Musician Assistance and Score Distribution (MASD)

Nathan Magnus
University of Regina
3737 Wascana Parkway
Regina, Saskatchewan, Canada
magnus2n@uregina.ca

David Gerhard
University of Regina
3737 Wascana Parkway
Regina, Saskatchewan, Canada
gerhard@cs.uregina.ca

ABSTRACT

The purpose of the Musician Assistance and Score Distribution (MASD) system is to assist novice musicians with playing in an orchestra, concert band, choir or other musical ensemble. MASD helps novice musicians in three ways. It removes the confusion that results from page turns, aides a musician's return to the proper location in the music score after the looking at the conductor and notifies musicians of conductor instructions. MASD is currently verified by evaluating the time between sending beats or conductor information and this information being rendered for the musician. Future work includes user testing of this system.

There are three major components to the MASD system. These components are Score Distribution, Score Rendering and Information Distribution. Score Distribution passes score information to clients and is facilitated by the Internet Communication Engine (ICE). Score Rendering uses the GUIDO Library to display the musical score. Information Distribution uses ICE and the IceStorm service to pass beat and instruction information to musicians.

Keywords

score distribution, score-following, score rendering, musician assistance

1. INTRODUCTION

Orchestras, concert bands and other large groups of musicians often rely upon a conductor to synchronize the group. The conductor's formal duties, however, extend beyond simply synchronizing the musicians. The conductor is also responsible for listening critically to the music and shaping it based on a unifying musical interpretation. The conductor's modifications are communicated to musicians through a complex set of visual instructions such as hand gestures, body movement, breathing and even eye contact [6].

Anecdotal evidence suggests novice musicians face challenges related to following conductors and reading music. The diversity of a conductor's gestures, variations between conductor styles and methods of conveying instructions may pose challenges for beginner musicians, making it common for these individuals to return to an incorrect measure after looking at the conductor. Furthermore, musicians that are still learning the score or are not yet skilled sight readers may completely fail to look at the conductor and thus miss some of the conductor's instructions. Learning the right time to turn pages is also a challenge for novice musicians, and a missed page turn can lead to missed notes on the next page. Page flips can be especially challenging for musicians that play instruments that require two hands. Thus, assisting with these problems may help beginner musicians focus more upon the music they are playing and improve the quality of their performance.

The MASD system was designed to assists musicians with some of the difficulties that they may encounter as a member of an orchestra, concert band, choir or other musical group. MASD is also intended to reach a broad audience of all levels of dedication. Thus, the project utilizes common network infrastructure and computers to synchronize multiple displays as well as transmit information from conductor to musician. Since live musical performances have very strict latency requirements due to the importance of synchronization, latency was the primary evaluation metric used for this system.

2. BACKGROUND

Researchers have investigated score following methods that can follow music played with errors as well as allow untrained partners to contribute to duets [5]. Some research has also been put into creating a virtual conductor to lead a group of musicians [7]. Other relevant research areas explore generating digital information from conductor signals and distributed rendering engines.

The problem of digitizing conductor tempo and other visual cues has been address by Peng [6]. Peng's system utilizes specialized equipment that extracts beat information and conductor gestures. This information is then used as input for a custom application. Similarly, iSymphony [4] extracts baton information and uses it as input to an application. MASD was developed to facilitate score and information distribution, and these papers demonstrate the potential for input control.

Some important work relevant to score engraving, conducted by Bellini, Nesi and Spinu [2], created a collaborative score editing and engraving system called Music Object Oriented Distributed System (MOODS). MOODS allows individual musicians to create notes about the music that are visible by other musicians and the conductor. MOODS also addresses score display and automatic page turning. The paper indicates that the system works and is practical.

A system know as muse [3] addresses music distribution and some information distribution. The system also provides automatic, real time page turning but determines when to flip pages by analyzing the music that is played. MASD approaches this problem differently by using the conductor's gestures to determine when to flip a page, removing the need for an acoustic analysis component in the system.

3. INTERNET COMMUNICATION ENGINE (ICE)

The Internet Communication Engine (ICE) [8] is used in several places within the MASD system. One of the advantages of this framework is the ability to write platform and language independent code. This is facilitated by an interface definition language, called Slice, and tools that convert Slice into languages such as C++, Java, C#, Python, Objective-C, Ruby, PHP and ActionScript. However, the main advantage of ICE is the ease and efficiency with which it handles communication between two or more machines. Using the language mapping tools provided with ICE, programmers can easily create objects that are stored on the server or on a client and transmitted to another machine with only a few lines of code.

ZeroC also provides a suite of programs which includes the IceBox server. IceBox allows developers to run other ICE services with minimal configuration and effort. For example, MASD uses IceBox to run the IceStorm service. IceStorm is a service that can be used by the ICE framework and provides a method of broadcasting information from one computer (publisher) to multiple other computers (subscribers). The publisher creates the information it wants to send and passes this information to the IceStorm service which distributes it to all subscribers. Subscribers are responsible for informing the IceStorm service about what information (topic) they desire to receive.

4. INTERNAL REPRESENTATION

Transfering complete scores between hetrogenous machines is a challenge. For this reason, an internal representation was developed which transfers only required information to each client machine. The rendering is then completed separately using this representation. The internal representation is declared in Slice files and created as part of the automatically generated code made by the `slice2java` tool which is provided by ZeroC [8]. This allows information to be transmitted via ICE between client and server.

4.1 Duration

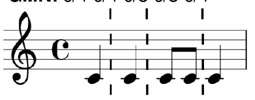
The Duration class is used to represent any note duration as a numerator and denominator pair. Originally it was created with a static common denominator of 128 and a note's duration was converted to its appropriate value as a fraction with a denominator of 128. However, it became apparent that certain notes, such as triplets, would cause errors due to rounding of the numerator. This resulted in the implementation of a static variable denominator and an instance denominator. This static denominator is calculated from a list of denominators within the score and will increase whenever a denominator is encountered that results in a rounding error. As a result, all note durations can be accurately represented and handled.

4.2 Beat Highlighting

Many beat highlighting systems, such as Garage Band [1], present a "playhead" metaphor showing the position of time relating to notes. This method, however, may confuse musicians because they only see the playhead briefly while glancing between music and conductor. Therefore, we decided to highlight the current beat without moving a playhead. This leads to a requirement to select the position of the underlying beat relating to the notes on the score.

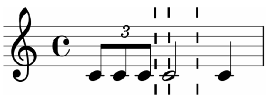
Beat highlighting information is contained within a single record. This record maintains a time signature, as well as a list of note offsets, a list of note durations and a list of denominators. Each of these lists is the same size as the number of beats in a measure. Thus, each entry corresponds to a single beat within a measure. The list values are used for highlighting the notes within a beat or measure. The denominator values represent beat units and are used as the denominator for both note offset as well as note duration within a measure. Offset is used to indicate the location within the beat that highlighting should begin, and duration indicates how much of the score should be highlighted. Figure 1 shows an example of how these values are used. The denominator changes when a note is encountered that cannot be accurately represented, and all subsequent values are represented using the new denominator. Offset is used to avoid overlapping a previously highlighted note by indicating where in the beat that highlighting must start.

GMN: c/4 c/4 c/8 c/8 c/4



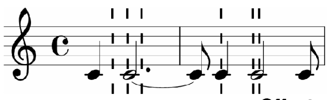
	Beat			
	1	2	3	4
Offset:	0	0	0	0
Duration:	1	1	2	2
Denominator:	4	4	8	8

GMN: c*1/12 c*1/12 c*1/12 c/2 c/4



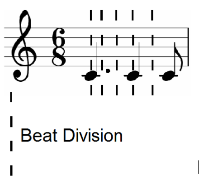
	Beat			
	1	2	3	4
Offset:	0	0	0	0
Duration:	3	6	0	3
Denominator:	12	12	12	12

GMN: c/4 c*7/8 c/4 c/2 c/8



	Beat							
	1	2	3	4	1	2	3	4
Offset:	0	0	0	0	0	1	3	1
Duration:	1	6	0	0	3	4	0	1
Denominator:	4	8	8	8	8	8	8	8

GMN: c/4. c/4 c/8



	Beat					
	1	2	3	4	5	6
Offset:	0	2	0	0	0	0
Duration:	3	0	0	2	0	1
Denominator:	8	8	8	8	8	8

Figure 1: Example Guido Music Notation (GMN) and associated transcription, offset, duration and denominator values. (see text for more details)

5. COMPONENTS

The project is composed of three major components. Each component is designed to be as modular as possible. Such a design allows individual components to be replaced with minimal reconfiguration. Thus, when new technologies, libraries or APIs are released that improve upon the functionality of the system, components can be partially or completely modified to take advantage of the improvements.

5.1 Score Distribution

The Score Distribution component parses a musical score and facilitates the transfer of musical scores to clients. Since the client's computer and score server are distinct machines, communication must occur between two potentially heterogeneous computers. This process is facilitated using the Internet Communication Engine (ICE) framework (mentioned above) and has two sub components.

Conductor scores are challenging to implement effectively. This is due to the non-linear fashion in which conductors look through music and the sheer quantity of information that must be presented. Although MASD does not implement a conductor interface, one could be implemented by making use of the internal representation that is already in place.

5.1.1 Score Distribution Client

The Score Distribution client's sole purpose is to connect to the Score Distribution server and obtain a Part object from the server. This Part object represents a musical score, such as "first trombone" or "soprano", and only contains information relevant to the specific musician requesting the part. For example, a clarinet Part object only contains music and relevant cue notes for the clarinet, but will not contain any information about another instrument's part unless required for queueing or reference.

5.1.2 Score Distribution Server

This component is responsible for parsing the Music XML and creating the internal representation. The parsing process separates the musical parts and creates unique sets of information (described in Section 4) that client machines can request. After completing the creation of the internal representation, the Score Distribution Server handles requests for part or score information. It will continue doing this until it is shut down.

5.2 Score Rendering

Score Rendering is the process of displaying the score to the musician. This involves drawing the score onto the screen using a library or external tool to create the images. The method of displaying this information to the user may vary. This could allow instrument specific styles, such as guitar tablature or percussion score, to be displayed on an individual basis. Thus, performers can view their music in whatever format they are most comfortable reading. Additionally, score rendering can be done with different rendering engines on different computers. In our case, we focus on one specific score rendering engine.

5.2.1 Page Display

This implementation of Score Rendering displays the score one page at a time. The number of lines displayed on each page may vary depending on display requirements for the music. Using the information received from the conductor, the Page Display updates the score by changing the color of the notes to red as the beat is played. This is referred to as highlighting. Once the note has been highlighted, it will remain red. There are two modes of score highlighting: Measure Highlighting; and Beat Highlighting. Measure Highlighting uses the Guido Engine Library (GUIDOLib) to highlight each new measure as it is reached. When the first beat of the measure is to be played, every note in the measure will change color. Beat highlighting, however, is much more complex and more susceptible to a highlighting bug in the Guido Engine Library that highlights the stem of a note but does not highlight the head until the next beat. The beat highlighting method highlights any note that is entirely or partially within the duration of a single beat.

Page Display also addresses another important issue. This issue is automatic page turning. As the note highlighting progresses, three measures ahead of the current location is checked to determine what page of the GUIDOLib rendering it is located upon. If it is located on the next page, a small portion of the top of the score is removed and replaced with the first bit of the next page. The replaced area is not highlighted since it has not yet been played. This allows the musician to look ahead in the music when at the bottom of a page. Once the musician has reached the next page of the score, the current page is completely replaced with the next page. This method removes the need to manually flip pages and will assist novice musicians by allowing them to focus on more important aspects of the conductor's gestures and the performance.

5.3 Information Distribution

The Information Distribution component is responsible for transmitting beat information and conductor instructions to each musician's client computer. The Information Distribution portion of the project has pieces present on both a centralized server and musician computers.

5.4 Combining Components

In order to complete the system, the Score Distribution, Information Distribution and Score Rendering components are combined to create two applications. These applications are the Client Application, which is used by musicians, and the Sever Application, which is configured by the conductor. Figure 2 shows the high level structure of the MASD applications and how they interact with each other.

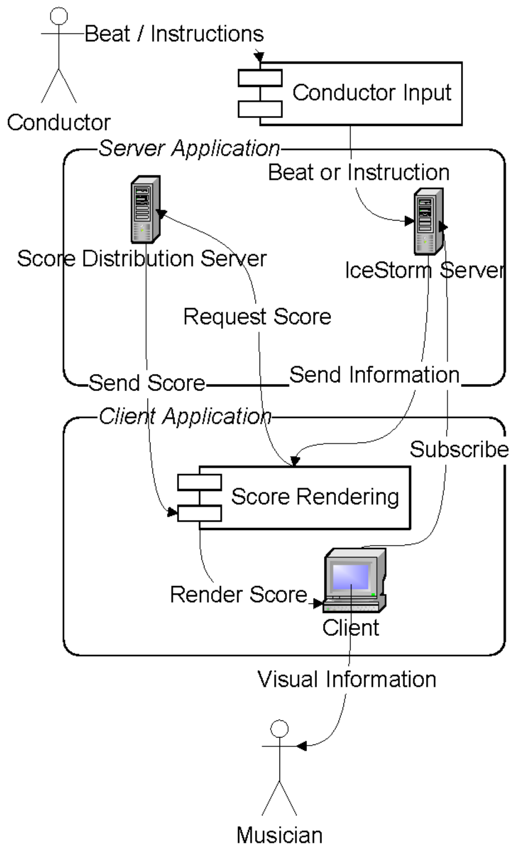


Figure 2: MASD high level structure and interaction.

5.4.1 Client Application

The Client Application's job is relatively simple. The application first determines which servers contain information about the parts and will be conducting the IceStorm distribution of beats. The application then presents the musician with a list of parts to choose from. Once the musician has selected what part he or she wishes to receive, the application requests only the relevant information from the Score

Distribution Server and renders the appropriate score. The application then waits for conductor instructions or beat information from the Information Distribution system. When information is received, the application updates what it is rendering using the buffered next image. The computer running the Client Application does not need to be particularly powerful because this program is waiting for updates throughout the majority of its execution and its responsibilities are relatively undemanding.

5.4.2 *Server Application*

The Server Application is responsible for creating and updating the Score Distribution and Information Distribution components of the project. This application allows the conductor to select a score to load and then broadcast incoming beat and instruction information until the application is closed. Since the digitization of beat information and conductor instructions is not the focus of MASD, an independent system is responsible for detecting this information and passing it to MASD. The Server Application is used to manage two heavily used functions and will likely require a relatively powerful machine to run well. These functions, however, could be run on two individual machines should performance become an issue.

6. PERFORMANCE

Latency tests were conducted to determine the efficiency of the system and speed at which it performs. Tests were performed using 1, 5, 10, 15, 20 and 25 client machines and a single server. These tests calculated the time between the server sending information and the client receiving this information (distribution time) and utilized the system clock of an independent machine to help reduce synchronization issues. Testing showed that distribution time increases dramatically with the number of clients. On average with 25 clients, beat distribution takes 38ms and instruction distribution takes 3ms. Rendering limitations make the theoretical limit of MASD approximately six beats per second, however this corresponding to a theoretical maximum tempo of 360 bpm which is sufficient for most musical applications.

7. CONCLUSIONS

MASD successfully implements page rendering, page turning and score distribution and can be used to aid novice musicians when playing from a musical score. The limited human resources of the project and difficulties accurately measuring transmission time make it difficult to definitively determine the usability and effectiveness of this system. However, score rendering can achieve approximately 360 bpm and subjective analysis indicates that the response time of MASD is acceptable. We believe exploring push distribution using common infrastructure as a method of synchronizing an orchestra or other musical group most certainly warrants further research.

7.1 Improvements & Future Research

A potential improvement to MASD is the use of multiple IceStorm servers. This change would allow multiple IceStorm services to be connected in a hierarchy or tree. This distributes the load of pushing updates to multiple servers. However, this distribution method would introduce additional system latency due to the increased number of servers that must process the information before reaching the musician. The ability of MASD to be used as a distribution method for non-traditional orchestras, such as laptop orchestras or musical groups with members in different physical locations, is an area for possible future research.

In order to improve displays, usability tests and different methods for displaying the music could be considered. For example, one possible method could be displaying an arbitrary number of lines of music on the screen at a time and replacing a line with the next line to be displayed when a line is completed. The musician would play a line and then look to the next line, with the order wrapping around from the last line on the page to the first line on the next page. Usability tests of display methods could assist in improving the interface and identifying improvements for the system.

In order to ensure that the beat and instruction distribution is prompt, several methods that do not rely on network protocols and mediums could be implemented. Specialized hardware, for example, could be created to directly connect the conductor's server to musicians. This would allow for near instantaneous transmission of simple information such as indicating the next beat has occurred. Infrared or other broadcast mediums could also be used to simultaneously send information to all clients. Another radically different direction that could be explored is predicting the occurrence of the next beat using previous beat arrival information and knowledge about the score's tempo. The musician computer would then use this information to update its graphical display and update the next beat prediction when it receives the notification from the server.

A hybrid method could also be employed, for example, using specialized hardware or broadcast mediums to communicate small amounts of information and a network to communicate larger pieces of data. This would allow for the immediacy required for beat synchronization as well as the transmission of more complex information such as conductor instructions without too much specialized or expensive hardware.

8. REFERENCES

[1] Apple Inc. Garage band '11, January 2012. `http://www.apple.com/ilife/garageband`.

[2] P. Bellini, P. Nesi, and M. B. Spinu. Cooperative visual manipulation of music notation. *ACM Trans. Comput.-Hum. Interact.*, 9:194–237, September 2002.

[3] C. Graefe, D. Wahila, J. Maguire, and O. Dasna. muse: a digital music stand for symphony musicians. *interactions*, 3:26–35, May 1996.

[4] E. Lee, H. Kiel, S. Dedenbach, I. Grüll, T. Karrer, M. Wolf, and J. Borchers. isymphony: an adaptive interactive orchestral conducting system for digital audio and video streams. In *CHI '06 extended abstracts on Human factors in computing systems*, CHI EA '06, pages 259–262, New York, NY, USA, 2006. ACM.

[5] C. Oshima, K. Nishimoto, and N. Hagita. A piano duo support system for parents to lead children to practice musical performances. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3, May 2007.

[6] L. Peng. A gestural interface in a computer-based conducting system. Master's thesis, University of Regina, 3737 Wascana Parkway, Regina, SK, S4S 0A2, 2008.

[7] D. Reidsma, A. Nijholt, and P. Bos. Temporal interaction between an artificial orchestra conductor and human musicians. *Comput. Entertain.*, 6:53:1–53:22, Dec. 2008.

[8] ZeroC, Inc. Ice, January 2012. `http://www.zeroc.com/`.