Virtual Collaborative Learning Environments for Music: Networked DrumSteps

Conor McCarthy*, James Bligh, Kevin Jennings, Brendan Tangney

Centre for Research in IT in Education

Trinity College Dublin

Abstract

This paper focuses on technologies that enable meaningful, constructionist interaction in collaborative music environments. In particular, it describes the design and implementation of Networked DrumSteps, an application that allows multiple users in different locations to collaborate in the process of music composition, but without the use of standard notation.

Findings from the testing of Networked DrumSteps in an educational environment show encouraging results. Learners agreed that the piece they made together was "different" from something they would have made on their own, and that they made it in different way. From a technical perspective Networked DrumSteps has proven itself to be highly scalable and extendable. Tests done with up to eight people collaborating together showed no signs of performance deterioration.

Introduction

Notation lies at the very heart of teaching, learning, composing and playing music. Yet notation is a hard and non-intuitive concept for any child to learn. For many it is the point at which interest in learning music ceases due to the difficulty in grasping this concept (Bamberger 1991). By merging technology with music, we have the ability and opportunity to develop new tools that allow a new avenue of exploration that eschews the rote methods of learning notation in favour of new ways of exploring the music. Multiple representations can lead to a better understanding.

The merits of collaboration and groupwork are manifold and much has been written about their use in various settings. Groupware is the name given to technologies that support and assist collaborative activity. Collaborative technologies have a lot to offer music education. CSCL (Computer Supported Co-Operative Learning) is concerned with the use of groupware technology in learning environments.

Based largely on Papert's work, DrumSteps (Jennings 2001) is a microworld designed to allow learners to experiment with various aspects of percussion and rhythm in a musical design space. This experimentation is performed in the absence of standard notation. The basic idea is to allow the learner to build a set of steps and create drum sounds by dropping a ball down the steps. The system has the full range of concepts specific to the domain of rhythm, such as pulse, tempo, timbre, texture, syncopation, accent, etc. Learners have complete control over the learning environment, as constructionism dictates.

In light of what has been said earlier about groupware and collaborative learning, this paper focuses on collaborative music microworlds. In particular it describes a version of DrumSteps that has been extended to become a networked groupware application designed to support collaborative learning, in effect becoming a distributed musical composition tool. The theory behind the design of a networked version of DrumSteps, hereafter called Networked DrumSteps, stems from findings in educational philosophy and learning theory that make a strong link between learning and social interaction and the idea that knowledge is constructed through interaction with others.

The rest of this paper is structured as follows. Section 2 is a breakdown and review of the theories supporting Networked DrumSteps, such as constructivism and constructionism. Section 3 describes the design process involved in turning DrumSteps into a networked application. It extracts user requirements and focuses on the rationale behind the various design decisions, and under what conditions these decisions were made. Section 4 describes the implementation of Networked DrumSteps. Section 5 presents the findings resulting from tests of the software with a group of learners. An evaluation of the software is also included. Section 6 concludes the thesis with a look at future work.

Background

Information Technology and Music Education

Most music education software packages designed today fall into one or more of the following categories: tutorial, drill and practise, game, simulation or problem solving, with drill and practice being arguably the most popular/common category (Pellone 1992). Most of these packages focus on information transfer or on testing knowledge acquired elsewhere. They tend to emphasise the lower cognitive processes such as knowledge (remembering the right information to complete an answer) and comprehension (Bloom 1956). This is highlighted by patterns of user interactions that usually focus on colourful interface widgets, as opposed to the underlying material or content (Kinshuk 2002). What is needed are applications that seek to delve a little deeper and enable a learner to grasp the concepts the software is trying to teach. Software must move from passive instruction to enabling engaging, constructive knowledge construction. Software that only demands the learner to remember facts and repeat them parrot fashion serve little purpose in education today. Effective music instruction depends on an active approach (Serafine 1988).

Notation and Composition

Standard music notation serves three purposes. It is a method of "storing" music, making it available to the composer and to others at a later date. It is used to communicate" music between people, to let them speak a common language of music (Chew 1980). Lastly, but most importantly, notation can be used as a tool or device to think about or reflect upon music. Notation lies at the very heart of teaching, learning, composing and playing music. Yet notation is a hard and non-intuitive concept for any child to learn. The few notation programs that have emerged over the last decade show limited success and few have managed to evolve to suit changing educational needs. Programs such as Finale™[1], Cubase™[2], Cakewalk™[3] and Sibelius™[4] have a steep learning curve and lack intuitiveness. Music notation software also typically has many boxes, symbols, tools and menus that must be learned and used. Their advantages lie in the ease with which they present the material to the users to be corrected and heard again immediately. Much of the interaction remains in the surface of the interface widgets, not with the underlying content. There exists plenty of software for learning pitch, notation etc, but little for higher order cognitive skills (Webster 2002).

What is needed is software that actually lets learners compose their own work. They must have complete control over every aspect of the composition from the outset, to feel that they are in control of their own learning. Software that 'fills in the gaps' only serves to confuse and distance the learner from the learning experience. An example of software that enables learners to compose their own work is Impromptu (Bamberger 2000).

Research indicates that there is also a need for software that simply represents notation in a way that is easier to grasp than its traditional form. Bamberger (Bamberger 2000) points out that a basis in music education that doesn't focus on learning traditional notation is beneficial to the learner. When the learner does need to learn notation at some later stage, they can draw on this experience with music and use it as a basis for answering questions.

Collaboration

"Working in groups brings some characteristics such as synergy, the ability to consider more information…cognitive stimulation and member learning from other members, which can be very useful to improve the learning process. Exposure to alternative points of view can enhance learning" (Fuks, Gerosa et al. 2002). Kedong and Jianhua (Kedong and Jianhua 2001) give 5 components to make learning truly collaborative:

- Clear, positive interdependence among students
- Regular group self-evaluation
- Interpersonal behaviours that promote each members learning and success
- Individual accountability and responsibility
- Frequent use of appropriate interpersonal and small group social skills

By positive interdependence, group member promote each others success by:

- Giving and receiving help and assistance
- Exchanging resources and info. Explaining, teaching ones own knowledge to others
- Giving and receiving feedback on behaviours and teamwork
- Challenging each other's reasoning. This promotes curiosity, motivation to learn and better decision making (Johnson and Johnson 1997)

Underpinning all these human aspects of collaborative learning must be the appropriate form of technology. After all, the technology is the enabling factor of collaboration in virtual learning environments. If learning describes the pedagogical theory behind it, then virtual describes the technology.

Educational Groupware Implementations

CSILE (Scardamalia and Bereiter 1993), The KidLink Network[5], Belvedere[6], CoVis (Pea 1993), KIE (Knowledge Integrate Environment)[7], CaMILE (Collaborative and Multimedia Interactive Learning Environment)[8] are all collaborative application for use in educational environments. Many of those shown are science based and enable students who are separated in either time or space to collaborate on a project. However, an extensive search through the literature failed to discover any systems available today that let children collaborate on a piece of music in real time. There is no existing technology that gives access to the actual music, to be able to manipulate it and experiment together, in real time, across distance. Essentially learners cannot communicate through music. Music requires active participation. What is needed is real time music collaboration. Yet real time or near real time music collaboration is still an idea facing many difficulties, the most serious of which is timing. Rocket[9] is an attempt to overcome this problem, but it too is flawed in that it is not designed for use in learning environments.

Java Groupware Implementations

Java[10] code can be run on any machine that has a Java Virtual Machine on it, overcoming any platform dependency problems. This enables the same Java application to be ported to Windows, Mac and Unix, with no code rewrites whatsoever needed. This portability, coupled with its powerful networking abilities has placed Java at the forefront of collaborative systems design. Applications such as NetChat (Reinhard, Kiesow et al. 2000), Habanero (Chabert, Grossman et al. 1998), JCE (Abdel-Wahab, Kvande et al. 1996), JAMM (Begole, Struble et al. 1997), JASMINE (Abdulmotaleb, Shirmohammadi et al. 2000), Java Remote Control Tool (Kuhmunch 1998) are all built using Java. Of all the applications described above, JASMINE stands out as offering the most in terms of functionality, ease of use and effectiveness. Yet the problem with JASMINE, and some of the others, is that they are all-or-nothing implementations. Either one uses the framework they offer and every single interaction and movement made in the application is noted and sent to all others, or one doesn't "collaboratize" one's application at all. The above applications, or a combination of them, will suit the needs of most

[1] http://www.codamusic.com

[2] http://www.steinberg.net

[3] http://www.cakewalk.com

[4] http://www.sibelius.com

[5] http://ctl.augie.edu/educ/middle/Education/Mavericks/kidlink

[6] http://www.pitt.edu/~suthers/belvedere

[7] http://kie.berkeley.edu/KIE/info/publications/AERA96/KIE_Instruction.html

[8] http://www.cc.gatech.edu/gvu/edtach/CaMILE.html

[9] http://www.rocketnetwork.com/

[10] http://java.sun.com

collaborative systems. But no evidence exists that any of them have been successfully utilised for interactive music environments.

Pedagogy

Music, being an open ended and abstract domain is well suited to the idea of microworlds (Papert 1980). Papert outlined some features a microworld should have. A microworld should:

- Draw on some well established personal knowledge
- Be engaging
- Involve building and experimentation
- Encourage procedural thinking
- Be useful to novice and expert alike

Ferguson (Ferguson 2001) provides a number of important design principles for learning environments such as microworlds:

- Create real-world environments in which learning is relevant
- Focus on solving real-world problems
- Use instructors as a guides
- Provide learner control
- Negotiate instructional goals with students
- Use evaluation as a self-analysis tool
- Provide tools which help learners interpret multiple perspectives
- Insure that learning is internally controlled and mediated by the learner
- Provide multiple representations of reality
- Focus on knowledge construction, not reproduction

These guidelines provide a blueprint for designing microworlds.

Microworlds for Music Education

Some examples of microworlds for music education include MICK (Thibault), Zicarelli's OvalTune™, music LOGO, Subotnick's Making Music™ and HyperScore (Farbood 2001). Most are graphical based, allowing users to visually manipulate musical motives. But the chief weakness with programs such as HyperScore and OvalTune, is that they don't allow learners access to the basic music concepts and ideas, such as pitch. The software is built so that the music is “snapped” to certain pre-defined patterns, which although close to the original, natural sound, are still not fully representative of what the learner is trying to do. As mentioned above, “filling in the gaps” only serves to confuse and distance the learner. It is this lack of control that inhibits the effective use of these programs. DrumSteps (Jennings 2001) is an attempt to apply the theory of microworlds to music. It is a microworld designed to let users experiment with percussion. DrumSteps allows for musical experimentation without the need for standard notation. DrumSteps is engaging in the sense that it is more interesting than learning about rhythm from a textbook or being told about it. It also relies heavily on building and experimenting, using iterative processes to encourage procedural thinking. It conforms to many of the rules set out for microworld environments, such as construction and experimentation. Using DrumSteps, learners can interact directly with the musical content in an intuitive way, without using standard music notation.

Design

This section outlines the design requirements for a collaborative music learning environment. These requirements are derived from the principles described in the background section.

Music Microworld Requirements

As well as those principles outlined above, microworlds for music learning must embody some additional principles:

- A microworld should ideally focus on some clearly defined aspect of music, in this case notation.
- The use of alternative notation can lead to more natural, in-depth learning. A microworld that presents new ways of presenting musical content would let the learner engage with the material in a more intuitive fashion.
- The microworld must be easy to use, and not too complicated. An important factor is that the learners have complete control of their own work. This is one of the core ideas of constructivism. In effect, this mean that nothing the user sees is there by accident, they are responsible for everything they see on the screen.
- A music microworld must allow for an iterative cycle of learning: trial and error, listening, making judgments and editing.
- A microworld must have no clearly defined problem, and no right answer. Its effectiveness lies in it ability to let the learner experiment by allowing for multiple solutions, through the method of iteration described above.

DrumSteps

Some characteristics of DrumSteps are:

- Music is embodied in the structure of a piece in DrumSteps.
- Suitable for all levels of proficiency.
- Encourages procedural thinking through its iterative building process. Learners can build, listen then refine and repeat the process.
- Consistency through actions and operations that produce clearly definable results.
- Intuitive understanding as it is based in real world knowledge.
- Standard music notation is not required.

In DrumSteps notation is intuitive. For example in DrumSteps 3 steps in a row represents 3 rests in music (Figure 1). This idea of the ball rolling along three steps in a row and not making a sound because it is not hitting anything makes more sense than standard music notation, where three rests are denoted by 3 abstract symbols (Figure 2). This familiarity with everyday occurrences helps the learner to ground himself in the software.

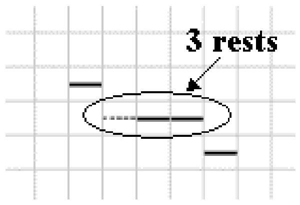

Figure 1: 3 rests in DrumSteps

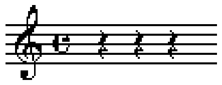

Figure 2: 3 rests in standard music notation

Collaborative Music Microworld Requirements

When music microworlds are transposed into a collaborative setting, a new set of design principles arise. A collaborative music microworld must allow for the creation and sharing of musical pieces between people located in potentially different places at different times. It achieves this through proper cooperation (to achieve a task), communication (by sharing ideas) and coordination (within a group). The background section outlined many of these ideas

Translating a standalone, single-user application into a multi-user, networked application introduces many issues. The requirements of a piece of collaborative software or multi-user application are more complicated than those of a standalone application. There are many details that must be considered for an application to be suitable in a networked environment.

Obviously the most important factor that must be accounted for in collaborative music microworlds is proper collaboration through the music, not about the music, as mentioned above. Students need to be able to do more than just share music files and chat about them. They need in-depth interaction with the content. As such, the microworld must adhere closely to musical concepts.

Networked DrumSteps

Networked DrumSteps is a shared virtual space in which two or more people can collaborate in real time on a piece of music. Learners can jointly create and manipulate a piece of music together, yet each learner is still responsible for their creation. Networked DrumSteps supports collaborative creative thinking. The use of a chat facility enhances the communication between participants, allowing them to talk through issues relating to the musical composition at hand.

There are two types of interaction in Networked DrumSteps: chat and graphical. The chat interface is useful for explicit communication between learners but the graphical communication is the most important type of interaction. Anything that a user does is automatically sent to all other users and represented on their screen.

Design Issues

When designing a collaborative microworld, the focus has to be on how learners interact. All design features and decisions had to apply directly to the learning process. Based on the principles set out above, this section looks at the issues involved. It is broken into two sections; the first looks at the user requirements, the second at the system or functional requirements to support this.

At the core of Networked DrumSteps, as with nearly all graphics-based collaborative applications, is the need for consistency. In an environment such as Networked DrumSteps, consistency ensures that what you see is what I see (WYSIWIS). Music in Networked DrumSteps is embodied in the graphical elements being transferred. Two different representations of what is supposed to be the same piece of music serves no purpose other than to confuse. It is tantamount to two performers being given different sheet music, and asked to perform the piece in concert.

User requirements

In creating a collaborative microworld from an existing application, most of the needs of the collaborative microworld fall under technical specifications of how to actually implement it. The original microworld, DrumSteps, embodies the necessary pedagogical facilities for music education. These pedagogical facilities and features must be kept and, if necessary, improved upon in a collaborative scenario and therefore warrant a review. The following comprises some requirements for Networked DrumSteps:

- Ability to connect/disconnect easily
- Ease of use/transparency
- Individual accountability and responsibility
- Support for longer pieces
- Private Chat

System Requirements

Opening up DrumSteps to collaboration meant that the original emphasis of DrumSteps as a musical education tool had to be maintained in a networked situation. Underpinning this idea are some functional issues that need addressing.

- System architecture
- Independent working
- Session Participation Awareness
- Local/Network Action Conflicts
- File sharing
- Workspace awareness
- Performance
- Updating Protocol
- Different Message Types
- Screen space and user representation
- Screen space and user tools
- Server logging
- Post session analysis
- Deleting GridElements/Balls
- Ownership
- Tagging GridElement's/Ball's
- Chat Notification
- Roles

IMPLEMENTATION

Networked DrumSteps builds upon the standalone version of DrumSteps. It is essentially a networking module that sits alongside the standalone version and handles all the network activities such as connection establishment, information transfer and connection release, effectively making network connectivity transparent to the user. Once connected, the software takes care of sending any data that affects the representation of the piece at other users machines.

Entity Structure

A Networked DrumSteps session involves 2 entities: a client and a server. These are typical distributed network components, but added to this we have the NetworkObject entity. The NetworkObject, described below, acts as the intermediary encapsulation mechanism between clients and server.

The NetworkObject Class

Chat messages are used solely for the chat session to transmit the users typed words to all other users. NetworkObject's carry all other interaction information in a Networked DrumSteps session. They contain all the essential updates of objects on the screen and as such are the underlying encapsulation mechanism to ensure consistency between clients. As well as object information they can also contain information about client status, voting decisions etc. The need for this type of object grew out of the emergence of several different types of interaction message that needed to be sent.

Every single interaction must be sent to and represented exactly at each user, to ensure consistency. So for example, if a user inserts a Ladder into their piece, this fact must be sent to all other connected users. The type of ladder and its position are the most important information to be sent. But then if that user decides to change the type of ladder, this updated information must then be sent. The same scenario occurs for Steps, Balls, and WormHoles. The deletion of an element must also be accounted for.

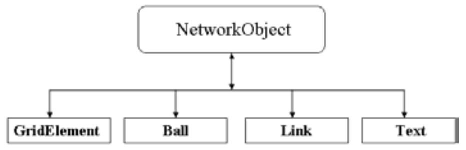

Figure 3: NetworkObject messages

Client

The client side of Networked DrumSteps acts as a module sitting alongside the standalone version. It contains all the necessary functions to access a Networked DrumSteps session, communicate with others logged into a Networked DrumSteps session, and to leave that session.

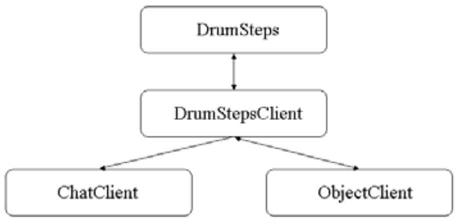

Figure 4: Client class structure

DrumStepsClient class

The DrumSteps class interacts directly with the DrumStepsClient class, which acts as the intermediary class between the actual network and the software.

ChatClient Class

The ChatClient class, as the name suggests, handles all chat interactions for a client or user in a session. Owing to its purely communicative nature, this class has direct access to the NetworkOptions class. The ChatClient runs as a separate Thread on the local machine, using the text socket passed to it from the DrumStepsClient class to perform its duties. Java's API provides appropriate methods to send and receive text over a socket. This class also provides methods to disconnect the socket upon closing or a network fault.

ObjectClient Class

The ObjectClient class is a little more complicated than the ChatClient class due to its importance for ensuring graphical consistency. Unlike the ChatClient class that just deals with text, the ObjectClient class must deal with numerous variations of content, all of which are embodied in the NetworkObject class described below. Like the ChatClient class, it contains methods for reading from and writing to the underlying object stream, as well as disconnecting the sockets, all provided by Java. This class is primarily concerned with receiving messages off the network from other users.

NetworkOptions Class

Although not shown in the diagram, the NetworkOptions class has 2 functions:

- Handle the graphical aspects of representing users to each other
- Act as an intermediary class for the chat function

This class is concerned with representing information about the network session and also for communication via the chat function. Due to the chat function being a simple duplex text service, it made sense to embed it in this class, where it would have direct access to the screen area for easy interaction. After all, a chat service either receives and displays text, or it takes text entered and sends it.

Another purpose of this class is to show the user who else is connected at any one time. As such the users can monitor who leaves and joins a session while they are connected. Users are represented by their names in Networked DrumSteps. Further to this is a feature that enables a user to see who has made what contributions to the current piece, as described in the design section.

Built in to Networked DrumSteps is a method of "tagging" each element that each user inserts into the piece. Each element on the screen belongs to one of the user's, and is tagged as such. Also recorded are any other interactions with elements such as moving or attribute changing. By double clicking on a users name, their contributions are highlighted in the piece.

Server

The Networked DrumSteps server is the heart of the DrumSteps network. Clients connected to it can share their work with all other connected clients. Although moving the majority of the computing work to the server creates the danger of having a single point of failure, it means that the system is highly scalable, to include many users, or extra services, such as a web-cam or other third-party software. The server provides support for latecomers, saving sessions to file etc.

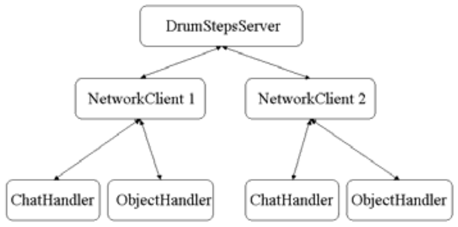

Figure 5: DrumSteps Server class structure

DrumStepsServer Class

The DrumStepsServer class is the root class for the server architecture. Like any server, it sits and awaits incoming connections from clients. Java provides a special form of sockets called ServerSockets, which basically sit and wait until they receive a connection request from a client.

NetworkClient Class

This class is used to represent a client on the server and to store all information and handle all interactions with that client in one place. The NetworkClient class stores the username, sockets, and references to the classes it uses to communicate directly with the client it represents. It has the ability to close the sockets and terminate the objects it creates.

Evaluation

This section looks at both pedagogical evaluations and technical testing.

Technical Testing

Performance testing was used to test the round trip times of messages over the network, as well as the sizes of the relevant messages. In order to perfectly synchronise the clocks during testing to achieve the best results, 2 copies of the client were run on the same host machine, with the server running on a different machine. In this way, both clients would be using the same internal clock, and therefore the results could accurately be measured against each other. For each message, the average of times taken to send a message from one client to another was recorded. For each of the 4 types of message, 1000 instances were sent, the times were recorded and then an average was calculated.

Table 1 contains the exact results from this testing. In this table, the average time taken for a message to travel from client to client over a wireless network is shown (first column), over a wired network (second column) and the average size of a message (column three).

	Wireless LAN (ms)	Wired LAN (ms)	Message Size (bytes)
GridElement	301	277	591
Ball	841	367	1,197
Link	380	461	663
Text	165	297	221

Table 1: Results of testing

The method used to determine the message size was a small Java program that measured the number of bytes that were being transferred when a message was sent. The average time taken to analyse the text of a message on the server was 16ms. Messages that simply get routed (GridElement/Ball/Link) took less than 1ms to process and send on the server. On average a chat message took 15 milliseconds to transfer on a wired network and 10 seconds on the wireless network.

Load testing tests how the system held up to an increased load by having 8 clients connect to the server and conduct a session. This aimed to test the robustness and tolerance of the system. In this testing scenario, both PC's and Mac's were used and testing was conducted over a mix of both wired and wireless access points.

The results of the testing with 8 clients show that the time taken to update each client did not deprecate noticeably with the increased load on the server. Each member still received each update within 1 second of it being sent.

Technical Observations

In the case of two people altering (deleting, updating etc) any GridElement, the person who performed and sent their message last will have their update appear at all clients. This is due to the fact that no one user can lock any or all of the elements for updating. Since everyone has the same access to all elements, those that perform an operation last will "own" that element.

A Java Thread is created for each client on the server. In this way, all clients run concurrently. In almost all testing scenarios, all updates took less than a second to transfer from client to client.

Pedagogical Test Details

Test timetable and participant details

The testing of Networked DrumSteps took place on the 13th March 2003 in The Lucan Educate Together School in Dublin. The test consisted of two sessions with Networked DrumSteps and a subsequent short interview with 2 children collaborating on a piece of music using Networked DrumSteps. Both participants had used standalone DrumSteps before and so were familiar with the interface and functionality. In this way, the outcome of using Networked DrumSteps happened outside of complications such as familiarity with the interface, buttons etc. Neither learner had any significant musical background.

Task

In the first session, the learners were asked to create a piece of music together. While sounding ambiguous, this allowed them to familiarise themselves with the operation of DrumSteps over the network in preparation for a more directed task in the second session.

In the second session, they were asked to make something different from the first session but with the added constraint that what they made had to be agreed upon first. They had to agree on the number of beats, how many sections etc in advance, and then go and build it. It effectively meant a more focused use of the communication facilities to structure the creation of a piece.

Methods of Evaluation

Information for the following evaluation was collated from different sources. These sources included videotapes of the sessions and subsequent interviews, the DrumSteps files stored at the server and conversations with the learners themselves. Also utilised was videotape evidence of past sessions of the same learners using standalone DrumSteps.

Pedagogical Observations

Section two outlined some components necessary for collaborative learning. These included:

<u>Giving and receiving help and assistance</u>

The chat facility was used extensively for the purpose of giving assistance. At one point, Conor had used a Pyramid in a piece he was building. Daniel, unaware of what its function was, asked:

D: what do pyramids do?

Which led to the following response from Conor:

C: they make the balls go left or right

This demonstrates that the learners were able to help each other with the interface and the functions of Networked DrumSteps in a constructive fashion. This is also evident during the creation of a piece (Figures 7, 8, 9).

<u>Explaining, teaching ones own knowledge to others</u>

In the interview, both learners affirmed the use of the others piece to formulate their own ideas:

Interviewer: Did either of you find that (pointing to Conor) you're working and you make something, and (pointing to Daniel) you see what he made, and that gave you an idea for something…

D: yeah

I: …and then you (pointing to Daniel) went and made something...did that happen?

C: yeah

This type of interaction was implicit in the building of pieces together, and so can only be observed by close examination of the building process. The chat message above also show how Conor was able to help and inform Daniel on the use of Pyramids.

<u>Challenging each other's reasoning</u>

After agreeing on a scheme:

D: Will we do 5 beats now

C: yes

Conor proceeded to build a pattern looking like Figure 6.

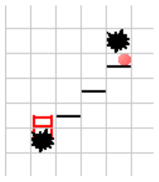

Figure 6: 4 beats

Conor seeks affirmation, but Daniel informs him of his mistake:

C: Does this look good

D: it looks good but its only 3 beats

and in doing so makes a mistake himself, as the section is actually 4 beats long Interestingly, this did not lead either Daniel or Conor to alter the flawed section. Conor proceeded to start a brand new section, while Daniel continued to add to the section shown.

<u>Clear, positive interdependence among students</u>

Frequently during the sessions, the learners asked each other questions relating to the piece being composed. In particular they sought opinions of their own work.

Conor: Does this look good?

and

Daniel: Does that sound good

C: yes

C: it looks good

Evidently Clearly each learner depended on the other for evaluation and affirmation of what they had done. Tasks were also set out during the chat conversation:

D: you make one part and ill make another

and

C: lets do 3 parts

After which they proceeded to engage in the process shown in Figure 7, Figure 8 and Figure 9. Figure 4 is mostly Conors creation. Figure 5 shows Conors piece combined with something Daniel built (on the right hand side). Finally the entire piece is shown in Figure 9. After deciding beforehand what each would do, they set about the task, each building separate parts, then eventually linking them up.

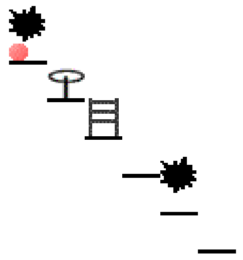
Figure 7: Stage 1

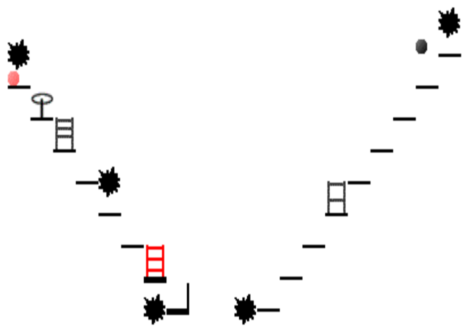
Figure 8: Stage 2

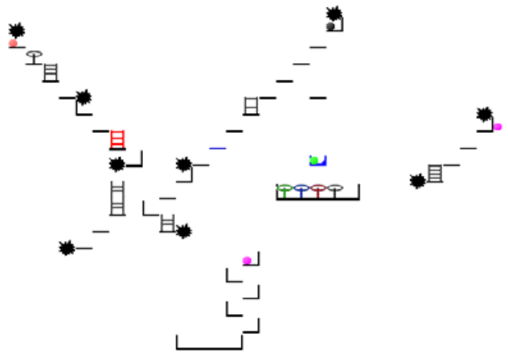
Figure 9: Stage 3

Figure 10 also demonstrates this. In this figure, we can see that both Conor and Daniel worked on the top left piece, whereas Daniel is responsible for the middle right piece, and after the following message:

D: will we do a more complicated one now

C: yes

both start to work together again on the bottom piece. The contributions of each are visible in the figure. Daniels contribution are those elements surrounded by a blue outline, Conor has none.

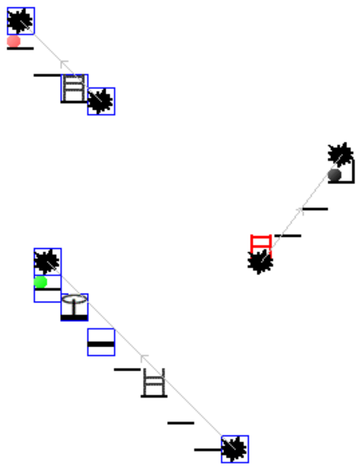
Figure 10: Working on different sections and together

Regular group self-evaluation

The constant messaging to one another during the creation of a piece demonstrated how the chat facility enabled the learners to monitor each other's progress and to offer criticisms and/or help regarding the piece.

C: Do you like this

D: yes

and

C: Does this look good?

D: It looks good but its only 3 beats

and

C: like mine

D: yes

and

C: like that

D: yes it sounds good

These type of message occurred frequently during each session. By constantly checking with the other learner, each one was able to evaluate their work, possibly with an aim to altering or extending it.

Interpersonal behaviours that promote each members learning and success

In a Networked DrumSteps session it is vital that the members work together to produce a piece of music that represents a combination of the talents of those involved. In both sessions, the learners constantly supported each other by offering help or advice throughout. Both fully realised that the aim of the exercises were to jointly create a piece of music, and used the chat facility extensively to accomplish this.

C: do you like this

D: yeah that's cool

and

D: will we do 5 beats now

C: yes

(after which Conor builds a piece)

D: that's not 5 beats…it looks good but its only 3 beats

At this point, it is worth noting that in the first session Conor did not listen to the piece once in the first session. He occupied himself with counting beats on the screen and building more and more elements. It is worth pointing out that evidence from tests with Conor using standalone DrumSteps show that he was able to clap almost all the rhythms he made, without having to listen to them. This may provide some information as to why he did not listen as much as Daniel but preferred to build and count the rhythms internally. Some of his chat messages indicated a leaning towards the visual aspect for him:

D: Does that sound good

C: yes

C: it looks good

and

C: do you like mine

D: yeah have you listened to it all

C: no

after which Conor listened to the piece for the first time. Daniel constantly listened to the piece in progress. He was always the one to suggest a more complicated piece, or to move on to a new piece.

<u>Individual accountability and responsibility</u>

At the start of the second session, the learners were asked to "make a piece of music that was different from the last piece…and at the start agree how many sections are going to be in the piece…decide which bit each of you makes…". This guidance led to the session beginning with the following:

D: I think we should make 8 beats in the first part

C: ok

D: you make one part and ill make another

C: lets do 3 parts

and the session progresses from there. Indeed the piece progressed from this idea and eventually they linked up with each other (Figure 4, 5, 6).

It is worth noting that of the two learners, Daniel was the faster typist, and was also able to build pieces more quickly than Conor. This led to him slightly dominating the session with ideas and suggestions.

D: I think we should make 8 beats in the first part.

and

D: Will we do 5 beats now

Indeed the majority of pieces belonged to Daniel by the end of the session. Daniel also experimented with triplets and Ball volumes, whereas Conor did not. He also used the "Step Forward" and "Step Back" feature to step through the piece one step at a time. When asked the following:

I: Were you mostly collaborating with each other on most of the stuff, or were you mostly just doing your own thing and the other fella [sic] had to just put up with it. Which was it? What would you say Conor?

C: the first one

I: You think mostly collaborating, and you think (to Daniel)...mostly doing your own thing.

This indicates that Although Daniel dominated the session, he saw his actions as "doing his own thing", whereas Conor saw his as collaborating. Indeed one of the main observations made was that Conor continuously looked for affirmation from Daniel, once again suggesting that Daniel led the session:

C: Do you like this

D: Yeah thats cool

and

C: Like mine

D: yes

<u>Frequent use of appropriate interpersonal and small group social skills</u>

Many of the snippets of conversation given above display the frequent use of group social skills necessary for the joint creation of a piece. These and other pieces of dialogue demonstrate how the creation moved in a stepwise fashion. Foe example, Conor would build a set of steps, and then ask Daniel how they looked or sounded. Depending on Daniels response, Conor was then free to alter and improve his piece, which he did frequently

Pedagogical Evaluation

Through interviews and analysis of their work, the session demonstrated that:

- Both learners enjoyed working together more than working alone. In particular Conor attributed the advantage of collaborative work to the fact that he could talk to someone else whilst creating a piece.
- The learners did not express too much concern at having to type messages into the chat box. Although slow they managed to express their ideas and suggestions adequately. It is worth mentioning that both learners had computers at home, which they regularly used.
- Neither seemed to mind sharing the workspace with the other. Both opposed the suggestion of each user having their own piece of the screen in which only they could work and everyone else could see but not alter. They enjoyed the fact that they could jointly create a bigger, "better" piece together because they shared the same space.
- They agreed that the piece they made together was "different" from something they would have made on their own. Using Networked DrumSteps, they made more complicated pieces, both together and alone. Making more complicated pieces proved to be "more fun".
- Conor pointed out that he made things

differently because Daniel was there. He changed the way he created because Daniel was there. His pieces were a little less organised, because he was trying to make something a little more fun.

- Both admitted to playing off each other. One would make something, the other would takes ideas from that and use it in his own etc.

Results

Networked DrumSteps provided for positive interdependence among students, as well as scaffolding behaviour that promotes the learning and success of each member. Learners enjoyed the ability to seek assistance whilst creating a piece. They also enjoyed providing help with a piece, as well as being able to exchange resources and info. Overall, working together provided more advantages than working alone, although learners have noted that it depends on the person they were creating with. They enjoyed sharing the same space, and being able to see the piece being created in real time. It is also worth noting that they saw Networked DrumSteps as being "more fun" than working alone. Networked DrumSteps demonstrated that using the software caused a change in both product and process. By working together in a virtual environment, they changed what they made (compared to when they were working alone) and also how they made it.

Conclusion and Further Work

Summary of Findings

Results derived from testing Networked DrumSteps in an educational environment are promising. The learners enjoyed working together more than working alone, citing the main benefit as having someone else to confer with whilst creating a piece of music, as well as being able to create "better" pieces of music. Learners agreed that the piece they made together was "different" from something they would have made on their own. It is also worth noting that the learners claimed Networked DrumSteps to be more "fun" than working alone.

Conclusion

To the extent that it was possible, this thesis proved that it is possible to design and implement a collaborative environment for learning music in the form of a microworld. Chapter three presented a careful consideration of the requirements necessary for the design of such a microworld, while Chapter four described the implementation of the system.

Microworlds have proven themselves to be effective learning environments in many disciplines. But their design and implementation needs to be well thought out and structured in order for them to be effective and useful. This is especially true in collaborative microworlds, where the demands of distribution meet the constraints of the medium head on.

As well as presenting a new method of learning notation, this thesis advanced the idea of sharing musical ideas through music and dialogue, instead of through discussion about music alone. This thesis goes some of the way towards proving that interaction through music is more effective than interaction about music.

This thesis also posits that serious consideration should be given to the idea of adding a networking module to all standalone music education applications. Collaboration, as this thesis demonstrates, is important in most areas of education, and its realisation in music education can have a large impact on learning. Not all music education applications are suitable for the addition of such a module, but the uses of software change over time. By providing a method to implement a networking element at an early stage, it may prove useful later on and could save on development time in the long term.

Of course, there is still much that can be done, both with Networked DrumSteps and in the area of music technology and education in general. There are constant advances in both information technology and music education, and with the proper research and guidance, future technologies will hopefully be able to embody the ideals of both strands in one.

Future Work

DrumSteps was designed as a Java application from the outset, although a scaled down applet version exists. The reason that the applet version wasn't expanded upon was that users could not save and re-open their work locally in the applet version. This was deemed too important a feature to leave out. Ideally, a networked applet version would serve many purposes, but would mean that due to security limitations, users would have to save their work to the server, and not to their local desktops. This introduces a whole new set of considerations at the server.

Due to time constraints, the sending of entire DrumSteps files to different users was not implemented. This feature would be useful if one user in a session wished to open a file locally. In order to maintain consistency, all other users would need to obtain a copy of this file. In this way, users could proceed to open files saved on the server, possibly by a tutor as part of a planned lesson.

Networked DrumSteps does not allow for the allocation of roles in a session. Everyone participating in a session has exactly the same rights. An addition to Networked DrumSteps, and which is seen in other collaborative systems, is to provide for an administrator role. An administrator could monitor a session, provide expert help, talk privately with individual learners, or have the power to terminate a users session if they are misbehaving.

Another addition that might provide useful is the provision of a full set of diagnostic and measuring tools at the server. Much data about a session could be collected and analysed later. The server could also benefit from it's own interface to manage these and other functions. This might be incorporated into a token system.

Bibliography

Abdel-Wahab, H., B. Kvande, et al. (1996). Using Java for Multimedia Collaborative Appli-cations. 3rd

International Workshop on Protocols for Multimedia Systems (PROMS'96), Madrid.

Abdulmotaleb, E. S., S. Shirmohammadi, et al. (2000). JASMINE: Java Application Sharing in Multiuser INteractive Environments. IDMS '2000, Enschede, Netherlands, Springer.

Bamberger, J. (1991). The mind behind the musical ear. Cambridge, MA, Harvard University Press.

Bamberger, J. (2000). Developing Musical Intuitions - A Project-Based Introduction to Making and Understanding Music. Boston, Developing Musical Intuitions - A Project-Based Introduction to Making and Understanding Music.

Begole, J., C. Struble, et al. (1997). Transparent Sharing of Java Applets: A Replicated Approach. 1997 Symposium on User Interface Software and Technology, New York, ACM Press.

Bloom, B. S. (1956). Taxonomy of educational objectives: The classification of educational goals: Handbook I, cognitive domain. New York; Toronto, Longmans, Green.

Chabert, A., E. Grossman, et al. (1998). "Java Object Sharing in Habanero." Communications of the ACM **41**(6): 69-76.

Chew, G. (1980). Notation (1500 to the present). The New Grove Dictionary of Music & Musicians. S. Sadie. London, Macmillan. **13:** 373-420.

Farbood, M. (2001). Hyperscore: A new approach to inter-active, computer-generated music. MIT Media Laboratory. Boston.

Ferguson, D. (2001). "Technology in a constructivist classroom." Information Technology in Childhood Education Annual: 45-55.

Fuks, H., M. A. Gerosa, et al. (2002). Using a Groupware Technology to Implement Cooperative Learning via the Internet - A case study,. Thirty-fifth Annual HAWAII International Conference on System Sciences on the Big Island of Hawaii, Hawaii.

Jennings, K. (2001). DrumSteps - A Constructionist Approach to Music Learning. 9th Technological Directions in Music Learning Conference, San Antonio, Texas.

Johnson, D. W. and R. T. Johnson (1997). "Cooperation and the Use of Technology." Proceedings of the Annual National Educational Computing Conference.

Kedong, L. and Z. Jianhua (2001). Analysis Method for Compositive Factors of Collaborative Learning in Classroom-Based and Web-Based Environment. International Conference on Computers in Education, Incheon, Korea.

Kinshuk (2002). Effects of media on students culture of learning. IEEE International Conference on Advanced Learning Technologies, Kazan, Russia.

Kuhmunch (1998). Java Teachware - The Java Remote Control Tool and its Applications. ED-MEDIA '98.

Papert, S. (1980). Mindstorms; Children, Computers and Powerful Ideas. Cambridge, MA, MIT Press.

Pea, R. D. (1993). "Distributed Multimedia Learning Environments: The Collaborative Visualization Project." Communications of the ACM **36**: 60-63.

Pellone, G. (1992). "Developing Instructional Software." Australian Journal of Educational Technology **8**(1): 65-8.

Reinhard, K., S. Kiesow, et al. (2000). "NetChat: Communication and Collaboration via WWW." Educational Technology & Society **3**(3).

Scardamalia, M. and C. Bereiter (1993). "Technologies for knowledge-building discourse." Communications of the Association for Computing Machinery **36**(5): 37-41.

Serafine, M. L. (1988). Music as Cognition: The Development of Thought in Sound. New York, Columbia University Press.

Thibault, S. (2001). MICK: A Design Environment for Musical Instruments. Department of Electrical Engineering and Computer Science. Boston, Massachusetts Institute of Technology.

Webster, P. (2002). Computer Based Technology and Music Teaching and Learning. Music Educators National Conference.