Flow: Musical Composition via Gesture Interactions with Musical Metacreation

A Thesis Proposal
Presented to
the Faculty of the College of Computer Studies
De La Salle University Manila

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science in Computer Science

by

CHAN, Kevin Gray
DANCEL, Migo POGI Andres
GONZALES, Allen Vincent
TOBIAS, John Patrick

Jordan Aiko DEJA
Adviser

November 14, 2017

Abstract

This study explores the design of an interaction that aims to balance the work of composers with the help of a mobile application. Musical composition is a delicate and disciplined art form that is tedious and repetitive. It involves three main activities: ideation, sketching, and revision. Certain compositional tasks such as figuring out succeeding notes often requires trial-and-error. Existing technology has employed musical metacreation to assist in this process. This endows machines with the artificial creative capacity to perform musical tasks. In review, the existing technology has not been generally used in all stages of the musical composition process. By combining several interaction technologies, composers can benefit by being able to do their tasks with significantly less cognitive load and time.

Keywords: Human Centric Computing, Human Computer Interaction, Interaction Design, User Interface Design, Interaction Techniques, Gestural Input, Sound and music computing, Computational creativity, Usability testing

Contents

List of Figures

List of Tables

Music is a science which must have determined rules. These rules must be drawn from a principle which should be evident, and this principle cannot be known without the help of mathematics. I must confess that in spite of all the experience which I have acquired in music by practicing it for a fairly long period, it is nevertheless only with the help of mathematics that my ideas became disentangled and that light has succeeded to a certain darkness of which I was not aware before.

Rameau (1722)

Chapter 1

Research Description

This chapter is composed of the introduction, overview of the current state of musical metacreation, the research problem, objectives, scope, limitations and significance. The motivation behind the study will also be discussed in this chapter.

1.1 Overview of the Current State of Technology

()

The discipline of composing music can be traced to as far back as ancient Greece (Burney, 1789). It is a specific and careful craft that requires many characteristics such as discipline, skill and immense patience. The approach of composing music itself has given birth to newer genres and sub-genres as composers have grown more creative and personal with their creations (C. R. Miller, 1984). Some composers even follow a strict set of theories and guidelines to maintain the aesthetic quality of the music they create (Rothgeb, 1975).

Music composition goes through multiple stages containing factors that composers have described as both spontaneous and deliberate (Bennett, 1976). This process is illustrated in Figure 1.1. Composers traverse these stages with different methods and actions. But generally, musical composition begins with an initial idea that undergoes several revisions until it culminates into the final draft. However, this usually takes time as musical composition requires trial-and-error and requires experimentation (Gartland-Jones & Copley, 2003; R. MacDonald et al., 2006). Even for experts, music composition can still be a daunting task (J. Kikuchi

et al., 2016).

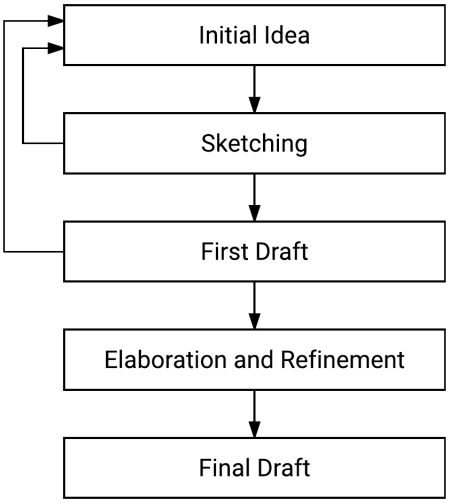

Figure 1.1: The stages of music composition (Bennett, 1976).

There have been several attempts to ease the drafting phase's repetitiveness. To solve this problem, composers along with computer scientists have looked towards *musical metacreation*. This subfield of computational creativity is focused on granting machines the creative capacity to perform musical tasks like composition (Pasquier et al., 2017).

One example of metacreation is found in the case of MorpheuS (Herremans & Chew, 2016), a music generation system that uses pattern detection techniques to generate music. Alternatively, J. Kikuchi et al. (2016) attempted to create a music composition tool with recommendations. They utilized gesture interactions, which are methods of communication with computers that use the fingers or hands to execute commands on a touchscreen device (Kammer et al., 2010). Touch gestures were noted to be a possible medium in the communication of a composer and the computer in the field of music composition (Kurtenbach, 1996). The tool recommended succeeding notes based on the input provided by the user.

However, despite these recent advancements, limitations are still present in the existing technologies. In the work of Herremans & Chew (2016), it was admitted that MorpheuS still lacked in terms of musical output and efficiency. The music composition tool created in the study of J. Kikuchi et al. (2016) was dependent on an existing musical composition because it extracted the rules such as pitch transition and rhythm to predict eventual notes, not from the known strict musical guidelines. This process led to a lack of variety in the generated musical

compositions.

Additional studies that desired for computer generated music to be less "robotic" focused on the expressive trait of a performer through various elements like emotion (Bresin & Friberg, 2000), imitation (Miranda et al., 2010), and selection (Ramirez et al., 2008). One such study focused on the structure of happy melodies where they discerned the emotion of happiness in the algorithm by defining explicit rules and using relations to form a generated theme phrase (Cao et al., 2015). It was also noted that the researchers created variance in the system by adding a factor of randomness in the pitch generation while still keeping true to the theme of happiness (Cao et al., 2015).

From the various algorithms available, most can generate expressive music based on their evaluation metrics (Cao et al., 2015; Bresin & Friberg, 2000; Miranda et al., 2010; De Mantaras, 2012; Ramirez et al., 2008; Miranda et al., 2010) but lacked a usable interface for composers. Computers can help increase the productivity of composers by aiding them during musical composition (Velardo & Jan, 2016). One such system is Computoser which provided the user with a form based interface that generated music based on choices from multiple selections, and a probability and rule-based algorithm (Bozhanov, 2014).

Similar to previous systems, Computoser takes in training data and performs a machine learning algorithm to generate a metacreation model. It was deployed on a website for ease-of-access. However, the interface is form-based and limits the possible compositions (see Figure 1.2).

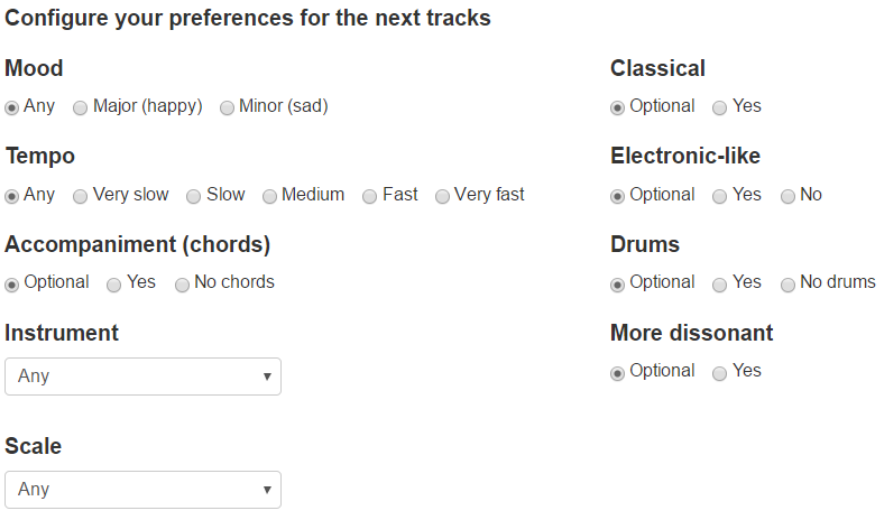

Figure 1.2: *Computoser* preference selection form (Bozhanov, 2014).

The underlying case behind all these findings is figuring out how existing studies on human-computer interactions may augment the current process of music composition. It was found by Brown et al. (2017) that the user experience component was often overlooked when designing interfaces aimed at enhancing the musical composition process. It was suggested in the study of (Levitt, 1992) that a computer "assistant" that helps the composer during creative lapses would be more beneficial than a system that automatically composes music without any control provided to the user.

Numerous works on musical metacreation have focused on the algorithms related to music generation and there are limited studies that explore the utilization of human-computer interaction to augment the musical composition process.

1.2 Research Objectives

1.2.1 General Objective

To augment the musical composition process by integrating rule-based musical metacreation and gesture interactions

1.2.2 Specific Objectives

1. To design and develop an application that users can utilize for basic musical composition tasks
2. To design interactions that will enable users to perform advanced musical composition tasks
3. To assign gesture interactions for doing compositional tasks in a mobile composition tool
4. To validate whether the interaction design improves the user experience when using the tool

1.3 Scope and Limitations of the Research

The application will be developed for a mobile platform. Through the application, composers can perform basic composition tasks. This includes creating a composition, adding notes, editing notes, and deleting notes. The target instrument of the tool for composition is the piano, due to its popularity.

Only certain musical notation symbols will be available for use in the system. The minimum requirement is to allow composers to perform basic musical notation on a mobile interface.

Composers may perform advanced compositional tasks such as musical metacreation and note transposition. Additionally, composers may also edit multiple notes at the same time by highlighting them. The musical metacreation model will also be limited to a rule-based model.

Given that the application is on a mobile platform, the interactions will be limited to multi-touch gestures that can be handled by this platform. Examples of these gestures are tap, swipe, and flick gestures. The interface will be designed to support finger taps, but a stylus may also be used.

The system aims to provide a dynamic and interactive musical composition experience. To achieve this, it is important that the system's interaction design is tested on actual composers while they accomplish compositional tasks. The target users and testers are composers who have basic knowledge in writing sheet music. These are people who can read and write in musical notation for instrumental music, regardless of the platform or tool .

The main goal of the proposed solution is not to improve the quality of music produced by the composers, but to aid them in their composition process. It would not replace the composers, as it would only provide suggestions based on the input given by the composer. It is up to the discretion of the composer whether or not to use the metacreated music.

1.4 Significance of the Research

This study will produce an accessible musical composition tool incorporating concepts from both musical metacreation and human-computer interaction. Given that there are limited studies that use rule-based models, this study will contribute to the emerging field of computational creativity and musical metacreation.

Additionally, the function of the tool will be to augment the musical composition experience of composers. Concepts from both human-computer interaction and music theory will be integrated into the tool to achieve this. Findings from this study will contribute to the knowledge on the process of musical composition, and will provide a background on the integration of human-computer interaction in musical composition.

With the developed system, the improved user experience will help the composer focus more on the composition and the output rather than operating the tool. This would benefit composers that experience creative blocks or have an initial melody in mind, but have no idea how to continue. The tool will aid composers to create music that will still reflect their own style. The personal style of a composer is formed by the culture of the composer. The music created by every composer may reflect their cultural background and personal experiences. This will result in a composition that is a product of an augmented experience but still fully the work of the composer.

Lastly, it is widely known that music has made a great impact on society through the entertainment it brings to people (Hawkins & Burns, 2013; Donnelly, 2005) and also its therapeutic benefits in common psychological disorders (Kemper & Danhauer, 2005). Helping make the composition easier for composers regardless of skill level would allow for better access to music and thereby also granting benefits to society through the several uses of music.

1.5 Research Methodology

This chapter lists and discusses the specific steps and activities that will be performed by the proponent to accomplish the project. The discussion covers the activities from pre-proposal to Final Thesis Writing. It also includes an initial discussion on the theoretical framework to be followed.

1.5.1 Research Activities

1.5.1.1 Planning

This phase concerns the whole group, with the guidance of the thesis adviser. It involves the formulation of the thesis topic, including the research problem, research objectives, as well as the scope and limitations. This would also include

the planning on the approach, and desired output of the study.

1.5.1.2 Review of Related Literature

Relevant works on musical metacreation, musical composition, and gesture interactions are being studied and synthesized by the group to provide context, and the necessary understanding to perform the study. Works related to musical metacreation are essential to the study due to the insight they provide on how past works have attempted to generate music through different algorithms and modalities. Works on musical composition will be reviewed because they provide a background on the musical theories, as well as the attempts made to augment the musical composition process. Finally, works on gesture interactions will also be reviewed to give context on how gestures play an important role in human-computer interaction as well as how past studies have implemented and studied the said technology in relation to the musical composition process.

1.5.1.3 Data Collection

The group will be doing two kinds of data collection. The first one involves research on music theory and the chord progressions present in music. The information gathered on this will be used for building the model for the system. For the second kind of data, at least three (3) composers will be interviewed to gather data on the musical composition process. Being a study related to human-computer interaction, it is essential that the group understand more about how composers compose music in order to design good interactions to augment the process. Interviews will mainly be done face-to-face, with questions focused on how the composer composes music. Specific details like how the composer transitions from testing a melody to writing it digitally or on paper will be observed. Finally, the composer will also be asked to validate or test a prototype of the proposed system. This will mainly be conducted in the School of Design and Arts building in the De La Salle-College of Saint Benilde. However, depending on the availability of the composers, interviews may also be done in other locations.

1.5.1.4 Interaction Design

This phase is all about design thinking. It is concerned with the application's user experience and how it would benefit the target users. The data gathered from the observations and interviews with composers would provide the insight necessary to

understand how they compose. This will be done repeatedly, and will be validated through more interviews and tests. This is to ensure that the proposed solution would integrate seamlessly with the musical composition process.

1.5.1.5 Implementation

Once the interaction design has been laid out and validated, the actual system can then be implemented. The music theories identified during the data collection phase will play a huge part in the development of the model, while the user experience will come from the ideas generated during the interaction design phase. Included in this phase will be the necessary testing in order to ensure the model's correctness, accuracy, and usability.

1.5.1.6 Experiment Design

In order to assess the effectiveness of the developed system, experiments will have to be conducted throughout the course of this study. However, in order for the results to be as accurate as possible, the experiments have to be designed accordingly so that they measure interaction and user experience. Research will be done on how to effectively measure user experience metrics and system interaction. This will involve creating use cases, test scenarios, and identifying the pains and gains of the users.

1.5.1.7 Results Analysis

This activity will include analysis on the findings gathered during review of related literature, data collection, and experimentation. It involves performing quantitative and/or qualitative analyses on the gathered data and review them in order to improve the system.

1.5.1.8 Documentation

The documentation will be done during the entire study. This will include documentation on all the activities performed, as well as their results and discussion.

1.5.2 Calendar of Activities

Table 1.1 shows a Gantt chart of the activities. Each bullet represents approximately one week worth of activity.

Table 1.1: Timetable of Activities

2017-2018	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul
Planning	●	●	●	●					●	●			●		
Review of Related Literature	●●	●●●●	●●●●	●								●●	●●		
Data Collection			●	●●●●	●										
Interaction Design				●●●●	●●					●●●●	●●●●				
Implementation					●●●●	●●●●	●●				●●●●	●●●●			
Experiment Design						●●	●●●●					●●●●	●●		
Results Analysis								●●	●●●	●●●●			●●●●	●●●●	
Documentation	●	●	●	●	●	●	●	●	●	●	●	●	●●	●●●●	●●●●

Chapter 2

Review of Related Literature

This chapter discusses the features, capabilities, and limitations of existing research, algorithms, or software that are related/similar to the study.

2.1 Mathematical Analysis of Music and its Melodies

This section provides an analysis on different works towards modeling music and melody mathematically. Because of the musical nature of this study, it is important to understand the terms used in music and how past studies have made steps to mathematically represent music and its elements.

Loy (2011) defines music as a creative form of art whose medium is sound. It includes several elements (e.g., note, pitch, rhythm, etc.) which can be represented mathematically. One such element is melody. The melody plays an important part in music (Unehara & Onisawa, 2001, 2005) as it is the sequence of notes that define the sound of music (Ryynnen & Klapuri, 2008).

The melody will most likely make the most impact on the listener (Jarret & Day, 2008). These succession of notes usually define the body or general idea of a composition through the form it takes in a musical framework (Jarret & Day, 2008). Whether this form contains quick tempos that provoke the listener into motion or a slow tempo leaving the listener relaxed, it is important for a composer to add melodies that can bring the composition to life (Jarret & Day, 2008).

Poliner et al. (2007) provided a more musicological definition of melody, which is the single pitch sequence that a listener would recognize to be the "essence"

or the "soul" of the composition being listened to. This definition was later supported in the study of Salamon & Gomez (2012), where they attempted to extract the melody from music signals by identifying the most significant pitch in a composition.

Both Poliner et al. (2007) and Salamon & Gomez (2012) borrowed the mathematical representation of melody that was provided by Goto (2004) in his earlier study. Goto represented melody through a predominant-F0 estimation method, or what he called *PreFEst*, where F0 denotes the fundamental frequency. In this method, the melody is identified by finding the strongest, or most dominant harmonic structure within specific regions of the audio signal.

On the other hand, Dörfler (2001) links music signals to Gabor analysis, a method of analyzing music by separating it into segments referred to as frames (Gabor, 1946). Given that music has many high-level and low-level music features that can be represented in the mathematical aspect, Dörfler (2001) relates how music is very similar to a Hilbert Space in terms of digital signal processing. This can be modeled into a set of energy signals that contains many features that she considers to belong in a space of square-integrable functions $L^2(\mathbb{R}), \mathbb{R} \to \mathbb{C}$. Following this, Dörfler (2001) mentions that such spaces which are capable of containing complex set of features are usually infinite in form. However, she mentions that though music may contain such set of features, it is not infinite but rather it is continuous in form. To accommodate this, she proposed a modification of the existing model that accommodates the limitations in finiteness.

She introduces that music is usually measured in length L as members of the complex vector space $\mathbb{C}^L$ where finite sequences are treated as period sequences in forms of what she introduces as *circular extensions* of the said signals coming from $\mathbb{Z}_L = \mathbb{Z} mod L$ (Dörfler, 2001).

This was supported by the earlier work of Fucks (1962) where he focused on analyzing the statistical aspects of music, specifically the frequency distribution of the pitch in music present from 1500 - 1960. This time, he represented the such sequences in what he calls *kurtosis* (K), that is defined as the measure of the sharpness of the peak in frequency distributions. In retrospect, the length of digital signals introduced by Dörfler (2001) are similar in meaning to the frequencies measured by Fucks (1962).

Voss & Clarke (1978) initiated the study on the spectral density of music, where spectral density is defined as a function that determines the strength of the energy of a signal (Stoica & Moses, 1997; Martin, 2001). In their study, Voss & Clarke found that music that exhibited a spectral density of $\frac{1}{f}$ noise, where f is the frequency, was particularly pleasing to hear. This was mainly due to how

the certain frequency was similar to that of human communication. The other spectral density that was observed, $\frac{1}{f^2}$ noise, was considered less pleasant (Gündüz & Gündüz, 2005), and too correlated (Voss & Clarke, 1978; Nettheim, 1992).

Buxton et al. (1978) investigated the possibility of modeling music as data structures in what he calls *hierarchies*. Buxton et al. defined a *hierarchy* as a structure that contains either a single note, or another *hierarchy*. He observed that a musical composition can be separated into chunks and be played separately. Within these said chunks could be other smaller chunks or up to just a single note. By treating music as *hierarchies* or chunks, the music could have multiple levels where whenever a level is played, all the levels beneath it would also be played. The composer would have the ability to choose smaller chunks or lower levels of the music to be able to play a certain part. This system of representing music as *hierarchies* was also supported by Brinkman (1984) in his study with the use of linked lists.

The same ideology was also present in the later study of Bozhanov (2014). In this study, music was modeled as a set of rules. A composition is considered as a structure, with its own properties like rhythm, meter, scale, etc. Similarly, the study of (Schulze & van der Merwe, 2011) also represented music using rules, but more specifically, Markov chains. The composition is modeled using probabilistic automata with the states as its notes.

Shown in Table 2.1 is a summary of the reviewed works and a comparison of their focus and model.

Table 2.1: Related Studies on the Mathematical Analysis of Music and its Melodies

Authors	Focus	Theory/Principle	Contributions/Details
Goto (2004); Salamon & Gomez (2012); Poliner et al. (2007)	Melody	Fundamental Frequency (F0)	Melody is the most dominant harmonic structure
Dörfler (2001)	Music as a digital signal	Gabor analysis & Hilbert space	Music contains features that are infinite in form, but it is still continuous
Fucks (1962)	Pitch	Kurtosis and Frequency Distribution	The pitch can be analyzed in terms of peaks in frequency distributions
Voss & Clarke (1978); Nettheim (1992); Gündüz & Gündüz (2005)	Frequency, Pitch & Melody	Spectral Density	Music generally exhibits a spectral density of $\frac{1}{f}$ noise
Buxton et al. (1978); Brinkman (1984)	Music	Hierarchies or Chunks	A composition is a structure that can be separated into substructures
Bozhanov (2014)	Music	Rule-based algorithm	A composition can be modeled using a structure with rules and properties
Schulze & van der Merwe (2011)	Music	Markov chains and Probabilistic automata	A composition can be represented as a set of states

2.2 Musical Composition & Recommendation Systems

This section contains a review of research papers that discuss systems that produced musical compositions with the aid of various algorithms.

Musical composition systems that attempted to augment the composition process have already been developed in the past. Eigenfeldt & Pasquier (2010) generated music through their system that makes use of Markov models, a method of generating sequences or melodies by deriving features from a provided corpus. However, this also resulted in an inability to generate a deeper, or a varied structure of music, which was also expressed in the earlier study of Ames (1989).

Herremans & Chew (2016) developed MorpheuS which is a system that generates music with the use of tension profiles that are computed from a template of a musical piece. The system uses an optimization approach which avoids the recursive behavior of Markov models during pattern finding to avoid the use of non-standard hierarchical models which lack efficiency (Herremans & Chew, 2016).

Another similar system is SuperWillow which was developed in the study of Schulze & van der Merwe (2011). SuperWillow takes in musical data in the form of MusicXML. The data is then used to generate Markov chains and probabilistic automatons to model musical elements. After composition analysis, the objects generated are then used in order to compose or imitate music similar to the training data. Distribution sampling is used in order to generate music instead of calculating the probable sequence of notes.

M. Kikuchi & Osana (2014) approached musical metacreation using a genetic algorithm to generate melody. The approach used N-gram models to represent rhythm and pitch in each melody block to use in the algorithm. In order to produce melodies, musical features are likened to human genes, which contain different properties. The first part of the process involves random generation of initial individuals that are based on rhythm, pitch, and chord progression. Fitness calculation will then be done to determine which individuals have similar musical features for the crossover. The parent individuals will be used to generate new individuals, repeatedly.

A genetic algorithm was also used the Composition in Genetic Approach (CONGA) (Tokui et al., 2000). The system takes in a fitness score or evaluation in order to generate a better rhythm. Every time the user submits a fitness evaluation, the genetic algorithm and genetic programming starts to improve or sustain the rhythm produced based from their input and returns a rhythm which

is needed to be evaluated by the user again.

The study of Geis & Middendorf (2008) developed a system that used Ant Colony Optimization (ACO) for the generation. ACO also depended on multiple iterations of training similar to previous studies that used genetic algorithms (Tokui et al., 2000; J. Kikuchi et al., 2016). It was believed that composers sometimes broke musical rules in their compositions. This study attempted to model that behavior with how ants followed certain pheromones while searching for food, hence the ACO algorithm. However, the study's limitation was that it did not include a Graphical User Interface (GUI) to allow for real time input (Geis & Middendorf, 2008).

Aforementioned systems focused on musical metacreation, leaving aside user experience, resulting in a lack of interaction. Hyperscore, developed by Farbood et al. (2004), focused on user experience by designing an appropriate graphical user interface for the musical structure. The idea behind Hyperscore was to allow people to compose short melodies with minimal expertise in musical composition. Farbood et al. believed that the new method of interaction allowed users to have an expressive way of shaping musical direction.

J. Kikuchi et al. (2016) supported the idea of musical composition with interaction. They developed a music composition tool that used the Microsoft Surface Pro 3 and the Surface Pen, which enabled the use of gestures for its interactions. It provides users possible pitches to continue writing the composition that is being written. The system involves extraction of pitch transition and rhythm from a musical piece provided by the user as input. Those features are used to generate the recommended pitches and present them to the user in real time. They are then stored in a local SQLite database and searched every time the system is trying to find the right pitch or pitches to recommend to display.

Another system that included a user interface was Computoser, developed by Bozhanov (2014). Similar to previous studies, it uses a rule-based approach but includes probabilistic characteristics in the algorithm for generating music. The rules in the system were based on music theory and with the help of some experts (Bozhanov, 2014). As seen in Figure 1.2, the system enables the user to adjust the weights in the preferences with which the music is generated. The result is a composition based to the choices the user made and can be immediately played in the web browser through a media player present in the website as seen in Figure 2.1.

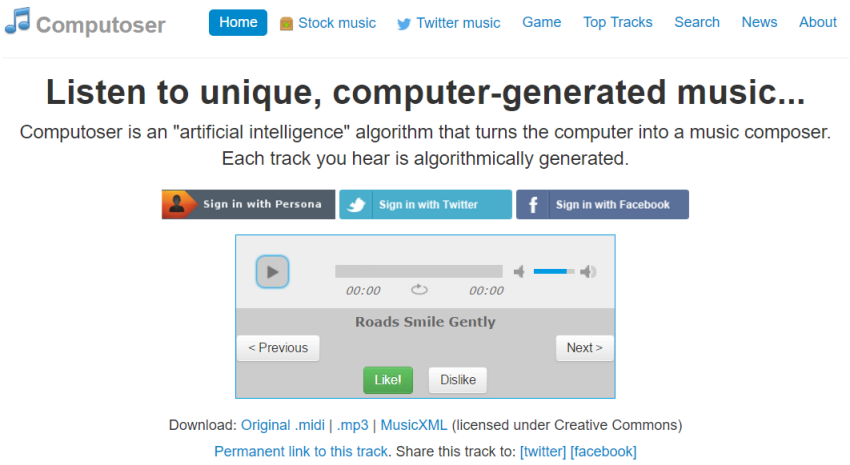

Figure 2.1: *Computoser* media player (Bozhanov, 2014).

Table 2.2: Related Studies on Musical Composition & Recommendation Tools and Systems

Authors	Name of Tool	Platform	Input/Modality	Algorithm
Ames (1989); Eigenfeldt & Pasquier (2010)	NA	NA	MIDI files	Markov models
Herremans & Chew (2016)	MorpheuS	NA	Template piece	Tension profiles
Camurri et al. (1991)	HARP	NA	Source composition/piece	Networks
Tokui et al. (2000); M. Kikuchi & Osana (2014)	CONGA; NA	PC; NA	Fitness Score; NA	Genetic Algorithm with Genetic Programming; Genetic Algorithm with N-grams
J. Kikuchi et al. (2016)	Music Composition with Recommendation	Tablet (Microsoft Surface Pro 3)	Touch with the use of Surface Pen	Database of Features
Bozhanov (2014)	Computoser	Web	Form-Based	Hybrid Rule-Based, Probability Driven Algorithm
Geis & Middendorf (2008)	NA	PC	Length of melody & Set of allowed pitches	Ant Colony Optimization

2.3 Music Generation and Evaluation

In the current metrics of computer generated music evaluation, the objective is to make the music feel expressive wherein the sound of the music does not sound monotonous and complies with the unique changes every composer adds to their piece (De Mantaras & Arcos, 2002). It is where the music generated feels human, and has these subtle differences in tempo, pitch, timbre, and volume (Bresin & Friberg, 2000). Expressiveness is a criterion for music and is defined as how close a computer-generated melody is to its human counterpart (De Mantaras & Arcos, 2002). Evaluation metrics range from how a generated piece not only conforms to the musical metrics and standards, but also to the change in dynamics that are found within songs composed to evoke certain emotions within the audience by the artist (Bresin & Friberg, 2000).

The TempoExpress system aimed to produce expressive music from monotonous music recordings (Grachten, 2006). The system used a principle called case-based reasoning (CBR), where it used cases as knowledge for the computer to refer to when generating a musical piece. CBR was used because of how the nature of music can differ with every piece. It can contain all the specific changes in each musical piece (De Mantaras, 2012). The evaluation was done through a survey of 92 participants and then run through a performance measure (Grachten, 2006).

Saxex, a system that generates expressive saxophone jazz solos, also uses CBR. CBR performed well to help the system deal with the dynamics, rubato, vibrato, articulation, and attack of the notes in the jazz solo generated by the system (Arcos et al., 1998). The resulting melodies where then run through a spectral modeling synthesis (SMS) technique then analyzed whether the music was expressive or not (Arcos et al., 1998).

An Imitative Multi-Agent Perform (IMAP) was used in the study of Miranda et al. (2010). The IMAP mimics how people listen to music and use it as inspiration for their own composition. The system used rules to evaluate given music, and only used the highest rated music as inspiration. It was then tested using certain algorithms to evaluate its success when generating melodies after training (Miranda et al., 2010).

MorpheuS, developed in the study of Herremans & Chew (2016), was tested by playing the results of the system in different musical events around the world. Although the results turned out to be favorable, the researchers plan to further improve the overall efficiency of the system. Other improvements in the system include the quality of musical output by imposing more restrictions regarding the playability and the analytical properties of a style of music.

In the study of Schulze & van der Merwe (2011), SuperWillow was evaluated through a partial Turing test. Respondents were asked to identify human compositions versus computer-generated compositions. It was found that 72% of the respondents either could not tell the difference between the composition composed by the computer and the human, or preferred computer-generated compositions over human compositions.

J. Kikuchi et al. (2016) performed testing on users with little experience in musical composition. They compared their system with another composition tool that was a Piano-roll type. It was found that their system allowed for more variety in compositions compared to the other system. Despite this, the system was still limited to the pitch transitions and rhythm of the training input provided (J. Kikuchi et al., 2016).

Yüksel et al. (2011) also developed a software that utilized a genetic algorithm. Their system generated notes using a genetic algorithm with recurrent neural networks. The algorithm used properties such as note, octave, and duration. Although it was only tested by the researchers, it was found that the metacreated music was only viable after a number of evolutions have been made.

Alternatively, in the study of Birchfield (2003), a genetic algorithm was used with consideration for musical emotion, meaning, and form. The testing found that average listeners found the metacreated music to be interesting. On the other hand, a focused intensity was noticed by the experts due to the shifting rhythms, gestures, and phrasing (Birchfield, 2003). Although experts also noted that the music contained some idle sections. This affected the emotion evoked by the music (Birchfield, 2003).

Table 2.3: Related Studies on Music Generation and Evaluation

Authors	Name of Tool	Testers	Comments and other findings
J. Kikuchi et al. (2016)	Music Composition with Recommendation	1 inexperienced composer	The compositions produced were found to have more variety compared to the other tested system.
Herremans & Chew (2016)	MorpheuS	Played in concerts	The system can generate polyphonic music with themes and tension profiles. The output of this system was said to be promising and further improvements in terms of efficiency of this system could be done.
Schulze & van der Merwe (2011)	SuperWillow	256 Stellenbosch University professors and students	Found that 38% of the respondents incorrectly identified the computer-generated composition as a human composition.
Grachten (2006)	TempoExpress	92 testers	The system used Case-Based Reasoning and was compared to Uniform Time Stretch and was found to be better for generating a tempo that was slower than the source tempo.
Arcos et al. (1998)	Saxex	Only tested by the researchers	The results satisfied the criteria of expressiveness for dynamics, rubato, and vibrato but it still needed work for the articulation and attack.
Yüksel et al. (2011)	N/A	Only tested by the researchers	Generated music that has passed through many generations can be judged as acceptable.
Birchfield (2003)	N/A	Tested on average and expert listeners but no mention of number	Average listeners describe the produced music to be varied and interesting while the expert listeners described it to have focused intensity and exciting moments but contained idle sections that could destroy the mood.

2.4 Gesture Interactions

Another way to augment the process of musical composition is by focusing on what feels natural to the target users. In recent years, the use of touch interfaces have proven to feel more natural than using external input devices like the mouse (Travis & Murano, 2014). Gesture interactions have been used to communicate with modern innovations like the touch-screen and are considered to be intuitive and interaction focused (Epps et al., 2006; Kammer et al., 2010).

Gestures improve the way humans interact with computers because it is the physical expression of their thoughts (Westerman et al., 2001). It is a more natural way of interacting when compared with other interfaces. The common keyboard and mouse setup according to Westerman et al. (2001) is inefficient for different reasons:

(a) Considerable amount of time is wasted when switching between keyboard and mouse input.

(b) It requires users to learn different skill sets when using two input devices.

(c) Most input devices only require one hand to operate.

(d) The setup of mouse and keyboard may cause musculoskeletal disorders.

On the other hand, the use of multi-touch gestures have several advantages over other interfaces:

(a) It gives users tangible and visual feedback.

(b) It allows users to relax their hands which causes their shoulders to relieve from stress.

(c) It simplifies the typing and pointing action into a more efficient one.

(d) It makes humans perform faster with its intuitive nature.

(e) It is cost-effective and less computationally heavy.

Kammer et al. (2010) support the use of gestures, specifically multi-touch gestures, as a method of interaction with computers. They introduced the Gesture Formalization for Multi-touch (GeForMT) approach to formalize gestures for

extensibility and re-usability. The GeForMT approach is rooted in a linguistic concept called *semiotics*, which describe the different factors that affect the creation and interpretation of signs and symbols (Kammer et al., 2010; Eco, 1976; Nespoulous et al., 2014).

In the study, the general term used for gestures is *atomic gestures* (Kammer et al., 2010). It includes basic shapes like lines, curves, and circles which may still be improved to form more complex gestures (see Figure 2.2). Other types of gestures include freeform (MOVE), which can perform commands for a specific gesture trail. Another is short contact (POINT, HOLD) which does not require moving the fingers. One can also specify the direction of a gesture trail like in the case of LINE_NE.

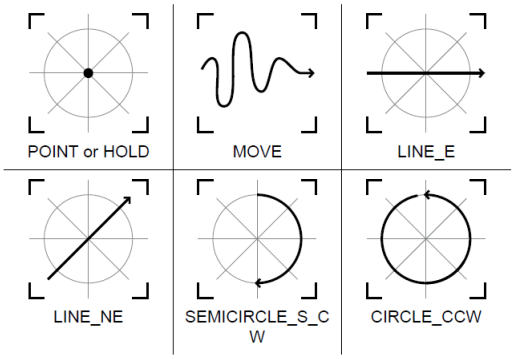

Figure 2.2: Examples of atomic gestures (Kammer et al., 2010).

The same idea is found in the study of Ip et al. (2005), where a system called the Cyber Composer was developed to make use of hand gestures to create music (see Figure 2.3). The system generated music from blueprints for musical structures. These can be accessed and modified through the use of an external input device called the CyberGlove. This form of input was considered helpful for people without any background in composition. With the help of the CyberGlove, they could create music with only simple hand gestures (Ip et al., 2005). However, despite the help provided by the CyberGlove, it was still found to be obtrusive and inaccessible.

Figure 2.3: A user operating the CyberGlove (Ip et al., 2005).

In the study of Epps et al. (2006), the use of gesture interactions as a method of input for tabletop displays was explored. It was found that the index finger was commonly used by the testers when interacting with a large, flat display. Thus, the study recommends to build gesture interactions that mainly revolve around the use of the index finger (Epps et al., 2006).

The aforementioned study of J. Kikuchi et al. (2016) also used gestures through the use of the Surface Pen. The developed system's mode of interaction involved using the Surface Pen to interact with a Microsoft Surface Pro 3's touchscreen. The simplified interface and intuitive interaction was found to be helpful for inexperienced composers write music.

Kazi et al. developed Vignette, an interactive illustration tool that allows users to manipulate textures through the use of freeform gestures. Although not a musical composition tool, Vignette has similar goals in that it aims to provide an easy-to-use interface that incorporates gestures to augment the illustration process. Texture manipulations such as duplication, synthesis, and fillings can be easily done in Vignette through simple gestures made in the interface (Kazi et al., 2012). The overall impressions of users during evaluation were positive, and comments were that it was easy to learn and allowed them to create quick illustrations in only a short time of use.

The later work of Kazi et al. (2014) builds on Vignette by extending it to allow animations. The new tool, called Draco, is a sketch and animation tool that provides a simplified interface for animation through the use of *kinetic textures*, which is their developed framework for endowing sketched components with motion properties. It allows users to easily create simple animations through intuitive gestures and freeform strokes. Raindrops can be animated by drawing simple curved or linear lines that denote its direction and length of travel.

The earlier study of (Mo et al., 2005) used cameras to detect gestures. Smart-

Canvas is a drawing desk system that allows users to create drawings through gestures. This is done with 2 cameras that detect the type of gesture and identify whether or not the user is touching the surface. This resulted in a minimal drawing system without the need for a touchscreen (Mo et al., 2005). However, there was one main drawback. Since users would draw on a surface and the drawing would be generated on the computer screen, the user would have to keep switching his/her focus when drawing.

Table 2.4: Related Studies on Gesture Interactions

Authors	Name of Tool	Input/Modality	Types of Gestures	Comments and other findings
Kammer et al. (2010)	GeForMT	Multi-touch gestures	Tap Point or Hold Linear gestures Circular gestures	Formalized an approach to create gestures
Ip et al. (2005)	Cyber Composer	CyberGlove	Hand motions Finger bends Finger flexions	Obtrusive, but allowed composition through hand gestures
J. Kikuchi et al. (2016)	Music Composition with Recommendation	Microsoft Surface Pro 3 and Surface Pen	Tap gestures	The interface was found to be helpful for inexperienced composers
Epps et al. (2006)	NA	Hand Gestures and Cameras	Single-touch gestures Hand-shape gestures	The index finger was commonly used for interactions
Kazi et al. (2012, 2014)	Vignette; Draco	Touchscreen device and stylus	Freeform sketches Straight lines Curved lines	The interface was easy-to-use for beginners and the gestures were intuitive
Mo et al. (2005)	SmartCanvas	Touch Gestures and Cameras	Freeform gestures	It was minimal but increased cognitive load due to users having to switch focus when drawing

Chapter 3

Theoretical Framework

This chapter provides a thorough discussion of the different studies and theories behind the three main concepts of this study. The first few sections will tackle research on Human-Computer Interaction. The next set of sections will focus on User Experience Evaluation methods, and finally, Musical Composition and Metacreation.

3.1 Human-Computer Interaction

The study of Human-Computer Interaction (HCI) is about how technology can make an impact to human life (Dix, 2009). Its birth can be traced back to the studies of Shackel (1959) and Engelbart (1962). Both studies explored the approach of developing systems or interfaces for the benefit of the human intellect. This research stemmed from the increasing complexity of human tasks, and the need for technology to augment the capabilities of humans when performing these tasks (Engelbart, 1962).

HCI is multi-disciplinary, borrowing principles and theories from psychology, sociology, and computer science (Dix, 2009; Sinha et al., 2010). In HCI, one must learn to understand the users, which can be done through user observation or interviews, to develop systems and interfaces that would fit their backgrounds and needs (Fischer, 2001).

One of the core topics in HCI is the issue of usability (Frøkjær et al., 2000; Dix, 2009). Usability tackles questions about effectiveness, efficiency, and satisfaction. Effectiveness asks if users can achieve their goals or perform their tasks properly

and correctly when using the system (Frøkjær et al., 2000). Efficiency talks about whether users can perform the said goals with minimum use of resources or effort (Frøkjær et al., 2000). Finally, satisfaction is about finding out if users enjoy using the system or feel comfortable using it (Frøkjær et al., 2000). These usability issues can be achieved through the proper use of design principles (discussed in Section 3.1.1.1).

The growth of technological study and the emergence of new technological platforms brought with it new subfields under Human-Computer Interaction such as interaction design, user interface software technology, and user experience.

3.1.1 Interaction Design

Interaction functions as an essential factor to society's progress (Rogers et al., 2011). Humans interact with other humans or objects on a daily basis. These objects could be mobile phones, laptops, ATMs, and more. But not all of these have good usability or interaction. In that case, the interaction of these objects might not have been designed with the user in mind (Rogers et al., 2011). Even though these objects may be considered usable, they might not provide the best user experience when being used.

The goal of interaction design is to lessen these bad experiences when using objects, or in this case, technology (Shedroff, 1999; Rogers et al., 2011). User satisfaction is imperative to interaction design. It is about designing interactions that would not only benefit users, but would also reduce cognitive load and make the experience enjoyable or pleasurable (Tidwell, 2010).

The continued study of interaction design allowed researchers and designers to identify a set of design principles. These principles serve as guidelines to designing good interactions and interfaces (Rogers et al., 2011).

3.1.1.1 Design Principles

What most developers or software development companies forget is that it is not always about how aesthetically pleasing the product is, rather, it is about how functional and easy it is for users of varying experience to understand how to use it (Blair-Early & Zender, 2008). The overall system design should incorporate both functionality and visual appeal to provide value to its users.

According to Blair-Early & Zender (2008), the goal of the software should

be decided and set at the start so that the interface is designed to support that goal. The design should make it obvious what the goal is all about. However, this should not make the design any less aesthetically pleasing. The designer should keep in mind what the software aims to do as well as making it pleasing for users to view and use.

Developers and designers should work together to form the design and user interface of the system. According to Galitz (2007), the interface is the most important part of the system because it is what the users could see and interact with all the time. A testament to this importance is the finding that good design could help users become more productive and reduce cognitive load by up to 40% (Galitz, 2007).

Similarly, it was found in the study of (Cope & Uliano, 1995) that redesigning just one web page to follow good design standards allowed a company to cut costs by $20,000. This redesigned web page increased user productivity and reduced user error and training time.

Mandelbaum (2014) notes four key principles to consider when designing products and interfaces (see Table 3.1):

Table 3.1: Key Design Principles

Design Principle	Description
Axis	The Axis, is an imaginary line that arranges, and aligns elements (Mandelbaum, 2014). It is considered as the most basic of the principles, but is equally important. When objects in an interface fall in or follow one axis, the design would give the impression of order. The axis allows movement in a straight line that the user's eyes can follow (Mandelbaum, 2014). The axis directs users towards something in the design.
Symmetry	The second principle, Symmetry, works together with the axis to promote balance (Mandelbaum, 2014). Symmetry enables balance between both sides of the axis, aligning and making both sides of an interface similar. Having symmetry gives users the feeling of harmony, and even the slightest difference in alignment or order can throw of the balance of a design and give a feeling of discomfort (Mandelbaum, 2014).

Hierarchy	Hierarchy is about showing the importance of certain elements in a design (Mandelbaum, 2014). One example of this is making one article in a news feed larger than the rest. The larger article would catch the users attention first and give the impression that it is more urgent or important compared to others. Other ways to show hierarchy could be through its placing or its shape.
Rhythm	Lastly, Rhythm, denotes patterns in the design. Having rhythm in the design allows users to find patterns and better recognize and know where to look for particular information (Mandelbaum, 2014). This can be seen in a Twitter feed where a list of tweets would look the same, helping users where to find the like button, or the user's twitter handle, etc.

Although they are not solid rules, these principles can be considered as building blocks that can be used to build an interface. The design is considered as the most important element of the system because it is what user can see and interact with (Galitz, 2007).. Thus, careful consideration should be put in an application's overall design.

3.1.1.2 Information Design

Communication is an essential aspect of the everyday lives of people. In businesses, it is important that they are able to communicate and get the point across to their clients and consumers. To do this, information must be designed in a way that can easily and efficiently be used and understood by people. This is where the art of information design comes in (Horn, 1999).

An integral part of interaction design, information design involves analysis and understanding of not only the content of the message or information, but how it will be presented as well (Pettersson, 2002). It is about how to clearly communicate or send across information which is a key aspect in designing a good interaction (Shedroff, 1999).

It is with proper organization that good information design is achieved, which can also lead to good interaction design (Shedroff, 1999). Information can be organized in several ways such as: alphabet, location, similarities, and more. Organizing these in such a way allows users to better understand and connect the individual piece of information (Shedroff, 1999). Additionally, Blair-Early & Zender (2008) advises the use of visualizations such as charts or graphs when presenting large information in order to make it easier for people to digest the

information.

Even though information design is its own discipline, it should still follow the common design principles so that the information can be presented neatly and effectively (Shedroff, 1999). The information, or the way it is presented, is still a part of the overall design of the system so similar to designing an interface, it is good practice that designers keep in mind the goal of the system when trying to formulate how the information will be presented.

Going back to the hierarchy principle, the most important information should be presented in a way that would easily grab a user's attention like making it larger than the others or placing it at a position that can easily be seen (Mandelbaum, 2014). This just goes to show that the design principles mentioned above encompass all kinds of design, from visual design to information design.

3.1.1.3 Input and Interaction

Without humans, there would be no technology (Commoner, 2014). Despite machines being more computationally powerful and efficient, there is still a need for human input. Because of the emergence of several machines for daily use, different input methods have been developed.

For computers, there was the mouse, keyboard, tablet, and trackball, with the mouse and keyboard being the most popular (Engelbart & English, 1968; MacKenzie et al., 1991). With the rise in popularity of mobile phones, users can now interact with touch screens through gesture interactions (Chen et al., 2014). As such, these methods of input will be further discussed in the next few chapters.

3.1.1.3.1 Traditional Computer Input

During the advent of the computer becoming a consumer product, the keyboard and mouse were present to accompany the Windows, Icons, Menus, and Pointing Devices (WIMP) interaction style (Porta, 2007). The keyboard and mouse were also innovations during their time. Both were used to test how new kinds of interaction between a human and computer could augment the overall output of a computer user (Engelbart & English, 1968).

Keyboards were mainly used to type commands into a computer device to fulfill a task the user would like to do (Karat et al., 1986). In the earlier times, keyboards were used alongside a command line interface, which made up most

of the interaction during the time where graphical user interfaces were yet to be known (Galitz, 2007). The keyboard is usually accompanied by a mouse where they are used in conjunction to perform tasks with computers (Engelbart & English, 1968). There have been events to also increase the comfort of using the keyboard to create less strain on the wrists or arms of the user (Grant, 1994).

The mouse is a commonly used tool for navigating a 2D space in a computer screen (Engelbart & English, 1968). A standard mouse has three (3) buttons that can be used to select or perform additional tasks on the computer. Other devices were also built to navigate a 2D space, but the mouse was found to be the best performer among these devices (Karat et al., 1986)

These traditional methods of input are commonly used in musical composition applications like Finale or Sibelius. In these applications, users usually input notes or modify the composition through a series of mouse clicks, while the more experienced users perform keyboard shortcuts (Knoder, 2017b,a). These methods can be efficient and easy, however the main drawback of these is that the keyboard and mouse are not easily portable, and are definitely not natural (Norman, 2010).

Touch-based interfaces were introduced at a later time to complement touch-screen devices and were meant to take advantage of the natural way humans interact. This was another step into finding out ways to improve the interaction between a human and a computer (Travis & Murano, 2014).

3.1.1.3.2 Gestural Interaction

The goal of Human-Computer Interaction is to allow humans to interact with machines or computers naturally and seamlessly (Hasan & Kareem, 2012). There is an increase in the use of gestures as a method of interaction in hand-held touch devices like tablets and smart phones and this is mainly attributed to gestures being considered as an an easy and natural method of interaction between humans and computers, compared to the use of a mouse or a stylus (Kurtenbach, 1996; Pavlovic et al., 1997; Hasan & Kareem, 2012).

As most gestures found in gesture interaction systems are based on semiotics, (Muntigl, 2004; Radford et al., 2009; Kammer et al., 2010), it is important to understand how gestures play a part in communication.

The work of Wesp et al. (2001) supports the idea that gestures help maintain spatial imagery. The use of gestures in communication helps in word retention because it builds up an idea through its perceivable signs (Krauss, 1998). This also applies to touch gestures since they build up perceivable actions that denote

specific functions which make them an intuitive and natural way of interaction (Rautaray & Agrawal, 2015).

In the study presented by Roth (2001), a gesture can be defined using four characteristics. The first characteristic of a gesture is that it moves from a resting position to a non-resting position and then back to a resting position. The second, *stroke*, defines the peak structure of the gesture where it denotes the function of the gesture movement. The third, *preparation* and *recovery* phase, is when the stroke goes back to its resting position. The fourth characteristic is that it is symmetrical.

Recent studies on gestures led to its taxonomy wherein they are are classified based on their functions or how they were modeled (Roth, 2001). There are four classifications of gestures according to McNeill (2007) (see Table 3.2):

Table 3.2: Classifications of Gestures

Gesture	Description
Beats or *Batons*	These are gestures that do not express opinions. They are simple gestures which include slight movement of hands such as tapping or flicking that express responsiveness.
Deictic gestures	These gestures attribute a contextual meaning such as pointing or extending of arms or hands. They are commonly used in referring to people or referring to locations.
Iconic gestures	Iconic gestures use hand movements which relate to concrete objects.
Metaphorical gestures	These are movements that relate to a certain concept and makes a visual reference to it. An example would be a professor telling the class about the destruction of the world while closing his hands slowly.

3.1.1.3.2 Touch Gestures

Touch gestures have been widely used in recent years because of its intuitive and convenient nature (Chen et al., 2014). Some commonly used gestures in touch screen devices are pinching, scrolling, and tapping. There are three different types of touch-based interactions. The first one, raw ink, records the freehand drawing of the user. Sketching applications such as Paint is an example that uses this type of touch interaction. The second one, direct manipulation, is used in manipulating the user interface and gives real-time feedback to the user such as zooming and scrolling. The third one, indirect command, treats gestural input as

a shortcut function in an application such as undo, redo, copy, and paste (Chen et al., 2014). In this study, direct manipulation will be used as the primary type of touch interaction for metacreation and other functionalities.

The work of Chen et al. (2014) presents a way to recognize multi-touch gestures and assign specific functions for each with the use of graph modeling. In order to do so, it is important first to discern the difference between direct manipulation and indirect command. Direct manipulation calculates the position between fingers and it gives continuous feedback to the user while performing a gesture. For example, the act of pinching to zoom out a text document requires the system to continuously change the view in sync with the pinching motion. Indirect command, on the other hand, is concerned with the motion's shape and trajectory right after when the user performed a gesture. It captures the shape of motion after it has been performed and executes the assigned function.

There are three types of information that needs to be considered when modeling a multi-touch gesture (Chen et al., 2014). The first one, spatial, stores the position of each gesture trail relative to each other. The second one, temporal, stores the order of trails and their corresponding duration. The third one, shape information, stores the shape of each trail which allows the system to distinguish the difference between a straight line swipe from a curve swipe.

Each gesture trail can be represented by three vertices in a graph (Chen et al., 2014). The first vertex, (V_b), represents the begin vertex. The second vertex, (V_s), represents the stroke vertex, and the third vertex, (V_e), represents the end vertex. Temporal and spatial information are represented using the following edges: E_s(x, y), E_{st}(x, y, t), and A_{st}(x, y, t), wherein subscripts s and st indicate whether an edge represents only spatial, or both spatial and temporal, respectively. Consider the following example in Figure 3.1:

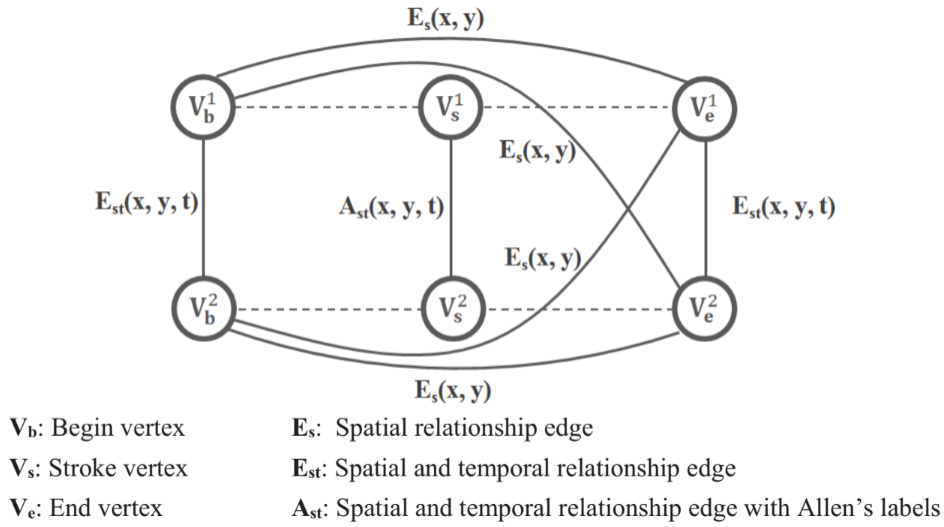

Figure 3.1: An example of a two strokes gesture graph (Chen et al., 2014).

In order to complete the information of the edges in Figure 3.1, Allen's relations that use discrete labels of time will be used as shown in Figure 3.2.

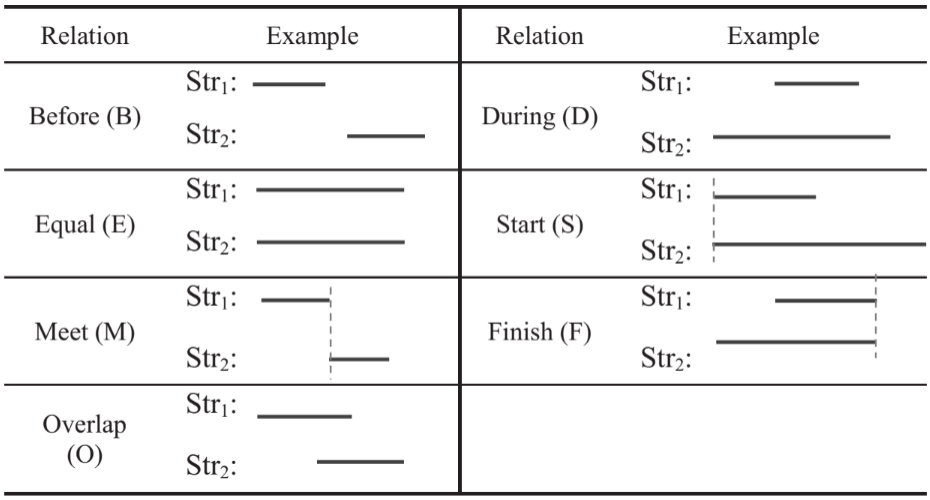

Figure 3.2: Examples of Allen's relations (Allen, 1983; Chen et al., 2014).

In general, labeling the edges results in two possible cases:

1. *Edges between stroke vertices (A_{st})*: This represents the measurement and relationship of strokes with regards to time, x-axis, and y-axis.

2. *Edges between extremity vertices (E_s and E_{st})*: This represents the touching and lifting of fingers in a touch screen and so it only encompasses *Equal* property for the time *(E_T)*, x-axis position *(E_X)*, and y-axis position *(E_Y)* in defining the edges between vertices.

Shown in Figure 3.3 is a flick gesture with its corresponding spatial relation and time duration. According to the relations of Allen (1983), the flick gesture has an *Equal_X (E_X)* and *Before_Y (B_Y)* labels which denote that the x-coordinates of both strokes are equal and the y-coordinate of the first stroke is above (Before) the y-coordinate of the second stroke. The time duration is labeled as *Equal_T (E_T)* to denote that the strokes were simultaneously performed. This results into edge A_{st} having an equation: A_{st}(x, y, t) = {E_X, B_Y, E_T}

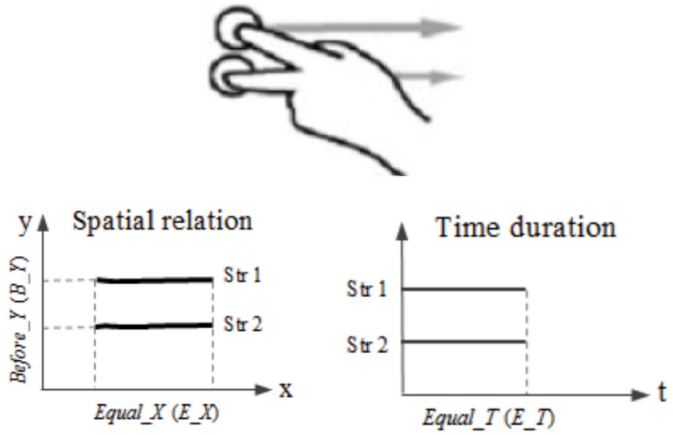

Figure 3.3: Flick gesture (Chen et al., 2014).

In the graph of the flick gesture, the label E_{st}={E_X, E_T} at the left and right side denote that the x-coordinates of both strokes are in the same region and they were performed at the same time. The label E_{st}={E_Y} on the top and bottom of the graph denote that both strokes were performed in the same region of the y-coordinate.

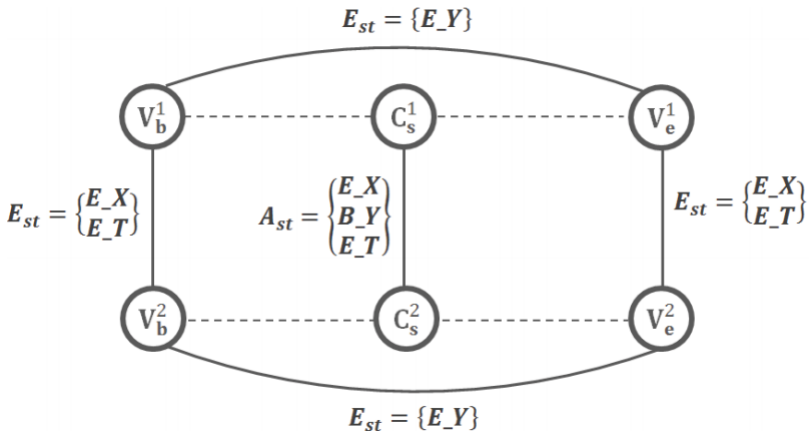

Figure 3.4: Flick graph (Chen et al., 2014).

In the application's selected platform, iOS, gestures are recognized through the `UIGestureRecognizer` class, implemented in the Swift programming language. This class is the one that detects gestures and passes them to other related objects that would act on that gesture. An important thing to note is that it does not perform anything related to the gestures, it only passes them. This was done in order to decouple the logic for recognizing the gesture and acting on it (Apple Inc., 2017).

By default, the `UIGestureRecognizer` class allows the detection of common gestures like taps, holds, pinchs, and more through its subclasses (Apple Inc., 2017). It communicates with `UIGestureRecognizerDelegate` classes to better control gestures in applications. The way gestures work in iOS is that the window or view reads the gestures and checks if any `UIGestureRecognizer` object recognizes the gesture performed. If one does, it would then pass control to that object (Apple Inc., 2017).

3.1.2 User Interface Software Technology

With the large, yet still growing number of applications and software available in the market today, it is becoming increasingly hard for developers to make their application stand out. One important factor that decides an application's fate in the market is design (Deka, 2016).

Application design is a process (Curtis et al., 1988; Humphrey, 1995) that takes time and effort not only from designers, but from developers and researchers as well. They must identify how best to design the interface in a way that users

would find intuitive and natural. This process involves research on the users' needs and behaviors and user experience validation through testing and heuristics (Deka, 2016).

Jakob Nielsen presents set of usability heuristics to follow for interface design (Nielsen & Molich, 1990; Nielsen, 1994, 2013):

- Visibility of system status
- Match between system and the real world
- User control and freedom
- Consistency and standards
- Error prevention
- Recognition rather than recall
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help users recognize, diagnose, and recover from errors
- Help and documentation

Visibility of system status denotes that users should always be informed about what's currently happening in the system. For example, the application is loading, then it should show that it is loading and not just a blank screen so the user won't think that the application froze.

Match between system and the real world means that systems should follow what is natural to the user. It should speak in the user's same language, using words and icons that the user can understand.

The *user control and freedom* heuristic simply means that the user should be able to easily undo or redo their state in case of a mistake.

Consistency and standards denotes that there should be consistency in the words used and the design. The system should not make the user think whether two buttons with the same name do different things.

Error prevention is important so that there will not be a need for error recovery. It is better to prevent users from committing errors rather than forcing them to recover from an error they made.

Recognition rather than recall should be kept in minding when designing a user interface to help reduce the cognitive load of users when using the system. Objects or icons should be clearly visible and easily retrievable to users.

Flexibility and efficiency of use means that the system should be more flexible to allow experienced users to speed up interaction or use shortcuts.

Systems should have *aesthetic and minimalist design* that only shows relevant information to the user. This is so that the screen clean and reduce the cognitive load of the user.

Systems should *help users recognize, diagnose, and recover from errors* through information that could easily be read and understood.

Help and documentation should also be included in the case that users would want to understand how to use the system better.

3.2 User Experience Evaluation

User experience plays a key role in software development as it can drastically increase a product's return of investment (Bellamy et al., 2011). Usability allows the discovery of problems sooner which may save a lot of budget costs and may improve sales. Producing applications that are usable is often difficult to pull off in actual practice because of its iterative process which uses human observation and analysis. This implies that there are biases in terms of testing and analysis (Bellamy et al., 2011). Over the years, multiple solutions for user experience evaluation have been proposed.

3.2.1 Quantitative Evaluation

3.2.1.1 KLM-GOMS

One example of a quantitative method for analyzing the usability of an application is by using *cognitive modeling*, an analytic technique that measures the motor operations performed by the user with regards to the brain (Sauro, 2009). An example of this is measuring how fast a user can navigate an interface or interact with it.

The most commonly used cognitive modeling technique is GOMS, which stands

for Goals, Operators, Methods, and Selection Rules. GOMS is a set of models for analyzing human-computer interaction and for removing user actions which are deemed useless or unnecessary (Card et al., 1983).

One such example of a GOMS technique is the Keystroke Level Modeling (KLM) technique (Sauro, 2011). It is commonly used by practitioners due to its simplicity. The KLM technique is used to estimate user actions such as pointing, clicking, typing, and thinking (Sauro, 2009). The technique simply counts the number of operations it would take a user to achieve a certain goal or navigate a system. From there, ideas can be gained from where the users experience pain points when using the system.

3.2.1.2 Fitts' Law

Another, and more descriptive method of analyzing usability is the Fitts' Law. Fitts' law is a predictive model for human movement that takes into account the distance from the target and the time it took the user to navigate to that target (MacKenzie, 1992).

Fitts' Law was conceived from information theory, specifically that of Shannon's Theorem 17 (Equation 3.1), which denotes the information capacity (C), or the amount of information that can be sent, using a signal power (S) within a specified bandwidth (B), while affected by noise (N) (MacKenzie, 1992).

$$C = Blog_2\frac{S+N}{N} \tag{3.1}$$

Although there was no direct mathematical relation of Fitts' Law to that of Shannon's Theorem, Fitts got the idea from the theorem of how information is processed through human channels when performing tasks (MacKenzie, 1992). He derived some of the elements of information theory to form what is now called Fitts' Law (Equation 3.2), where the index of performance (IP) can be derived by dividing the specified task's index of difficulty (ID by the movement time (MT) it took to perform the task.

$$IP = \frac{ID}{MT} \tag{3.2}$$

From this, Fitts Law can be used in domains like physical movement, or movement on a digital screen. Using Fitts Law, pain points can be identified, and

interface elements that affect a user's effectiveness when performing tasks can be found (MacKenzie, 1992).

Fitts Law will be heavily used in the study to help the researchers identify these pain points in the design. It will provide a numeric measure on the effectiveness of a user interface, making it easier for the researchers to document and compare results between previous versions of the interface. The measurement of Fitts Law can be made easier through CogTool (to be discussed in Section 3.2.1.3), which will also be used in this study.

3.2.1.3 CogTool

Although useful for quantitatively understanding the user experience of an application, both KLM and Fitts' Law require that the users being tested are observed in order to calculate the time and make estimates for both models. This poses a problem because it requires manual work and is repeated multiple times depending on the number of tests required. Having to manually make estimates for KLM and Fitts' Law can be tedious and prone to error (Sauro, 2009). However, IBM's solution for this problem is the implementation of an iterative and quantifiable method of testing and analyzing a product's usability through the software, *CogTool*.

CogTool is similar to other prototyping tools like InVision, but is more powerful for quantitative evaluation. Once an application's interface has been set in CogTool, with the simple click of a button a cognitive model for evaluation will be generated. CogTool eliminates the need to manually calculate KLM or Fitts Law, making it easier to quantitatively evaluate a user interface (Bellamy et al., 2011).

It uses *storyboards* that demonstrate the different tasks of a user (Bellamy et al., 2011). These storyboards serve as the user interface of the system that is being tested. Inside these storyboards are *frames* that serve as the different states of the UI. Each frame may have *widgets* which can be used by the user to interact with. The assigned action for each widget is called *transition*. A sample storyboard is shown in Figure 3.5. The rectangles inside the storyboard are the frames. The orange area inside the frame is the widget. The arrow between the frames is the transition.

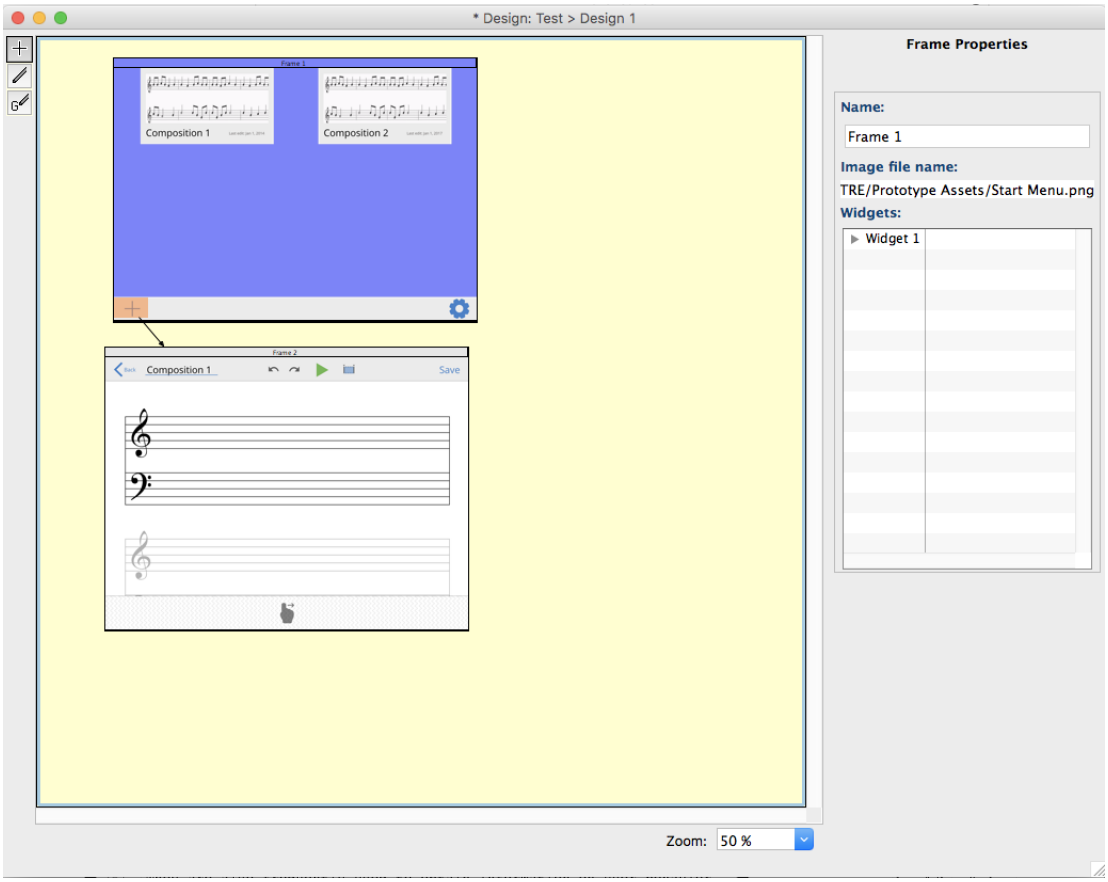

Figure 3.5: CogTool storyboard of application mockup.

Once a storyboard has been setup completely, a starting frame must be chosen for the system to start with (see Figure 3.6).

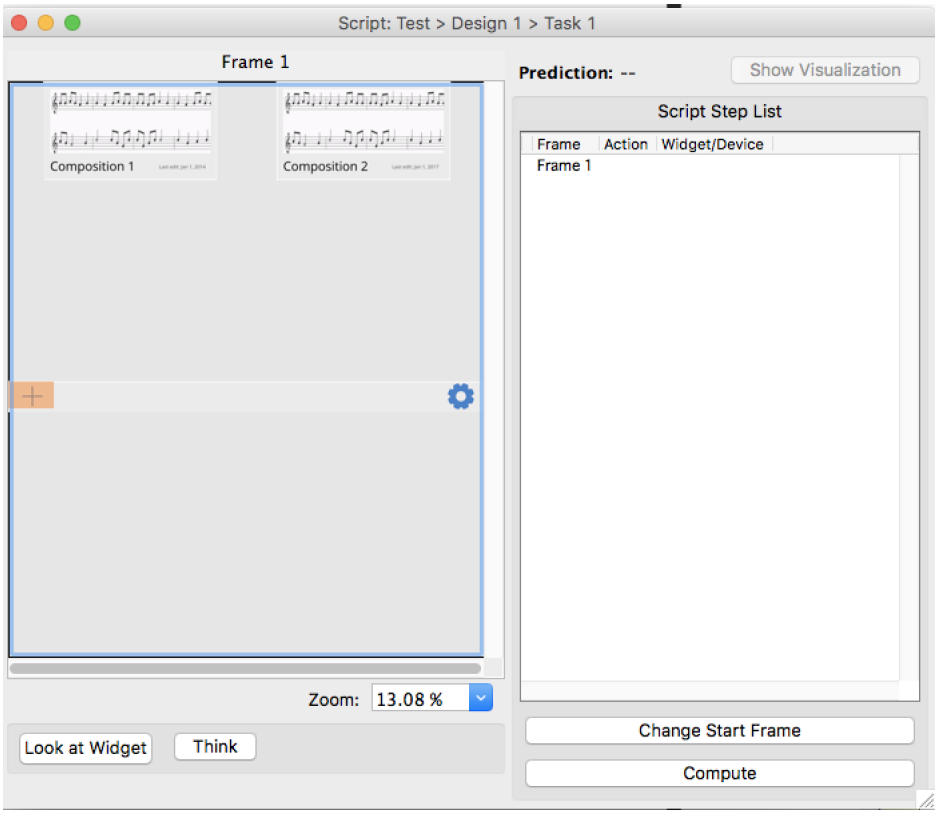

Figure 3.6: Choosing a start frame for a storyboard.

CogTool has the ability to run an automated user that would go through the whole storyboard. The steps that the automated user takes will also be shown in the script step list (see Figure 3.7). The list allows designers and analysts to easily identify what the user is doing and how long it takes the user to do it.

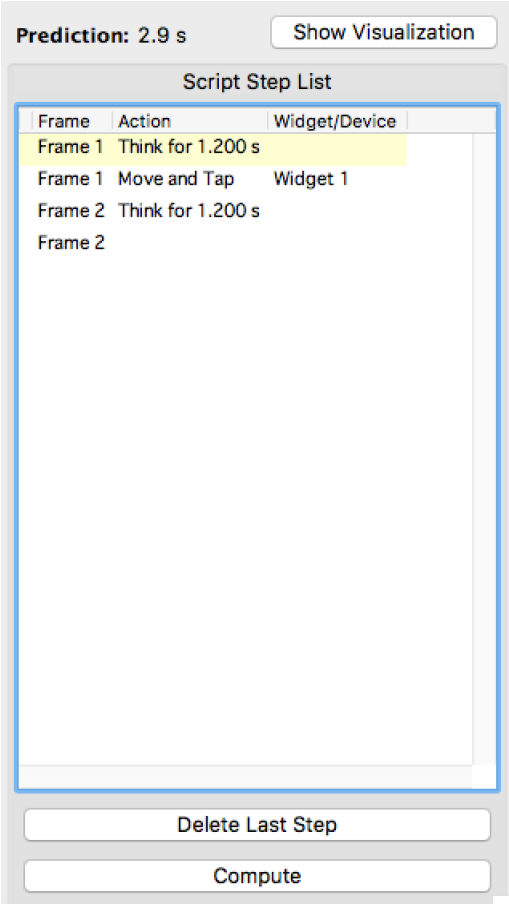

Figure 3.7: Script step list of CogTool.

When the user clicks on the compute button, CogTool computes for the estimate of the quantitative time of execution and visualization of timeline. The visualization shows the different tasks of the cognitive model on how it was able to derive the estimate (see Figure 3.8).

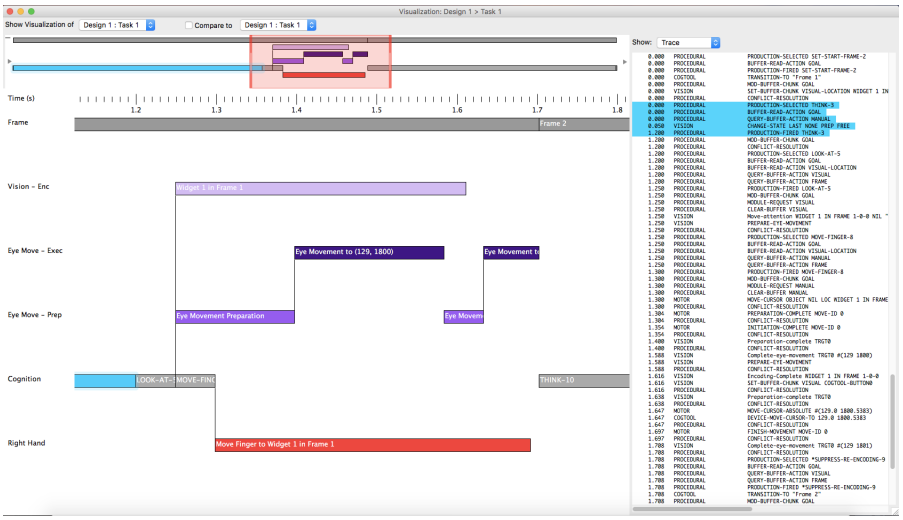

Figure 3.8: The highlighted part is the selected section of the user's action timeline.

The data presented by CogTool are useful for quickly evaluating the user experience of a system. It provides a quantitative model for analyzing the different tasks of a user and it can also be used for comparing different storyboards. This is helpful for designers to allow them to compare their design with competing designs (Bellamy et al., 2011).

3.2.1.4 Survey Instruments

Simple and easy to administer, surveys are one of the most commonly used methods of research (Goddard III & Villanova, 1996; Mathers et al., 2007). Surveys can be used to gather information on individuals, given a set of questions. They are also frequently used to obtain information on user behavior and attitude (Mathers et al., 2007). Additionally, surveys may provide answers that may be generalized, depending on the way it was administered (Newsted et al., 1998).

Surveys are good at finding out how many people do one thing or how many people think this way about something. However, they are limited in the sense that they fail to find out or answer why people do that one thing or why they think that way about something (Mathers et al., 2007). In cases like this, qualitative evaluation methods like focus group discussions are better at answering the "why" questions.

Given that surveys are used to gather information on respondents' attitudes, several methods of measuring these have been developed. The most common one,

however, is the Likert scale (McLeod, 2008). The Likert scale was developed by Rensis Likert in 1932 in order to provide a system for "measuring" the attitude of respondents.

Since then, the Likert scale has been commonly used as a five level scale measuring different levels of agreement or disagreement, frequency, importance, and likelihood on certain topics or concepts (McLeod, 2008). The advantage of this scale is that answers are not simply yes or no, and provide more insight on the attitude or behavior or the respondents (Likert, 1932; McLeod, 2008).

3.2.2 Qualitative Evaluation

3.2.2.1 User Acceptance Testing

The research will implement a mobile application for augmenting musical composition. To determine how effective the implementation of the system is, it will be tested on composers of varying skill. A method for getting feedback that would help identify issues in the application is User Acceptance Testing (UAT) (Davis & Venkatesh, 2004).

UAT can be used to measure the time it takes a user to finish a task, cognitive load, and their impression of the system after testing. These measures are seen in some experiments that performed user tests on similar systems like the touch-based system in the study of Findlater et al. (2012) where touch gestures and pop-up menus were the main features of the system. Music interaction research was also evaluated in the study of Brown et al. (2017) wherein it was found which factors were mostly used in testing, and how usability and aesthetics were the primary focus. These factors can be taken into consideration when planning the UAT for the system of the research.

Before the UAT, there can be different tasks that can be designed for the user to accomplish based on the needs of the user. In evaluating music interaction research, Brown et al. (2017) enumerated the following categories of tasks given to participants:

- Specific Task
- Open Exploration
- Guided Exploration

- Watch Performance
- Prepare and/or Give Performance
- Workshop
- In The World Use
- Other

These categories can be used during the UAT of the system. There are also other concepts or methodologies that the group will consider when testing the system such as cognitive load through the use of CogTool, which was discussed in Section 3.2.1.3. These concepts can either be observed or used in conjunction with the UAT or they are implemented differently and individual from the UAT.

3.2.2.1.1 Think-Aloud Protocol

A concept that can be used during the UAT is the Think-Aloud Protocol. The Think-Aloud Protocol encourages the user to share their thought while performing tasks during the experiment. This protocol can help the user open up during the experiment and voice out whatever they are thinking whether it is their frustration with the system or their strategy when trying to complete a task (Gomoll, 1996). During testing, the user is reminded to constantly voice out their thoughts while completing tasks so that observations can include both actions and user motives (Gomoll, 1996). Although there is no specific standard followed by researchers in executing the Think Aloud Protocol (McDonald & Petrie, 2013; Gill & Nonnecke, 2012), there are still studies that provide guidelines such as the classical work of Ericsson & Simon (1998) and augmented methods like the Explicit Think-Aloud (ETA) from the study of McDonald & Petrie (2013).

In the work of Ericsson & Simon (1998), the Think-Aloud Protocol was mainly used during observation to detect problems as they occur. In this case, the information provided by the user must be validated due it being the user's own ideas and opinions (Ericsson & Simon, 1998). There are 3 guidelines provided in the study of Ericsson & Simon (1998) to ensure the success of the Think-Aloud Protocol, these are:

- A neutral instruction that does not request specific types of information
- A practice session

- A neutral "keep talking" reminder given by the researcher

The Think-Aloud Protocol was used in the study of Yu & Voll (2011) to better understand how students solve mathematical induction problems. These problems can be related to music as mentioned in the study of Dörfler (2001), where music has a complex set of features that are usually continuous in form. A musical composition can then be seen as a series of solutions to a mathematical problem. From this, the thought process of a composer can be captured using the Think-Aloud Protocol similar to the study of Yu & Voll (2011).

3.2.2.1.2 Inspection

According to Holzinger (2005), a usable application should have the following characteristics: (1) *learnability*, meaning that the user should be able to learn how to use the system quickly; (2) *efficiency*, so that the users can perform their tasks in an effective manner; (3) *memorability*, which means it should be easy for users to use the system even after a long period of not using it (4) *low error rate*, where the system should prevent users from committing errors; and lastly, (5) *satisfaction*, meaning the system should be satisfying to use.

One common and effective method of identifying whether an application has these characteristics is through the use of usability inspection methods (Nielsen, 1994; Porter et al., 1996; Fagan, 1999; Laitenberger, 2002). These are a set of methods that can be used to evaluate user interfaces and are usually in the form of heuristic evaluations, walkthroughs, and even mathematical theories of evaluation.

Inspection can be conducted during the UAT. One example of an inspection method is where the testers will not be provided help during the experiment, and are asked to navigate the application by themselves. This provides the users a natural environment, granting the test the most realistic result from users (Gomoll, 1996).

Nielsen & Molich (1990) introduced the concept of heuristic evaluations, which is mentioned in Section 3.1.2. These are usually done by experts, evaluating whether the interface violates any of the said heuristics (Hollingsed & Novick, 2007). It was found in a later study that this method found the most number of problems in a user interface compared to other methods (Jeffries et al., 1991).

In cognitive walkthroughs, the user will be given a goal to perform in the system. The evaluator or the person administering the test will then observe the steps or the actions the user will take to achieve that goal (Hollingsed & Novick,

2007). These observations would help the developers and designers identify the pain points or the areas of confusion that the users experience within the application.

On the other hand, pluralistic usability walkthroughs allow inspection even without a fully developed user interface. Users are shown sketches of the screens on pieces of paper, where they will then identify the actions they would perform for each screen (Hollingsed & Novick, 2007). This allows an easy method of usability inspection even in the early stages of development.

These methods of inspection are widely used in development processes (Hollingsed & Novick, 2007). There is no best method of inspection, as each method has its own strengths and weaknesses. Heuristic evaluations are quick and easy, but they could fail at identifying user needs. Both cognitive walkthroughs and pluralistic usability walkthroughs fix this problem, but can be time-consuming and representative rather than comprehensive depending on how the inspection was administered (Hollingsed & Novick, 2007). However, it is common occurrence that these methods are used together to achieve better results that would benefit the study or the development goal.

3.3 Musical Composition and Its Systems

3.3.1 Musical Composition

3.3.1.1 The Composition Process

The theories involving the creative process, specifically musical composition, was explored in the study of Collins (2005). In this study, four (4) theories were mentioned, these were: (1) stage theory, (2) Gestalt theory, (3) emerging systems theory, and (4) information processing theory. It was noted, however, that the Gestalt theory was the most evident in application in the musical composition process.

The Gestalt theory divides the process into different sub-parts that would eventually combine in the end to form the composition. Its main essence is the restructuring phase of the process, also known as 'the flash of illumination' (Collins, 2005). This phase is described as an instance during the process where the composer experiences an unexpected solution to the problem or obstacle and uses it to progress further in completing the composition.

The earlier study of Bennett (1976) reflects the Gestalt theory. It was highlighted in the study that the composition process involved various stages (see Figure 1.1). A composition starts off with a germinal or initial idea for the theme or motif of the composition (Bennett, 1976; Collins, 2005). Once a motif is in mind, the composer moves on to the sketching phase. However, it is still possible for the composer to go back to the ideation stage, even if a draft has already been completed if the composer feels the need for a revision or an unexpected idea comes up. This is what Collins (2005) calls *backtracking*, where the composer goes back to previous stages of the process to reformulate ideas or segments of a draft.

The drafts go through a lot of refinement and revisions as long as the composer is not satisfied with the output (Bennett, 1976). Even with the idea in mind, some pieces of the draft might still be improved to better fit the composition. This phase is similar to that of the recurring restructuring stage defined in the study of Collins (2005). This restructuring involves reformulating pieces of the composition to achieve the final draft.

3.3.1.2 Concepts

3.3.1.2.1 Fundamentals of Music

Music cannot be taught without knowing the fundamentals first, namely: pitch and melody, rhythm, texture and harmony, color and timbre, and form or design (Rivadelo, 1986). By combining all of them, they form into what is known today as music (Willoughby, 1971).

Music is mainly formed by several notes or rests in a sequence. A consecutive progression of notes that are relative to each other is called the melody (Rivadelo, 1986). In Paul Hindemiths theory of melody, melody is considered as the element of a composition that truly reveals the personality of the composer (Cheng, 2016).

Pitch is defined as the highness or lowness of a tone or musical sound relative to the amount of vibrations per second (Rivadelo, 1986). Each pitch is represented by the first seven letters in the alphabet, which are A, B, C, D, E, F, and G (M. Miller, 2005). However, the distance of a pitch from each other can be described as a whole step and a half step or semitone. The distances can be easily derived from the keys of a piano or keyboard. Moving from one key to a next key is called half step or semitone respectively. A whole step on the other hand is described as moving from one key to another by skipping a key between them. Both can be used to traverse forward and backward by adding the word up or down after

the distance. Multiple steps can also be achieved by putting a number before the distance.

The pitch can also be represented using the Solfeggio Method, commonly known as the Do-Re-Mi. However, the same pitch representation may also be found on higher or lower tones which are divided by an octave, which is a fixed interval between pitches.

Another element of music is the duration, which is the amount of time a tone lasts (Rivadelo, 1986). The duration is shown in a composition through the use of different notes or rests that indicate the duration of a certain pitch (Rivadelo, 1986; Burrows, 1999). More of this will be discussed in Section 3.3.1.2.3.

In a composition, Rivadelo (1986) mentions that rhythm represents the pace of the musical flow through the common patterns that currently exist. It can be understood as the factor that makes musicians and composers aware of the beat or tempo of composition. It allows them to align their ideas in order to create music on top of the rhythm.

Rhythm also consists of 4 parts, which are: beat, accent, meter, rhythmic pattern, and phrase (Rivadelo, 1986). The beat is the basic time unit of the composition. The accent forms the theme of the rhythm, while the meter divides the beats into sections. The rhythmic pattern groups the beats to form a coherent pattern of sound. Finally, the phrase is considered as the musical thought, that is a part of the whole musical sentence.

3.3.1.2.2 Basic Music Theories

Knowledge of music theory can aid a musician to compose music without having to waste time thinking about succeeding notes or melodies. According to Meyer (1989), it can be likened to rules or guidelines that can change over time due to changes in cultural style. The theories in music also utilize some of the fundamentals of music to create these guidelines. It assembles the basic fundamentals into a harmonic structure that characterizes and builds music (Meyer, 1989).

There are multiple concepts in music theory that can start off a musician to compose his or her own music piece. Although music theory can serve as a guide for composition, it is not an all-in-one solution. Composers will still have to think creatively to identify which theories are applicable or could be applied in the current composition (Meyer, 1989). Some form of trial-and-error would still be present in this process.

Tetrachords are one of the most frequenly used concepts in music theory (Spencer, 1996). They are four succeeding notes that form a structure that dictates the perfect fourth of the first note. Spencer (1996) stated 4 kinds of tetrachords: major, minor, natural, and harmonic. They are differentiated by the number of semitones being skipped relevant to the first note. Major follows the pattern 2-2-1 semitones, minor goes 2-1-2, natural uses 1-2-2, and lastly, harmonic follows 1-3-1.

Scales can then be derived from tetrachords (Spencer, 1996). Combining two (2) tetrachords and adding a whole step or a tone in the middle of the two tetrachords would result in a scale. For example, two major tetrachords (2-2-1) with a whole step in the middle would result in: 2-2-1-(2)-2-2-1, which is actually the major scale.

There are certain harmonic intervals that composers can utilize to build a good musical composition (Spencer, 1996). These are called melodic intervals. They can be described as various note sequences that composers can use in their composition to build the motif or the whole composition as a whole (Spencer, 1996).

There are two characters used to distinguish intervals from each other. The first character denotes the quality of the interval (Spencer, 1996). This can be any of the letters: M, P, m, d, A. M is described as a major interval, P as a perfect interval, m as a minor interval, d as diminished, and A as augmented. Each of these intervals give off different sounds for the note sequences they produce (Spencer, 1996).

The second character is about the quantity of an interval. It is a number which identifies the length of the interval or the distance of one note from another (Spencer, 1996). However, not all interval qualities contain all interval lengths. Some may contain the same numbers from another melodic interval but they are different in terms of how many semitones are skipped in order to dictate the next note.

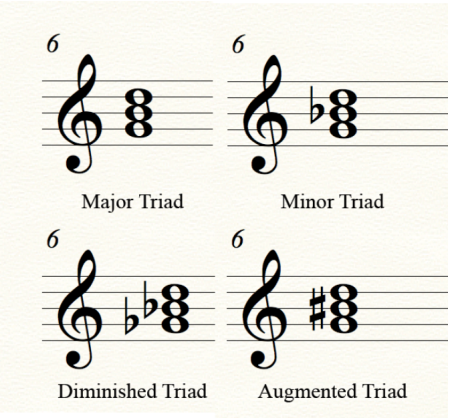

Figure 3.9: Various triads (Spencer, 1996).

The basic chords in a musical composition are defined by triads. It is basically three notes that are played simultaneously. Like tetrachords, there are four types of triads: major, minor, augmented, and diminished (Spencer, 1996). Figure 3.9 show the various triads in a musical composition. The lowest note in the triad is called the root, the middle note being the third, and the highest note is called the fifth. However, triads have different qualities in different scales, specifically the major scale, natural minor scale, melodic minor scale, and the harmonic minor scale. In these scales, triads are classified into scale degrees which are like roman numerals which are both lower-case and upper-case. These scale degrees are often used in defining different chord progressions.

3.3.1.2.3 Musical Notation

The modern notation system in music today, specifically western musical notation, was derived from places where the western culture continued to thrive (Read, 1969). However, the practice of writing musical notation did not originate from western culture (Read, 1969). Musicologists discovered that pre-Christian Greeks have already been using musical notation, and have developed four (4) unique systems for writing them (Read, 1969). Nonetheless, the modern musical notation system holds some similarity to the system developed in the past.

According to Read (1969), musical notation is known as a method for visually representing musical sound. It is what musicians use to graphically display their musical ideas and can be read by other musicians as well. In relation to writing and reading any character in any language, it also requires understanding of the symbols used in music notation in order to be literate in music (Read, 1969).

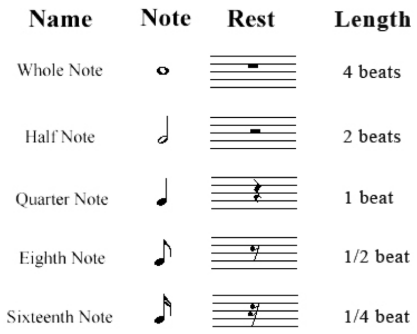

Figure 3.10: Basic notes and rests used in modern musical notation (Riso, 2016).

The basic symbols in musical notation are the notes and rests seen in Figure 3.10. These are found throughout the staff, which is the set of lines and spaces that represent the different levels of pitches. Beams are the black bars that are used to group consecutive notes that are less than a quarter note in order to improve the visualization of the composition itself (Spencer, 1996). However, the pitches that represent each space or line in the staff is different based on the clef used at the start. The G-Clef or treble clef and the F-Clef or bass clef (see Figure 3.11) change the pitch representations on each line and space on the staff. When the G-Clef is used at the start of the staff, each line starting from the bottom represents the pitches E-G-B-D-F and spaces represent the pitches F-A-C-E. On the other hand, when the F-Clef is used at the start of the staff, each line starting from the bottom represents pitches G-B-D-F-A and the spaces represents the pitches A-C-E-G.

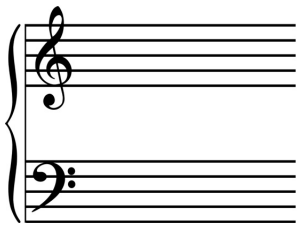

Figure 3.11: The G-Clef and F-Clef (Stamp, 2013).

There is only one element in musical notation that indicates the distinct rhythm throughout the composition, and that is the time signature (Rivadelo, 1986). There are two numbers in the time signature, (see Figure 3.12) called the numerator and the denominator. When writing the time signature, the numerator takes up the upper two spaces of the staff while the denominator takes up the lower two spaces of the staff (Read, 1969). Each part of the time signature defines the notes to be used in the musical composition. The numerator or the

upper number denotes the amount of beats in one bar or measure while the denominator or the lower number indicates the type of note that would equal to one beat (Read, 1969; Rivadelo, 1986; Burrows, 1999).

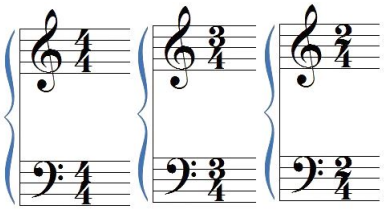

Figure 3.12: Various time signatures (Rose, 2014).

A series of notes in a musical sheet is divided by bar lines (see Figure 3.13). However, there are two kinds of bar lines in musical pieces. The first is the simple bar line which is one vertical line that divides a series of notes based on the time signature. The second type of bar line is the double bar line which is made up of two vertical lines. It specifies the end of a part in a music composition (Read, 1969). This is where composers can change the time signature and key signatures of the current staff. However there are variations to the double bar. The first is the one that signifies the end of the composition which is the "period" form (Read, 1969) or final bar line. This kind of bar line is found at the end of the composition. The other two variations are double bars that work together, they are the start repeat and the end repeat. It encloses a number of bars between the start and end repeat which sums up the number of beats as a whole that would satisfy the time signature declared before the start repeat bar.

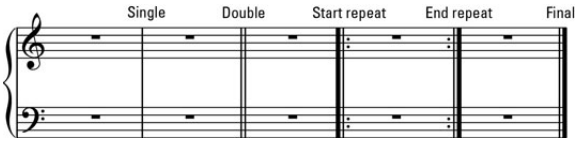

Figure 3.13: The different bar lines (John Wiley & Sons Inc., 2017).

There are certain annotations that are used to dictate the strength of the sound of notes, these are called dynamics. Shown in Figure 3.14 are several levels of dynamics that can be used in musical compositions. These markings can be usually found above or below the notes that are affected by the dynamic marking (Read, 1969). However, there are symbols that also complement dynamic markings, which are called dynamic signs.

Crescendo and diminuendo or decrescendo are the only dynamic signs. Both signs are used when a succession of notes is gradually increasing or decreasing to a

certain dynamic. The example shown in Figure 3.15 exhibits the use of both signs. Basically, crescendo is when the loudness increases up to a certain dynamic and decrescendo is when the loudness decreases down to a certain dynamic. In Figure 3.15, crescendo is used when the loudness of the notes increased from mezzo forte to forte. Decrescendo is used when the loudness of the notes descended from forte to mezzo forte.

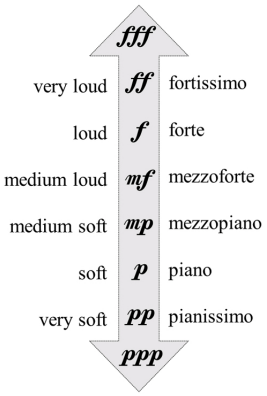

Figure 3.14: The various dynamic markings (Peter, 2016).

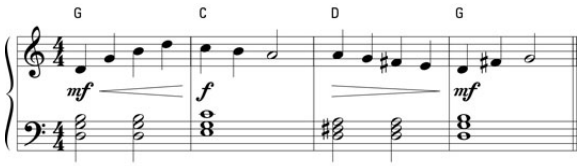

Figure 3.15: Dynamic markings and signs (John Wiley & Sons Inc., 2016).

3.3.1.2.4 Circle of Fifths

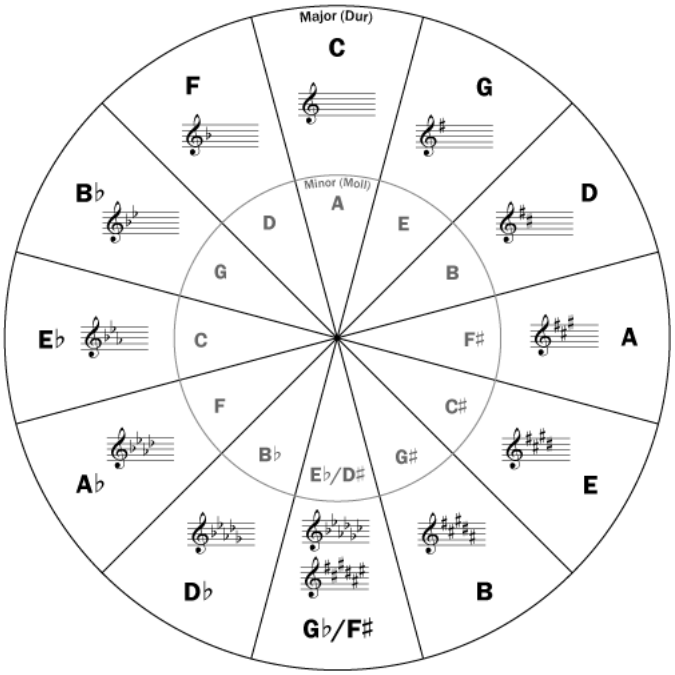

Figure 3.16: The circle of fifths (Gleaves, 2015).

The circle of fifths (see Figure 3.16) was derived from the the Pythagoras Circle made by famous Greek philosopher Pythagoras (Jensen, 1992). It was later expanded in the work of Nikolai Diletskii on amplification and Johann Heinichen's work on modulation (Jensen, 1992). It is also now commonly used today as a basis for teaching techniques in composing and playing music. Some of these techniques include modulation and chord progressions.

It can be compared to a 12-hour round clock with the numbers replaced by the names of different pitches (see Figure 3.16). All of the succeeding pitch representations going clockwise starting from C are derived from the fifth of the previous pitch (Jensen, 1992). For example, starting from C, the fifth of C would be G, and the fifth of G would be D and so on. It would revolve back to C when it reaches F. The inner circle is a copy of the outer circle only moved counter-clockwise twice.

Although it may look like a clock, counting with the circle of fifths is different. It starts at 0 from the top or north and both sides (clockwise and counter-clockwise) count from 1 and end at 6 at 6 o'clock. Those numbers denote the key signatures that are present in a major or minor scale in a musical composition (Jensen, 1992). Key signatures are derived from the circle of fifths through the

counting. The sharps in the key signature begin counting clockwise at F and flats begin counting counter-clockwise at B (Clough & Myerson, 1986).

From the circle of fifths, major triads can be derived (Jensen, 1992; Spencer, 1996). This can be done by choosing a root note from the outer circle, then getting its third and fifth (Jensen, 1992). The third can be found by traveling four (4) semitones, or two (2) tones clockwise from root note. The fifth can be easily found by looking at the note next to the root note, clockwise. For example, if C is chosen as the root note, the third, which is four (4) semitones clockwise from C, would E, and the fifth would be G, which is just next to C.

Minor triads can also be found in the circle of fifths but through a different way compared to deriving major triads (Jensen, 1992). Again, the root is chosen from the outer circle. The third is found by moving three semitones counter-clockwise from the root. The fifth is taken similarly to how it is taken in major triads, just get the note next to the root note, clockwise. Similar to the previous example, C will be chosen as the root note. In this case, its third would be E minor, which is three semitones counter-clockwise to C. The fifth in this triad would be G.

The circle of fifths can also be used in modulation from different scales (Clough & Myerson, 1986; Jensen, 1992). One event where modulation would be needed is when a composition needs to fit the singer's vocal range and key. Using the key signatures derived from the circle of fifths, composers can easily transpose the key signatures of composition.

3.3.1.3 Musical Metacreation

For a long time, artificial intelligence has mainly been used in pursuing problem solving and mathematical and analytical applications (Steels, 1993). However, with its growing possibilities, artificial intelligence has seen applications in several domains such as speech, image recognition, and even creative endeavors (Russell et al., 1995). Computational creativity, is a form of artificial intelligence with a focus on understanding and modeling the creative human brain (Pasquier et al., 2017).

The subfield of computational creativity, musical metacreation, is mainly concerned with the study of endowing or giving machines the capacity to perform and undergo creative musical tasks like composition or interpretation Musical metacreation can be done through several algorithms or techniques. In the study of Tokui et al. (2000), Birchfield (2003), and M. Kikuchi & Osana (2014), genetic algorithms were used to metacreate melodies and rhythms, while Geis & Middendorf (2008) used an Ant Colony Optimization algorithm to account for the variances

that composers add to their composition.

However, given the limited time allowed in this study, the researchers opted to use a rule-based algorithm in favor of the other algorithms.

3.3.1.3.1 Rule-based Algorithms

The goal of the study of Bozhanov (2014) was to improve on the limitations and alleviate problems present in other algorithms such as the lack of a user interface, the composition's pleasantness, but more specifically, the lack of variation. The result of this study was Computoser, a probabilistic rule-based musical metacreation algorithm. When composing music, Computoser follows a set of rules coming from music theory. Additionally, since the algorithm is a hybrid between probability-driven and rule-based, Computoser attaches probabilities to these rules, for example in the set of notes that can be generated (Table 3.3), ensuring that there is variation between the compositions (Bozhanov, 2014).

Table 3.3: Computoser Note Probability (Bozhanov, 2014)

Note	Probability
Sixteenth	10%
Eighth	31%
Quarter	40%
Dotted Quarter	7%
Half	9%
Whole	3%

The core of the Computoser algorithm comes from its rules (Bozhanov, 2014). The algorithm has rules on the following:

- Structure
- Rhythm
- Repetition
- Variations
- Dissonance and syncopation
- Endings

- Effects

The most interesting of these, is the variation. Variations prevent music from becoming boring, yet still allow some form of repetition (Kivy, 1993; Bozhanov, 2014). Because the goal of the study was to improve on the lack of variation in other algorithms, Computoser allowed several kinds of variations in its compositions whenever needed, which on average was found to be after every 2 measures (Bozhanov, 2014).

The variations present in Computoser that will also be used in this study are: transposition, inversion, and retrograding. Transposition is done by simply moving a set of notes to a different scale, either lower or higher (Owens, 1998). Inversion is done by reversing the notes on a horizontal axis or in layman's terms, turning them upside down based on a scale while keeping their intervals (Owens, 1998). On the other hand, retrograding is done by reversing the notes on a vertical axis, or basically playing them backwards (Owens, 1998). These variations will be used in the proposed system to allow composers to alter a set of notes but maintain some similarity or resemblance to the original set of notes.

Another rule-based musical metacreation algorithm is the one developed in the study of (Schulze & van der Merwe, 2011). The result of the study was SuperWillow, a system whose aim was to generate music that were pleasurable to the human ear. The system used probabilistic automata and Markov chains built by analyzing music data to generate its music.

SuperWillow was built by analyzing music data in the form of a MusicXML file (Schulze & van der Merwe, 2011). Because the data was in this format, it was easy for them to retrieve information like the tempo, scale, and time signature. The analyzed data was used to generate analysis objects: Markov chains which can be used to model a sequence of tasks or events through states. Markov chains satisfy the Markov assumption, which states that:

$$P(q_t|q_{t-1}, q_{t-2}, ..., q_1) = P(q_t|q_{t-1}) \tag{3.3}$$

$q_1, ..., q_t$ are a set of states, while t is the time. What this algorithm means is that the transition of each state must not depend on previous states, and only depend on the current state (Schulze & van der Merwe, 2011).

These analysis objects are then used when generating music to imitate the styles of the music that was used as input (Schulze & van der Merwe, 2011). Musical styles can be generated randomly or chosen by the user from the list

available from the input. Following this, other important musical information is selected randomly, again from the set of input music. The specific Markov chains for generating the chord progression and rhythm are then used to generate the set of notes that will be played (Schulze & van der Merwe, 2011).

The music generated by SuperWillow were evaluated through a partial Turing test by making 263 respondents, composed of professors and students from Stellenbosch University, listen to a mix of both human composed music and the generated music by SuperWillow (Schulze & van der Merwe, 2011). The results of the survey were promising, showing that 38% of the respondents thought that the music composed by SuperWillow was composed by a human. Given this, the researchers are still aiming to add to the capabilities of SuperWillow by improving how it analyzes the training data Schulze & van der Merwe (2011). The system can be found at `http://superwillow.sourceforge.net`.

3.3.2 Existing Systems

3.3.2.1 komp

"*komp*" is a musical composition application for the iOS platform developed by Semitone (D. MacDonald, 2017). komp promotes the natural way of musical notation, given that its method of input is by drawing notes on a digital music sheet. The drawn note would then be recognized by the application's built-in image recognition model, and rendered as a digital note. However, the main drawback of this method is that the faulty image recognition is an annoyance or obstruction to composition (D. MacDonald, 2017)

Drawing notes into the musical sheet may seem natural but it can be obtrusive in some instances (D. MacDonald, 2017). Sometimes the recognition of the location of the drawing may bug out and place notes in places that the user did not desire. The recognized note from the drawing may also be wrong during some instances which results in wasted time erasing the generated note (D. MacDonald, 2017).

This application presents a solution to this problem by allowing users to use their built-in learning tool. The tool requests users to draw a specified note multiple times to better learn how the user draws the note and adjust the image recognition model to suit the user's style (D. MacDonald, 2017).

Although this looks like a good solution, the application still has limitations. According to D. MacDonald (2017), the handwriting feature cannot detect six-

teenth rests or notes. This could still be remedied using the built-in learning tool but the user would have to create their own way of writing the sixteenth note or rest in order for it to be possible. This solution would then remove the naturalness of the input method because composers would have to write sixteenth notes or rests differently from it is actually written (D. MacDonald, 2017).

3.3.2.2 Finale

Finale is a musical notation software that is available for both the Windows and macOS platforms. It is widely known for its large set of features or functions such as keyboard shortcuts (Knoder, 2017a) and the speedy entry tool that can make musical notation faster. It also uses MusicXML files to store and share compositions with other composers, or for use in other musical notation software that support MusicXML (Otter, 2017).

Finale allows composers to drag notes or rests to the music sheet. This presents an easy method of adding notes to the sheet (Otter, 2017). However for expert users, Finale allows the use of keyboard shortcuts to speed up musical notation. Notes are assigned to specific number key on the keyboard, so adding a note to the music sheet can be as simple as pressing a key (Knoder, 2017a). This method saves time from having to find the right note or rest and dragging it to the music sheet. Although it might not feel natural for musical composers that are used to writing musical notes on paper, it is faster (Knoder, 2017a).

Finale's speedy entry tool works hand-in-hand with the application's keyboard shortcuts. The tool changes the default indicator found in the staff and replaces it with a box that has a small black block indicating which space or line is currently selected. The block can be moved up or down to change the selected space or line by pressing the up and down arrow keys respectively. When combined with the shortcut for note input, the application allows composers to write compositions faster (Knoder, 2017a).

3.3.2.3 Sibelius

Sibelius is another musical notation software that is also available for both Windows and macOS machines. However, compared to Finale, Sibelius offers less keyboard shortcuts and an overloaded interface with too many options (Hess, 2008). Despite this, Sibelius still proves to be a viable option for musical composition on computers (Knoder, 2017b).

Sibelius requires users to input notes by selecting a note from the floating menu, and clicking on the desired location in the staff. Unlike Finale, Sibelius does not support keyboard shortcuts for faster note input (Knoder, 2017b). This task proves to be too heavy for longer compositions, and is a burden for expert users (Hess, 2008). However, this method is helpful for beginner composers because of its simplicity.

An interface that presents too many options or features may be good for people that like to explore and learn the software, but it may be obtrusive for some users that wish to learn the software with minimal exploration (Galitz, 2007). However, Sibelius presents an organized set of options by separating them into sorted tabs. This makes it easier for users to find what they are looking for (Hess, 2008). The menus are also designed in a way that it looks like the menus present in Microsoft Office 2007 and recent versions.

Chapter 4

Research Framework

This chapter discusses the features of the proposed solution, the system objectives, frameworks used and specifications. The research experiment design, user stories, and use cases are also described in this section.

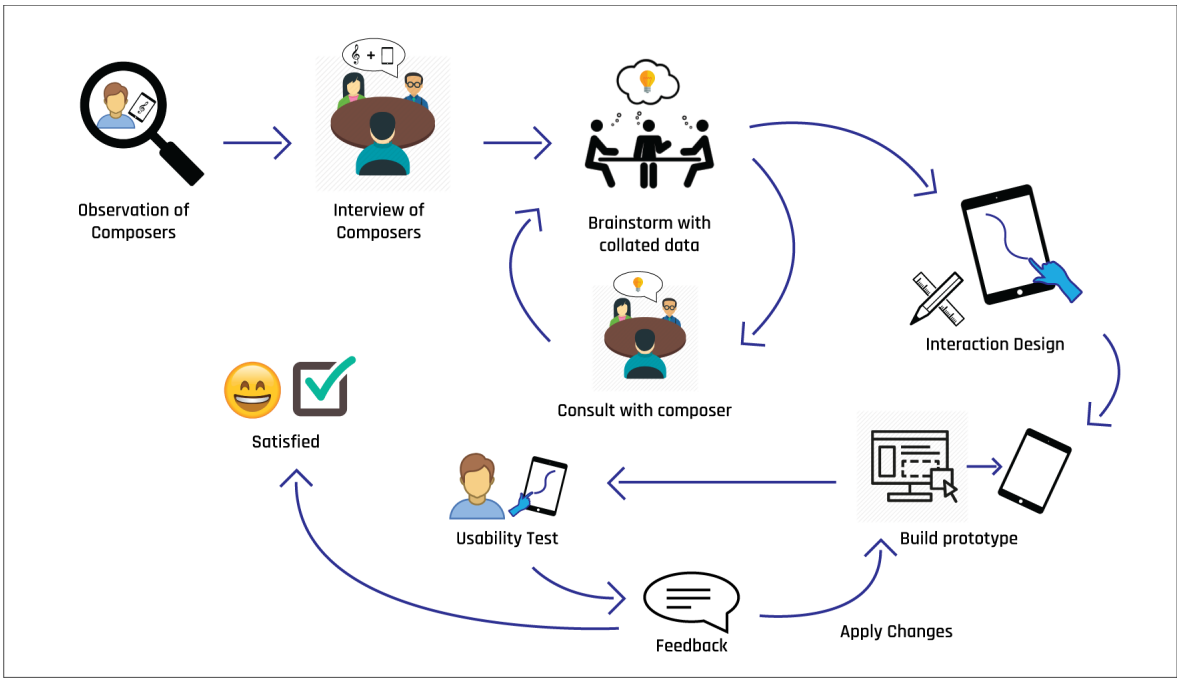

Figure 4.1: The research framework for Flow.

This study was designed to be repetitive and iterative, to allow for continuous improvement of the system. User research and observation are important steps

to understanding the behavior of users and figuring out their pain points. This is followed by interviews with composers to confirm and distill the information gathered from the research and observations. The points that were given by the composers would then be used in the brainstorming phase to determine a suitable solution. The team will then consult with the researchers about the proposed solution to gather their insights about it. The brainstorming would then be repeated to improve on the solution followed by another consultation. This would be repeated until the team is satisfied with the solution.

The next phase would involve designing the interaction of the proposed solution. This would be followed by the prototype phase where a prototype would be built, tested, and improved repeatedly. Usability tests would be performed with different prototypes to create a good user experience. This phase would be repeated multiple times until the researchers are satisfied with the results of the tests.

4.1 System Description

4.1.1 System Overview

The system will integrate music theory and user experience in an interface for musical composition. The composition process of composers will be observed and analyzed so that the system can be built to fit this process. It will be built for the iOS, but will be optimized for the iPad. It will provide users an interface to create/edit, and view compositions. The user can interact with the system through a set of gestures that have been defined. Some gestures are intended for adding/editing/deleting notes while others are for musical metacreation. Finally, the system will allow users to save their composition, which would be saved using a MusicXML file unknown to the users.

4.1.2 System Objectives

4.1.2.1 General Objective

To provide an interface that allows composers to perform basic and advanced musical composition tasks via gesture interactions.

4.1.2.2 Specific Objectives

Specifically, the system will allow users to do the following:

1. View, create, edit, and delete their own compositions
2. Play their compositions
3. Set the clef, key signature, and time signature of their compositions
4. Add, edit, and delete notes through tap and hold gestures
5. Highlight a set of notes
6. Play the highlighted set of notes
7. View information on the pitch and octave of the notes
8. Make a horizontal gesture to the right to generate a sequence of notes based on previous notes and coded rules
9. Make a gesture to the right that goes up to generate a sequence of notes with an increasing pitch
10. Make a gesture to the right that goes down to generate a sequence of notes with an decreasing pitch
11. Make a swirling gesture to the right to generate a repeating sequence of notes
12. While gesturing, decrease the generated gestures by going to the left
13. Choose to confirm or cancel a sequence of notes generated by a gesture
14. Save the composition to be stored in local storage

4.1.3 Scope and Limitations of the System

Given that the system is on a mobile platform, it would be limited in ability compared to that of desktop applications. This system aims to be a sketching application for composers that are on the go, or do not have their computers available. It will not contend with full-blown desktop musical composition applications like Finale or Sibelius.

Musical composition can be done for several kinds of instruments. However, the way music is composed varies from one instrument to another. With that said, the system will only support compositions for the piano. The system will also limit the musical notation symbols that can be used. The ones available for use are:

- Sixty-fourth note to whole note
- Sixty-fourth rest to whole rest
- Accidentals (sharp, flat, double-sharp)

Composers will also need to save their compositions in cases where they could not finish entirely and want to go back to it. A composition also has several elements which would be hard to model in databases. Thus, MusicXML will be used as the file format for saving data. This also makes it easy to transfer work from the system to other musical composition applications.

Gesture interactions are the main method of interaction within the system. Some gestures like tapping on the line need to be accurate and precise. To improve this precision, bigger screens will be needed. The iPad will be the best for this due to it having a large screen, yet still being portable. The testing will also be done on the iPad only.

Finally, the system's musical metacreation will need to have a model for generating the succeeding notes. To make it lightweight, the system will make use of a rule-based model similar to that of Computoser (Bozhanov, 2014) and SuperWillow (Schulze & van der Merwe, 2011). The model will take into account music theory for its rules.

4.1.4 Data Design

Given that musical compositions have a lot of elements which need to be stored as data, using relational databases for storage would prove to be inefficient and unintuitive (Hristidis et al., 2003). That is why the system would represent data through the use of MusicXML, which was also used in the SuperWillow system found in the study of Schulze & van der Merwe.

MusicXML is a method of storing and representing digital sheet music through XML (MakeMusic, 2017). Because it is an Extensible Markup Language (XML), it follows a specific format that defines a logical structure (Bray et al., 1997), which

in this case is a musical composition. The advantage of the MusicXML format is that it is used in several musical composition applications and can easily be shared between these applications (MakeMusic, 2017). Also, it can represent the most complicated aspects of musical notation like repeats, slurs, and more.

Shown in Table 4.1 are the some of the commonly used elements and their respective descriptions in MusicXML.

Table 4.1: Commonly Used MusicXML Elements

Element	Description
`<score-partwise>`	Defines that the composition is divided into several parts, and these parts can have multiple measures.
`<score-timewise>`	An alternative to the `<score-timewise>`, it defines the composition to have multiple measures where the measures can have many parts.
`<part-list>`	Lists the parts of the composition.
`<score-part>`	To be used as a child of `<part-list>`, this element adds a new part to the composition. Commonly supplied with the `id` attribute.
`<part-name>`	A child of the `<score-part>` element, specifies the name of its parent part.
`<attributes>`	Lists essential information in the composition such as the key, time signature, and clef.
`<divisions>`	Used in the production of sound. It works with the `<duration>` element to tell how many divisions per quarter note equal to the duration indicated.
`<key>`	Denotes which key signature the composition is in and contains the `<fifths>` element.
`<fifths>`	This element is derived from the circle of fifths and says how many sharps or flats the composition has.
`<time>`	The time element contains information about the time signature which can be set using the `<beats>` and `<beat-type>` tags.
`<beats>`	The numerator of the time signature.
`<beat-type>`	The denominator of the time signature.
`<clef>`	Tells the clef to be used in the composition through the `<sign>` and `<line>` tags.
`<sign>`	Specifies the sign to be used for the clef.
`<line>`	Specifies which line the set sign will start.
`<note>`	Contains information inside that is needed to define a single note.

`<pitch>`	Located inside the `<note>` element, it contains the `<step>` and `<octave>` elements that would indicate where the note would be placed.
`<step>`	Indicates the pitch step. Must always be supplied in the `<pitch>` element.
`<octave>`	Indicates the octave of the pitch. Also required.
`<alter>`	An optional element in the `<pitch>` that indicates if the note has a sharp or flat.
`<duration>`	Also an element inside the `<note>`, it works with the `<division>` element to denote what kind of note or sound it would play.
`<type>`	Mainly serves to indicate how the note will be displayed or notated.

Whenever a user creates a composition and saves it in the application, a corresponding MusicXML will be generated and saved in the device's local storage. The MusicXML will be used to load the composition again, in case the user wants to edit or view it.

4.1.5 System Framework

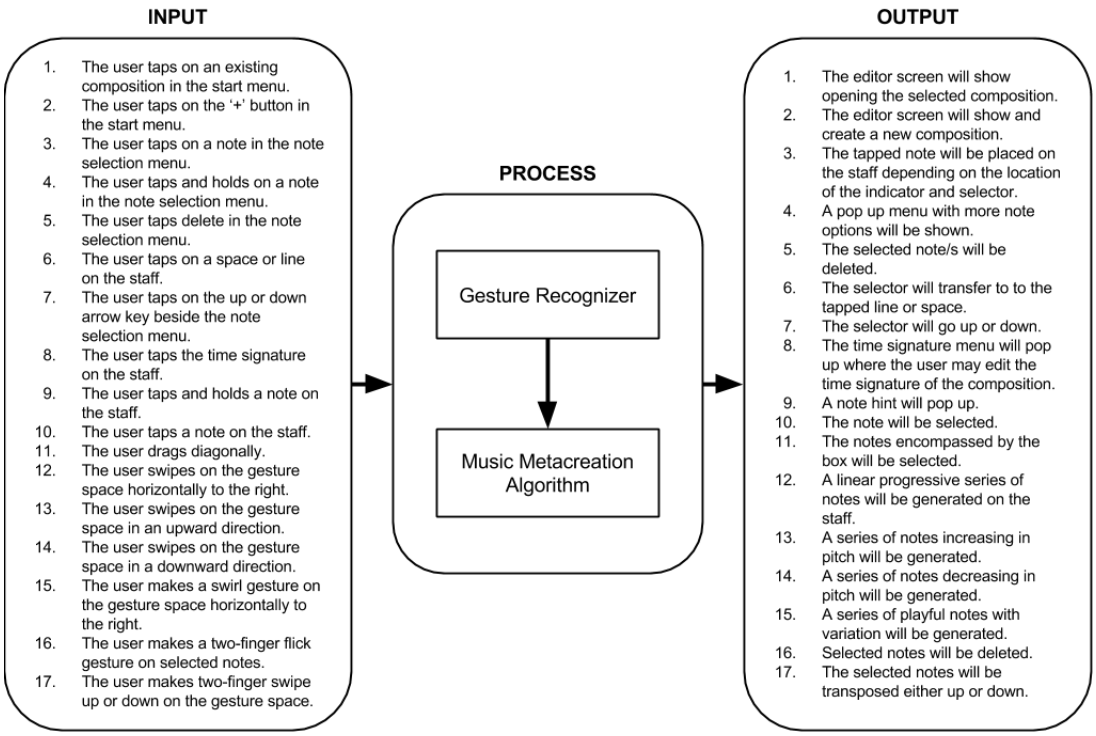

Figure 4.2: The system framework for Flow.

The system framework, shown in Figure 4.2, illustrates an overview of how Flow works. Input is relatively simple, mainly in the form of gestures. Users can tap, swipe, hold, or drag on specific objects. The application's gesture recognizer would then analyze the gesture performed by the user. The system would then output or perform the specific action assigned to each gesture. However, if the gesture is tied to musical metacreation, the application would output a set of notes based on its built-in algorithm.

4.1.6 System Walkthrough

1. Start Menu - The user may create a new composition or edit existing ones on this screen. After opening a composition or creating a new one, the editor screen will be shown.

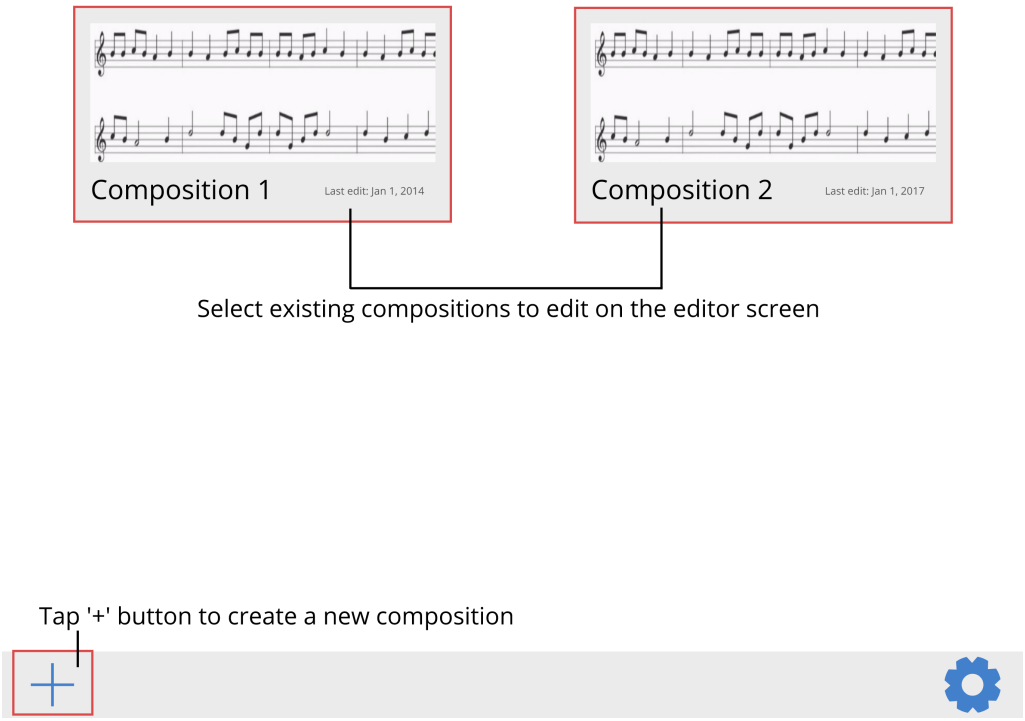

Figure 4.3: Start menu screen.

2. Editor - The user may start creating a composition on this screen. The user may add, edit, delete, copy, cut, or paste notes using the menu. The user may also generate music using the gesture space and set or edit the time signature.

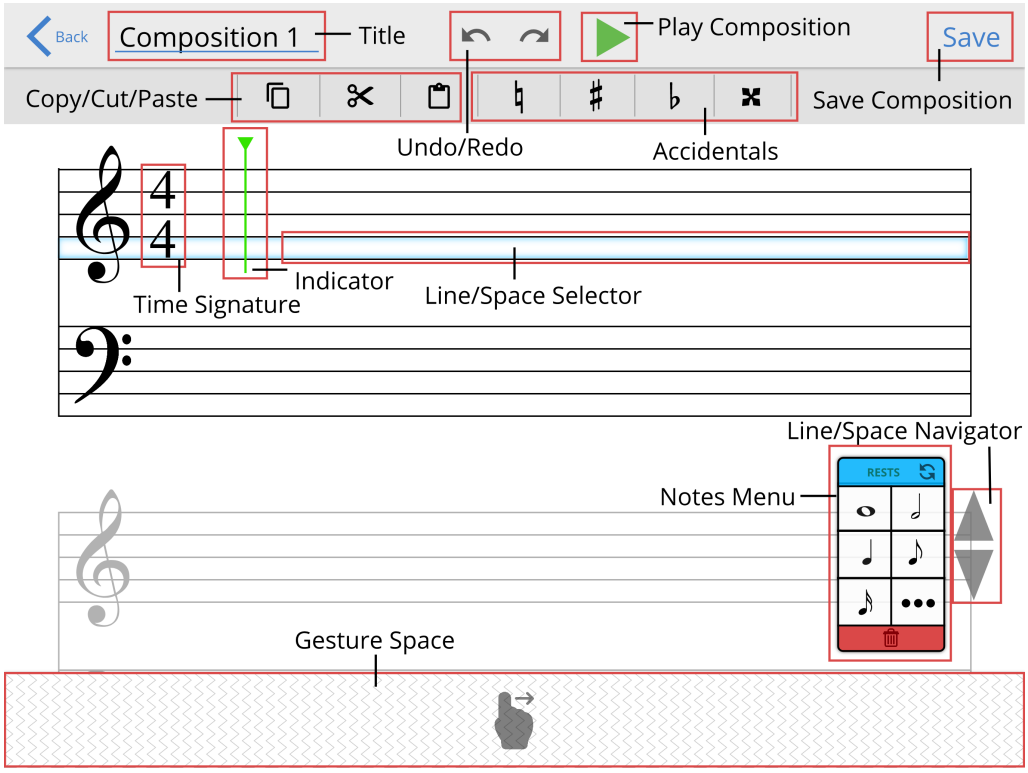

Figure 4.4: Editor screen.

3. Selecting a Note

   To select a note, tap on a note on the staff.

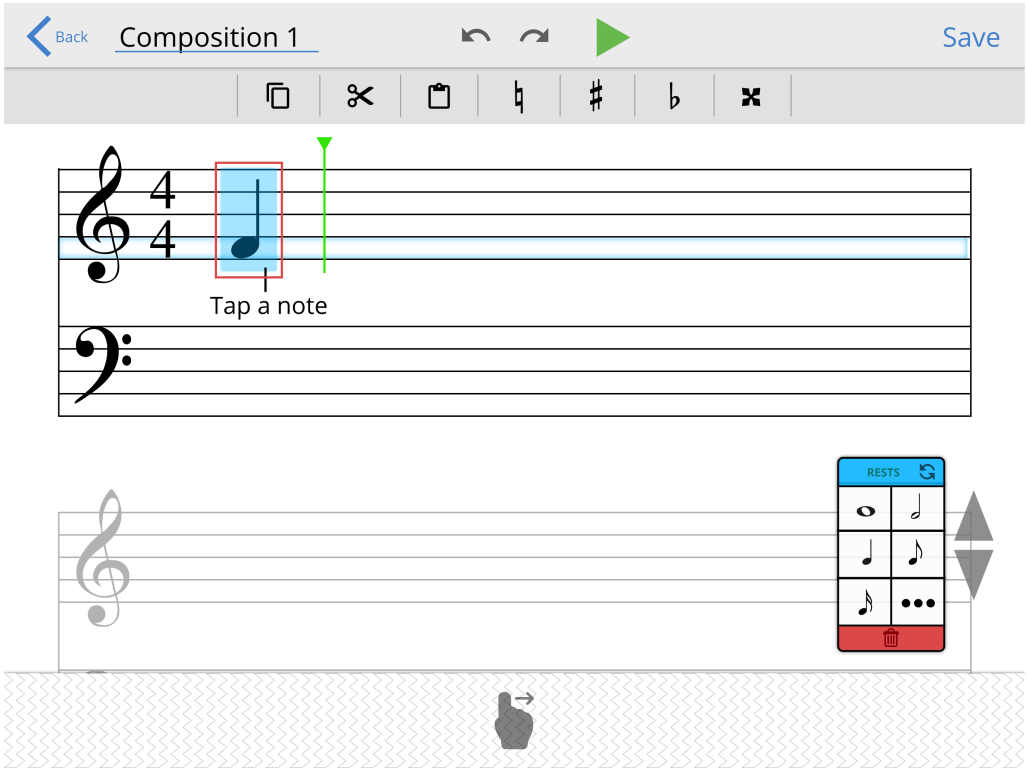

Figure 4.5: Tap on a note to select it.

4. Selecting Notes

   To select multiple notes, drag diagonally upward or downward on the notes.

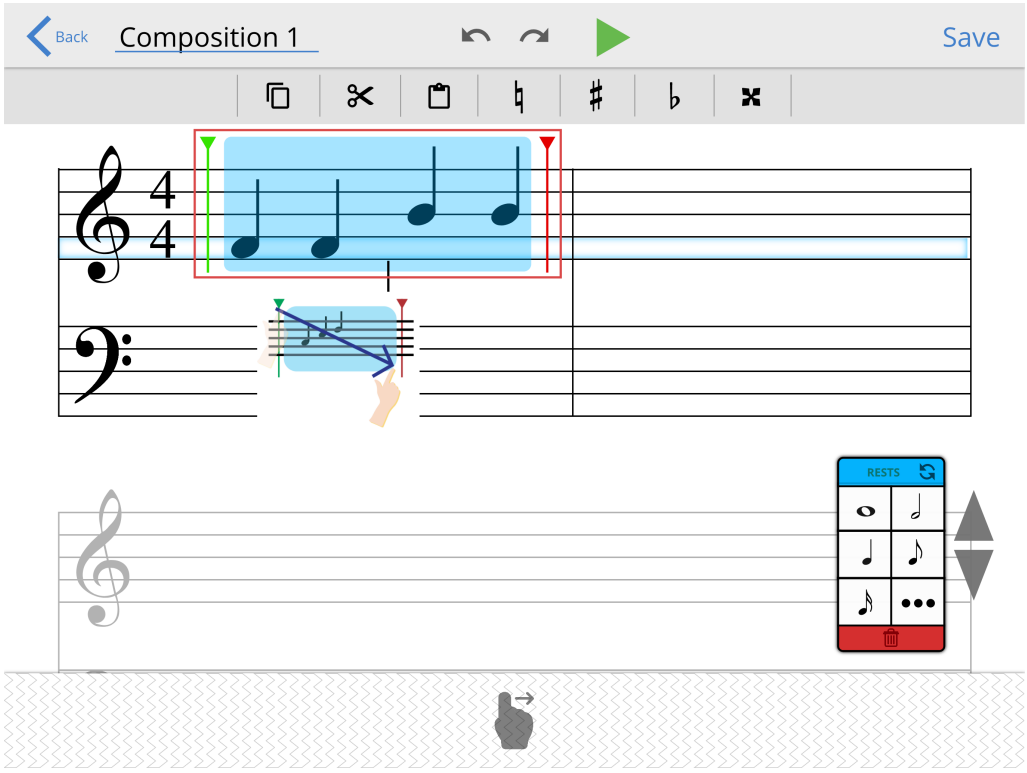

Figure 4.6: Drag diagonally to select multiple notes.

5. Changing Time Signature and Key Signature

   To change the time signature of a composition, tap on the time signature and a menu will pop up.

   Slide the number of beats and/or the beat duration to change the number.

   Tap key signature to change.

   Tap save button to save changes.

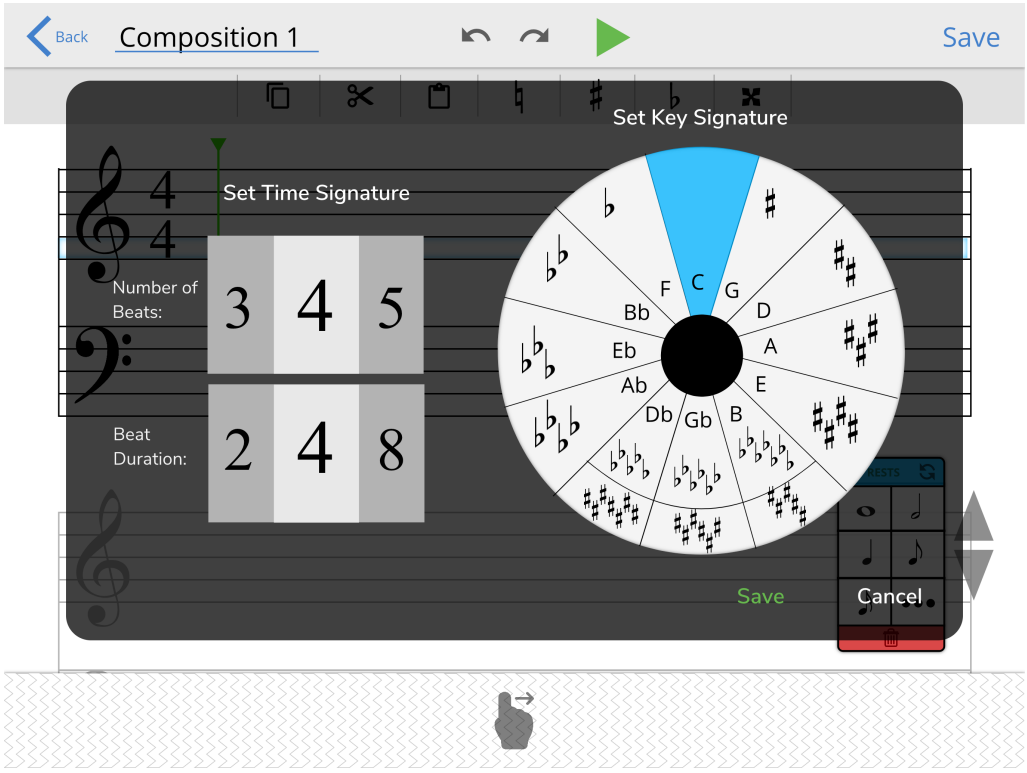

Figure 4.7: Slide the number of beats or beat duration to change the number.

6. Adding Notes

   To add a note, tap on a preferred note in the menu and the note will be automatically placed on the location of the green indicator on the staff with the highlighted space or line.

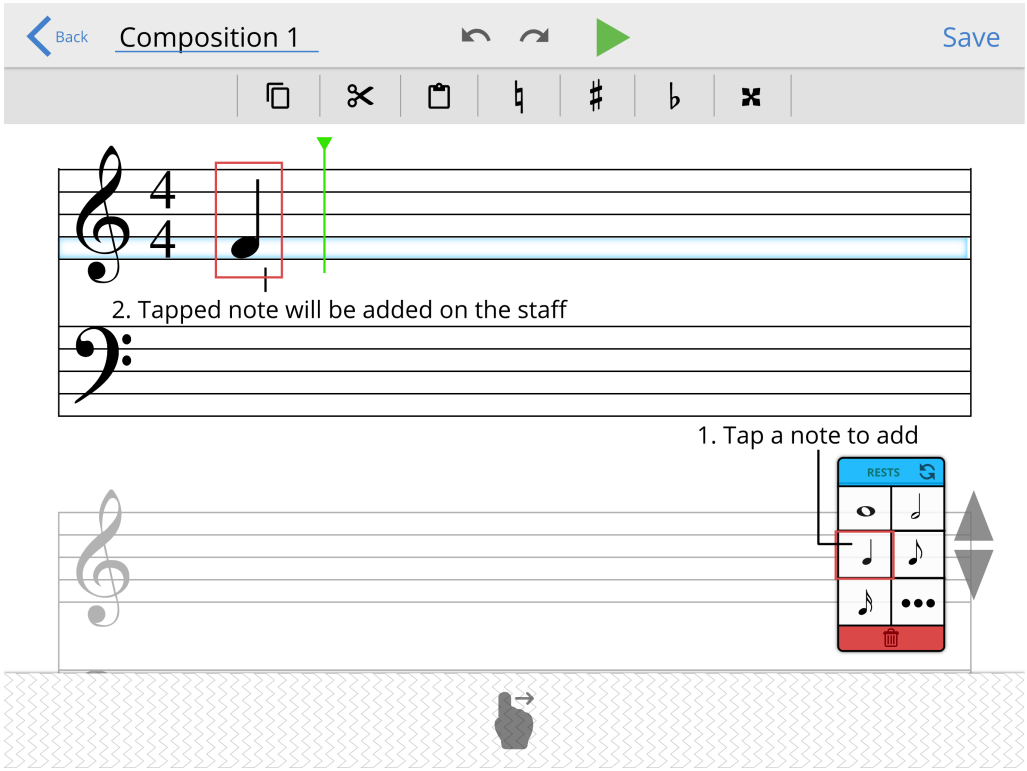

Figure 4.8: Tap a note on the menu to add on the staff.

7. Adding Accidentals

   To add an accidental, choose an accidental by tapping on the menu. Add a note and the accidental will be added with it.

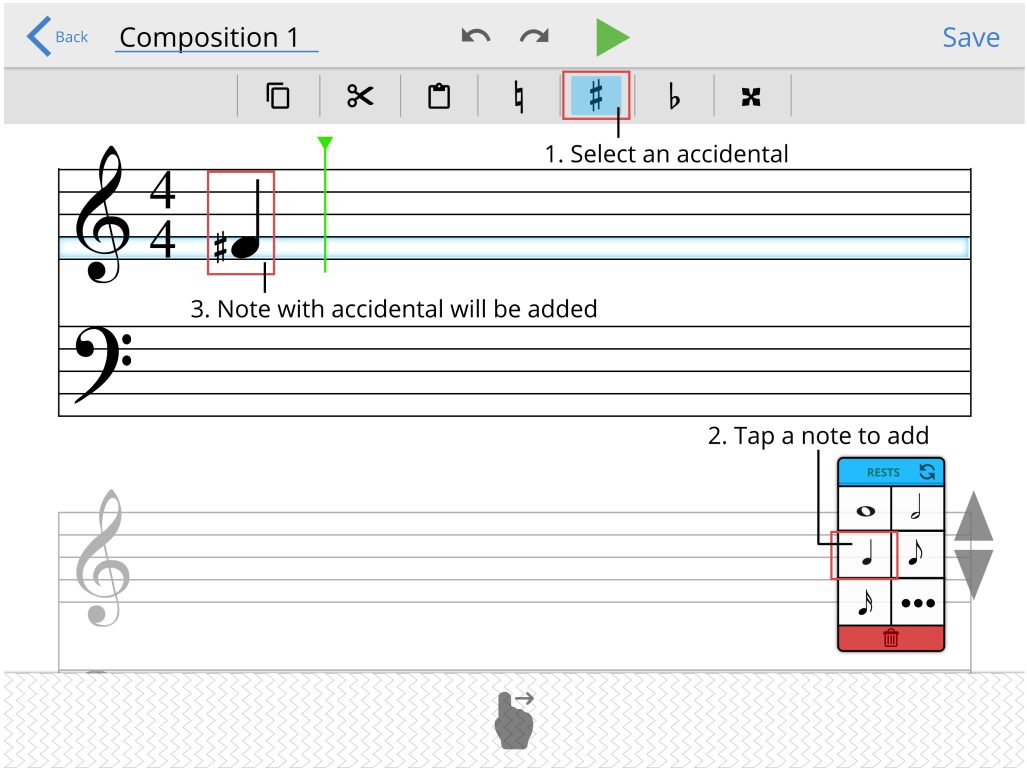

Figure 4.9: Tap an accidental on the menu and add a note.

8. Editing Notes

   To edit a note, tap a note to select it, then click on the desired note on the menu.

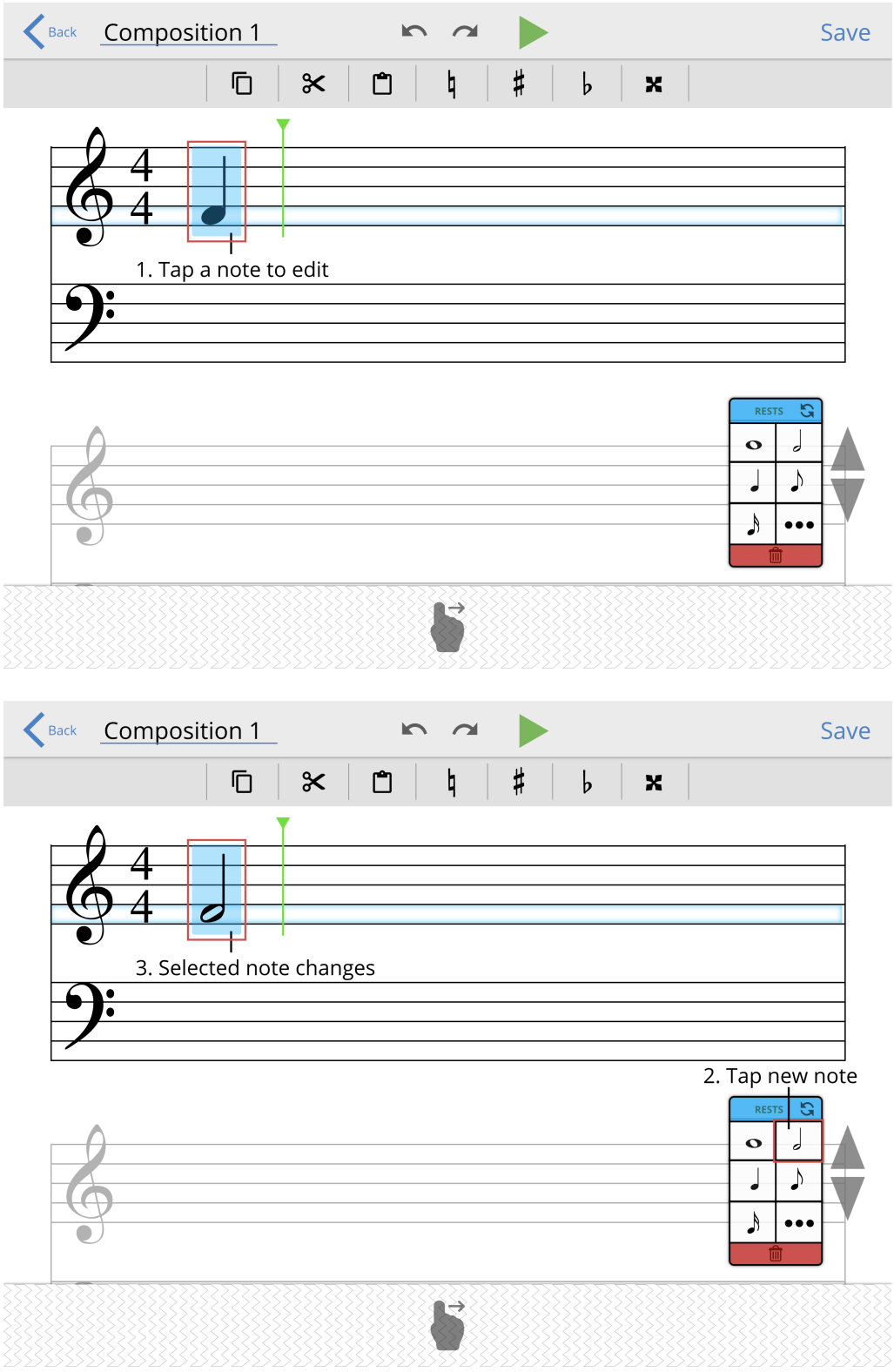

Figure 4.10: Edit note/s by tapping the desired note on the menu.

9. Copy/Cut/Pasting Notes

   To copy or cut a note or a set of notes, tap a note or highlight a set of notes on the staff to select, then tap the copy or cut button on the menu below the composition title to copy. Tap the paste button to paste the notes.

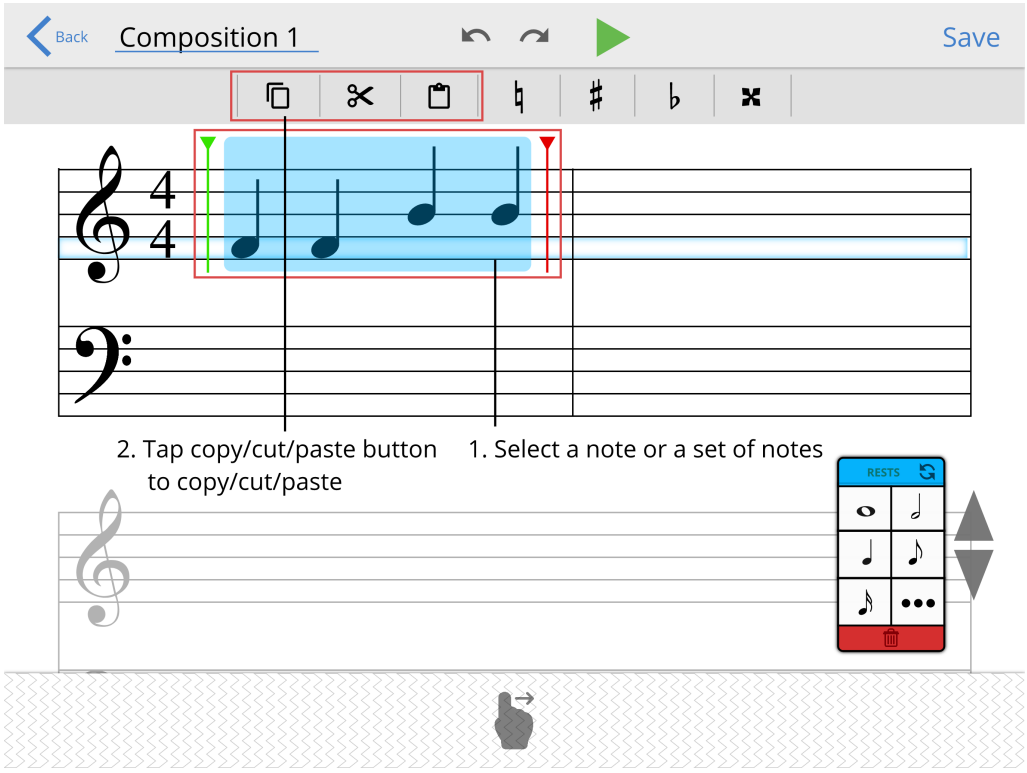

Figure 4.11: Copy or cut note/s by selecting note/s and then tapping copy/cut button. Tap paste to paste notes.

10. Deleting Notes

    To delete a note or a set of notes, tap a note or highlight a set of notes on the staff to select, then make a two-finger flick gesture on the selected note/s or tap the delete button on the menu.

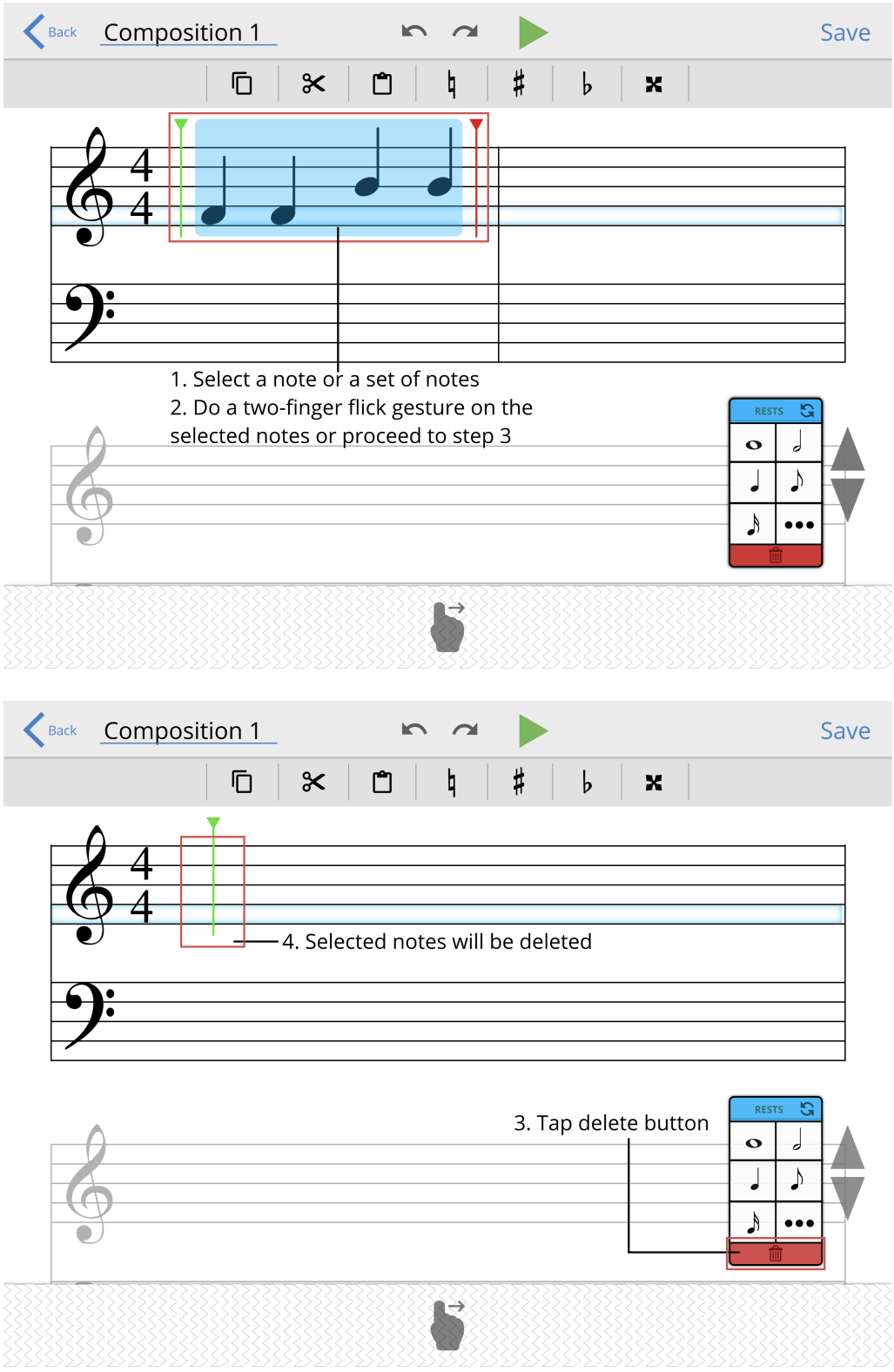

Figure 4.12: Delete note/s by tapping delete on the menu or by making a flick gesture on the selected note/s.

11. Transposing Notes

    To transpose selected notes, make a two-finger swipe up or down gesture on the gesture space.

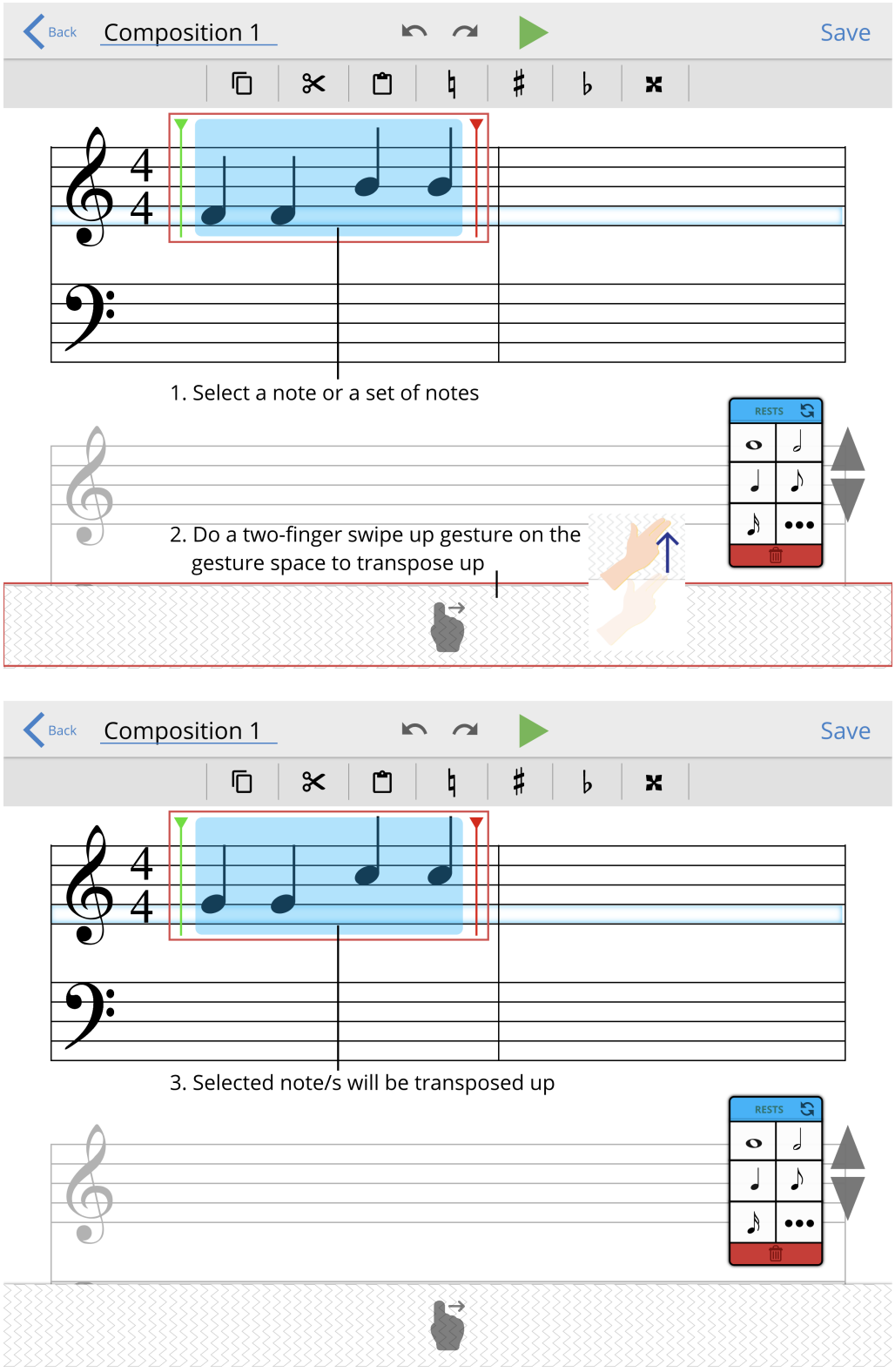

Figure 4.13: Transpose notes by swiping with two fingers up or down on the gesture space.

12. Navigating Line/Space Selector

    To navigate the line/space selector, the user may tap directly on the line/space. Alternatively, the user may use the arrow buttons beside the notes menu to traverse the selector on the staff.

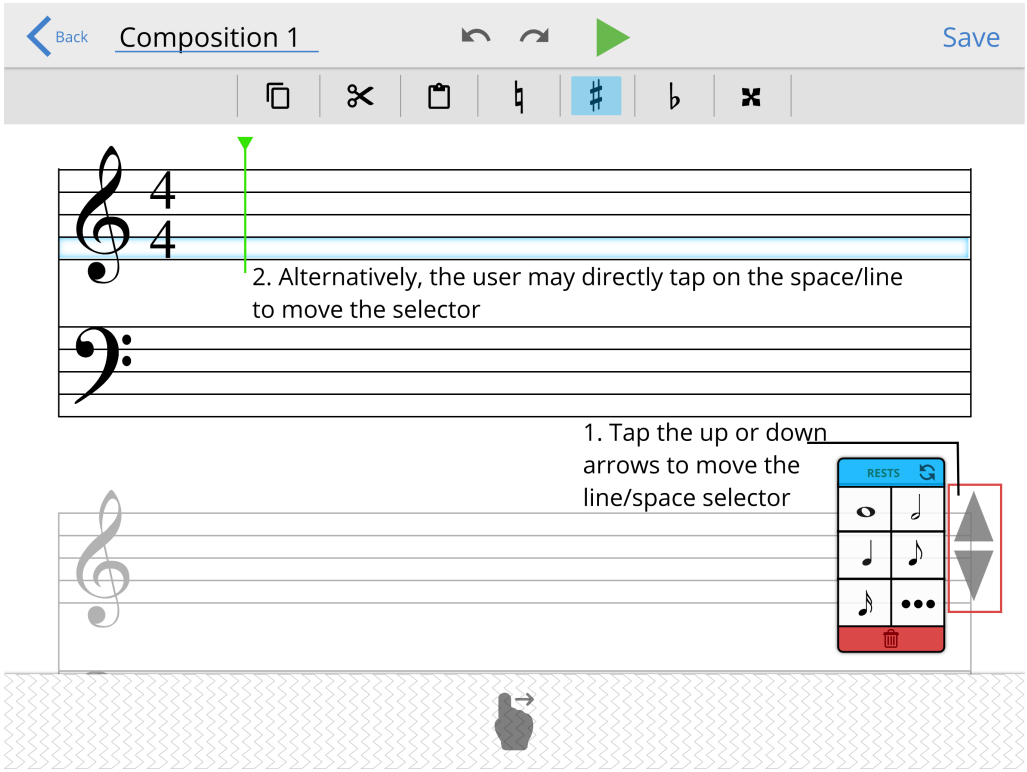

Figure 4.14: Tap directly on the space/line or use the arrow buttons to navigate selector.

13. Music Generation

    Swipe right, up, or down on the gesture space to generate notes starting from the indicator on the staff.

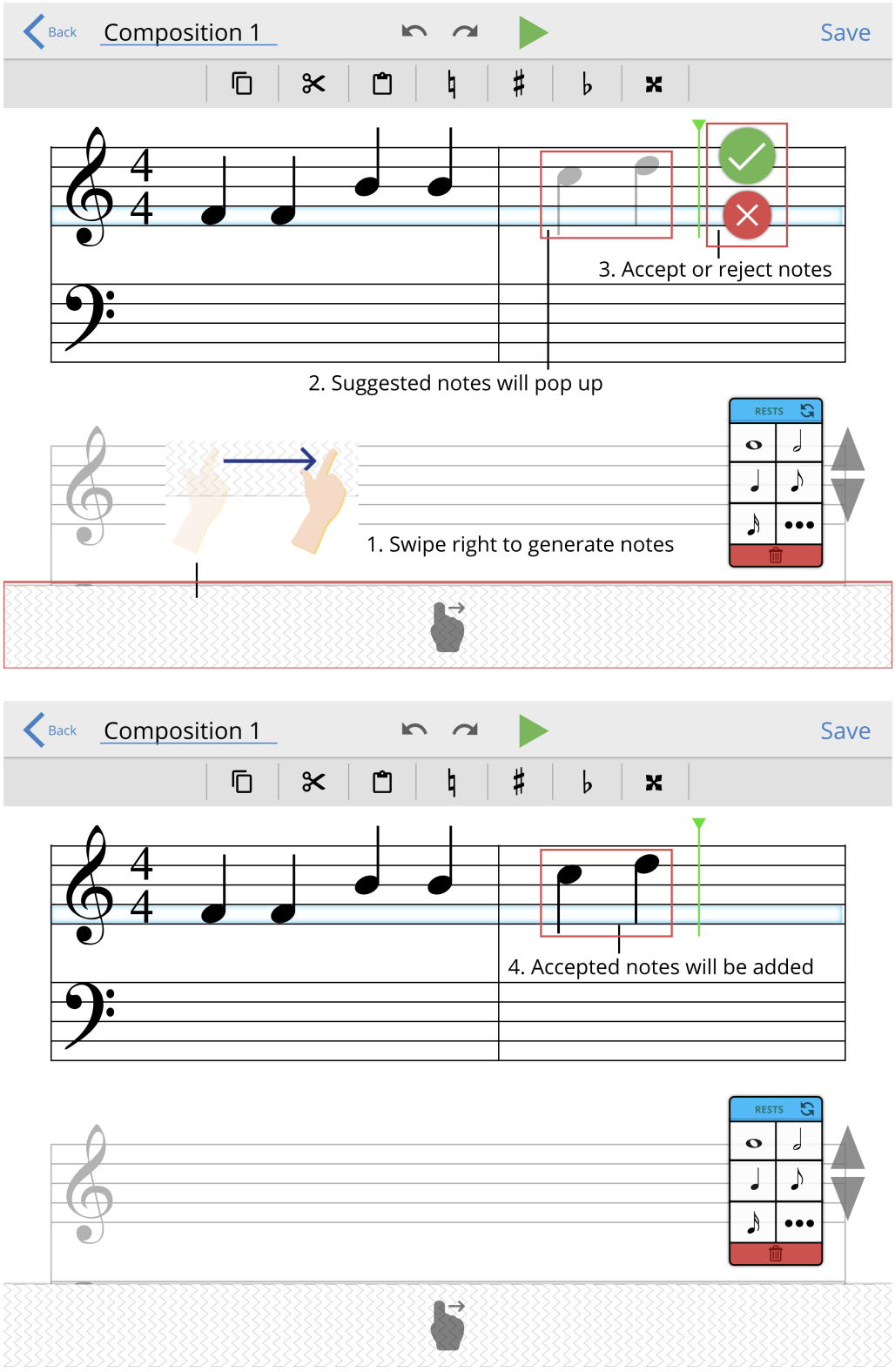

Figure 4.15: Swipe right to generate notes.

14. Note Hint - Tap a note to see the letter and octave number.

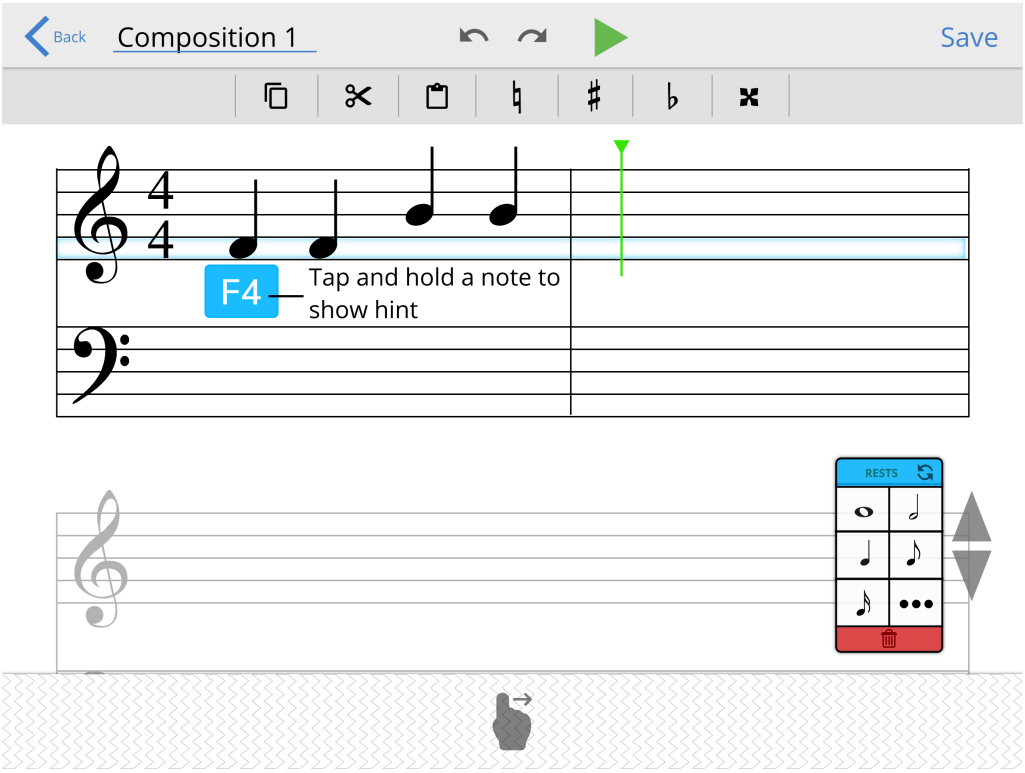

Figure 4.16: Note details popup.

4.2 Experiment Design

This chapter will describe the user stories and use cases that will be highlighted in the testing of the application. The intended users for the application will be expert composers who have at least 7 years of experience with composing music and amateur composers who have less than a year of experience with composing music. These users should also have a basic knowledge on musical terms. The details on the testing to be conducted on these users will also be discussed in this chapter.

4.2.1 User Stories

The user stories of the system will be focused on the main functions of the application and will highlight each specific need that the user has for the system. These

user stories are created with the context that the system will run on a mobile platform with the mode of interaction mainly being touch gestures.

1. As a user, I want to be able to type the name of my composition, so that I can differentiate it from my other compositions
2. As a user, I want to be able to place a note in any valid space in my composition, so that I can create my composition
3. As a user, I want to be able to erase a note in my composition, so that I can make space for other notes in my composition
4. As a user, I want to be able to delete a highlighted group of notes in my composition, so that I can make space faster for other notes I'd like to place in my composition
5. As a user, I want to be able to change the pitch class of a note in my composition, so that I can make adjustments to my composition
6. As a user, I want to be able to move the position of a note in my composition, so that I can move that note to a better position
7. As a user, I want to be able to move the position of a highlighted group of notes in my composition, so that I can reposition multiple notes at a time to a better position
8. As a user, I want to be able to listen to a highlighted section of my composition, so that I can hear a segment of my composition
9. As a user, I want to be able to listen to my whole composition, because I want to listen to it completely
10. As a user, I want to be able to perform a swipe gesture that will generate a succession of notes based on the orientation of my gesture and my current composition because I'm interested in knowing what series of notes match my current composition
11. As a user, I want to be able to manipulate the generated series of notes, because the generated series of notes just needs a little bit more adjustments before I accept it into my composition
12. As a user, I want to be able to discard the series of notes generated by the application after a swipe gesture, so I don't have to delete them one by one

13. As a user, I want to be able to confirm the addition of the series of notes given by the application after a swipe gesture, so I can create my composition quickly

14. As a user, I want to be able to copy the highlighted group of notes, so that I can copy a recurring segment in my composition

15. As a user, I want to be able to paste a copied selection of notes onto my composition, so that I do not need to add a recurring segment in my composition manually all the time

16. As a user I want to be able to undo an action or series of actions, so that I can undo an unintended action or series of unintended actions quicker

17. As a user I want to be able to redo an action or a series of actions, so that I can redo an intended action or a series of intended actions quicker

18. As a user, I want to be able to create a blank composition, so that I can start my work

19. As a user, I want to be able to reposition the menu because I want to place it where it is not an obstacle for me while composing

20. As a user, I want to be able to save my current composition, so that I can move on to a different composition or come back to it later

21. As a user, I want to be able to view my current composition, so that I can read the my current work

22. As a user, I want to be able save my current composition in a format that can be sent to other users or printed on paper

23. As a user, I want to be able to view all my saved compositions in a list, so that I can keep track of everything

24. As a user, I want to be able to delete a composition, so I can discard unneeded compositions to make space for other rests or notes

25. As a user, I want to be able to open a saved composition so that I can perform more actions on it

26. As a user, I want to be able to highlight a group of notes in the composition, so that I can perform actions on the group

27. As a user, I want to be able to highlight a single note, so that I can perform actions on it

28. As a user, I want to be able to set the time signature of the composition
29. As a user, I want to be able to see the details of a single note, so that I can know the specifications of the note
30. As a user, I want to be able to see the details of a selected group of notes, so that I can know the specification of the selected group of notes
31. As a user, I want to be able to see the details of the generated series of notes like the pitch and type so that I can know what notes the system has generated after a gesture
32. As a user, I want to be able to set the clef of the composition
33. As a user, I want to be able to set the key signature of the composition
34. As a user, I want to be able to highlight a rest, so that I can perform actions on that rest
35. As a user, I want to be able to highlight a group of rests, so that I can perform actions on the group of rests
36. As a user, I want to be able to place a rest in the composition, so that my composition can have rests
37. As a user, I want to be able to erase a rest in the composition, so that I can make space for other rests or notes
38. As a user, I want to be able to change a rest in the composition, so that I can make adjustments to my composition
39. As a user, I want to be able to move the position of a rest, so that I can move that rest to a better position
40. As a user, I want to be able to move the position of a selected group of rests, so that I can move multiple rests at a time to a better position
41. As a user, I want to be able to add accidentals to my notes, so that I can control the pitch of my notes

4.2.2 Use Cases

4.2.2.1 Use Case 1: Add a Single Note

Trigger	The composer needs to add a note
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The addition is valid based on musical rules
Process Steps	1. User positions the indicator to a desired location along the musical staff 2. User taps on the desired note to add in the indicated location from the side menu
Minimal Guarantees	A note will be added
Success Guarantees	The desired note is added to the indicated location
Quality Requirements	• Note should be in the valid position • The addition of the note should be valid according to the musical rules

4.2.2.2 Use Case 2: Edit a Single Note

Trigger	The composer needs to delete a note
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The edit is valid based on music rules
Process Steps	1. User selects the note to be deleted 2. User taps the delete option in the side menu to confirm deletion
Minimal Guarantees	A note will be deleted
Success Guarantees	The selected note will be deleted
Quality Requirements	• Deletion of note should not violate music rules

4.2.2.3 Use Case 3: Delete a Single Note

Trigger	The composer needs to delete a note
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The deletion is valid based on music rules
Process Steps	1. User selects the note to be deleted 2. User taps the delete option in the pop-up menu to confirm deletion
Minimal Guarantees	A note will be deleted
Success Guarantees	The correct note will be deleted
Quality Requirements	• Deletion of note should not violate music rules

4.2.2.4 Use Case 4: Horizontal Swipe Gesture to Generate a series of Notes

Trigger	User wants to find a series of notes compatible to his/her current composition
Primary Actor	Composer

Supporting Actors	• Composer • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The user's gesture is valid
Process Steps	1. User performs a swipe gesture across a valid location in Flow 2. User can calibrate or discard the generated notes 3. An initial set of notes have already been placed
Minimal Guarantees	Nothing will happen
Success Guarantees	A series of notes matching the form of the initial set of notes will be presented to the user
Quality Requirements	• The series of notes should be editable by the user • The series of notes should be completely discarded if the user selects the option • The series of notes should follow the musical rules • The series of notes should match the feel of the initial composition

4.2.2.5 Use Case 5: Add a Single Rest

Trigger	The composer needs to add a rest
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The addition is valid based on musical rules
Process Steps	1. User positions the indicator to a desired location along the musical staff 2. User taps on the toggle button on the side menu to open the selection of rests 3. User taps on the desired rest to add in the indicated location from the side menu
Minimal Guarantees	A rest will be added
Success Guarantees	The desired rest is added to the indicated location
Quality Requirements	• Rest should be in the valid position • The addition of the rest should be valid according to the musical rules

4.2.2.6 Use Case 6: Delete a Highlighted Group of Notes or Rests

Trigger	The user needs to delete a group of notes
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The group deletion is valid based on musical rules
Process Steps	1. User highlights a group of notes 2. User taps the delete option in the side menu
Minimal Guarantees	A group of notes will be deleted
Success Guarantees	The highlighted group of notes will be deleted
Quality Requirements	• The deletion of the group of notes does not violate the musical rules • The group of notes will be removed from the composition environment

4.2.2.7 Use Case 7: View the Details of a Single Note or Rest

Trigger	The user wants to know the details of a particular note or rest
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The note or rest exists within the composition
Process Steps	1. The user selects a note or rest 2. The user taps on the toggle details option
Minimal Guarantees	The toggle button graphic will change
Success Guarantees	The toggle button graphic will change and details of the selected note or rest will be displayed
Quality Requirements	• The details displayed about the selected note or rest should be correct • The toggle button icon should be correct

4.2.2.8 Use Case 8: View the Details of a Highlighted Group of Notes or Rests

Trigger	The user needs to view the details of a group of notes
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The group of notes or rests exists
Process Steps	1. The user highlights a group of notes or rests 2. The user taps on the toggle details option
Minimal Guarantees	The toggle button graphic will change
Success Guarantees	The toggle button graphic will change and the details of the group of notes or rests will be displayed
Quality Requirements	• The information displayed about the group of notes or rests should be correct • The toggle button icon should be correct

4.2.2.9 Use Case 9: Listen to the Whole Composition

Trigger	The user wants to listen to the whole composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The composition is not empty
Process Steps	1. The user taps on the play button
Minimal Guarantees	A note will be played
Success Guarantees	The whole composition will be played correctly
Quality Requirements	• There should be no incorrect notes or rest played • The playing should only conclude after the last note or rest is played

4.2.2.10 Use Case 10: Listen to a Highlighted Section of the Composition

Trigger	The user only wants to hear a section of the composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The composition is not empty • The highlighted section contains at least one note or rest
Process Steps	1. The user highlights a section of the composition 2. The user taps the play button
Minimal Guarantees	A note will be played
Success Guarantees	The highlighted section will be played correctly
Quality Requirements	• There should be no incorrect note or rest played • The playing should only conclude after the last note or rest in the highlighted section is played

4.2.2.11 Use Case 11: Create Blank Composition

Trigger	The composer needs to create a new composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user is in the main menu screen • Device memory is not full • Name of the new composition is not taken • Name of the new composition does not exceed the maximum number of characters • Name of the new composition does not contain any invalid characters
Process Steps	1. User taps new composition option from the main menu 2. User writes name for new composition 3. User taps confirm
Minimal Guarantees	A new composition will be created
Success Guarantees	A new blank composition will be created with preliminary musical elements in place

Quality Requirements	• The composition will have all the musical elements setup within the composition environment already • The composition will remain in the memory of the device even after Flow is terminated • The composition can still be opened, viewed, and edited after Flow is relaunched

4.2.2.12 Use Case 12: View Existing Composition

Trigger	The composer needs to view an existing composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user is in the main menu screen • The composition is in storage and is not corrupted
Process Steps	1. User edit composition option from the main menu
Minimal Guarantees	A composition will open
Success Guarantees	The selected composition will open
Quality Requirements	• The opened composition should have all the previous elements placed before in the correct position

4.2.2.13 Use Case 13: Delete Existing Composition

Trigger	The composer needs to delete a composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user is in the main menu screen • The composition is in storage and is not corrupted
Process Steps	1. User taps delete composition option from the main menu 2. User taps confirm
Minimal Guarantees	A composition will be deleted
Success Guarantees	The correct composition will be deleted and will not remain in the system memory
Quality Requirements	• Composition should absolutely be gone from the system memory • Composition name will be made available for future compositions

4.2.2.14 Use Case 14: Redo an Action

Trigger	The user wants to redo an action
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• An undo action has been performed at least once • Action can be redone
Process Steps	1. The user taps on the redo button
Minimal Guarantees	The redo button will give feedback
Success Guarantees	The action is redone
Quality Requirements	• The redone action is the correct one

4.2.2.15 Use Case 15: Undo an Action

Trigger	The user wants to undo an action
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• At least one action has been performed • Action can be undone
Process Steps	1. The user taps on the undo button
Minimal Guarantees	The undo button will give feedback
Success Guarantees	The action is undone
Quality Requirements	• The undone action is the correct one

4.2.2.16 Use Case 16: Highlight a Group of Notes or Rests

Trigger	The user wants to highlight a group of notes or rests
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The section to be highlighted contains at least one note or rest
Process Steps	1. The user highlights a section of the composition
Minimal Guarantees	None
Success Guarantees	A section of the composition will be highlighted based on the gesture of the user
Quality Requirements	• The highlighted section should be the one the user wanted to highlight

4.2.2.17 Use Case 17: Change the Time Signature of the Composition

Trigger	The user wants to change the time signature of the composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The input for the new time signature follows the musical rules
Process Steps	1. The user taps on the time signature 2. The user inputs the desired new time signature 3. The user taps confirm
Minimal Guarantees	The pop-up for the input of the new time signature closes
Success Guarantees	The new time signature is applied to the composition
Quality Requirements	• The notes and rests within the composition will adjust based on the new time signature

4.2.2.18 Use Case 18: Change a Clef

Trigger	The user wants to change a clef in the composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The change does not violate the musical rules
Process Steps	1. The user taps on a clef to change 2. The user selects the new desired clef 3. The user taps confirm
Minimal Guarantees	The menu containing the selection of clef does not remain in the screen
Success Guarantees	The selected clef is changed into the desired clef
Quality Requirements	• The desired clef is in the right staff within the composition • The desired clef is the correct one that the user selected

4.2.2.19 Use Case 19: Change the Key Signature of the Composition

Trigger	The user wants to change the key signature of the composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The input for the new key signature follows the musical rules
Process Steps	1. The user taps on the key signature 2. The user inputs the desired new key signature 3. The user taps confirm
Minimal Guarantees	The pop-up for the input of the new key signature closes
Success Guarantees	The new key signature is applied to the composition
Quality Requirements	• The notes and rests within the composition will adjust based on the new key signature

4.2.2.20 Use Case 20: Rename the Composition from the Composition Environment

Trigger	The user wants to change the name of the composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The new name does not violate any restrictions or limits set within Flow
Process Steps	1. The user taps on the name of the composition 2. The user inputs the new name of the composition 3. The user taps confirm
Minimal Guarantees	The pop-up for the input of the new name closes
Success Guarantees	The new name is set on the composition
Quality Requirements	• The name should remain the same even after Flow reboots or the composition is closed

4.2.2.21 Use Case 21: Rename the Composition from the Main Menu

Trigger	The user wants to change the name of the composition
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user is on the main menu screen • The new name does not violate any restrictions or limits set within Flow
Process Steps	1. The user taps on edit name option 2. The user inputs the new name of the composition 3. The user taps confirm
Minimal Guarantees	The text box containing the name reverts back to normal and should no longer ask for user input
Success Guarantees	The new name is set on the composition
Quality Requirements	• The name should remain the same even after Flow reboots or the composition is closed

4.2.2.22 Use Case 22: Move the Note Menu to a Different Position

Trigger	The user wants to move the note menu to a different position
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The new location for the note menu is valid
Process Steps	1. The user drags the note menu to the desired location
Minimal Guarantees	None
Success Guarantees	The note menu will move to its desired location
Quality Requirements	• The note menu will remain on its desired location after the user lifts his/her finger off the screen • The note menu will remain on its desired location even after the composition wherein the note menu was moved in is reopened

4.2.2.23 Use Case 23: Toggle Accidental while Adding Notes

Trigger	The user wants to add an accidental to the next note
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The addition of a new note with an accidental follows the musical rules
Process Steps	1. The user toggles accidental from the top menu 2. The user taps on a note from the side menu
Minimal Guarantees	The graphic for the toggled accidental will change
Success Guarantees	The note with the toggled accidental will be added
Quality Requirements	• The note will be added with the toggled accidentals • The note with the accidentals will be placed on the desired location along the musical staff

4.2.2.24 Use Case 24: Export to MusicXML from the Main Menu

Trigger	The user wants to export the composition to MusicXML
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user in the main menu of Flow • The composition exists
Process Steps	1. The user taps on the settings button 2. The user taps on the export to MusicXML button 3. The user select one of the existing compositions to export 4. The user taps confirm 5.
Minimal Guarantees	The settings menu pop-up will close
Success Guarantees	The composition is exported to MusicXML and stored within device storage

Quality Requirements	• The composition is in MusicXML format • The composition is exported into device storage • The composition can be opened by applications that can open Music XML • The composition exported will remain in the system after export

4.2.2.25 Use Case 25: Transpose a Single Note

Trigger	The user wants to transpose a note to change its pitch
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The transposition is valid
Process Steps	1. The user selects a note 2. The user gestures to transpose the note
Minimal Guarantees	None
Success Guarantees	A note will be transposed
Quality Requirements	• The note is correctly transposed based on the gesture of the user

4.2.2.26 Use Case 26: Transpose a Selected Group of Notes

Trigger	The user wants to transpose a selected group of note to change their pitch
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The transposition is valid
Process Steps	1. The user selects group of notes 2. The user gestures to transpose the group of note
Minimal Guarantees	None
Success Guarantees	A group of notes will be transposed
Quality Requirements	• The group of notes are correctly transposed based on the gesture of the user

4.2.2.27 Use Case 27: Change a Note to its Dotted Version

Trigger	The user wants to change a note to its dotted version
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow • The change into dotted version does not violate the musical rules
Process Steps	1. The user tap and holds on a selected note 2. The user hovers his/her finger on the selected dotted version 3. The user lifts his/her finger from the device screen
Minimal Guarantees	The pop-up menu for dotted versions of the notes will close
Success Guarantees	The note will change to its selected dotted version
Quality Requirements	• The dotted version of the note is in the correct position along the musical staff • The dotted version of the note will remain even after the composition is reopened or Flow is relaunched

4.2.2.28 Use Case 28: Toggle from Note Menu to Rest Menu

Trigger	The user wants to view the rest menu
Primary Actor	Composer
Supporting Actors	• Listener • Instrumentalist • Producer
Precondition	• The user has a composition environment open in Flow
Process Steps	1. The user taps toggle to rest menu button in the side menu
Minimal Guarantees	The toggle to rest menu button will give feedback
Success Guarantees	The rest menu will be displayed
Quality Requirements	• The proper rests are shown in the rest menu • The note menu will be replaced by the rest menu • The toggle note menu button will be in place of the toggle rest menu button

4.2.3 User Test Plan

The tests will be conducted in an environment wherein a tester would not be influenced by the researchers to ensure that their interaction with the application is as close as to a natural setting as possible. The tests will be conducted through

a tablet device that can run the application. The prototypes for testing will be a mid-fidelity prototype created through the use of the Invision application, a prototyping tool, and the actual high-fidelity system developed using Swift.

Before the starting the test, the testers are given a brief description of the system. They will also be informed of the objectives of the system. The researchers will be collecting 3 kinds of data: time, cognitive load, and quality. There will be tasks that the testers will be asked to accomplish, and during and after those tasks, data will be collected. The time it takes to finish each task will be recorded. The cognitive load will be measured by using CogTool to receive a predicted cognitive load, and identifying moments of confusion for the users. After the testing, testers are then asked to answer a questionnaire to evaluate the quality of the application.

The objective of the test will be to determine if the user interface and design of the system does not hinder the productivity of the target user. The functionality of the system will also be tested to see if there are problems that will occur during the testing.

4.2.3.1 Tasks

Some tasks may be omitted in different test setups when a crucial function to carry out the task is unavailable. During the final task, testers are encouraged to speak aloud their concerns and opinions regarding the application.

1. Add a note in the composition
2. Add a series of notes through a swipe gesture
3. Compose a musical piece worth 2 staffs
4. Compose a musical piece worth 2 staffs without using gestures
5. Free roam to use the application for 3-5 minutes

4.2.3.2 Test Setups

The test setups are meant to compare existing methods of composition with composition using Flow. The tasks enumerated in Section 4.2.3.1 will be used in each test setup whenever possible since some tasks can only be accomplished within a specific setup.

There will be 3 different test setups to be used during user testing namely:

1. Tester composing using Flow
2. Tester composing using music sheets
3. Tester composing using Komp

The first setup will be focused on the composer using the developed system, Flow. During the early periods of testing, a mid-fidelity InVision prototype will be used as a substitute. Using this prototype, the only task that the composer will be able to do is to explore the application. The main goal of this is to develop and improve the interaction design of the application. One a high-fidelity prototype has been developed, all tasks will be performed in the testing.

The second setup simulates the traditional way of musical composition. This is through music sheets and a writing instrument. Traditional music sheets will be provided by the researchers as well as a choice of using a pen or pencil for writing musical elements. The composers are free to use as much music sheets as they want for drafting and experimenting with their composition as long as the time and resources provided by the researchers allow them.

The third setup will be through an existing mobile application for iOS platforms, komp. This setup will be done mainly to compare the performance of the user when performing certain tasks that are also available in Flow. The researchers expect this setup to have similar results with the first setup and any data collected in this setup will be used to evaluate Flow.

Appendix A

Research Ethics Forms

Appendix B

Turnitin Similarity Report

Appendix C

Data Collection Artifacts

Appendix D

Design Artifacts

D.1 Affinity Diagram

D.2 User Personas

Mary: The Amateur

- 18 years old
- Started composing as a hobby when she was 16 years old
- 2nd Year undergraduate student
- Currently taking up a Bachelor in Music degree program
- Passing grades during her 1st Year

"Its such a hassle to write notes on music sheets. Its hard to keep track which are the drafts."

"I haven't found any app like Finale in the app store."

Mary lives in a dorm with 3 other people and has her own laptop and tablet. She uses both of these devices to do her work in any place.

Mary mostly uses music sheets when she needs to compose music for assignments. She usually uses her laptop for research and the tablet as a substitute when she doesn't feel like bringing her laptop.

Mary has passing marks in her courses. She experiences difficulty getting high grades because she has a hard time looking for inspiration and ideas. She also easily gets tired writing, revising, and finalizing her compositions on paper.

She is knowledgeable in using Sibelius on her laptop and has started to learn Finale due to her professors urging her to use it. She appreciates how these kinds of applications reduce the steps in some repetitive or tedious processes in composing music.

Warren: The Experienced

- 34 years old
- 10 years of composing music professionally
- Music teacher in a high school
- Loves classical music
- Music fundamentalist
- Wants his students to be more engaged in music and appreciate the art form
- Faculty for 5 years

"Music is a way to truly express yourself"

"I want to share my passion in music to my pupils"

Warren is currently a faculty member of a private high school that pays him enough that he can make a living and support his family.

Warren has been teaching for 5 years. He has been involved in organizing the curriculum in the school. He has always believed that music as an art form is timeless and should be learned by everyone.

He wants his students to appreciate music as much as he does. He shares the works of Beethoven and Mozart, which are his favorites, to his class.

He often observes students using their mobile devices in school. During class, he's made it clear that the usage of cellphones is restricted. However, he understands the potential of these mobile devices as tools for musical composition.

He is experienced in using Finale but still uses music sheets during occasions where he cannot use his laptop or is still sketching. He has been wondering whether there is a viable mobile alternative to using music sheets.

Winston: The Veteran

- 62 Years Old
- 40 years of experience in composing music professionally
- Composes music for different companies and artists
- Mentors up and coming composers
- Married for 35 years
- Has experienced the numerous shifts of popularity between music genres

"Music has always been a part of everyone's lives"

"I wonder whats next the next big thing that can change music"

Warren always reads on music-related news and enjoys doing research on topics that concern music. He writes his findings and his thoughts on his blog. He has lived through several shifts in mainstream music and has studied the change agent in every shift.

He has been able to adjust and make a living from these hobbies. He has always been able to make music that matched what was popular at the time. He fully accepts that music changes for every new generation of artists and composers.

He is waiting for the next big thing that can happen for music, always prepared to make the needed adjustments to stay in the game. He is no stranger to using technology like Finale or Sibelius to compose his pieces.

Appendix E

Literature Map

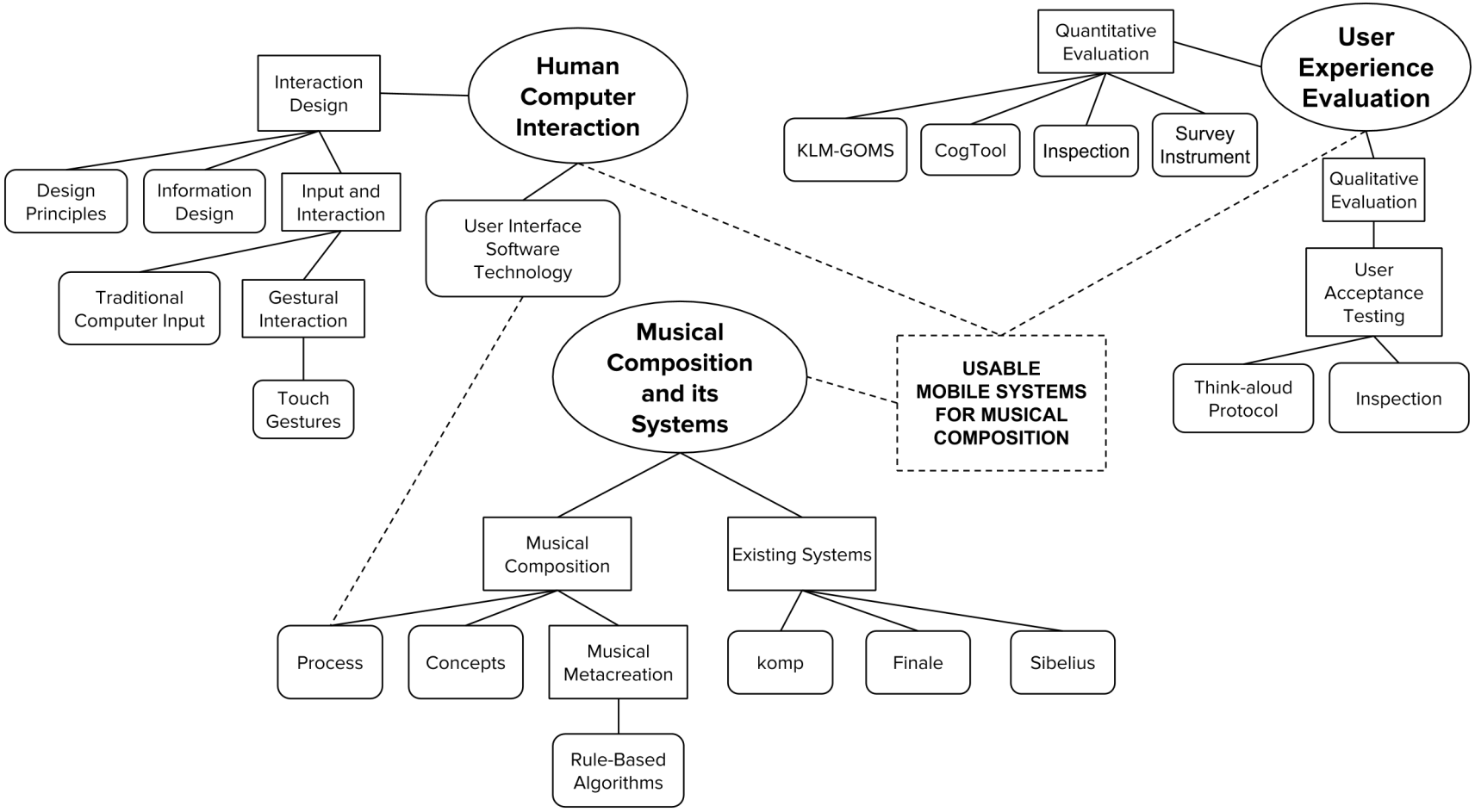

Figure E.1: Literature map.

Appendix F

Initial Results

Table F.1: Motor Module

Attribute	Attribute Description	Value
PECK-FITTS-COEFF	b coefficient in Fitts's equation for PECK movements.	0.075
DEFAULT-TARGET-WIDTH	Effective width, in degrees visual angle, of targets with undefined widths.	1.0
MIN-FITTS-TIME	Minimum movement time for an aimed [Fitts's] movement.	0.1
MOTOR-BURST-TIME	Minimum time for any movement.	0.05
MOTOR-INITIATION-TIME	Time to initiate a motor movement.	0.05
MOTOR-FEATURE-PREP-TIME	Time to prepare a movement feature.	0.001

Table F.2: Imaginal Module

Attribute	Attribute Description	Value
IMAGINAL-DELAY	Time in seconds to respond to an imaginal request	0.2

Table F.3: Temporal Module

Attribute	Attribute Description	Value
TIME-NOISE	Temporal noise	0.015

TIME-MASTER-START-INCREMENT	Temporal start interval	0.011
TIME-MULT	Temporal multiplier	1.1
RECORD-TICKS	Record each time increment as a buffer event	T

References

Allen, J. F. (1983, November). Maintaining knowledge about temporal intervals. *Commun. ACM*, *26*(11), 832–843. Retrieved from `http://doi.acm.org/10.1145/182.358434` doi: 10.1145/182.358434

Ames, C. (1989). The markov process as a compositional model: A survey and tutorial. *Leonardo*, 175–187.

Apple Inc. (2017). *Uigesturerecognizer.* Retrieved from `https://developer.apple.com/documentation/uikit/uigesturerecognizer`

Arcos, J. L., de Mantaras, R. L., & Serra, X. (1998). Saxex: A case-based reasoning system for generating expressive musical perfomances. *Journal of New Music Research*, *27*(3), 194.

Bellamy, R., John, B., & Kogan, S. (2011, May). Deploying cogtool: integrating quantitative usability assessment into real-world software development. In *2011 33rd international conference on software engineering (icse)* (p. 691-700).

Bennett, S. (1976). The process of musical creation: Interviews with eight composers. *Journal of research in music education*, *24*(1), 3–13.

Birchfield, D. (2003). Generative model for the creation of musical emotion, meaning, and form. In *Proceedings of the 2003 ACM SIGMM workshop on experiential telepresence* (pp. 99–104).

Blair-Early, A., & Zender, M. (2008). User interface design principles for interaction design. *Design Issues*, *24*(3), 85–107.

Bozhanov, B. (2014). Computoser-rule-based, probability-driven algorithmic music composition. *arXiv preprint arXiv:1412.3079*.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., & Yergeau, F. (1997). Extensible markup language (xml). *World Wide Web Journal*, *2*(4), 27–66.

Bresin, R., & Friberg, A. (2000). Emotional coloring of computer-controlled music performances. *Computer Music Journal*, *24*(4), 44–63.

Brinkman, A. R. (1984). *A data structure for computer analysis of musical scores.* Ann Arbor, MI: Michigan Publishing, University of Michigan Library.

Brown, D., Nash, C., & Mitchell, T. (2017). A user experience review of music interaction evaluations. In *Proceedings of the international conference on new interfaces for musical expression 2017.*

Burney, C. (1789). *A general history of music, from the earliest ages to the present period* (No. 4). New York, Harcourt, Brace and Company.

Burrows, T. (1999). *How to read music in 10 lessons.* Dubai: Carlton Books Limited.

Buxton, W., Reeves, W., Baecker, R., & Mezei, L. (1978). The use of hierarchy and instance in a data structure for computer music. *Computer Music Journal*, *2*(4), 10-20.

Camurri, A., Canepa, C., Frixione, M., & Zaccaria, R. (1991). Harp: a system for intelligent composer's assistance. *Computer*, *24*(7), 64–67.

Cao, X., Sun, L., Niu, J., Wu, R., Liu, Y., & Cai, H. (2015). Automatic composition of happy melodies based on relations. *Multimedia Tools and Applications*, *74*(21), 9097–9115.

Card, S. K., Newell, A., & Moran, T. P. (1983). *The psychology of human-computer interaction.* Hillsdale, NJ, USA: L. Erlbaum Associates Inc.

Chen, Z., Anquetil, E., Mouchre, H., & Viard-Gaudin, C. (2014, Sept). A graph modeling strategy for multi-touch gesture recognition. In *2014 14th international conference on frontiers in handwriting recognition* (p. 259-264). doi: 10.1109/ICFHR.2014.51

Cheng, C. (2016). On approaching a performance of paul hindemith's der schwanendreher.

Clough, J., & Myerson, G. (1986). Musical scales and the generalized circle of fifths. *The american mathematical monthly*, *93*(9), 695–701.

Collins, D. (2005). A synthesis process model of creative thinking in music composition. *Psychology of music*, *33*(2), 193–216.

Commoner, B. (2014). *The closing circle: nature, man, and technology.* Knopf.

Cope, M. E., & Uliano, K. C. (1995). Cost-justifying usability engineering: A real world example. In *Proceedings of the human factors and ergonomics society annual meeting* (Vol. 39, pp. 263–267).

Curtis, B., Krasner, H., & Iscoe, N. (1988). A field study of the software design process for large systems. *Communications of the ACM*, *31*(11), 1268–1287.

Davis, F. D., & Venkatesh, V. (2004). Toward preprototype user acceptance testing of new information systems: implications for software project management. *IEEE Transactions on Engineering management*, *51*(1), 31–46.

Deka, B. (2016). Data-driven mobile app design. In *Proceedings of the 29th annual symposium on user interface software and technology* (pp. 21–24). New York, NY, USA: ACM. Retrieved from `http://doi.acm.org/10.1145/2984751.2984786` doi: 10.1145/2984751.2984786

De Mantaras, R. L. (2012). Playing with cases: rendering expressive music with case-based reasoning. *AI Magazine*, *33*(4), 22.

De Mantaras, R. L., & Arcos, J. L. (2002). Ai and music: From composition to expressive performance. *AI Magazine*, *23*(3), 43.

Dix, A. (2009). *Human-computer interaction*. Springer.

Donnelly, K. (2005). *The spectre of sound: Music in film and television*. British Film Institute; University of California Press.

Dörfler, M. (2001). Time-frequency analysis for music signals: A mathematical approach. *Journal of New Music Research*, *30*(1), 3–12.

Eco, U. (1976). *A theory of semiotics* (Vol. 217). Indiana University Press.

Eigenfeldt, A., & Pasquier, P. (2010). Realtime generation of harmonic progressions using controlled markov selection. In *Proceedings of iccc-x-computational creativity conference* (pp. 16–25).

Engelbart, D. C. (1962). Augmenting human intellect: a conceptual framework. *PACKER, Randall and JORDAN, Ken. Multimedia. From Wagner to Virtual Reality. New York: WW Norton & Company*, 64–90.

Engelbart, D. C., & English, W. K. (1968). A research center for augmenting human intellect. In *Proceedings of the december 9-11, 1968, fall joint computer conference, part i* (pp. 395–410). New York, NY, USA: ACM. Retrieved from `http://doi.acm.org/10.1145/1476589.1476645` doi: 10.1145/1476589.1476645

Epps, J., Lichman, S., & Wu, M. (2006). A study of hand shape use in tabletop gesture interaction. In *CHI '06 extended abstracts on human factors in computing systems* (pp. 748–753). New York, NY, USA: ACM.

Ericsson, K. A., & Simon, H. A. (1998). How to study thinking in everyday life: Contrasting think-aloud protocols with descriptions and explanations of thinking. *Mind, Culture, and Activity*, *5*(3), 178–186.

Fagan, M. E. (1999). Design and code inspections to reduce errors in program development. *IBM Systems Journal*, *38*(2), 258-287. Retrieved from `https://search.proquest.com/docview/222415339?accountid=190474` (Copyright - Copyright International Business Machines Corporation 1999; Last updated - 2012-02-07; CODEN - IBMSA7)

Farbood, M. M., Pasztor, E., & Jennings, K. (2004). Hyperscore: a graphical sketchpad for novice composers. *IEEE Computer Graphics and Applications*, *24*(1), 50–54.

Findlater, L., Lee, B., & Wobbrock, J. (2012). Beyond QWERTY: augmenting touch screen keyboards with multi-touch gestures for non-alphanumeric input. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 2679–2682).

Fischer, G. (2001). User modeling in human–computer interaction. *User modeling and user-adapted interaction*, *11*(1), 65–86.

Frøkjær, E., Hertzum, M., & Hornbæk, K. (2000). Measuring usability: are effectiveness, efficiency, and satisfaction really correlated? In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 345–352).

Fucks, W. (1962). Mathematical analysis of formal structure of music. *IRE Transactions on Information Theory*, *8*(5), 225–228.

Gabor, D. (1946). Theory of communication. part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, *93*(26), 429–441.

Galitz, W. O. (2007). *The essential guide to user interface design: an introduction to gui design principles and techniques*. John Wiley & Sons.

Gartland-Jones, A., & Copley, P. (2003). The suitability of genetic algorithms for musical composition. *Contemporary Music Review*, *22*(3), 43–55.

Geis, M., & Middendorf, M. (2008). Creating melodies and baroque harmonies with ant colony optimization. *International Journal of Intelligent Computing and Cybernetics*, *1*(2), 213–238.

Gill, A. M., & Nonnecke, B. (2012). Think aloud: Effects and validity. In *Proceedings of the 30th acm international conference on design of communication* (pp. 31–36). New York, NY, USA: ACM. Retrieved from `http://doi.acm.org/10.1145/2379057.2379065` doi: 10.1145/2379057.2379065

Gleaves, T. (2015). *Understanding the circle of fifths.* `http://www.piano-lessons-central.com/music-notation/circle-of-fifths/`. (Accessed: 2017-09-26)

Goddard III, R. D., & Villanova, P. (1996). Designing surveys and questionnaires for research. *The psychology research handbook: A guide for graduate students and research assistants*, 85–97.

Gomoll, K. (1996). Some techniques for observing users. In Laurel (Ed.), (Vol. 1, p. 85-90). Addison-Wesley Publishing Company.

Goto, M. (2004). A real-time music-scene-description system: Predominant-f0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication*, *43*(4), 311–329.

Grachten, M. (2006). A case based approach to expressivity-aware tempo transformation. , *vol. 65, no. 2-3 (Dec 2006), p. 411*.

Grant, A. (1994, August 16). *Computer keyboard.* Google Patents. Retrieved from `https://www.google.com/patents/US5339097` (US Patent 5,339,097)

Gündüz, G., & Gündüz, U. (2005). The mathematical analysis of the structure of some songs. *Physica A: Statistical Mechanics and its Applications*, *357*(3), 565–592.

Hasan, H. S., & Kareem, S. A. (2012, Nov). Human computer interaction for vision based hand gesture recognition: A survey. In *2012 international conference on advanced computer science applications and technologies (acsat)* (p. 55-60). doi: 10.1109/ACSAT.2012.37

Hawkins, S., & Burns, L. (2013). *From pac-man to pop music: interactive audio in games and new media.* Ashgate Publishing, Ltd.

Herremans, D., & Chew, E. (2016). Morpheus: Automatic music generation with recurrent pattern constraints and tension profiles [technical report]. In *IEEE TENCON.* Singapore: IEEE.

Hess, R. (2008). *Sibelius vs. finale: How to choose?* `http://www.filmmusicmag.com/?p=1820`. (Accessed: 2017-10-06)

Hollingsed, T., & Novick, D. G. (2007). Usability inspection methods after 15 years of research and practice. In *Proceedings of the 25th annual acm international conference on design of communication* (pp. 249–255).

Holzinger, A. (2005). Usability engineering methods for software developers. *Communications of the ACM*, *48*(1), 71–74.

Horn, R. E. (1999). Information design: Emergence of a new profession. *Information design*, 15–33.

Hristidis, V., Gravano, L., & Papakonstantinou, Y. (2003). Efficient ir-style keyword search over relational databases. In *Proceedings of the 29th international conference on very large data bases-volume 29* (pp. 850–861).

Humphrey, W. S. (1995). *A discipline for software engineering*. Addison-Wesley Longman Publishing Co., Inc.

Ip, H. H. S., Law, K. C. K., & Kwong, B. (2005). Cyber composer: Hand gesture-driven intelligent music composition and generation. In *11th international multimedia modelling conference* (p. 46-52).

Jarret, & Day. (2008). *Music composition for dummies*. Wiley Publishing.

Jeffries, R., Miller, J. R., Wharton, C., & Uyeda, K. (1991). User interface evaluation in the real world: a comparison of four techniques. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 119–124).

Jensen, C. R. (1992). A theoretical work of late seventeenth-century muscovy: Nikolai diletskii's "grammatika" and the earliest circle of fifths. *Journal of the American Musicological Society*, *45*(2), 305-331. Retrieved from `http://www.jstor.org/stable/831450`

John Wiley & Sons Inc. (2016). *How to use volume in your piano playing.* `http://www.dummies.com/art-center/music/piano/how-to-use-volume-in-your-piano-playing/`. (Accessed: 2017-10-07)

John Wiley & Sons Inc. (2017). *Musical punctuation: Bar lines and measures.* `http://www.dummies.com/art-center/music/piano/musical-punctuation-bar-lines-and-measures/`. (Accessed: 2017-10-07)

Kammer, D., Wojdziak, J., Keck, M., Groh, R., & Taranko, S. (2010). Towards a formalization of multi-touch gestures. In *ACM international conference on interactive tabletops and surfaces* (pp. 49–58).

Karat, J., McDonald, J. E., & Anderson, M. (1986). A comparison of menu selection techniques: touch panel, mouse and keyboard. *International Journal of Man-Machine Studies*, *25*(1), 73 - 88.

Kazi, R. H., Chevalier, F., Grossman, T., Zhao, S., & Fitzmaurice, G. (2014). Draco: bringing life to illustrations with kinetic textures. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 351–360).

Kazi, R. H., Igarashi, T., Zhao, S., & Davis, R. (2012). Vignette: interactive texture design and manipulation with freeform gestures for pen-and-ink illustration. In *Proceedings of the SIGCHI conference on human factors in computing systems* (pp. 1727–1736).

Kemper, K. J., & Danhauer, S. C. (2005). Music as therapy. *South Med J*, *98*(3), 282–8.

Kikuchi, J., Yanagi, H., & Mima, Y. (2016). Music composition with recommendation. In *Proceedings of the 29th annual symposium on user interface software and technology* (pp. 137–138).

Kikuchi, M., & Osana, Y. (2014). Automatic melody generation considering chord progression by genetic algorithm. In *Nature and biologically inspired computing (nabic), 2014 sixth world congress on* (pp. 190–195).

Kivy, P. (1993). *The fine art of repetition: Essays in the philosophy of music.* Cambridge University Press.

Knoder, J. (2017a). *Finale printmusic 2014 review.* `http://www.toptenreviews.com/software/home/best-music-notation-software/print-music-review/`. (Accessed: 2017-10-06)

Knoder, J. (2017b). *Sibelius first review.* `http://www.toptenreviews.com/software/home/best-music-notation-software/sibelius-student-review/`. (Accessed: 2017-10-06)

Krauss, R. M. (1998). Why do we gesture when we speak? *Current Directions in Psychological Science*, *7*(2), 54-60.

Kurtenbach, H. (1996). Gestures in human-computer communication. In Laurel (Ed.), (Vol. 1, p. 309-317). Addison-Wesley Publishing Company.

Laitenberger, O. (2002). A survey of software inspection technologies.

Laurel (Ed.). (1996). *The art of human-computer interface design.* Addison-Wesley Publishing Company.

Levitt, D. A. (1992). A representation for musical dialects. In *Machine models of music* (pp. 455–469).

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of psychology*.

Loy, G. (2011). *Musimathics: the mathematical foundations of music* (Vol. 1). Mit Press.

MacDonald, D. (2017). *Komp is a beautiful and ambitious new scoring app for ipad.* https://www.scoringnotes.com/reviews/komp-beautiful-ambitious-new-scoring-app-ipad/. (Accessed: 2017-10-10)

MacDonald, R., Byrne, C., & Carlton, L. (2006). Creativity and flow in musical composition: An empirical investigation. *Psychology of Music*, *34*(3), 292–306.

MacKenzie, I. S. (1992). Fitts' law as a research and design tool in human-computer interaction. *Human-computer interaction*, *7*(1), 91–139.

MacKenzie, I. S., Sellen, A., & Buxton, W. A. S. (1991). A comparison of input devices in element pointing and dragging tasks. In *Proceedings of the sigchi conference on human factors in computing systems.* New York, NY, USA: ACM.

MakeMusic, I. (2017). *musicXML.* Retrieved from http://www.musicxml.com/

Mandelbaum, M. (2014). *Design principles.* Retrieved from http://learndesignprinciples.com/index.html

Martin, R. (2001). Noise power spectral density estimation based on optimal smoothing and minimum statistics. *IEEE Transactions on speech and audio processing*, *9*(5), 504–512.

Mathers, N., Fox, N., & Hunn, A. (2007). Surveys and questionnaires. *Trent: RDSU*.

McDonald, S., & Petrie, H. (2013). The effect of global instructions on think-aloud testing. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 2941–2944). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/2470654.2481407 doi: 10.1145/2470654.2481407

McLeod, S. (2008). *Likert scale.* Retrieved from https://www.simplypsychology.org/likert-scale.html

McNeill, D. (2007). *Hand and mind: what gestures reveal about thought.* University of Chicago Press.

Meyer, L. B. (1989). *Style and music: Theory, history, and ideology.* University of Chicago Press.

Miller, C. R. (1984). Genre as social action. *Quarterly Journal of Speech*, *70*(2), 151–167.

Miller, M. (2005). *The complete idiot's guide to music theory.* Penguin.

Miranda, E. R., Kirke, A., & Zhang, Q. (2010). Artificial evolution of expressive performance of music: An imitative multi-agent systems approach. *Computer Music Journal*, *34*(1), 80-96.

Mo, Z., Lewis, J. P., & Neumann, U. (2005). Smartcanvas: A gesture-driven intelligent drawing desk system. In *Proceedings of the 10th international conference on intelligent user interfaces* (pp. 239–243). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/1040830.1040881 doi: 10.1145/1040830.1040881

Muntigl, P. (2004). Modelling multiple semiotic systems: The case of gesture and speech. *Perspectives on multimodality*, 31–50.

Nespoulous, J.-L., Perron, P., & Lecours, A. R. (2014). *The biological foundations of gesture: Motor and semiotic aspects*. Psychology Press.

Nettheim, N. (1992). On the spectral analysis of melody. *Interface*, *21*(2), 135-148.

Newsted, P. R., Huff, S. L., & Munro, M. C. (1998). Survey instruments in information systems. *MIS quarterly*, *22*(4), 553.

Nielsen, J. (1994). Usability inspection methods. In *Conference companion on human factors in computing systems.* New York, NY, USA: ACM.

Nielsen, J. (2013). *10 usability heuristics for user interface design.* Retrieved from http://www.designprinciplesftw.com/collections/10-usability-heuristics-for-user-interface-design

Nielsen, J., & Molich, R. (1990). Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI conference on human factors in computing systems* (pp. 249–256).

Norman, D. A. (2010). Natural user interfaces are not natural. *interactions*, *17*(3), 6–10.

Otter, S. (2017). *Finale notepad.* https://www.techsupportalert.com/content/finale-notepad.htm-0. (Accessed: 2017-10-06)

Owens, J. A. (1998). *Composers at work: the craft of musical composition 1450-1600*. Oxford University Press.

Pasquier, P., Eigenfeldt, A., Bown, O., & Dubnov, S. (2017, jan). An introduction to musical metacreation. *Comput. Entertain.*, *14*(2), 2:1–2:14.

Pavlovic, V. I., Sharma, R., & Huang, T. S. (1997). Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on pattern analysis and machine intelligence*, *19*(7), 677–695.

Peter, F. (2016). *How to read music: Dynamic markings.* `https://piano-ology.org/2016/10/22/how-to-read-music-notation-dynamic-markings/`. (Accessed: 2017-10-07)

Pettersson, R. (2002). *Information design: An introduction* (Vol. 3). John Benjamins Publishing.

Poliner, G. E., Ellis, D. P. W., Ehmann, A. F., Gomez, E., Streich, S., & Ong, B. (2007, May). Melody transcription from music audio: Approaches and evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, *15*(4), 1247-1256.

Porta, M. (2007, Oct 01). Human–computer input and output techniques: an analysis of current research and promising applications. *Artificial Intelligence Review*, *28*(3), 197–226. Retrieved from `https://doi.org/10.1007/s10462-009-9098-5` doi: 10.1007/s10462-009-9098-5

Porter, A., Siy, H., & Votta, L. (1996). A review of software inspections. *Advances in Computers*, *42*, 39–76.

Radford, L., Edwards, L., & Arzarello, F. (2009). Introduction: beyond words. *Educational studies in mathematics*, *70*(2), 91–95.

Rameau, J.-P. (1722). *Treatise on harmony.* Imp. de J.-B.-C. Ballard.

Ramirez, R., Hazan, A., Maestre, E., & Serra, X. (2008). A genetic rule-based model of expressive performance for jazz saxophone. *Computer Music Journal*, *32*(1), 38–50.

Rautaray, S. S., & Agrawal, A. (2015). Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review*, *43*(1), 1–54.

Read, G. (1969). *Music notation: A manual of modern practice* (2nd ed.; Tech. Rep.).

Riso, C. (2016). *Basic music theory - 1st post - notes, rests, and note particulars.* `http://cathyriso.blogspot.com/2016/08/basic-music-theory-1st-post.html`. (Accessed: 2017-09-26)

Rivadelo, R. F. (1986). *Fundamentals of music.* National Book Store.

Rogers, Y., Sharp, H., & Preece, J. (2011). *Interaction design: beyond human-computer interaction*. John Wiley & Sons.

Rose, T. (2014). *Reading music lesson 34 time signature.* `http://www.musicreadingsavant.com/reading-music-lesson-34-time-signature/`. (Accessed: 2017-10-18)

Roth, W.-M. (2001). Gestures: Their role in teaching and learning. *Review of Educational Research*, *71*(3), 365-392. doi: 10.3102/00346543071003365

Rothgeb, J. (1975). Strict counterpoint and tonal theory. *Journal of Music Theory*, *19*(2), 260–284.

Russell, S., Norvig, P., & Intelligence, A. (1995). A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, *25*, 27.

Ryynnen, M. P., & Klapuri, A. P. (2008). Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, *32*(3), 72-86.

Salamon, J., & Gomez, E. (2012, Aug). Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, *20*(6), 1759-1770.

Sauro, J. (2009). Estimating productivity: composite operators for keystroke level modeling. *Human-Computer Interaction. New Trends*, 352–361.

Sauro, J. (2011). *Measuring task times without users.* Retrieved from `https://measuringu.com/predicted-times/`

Schulze, W., & van der Merwe, B. (2011). Music generation with markov models. *IEEE MultiMedia*, *18*(3), 78-85.

Shackel, B. (1959). Ergonomics for a computer. *Design*, *120*, 36–39.

Shedroff, N. (1999). Information interaction design: A unified field theory of design. *Information design*, 267–292.

Sinha, G., Shahi, R., & Shankar, M. (2010). Human computer interaction. In *Emerging trends in engineering and technology (icetet), 2010 3rd international conference on* (pp. 1–4).

Spencer, P. (1996). *Music theory for non-music majors*. Florida State University.

Stamp, J. (2013). *The evolution of the treble clef.* `http://www.smithsonianmag.com/arts-culture/the-evolution-of-the-treble-clef-87122373/`. (Accessed: 2017-09-26)

Steels, L. (1993). The artificial life roots of artificial intelligence. *Artificial life*, *1*(1_2), 75–110.

Stoica, P., & Moses, R. L. (1997). *Introduction to spectral analysis* (Vol. 1). Prentice hall Upper Saddle River.

Tidwell, J. (2010). *Designing interfaces: Patterns for effective interaction design.* " O'Reilly Media, Inc.".

Tokui, N., Iba, H., et al. (2000). Music composition with interactive evolutionary computation. In *Proceedings of the 3rd international conference on generative art* (Vol. 17, pp. 215–226).

Travis, C., & Murano, P. (2014). A comparative study of the usability of touch-based and mouse-based interaction. *International Journal of Pervasive Computing and Communications*, *10*(1), 115–134.

Unehara, M., & Onisawa, T. (2001). Composition of music using human evaluation. In *The 10th IEEE international conference on fuzzy systems, 2001* (Vol. 3, pp. 1203–1206).

Unehara, M., & Onisawa, T. (2005). Music composition by interaction between human and computer. *New Generation Computing*, *23*(2), 181–191.

Velardo, V., & Jan, S. (2016). *Study day on computer simulation of musical creativity.* MIT Press.

Voss, R. F., & Clarke, J. (1978). 1/f noisein music: Music from 1/f noise. *The Journal of the Acoustical Society of America*, *63*(1), 258–263.

Wesp, R., Hesse, J., Keutmann, D., & Wheaton, K. (2001). Gestures maintain spatial imagery. *The American Journal of Psychology*, *114*(4), 591-600. Retrieved from `http://0-www.jstor.org.lib1000.dlsu.edu.ph/stable/1423612`

Westerman, W., Elias, J. G., & Hedge, A. (2001). Multi-touch: A new tactile 2-d gesture interface for human-computer interaction. In *Proceedings of the human factors and ergonomics society annual meeting* (Vol. 45, pp. 632–636).

Willoughby, D. (1971). Comprehensive musicianship and undergraduate music curricula..

Yu, B., & Voll, K. (2011). Probing student problem solving skills in mathematical induction using a scenario based think aloud protocol. In *Proceedings of the 16th western canadian conference on computing education* (pp. 33–37). New York, NY, USA: ACM. Retrieved from `http://doi.acm.org/10.1145/1989622.1989631` doi: 10.1145/1989622.1989631

Yüksel, A. Ç., Karci, M. M., & Uyar, A. Ş. (2011). Automatic music generation using evolutionary algorithms and neural networks. In *2011 international symposium on innovations in intelligent systems and applications (INISTA)* (pp. 354–358).