Functional Requirements Document

Musicalgorithms 3.0

Absolute Harmonics

Anthony Della
Dylan Paulus
Michelet Chery
Kristi Marks

Contents

Sections which were outside of the scope of this project, or were not relevant within the given time frame are not included.

-Not Included

23. Risks
    -Not Included
24. Costs
    -Not Included

25. User Documentation and Training
26. Waiting Room (this is where ideas for FUTURE VERSIONS should be included)
27. Ideas for Solutions

1. The Purpose of the Project

1a. The User Business or Background of the Project Effort

Content

The Web-based application found on the previous Musicalgorithms site was designed for exploring algorithms in an interdisciplinary environment. The Web site contains interactive tools, which provide a unique learning experience for users, regardless of their musical training. Students of music composition can explore algorithmic composition, while others can create musical representations of models for the purpose of aural interpretation and analysis. Here, the algorithmic process is used in a creative context so that users can convert sequences of numbers into sounds. For example, the Web-based application at this site can provide aural representations of Pi, DNA sequences, or Dow Jones Industrial averages. This learning environment allows users a hands-on approach to creating music by means of conceptualization.

Musicalgorithms 2.0 was designed with the intent to update the functionality of the original website by adding new features such as “scale” filtering and the option to process two tracks of music, referred to as “voices.”

Musicalgorithms3.0 will have the functionality of Musicalgorithms2.0, using html5/css with javascript behind. Another goal of Musicalgorithms3.0 is to implement a new compression algorithm “morph”. Morph will be used specifically when there are 2 or more voices, and interpolate between them.

Motivation

Musicalagorithms 2.0 was written in Java in 2012. Since then, many browsers have added safety precautions to avoid allowing unsafe Java code to run without user permission. In addition to unfriendly pop-ups and unnecessary warnings, not all browsers have support for Java applets which limits the user base. The motivation for version 3.0 was to make the application more user friendly and accessible by using HTML5 to improve cross-browser compatibility.

Form

The original website is currently (as of this writing) hosted at the following location:
http://musicalgorithms.org/

Musicalgorithms 3.0 was meant to be as similar to version 2.0 as possible while adding functionality and improving usability.

1b. Goals of the Project

Content

Musicalgorithms 3.0 intends to improve the usability and cross-browser support of version 2.0. The software will be written in HTML5 to accomplish these goals. Version 3.0 will have a new morphing algorithm and will be written to allow for more algorithms and features to be added in the future. The website will have a proper login system with additional benefits for users who choose to sign up.

Motivation

There are several motivational factors behind this project. The first one is to address the limited usability of the current site. Version 2.0 functions well enough, but because of new browser security standards and frequent Java updates unfriendly pop-ups can cause unfamiliar users to resist using the site. In addition to scaring away the user, the application does not work the same on all browsers or operating systems. Particularly noticeable is the consistent crashing and irregularities when using Safari. Since the site is already written in HTML5 and is supported by most

browsers, version 3.0 will be much more user-friendly and eliminate worrying whether an update will crash the application or not.

Examples

Musicalgorithms 3.0 users will not experience unfriendly pop-ups, fatal warnings, and unresponsive browsers.

Measurement

Our success at improving version 2.0 will be measured by the increase of users being able to efficiently use the site and create content compared to the current site.

Form

Purpose: To build upon the success of the original Musicalgorithms website by adding useful functionality, and to create a solid software foundation for this project. The software foundation is intended to be expandable for future projects.

Advantage: Users of Musicalgorithms 2.0 will have the advantage of having their data sets more accurately represented by our compression algorithm. They will have the advantage of being able to output their results to a scale and a key. Future teams will have the advantage of not having to start from scratch to build upon this project, or create new ones from scratch.

Measurement: The measurement of our success will be the output generated by the users, and their satisfaction with the results.

2. The Stakeholders

2a. The Client

Dr. Jonathan Middleton.

Motivation

The motivation is to provide Dr. Middleton with an updated version of his website that will enrich the experience of users who visit his site. The new site will provide enhanced functionality to Dr. Middleton and his students who may be interested in creating original musical compositions based on the algorithmic output generated by Musicalgorithms 3.0.

Considerations

Considerations involve balancing the needs of our client with the limitations of time and technology available. To handle this we are discussing the direction of our project, and the goals we would like to achieve with it. We are in agreement with our client that we would like to re-create Musicalagorithms 2.0 in HTML5, implement a third voice, and a new algorithm for data manipulation dubbed “morph”.

Form

This project is the brainchild of Dr. Jonathan Middleton. He has the final say on how the project is implemented and distributed.

2b. The Customer

Content

Dr. Jonathan Middleton is the main customer, as well as the client. Additional customers will be the users who visit the website and make use of the tools. The archetypical visitor to the website will generally have some academic interest in the tools provided, either musically or from the standpoint of providing data sets to be manipulated into musical output.

Motivation

Dr. Middleton has had a successful run with the original website, generating positive feedback from users around the world. His motivation for the re-design of Musicalgorithms 2.0 is to improve

usability and make the site more sensible and user-friendly.

Form

Dr. Middleton is responsible for signing off on all features to be included in the project, as well as prioritizing which features need to make it in this round and which ones can be addressed at a later time.

2c. Other Stakeholders

Content

- Client/Sponsor (refer to 2a)
- Customer (refer to 2b)
- Members of the public
- Users of the current system
- Musicians
- Composers
- Scientists
- Economists
- Students
- Instructors
- Schools/ Universities

Motivation

Recognizing each of the potential stakeholders provides inspiration and drive to ensure as many users as possible have easy access and pleasant experience when using the application.

2d. The Hands-On Users of the Product

Content

- Client/Sponsor (refer to 2a)
  User role: Responsible for the successful design and implementation of the Musicalgorithms 2.0 website.
  Subject matter experience:
  Jonathan Middleton, DMA
  Professor of Theory and Composition
  Department of Music
  Technological experience: Master

- Customer (refer to Client/Sponsor)

- Members of the public
  User role: Users with casual interest in the subject matter
  Subject matter experience: Little to none
  Technological experience: Novice

- Users of the current system
  User role: Already established an interest in the system. Likely to have a better understanding of what to expect.
  Subject matter experience: Varies depending on return user's level of involvement.
  Technological experience: Journeyman

- Musicians
  User role: Interested in the musical performance aspects of the project.
  Subject matter experience: Experienced in playing music, by ear or by reading it from sheet music.
  Technological experience: Novice to Master

- Composers
  User role: Interested in the musical composition aspects of the project.
  Subject matter experience: Experienced in music composition
  Technological experience: Novice to Master

- Scientists
  User role: Produce original data sets based on individual area

of expertise. Interested in seeing the data sets converted to music.
Subject matter experience: Minimal expected experience with the Musicalgorithms project.
Technological experience: Experts in their own field, minimal experience expected with the Musicalgorithms project.

- Economists
  User role: Produce original data sets based on individual area of expertise. Interested in seeing the data sets converted to music.
  Subject matter experience: Minimal expected experience with the Musicalgorithms project.
  Technological experience: Experts in their own field, minimal experience expected with the Musicalgorithms project.

- Students
  User role: Varying based on their field of study
  Subject matter experience: Varying based on their field of study
  Technological experience: Varying based on their field of study
- Instructors
  User role: Varying based on their field of study
  Subject matter experience: Varying based on their field of study
  Technological experience: Varying based on their field of study

- Schools/ Universities
  User role: Provide students to make use of the website
  Subject matter experience: NA
  Technological experience: NA

Motivation

Each of the users who make use of the Musicalgorithms 3.0 website will do so for their own reasons. The unifying factor is a curiosity in generating music through algorithmic means by starting with either one of the provided data sets, or by providing their own data set to be transformed into musical output.

Examples

The most immediate user of the website will likely be students of music studying at Eastern Washington University. These students may come to the website out of academic curiosity, or they may be directed to visit by their instructors.

Another group of users will be scientists and academics from around the world. This group will be interested in seeing original sets of data turned into algorithmically generated musical compositions. The original sets of data may be based on their own work, or on work coming from fields that they are interested in studying.

There will be casual users as well. This group is not expected to come to the website with high-expectations of what they will find there, rather their visit will be based on idle curiosity in some aspect of the process, from data set generation to algorithmic compression to scaling the output and finally hearing the results play back from their speaker system.

2e. Personas

Persona number 1:

Abigail Goslinger studies patterns of bacterial mutation over a given period of time. She has discovered that the bacterium have patterns of traits that evolve at given intervals over the course of a population sample's life cycle. She has charts and spread sheets which depict the changing patterns numerically, but staring at numbers all day has gotten a bit dull for her. Now she has just discovered the Musicalgorithms 3.0 website. She is curious to listen to how the patterns of bacterial evolution sound musically. After giving the pre-defined datasets a whirl she copies several pages of her own output into the Musicalgorithms 3.0 custom input box. First she runs it through the Div compression algorithm, then the Mod. Neither of these algorithms produce output that she finds to be accurately representative of her dataset. Next she tries the new Logarithmic compression algorithm and is excited to hear patterns which match the data she has been looking at visually for the past few years. Inspired by this aural reflection of her work, she jumps back in with renewed vigour and enthusiasm, eager to hear what her next set of findings might sound like.

Persona number two:

Gary Pemberson has been a struggling

composer for years. Some of his work has been published for use as commercial jingles, and a few TV spots. His output has gotten him some attention in the industry, and he feels like he could be on to his next professional breakthrough, but he is stumped by a particularly stubborn bout of writers block. Scouring the internet for ideas Gary comes across Musicalgorithms 3.0 and decides to give it a whirl. After several attempts at generating output he finally stumbles upon an interesting composition by combining two voices generated in contrasting scales and keys from two different datasets. This results in an “aha moment” giving Gary enough impetus to get to work creating his next creative masterpiece.

Persona number three:

MusicLover88 is a frequent visitor to the website known as reddit. He enjoys creating electronic music compositions by taking other users work and remixing it until it resembles the theme song of a popular space themed sports movie from 1996 starring Bugs Bunny. He has been struggling to find the right song to work with because he has exhausted his entire music library and has been reduced to remixing the theme song with itself to little success. Then, one day, he stumbles upon a link to Musicalgorithms.org posted by xXg_PeMbErSoNXx, a successful commercial jingle artist who credits his breakthrough and latest masterpiece to the site. There he finds a tool to create an unlimited amount of tunes by converting his remixes into numerical form and running it through the musicalgorithms 3.0 application. Now he can spend the rest of his days uploading his custom made content to reddit while crediting his success to Jonathan Middleton himself for his creation that provided him with the ability to never run out of music to play with.

2f. Priorities Assigned to Users

●Key users: Dr. Middleton, his students, associates and composers who are interested in algorithmic music composition.

●Secondary users: Academics and Scientists. This group generates unique datasets that they are interested in hearing algorithmically converted into music.

●Unimportant users: Random visitors to the website. These people will have varying degrees of interest in the product, and a wide variety of skills and backgrounds that they will bring to it.

2g. User Participation

- Key users will make the most use of the website, exploring and creating original musical compositions based on the capabilities of the software and the uniqueness of the datasets provided.

- Secondary users provide inspiration for additional features, such as creating sets of patterns from a given source (such as DNA sequencing.) These people are expected to have some interest in the capabilities of the website and what it can produce, but they most likely will not spend a great deal of time working with the output given by the website. This group could provide interesting feedback through the website forums.

- Unimportant users. The general population of casual users who happen to come across the website. Feedback given by this group may be more related to the mechanics of the user experience than the conceptual part of the project as related to algorithmic music composition based on a variety of datasets and compression algorithms.

2h. Maintenance Users and Service Technicians

Maintenance and upkeep of the website will be overseen by Dr. Middleton. He may seek help from future teams of students to update the site and provide content.

3. Mandated Constraints

3a. Solution Constraints

The Musicalgorithms 3.0 application must implement all the features in the previous version while being written in HTML5. Version 3.0 must not introduce any new problems to uasability and must improve on the previous version's usability problems. The program must be expandable for future development.

Description: This product will be able to save results to a valid MIDI file.
Rationale: Users should be able to save and share their work. MIDI is a universal tool which can be used in many different environments.
Fit Criterion: MIDI output should meet the MIDI standard

Description: MIDI notes must be within the piano range.
Rationale: Notes outside of the piano range are of little use to anyone interested in composing music that can be played by standard instruments.
Fit Criterion: The generated notes must be within the MIDI range of 21 to 108.

Description: Program will be open source and expandable for future development.
Rationale: Future teams must be able to use the code from this project to enhance the capabilities of the website, as well as to aid in the creation of new projects.
Fit Criterion: JavaScript Source code will be provided.

Description: Program will be hosted on the internet.
Rationale: This is the best way to distribute the program to the broadest possible user base.
Fit Criterion: The program will be functionally accessible from a variety of standard web browsers.

3b. Partner or Collaborative Applications

Musicalgorithms 3.0 makes extensive use of the MIDI.js library. In addition to this, the playing piano graphic from the original website has been modified to work with the new implementation

of the project. Project constraints come from both of these sources. The MIDI.js implementation is a given, but the playing piano graphic is a style feature which adds user feedback, but limits the current implementation of the Musicalgorithms 3.0 engine to only 16 tracks. Future implementations of the package may have time to find a way around this limitation to be able to add additional tracks to the project.

The success of the software also depends on the user having JavaScript enabled in their browser.

Web implementation means that Musicalgorithms 3.0 depends on the stability of users' web browsers. It has been tested with current mainstream browsers.

3c. Off-the-Shelf Software

The Off-The-Shelf software implemented in Musicalgorithms 3.0 includes HTML 5, Jquery, Bootstrap, and MIDI.js library.

3d. Anticipated Workplace Environment

Users are expected to use the software on personal computers, either in their home or at the office. The nature of the output requires that they either have speakers or headphones available. The user will expect audio output, so it will be up to them to be aware of environmental noise constraints, such as using the software in a library setting.

3e. Schedule Constraints

The deadline is fixed by the end of the second quarter of the two quarter Senior Project schedule. The first quarter was divided into research and implementation. The second quarter was divided into further implementation, testing and deployment of the software.

Because the deadlines were set in stone the project was approached from the start to be small enough to complete in time, as well as making sure to create code that can be used by other teams in the future- allowing them to save time in research and setup that our team had to handle.

Some features were left out of this version in order to meet the project deadline. One key feature that the client would like to see implemented in the future was to give users the ability to print sheet music from their output directly on the website. Leaving this feature out was seen as being only an inconvenience, since users can save MIDI files of their work which can then be used in a variety of freely available MIDI sheet music printing software.

The link below directs to a websites that converts midi files to sheet music.

http://www.8notes.com/

Another feature that was left out that our client would have loved to see was the ability to play silent notes. Silent notes were left out because we ran out of time.

3f. Budget Constraints

As a student project the budget for this project was pretty sparse. The work and coding was all done at no cost. The client will be responsible for the cost of hosting the project on his/her web.

4. Naming Conventions and Terminology

4a. Definitions of All Terms, Including Acronyms, Used in the Project

Musicalgorithms 3.0: The name of this version of the Musicalgorithms project.

Algorithmic Composition: The process of converting sets of numbers, data sets, into music notation that remains meaningfully representative of the original data set.

Division: This term is used to describe the Division based compression algorithm used to fit data sets into the constraints of the musical output. This is sometimes referred to as Div.

Modulo: This term is used to describe the Modulo based compression algorithm used to fit data sets into the constraints of the musical output. This is sometimes referred to as Mod.

Logarithmic: This term is used to describe the new compression algorithm which is based on a Logarithmic algorithm. This is sometimes referred to as Log.

Pitch Input: This is used to refer to the raw data sets before they are converted to music notation.

Duration Input: This refers to the data set that is used to determine the length of each note. In other words, it gives your music rhythm.

Pitch Mapping: This refers to the step of compressing the Pitch Input data to fit within the MIDI piano range.

Duration Mapping: This refers to the step of compressing the raw Duration Input data to a range of pre-defined note duration values.

Scale Options: This refers to the step of converting the Pitch Mapped data into a pre-defined scale and key or morph toward Beethoven's 9th symphony.

Each of the scales is based on a numerical algorithm which locks every note into a simple scale based range.

The scales as defined in the current version of the project are:

Blues
Chromatic
Major
Minor
Pentatonic 1
Pentatonic 2
Wholetone
Morph

Software naming conventions used follow that of standard Java naming conventions.

Classes:
Class names should be nouns, in mixed case. Try to keep your class names simple and descriptive. Use whole words-avoid acronyms and abbreviations (unless the abbreviation is much more widely used than the long form, such as URL or HTML).
Interfaces:
Interface names should be capitalized like class names.

Methods:
Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized.
Variables:
Except for variables, all instance, class, and class constants are in mixed case with a lowercase first letter. Internal words start with capital letters. Variable names can start with underscore _ or dollar sign $ characters. Variables that start with $ character refers to Jquery objects.

Variable names can be short and meaningful. The choice of a variable name should be mnemonic- that is, designed to indicate to the casual observer the intent of its use. One-character variable names should be avoided except for temporary "throwaway" variables. Common names for temporary variables are i, j, k, m, and n for integers; c, d, and e for characters.

5. Relevant Facts and Assumptions

5a. Relevant Facts

Content

Our project relies on integer to MIDI conversion properties. Its success also relies heavily on our ability to embed our application into a web site and have it run across several browsers.

5b. Business Rules

Not discussed in relation to this project.

5c. Assumptions

Content

Our assumptions are that the MIDI standard will remain constant and that our application and website will be able to persist and

correctly operate across new versions of browsers. We assume that all parties that will be interested in this project will have a compatible system with JavaScript enabled.

We assume that the product will not print sheet music at this time, though future versions may add this functionality in later. Musicalgorithms 3.0 will save MIDI files.

6. The Scope of the Work

6a. The Current Situation

The program will exist in a JavaScript based application that is intended for web hosting on the new music algorithms web site. The project will be in HTML 5 technology. The input will be pre-defined patterns of data, or variable user-input. The output will be a midi file, and sound from the program.

The big picture is a user runs the application, enters in some mathematical model they would like to hear, and then is able to hear what it sounds like, and see it depicted.

6b. The Context of the Work

The Musicalgorithms project combines disciplines from the Computer Science department, the Music department and the Math department. The Computer science component is integral to the project and discussed in detail throughout this document. The Music department provides the initial inspiration behind the project, through the work of Dr. Jonathan Middleton. Where his previous efforts in computer assisted algorithmic composition have focused primarily on the algorithmic compression of data sets, his focus with this project has branched further out into music theory with the application of scale filtering, and an initial foray into multi track recording.

Mr. David Goering provided the input from the Math department by providing mathematical solutions for the scaling used in Musicalgorithms 2.0. He also worked out the mathematical foundation behind the algorithm which became the new Logarithmic compression function.

6c. Work Partitioning

All panels contain a save and load options that allows users to save their work for a latter time. Users also have the option of change views from Full View to Tabbed View.

Welcome panel:

Introductory information about the nature of Musicalgorthms along with project director and developers.

Pitch Input panel:

Users chose predefined raw data sets or enter custom input. Users chose number of notes to process for predefined datasets.

Duration Input panel:

Users chose predefined raw data sets or enter custom input.

Pitch Mapping panel:

User choses method to compress the raw Pitch data, as well as the range to apply it to (as a subset of the piano range.)

User has the option to replace one value with another.
User has the option to add silence.

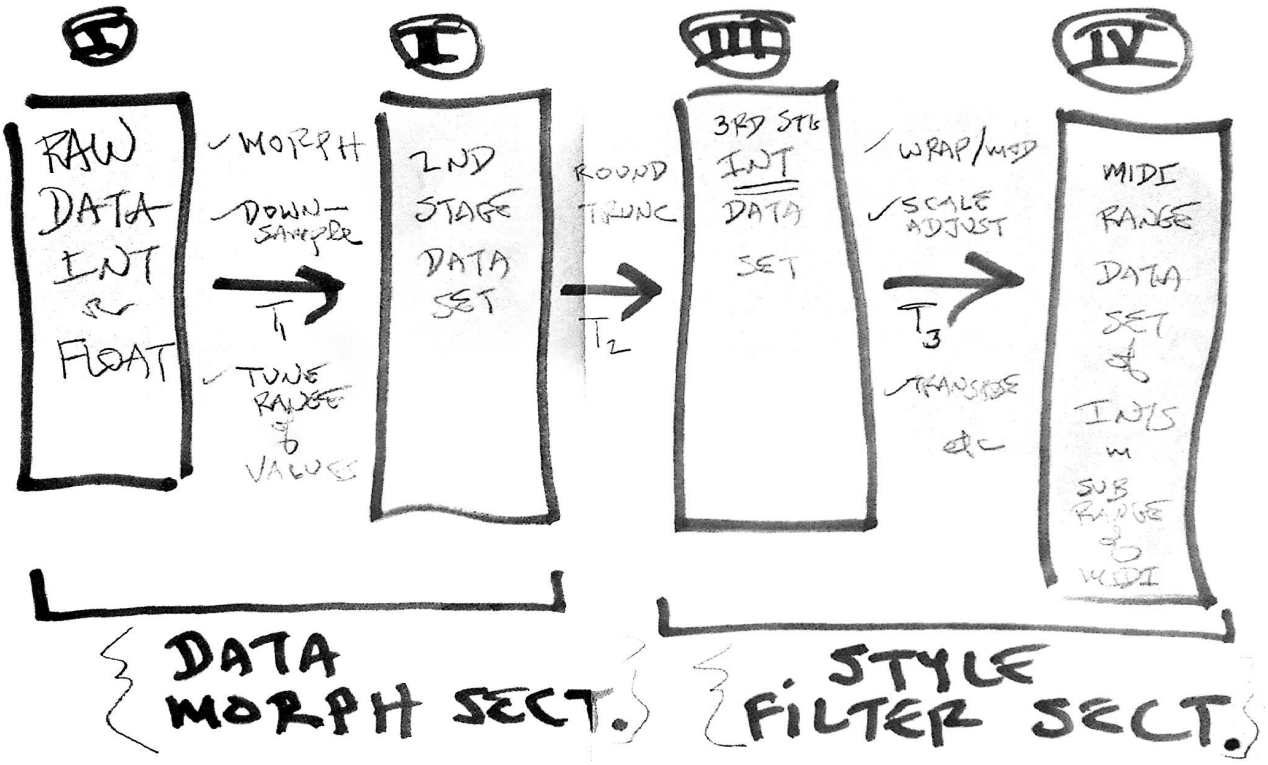
Duration Mapping panel:

User choses method to compress raw Duration data, as well as the range of sustain values from 0 to 9.
User has the option to replace one value with another.

Scale Options panel:

User can select a scale option and key or they can decide to morph their scale to Beethoven's 9th Symphony.

Early work partitioning diagram:



8. The Scope of the Product

8a. Product Boundary

Our projects primary goal is to convert various sets of data into audible music. We accept input in various forms such as integer data, decimals, negatives, and in the future perhaps other types of data such as binary and hex. These datasets can originate from text files, or scientific projects, or a variety of mathematical formula. After obtaining this data we implement various algorithms to recompile the data into a new data set from which we can transform it further. These initial algorithms convert the raw data into a useable set of data by our engine based on a given midi range. We can use a number of methods to distribute data to a specific range such as modulo, division, and logarithms. After filtering the data through these transformation algorithms, we process the data further to fit it into specific patterns. These patterns are designed to shift data into musical scales based on a given key on the piano scale. In future revisions we may include a greater variety or scaling options, more complex scales, or scales on different instruments.

After converting the data into scaled musical notes, we can apply further filtering to add new effects or make it play in a certain fashion using timing, pitch bend, multiple ranges, multiple instruments or multiple tracks. After compiling all the necessary data to create the music file, it is sent to a midi construction class. This class uses the various data sets to create a midi file based on the notes, durations, instruments, tracks, and other factors we wish to include in its formation. The end result will ideally be a musical representation of the original data set.

9. Functional and Data Requirements

9a. Functional Requirements

• GUI for input of data values and sets

• Interface between GUI and music engine

• Music engine to contain a package of classes that manipulate data and create midi files

• Data set (Array) to contain data

• Transformation Classes which implement algorithms to transform data to a specific range

• Scaling Classes which modify data sets to specified musical scales

• Midi Maker Classes which convert given data sets into playable midi files

Non-functional Requirements

The following sections 10-17 describe the non-functional requirements. The form of these requirements is the same as for the functional requirements as described above.

10. Look and Feel Requirements

10a. Appearance Requirements

Content

Currently the look and feel requirements are minimal. The goal is to have a user-friendly applet graphical user interface that is easy to interact with and perhaps even entertaining. The applet should look neat and professional.

The play feature needs to make use of the playing piano graphic from the original website.

The user interface needs to be technical enough to handle each of the features outlined for this project, but simple enough for the average computer literate user to understand. To facilitate this, a default state will be set to ensure that any step can be skipped without compromising the stability of the program and ensuring that the user will be able to play and save at least a basic MIDI file at the end.

Any graphic embellishments will be simple and be contextually relevant to the project.

The only branding requirement as of this writing is that the program splash screen be labelled “Musicalgorithms 2.0.”

10b. Style Requirements

The style needs to reflect the professional academic nature of the product. Clean lines and an intuitive interface should greet the user and not distract from the functional use of the program.

11. Usability and Humanity Requirements

11a. Ease of Use Requirements

Efficiency of use:
The website should be easy enough for the average user, as well

as professionals from varying fields of study to use. The user should be able to enter numerical data and retrieve the results expected.

Ease of remembering:
The website should be easy enough to use that the users do not have to memorize special tools or techniques to obtain results.

Error rates:
The user will be expected to enter data that is formatted in a specific way, such as numbers separated by space or number separated by commas.

Overall satisfaction in using the product:
We are striving to make our product more intuitive than the previous website. There is no competition that we are aware of for our product.

Feedback:
We are laying the groundwork for a new website design that will encourage community involvement, hopefully increasing user feedback. User feedback could be helpful in designing future enhancements to the Musicalgorithms website.

Motivation:
To enhance the current website with additional functionality that meets the expectations of current and future users.

Examples:
The website should be easy for the average computer-literate adult to use.
The website should provide clear instructions for the user to avoid making mistakes.
It is expected that users will come to the site because they are interested in the subject.

11b. Personalization and Internationalization Requirements

The website will be produced in English, no internationalization is planned.
No personal configuration options are being considered at this point.
Music and numbers are universally understood. The design of the

website should be simple enough for international users to use even with limited understanding of the English language.

11c. Learning Requirements

The website should be easy enough for the average user to use with no training. The website will provide information to help curious users learn more about the mathematical algorithms used to generate data sets for this project.

Fit Criterion:
An average user should be able to produce midi results with little to no training.

11d. Understandability and Politeness Requirements

The website shall hide the details of its construction from the user.
The musical and mathematical terms are universal, and should be understood at a basic level by anyone interested in this subject.

11e. Accessibility Requirements

The website turns numerical data into musical output. Partially sighted users should be able to hear the results and deaf users should be able to read the numerical output to see visually how it maps to the keyboard.

12. Performance Requirements

12a. Speed and Latency Requirements

Speed is not critical to the Musicalgorithms 3.0 project, but users will be turned away if the program seems unresponsive to their input. Internet lag time and users individual system performance are beyond our control, but in most web environments, the program should respond quickly to user input.

The interface between the user and the Musicalgorithms 3.0 engine should have a minimal response time. The response should be fast enough to avoid interrupting the user's flow of thought.

Fit Criterion

The product shall respond in less than 1 second for 90 percent of the interrogations. No response shall take longer than 2.5 seconds.

12b. Safety-Critical Requirements

The only safety issue that may arise from use of this product could involve volume levels. The individual user's audio playback environment is beyond the scope of our project.

12c. Precision or Accuracy Requirements

Compression accuracy:

One of the primary goals of the Musicalgorithms 3.0 engine is to be able to accurately reflect a wide variety of numerical data sets in a musical environment. Judging the accuracy of each conversion will be subject to individual user's requirements and familiarity with their individual unique datasets.

While the original Musicalgorithms website did a good job of converting datasets, its accuracy with large datasets was limited. To address this one of the primary goals of Musicalgorithms 2.0 was to create a compression algorithm that would be more accurate in reflecting the data from datasets with a large spread between the minimum and maximum values.

Scale value accuracy:

This area we have more control over. Scale mapping was designed based on hard coded number sets for each scale. Notes are counted from the root key chosen by the user, resulting in whole and half steps that are reflective of the scale. It is understood that this scale data does not accurately reflect real world composition settings, since there is no room for individual

notes to step outside of these pre-defined ranges. With these limitations understood we were able to create an algorithmic solution to the scale mapping problem.

Testing should show that the scales stay within their predefined ranges.

12d. Reliability and Availability Requirements

The product should be reliable enough to satisfy the average users reliability and functional expectations. While the product does not have any safety-critical features, it should be robust enough that the average user is not driven away by excessive lag or instability.

Availability of the product will depend on the user's ability to connect to the product through the internet. Many of the issues involved in the connection between the user and our product remain out of our control, but we will address the issues that we are able to have some control over. To ensure availability of the product from our end the program will be tested with a variety of current internet browsers on a variety of systems.

The product should be available at any hour of the day since users from all around the world are expected to make use of it.

12e. Robustness or Fault-Tolerance Requirements

The product has been extensively tested and crash-inducing bugs have been worked out. While it is not a safety-critical product, the average user has the right to expect a crash-free experience 99% of the time. Musicalgorithms 3.0 has kept the same functionality as Musicalgorithms 2.0 in this regard.

Safety features have been hard-coded into the product to minimize the user's ability to crash our product. Final results are available without any user input at all. Error checking has been used throughout the program to minimize user impact on the system.

12f. Capacity Requirements

The Musicalgorithms 3.0 program has the ability to store a CSV file that contains the userss current data set. Users will also be able to save their work to their own hard drive as a standard MIDI file.

The original website had a small, but engaged audience. Basic webhosting should handle enough traffic for the new website to remain operational at all times. Since we are using HTML, most of the programs calculations is performed on the clients side; therefore, the server will not be overwhelmed.

12g. Scalability or Extensibility Requirements

From the start Musicalgorithms 2.0 was designed with the intention that the current project could be expanded upon, and that it would serve as a foundation for new projects in the future.

With this in mind the core engine code underlying the Musicalgorithms 2.0 user experience has been extensively documented. The code is modular and provides functionality beyond what was able to be realized by the current version of the project within the time frame allowed.

Motivation

We have allowed the capacity for future growth.

12h. Longevity Requirements

The original Musicalgorithms website has been operational since 2004. Musicalgorithms 3.0 should last as long, but the engine may be enhanced and built upon. This means that while the website may remain operational for a decade or more, the capability of the software to evolve means that it may only improve over time.

13. Operational and Environmental Requirements

13a. Expected Physical Environment

It is expected that this product will be used in a home or office environment. The environment is not critical, as long as the user feels comfortable using the website.

For best experience the user's pc should meet the following minimum requirement:

- Pentium 233-megahertz (MHz) processor or faster
- At least 64 megabytes (MB) of RAM (128 MB or higher recommended)
- At least 1.5 gigabytes (GB) of available space on the hard disk
- Keyboard and Mouse or some other compatible pointing device
- Video adapter and monitor with 800 x 600 or higher resolution
- Sound card
- Speakers or headphones

13b. Requirements for Interfacing with Adjacent Systems

The product is expected to run on current web browsers such as Chrome, Mozilla Firefox, Opera, and Internet Explorer.

The user is not expected to download any dependencies to run our software, but users needs to have JavaScript enabled in their browser. A simple google search will demonstrate how one should go about enabling JavaScript in their browser.

The software makes use of MIDI.js, a JavaScript library that plays midi files.

13c. Productization Requirements

The product will be installed on the client’s website where it will be made accessible to users around the world.

In the future the product may be monetized to pay for future research and development of the product.

13d. Release Requirements

Musicalgorithms 1.0 was developed in 2004. Musicalgorithms 2.0 was developed in 2012. No timeline is given for the products lifecycle, which is entirely dependent on the availability of resources and future student teams.

Musicalgorithms 3.0 was developed in 2013 ported Musicalgorithms2.0 from Java to Javascript to solve issues of usability and stability. Future student teams may add new functionalities that may be requested by the client.

14. Maintainability and Support Requirements

14a. Maintenance Requirements

Maintenance and upkeep will need to be handled by individual students, future student teams or be funded through grants or by other special means, so the website is expected to have minimal maintenance requirements after launch. A way may be found to monetize the project in the future, allowing for more maintenance and support options.

14b. Supportability Requirements

Maintenance and upkeep of the website will be overseen by the client and students or future student teams that will be available to help.

14c. Adaptability Requirements

The product is expected to run on any machine that can run current web browsers.

Since Musicalgorithms 3.0 was written in HTML5, the application is very potable. The only requirement to run the application is a modern web browser.

15. Security Requirements

15a. Access Requirements

Currently anyone with access to the internet can use Musicalgorithms 3.0 to manipulate data and save MIDI files to their personal computer.

The website that is being set up to host the software allows users to register a personal account which will allow them to post to the forums. It is hoped that this will create a community of people who engage with each other and come up with new and interesting ways to utilize the software.

Dr. Middleton will be given administrative control over the website.

15b. Integrity Requirements

Web site must remain secure and consistent. The consequences of an insecure sight could be but is not limited to, any of the following:

- Corruption of data
- Breach of personal privacy
- Loss of personal information or data
- Corruption of web hosted services

15c. Privacy Requirements

The product shall make its users aware of its information practices before collecting data from them.
The product shall notify customers of changes to its information policy. The product shall reveal private information only in compliance with the organization's information policy.
The product shall protect private information in accordance with the relevant privacy laws and the organization's information policy.

15d. Audit Requirements

Registration with the website only requires that users create a

unique username and enter a valid email address. Only the system administrator will have access to this information. Future versions of the website may seek to monetize user interaction to pay for future research and development of the project. The Audit requirements may change at such a time as that becomes a viable option.

The intention is to provide security such that a user may not later deny having used the product or participated in some form of transaction using the product.

15e. Immunity Requirements

Website security is currently based on the security provided by the 3rd party hosting service.

17. Legal Requirements

17a. Compliance Requirements

The product was created in an open source environment. The hosting environment, domain and related resources are under the client's ownership and name.

- HTML 5: Open source
  - Javascript
  - Jquery
  - Bootstrap
  - CSS
  - PHP 5

- Drupal: Website generation
- Jonathan's webhost

18. Open Issues

Issue number 1:

Printing sheet music from the Musicalgorithms 2.0 application.

Users are able to print sheet music from the MIDI file they save at the end, but the functionality needs to be added back into the website.

19. Off-the-Shelf Solutions

Issue number 1:

Free off the shelf software exists for printing sheet music from a MIDI file. Perhaps one of these programs can be utilized on the website directly.

25. User Documentation and Training

25a. User Documentation Requirements

Two user training documents are supplied.

The first one is the User Interface Manual, which is intended to help the average user with the website

The second document provided is the Developer Guide. This document provides a general overview of the code behind Musicalgorithms 3.0. It is intended to give future programmers a brief introduction to the software package.

25b. Training Requirements

Future Expansion

With the capability of adding more than four voices, a new GUI interface will need to be design. The current interface is an update from the previous version of this software but design ingenuity is needed to take full advantage of adding more voices.

26. Waiting Room

Several ideas have been floated for future projects that may make use of the Musicalgorithms 3.0 MIDI engine. These include, but are not limited to: porting the program to a mobile environment such as Android or iPhone, or creating games based on the algorithmic composition of music.

27. Ideas for Solutions

From the start the code behind Musicalgorithms 3.0 was designed with the intent that future teams would be able to use it as the foundation for their own work, whether it involved updates to the current application or creating entirely new applications on top of the Musicalgorithms 3.0 base engine.