Multi-User Audio Composition Application

Sanjeev Rao[1], V. Arun[2]

[1]Student, Department of Computer Science & Engineering, SRM University, Chennai, Tamil Nadu-600089
[2]Assistant Professor, Department of Computer Science & Engineering, SRM University, Chennai, Tamil Nadu-600089

Abstract - *Creating compositions digitally can pose pitfalls when attempting to collaborate, as it becomes increasingly difficult to coordinate activities with multiple users; as the number of users increases, each user has to, at best, communicate with the next one ahead-of-time and continue as per schedule, or wait until the next user is available, resulting in unexpected delays, especially if the participants are spread out geographically, and contact via other means is unreliable. The paper describes a desktop application, similar to a DAW (Digital Audio Workstation application) in which 1 or more users can collaborate to create an audio piece, using a combination of wave synthesizers. Users can make their own compositions, or join others', at any time and subsequently make changes to them in order to easily and collaboratively make audio pieces. It is expected that such an application may make collaboration easier in this regard.*

Key Words: Digital audio, application, audio synthesis, software, collaboration, composition, multi-user system

1. INTRODUCTION

Audio composition involves the use of audio synthesizers, typically packaged together along with other effects, which allows a user to create and edit compositions. Typically, a user would only work alone; sometimes, however, multiple users would want to collaborate. This has become more common since the Internet received widespread usage; however, the mode of collaboration has remained much the same – users share the composition (a file which effectively describes the entire composition such that it can be edited by the same software used to create it) with each other. However, this method is rife with pitfalls – it becomes increasingly difficult to coordinate activities with multiple users; as the number of users increases, each user has to, at best, communicate with the next one ahead-of-time and continue as per schedule, or they just wait until the next user is available, which may result in unexpected delays, especially if the participants are spread out geographically, and contact via other means is unreliable. As such, the compositions which are most likely to work effectively and succeed are those where all the participants are in proximity (i.e. in the same geographical region or time zone).

Another challenge, following from the fact that maintaining contact with participants can be unreliable, is the issue of revision control: maintaining the latest revision of the composition becomes an increasing challenge as the number of users grows, especially when a group of (hitherto unreachable) participants have their work overwritten because the latest revision has moved beyond them. In other words, if the composition is not synchronized carefully, then changes may be lost as unreachable participants get left behind. In the event of their return, they often have to scrap their work and start from wherever the composition was, losing opportunities for changes in the time between their departure and (subsequent) return from/to the composition. The application thus described in this paper aims to reduce or solve the aforementioned problems; by maintaining a constant log of users who are in a given composition, and passing control to users in real-time, all users are forced to make changes in a sequential and turn-based manner wherein the latest revision is accessible to each user. Unreachable and inactive participants are skipped until they become active, at which point they can make changes when it is their turn; when users finish or pass their turn, the latest revision (available in a centralized database) will become available to the next user.

2. SURVEY ON SYSTEMS

2.1 Existing System

Software which allows a user to create a composition by intermixing sound from various sources are typically called "Digital Audio Workstations", or DAWs for short. They typically allow the user to work with a variety of effects, and use a multi-track layout for composition. However, intermediate files are typically locked to the particular brand of software, and are not accessible by users with another type of DAW, or sometimes, even a different version.

2.2 Issues with Existing System

The main issue is that of interoperability between DAWs – it is rather poor due to proprietary standards amongst various manufacturers; therefore, in order for collaboration to take place, all parties involved must have a copy of the same software (which is in itself restricted by licenses that range from permissive to very restrictive); furthermore, DAWs do not have an inherent capability for collaboration; typically, the data is stored in files on the disk, and these are shared to the users who are working together. This can lead to problems, especially when people lose contact with other members in a given composition; the composition has to be put on hold until the user passes their file (and hence the cumulative work done) to the next user, or they are skipped and a previous revision is sent to the next user, which results

in partial loss of work. Furthermore, DAWs usually work with instruments, and do not allow for finer control without the help of specialized plugins; a method of generating audio using basic tones, instead of instruments, allows for far more freedom and power. Based on the superposition principle, the audio that is generated can be combined in multiple ways using various tones in order to achieve a composition as the end result; a large group of people working together can be expected to reasonably complete such a task without much difficulty, as opposed to a single user [1]. Sound can be created using the Inverse Fast Fourier Transform (IFFT), the principle behind which is a method to quickly compute the inverse Fourier transform, that is, to combine many frequencies into a resulting signal – in this case, sound – and emit it as an output [2]. Another solution is to use distortion products, which has received little acceptance yet due to their drawbacks (very high amplitude requirement, amongst others), although research is ongoing in the area [3].

2.1. Proposed System

The system thus proposed, is an application which aims to solve the collaboration problem; while it does not possess as vast a feature-set as existing software, it aims to focus on the collaborative aspect, as a system designed with this in mind from the beginning is undoubtedly easier to expand than retrofitting an existing system with its capabilities.

3. SYSTEM MODULES

The system is composed of 4 modules, which are listed below in more detail. These 4 modules are responsible for the audio composition, collaboration and compilation, and are made so as to be more-or-less loosely coupled to achieve independent working, and to make extensions to it easy and smooth.

3.1 Project Creation & Browsing Module

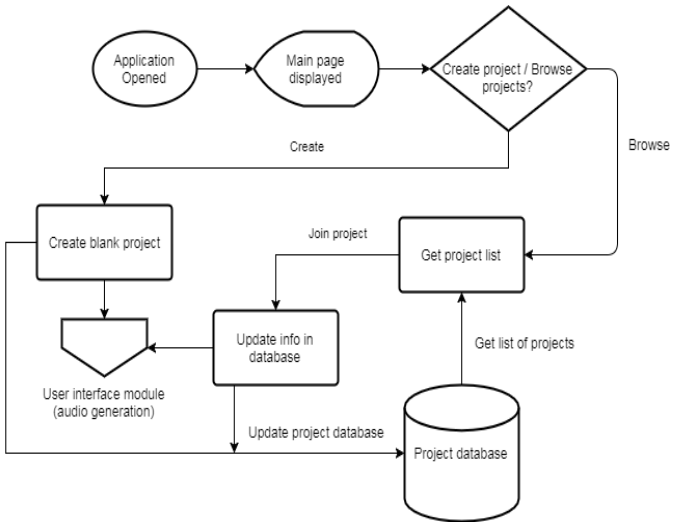

Fig - 1: Flowchart of the project creation & browsing module

The project creation and browsing module is that module which allows the user to create their own, or join an existing, composition. A composition consists of a database which represents all the possible actions that have taken place by all the users who have joined, and made changes to the composition. The underlying database is an SQL database which contains two tables per composition, the data table and the users table.

The data table contains the data for the composition; it includes all the actions that all the users in the composition have taken since the creation of the composition, in sequential order. In the table, each row represents an action (otherwise known as an operation.) A user can commit one or more operations to the table, every time they finish their turn. If they do not make any changes to the composition (i.e. they do not modify, create or delete tones) then no new rows will be added to the table, and the turn will simply be passed to the next user. Hence, on opening (or connecting to) the composition, the application retrieves all the rows that have been written to the table over the course of the composition, and each row is internally converted to an action in the application, and executed, transparent to the user, who sees a list of tones that represent the latest state of the composition. The process of connecting usually requires connection to a database that is known in advance; this can be either a local, or an online database. Only those users who have made changes to their composition are recorded in the user database history; for example, if a user makes even one change, their presence is recorded in the database until the end of the composition.

3.2 Audio Generation & Compilation Module

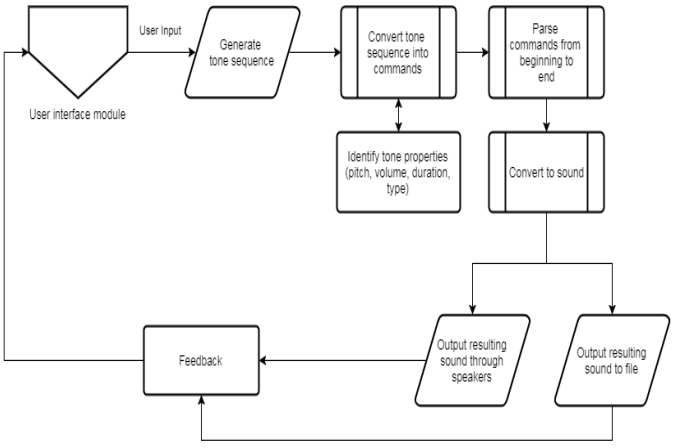

Fig - 2: Flowchart of the audio generation & compilation module

The audio generation and composition module is the module which is responsible for the creation, modification, deletion, playback and compilation of the tones in the composition. This module consists of an audio generator, which accepts tone descriptors as input (which consist of the tone type, volume, and frequency, which is enough to describe any given

tone fully), and produces audio as output. Initially, a series of tone descriptors are created. This is typically done by the user, although it can also be created from another trigger (for example, when a new user joins the composition, and the tone descriptors which have been created by the other users in the past are loaded into the new user's session); then, these tone descriptors are parsed, and an audio sequence is created which is then played back through the speakers, based on the inputs given by the tone descriptors; alternatively, if the composition is being compiled, then the output, instead of being played back through the speakers, is routed to a file, which is then saved on the user's hard drive.

3.3 Synchronization Module

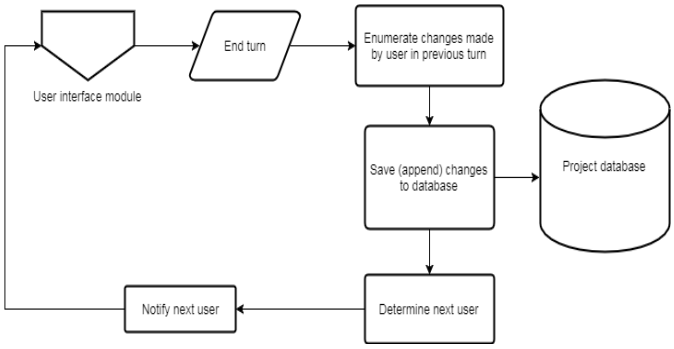

Fig - 3: Flowchart of the synchronization module

This module is responsible for the synchronization of all the actions performed by the users, and the coordination behind-the-scenes in order to reflect the changes needed to update each user to the latest revision of the composition. That is, this module records all changes made by the user to a central database (which is either locally hosted or hosted on the Internet), and it is responsible for making sure that only one user can edit the composition at a time – when it is their turn; it is also responsible for passing control to the next user in the queue.

3.4 User Interface

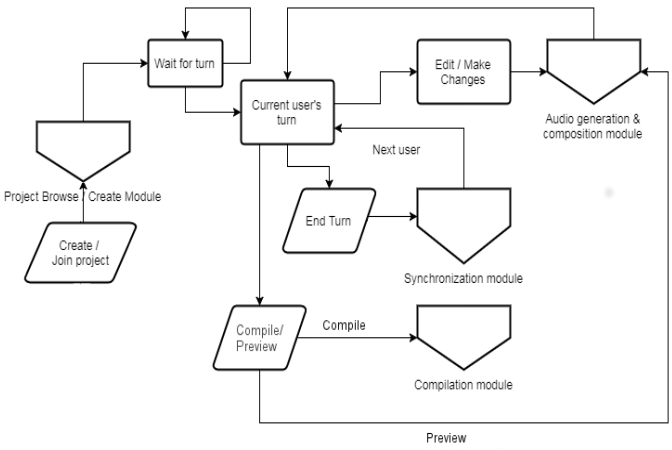

Fig - 4: Flowchart of the user interface

This module is the overarching module which connects all the other modules to the user; using this module, the user can communicate with the other modules, as shown above; this module retains connections to all other modules. It consists of three parts (not shown), called "pages", which represents the front-end of each module it communicates with, which are the Project page, the Audio Composition page and the Connection page.

The project page is the default page that is visible to the user, upon opening of the application. This allows them to either join an existing composition or create a new composition with only them as the initial member; joining an existing composition requires them to connect to its database, which can be done via the project creation and browsing module.

The audio composition page is the main page where all the tasks related to composition management, and composition, take place. It can be considered as the front-end view for the audio generation and composition module; this page allows the user to create, edit, and delete tones as per their choices, and to preview the resulting composition; it also allows them to pass their turn to the next user, and, if the user is an administrator (as is common for compositions with less than 5 people in it) they have the option to view, and kick, other users from the composition. They are also allowed to modify composition properties from this page, such as tempo, master volume, etc.

The connection page is that page which allows the user to connect to another users' composition (which is stored as a database), if they are given sufficient permissions to do so. This page can be considered as the front end for the project creation and browsing module; it is embedded within the project page, and so the two can be considered as one logical page (even though they are two physically distinct pages) since the connection is a prerequisite to joining a given composition.

4. REQUIREMENTS

4.1 Performance Requirements

While the application is not particularly demanding on the end-user/client-side, the server-side performance is a different story. Relational databases are less suited for scalability [4] – after a certain point, access to the database (reads, writes, notifications, etc.) becomes unreliable and unresponsive, which can lead to user dissatisfaction. Relational databases are usually vertically scalable, rather than horizontally scalable; this means that in order to increase capacity, more powerful computers have to be used to host the database. Thus, sufficient resource provisioning has to be carried out in anticipation of the expected demand on the database; this responsibility typically lies on the database host.

4.2 Security Requirements

Security is a big issue when working with network-enabled applications. This application connects to a database which may or may not be located in the same region as the host; as with any online database, the database should be password protected in order to prevent access by unknown intruders. It is well documented that leaving a database unencrypted and accessible by anyone is a huge security risk, allowing unauthorized users to gain access to the database and assume other users' identity, as well as steal database information. [5]

The application does not intend to store too much personally identifying information; the extent of sensitive information which is stored in the database by the application is that of the users' IP addresses (as an alternative to user names and passwords, to identify users). However, an unauthorized user who has access to the database would likely have access to it outside of the application too, and may be able to modify the database so as to affect other users' compositions, possibly leading to loss of work or modification which nobody knows about.

4. SCREENSHOTS

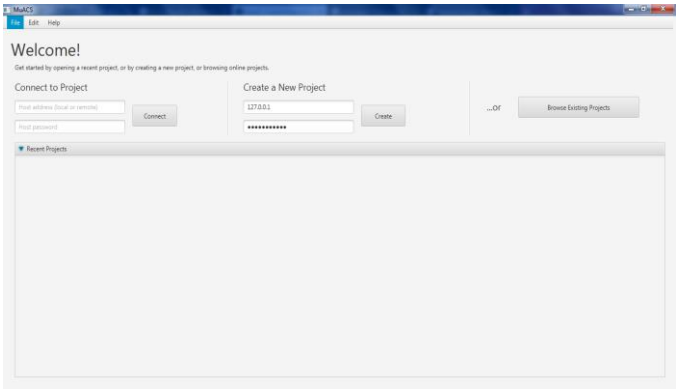

Fig - 5: Screenshot of project being created on localhost

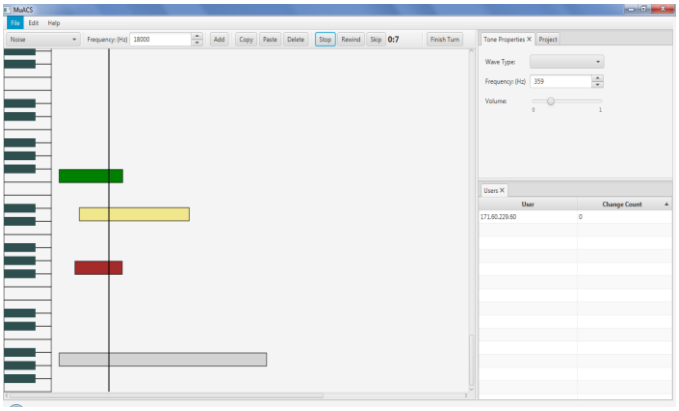

Fig - 6: Screenshot of editor interface. A number of tones have been created and the resulting track is played back.

5. FUTURE WORKS

It has been established earlier that traditional implementations of relational databases have challenges regarding scalability to accommodate a large number of users. This can be alleviated to some extent by using NoSQL datastores, which scale horizontally rather than vertically, and hence are easier to scale. By switching to a NoSQL datastore, it is expected that the number of users connected to a given composition can increase manifold.

6. CONCLUSIONS

The application thus described, which allows multiple users to collaborate in a turn based manner, in order to produce an audio composition has been designed using the appropriate tools. As with systems nowadays, it is known that collaboration can become difficult and/or tedious with existing tools; it is hoped that such a project will make such present and future collaborations easier.

REFERENCES

[1] O. Bown, "Experiments in Modular Design for the Creative Composition of Live Algorithms". Computer Music Journal, vol. 35, Sep. 2011, pp. 73-85, doi:10.1162/COMJ_a_00070

[2] H. Chamberlin, Musical Applications of Microprocessors. New Jersey: Hayden Book Co., Inc., 1980.

[3] G. S. Kendall, C. Haworth, & R.F Cádiz. "Sound Synthesis with Auditory Distortion Products". Computer Music Journal, vol. 38, Dec. 2014, pp. 5-23, doi:10.1162/COMJ_a_00265

[4] Pokorny, J. "NoSQL databases: a step to database scalability in web environment." International Journal of Web Information Systems, 9(1), 69-82., pp. 69-82, 2013.

[5] Shulman, A. "How to Mitigate the Most Significant Database Vulnerabilities". Top ten database security threats, 2006.