dynamic is the loudness of the note. RhythmValue: RhythmValue is the length of the note. Pan: Pan is the notes position in the stereo diapason. Duration: Duration is the length of the note in milliseconds Offset: Offset is a divagation from the launch of the note. JMusic helps musicians by providing a familiar music data format based on note/sound events, as well as techniques for organizing, editing, and analyzing that data. Music scores can be saved as MIDI or audio lines, which can then be recycled or played back in real time. JMusic is capable of reading and writing MIDI, audio, XML, and its own lines. JMusic is used in tools, similar as – ChordATune, AI Bass Mutations, Elevated Pitch, Impro- visor, Band Machine, etc.

8. COMPARISON

Tech/Dev	Tool	Synthesizer	Filter	Embedded	Instruments
Alda	Yes	No	No	No	Yes
SynthEdit	Yes	Yes	Yes	No	Yes
Pure Data	Yes	Yes	Yes	Yes	Yes
JFugue	Yes	No	No	No	Yes
JMusic	Yes	No	No	No	Yes

9. CONCLUSION

Comparing these technologies we get a deeper understanding of what are the possibilities for the development of a music tool, synth, etc. Comparing these technologies gives us a better understanding of which tool could potentially solve which one of the real life problems. These observations are essential in the development of any software tool. It is important to have a prior knowledge of the possibilities before starting any discussions of the development. Developing a tool will become easier. In future this document will help all the fellow musicians to understand the importance of the music creation process using the technologies of music.

REFERENCES

[1] https://github.com/alda-lang/alda/blob/master/doc/index.md

[2] https://puredata.info/docs/StartHere/

[3] https://puredata.info/docs/SettingUpADebianBoxForInstallations/

[4] http://www.synthedit.com/software-development-kit/sdk-version-3-documentation/

[5] https://stackabuse.com/jfugue-beginners-guide-part-i-notes-durations-patterns/

[6] https://usermanual.wiki/Document/The20Complete20Guide20to20JFugue2C20Second20Edition2C20v200.723471053/help

[7] https://explodingart.com/jmusic/jmDocumentation/index.html